



UNIVERSIDAD DEL BÍO-BÍO

**FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE SISTEMAS DE INFORMACION**

TESIS

Presentada para optar por el Título de
INGENIERO CIVIL EN INFORMÁTICA

Por:

Cristian Fernando Zenteno Díaz

**Propuesta de Arquitectura para el desarrollo de
Aplicaciones Móviles en la Dirección De Informática de la
Universidad del Bio-Bío**

Profesor Guía: Christian Vidal Castro

28 de mayo de 2018

Concepción - Chile

Resumen

Este proyecto se presenta para dar conformidad a los requisitos exigidos por la Universidad de Bío-Bío en el proceso de titulación para a la carrera de Ingeniera Civil Informática.

El proyecto titulado “Propuesta de Arquitectura de Sistemas para el desarrollo de Aplicaciones Móviles en la Dirección De Informática de la Universidad del Bío-Bío” es una investigación de nuevas tecnologías de desarrollo de software orientadas a plataformas móviles.

Dentro de la arquitectura que utiliza la universidad, actualmente el desarrollo se realiza de forma nativa lo que significa un esfuerzo mayor a la hora de elaborar una aplicación, ya que el proceso se repite según la cantidad de plataformas en las que se quiera desplegar el software, lo que conlleva un mayor costo a la hora de implementar nuevas soluciones móviles. Es por esta razón que se hace necesario investigar nuevas formas de estructurar y desarrollar aplicaciones móviles.

Adicionalmente, se realiza una investigación sobre una nueva tecnología de comunicación entre capas, que supone una mejor respuesta a la hora de acceder a los datos desde las aplicaciones.

AGRADECIMIENTOS

En primer lugar, agradezco a mi inmensa familia, y sobre todo a los que conforman mi núcleo, mi madre Miryan, mi padre Ricardo y mi hermano Rodolfo, por apoyarme en todo momento a lo largo de estos años de estudio.

Aprovecho también esta página para agradecer a mis amigos de la vida, a los que ya considero hermanos, Miguel P., Alejandro C. y John N. que estuvieron conmigo en los buenos y sobre todo en los malos momentos que pase durante este extenso trayecto.

Agradezco a mis compañeros de universidad, Pablo R., Fabián S., Fabián A., Rodrigo A., Alex V., Cesar O., Richard F. y Daniela N., que han sido un apoyo fundamental a la hora de enfrentar los desafíos que nuestra carrera nos impuso, siempre dispuestos a escuchar y ayudar, convirtiéndose en grandes amigos que la vida puso en mi camino.

A todos los profesores que con su apoyo y formación aportaron conocimiento fundamental para mi desarrollo como persona y profesional.

A todas las personas que han sido partícipes de este largo proceso que culmina con la titulación, de corazón.

Muchas Gracias

Índice General

1	<u>INTRODUCCIÓN.....</u>	<u>11</u>
2	<u>DEFINICION DE LA EMPRESA O INSTITUCIÓN.....</u>	<u>13</u>
2.1	DESCRIPCIÓN DE LA EMPRESA.....	13
2.2	FORMACIÓN.....	14
2.3	ORGANIGRAMA DE LA INSTITUCIÓN	15
2.4	DESCRIPCIÓN DEL ÁREA DE ESTUDIO	17
2.5	DESCRIPCIÓN DE LA PROBLEMÁTICA	17
3	<u>DEFINICIÓN PROYECTO</u>	<u>18</u>
3.1	OBJETIVOS DEL PROYECTO.....	18
3.2	AMBIENTE DE INGENIERÍA DE SOFTWARE.....	18
3.3	DEFINICIONES, SIGLAS Y ABREVIACIONES	23
4	<u>MARCO TEÓRICO.....</u>	<u>25</u>
4.1	QUE ES LA ARQUITECTURA DE SOFTWARE	25
4.2	PRINCIPIOS CLAVE DE LA ARQUITECTURA.....	26
5	<u>ARQUITECTURA ACTUAL UTILIZADA EN LA UNIVERSIDAD DEL BÍOBÍO</u>	<u>39</u>
5.1	CAPA DE APLICACIÓN.....	40
5.2	CAPA DE NEGOCIO.....	41
5.3	CAPA DE DATOS.....	44
6	<u>SELECCIÓN DE ESTILO DE ARQUITECTURA.....</u>	<u>45</u>
7	<u>CAPA DE APLICACIÓN, UN ANÁLISIS DE ALTERNATIVAS EN PROFUNDIDAD.....</u>	<u>47</u>
7.1	CAPA DE APLICACIÓN.....	47
8	<u>IMPLEMENTACION Y PRUEBA DE TECNOLOGIAS EN LA CAPA DE APLICACION.....</u>	<u>63</u>
8.1	CARACTERÍSTICAS DE DESPLIEGUE	64
8.2	MODELAMIENTO DEL PROBLEMA	64
8.3	APLICACIÓN NATIVA ANDROID.....	83
8.4	APLICACIÓN EN XAMARIN .FORMS	97
8.5	APLICACIÓN EN IONIC.....	112
9	<u>ANÁLISIS EMPÍRICO DE LOS RESULTADOS DE LAS PUEBAS EN LA CAPA DE APLICACIÓN.....</u>	<u>128</u>
9.1	PESO TOTAL SIN USO.....	128
9.2	COMPARACIÓN DE DATOS GUARDADOS EN LAS CONSULTAS	129
9.3	PESO FINAL APLICACIONES.....	130
9.4	ANÁLISIS DE TIEMPOS	131
9.5	DATOS CONSUMIDOS.....	137
9.6	CONCLUSIÓN DE ANÁLISIS CAPA DE APLICACIÓN	142
10	<u>FACTIBILIDAD DE LA PROPUESTA.....</u>	<u>145</u>
10.1	FACTIBILIDAD TÉCNICA.....	145
10.2	FACTIBILIDAD OPERATIVA.....	148

10.3	FACTIBILIDAD ECONÓMICA.....	148
11	<u>ESTRUCTURA DE LA PROPUESTA DE ARQUITECTURA.....</u>	154
12	<u>CONCLUSIONES</u>	156
13	<u>BIBLIOGRAFÍA.....</u>	158
14	<u>ANEXO N°1: GUIA PARA EL DESARROLLO DE APLICACIONES.....</u>	159
14.1	ELECCIÓN DE TECNOLOGÍA DE DESARROLLO.....	160
14.2	DESARROLLO NATIVO.....	160
14.3	DESARROLLO MULTIPLATAFORMA XAMARIN.....	160
14.4	DESARROLLO MULTIPLATAFORMA IONIC FRAMEWORK.....	160
14.5	CONFIGURACIÓN PARA DESARROLLO NATIVO ANDORID.....	161
14.6	CONFIGURACIÓN PARA DESARROLLO MULTIPLATAFORMA CON XAMARIN	165
14.7	CONFIGURACIÓN PARA DESARROLLO MULTIPLATAFORMA CON IONIC.....	168
14.8	CONCLUSIONES.....	170
15	<u>ANEXO N°2: ESTUDIO CAPA DE NEGOCIO.....</u>	171
15.1	GRAPHQL.....	171
15.2	IMPLEMENTACIÓN Y PRUEBA DE TECNOLOGÍAS DE LA CAPA DE NEGOCIO.....	174
16	<u>ANEXO N°2: DOCUMENTACIÓN WIKI.....</u>	179
17	<u>ANEXO N°3: ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE PARA EL DESARROLLO DE MIUBB.....</u>	180
17.1	ALCANCES.....	180
17.2	OBJETIVO DEL SOFTWARE.....	180
17.3	ATRIBUTOS DEL PRODUCTO.....	180
17.4	INTERFACES DE COMUNICACIÓN	181
17.5	REQUERIMIENTOS ESPECÍFICOS	182

Índice Tablas

TABLA N° 1: CATEGORÍAS DE ESTILOS DE ARQUITECTURA	26
TABLA N° 2: APLICACIONES NATIVAS	40
TABLA N° 3: CARACTERÍSTICAS DEL SMARTPHONE DONDE SE DESPLEGARÁN LAS SOLUCIONES	64
TABLA N° 4: ESPECIFICACIÓN CASO DE USO LOGIN	65
TABLA N° 5: ESPECIFICACIÓN CASO NOTAS	66
TABLA N° 6: ESPECIFICACIÓN CASO HORARIO	67
TABLA N° 7: ESPECIFICACIÓN CASO DE USO INSCRIPCIÓN_1	68
TABLA N° 8: ESPECIFICACIÓN CASO DE USO INSCRIPCIÓN_2	69
TABLA N° 9: ESPECIFICACIÓN CASO DE USO CONSULTA PROFESOR_1	70
TABLA N° 10: ESPECIFICACIÓN CASO DE USO CONSULTA PROFESOR_2	71
TABLA N° 11: ESPECIFICACIÓN CASO DE USO LISTADO_1	72
TABLA N° 12: ESPECIFICACIÓN CASO DE USO CONSULTA LISTADO_2	73
TABLA N° 13: ESPECIFICACIÓN DE MODULO	78
TABLA N° 14: ESPECIFICACIÓN DE MODULO	78
TABLA N° 15: ESPECIFICACIÓN DE MODULO	79
TABLA N° 16: ESPECIFICACIÓN DE MODULO	79
TABLA N° 17: ESPECIFICACIÓN DE MODULO	80
TABLA N° 18: ESPECIFICACIÓN DE MODULO	80
TABLA N° 19: ESPECIFICACIÓN DE MODULO	81
TABLA N° 20: ESPECIFICACIÓN DE MODULO	82
TABLA N° 21: ESPECIFICACIÓN DE MODULO	82
TABLA N° 22: DATOS APLICACIÓN NATIVA RECIÉN INSTALADA	84
TABLA N° 23: PRUEBAS TIEMPOS DE APERTURA	84
TABLA N° 24 PRUEBAS LOGIN	85
TABLA N° 25: TIEMPOS DE RESPUESTA LOGIN	85
TABLA N° 26: PRUEBAS CONSULTA HORARIOS	86
TABLA N° 27: TIEMPOS DE RESPUESTA HORARIOS	87
TABLA N° 28: PRUEBAS CONSULTA NOTAS	88
TABLA N° 29: TIEMPO RESPUESTAS PRUEBAS NOTAS	88
TABLA N° 30: PRUEBAS CONSULTA INSCRIPCIÓN_1	89
TABLA N° 31: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN_1	89
TABLA N° 32: PRUEBAS CONSULTA INSCRIPCIÓN_2	90
TABLA N° 33: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN_2	91
TABLA N° 34: PRUEBAS CONSULTA PROFESOR_1	92
TABLA N° 35: TIEMPO RESPUESTAS PRUEBAS PROFESOR_1	92
TABLA N° 36: PRUEBAS CONSULTA PROFESOR_2	93
TABLA N° 37: TIEMPO RESPUESTAS PRUEBAS PROFESOR_2	93
TABLA N° 38: PRUEBAS CONSULTA LISTADO_1	94
TABLA N° 39: TIEMPO RESPUESTAS PRUEBAS LISTADO_1	95
TABLA N° 40: PRUEBAS CONSULTA LISTADO_1	96
TABLA N° 41 : TIEMPO RESPUESTAS PRUEBAS LISTADO_1	96
TABLA N° 42: DATOS APLICACIÓN XAMARIN RECIÉN INSTALADA	98
TABLA N° 43: PRUEBAS TIEMPOS DE APERTURA	98
TABLA N° 44 PRUEBAS LOGIN	99
TABLA N° 45: TIEMPOS DE RESPUESTA LOGIN	100
TABLA N° 46: PRUEBAS CONSULTA HORARIOS	101
TABLA N° 47: TIEMPOS DE RESPUESTA HORARIOS	101
TABLA N° 48: PRUEBAS CONSULTA NOTAS	102

TABLA N° 49: TIEMPO RESPUESTAS PRUEBAS NOTAS	103
TABLA N° 50: PRUEBAS CONSULTA INSCRIPCIÓN _1	104
TABLA N° 51: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN _1	104
TABLA N° 52: PRUEBAS CONSULTA INSCRIPCIÓN _2	105
TABLA N° 53: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN _2	106
TABLA N° 54: PRUEBAS CONSULTA PROFESOR _1	107
TABLA N° 55: TIEMPO RESPUESTAS PRUEBAS PROFESOR _1	107
TABLA N° 56: PRUEBAS CONSULTA PROFESOR _2	108
TABLA N° 57: TIEMPO RESPUESTAS PRUEBAS PROFESOR _2	108
TABLA N° 58: PRUEBAS CONSULTA LISTADO _1	110
TABLA N° 59: TIEMPO RESPUESTAS PRUEBAS LISTADO _1	110
TABLA N° 60: PRUEBAS CONSULTA LISTADO _2	111
TABLA N° 61 : TIEMPO RESPUESTAS PRUEBAS LISTADO _2	111
TABLA N° 62: DATOS APLICACIÓN IONIC RECÍEN INSTALADA	113
TABLA N° 63: PRUEBA TIEMPOS DE APERTURA	113
TABLA N° 64 PRUEBAS LOGIN	114
TABLA N° 65: TIEMPOS DE RESPUESTA LOGIN	115
TABLA N° 66: PRUEBAS CONSULTA HORARIOS	116
TABLA N° 67: TIEMPOS DE RESPUESTA HORARIOS	116
TABLA N° 68: PRUEBAS CONSULTA NOTAS	118
TABLA N° 69: TIEMPO RESPUESTAS PRUEBAS NOTAS	118
TABLA N° 70: PRUEBAS CONSULTA INSCRIPCIÓN _1	119
TABLA N° 71: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN _1	119
TABLA N° 72: PRUEBAS CONSULTA INSCRIPCIÓN _2	121
TABLA N° 73: TIEMPO RESPUESTAS PRUEBAS INSCRIPCIÓN _2	121
TABLA N° 74: PRUEBAS CONSULTA PROFESOR _1	122
TABLA N° 75: TIEMPO RESPUESTAS PRUEBAS PROFESOR _1	122
TABLA N° 76: PRUEBAS CONSULTA PROFESOR _2	124
TABLA N° 77: TIEMPO RESPUESTAS PRUEBAS PROFESOR _2	124
TABLA N° 78: PRUEBAS CONSULTA LISTADO _1	125
TABLA N° 79: TIEMPO RESPUESTAS PRUEBAS LISTADO _1	125
TABLA N° 80: PRUEBAS CONSULTA LISTADO _2	127
TABLA N° 81 : TIEMPO RESPUESTAS PRUEBAS LISTADO _2	127
TABLA N° 82: COMPARACIÓN DE PUNTOS	142
TABLA N° 83: COMPARACIÓN TIEMPO Y DATOS CONSUMIDOS	143
TABLA N° 84: COSTO TOTAL DE IMPLEMENTACIÓN	152
TABLA N° 85: ESPECIFICACIÓN DE MÓDULO	197
TABLA N° 86: ESPECIFICACIÓN DE MÓDULO	197
TABLA N° 87: ESPECIFICACIÓN DE MÓDULO	198
TABLA N° 88: ESPECIFICACIÓN DE MÓDULOS	198
TABLA N° 89: ESPECIFICACIÓN DE MÓDULO	198
TABLA N° 90: ESPECIFICACIÓN DE MÓDULO	199
TABLA N° 91: ESPECIFICACIÓN DE MÓDULO	199
TABLA N° 92: ESPECIFICACIÓN DE MÓDULO	199
TABLA N° 93: ESPECIFICACIÓN DE MÓDULO	200
TABLA N° 94: ESPECIFICACIÓN DE MÓDULO	200
TABLA N° 95: ESPECIFICACIÓN DE MÓDULO	200
TABLA N° 96: ESPECIFICACIÓN DE MÓDULO	201

Índice Figuras

FIGURA N° 1: ORGANIGRAMA DE LA UNIVERSIDAD DEL BÍO-BÍO.....	15
FIGURA N° 2: ARQUITECTURA DE 3 CAPAS.....	39
FIGURA N° 3: API REST.....	42
FIGURA N° 4: TECNOLOGÍAS DE DESARROLLO A INVESTIGAR.....	48
FIGURA N° 5: COMPILACIÓN XAMARIN.IOS.....	51
FIGURA N° 6: COMPILACIÓN XAMARIN.ANDROID.....	52
FIGURA N° 7: ARQUITECTURA XAMARIN.....	53
FIGURA N° 8: ARQUITECTURA XAMARIN.FORMS.....	54
FIGURA N° 9: PROYECTO XAMARIN.FORMS.....	55
FIGURA N° 10: XAMARIN.FORMS DESPLEGADA EN DIFERENTES SISTEMAS.....	56
FIGURA N° 11: BASES DEL FUNCIONAMIENTO DE IONIC.....	59
FIGURA N° 12: ARQUITECTURA DE IONIC.....	60
FIGURA N° 13: CASOS DE USO.....	64
FIGURA N° 14: DISEÑO DE ARQUITECTURA.....	74
FIGURA N° 15: WIREFRAME.....	75
FIGURA N° 16: WIREFRAME.....	76
FIGURA N° 17: WIREFRAME.....	76
FIGURA N° 18: WIREFRAME.....	76
FIGURA N° 19: WIREFRAME.....	77
FIGURA N° 20: WIREFRAME.....	77
FIGURA N° 21: WIREFRAME.....	77
FIGURA N° 22: WIREFRAME.....	77
FIGURA N° 23: WIREFRAME.....	78
FIGURA N° 24: WIREFRAME.....	78
FIGURA N° 25: APLICACIÓN NATIVA ANDROID SIN USO.....	83
FIGURA N° 26: LOGIN MAYOR ALMACENAMIENTO.....	85
FIGURA N° 27: LOGIN MENOR ALMACENAMIENTO.....	85
FIGURA N° 28: LOGIN MENOR CONSUMO.....	85
FIGURA N° 29 LOGIN MENOR CONSUMO.....	85
FIGURA N° 30: HORARIOS MENOR ALMACENAMIENTO.....	86
FIGURA N° 31: HORARIOS MAYOR ALMACENAMIENTO.....	86
FIGURA N° 32 HORARIOS MAYOR CONSUMO.....	86
FIGURA N° 33: HORARIOS MENOR CONSUMO.....	86
FIGURA N° 34: NOTAS MAYOR ALMACENAMIENTO.....	87
FIGURA N° 35: NOTAS MENOR ALMACENAMIENTO.....	87
FIGURA N° 36: NOTAS MAYOR CONSUMO.....	87
FIGURA N° 37: NOTAS MENOR CONSUMO.....	87
FIGURA N° 38: INSCRIPCION_1 MAYOR PESO.....	88
FIGURA N° 39: INSCRIPCION_1 MENOR PESO.....	88
FIGURA N° 40: INSCRIPCION_1 MAYOR CONSUMO.....	89
FIGURA N° 41: INSCRIPCION_1 MENOR CONSUMO.....	89
FIGURA N° 42: INSCRIPCION_2 MAYOR PESO.....	90
FIGURA N° 43: INSCRIPCION_2 MENOR PESO.....	90
FIGURA N° 44: INSCRIPCION_2 MENOR CONSUMO.....	90
FIGURA N° 45: INSCRIPCION_2 MAYOR CONSUMO.....	90
FIGURA N° 46: PROFESOR_1 MENOR PESO.....	91
FIGURA N° 47: PROFESOR_1 MAYOR PESO.....	91
FIGURA N° 48 PROFESOR_1 MAYOR CONSUMO.....	91

FIGURA N° 49: PROFESOR_1 MENOR CONSUMO.....	91
FIGURA N° 50: PROFESOR_2 MAYOR PESO.....	92
FIGURA N° 51: PROFESOR_2 MENOR PESO.....	92
FIGURA N° 52: PROFESOR_2 MENOR CONSUMO.....	93
FIGURA N° 53: PROFESOR_2 MAYOR CONSUMO.....	93
FIGURA N° 54: LISTADO_1 MAYOR PESO.....	94
FIGURA N° 55: LISTADO_1 MENOR PESO.....	94
FIGURA N° 56: LISTADO_1 MAYOR CONSUMO.....	94
FIGURA N° 57: LISTADO_1 MENOR.....	94
FIGURA N° 58: LISTADO_2 MENOR PESO.....	95
FIGURA N° 59: LISTADO_2 MENOR PESO.....	95
FIGURA N° 60: LISTADO_2 MENOR.....	96
FIGURA N° 61: LISTADO_2 MENOR.....	96
FIGURA N° 62: APLICACIÓN XAMARIN SIN USO.....	97
FIGURA N° 63: LOGIN MAYOR ALMACENAMIENTO.....	99
FIGURA N° 64: LOGIN MENOR ALMACENAMIENTO.....	99
FIGURA N° 65 LOGIN MENOR CONSUMO.....	99
FIGURA N° 66: LOGIN MENOR CONSUMO.....	99
FIGURA N° 67: HORARIOS MAYOR ALMACENAMIENTO.....	100
FIGURA N° 68: HORARIOS MENOR ALMACENAMIENTO.....	100
FIGURA N° 69 HORARIOS MAYOR CONSUMO.....	101
FIGURA N° 70: HORARIOS MENOR CONSUMO.....	101
FIGURA N° 71: NOTAS MENOR ALMACENAMIENTO.....	102
FIGURA N° 72: NOTAS MAYOR ALMACENAMIENTO.....	102
FIGURA N° 73: NOTAS MAYOR CONSUMO.....	102
FIGURA N° 74: NOTAS MENOR CONSUMO.....	102
FIGURA N° 75: INSCRIPCIÓN_1 PESO.....	103
FIGURA N° 76: INSCRIPCIÓN_1 MAYOR CONSUMO.....	104
FIGURA N° 77: INSCRIPCIÓN_1 MENOR CONSUMO.....	104
FIGURA N° 78: INSCRIPCIÓN_2 MAYOR PESO.....	105
FIGURA N° 79: INSCRIPCIÓN_2 MENOR PESO.....	105
FIGURA N° 80: INSCRIPCIÓN_2 MENOR CONSUMO.....	105
FIGURA N° 81: INSCRIPCIÓN_2 MAYOR CONSUMO.....	105
FIGURA N° 82: PROFESOR_1 MENOR PESO.....	106
FIGURA N° 83: PROFESOR_1 MAYOR PESO.....	106
FIGURA N° 84 PROFESOR_1 MAYOR CONSUMO.....	106
FIGURA N° 85: PROFESOR_1 MENOR CONSUMO.....	106
FIGURA N° 86: PROFESOR_2 MENOR PESO.....	107
FIGURA N° 87: PROFESOR_2 MAYOR PESO.....	107
FIGURA N° 88: PROFESOR_2 MENOR CONSUMO.....	108
FIGURA N° 89: PROFESOR_2 MAYOR CONSUMO.....	108
FIGURA N° 90: LISTADO_1 MAYOR PESO.....	109
FIGURA N° 91: LISTADO_1 MENOR PESO.....	109
FIGURA N° 92: LISTADO_1 MAYOR CONSUMO.....	109
FIGURA N° 93: LISTADO_1 MENOR.....	109
FIGURA N° 94: LISTADO_2 MENOR PESO.....	110
FIGURA N° 95: LISTADO_2 MENOR PESO.....	110
FIGURA N° 96: LISTADO_2 MAYOR.....	111
FIGURA N° 97: LISTADO_2 MENOR.....	111
FIGURA N° 98: APLICACIÓN IONIC SIN USO.....	112
FIGURA N° 99: LOGIN MENOR ALMACENAMIENTO.....	114
FIGURA N° 100: LOGIN MAYOR ALMACENAMIENTO.....	114
FIGURA N° 101 LOGIN MENOR CONSUMO.....	114

FIGURA N° 102: LOGIN MENOR CONSUMO.....	114
FIGURA N° 103: HORARIOS MENOR ALMACENAMIENTO.....	115
FIGURA N° 104: HORARIOS MAYOR ALMACENAMIENTO.....	115
FIGURA N° 105 HORARIOS MAYOR CONSUMO.....	116
FIGURA N° 106: HORARIOS MENOR CONSUMO.....	116
FIGURA N° 107: NOTAS MAYOR ALMACENAMIENTO.....	117
FIGURA N° 108: NOTAS MENOR ALMACENAMIENTO.....	117
FIGURA N° 109: NOTAS MAYOR CONSUMO.....	117
FIGURA N° 110: NOTAS MENOR CONSUMO.....	117
FIGURA N° 111: INSCRIPCION_1 PESO MENOR.....	118
FIGURA N° 112: INSCRIPCION_1 PESO MAYOR.....	118
FIGURA N° 113: INSCRIPCION_1 MAYOR CONSUMO.....	119
FIGURA N° 114: INSCRIPCION_1 MENOR CONSUMO.....	119
FIGURA N° 115: INSCRIPCION_2 MAYOR PESO.....	120
FIGURA N° 116: INSCRIPCION_2 MENOR PESO.....	120
FIGURA N° 117: INSCRIPCION_2 MENOR CONSUMO.....	120
FIGURA N° 118: INSCRIPCION_2 MAYOR CONSUMO.....	120
FIGURA N° 119: PROFESOR_1 MENOR PESO.....	121
FIGURA N° 120: PROFESOR_1 MAYOR PESO.....	121
FIGURA N° 121 PROFESOR_1 MAYOR CONSUMO.....	122
FIGURA N° 122: PROFESOR_1 MENOR CONSUMO.....	122
FIGURA N° 123: PROFESOR_2 MAYOR PESO.....	123
FIGURA N° 124: PROFESOR_2 MENOR PESO.....	123
FIGURA N° 125: PROFESOR_2 MENOR CONSUMO.....	123
FIGURA N° 126: PROFESOR_2 MAYOR CONSUMO.....	123
FIGURA N° 127: LISTADO_1 MAYOR PESO.....	124
FIGURA N° 128: LISTADO_1 MENOR PESO.....	124
FIGURA N° 129: LISTADO_1 MAYOR CONSUMO.....	125
FIGURA N° 130: LISTADO_1 MENOR.....	125
FIGURA N° 131: LISTADO_2 MAYOR PESO.....	126
FIGURA N° 132: LISTADO_2 MENOR PESO.....	126
FIGURA N° 133: LISTADO_2 MENOR.....	126
FIGURA N° 134: LISTADO_2 MENOR.....	126
FIGURA N° 135: COMPARATIVA PESO APLICACIONES.....	128
FIGURA N° 136: COMPARATIVA DE MEMORIA CAHE Y DATOS AMACENADOS.....	129
FIGURA N° 137: PESO FINAL.....	130
FIGURA N° 138: COMPARATIVA TIEMPOS DE APERTURA.....	131
FIGURA N° 139: COMPARATIVA TIEMPOS DE RESPUESTA LOGIN.....	132
FIGURA N° 140: COMPARATIVA TIEMPOS DE RESPUESTA HORARIOS.....	133
FIGURA N° 141: COMPARATIVA TIEMPOS DE RESPUESTA NOTAS.....	133
FIGURA N° 142: COMPARATIVA TIEMPOS DE RESPUESTA INSCRIPCION_1.....	134
FIGURA N° 143: COMPARATIVA TIEMPOS DE RESPUESTA INSCRIPCION_2.....	134
FIGURA N° 144: COMPARATIVA TIEMPOS DE RESPUESTA PROFESOR_1.....	135
FIGURA N° 145: COMPARATIVA TIEMPOS DE RESPUESTA PROFESOR_2.....	135
FIGURA N° 146: COMPARATIVA TIEMPOS DE RESPUESTA LISTA_1.....	136
FIGURA N° 147: COMPARATIVA TIEMPOS DE RESPUESTA LISTA_2.....	136
FIGURA N° 148: COMPARACIÓN CONSUMO DE DATOS LOGIN.....	137
FIGURA N° 149: COMPARACIÓN CONSUMO DE DATOS CONSULTA HORARIOS.....	138
FIGURA N° 150: COMPARACIÓN CONSUMO DE DATOS CONSULTA NOTAS.....	138
FIGURA N° 151: COMPARACIÓN CONSUMO DE DATOS CONSULTA INSCRIPCIÓN_1.....	139
FIGURA N° 152: COMPARACIÓN CONSUMO DE DATOS CONSULTA INSCRIPCIÓN_2.....	139
FIGURA N° 153: COMPARACIÓN CONSUMO DE DATOS CONSULTA PROFESOR_1.....	140
FIGURA N° 154: COMPARACIÓN CONSUMO DE DATOS CONSULTA PROFESOR_2.....	140

FIGURA N° 155: COMPARACIÓN CONSUMO DE DATOS CONSULTA LISTADO_1	141
FIGURA N° 156: COMPARACIÓN CONSUMO DE DATOS CONSULTA LISTADO_2	141
FIGURA N° 157: COSTO IMPLEMENTACIÓN ANDROID STUDIO.....	149
FIGURA N° 158: COSTO APRENDIZAJE XAMARIN.....	150
FIGURA N° 159: COSTO APRENDIZAJE IONIC.....	151
FIGURA N° 160: ARQUITECTURA GRAPHQL.....	171
FIGURA N° 161: MODELO CONSULTA GRAPHQL.....	172
FIGURA N° 162: INICIO SESIÓN.....	174
FIGURA N° 163: MODELADO DE DATOS.....	176
FIGURA N° 164: INICIO SESIÓN GRAPHQL.....	176
FIGURA N° 165: MODELADO DE DATOS.....	178
FIGURA N° 166: ARQUITECTURA UTILIZADA.....	181
FIGURA N° 167: DIAGRAMA CASOS DE USO.....	183
FIGURA N° 168: WIREFRAME INICIO DE SESIÓN	194
FIGURA N° 169: WIREFRAME MENU PRINCIPAL.....	195
FIGURA N° 170: WIREFRAME MENU PRINCIPAL.....	195
FIGURA N° 171: WIREFRAME ASISTENCIA.....	195
FIGURA N° 172: WIREFRAME NOTAS.....	195
FIGURA N° 173: WIREFRAME MALLA CURRICULAR.....	196
FIGURA N° 174: WIREFRAME CUENTAS.....	196
FIGURA N° 175: WIREFRAME SALUD	196
FIGURA N° 176: WIREFRAME ASISTENTE SOCIAL.....	196
FIGURA N° 177: WIREFRAME CANCHAS.....	197

1 INTRODUCCIÓN

El presente informe de Proyecto De Título contiene la documentación correspondiente al desarrollo de una investigación para la Dirección De Informática de la Universidad del Bío-Bío.

En la actualidad, los desarrolladores con los que cuenta la universidad deben lidiar con la sobrecarga de esfuerzo a la hora de desarrollar una aplicación móvil, ya que se debe realizar el proceso de desarrollo según la cantidad de plataformas en las que se quiera desplegar la solución.

Este proyecto de título “Propuesta de Arquitectura para el desarrollo de Aplicaciones Móviles en la Dirección De Informática de la Universidad del Bio-Bío”, pretende dar una alternativa al desarrollo que actualmente se utiliza, que permita a los desarrolladores desplegar una solución para más de una plataforma móvil.

El presente informe de proyecto tiene como finalidad dar a conocer un análisis completo de la arquitectura que actualmente se utiliza en la universidad para la creación de aplicaciones, y mediante una investigación definir una estructura que ayude a la Dirección de Informática llevar un control y orden de las distintas aplicaciones que se desarrollarán en un futuro para la universidad.

En el capítulo dos se presenta la definición de la empresa o institución sobre la cual se desarrollará el proyecto además de entregar una breve descripción de la problemática que se resolverá

En el capítulo tres, se define el proyecto. Primero definiendo el objetivo general y los objetivos específicos del proyecto.

El capítulo cuatro define el marco teórico el cual tiene como objetivo interiorizar al lector sobre las distintas arquitecturas existentes a la hora de desarrollar un software

El capítulo cinco es un estudio de la arquitectura que actualmente utiliza la Universidad Del Bío-Bío, desglosando cada uno de los componentes que pertenecen ésta.

El capítulo seis da una justificación con respecto a la elección de una arquitectura de sistema sobre la cual se realizara un estudio en profundidad.

El capítulo siete, es la investigación de alternativas en los componentes de la Arquitectura Utilizada en la Universidad, entregando detalles conceptuales acerca de las nuevas tecnologías investigadas.

El Capitulo ocho entrega una implementación de cada tecnología investigada en la capa de aplicación, entregando un desarrollo que servirá para comparar las tecnologías anteriormente presentadas.

El capítulo nueve tiene como fin realizar un análisis comparativo de los resultados de las pruebas en la capa de aplicación desarrolladas en el capítulo anterior por medio de gráficos y esquemas.

En el capítulo diez se analiza la factibilidad técnica, operativa y económica de cada una de las soluciones propuestas, con el fin de obtener un nuevo parámetro a comparar entre las tecnologías de desarrollo móvil, que se han presentado y comparado, en los capítulos anteriores.

El capítulo once entrega la estructura de la Arquitectura que se selecciono según el estudio realizado.

En el capítulo doce se describen las conclusiones con respecto al proyecto de investigación.

2 DEFINICION DE LA EMPRESA O INSTITUCIÓN

2.1 Descripción de la empresa

Nombre: Universidad del Bío-Bío.

Dirección: Av. Casilla 5-C - Collao 1202, Concepción, Región del Bío-Bío

Rubros: Gestión, Docencia, Investigación y Vinculación con el medio.

Productos o Servicios que ofrece: Servicio de formación integral de personas profesionales.

Historia

“La Universidad del Bío-Bío es heredera de la más antigua tradición de la educación superior estatal y pública en la Región del Bío-Bío. Sus orígenes se remontan a la creación de la Universidad Técnica del Estado, UTE, el 9 de abril de 1947, bajo la presidencia de Gabriel González Videla.

Dependiente del Ministerio de Educación Pública, la nueva institución fusionó en su interior la Escuela de Ingenieros Industriales y los grados técnicos de la Escuela de Artes y Oficios de Santiago; Escuela de Minas de Antofagasta, Copiapó y La Serena y las Escuelas Industriales de Concepción, Temuco y Valdivia.

La UTE abrió oficialmente sus puertas en 1952, luego que el Senado aprobó su Estatuto Orgánico, contrariando la férrea oposición de la Universidad de Chile, cuyas autoridades consideraban que la nueva casa de estudios superiores debía funcionar bajo su tuición. En Concepción, el plantel jugaría un importante papel para responder a las necesidades y desafíos que plantea la Región como uno de los polos del desarrollo industrial del país, no sólo a través de la docencia de pregrado sino que también mediante la investigación científica y tecnológica, ...¹

¹ Para conocer más sobre la historia de la universidad, ingrese a: <http://www.ubiobio.cl/w/#Historia> o bien www.ubiobio.cl

2.2 Formación

Carreras de Pregrado

Forma profesionales de excelencia en carreras de pregrado.

Postgrado

A través de carreras de postgrado forma profesionales altamente calificados.

Programas Especiales de continuidad de Estudios

Permite al estudiante, bajo una modalidad especial, optar al grado de técnico, licenciado y/o título profesional de pregrado.

Publicaciones

Revistas y publicaciones electrónicas de la Universidad del Bío-Bío.

Investigación y Desarrollo

Contribuye con investigación fundamental, aplicada y de desarrollo.

Transferencia Tecnológica y Servicios

Vincula a la Universidad con el sector productivo.

Incubadora de Empresas

Dedicada a la formación de emprendedores y su promoción.

Centros y Programas

Cultiva el conocimiento.

Extensión

Difunde sistemáticamente la Universidad en su entorno.

Cultura UBB

Contribuye a actividades de cultura.

Ediciones UBB

Difunde el conocimiento generado en la academia.

Formación Continua

Capacita a personas provenientes del mundo del trabajo y profesionales en general de diversos ámbitos.

2.3 Organigrama de la Institución

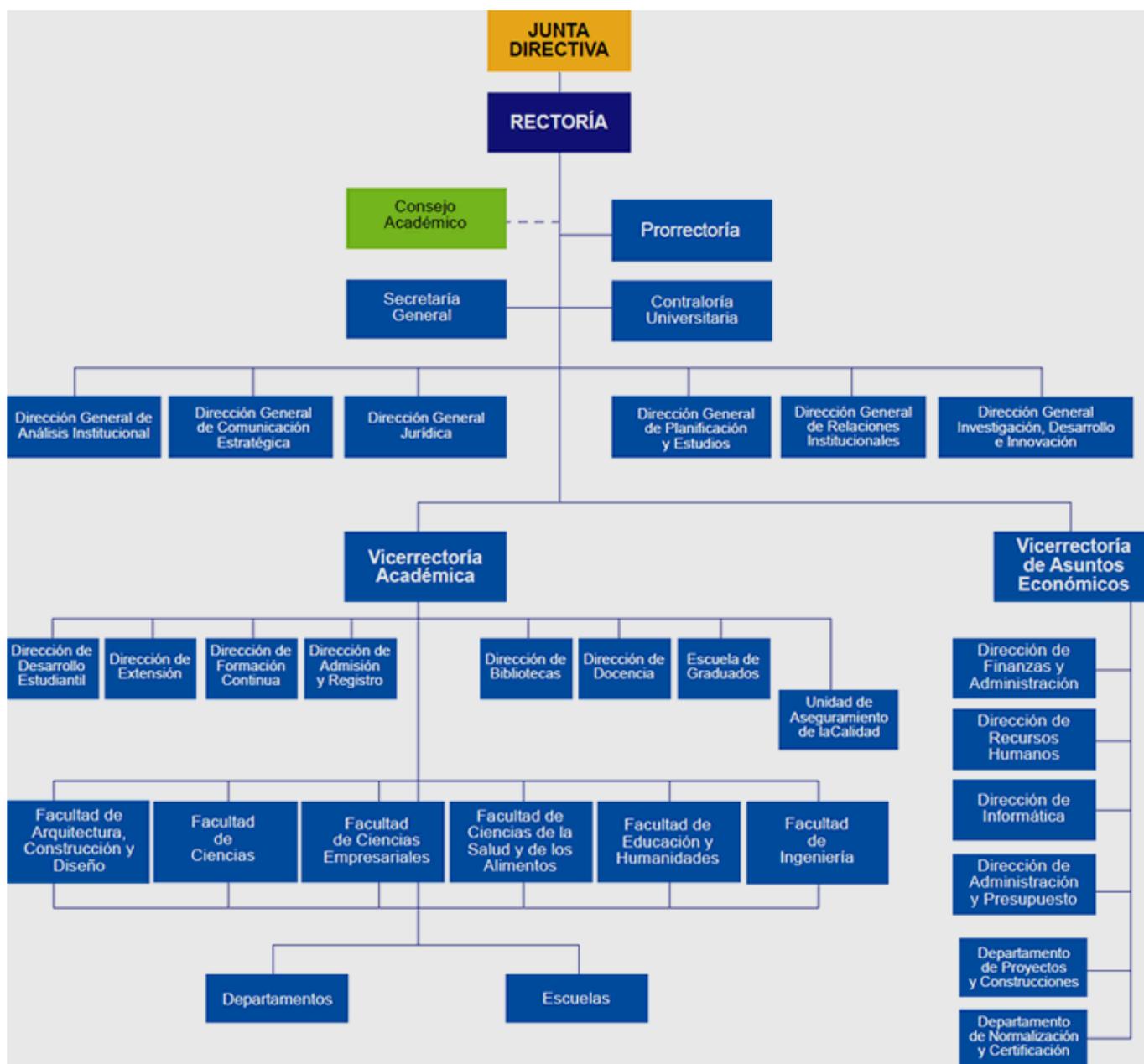


Figura N° 1: Organigrama de la Universidad Del Bío-Bío

Este es el organigrama completo de la institución actualizado al 16 de enero de 2018, donde están presentes todas las facultades y departamentos de la Universidad²

² Esta figura fue obtenida del siguiente enlace: http://ubiobio.cl/miweb/web2012.php?id_pagina=5152

Misión

La Universidad del Bío-Bío, a partir de su naturaleza pública, responsable socialmente y estatal, tiene por misión, desde la Región del Biobío, aportar a la sociedad con la formación de personas integrales, a través de una Educación Superior de excelencia. Comprometida con los desafíos de la región y del país, contribuye a la movilidad e integración social por medio de; la generación y transferencia de conocimiento avanzado, mediante la docencia de pregrado y postgrado de calidad, la investigación fundamental, aplicada y de desarrollo, la vinculación bidireccional con el medio, la formación continua y la extensión. Asimismo, impulsa el emprendimiento y la innovación, el fortalecimiento de la internacionalización y el desarrollo sustentable de sus actividades, basada en una cultura participativa centrada en el respeto a las personas.

Visión

Ser reconocida a nivel nacional e internacional como una Universidad pública, responsable socialmente y regional que, comprometida con su rol estatal, desde la Región del Biobío, forma personas integrales de excelencia y aporta a través de su quehacer al desarrollo sustentable de la región y el país.

Objetivos

El Modelo Educativo de la UBB tiene como propósito la formación integral de los estudiantes, considerando su desarrollo como persona y en la disciplina, aunando para ello en el modelo enseñanza y aprendizaje, el conocimiento, las habilidades y las actitudes.

La dimensión del desarrollo de la persona tiene como finalidad el perfeccionamiento de capacidades, que permitan el crecimiento humano como son las habilidades comunicativas, reflexivas, la responsabilidad personal y social.

La dimensión disciplinaria tiene como propósito fundamental el desarrollo de conocimientos y habilidades que le permitan la integración socio-laboral.

En consecuencia, la Universidad del Bío-Bío formará un profesional competente y autónomo que conjugue en su desempeño laboral las dimensiones de formación humana y disciplinares, que satisfaga las necesidades de la sociedad y de un mercado profesional cambiante en un mundo globalizado.

2.4 Descripción del área de estudio

El área de estudio del proyecto se centra en proveer a la Dirección de Informática de la Universidad Del Bio-Bío una arquitectura para el desarrollo de aplicaciones móviles dentro de la Universidad, para ello se realizará un estudio considerando la arquitectura que se utiliza actualmente dentro de la universidad y se desarrollarán distintas alternativas para la arquitectura que se propondrá.

2.5 Descripción de la problemática

En la actualidad la universidad cuenta con distintos tipos de aplicaciones móviles desarrolladas como lo son CA UBB, Admisión UBB, RA UBB o Tutor Virtual UBB, el Know-How en el desarrollo de estas aplicaciones no ha quedado en manos de la Dirección de Informática, ya que han sido desarrolladas por externos, y es por esto que han sido acomodadas dentro de los servidores de la universidad en la medida que fueron desarrolladas.

Debido a distintos factores, el Departamento de Desarrollo no ha podido definir una Arquitectura de Sistema para el desarrollo escalable de estas aplicaciones en el tiempo. Es por ello que se hace necesario ordenar y estructurar una alternativa capaz de evolucionar en medida de las necesidades de la comunidad universitaria.

Con el fin de satisfacer estas necesidades, en cuanto a las herramientas móviles disponibles en la universidad, se desarrollarán distintas aplicaciones con el fin de determinar cuál de ellas es la mejor alternativa a la hora de desarrollar una aplicación móvil, con esto se busca ayudar al Departamento de Desarrollo, generando una guía y Arquitectura adecuada para las futuras aplicaciones que se implementarán dentro y fuera de la Universidad.

3 DEFINICIÓN PROYECTO

3.1 Objetivos del proyecto

3.1.1 Objetivo General

Proponer una Arquitectura de Sistemas para el Desarrollo de Aplicaciones Móviles, que le permita a la Universidad del Bío-Bío un crecimiento móvil sostenible en el tiempo, con el objetivo de permitir el desarrollo e integración de las aplicaciones existentes dentro de la Universidad.

3.1.2 Objetivos Específicos

- Investigar las mejores prácticas desde el punto de vista de desarrollo de aplicaciones móviles.
- Definir una arquitectura que permita un mejor desarrollo de aplicaciones móviles.
- Definir una guía de desarrollo para el desarrollo de Aplicaciones Móviles dentro de la Universidad del Bío-Bío
- Desarrollar Distintas aplicaciones Móviles con el objetivo de comparar distintas tecnologías.

3.2 Ambiente de Ingeniería de Software

3.2.1 Metodología de Desarrollo

La metodología de desarrollo utilizada en el desarrollo de este proyecto será desarrollo iterativo e incremental, ya que, nos da la oportunidad de gestionar de mejor manera las expectativas del cliente, también se pueden obtener resultados importantes y usables desde las primeras iteraciones, además ante potenciales cambios en requisitos, esta metodología es la más adecuada porque al ser ágil es flexible a cambios de requisitos y estos cambios pueden ser tratados de acuerdo aparecen durante el desarrollo del proyecto, también en la finalización de cada iteración es la oportunidad donde el cliente puede proporcionar un feedback tras examinar el resultado obtenido. Con esta información ya es posible planificar los cambios necesarios para alinearse con las expectativas del cliente desde las primeras iteraciones, de manera que al finalizar el proyecto el cliente obtenga los objetivos esperados.

3.2.2 Técnicas

3.2.2.1 Lenguaje unificado de modelado

Para la documentación y modelado de este proyecto de software, se utiliza el Lenguaje unificado de Modelado (UML), específicamente para la elaboración de los Diagramas de Casos de Usos y Diagrama de Clases. UML es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema, cabe destacar que este lenguaje de modelos sirve para especificar o describir métodos o procesos.

3.2.2.2 Modelamiento de datos

Para la creación y modelado de la base de datos para este proyecto de software, se utiliza el modelo de entidad-relación que es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades. También se utiliza el modelo relacional que es para la creación de la base de datos.

3.2.3 Notación

Software: Sistema o aplicación informática que ayuda a automatizar un problema acotado.

SQL: Abreviación de Structured Query Language (Lenguaje Estructurado de Consultas), es el lenguaje actual para la gestión de base de datos relacionales.

Wireframe: Guía visual que representa el esqueleto o estructura visual del sitio web.

MER: Un diagrama o modelo entidad-relación es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades.

MR: El modelo relacional utilizado para la gestión de una base de datos, es un modelo de datos basado en la lógica de predicados y en la teoría de conjuntos.

UML: Lenguaje Unificado de Modelado, Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

HTML: Abreviación de Hypertext MarkupLanguage (Lenguaje de marcado de hipertexto), el cual es utilizada para la parte visible de la página.

PHP: Abreviación de PHP Hypertext Pre-processor, consiste en un lenguaje de programación usado generalmente en la creación de contenidos para sitios web.

HTTP: Abreviación de Hypertext Transfer Protocol (Protocolo de transferencia de hipertexto), el cual es utilizada para las aplicaciones web en una arquitectura cliente-servidor.

Herramienta CASE: Abreviación de Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora.

Base de Datos Relacional: Es una base de datos en donde todos los datos están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos operan sobre estas tablas.

Sistema: Es el conjunto de partes interrelacionadas (hardware, software) que permite almacenar y procesar información.

Usuario: Persona que utiliza o interactúa con las funciones del sistema.

XML: Abreviación de Extensible Markup Language, es un lenguaje de marcas utilizado para almacenar datos en forma legible.

3.2.4 Estándares

Para mantener una lectura y comprensión técnica del documento que respalda el desarrollo de este sistema, se deberá usar un estándar de acuerdo a la necesidad del proyecto. Es por esto que se utilizará una adaptación basada en IEEE Software Test Documentation Std 829-1998, la cual proveerá de las herramientas para la comprensión y correcta interpretación del desarrollo.

3.2.5 Herramientas de apoyo utilizadas

Sybase PowerDesigner

Herramienta CASE que permite diseñar, guardar y modificar los modelo de datos, los diagramas de casos de usos y el modelo relacional, todo esto en base reglas de UML. La versión utilizada es la 16.1.3637.

PHP

Es un lenguaje de programación que permite orientación a objetos, interpretado en el servidor y permite trabajar con toda la parte lógica del software. La versión utilizada es la 5.5.12

Sublime Text

Editor de texto con resaltado de sintaxis que entre sus características más relevantes están: su compatibilidad con múltiples lenguajes, autocompletado de sintaxis, auto indentación y fácil sistema para manejar proyectos grandes. La versión utilizada es la 2.0.2

MySQL database

Sistema administrador de bases de datos relacionales MySQL.

WampServer

Entorno de desarrollo web para Windows con el que podrás crear aplicaciones web con Apache, PHP y bases de datos MySQL database.

Navicat Premium

Es un administrador de bases de datos de múltiples conexiones que le permite conectarse a MySQL.

Photoshop

Es un editor de gráficos rasterizados desarrollado por Adobe Systems Incorporated. Usado principalmente para el retoque de fotografías y gráficos, su nombre en español significa literalmente "taller de fotos". La versión utilizada fue la CS3 Extended 10.0

WinSCP

Es un cliente SFTP gráfico para Windows que emplea SSH. Su función principal es facilitar la transferencia segura de archivos entre dos sistemas informáticos. La versión utilizada fue 5.6.7

Google Chrome

Es un navegador web desarrollado por Google y compilado con base en varios componentes e infraestructuras de desarrollo de aplicaciones. La versión utilizada fue 47.0.2526.106m

Microsoft Visual Studio

Es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación, tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby y PHP, al igual que entornos de desarrollo web, como ASP.NET MVC, Django, etc.

Ionic

Es un SDK completo de código abierto para el desarrollo de aplicaciones móviles híbridas. La versión original fue lanzada en 2013 y construida sobre AngularJS y Apache Cordova.

Android Studio

Es el entorno de desarrollo integrado oficial para la plataforma Android.

3.3 Definiciones, Siglas y Abreviaciones

Este ítem se incluye la definición de las siglas, abreviaciones, conceptos técnicos o de negocio que son necesarios para el buen entendimiento de este documento.

IDE

Integrated Development Environment, en español “Ambiente De Desarrollo Integrado”.

POST, GET, PUT y DELETE

Estos corresponden a operaciones del protocolo HTTP para la Creación (POST), Lectura (GET), Actualización (PUT) y Eliminación (DELETE) de datos.

Aplicaciones Híbridas

Las aplicaciones híbridas son básicamente páginas web que se ejecutan o renderizan dentro de un navegador web de una aplicación nativa. Es decir, el desarrollador crea la aplicación usando código web (como HTML, CSS, JavaScript, etc.)

API's

Application Programming Interface, es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Frameworks

Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Aplicaciones móviles

Son aplicaciones informáticas diseñadas para ser ejecutadas en teléfonos inteligentes, tabletas, etc.

Bytecode

Es un código intermedio más abstracto que el código máquina.

Ahead of the Time

En informática, es el acto de compilar un lenguaje de programación de alto nivel como C#, AOT transforma el bytecode de una máquina virtual (VM) a código máquina.

Just-in-Time

Es una técnica para mejorar el rendimiento de sistemas de programación que compilan a bytecode, consistente en traducir el bytecode a código máquina nativo en tiempo de ejecución.

APK's

Es un formato de archivo que se usa para empaquetar aplicaciones para el sistema operativo para dispositivos móviles Android

Asincrónico

Es aquella comunicación que se establece de manera diferida en el tiempo, es decir, cuando no existe coincidencia temporal.

Interfaz de usuario

Es el medio con que el usuario puede comunicarse con una máquina, equipo, computadora o dispositivo, y comprende todos los puntos de contacto entre el usuario y el equipo.

FrontEnd

Parte del sistema responsable de recolectar los datos de entrada del usuario, en palabras más simples es la aplicación que interactúa con el usuario.

Backend

Parte del sistema que se encarga de interactuar y administrar las bases de datos.

Bindings

Es una adaptación de una biblioteca para ser usada en un lenguaje de programación distinto de aquel en el que ha sido escrita.

4 MARCO TEÓRICO

El origen de la arquitectura está ligado con la necesidad de estructurar, se refiere a la forma en la que es diseñada, estas pueden ser definidas de una forma física o de forma lógica.

El diseño físico está ligado a la acción o proceso de construir algo, podemos describir esta arquitectura como el arte o ciencia de la construcción de edificios para el uso humano. El Diseño lógico en cambio desde un punto de vista conceptual podemos hacer referencia a la organización lógica de una aplicación y sus componentes o de un sistema basado en computadoras, desde la perspectiva de su uso y diseño.

En este capítulo se tratan los temas de las Arquitecturas de Sistemas, una breve historia de sus orígenes, descripción de arquitecturas existentes para el desarrollo de aplicaciones (se realizara un análisis de la arquitectura implementada en la Universidad del Biobío y de la arquitectura que se propondrá).

El presente capítulo está basado en el libro Microsoft Application Architecture Guide, 2nd Edition, 2001. (Microsoft, 2001)

4.1 Que es la Arquitectura de Software

La Arquitectura de una aplicación de software es el proceso de definición de una solución estructurada que satisfaga todos los requisitos técnicos y operativos, al tiempo que optimiza sus características, como son el rendimiento, la seguridad y la capacidad de gestión. Implica una serie de decisiones basadas en una amplia gama de factores y cada una de estas decisiones pueden tener un impacto considerable en la calidad, el rendimiento, la facilidad de mantenimiento y el éxito general de la aplicación.

"La arquitectura de software de un programa o sistema informático, es la o las estructuras del sistema, que comprenden elementos de software, las propiedades externamente visibles de esos elementos y las relaciones entre ellos. La arquitectura se refiere al lado público de las interfaces; Los detalles que tienen que ver únicamente con la implementación interna, no son arquitectónicos" (Bass, Clements, & Kazman, 2003).

Martin Fowler describe algunos temas recurrentemente comunes a la hora de explicar la arquitectura. Identifica estos temas como:

"El desglose de nivel más alto de un sistema en sus partes; Las decisiones que son difíciles de cambiar; Hay múltiples arquitecturas en un sistema; Lo que es significativo desde el punto de vista arquitectónico puede cambiar a lo largo de la vida de un sistema; Y, al final, la arquitectura se reduce a lo que sea importante" (Fowler, 2002).

4.2 Principios clave de la arquitectura

4.2.1 Estilos de Arquitectura

Un estilo de Arquitectura, conocido también como Patrón de Arquitectura, es un conjunto de principios que proporcionan un guía para un conjunto de sistemas. Una arquitectura mejora la partición y promueve la reutilización de algún diseño para satisfacer problemas que son frecuentes, podemos ver a la arquitectura como los principios que configuran una aplicación. Entender cómo funcionan los estilos de arquitectura nos proporcionan muchos beneficios entre ellos el más importante es que se proporciona una lenguaje común. Además proporcionan oportunidades para las personas que no entiende cómo funciona la tecnología. Esto facilita la comunicación y comprensión de los patrones utilizados sin entrar en mucho detalle.

La Tabla N° 1³ muestra las principales categorías o áreas de desarrollo con sus respectivas Arquitecturas.

CATEGORÍA	ESTILO DE ARQUITECTURA
COMUNICACIÓN	Arquitectura Orientada A Servicios (SOA)
	Bus de Mensajes (ESB)
	Cliente/servidor
DESPLIEGUE	N-Capas
	3 Capas
DOMINIO	Diseño Impulsado Por el Dominio
ESTRUCTURAL	Arquitectura basada en componentes
	Orientada a objetos

Tabla N° 1: Categorías de Estilos de Arquitectura

³ Tabla basada en la lectura del libro Microsoft Application Architecture Guide, 2nd Edition, 2001.

4.2.2 Arquitectura Cliente/Servidor

Esta Arquitectura incluye sistemas distribuidos que implican un Cliente y un servidor independiente además de una red de conexión. La forma más simple de esta arquitectura abarca una aplicación Servidor a la que los clientes acceden directamente, este tipo de Arquitectura es denominada Arquitectura de 2 Capas.

Desde siempre la arquitectura Cliente/Servidor obligaba a tener una aplicación gráfica de interfaz para el usuario que se comunicaba con un servidor de bases de datos, este último es quien contenía gran parte de la lógica usada por la empresa en forma de procedimientos de almacenamiento o de un servidor de archivos dedicados. De Forma general, esta arquitectura describe la relación entre un cliente y uno o más servidores, donde el cliente inicia una o más solicitudes, espera las respectivas respuestas y las procesa. El Servidor normalmente autoriza o valida al usuario y luego lleva a cabo el procesamiento requerido para generar el resultado, este puede enviar respuestas usando una variedad de protocolos y formatos de comunicación para informar al cliente.

Hoy en día, algunos modelos de esta arquitectura incluyen programas basado en navegación web que se ejecutan en internet o en una red Interna, podemos nombrar entre estas las aplicaciones basadas en sistemas operativos que acceden a servicios de red, aplicaciones que acceden a almacenes de datos remotos y herramientas que manipulan sistemas remotos.

4.2.2.1 Variaciones de Esta Arquitectura

Cliente - Cola - Cliente

Este enfoque permite al cliente comunicarse los unos con los otros a través de una cola en el servidor. Los Clientes pueden leer datos y enviar datos a un servidor, que actúa como una cola para almacenar los datos. Esto permite que el cliente decida cómo distribuir y sincronizar los archivos e información. Esta variación también es conocida como una Arquitectura de Cola pasiva

Aplicaciones Punto a punto (P2P)

Desarrollado a partir del modelo anterior el estilo P2P permite que el cliente y servidor intercambien sus funciones para la distribución y sincronización de archivos e información a

través de varios clientes. Deriva de la Arquitectura Cliente/Servidor, ya que puede realizar múltiples respuestas a solicitudes.

- ***Servidores de Aplicaciones***

Estilo de Arquitectura en el que el servidor hospeda y ejecuta aplicaciones y servicios, a los que el cliente accede desde un navegador web o una aplicación instalada.

4.2.2.2 Beneficios de la Arquitectura Cliente/Servidor

- ***Mayor Seguridad***

Todos los datos se almacenan en el servidor, lo que entrega un mayor control de seguridad los terminales de los clientes

- ***Acceso centralizado a datos***

Puesto que los datos se almacenan solo en el servidor, el acceso y las actualizaciones de los datos son mucho más fáciles de administrar que en otro tipo de arquitectura.

- ***Facilidad De Mantenimiento***

Las funciones y responsabilidades de un sistema se distribuyen entre varios servidores que se conocen entre sí a través de una red. Esto garantiza que un cliente no se vea afectado a la hora de realizar una reparación, actualización o reubicación del servidor.

Este estilo arquitectónico es recomendable si la aplicación está basada en servicios y con un gran número de clientes, es recomendado cuando se desea centralizar las funciones de almacenamiento, administración de datos, o cuando la aplicación debe admitir diferentes tipos de Clientes que vienen de distintos dispositivos.

Sin embargo el estilo tradicional de Cliente servidor lleva consigo numerosas desventajas, en las que se incluyen por ejemplo, la tendencia a que los datos de la aplicación y la lógica de negocios se combinen en el servidor, y por lo tanto se ve afectada la extensibilidad y escalabilidad del sistema, esto se traduce en una menor fiabilidad del sistema.

4.2.3 Arquitectura Basada en Componentes

La Arquitectura basada en componentes describe un enfoque de Ingeniería de Software para el diseño y desarrollo de un sistema. Se centra en la descomposición del diseño en componentes funcionales o lógicos que entregan interfaces de comunicación bien definidas, que contienen métodos, eventos y propiedades. Esto proporciona un mayor nivel de abstracción que el que entrega un diseño orientado a objetos, además no se centra en protocolos de comunicación ni estado compartido. Esta Arquitectura y como su nombre lo dice se basa en componentes y deben cumplir al menos con estos principios

- **Reutilizable**
Los Componentes suelen estar diseñados para ser reutilizados en diferentes circunstancias de distintas aplicaciones, esto no significa que alguno de sus componentes pueda estar diseñados para una tarea específica.
- **Reemplazable**
Los componentes pueden ser fácilmente sustituidos con otros similares
- **No específico de contexto**
Los Componentes están diseñados para operar en diferentes entornos y contextos. La información específica, como los datos de estado, deben ser entregadas al componente en vez de ser incluidas como parte de este.
- **Extensible**
Puede extenderse de uno o más componentes para proporcionar un nuevo comportamiento o servicio.

- **Encapsulado**

Los Componentes entregan una interfaz que permite al usuario que utiliza su funcionalidad pero no revela detalles de los procesos internos ni de ninguna variable o estado interno.

- **Independencia**

Los componentes están diseñados para tener dependencias mínimas de otros componentes, y es gracias a esto que pueden ser implementados en cualquier otro entorno compatible, sin afectar a otros componente o sistemas.

Los tipos más comunes de componentes utilizados en aplicaciones incluyen componentes de interfaz de usuario que están compuestas por controles (particiones y botones), componentes auxiliares, utilizados para plantear un subconjunto específico de funciones empleadas en otros componentes. Otros tipos comunes de componentes son aquellos que requieren muchos recursos y no acceden con frecuencia, estos deben acceder o activarse con un enfoque justo a tiempo (JIT), muy comunes en escenarios donde existen componente remotos o distribuidos.

Los componentes dependen de un dispositivo dentro de la plataforma que proporciona un entorno en el que pueden ejecutarse. Las Arquitecturas Basadas en componentes gestionan el funcionamiento de localización de componentes y sus interfaces, pasando mensajes o comandos entre componentes, y en algunos casos incluyen el mantenimiento de estados.

4.2.3.1 Beneficios de la Arquitectura Basada en Componentes

- ***Facilidad de Despliegue***

A medida que existan nuevas versiones disponibles, puede reemplazar las versiones existentes sin ningún impacto en los demás componentes o en el sistema como conjunto.

- ***Costo Reducido***

El uso de componentes de terceros permite distribuir los costos de desarrollo y mantenimiento.

- **Facilidad de Desarrollo**

Los componentes implementan interfaces que se conocen entre sí, para proporcionar funcionalidades definidas, permitiendo el desarrollo sin afectar otras partes de sistema.

- **Reutilización**

El uso de componentes reutilizables significa que pueden utilizarse para distribuir el costo del desarrollo y mantenimiento en varias aplicaciones o sistemas.

- **Disminuye la complejidad Técnica**

Los componentes disminuyen la complejidad mediante el uso de un contenedor de componentes y sus servicios, estos últimos incluyen la gestión y activación de componentes con sus eventos y transacciones.

Esta arquitectura es recomendada cuando ya se cuentan con los componentes adecuados; Cuando la aplicación a ejecutar es de procesamiento, con poca o nula entrada de datos; Cuando se desea crear una arquitectura confiable que permita reemplazar y actualizar fácilmente los componentes.

4.2.4 Arquitectura impulsada por Dominio

Es un enfoque orientado objetos para diseñar software basado en el dominio del negocio, sus elementos y comportamientos y las relaciones entre ellos. Su objetivo es permitir que los sistemas que son diseñados por la empresa puedan ser utilizados por los expertos en el negocio. El modelo de dominio puede ser visto como un marco en el que las soluciones son racionalizadas.

Para aplicar esta arquitectura se debe tener una muy buena comprensión del dominio de negocio que se desea modelar, o ser diestro en la adquisición de los conocimientos necesarios. Esta arquitectura sugiere que el equipo de desarrollo a menudo trabaje con expertos en el dominio de negocios para darle forma al dominio. Arquitectos, desarrolladores y expertos en el área se comprometen a utilizar solo un lenguaje que se centre en el dominio del negocio, excluyendo cualquier jerga técnica que pueda surgir de alguna de las partes.

Como núcleo de la arquitectura se utiliza el lenguaje compartido, permitiendo encontrar rápidamente algún vacío del software solo con analizar el lenguaje que lo rodea. La creación de un lenguaje común se implementa como solución que resuelve el problema de malentender el lenguaje del dominio ya que este en si puede llegar a ser ambiguo.

Con el fin de ayudar a mantener la arquitectura como una construcción del lenguaje útil, se debe implementar una gran cantidad de aislamiento y encapsulamiento dentro de la arquitectura de dominio, por lo que puede llegar a tener un costo relativamente alto. Si bien este diseño proporciona muchos beneficios técnicos, como lo son la capacidad de mantenimiento, debe aplicarse solo a dominios complejos donde los modelos y los procesos lingüísticos proporcionen claros beneficios en la comunicación de información compleja y la comprensión común del dominio.

4.2.4.1 Beneficios de la Arquitectura impulsada por Dominio

- ***Comunicación***

Todas las partes dentro del equipo de desarrollo pueden utilizar el modelo de dominio y las entidades que definan para comunicar conocimientos y requisitos, utilizando un lenguaje común sin necesidad de incorporar jergas técnicas

- ***Extensibilidad***

El modelo de dominio es a menudo modular y flexible, por lo que es fácil actualizar y ampliar a medida que se cambian las condiciones y requisitos

- ***Probable***

Los objetos del modelo están ligeramente acoplados cohesivos, permitiendo que sean probados más fácilmente. Esta Arquitectura puede ser implementada si se tiene un modelo complejo y se desea mejorar la comunicación y la comprensión dentro del equipo de desarrollo, o donde se deba declarar el diseño de una aplicación con un lenguaje común que todas las partes de equipo pueden comprender.

4.2.5 Arquitectura en capas

La arquitectura en capas se enfoca en el agrupamiento de la funcionalidad dentro de una aplicación, en distintas capas que se apilan verticalmente una encima de la otra. La funcionalidad dentro de cada capa está relacionada por un rol o responsabilidad común. La comunicación entre capas es explícita y ligeramente acoplada.

Este estilo de arquitectura en capas se puede describir como una pirámide invertida de la reutilización donde cada capa agrega las responsabilidades y las abstracciones de la capa que está por debajo de ella misma. Con capas estrictas, los componentes de una capa solo pueden interactuar con componentes de la misma capa o con componentes que se encuentren la capa que está directamente debajo de ella. Una capa más relajada permite que los componentes de una capa interactúen con componentes de la misma capa o con componente de cualquier capa inferior.

Las capas de una aplicación pueden habitar en el mismo equipo físico (Mismo Nivel) o pueden distribuirse en ordenadores separados (N-Capas) y los componentes de cada capa se comunican con componentes de otras capas a través de interfaces bien definidas. Un ejemplo de ello puede ser una aplicación web, que consiste en una capa de presentación (Funcionalidad relaciona con la interfaz de usuario), una capa empresaria (donde se procesan las reglas de negocio) y una capa de datos (Funcionalidad relacionada con el acceso a los datos).

4.2.5.1 Principios de una Arquitectura en capas

- **Abstracción**

La arquitectura en capa abstrae la vista del sistema como un todo, proporcionando detalles suficientes para comprender los roles y las responsabilidades de las capas individuales y la relación entre ellas.

- ***Encapsulamiento***

No se necesita hacer suposiciones sobre tipos de datos, métodos y propiedades o implementación durante el diseño, ya que estas características no se muestran en los límites de la capa

- ***Capas funcionales claramente definidas***

La separación entre funcionalidades en cada capa es clara. Las capas superiores, como la capa de presentación, envían comandos a capas inferiores, como las capas empresariales y de datos, y pueden reaccionar a eventos en estas capas, permitiendo que los datos fluyan hacia arriba y abajo entre las capas.

- ***Alta cohesión***

Los límites de responsabilidad están bien definidos para cada capa, de esta forma se asegura que cada capa contenga funcionalidad directamente relacionada con las tareas que le corresponden, lo que ayuda a maximizar la unión dentro de cada capa.

- ***Reutilizable***

Las capas inferiores no tienen dependencias en las capas superiores, lo que potencialmente les permite ser reutilizables en otros escenarios.

- ***Acoplamiento suelto***

La comunicación entre capas se basa en la abstracción y eventos para proporcionar un acoplamiento suelto entre las capas.

Un ejemplo de la arquitectura en capas son las aplicaciones enfocadas en la línea de negocios, tales como sistemas de contabilidad y gestión de clientes; aplicaciones y sitios web basados en Webs empresariales, y servidores de aplicaciones centralizados para la lógica empresarial.

Los patrones de Presentación Separada manejan las interacciones del usuario desde la interfaz de usuario, la presentación y la lógica de negocio y los datos de aplicación con los que trabaja el usuario. Esto permite a los diseñadores gráficos crear una interfaz de usuario mientras los desarrolladores generan el código que la moverá.

4.2.5.2 Principios Claves de patrones de presentación Separados

- ***Separación de Preocupaciones***

Los patrones de presentación separados dividen las preocupaciones de procesamiento de la interfaz de usuario en papeles distintos; Por ejemplo, el modelo MVC tiene tres funciones; El modelo, la vista y el controlador. El modelo representa datos; La vista representa la interfaz de usuario; Y el controlador maneja las peticiones, manipula el modelo y realiza algún otro tipo de operaciones

- ***Notificación basada en eventos***

Los patrones de observación se utilizan comúnmente para proporcionar notificaciones a la vista cuando los datos administrados por el modelo cambian.

- ***Manejo de eventos delegados***

El controlador controla los eventos desencadenados desde los controles de la interfaz de usuario en la vista.

4.2.5.3 Beneficios de la Arquitectura en Capas

- ***Abstracción***

Las capas permiten que los cambios se realicen en el nivel abstracto. Pueden aumentar o disminuir el nivel de Abstracción que utiliza cada capa de la pila jerárquica.

- ***Aislamiento***

Permite aislar actualizaciones de tecnología a capas individuales para reducir el riesgo y minimizar el impacto en el sistema general

- ***Manejabilidad***

La separación ayuda a identificar las dependencias y organiza el código en secciones manejables

- **Rendimiento**

La distribución de las capas en varios niveles físicos puede mejorar la escalabilidad, tolerancia a fallos y rendimiento.

- **Reutilización**

Los roles promueven la reutilización. Por ejemplo en MVC, el controlador a menudo puede reutilizarse con otras vistas compatibles con el fin de proporcionar un rol específico o una vista personalizada por el usuario a los mismos datos y funcionalidad.

- **Testabilidad**

El aumento de la Testabilidad surge de tener interfaces de capa bien definidas, así como la capacidad de cambiar entre diferentes implementaciones de las interfaces de capa. Los patrones de presentación separados permiten crear objetos simulados que imitan el comportamiento de objetos concretos como el modelo, el controlador o la vista durante la prueba.

Este estilo de arquitectura es recomendable si las capas existentes son adecuadas para la reutilización en otras aplicaciones, si la aplicación debe admitir diferentes tipos de clientes y dispositivos diferentes o si se desea implementar reglas y proceso empresariales complejos y/o configurables.

4.2.6 Arquitectura De N-Capas/3-capas

Es un estilo de Implementación Arquitectónica en que se describen la separación de funcionalidad en segmentos de la misma manera que el estilo en capas, pero con cada segmento siendo un nivel que puede ubicarse en un equipo físicamente separado, ellos evolucionaron a través del enfoque orientado a componentes, generalmente utilizando métodos específicos de plataforma para la comunicación en lugar de un enfoque basado en mensajes.

La arquitectura de aplicación de N niveles se caracteriza por la descomposición funcional de las aplicaciones, los componentes del servicio y su implementación distribuida, proporcionando escalabilidad, disponibilidad, capacidad de administración y utilización de

recursos mejoradas. Cada nivel es completamente independiente de todos los demás niveles, excepto los inmediatamente superiores o inferiores. El N-ésimo nivel solo tiene que saber cómo manejar una solicitud desde el nivel N+1, como reenviar esa solicitud al nivel n-1 (si existe) y cómo manejar los resultados de la solicitud. La comunicación entre niveles es típicamente asíncrona para soportar la escalabilidad.

Las Arquitecturas de nivel N normalmente tienen al menos 3 partes lógicas separadas, cada una ubicada en un servidor físico independiente. Cada parte es responsable de la funcionalidad específica. Cuando se utiliza un enfoque de diseño en capas, se despliega una capa en un nivel si más de un servicio o aplicación depende de la funcionalidad expuesta por la capa.

4.2.6.1 Los principales beneficios de la Arquitectura N-Capas/3-Capas

- ***Mantenibilidad***

Debido a que cada nivel es independiente de los otros niveles, las actualizaciones o cambios se pueden llevar a cabo sin afectar la aplicación en su conjunto.

- ***Escalabilidad***

Debido a que los niveles se basan en el despliegue de capas, la ampliación de una aplicación es razonablemente sencilla.

- ***Flexibilidad***

Debido a que cada nivel puede ser administrado o escalado independientemente, la flexibilidad se incrementa.

- ***Disponibilidad***

Las aplicaciones pueden aprovechar la arquitectura modular de sistemas habilitadores que utilizan componentes fácilmente escalables, lo que aumenta la disponibilidad.

Este estilo arquitectónico puede ser utilizado si los requisitos de procesamiento de las capas de aplicación difieren de tal manera que el procesamiento en una capa podría absorber suficientes recursos para retardar el procesamiento en otras capas o si los requisitos de seguridad de las capas de la aplicación difieren. Un ejemplo puede ser que la capa de presentación no debe almacenar datos confidenciales, mientras que esto puede almacenarse en las capas de datos y de negocio. El estilo Arquitectónico de N Niveles o de 3 niveles también es apropiado si se desea compartir la lógica empresarial entre aplicaciones y tiene suficiente hardware para asignar el número de servidores necesario en cada nivel.

5 ARQUITECTURA ACTUAL UTILIZADA EN LA UNIVERSIDAD DEL BÍOBÍO

Al momento de ser escrito este documento la Universidad del Bio-Bío utiliza una arquitectura de tres capas para el desarrollo de software, esta arquitectura se compone de tres partes que se ejemplifican en el Figura N°2.

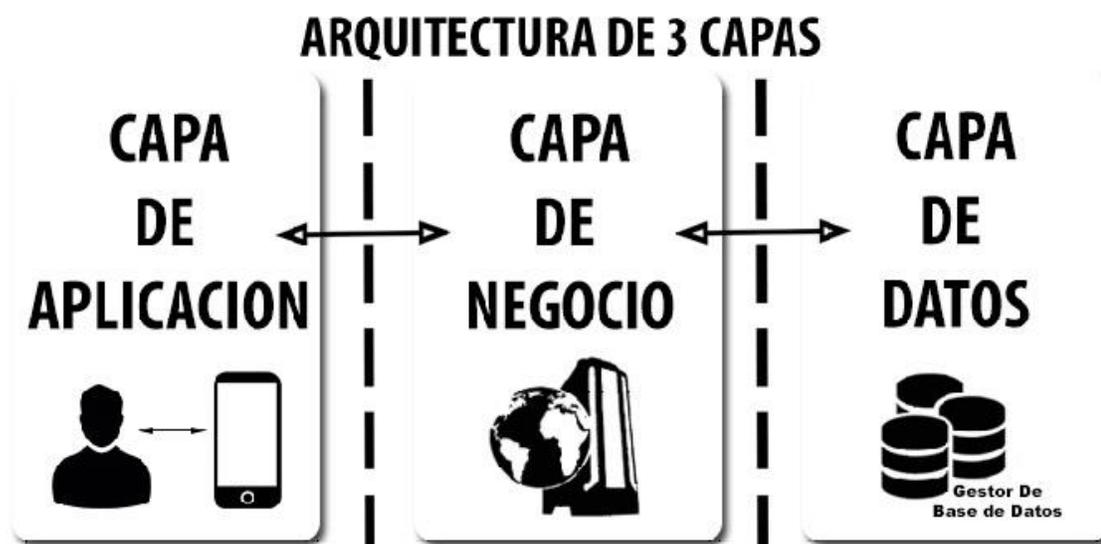


Figura N° 2: Arquitectura de 3 Capas

Primero la Capa de Aplicación, es la que el usuario final ve, también se le denomina capa de usuario, ya que es quien presenta el sistema al usuario y suele ser amigable con quien utilice el sistema. Esta capa solo se comunica con la capa de negocio.

En segundo lugar tenemos la Capa De Negocio que es donde habitan los programas que se ejecutan, se reciben las peticiones y se envían las respuestas tras el proceso. Esta capa se comunica con las otras dos capas que componen esta arquitectura, primero con la capa de presentación, para recibir las solicitudes y presentar los resultados al usuario, y luego con la capa de datos para solicitar al gestor de base de datos almacenar o recuperar datos de él.

Por último nos encontramos con la Capa de Datos que es la encargada de acceder a los datos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento

de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

En el actual capítulo descompondremos esta arquitectura para revisar detalladamente cada una de las capas para observar los componentes que conforman cada una de ellas.

5.1 Capa De Aplicación

Esta capa está compuesta por un desarrollo de Aplicaciones Nativas por parte de ingenieros contratados en la universidad y a quienes se les da la tarea de crear e implementar soluciones móviles, ellos utilizan plataformas o ambientes de desarrollo nativos como Android Studio para crear aplicaciones móviles que puedan ser usadas por la universidad.

Las aplicaciones nativas están desarrolladas y optimizadas específicamente para el sistema operativo para el que se desarrollaron (Android, BlackBerry, IOS, etc). Este tipo de aplicaciones se adapta en un cien por ciento a las funcionalidades y características del dispositivo, obteniendo una mejor experiencia de uso.

Dentro de las ventajas de desarrollar con aplicaciones Nativas tenemos:

- Obtener un máximo rendimiento en dispositivos ya que se conoce el funcionamiento interno de la aplicación y se desarrolla exclusivamente en ese lenguaje
- Se incrementa la buena experiencia de usuario ya que se adapta al dispositivo que el usuario utiliza diariamente.

Actualmente para desarrollar en cada plataforma se utiliza un lenguaje de programación distinto como lo describe la Tabla N° 2

Sistema Operativo	Fabricante	Lenguaje de Programación
Android	Google	java
iOS	Apple	Objective C, Swift
Windows Phone	Microsoft	C#, Visual Basic, .NET
BlackBerry	RIM	C/C++

Tabla N° 2: Aplicaciones Nativas

(Cruz, 2014) “Las aplicaciones nativas se denominan así porque se desarrollan en el lenguaje nativo del propio terminal. Estas aplicaciones se alimentarán de los recursos del propio Smartphone, teniendo acceso a diferentes características como la cámara, el GPS, entre otras.”

Sin embargo, el desarrollo de una aplicación nativa conlleva un mayor costo, puesto que si se desea realizar una aplicación multiplataforma se debe fabricar una nueva versión para cada sistema operativo, multiplicando así los costos de desarrollo.

5.2 Capa De Negocio

Actualmente la universidad en su capa de negocios trabaja con una API Rest, para entender más de cómo funciona esta lógica de negocios describiremos brevemente que es una api y para que están diseñadas.

5.2.1 Objetivos del diseño de una API

Dentro de los amplios objetivos del diseño de API's se pueden definir de la siguiente manera:

- Ofrecimiento de autoservicio para desarrolladores y usuarios de aplicaciones
- Reducción de barreras de acceso a valiosos recursos empresariales
- Priorización de las necesidades y preferencias de los desarrolladores de aplicaciones de clientes
- Fomento de la colaboración entre recursos internos y externos
- Enfoque de los problemas de seguridad y escalación derivados de la exposición de los activos de TI al mercado abierto

(CA Technology, 2015). “*Lo más importante de todo es que el diseño de API debe centrarse en la maximización del valor de negocio de la interfaz*”.

5.2.2 ¿Qué es Una Api REST?

Una Api Rest o Api de Transferencia de Estado Representacional (siglas en Ingles) fue definida el año 2000 por Roy Fielding quien es considerado el padre de la especificación HTTP. Api Rest cambió por completo la Ingeniería de Software ya que propuso un nuevo enfoque de desarrollo de proyectos y servicios web. Hoy en día no existe proyecto o

aplicación que no disponga de Una Api Rest para la creación de algún servicio a través de este software. Siendo más específicos Rest es cualquier interfaz entre sistemas que utilice los protocolos de HTTP para obtener datos o generar operaciones sobre esos datos, utilizando uno de los dos principales formatos XML y JSON.

Rest posee un protocolo cliente/servidor sin estado que trata cada petición como una transacción independiente que no tiene relación con alguna solicitud anterior, la comunicación se compone de pares independientes de solicitud y respuesta, por lo tanto ni cliente ni servidor necesitan recordar ningún estado previo.

La Api puede manipular los datos utilizando las peticiones POST, GET, PUT y DELETE que son propias del protocolo HTTP. Los objetos obtenidos por rest siempre se manipulan a partir de la URI (del inglés uniform resource identifier), quien es la encargada de identificar cada recurso del sistema Rest. Trabaja basado en una Arquitectura jerárquica de capas, cada una de estas capas lleva a cabo una funcionalidad dentro del sistema REST. Mostrándolo de manera sencilla Api Rest se representa en la figura N° 3

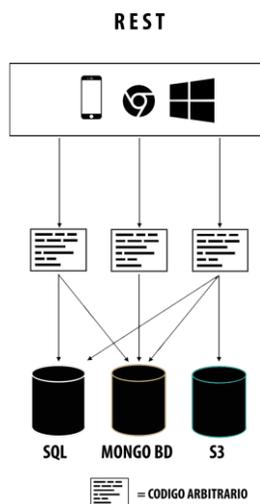


Figura N° 3: API Rest

La Figura anterior muestra el funcionamiento de una Api Rest quien recibe en su capa de negocio dispone de distintos archivos encargados de procesar información específica para cada petición.

5.2.3 Ventajas de Una Api Rest

Los siguientes párrafos están basados en el artículo “Ventajas e inconvenientes de API REST para el desarrollo” (Alvarez, 2014).

5.2.3.1 La separación Cliente/Servidor

El cliente y el servidor son totalmente independientes, al cliente le da igual cómo está hecha la Api, y con el servidor pasa lo mismo, a este no le importa que es lo que se hará con los datos que ha proporcionado. Si se necesita evolucionar o re factorizar uno de los dos, ya sea el FrontEnd o el Backend, estas tareas se realizan de forma independiente siempre y cuando se mantengan los estándares que les proporciona la API. Pueden coexistir varios FrontEnd con un único Backend, es decir, no necesariamente el FrontEnd debe ser un servicio WEB, puede ser una App para Android y otra App para iOS.

5.2.3.2 Independencia de tecnologías / lenguajes

Se puede desarrollar en cualquier tipo de tecnología o lenguaje que necesite el proyecto siempre que se respete el convenio, es decir que sigan teniendo las mismas operaciones con las que trabaja la Api y realicen las mismas tareas que se supone deben hacer.

5.2.3.3 Fiabilidad y escalabilidad

Como tercer punto Álvarez destaca la Fiabilidad que Rest entrega, ya que solo se debe procurar que la comunicación entre el Cliente y el servidor sea el correcto, se pueden realizar cambios en el servidor, lenguajes, bases de datos etc. pero mientras se devuelvan los datos correctamente todo irá bien. En cuanto a la escalabilidad se destaca la facilidad para crecer según lo que se necesite, la api puede responder a otros tipos de operaciones o puede versionarse tanto como se desee, además se puede crecer por el lado del cliente, no solo en la parte web sino que también en el desarrollo de apps para cualquier dispositivo.

5.2.3.4 Operaciones muy bien definidas

Se aplica todos los recursos de Información, HTTP en si define un conjunto de operaciones, y dentro de este los más importantes son POST, GET, PUT, DELETE. Con frecuencia estas operaciones son comparables con las utilizadas para las modificaciones dentro de las bases de datos conocidas como CRUD.

5.3 Capa de Datos

La capa de Datos es donde habitan los datos de un sistema y es la encargada de gestionar estos mismos. Está formada por uno o más gestores de bases de datos que realizan todas las tareas de almacenamiento, administración o recuperación de datos que llegan desde la capa de negocio.

Actualmente la Universidad del Bio-Bío cuenta con una Base De Datos Corporativa que se encarga de administrar los datos del sistema.

5.3.1 Base De Datos Corporativa

Es una aplicación cliente-servidor, desarrollada a la medida de la empresa y que se crea empleando las distintas tecnologías, las principalmente usadas son

Bases de datos relacionales

Programación orientada a objetos

Lenguajes .NET

Sistema de Gestión de Bases de Datos Relacional Microsoft SQL Server

Su principal función es la gestión de datos almacenados en tablas relacionadas y optimizadas de forma práctica, para obtener de forma eficaz los datos.

6 SELECCIÓN DE ESTILO DE ARQUITECTURA

A la hora de escoger una arquitectura para el desarrollo de aplicaciones móviles, se deben tener en cuenta muchos factores con respecto al tipo de aplicaciones que se quiere implementar, estas se dividen en dos grupos según la información que imparten:

- Aplicaciones auto-contenidas
- Aplicaciones con conexión a Internet

6.1.1 Aplicaciones móviles auto-contenidas

En este tipo de aplicaciones el contenido es estático, sus imágenes, información, menús, etc. no cambian ya que todo su contenido se encuentra guardado dentro de la misma aplicación, un ejemplo de esto puede ser la calculadora que está integrada en cada uno de los Smartphone ya que esta no tiene la necesidad de cambiar los datos y fórmulas que tiene implementadas y la información que necesita para su funcionamiento se ingresa de manera manual cada vez que se requiera realizar un nuevo cálculo.

La principal ventaja en este tipo de aplicaciones es que su costo de desarrollo es bajo y no hay que incurrir en gastos extras como contratar servicios de hosting o de bases de datos. Pero esto también significa que a la hora de realizar algún cambio en la aplicación esta tendría que desarrollarse y compilarse nuevamente para la plataforma en la que fue desarrollada.

6.1.2 Aplicaciones móviles con conexión a Internet

En estas aplicaciones la información que se muestra al usuario esta almacenada en un Back-End, a la cual se accede a través de una API de servicios Web, un ejemplo claro de esto son las aplicaciones de redes sociales como Facebook, Twitter, Whatsapp, las cuales necesitan el servicio de internet para poder acceder a información específica o para realizar alguna acción dentro de la misma.

La principal ventaja es que los cambios en la información que entregan las aplicaciones se realizan de manera remota y se actualizan cada vez que estas sufren algún cambio en el servidor, esto también conlleva un mayor costo a la hora de desarrollarlas, debido a que su desarrollo es más complejo se necesita contratar personal especializado para el desarrollo y agregar un costo para la contratación de un servicio de hosting.

Ya que las aplicaciones desarrolladas dentro de la universidad, son de conexión a Internet, de contenido dinámico, con alta demanda de usuarios, y teniendo en mente las arquitecturas descritas en el capítulo anterior, la arquitectura que mejor se adapta estas necesidades es la Arquitectura De Tres Capas, ya que cada una de sus capas cumple una función específica y se comunican entre ella para dar un mejor funcionamiento a la aplicación que se está desarrollando.

Con el fin de determinar las mejores prácticas de implementación de esta, en los siguientes capítulos se desglosan y estudian las dos primeras capas de la arquitectura, proponiendo en cada una de estas una alternativa al desarrollo actual.

7 CAPA DE APLICACIÓN, UN ANÁLISIS DE ALTERNATIVAS EN PROFUNDIDAD

En el presente capítulo se dará a conocer la investigación desarrollada en 2 capas dentro de la arquitectura de la universidad, primero la capa de Aplicación y luego la capa de Negocios.

7.1 Capa De Aplicación

Si existe algo que no podemos negar hoy en día, es el crecimiento y protagonismo que las Aplicaciones móviles han tomado en el mundo de las nuevas tecnologías. Esto lleva a que la tecnología también se adapte a ello creando nuevas soluciones o herramientas.

Vista la necesidad de las empresas e instituciones de cubrir la mayor parte del público objetivo, la implementación de aplicaciones multiplataforma se considera una opción atractiva tomando como objetivo la reducción de tiempos y costos. Considerando la dificultad de desarrollar aplicaciones nativas para las múltiples plataformas móviles se realizará un análisis experimental de desarrollo de aplicaciones móviles multiplataforma

A nivel tecnológico podemos observar el nacimiento de nuevos Framework y herramientas orientados al desarrollo de aplicaciones móviles multiplataforma, entre ellos se encuentran Xamarin quien entrega un desarrollo de forma nativa utilizando un SDK nativo, y Ionic que cuenta con un desarrollo híbrido utilizando como base Córdoba⁴, estas tecnologías serán investigadas en profundidad en el presente capítulo.

La figura N° 4 muestra las distintas tecnologías a ser investigadas dentro de este capítulo con las distintas alternativas de desarrollo.

⁴ Apache Córdoba es un Framework de desarrollo móvil de código abierto que permite usar tecnologías web estándar como HTML5, CSS3 y JavaScript para el desarrollo multiplataforma

Desarrollo Nativo Desarrollo Híbrido



Figura N° 4: Tecnologías De Desarrollo a Investigar

7.1.1 Xamarin

El siguiente apartado intenta explicar con mayor detalle que es y cómo funciona Xamarin a la hora de compilar una aplicación móvil, basados en la documentación oficial que Xamarin mantiene en su página web (Xamarin Inc, 2017).

7.1.1.1 ¿Qué es Xamarin?

Xamarin es una plataforma de desarrollo que nos permite llevar a cabo el desarrollo multiplataforma a través de un enfoque único en su tipo, este enfoque hace referencia a que se puede tener proyectos orientados a cada una de las plataformas con un proyecto que comparte el código entre ellos, todo codificado dentro del lenguaje C#, este enfoque único permite tener un código base compartido en C#, un acceso 100% nativo a las API's de cada plataforma y un rendimiento alto.

Las aplicaciones que se desarrollan en Xamarin no son aplicaciones a las que se le agrega una capa adicional en tiempo de ejecución para que funcionen, sino que son aplicaciones 100% nativas, cada proyecto que se desarrolla en Xamarin compila y despliega una aplicación por cada plataforma. Es decir que cualquier cosa que se desarrolle en Objective-C, Java, o Swift puede desarrollarse en C#.

Los creadores de Xamarin no solo idearon una forma de poder compartir la lógica de negocios, además permite compartir código relativo a la interfaz de usuario ya sea a través

de C# o XAML teniendo algunas prestaciones de Xamarin Tradicional, a esta característica de código de interfaz de usuario compartido se le conoce como Xamarin.forms que es un conjunto de librerías que permite escribir aplicaciones nativas para iOS, WatchOS, tvOS, Android, Andorid Wear, Mac y las plataformas de Windows. Con Xamarin se puede utilizar la herramienta “Silo” que es aprovechar todas las herramientas nativas que nos ofrece cada plataforma. Esto permite desarrollar nativamente, tomando ventaja de todas las herramientas que ofrece cada dispositivo. Xamarin es algo parecido a las aplicaciones híbridas, con la excepción que se utiliza un solo lenguaje y entorno de desarrollo, en este caso puede ser Visual Studio o Xamarin Studio.

Esta herramienta hace mucho énfasis en el uso de un solo lenguaje ya que esto implica el poder compartir código.

Zamora en su artículo “Xamarin, la API para crear aplicaciones multiplataforma en C#/.NET” nos da la siguiente definición de este Framework

(Zamora, 2014) *“Con Xamarin podríamos evitar tener que utilizar Java para desarrollar una aplicación para Android u Objective C o Swift para desarrollar en Apple.”*

“Xamarin nos permite generar nuestra aplicación para iOS (.APP) y para Android (.APK), la cual correrá de forma nativa en el dispositivo móvil. Al poder desarrollar aplicaciones nativas para Windows Phone, Windows 8 y web en C# y .NET, resulta que la reutilización del código la estamos exportando a aún más plataformas que iOS o Android. Según Xamarin, deberíamos poder reutilizar el 90% del código aproximadamente.”

En la búsqueda de una definición más fidedigna de lo que es Xamarin, los siguientes apartados están basados en la documentación oficial, específicamente en el artículo Introduction to Mobile Development–Xamarin (Xamarin Inc, 2017).

7.1.1.2 Introducción a Xamarin

Al considerar cómo construir aplicaciones iOS y Android, muchas personas piensan que los lenguajes nativos, Objective-C, Swift y Java, son la única opción. Sin embargo, en los últimos años, ha surgido un nuevo ambiente de plataformas para la construcción de aplicaciones móviles. Xamarin es único en este espacio al ofrecer un solo lenguaje - C #, bibliotecas de clases y tiempos de ejecución que funcionan en las tres plataformas móviles de iOS, Android y Windows Phone (el idioma nativo de Windows Phone ya es C #). Cada una de estas plataformas tiene un conjunto de características diferentes y cada una varía en su capacidad para escribir aplicaciones nativas, es decir, aplicaciones que compilan código nativo y que interactúan fluidamente con el subsistema Java subyacente. Por ejemplo, algunas plataformas sólo permiten que las aplicaciones se construyan en HTML y JavaScript, mientras que algunas son de muy bajo nivel y solo permiten el código C / C ++. Algunas plataformas ni siquiera utilizan el juego de herramientas de control nativo.

7.1.1.3 ¿Cómo funciona Xamarin?

Xamarin ofrece dos productos comerciales: Xamarin.iOS y Xamarin.Android. Ambos están contruidos sobre Mono (Software Libre capaz de ejecutar aplicaciones .NET), una versión de código abierto del .NET Framework basada en los estándares publicados de .NET ECMA. Mono ha existido casi tanto tiempo como el propio .NET Framework, y funciona en casi todas las plataformas imaginables incluyendo Linux, Unix, FreeBSD y Mac OS X.

7.1.1.3.1 *Xamarin.iOS*

Xamarin para iOS compila directamente a código ensamblador ARM nativo, utilizando Ahead Of Time (AOT). Para ejemplificar de mejor forma la forma de compilar que nos entrega Xamarin esquematizaremos este proceso en la Figura N° 5.

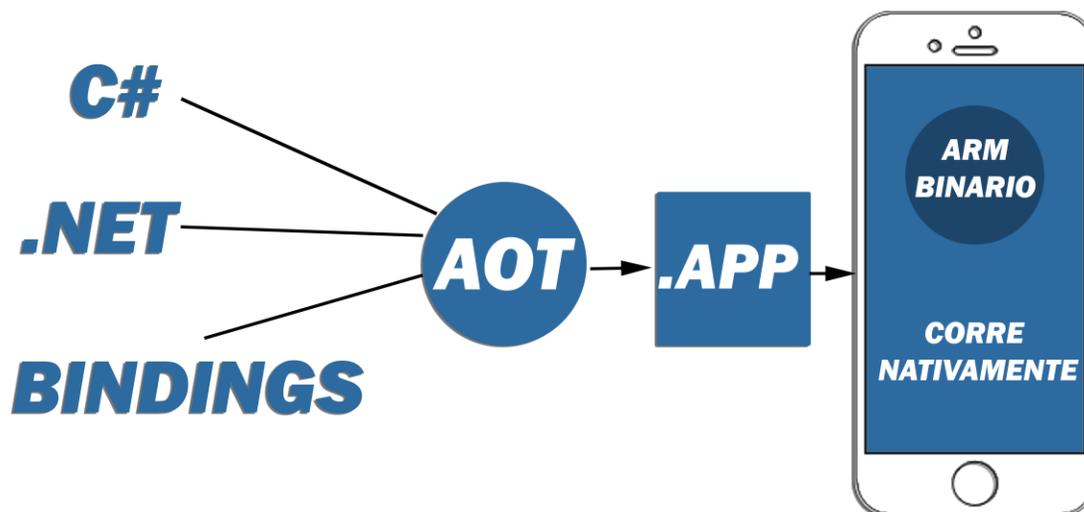


Figura N° 5: Compilación Xamarin.IOS

Como se puede apreciar en la Figura anterior Xamarin toma el código desarrollado en C#, las bibliotecas .NET y todos los Bindings que necesita, y los ejecuta mediante AOT, que significa que en lugar de esperar hasta que el código sea ejecutado, toma el código inicial (por lo general el código de bytes) y lo compila para el dispositivo que se está ejecutando.

7.1.1.3.2 Xamarin.Android

En Android, el compilador de Xamarin compila a Lenguaje Intermedio (IL), que es Just-in-Time (JIT) compilado para el montaje nativo cuando se inicia la aplicación. Para ejemplificar esta práctica tenemos la Figura N° 6

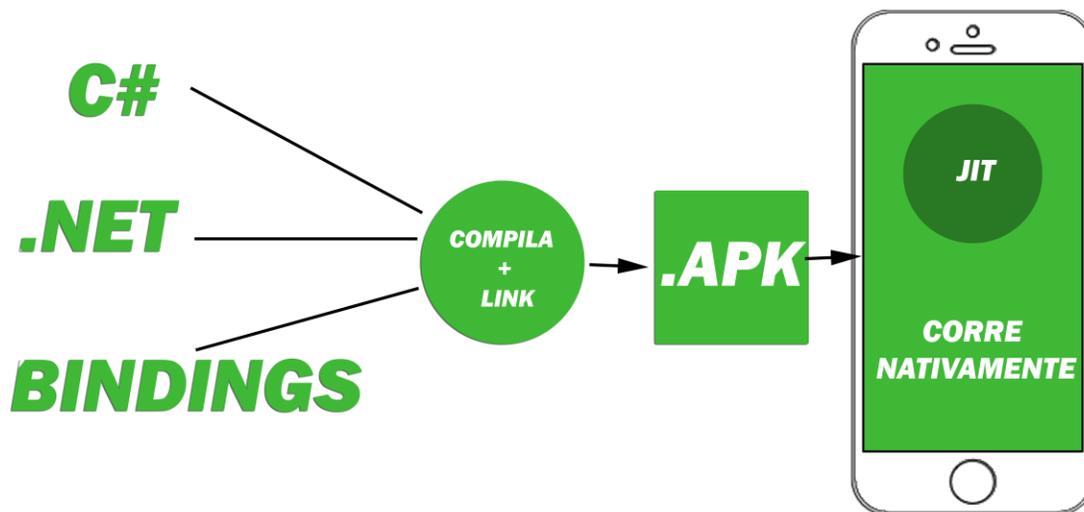


Figura N° 6: Compilación Xamarin.Android

De la Figura anterior y la igual que en el caso de Xamarin.iOS, se encapsula el código C#, .NET y los Bindings necesarios y se empaquetan en una APK, para luego y una vez instalada se ejecute de manera nativa por el método JIT.

Esta opción es una especie de híbrido entre un lenguaje compilado e interpretado. Y es cómo trabajan la mayoría de entornos de ejecución de Java. La idea detrás de esto es simple, en lugar de compilar todo de antemano, se compila sólo lo que es necesario para la tarea en cuestión.

En ambos casos, las aplicaciones Xamarin utilizan un tiempo de ejecución que maneja automáticamente cosas como asignación de memoria, recolección de basura, interoperabilidad de plataforma subyacente, etc.

7.1.1.4 Arquitectura de Xamarin

Xamarin nos da la posibilidad de tener una base de código compartido que contiene entidades de negocio, lógica de negocio, acceso a servicios, etc., y simplemente tener código distinto para los detalles de interfaz de usuario para la capa de cliente en cada plataforma.

Básicamente es desarrollar al mismo tiempo para todas las plataformas, por lo que minimizamos tiempos de desarrollo.

La Figura N°7 representa la arquitectura que tiene Xamarin para el desarrollo multiplataforma

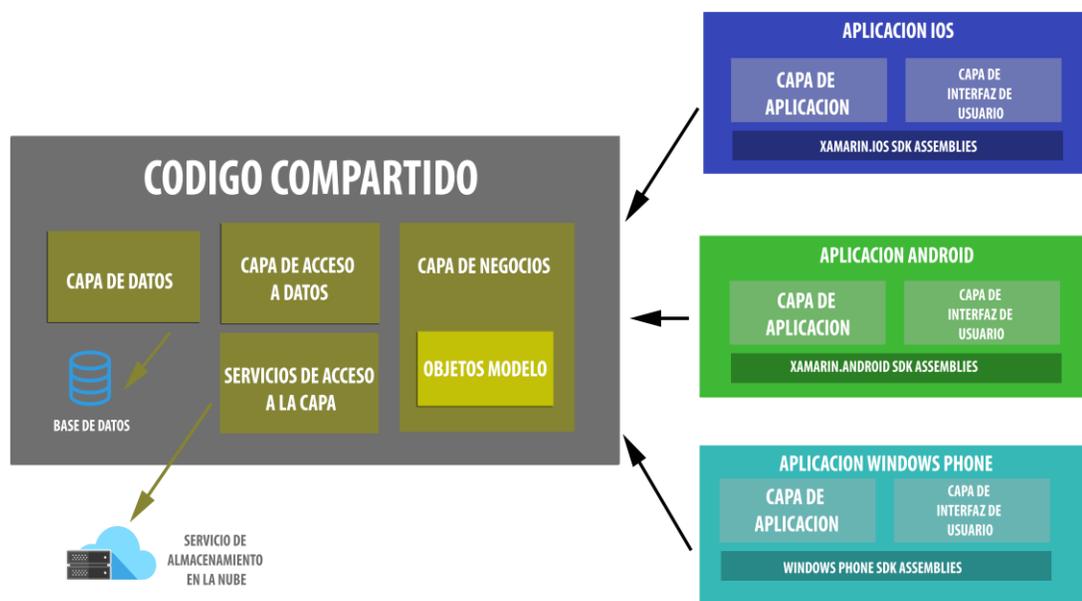


Figura N° 7: Arquitectura Xamarin

De la Figura N°7 podemos interpretar que tenemos una librería compartida que contiene los componentes de las capas de negocio, servicios y datos. Por otro lado, tenemos distintas aplicaciones para cada tipo de cliente, que utilizan las librerías y componentes nativos de la interfaz de usuario de cada plataforma. Las aplicaciones cliente acceden al código compartido para interactuar con las capas de negocio, servicios y datos.

Xamarin se encarga de crear aplicaciones específicas para cada plataforma por ejemplo utilizando Xamarin.iOS y Xamarin.Android, dependiendo del caso. Esto permite aprovechar

los elementos de UI⁵ específicos de cada plataforma y construir así interfaces de usuario avanzadas y personalizadas para cada una.

Con la aparición de Xamarin.Forms cambio mucho la forma de afrontar el diseño de las aplicaciones y se puede tener una capa de interfaz común para diseñar las interfaces multiplataforma. La Figura N° 8 nos da una idea de la arquitectura de la aplicación con Xamarin.Forms



Figura N° 8: Arquitectura Xamarin.Forms

Como lo muestra la Figura anterior Xamarin.Forms es una serie de componentes que permite definir interfaces de usuario para distintas plataformas desde una misma base de código, está orientado a construir aplicaciones con interfaces de usuario sencillas, donde es más importante la capacidad de compartir código que el brindar interfaces de usuario avanzadas y personalizadas para cada plataforma, sin embargo, existe la posibilidad de hacer la personalización de los componentes a cada plataforma heredando el componente básico de Xamarin.Forms a uno que se implementara en el proyecto de iOS y Android, donde se usan instrucciones típicas de Objective-C y Java respectivamente.

⁵ Interfaz de usuario

7.1.1.5 ¿Por qué usar Xamarin?

Con Xamarin se pueden crear aplicaciones móviles una vez y desplegarlas en las tres plataformas más importantes del mercado, iOS, Android y Windows Phone, todo esto en el lenguaje de programación C#, es decir no es necesario desarrollar en Java para Android u Objective-C para iOS, con tan solo tener conocimientos en C#, se pueden desarrollar aplicaciones multiplataforma, es decir tener una aplicación en iOS, Android, y Windows Phone. Se ejemplificaran los beneficios de Xamarin en los siguientes puntos:

- **Reutilización De Código**

Xamarin da mucho énfasis en la reutilización de código, un ejemplo de cómo se estructura un proyecto dentro de Xamarin está representado por la Figura N° 9

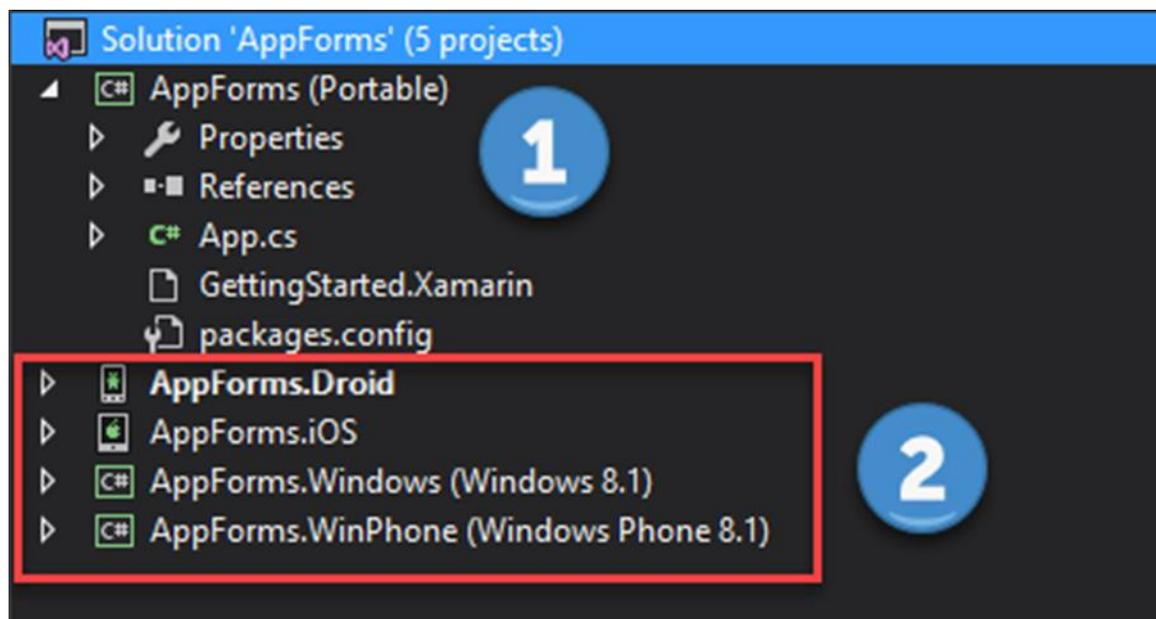


Figura N° 9: Proyecto Xamarin.Forms

De la Figura anterior, se observa como al crear un proyecto en Xamarin, el punto numero 1 es el apartado de código común que referenciará cada una de las plataformas, y el punto número 2 es la posibilidad de modificar el código específico para cada plataforma.

• Experiencia Nativa

Xamarin.Forms proporciona una interfaz de usuario nativa, lo que significa que el código de interfaz escrito será traducido automáticamente al lenguaje de la plataforma en la que se desplegará la aplicación, proporcionando acceso a todo el hardware del móvil. La Figura N° 10 es un ejemplo de cómo trabaja Xamarin.Forms a la hora de desplegar un único código C# en las distintos tipos de Smartphone.



Figura N° 10: Xamarin.Forms Desplegada en Diferentes Sistemas

Como se aprecia en la figura anterior se codifica una interfaz en XAML⁶ ligada a código C#, y el mismo es interpretado de forma nativa en cada una de las plataformas.

• Acceso Total A La Api De Cada Plataforma

Las plataformas de desarrollo nativo están con constantes actualizaciones y Xamarin dispone de tantas librerías de terceros como el propio código nativo. Por lo tanto, se ven eliminadas todas las limitaciones que se pudieran tener en cuanto a actualizaciones del sistema operativo nativo.

⁶ Lenguaje de formato para la interfaz de usuario.

• **Máxima Rentabilidad**

Con Xamarin se tiene la ventaja de no tener que contratar diferentes equipos de desarrolladores que dominen diferentes lenguajes de programación para cada plataforma. Un único equipo es válido para desarrollar en todas las plataformas con un resultado nativo y una gran experiencia de usuario.

7.1.2 Ionic

El presente apartado tiene como fin explicar con mayor exactitud como Ionic utiliza lenguajes web para crear una aplicación híbrida basados en córdoba, todo esto basado en el Manual De Introducción a Ionic (Gallego, 2016).

7.1.2.1 ¿Qué es Ionic?

Ionic es un framework de código abierto construido usando HTML5, CSS3 y JavaScript para el desarrollo de aplicaciones híbridas para dispositivos móviles. La librería está orientada únicamente a su visualización en dispositivos móviles, es decir, no es una librería responsiva tipo Bootstrap⁷ que se adapte a distintos tamaños de pantalla. Todo esto hace que los componentes tengan una apariencia más similar a la nativa, que funcionen más rápido, y por lo tanto que mejore la experiencia del usuario que use la aplicación.

Ionic incluye una completa librería de componentes, estilos y animaciones que simulan el aspecto nativo de las distintas plataformas. Estos componentes adoptarán automáticamente la apariencia nativa del dispositivo en el que se visualicen. Por ejemplo, si incluimos un checkbox en un formulario, al compilar y visualizar dicha aplicación en Android, el checkbox adoptará un aspecto distinto al que mostraría en iOS.

Una de las características interesantes de Ionic es que utiliza Angular para el desarrollo del código dinámico de la aplicación. Ionic nos permite crear una aplicación sin necesidad de escribir ni una línea de JavaScript ni de Angular, solamente con CSS y HTML. Pero para aplicaciones más complejas podemos usar Angular, lo que nos permitirá una mayor potencia a la hora de crear contenidos dinámicos, para por ejemplo mostrar un listado a partir de

⁷ Conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web.

datos cargados de Internet o almacenados en el móvil. Además Angular nos permite estructurar el código de la aplicación y modularizar las distintas partes del mismo siguiendo patrones de desarrollo como MVC, donde tendríamos el código separado en modelos de datos, vistas y controladores.

Las aplicaciones híbridas son básicamente páginas web que se ejecutan o renderizan dentro de un navegador web de una aplicación nativa. Es decir, el desarrollador crea la aplicación usando código web (como HTML, CSS, JavaScript, etc.) y el framework Ionic en este caso. Una vez finalizado el desarrollo, este código se copiará dentro de una aplicación nativa, que consistirá en una única pantalla de tipo visor web en la que se cargará el código de la página web. Esta técnica nos permite generar a partir de código web una aplicación compilada como si fuera nativa, que se puede instalar en cualquier dispositivo y que la podemos publicar en el market de aplicaciones.

Además, las aplicaciones híbridas tienen muchas otras ventajas, pero la principal es la velocidad de desarrollo, ya que con un único código web podemos generar aplicaciones para todas las plataformas, Android, iOS, Windows Phone, etc.

7.1.2.2 ¿Cómo Funciona Ionic?

Ionic viene integrado con Cordova, que en términos simples es el encargado de unir la aplicación web y la aplicación nativa. Además permite el acceso a las características nativas de los dispositivos. Es decir, desde el código web podremos hacer uso de los sensores del dispositivo como GPS, cámara, acelerómetros, brújula, etc.

La Figura N° 11 es una representación de cómo funciona Ionic

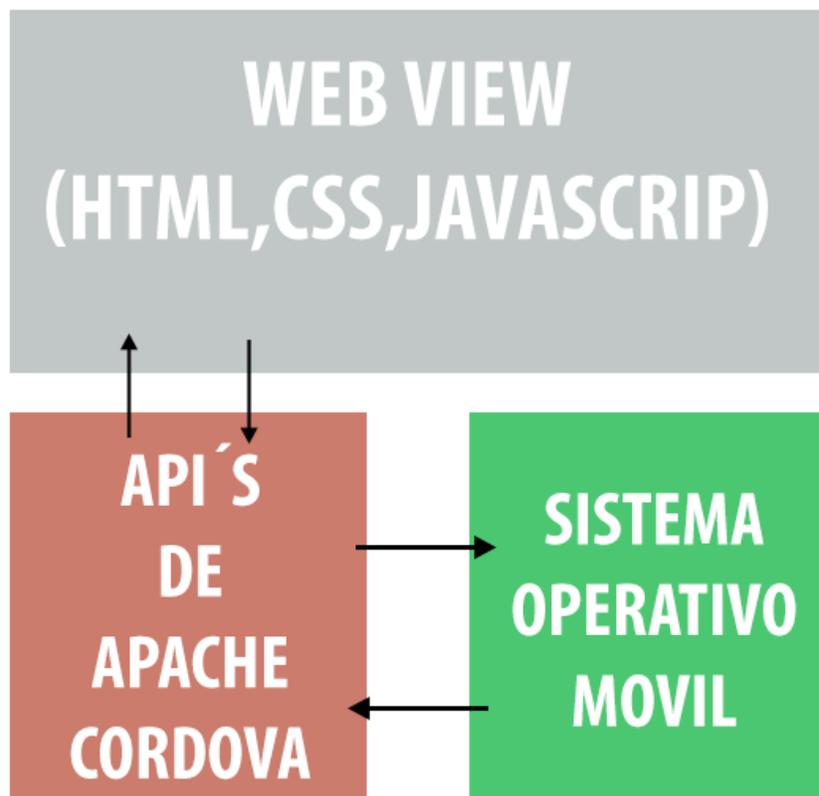


Figura N° 11: Bases del funcionamiento de Ionic

Como se muestra en la Figura N° 11 se tiene un Web View en la que se ejecuta el código HTML, CSS y JavaScript, y utilizando las API's de Cordova se puede tener acceso a funciones nativas del dispositivo.

7.1.2.3 Arquitectura de una aplicación con Ionic

Ionic, al estar basado en Angular, utiliza el patrón de diseño conocido como Vista-Controlador (View-Controller). En este tipo de patrón se separa los datos, la lógica y las interfaces de usuario. Como su nombre indica, está separado en tres componentes: Modelo, Vista y Controlador. Está basado en la ideología de separación de conceptos y cumple perfectamente con los objetivos de los patrones de diseño.

El Modelo es la capa encargada de los datos, es decir, la que se encarga de hacer peticiones a las bases de datos para enviar o recibir información. Estas bases de datos pueden estar alojadas de forma local en nuestra app o de forma remota en un servidor externo.

La Vista es el código que nos permite presentar los datos que el modelo proporciona, como ejemplo podemos decir que en una aplicación es el código HTML quien nos permite mostrar la salida de los datos procesados.

El Controlador es la capa que sirve de enlace entre la vista y el modelo. Envía comandos al modelo para actualizar su estado, y a la vista correspondiente para cambiar su presentación.

En la Figura N°12 se puede ver un esquema de la arquitectura completa que sigue Ionic y Angular

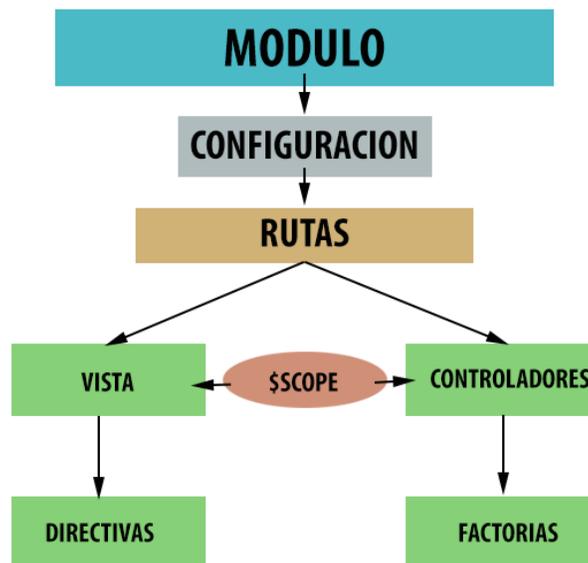


Figura N° 12: Arquitectura De Ionic

En la arquitectura de una aplicación intervienen muchos tipos de componentes además de las Vistas y Controladores, conforme se avancen en el documento se verán cada uno de ellos poco a poco, por ahora quienes principalmente nos interesan son las Vistas, Controladores, Servicios o Factorias, y la Configuración y Rutas.

El área de cada uno de estos componentes en una aplicación con Ionic es el siguiente:

Los controladores obtienen los datos de uno o varios Servicios o Factorías y lo envían a una vista o plantilla a través de la variable \$scope.

Las vistas o plantillas contienen la descripción visual de una pantalla (o de una parte de una pantalla) y obtienen los datos a mostrar de la variable \$scope.

La configuración y las rutas de la aplicación permiten enlazar los controladores con las vistas o plantillas correspondientes.

Las directivas permiten crear y usar componentes con aspecto y comportamiento personalizado.

7.1.2.4 ¿Por qué utilizar Ionic?

- **Utilidad Multiplataforma**

Ionic es compatible en varias plataformas y puede ser optimizado para varios sistemas operativos móviles. Desarrollar código es mucho más fácil y rápido ya que Ionic se integra fácilmente con AngularJS⁸ para configurar la estructura del código. Esto lleva al desarrollo de nuevos conceptos, más rápido y con presupuestos más bajos.

- **Interfaz de usuario Amigable**

Ionic incorpora conceptos clave como Java Scripts y componentes CSS para optimizar interfaces fáciles de usar en cualquier aplicación móvil. Componentes como pestañas de navegación, menú deslizable, botones, listas, entradas de formularios, ventanas emergentes y avisos son muy simples y elegantes y fácilmente personalizables para una aplicación móvil

⁸ Es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

concreta. Ionic, por lo tanto, crea interfaces de usuario atractivas e interactivas en un período de tiempo notablemente corto.

• **Framework de Código Abierto**

Dado que Ionic es de código abierto, los desarrolladores pueden personalizar distintos diseños en varios sistemas operativos móviles sin incurrir en grandes costos. Ionic da códigos de componentes CSS, JS y HTML que reducen la necesidad de reescribirlo para un nuevo sistema operativo móvil. La fácil integración de Ionic con Angular JS facilita la creación de una estructura de código más fácil y mejor.

Desarrollar aplicaciones en Windows, Android e iOS es así más rápido y fácil de hacer. La creación de aplicaciones pasan rápidamente del desarrollo al mercado y esto aumenta la comercialización y crea conciencia y popularidad de las aplicaciones. Ionic ayuda a ahorrar dinero, tiempo y esfuerzos.

• **Basado en Angular**

Ionic se basa en la compatibilidad con Angular JS (Patrón de diseño Modelo-vista-controlador) que luego amplía la facilidad y funcionalidad del framework de Angular para hacer la creación de aplicaciones móviles muy fáciles. A partir de hoy, Angular JS nos muestra el marco favorito de JavaScript en uso, que cuenta con el respaldo de Google.

• **Desarrollo Cross Mobile**

Ionic se destaca en el desarrollo de aplicaciones por ser eficiente, utilizando herramientas estándar con una única base de codificación que ahorra dinero, tiempo y esfuerzo y proporciona una apariencia integrada.

8 IMPLEMENTACION Y PRUEBA DE TECNOLOGIAS EN LA CAPA DE APLICACION

En el presente capítulo se presentará la comparación de las tecnologías investigadas anteriormente, referentes a la capa de aplicación, con el fin de obtener aplicaciones comparables. Esto ayudara a definir distintos tipos de pruebas que serán implementadas más adelante.

Para realizar una comparación entre las distintas formas de desarrollar un software móvil, se implementaran 3 soluciones basadas en Android, primero con un desarrollo nativo en Android Studio, en segundo lugar con Xamarin quien nos entrega una versión para cada una de las plataformas (IOS y Andorid) y por ultimo con Ionic que es un Framework de desarrollo basado en HTML, CSS y JavaScript.

Para esto se implementará un “Inicio de Sesión” simple en cada una de las plataformas anteriormente mencionadas y con el fin de realizar pruebas comparables todas realizarán el acceso a un servidor dentro de la universidad, y con el objetivo de evaluar el comportamiento de cada de una de ellas se consumirá un servicio Rest que entrega las asignaturas con sus notas y las asignaturas con sus horarios.

Algunos de los puntos a comprar son sus pesos una vez transformados en APK's, comparar los tiempos de respuesta, y su comportamiento dentro de los dispositivos en los que se desplegaran; Adicionalmente y utilizando la aplicación “Data Monitor” se obtendrá el consumo de Datos de cada una de las aplicaciones por cada una de las consultas que se realizaran.

Para el desarrollo de estas aplicaciones se ejemplificará sus casos de uso y modelados de datos una sola vez, ya que se desarrolla la misma problemática en diferentes plataformas.

8.1 Características de despliegue

Con el fin de realizar pruebas comparables, el sistema operativo en el que se despliegan las aplicaciones que fueron desarrolladas están especificadas en la tabla N° 3

Marca	Motorola
Modelo	Moto G5 Plus
Producto	Smartphone
Sistema Operativo	Android 7.0 Nougat
Procesador	Qualcomm Snapdragon 625 octa-core de 2.0 GHz
Tamaño de Pantalla	5.2'
Memoria Interna	32 GB
WI-FI	Sí. 802.11 a/b/g/n
Conectividad 4G	4G LTE (Cat 6)
GPS	Sí

Tabla N° 3: Características Del Smartphone donde se desplegarán las soluciones

8.2 Modelamiento Del Problema

8.2.1 Caso de uso

La figura N° 15 representa las funciones que realizan las aplicaciones, con consultas proporcionadas por la Dirección de Informática.

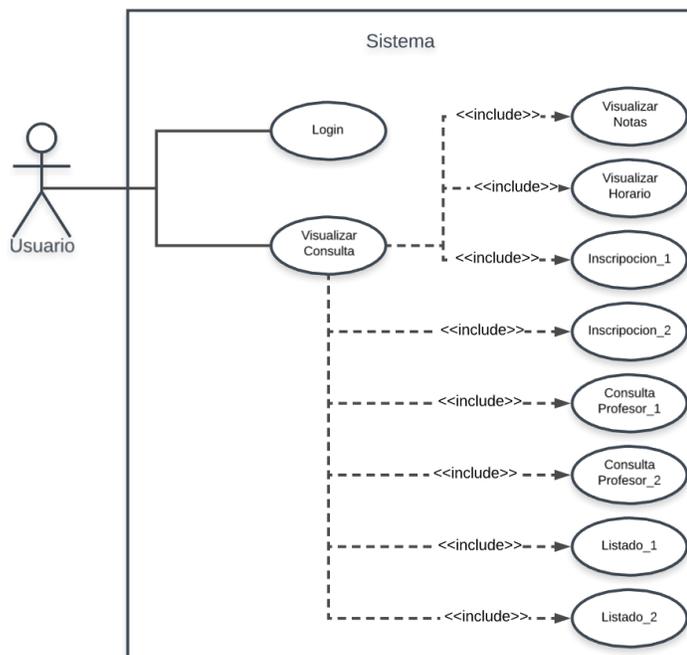


Figura N° 13: Casos de Uso

8.2.1.1 Descripción de actores

Actor: USUARIO.

Rol: Puede tener acceso a la aplicación y acceder a las opciones que ésta entrega.

Nivel de conocimiento técnico requerido: Conocimientos a nivel de usuario en aplicaciones funcionales de oficina y en específico navegación web y móvil.

8.2.1.2 Especificación de caso de uso

- **Caso de Uso:** <Login>
- **Actores:** Usuario
- **Descripción:** Se Inicia la aplicación y lanza el formulario de inicio de sesión
- **Pre-Condiciones:** El usuario debe tener instalada y abierta la aplicación móvil
- **Flujo de Eventos Básicos:**

Al actor	El sistema
1.- Este caso de uso comienza cuando el usuario ingresa al sistema (Aplicación móvil) a través de un dispositivo móvil.	2.- El sistema muestra el formulario para el inicio de sesión
3.- El usuario ingresa los datos, Usuario y Contraseña, en el formulario y presiona el botón “Iniciar Sesión”.	4.- El sistema Valida que los datos ingresados serán correctos
	5.- El sistema Muestra la pantalla principal para el usuario

Flujo de Eventos Alternativo:

Al actor	El sistema
3(a) El Usuario Ingresa datos de inicio de Sesión erróneos	4(a) El sistema Valida que los datos ingresados están correctos.
	4(b) El sistema no verifica los datos de la cuenta.
	4(c) El Sistema notifican que los datos para el inicio de sesión son erróneos y solicita su ingreso nuevamente.

Tabla N° 4: Especificación Caso de Uso Login

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Caso de Uso <Visualizar Notas>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Notas, donde se despliegan los ramos con sus respectivas notas y ponderaciones.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de notas dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos con sus respectivos, nombre, código, ponderaciones y notas.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Notas no ingresadas al sistema	2.-) La aplicación no despliega los ramos ni las notas con sus ponderaciones
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 5: Especificación Caso Notas

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Visualizar Horario>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Horarios, donde se despliegan los ramos con sus respectivos horarios y salas donde se impartirá.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Horario dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos con sus respectivos horarios y número de sala donde se impartirá el ramo.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Horario y Salas no ingresadas al sistema.	2.-) La aplicación no despliega la lista de ramo con sus salas ni horarios
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 6: Especificación Caso Horario

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Inscripcion_1>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Inscripción_1, donde se despliegan los ramos inscritos por un alumno durante el segundo semestre del 2015.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Inscripcion_1 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de ramos inscritos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 7: Especificación caso de uso Inscripcion_1

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Inscripcion_2>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Inscripción_2, donde se despliegan los ramos inscritos por un alumno durante el primer semestre del 2016,

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Inscripcion_1 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de ramos inscritos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 8: Especificación caso de uso Inscripcion_2

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Consulta Profesor_1>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Consulta profesor_1, donde se despliegan los ramos impartidos por un profesor durante el año 2016.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Consulta Profesor_1 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos que el profesor impartió durante el 2016.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de ramos impartidos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 9: Especificación caso de uso Consulta Profesor_1

- **Post Condición:** El usuario podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Consulta Profesor_2>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Consulta Profesor_2, donde se despliegan los ramos impartidos por un profesor durante el año 2015.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Consulta Profesor_2 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos que el profesor impartió durante el 2015.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de ramos impartidos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 10: Especificación caso de uso Consulta Profesor_2

- **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

- **Casos de Uso <Listado_1>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Listado_1, donde se despliegan un listado con los datos de los alumnos inscritos en un ramo.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Listado_1 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de alumnos inscritos en un ramo con sus respectivos datos.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de alumnos inscritos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 11: Especificación caso de uso listado_1

- **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta.

- **Casos de Uso <Listado_2>**

- **Actores:** Usuario

- **Descripción:** El Usuario ingresa a la sección de Listado_2, donde se despliegan un listado con los datos de los alumnos inscritos en un ramo.

- **Precondiciones:** Haber iniciado sesión y estar en el menú principal

- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el usuario selecciona la opción de Listado_1 dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de alumnos inscritos en un ramo con sus respectivos datos.
	3.-) La aplicación móvil permite al usuario volver al menú principal para poder visualizar las demás opciones.
4.-) El usuario puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del usuario.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Consulta retorna un valor vacío.	2.-) La aplicación no despliega la lista de alumnos inscritos.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

Tabla N° 12: Especificación caso de uso Consulta listado_2

- **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

8.2.2 Diseño Arquitectura



Figura N° 14: Diseño De Arquitectura

8.2.2.1 Justificación Diseño Arquitectónico

En diseño arquitectónico se identifican los subsistemas integrados en nuestro proyecto, además se identifican detalladamente y se estructuran, permitiendo ver de manera clara las relaciones y el control que ejercen cada una de las partes del sistema, además de mostrar cómo se relacionan entre sí. Gracias a esto podemos localizar los puntos críticos de nuestro sistema, por ejemplo cuando se produce una caída de sistema, también los subsistemas que requieran una mayor seguridad, etc.

En nuestro sistema se utilizó la Arquitectura de 3 Capas, ya que nuestro sistema realiza la totalidad de sus funcionalidades en un servidor web, por consiguiente, como se debe acceder desde cualquier Smartphone con la aplicación instalada, es necesario realizar esta separación entre el cliente y servidor.

8.2.3 Diseño: Interfaz y Navegación

8.2.3.1 Wireframes

Plantilla <Inicio de Sesión>

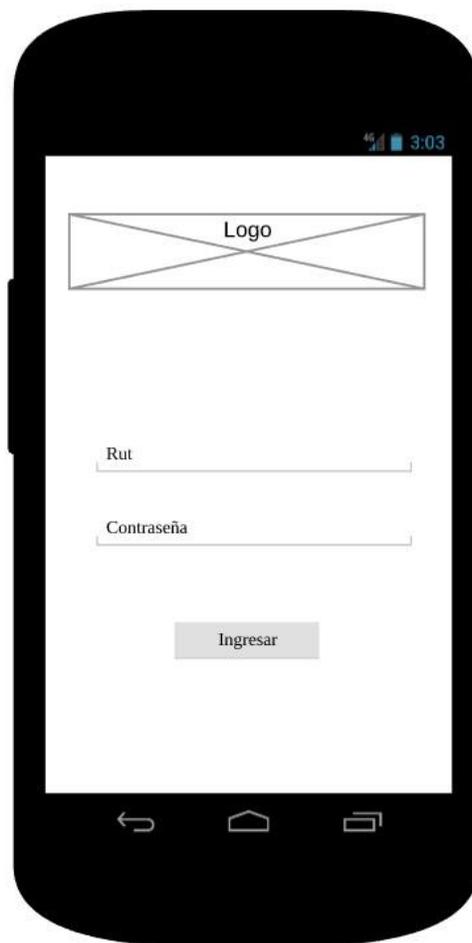


Figura N° 15: Wireframe

Plantilla <Menú Principal>

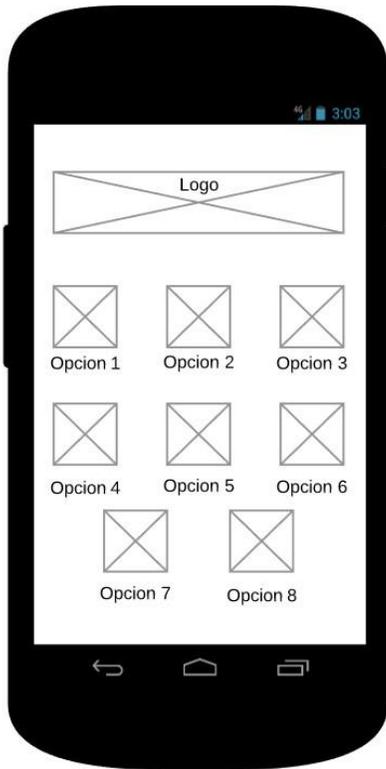


Figura N° 17: Wireframe

Plantilla <Asignaturas-Horario>

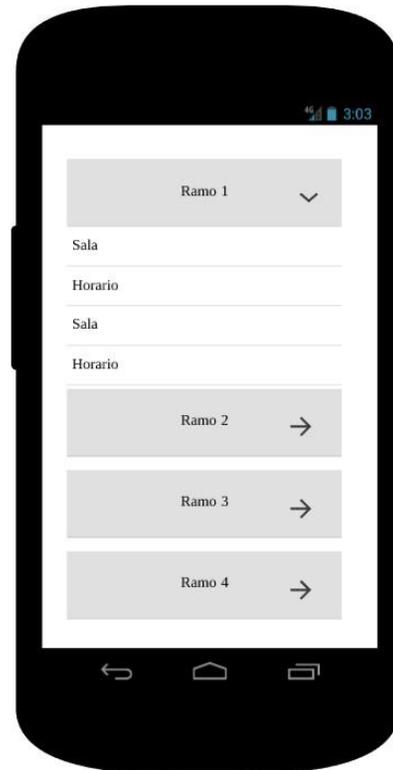


Figura N° 16: Wireframe

Plantilla <Asignaturas-Notas>



Figura N° 18: Wireframe

Plantilla <Consulta Inscripcion_1>

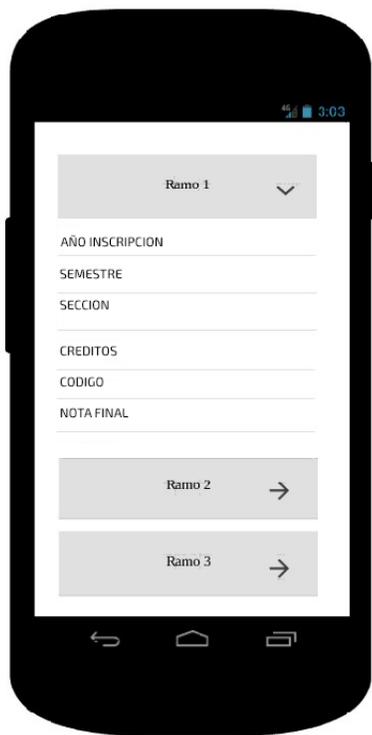


Figura N° 20: Wireframe

Plantilla <Consulta Inscripcion_2>

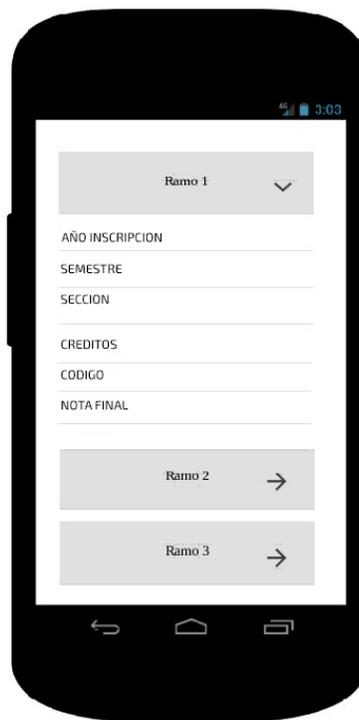


Figura N° 19: Wireframe

Plantilla <Consulta Profesor_1>

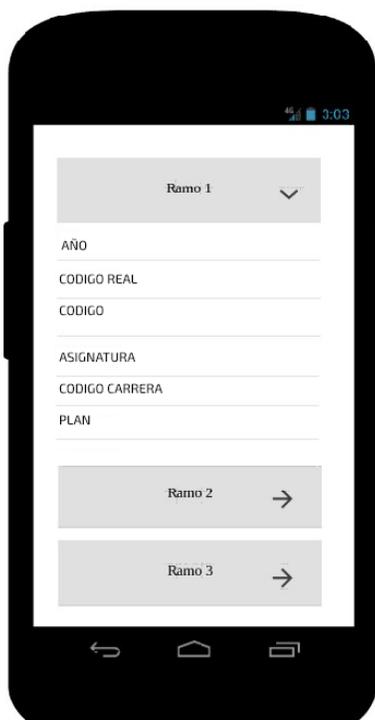


Figura N° 22: Wireframe

Plantilla <Consulta Profesor_2>

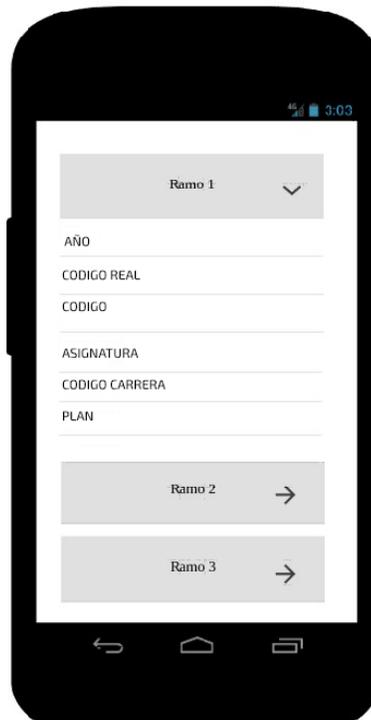


Figura N° 21: Wireframe

Plantilla <Consulta Lista_1>

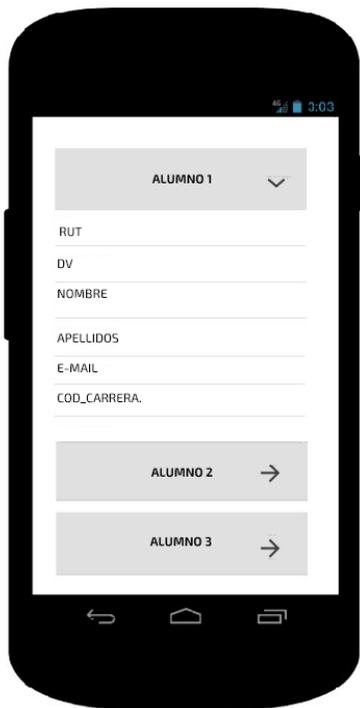


Figura N° 23: Wireframe

Plantilla <Consulta Lista_2>

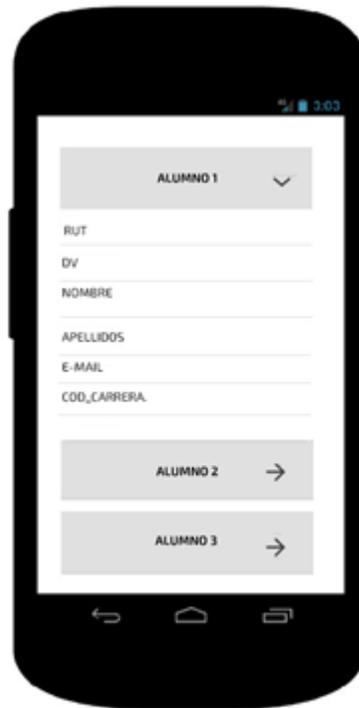


Figura N° 24: Wireframe

8.2.4 Especificación de módulos

N° Modulo: 1.0		Nombre Modulo: Login	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Rut_Alumno	Varchar(250)	Confirmación	Boolean
Contraseña_Alumno	Varchar(250)		

Tabla N° 13: Especificación de Modulo

N° Modulo: 2.0		Nombre Modulo: Visualizar Horarios	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	ramo_alumno	Varchar(250)
		horario_ramo	Varchar(250)
		sala_ramo	Varchar(250)

Tabla N° 14: Especificación de Modulo

N° Modulo: 4.0		Nombre Modulo: Visualizar Notas	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	ramo_alumno	Varchar(250)
		nota_ramo	Varchar(250)
		ponderacion_ramo	Varchar(250)

Tabla N° 15: Especificación de Modulo

N° Modulo: 5.0		Nombre Modulo: Visualizar Inscripción _1	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	cod_asignatura	Varchar(250)
		mat_ano_academico	Varchar(250)
		mat_periodo_inscripcion	Varchar(250)
		seccion	Varchar(250)
		creditos	Varchar(250)
		nom_asignatura	Varchar(250)
		nota_final	Varchar(250)
		estado	Varchar(250)
		origen	Varchar(250)

Tabla N° 16: Especificación de Modulo

N° Modulo: 6.0		Nombre Modulo: Visualizar Inscripción _2	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	cod_asignatura	Varchar(250)
		mat_ano_academico	Varchar(250)
		mat_periodo_inscripcion	Varchar(250)
		seccion	Varchar(250)
		creditos	Varchar(250)
		nom_asignatura	Varchar(250)
		nota_final	Varchar(250)

		estado	Varchar(250)
		origen	Varchar(250)

Tabla N° 17: Especificación de Modulo

N° Modulo: 7.0		Nombre Modulo: Visualizar Consulta Profesor_1	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	plataforma	Varchar(250)
		SEC_ANO	Varchar(250)
		codigo_real	Varchar(250)
		codigo	Varchar(250)
		nombre_asignatura	Varchar(250)
		seccion	Varchar(250)
		nombre_carrera	Varchar(250)
		codigo_carrera	Varchar(250)
		plan_carrera	Varchar(250)
		ano_carrera	Varchar(250)
		periodo_carrera	Varchar(250)
		dia	Varchar(250)
		hora_inicio	Varchar(250)
		hora_termino	Varchar(250)

Tabla N° 18: Especificación de Modulo

N° Modulo: 8.0		Nombre Modulo: Visualizar Consulta Profesor_2	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Nombre	Varchar(250)
		plataforma	Varchar(250)
		SEC_ANO	Varchar(250)
		codigo_real	Varchar(250)
		codigo	Varchar(250)

		nombre_asignatura	Varchar(250)
		seccion	Varchar(250)
		nombre_carrera	Varchar(250)
		codigo_carrera	Varchar(250)
		plan_carrera	Varchar(250)
		ano_carrera	Varchar(250)
		periodo_carrera	Varchar(250)
		dia	Varchar(250)
		hora_inicio	Varchar(250)
		hora_termino	Varchar(250)

Tabla N° 19: Especificación de Modulo

N° Modulo: 9.0		Nombre Modulo: Visualizar Consulta Listado_1	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	rut	Varchar(250)
		dv	Varchar(250)
		nombre	Varchar(250)
		apellido_paterno	Varchar(250)
		apellido_materno	Varchar(250)
		email	Varchar(250)
		cod_carrera	Varchar(250)
		plan_carrera	Varchar(250)
		ind_competencia	Varchar(250)
		nom_carrera	Varchar(250)
		seccion	Varchar(250)
		mat_ano_academico	Varchar(250)
		mat_periodo_inscripcion	Varchar(250)
		alc_periodo	Varchar(250)
		mca_mca_nro_ano	Varchar(250)
		mca_mca_nro_periodo	Varchar(250)
		nota_final	Varchar(250)
		promocion	Varchar(250)

		critério1	Varchar(250)
		critério2	Varchar(250)
		eat_codigo	Varchar(250)

Tabla N° 20: Especificación de Modulo

N° Modulo: 10.0		Nombre Modulo: Visualizar Consulta Listado_2	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	rut	Varchar(250)
		dv	Varchar(250)
		nombre	Varchar(250)
		apellido_paterno	Varchar(250)
		apellido_materno	Varchar(250)
		email	Varchar(250)
		cod_carrera	Varchar(250)
		plan_carrera	Varchar(250)
		ind_competencia	Varchar(250)
		nom_carrera	Varchar(250)
		seccion	Varchar(250)
		mat_ano_academico	Varchar(250)
		mat_periodo_inscripcion	Varchar(250)
		alc_periodo	Varchar(250)
		mca_mca_nro_ano	Varchar(250)
		mca_mca_nro_periodo	Varchar(250)
		nota_final	Varchar(250)
		promocion	Varchar(250)
		critério1	Varchar(250)
		critério2	Varchar(250)
		eat_codigo	Varchar(250)

Tabla N° 21: Especificación de Modulo

8.3 Aplicación Nativa Android

Una aplicación nativa está desarrollada y optimizada específicamente para el sistema operativo determinado y la plataforma de desarrollo del fabricante (Android, iOS, Windows Phone, etc).

Este tipo de aplicaciones se adapta por completo a las funcionalidades y características del dispositivo obteniendo así una mejor experiencia de uso.

Para este caso en específico se desarrolló una aplicación llamada “Login Rest” la que sin ser utilizada entrega los datos especificados en la Figura N°27.

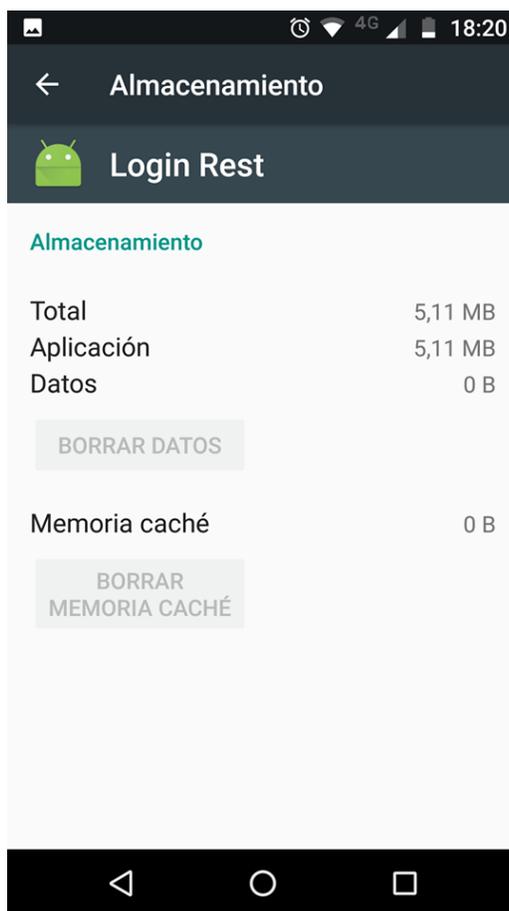


Figura N° 25: Aplicación Nativa Android Sin Uso

De la figura anterior se obtienen los datos expresados en la Tabla N° 22, “Login Rest” presenta dentro del SmarthPhone un peso de 5.11 MB sin uso alguno.

APLICACIÓN RECIÉN INSTALADA	
APP	5.11 MB.
DATA	0 MB.
CACHE	0 MB.
Total	5.11 MB.

Tabla N° 22: Datos Aplicación Nativa Recién Instalada

8.3.1 Pruebas

Luego de presentar las características de aplicación en Android Nativo se procede a realizar las pruebas de las consultas con acceso al servidor, con el fin de obtener una muestra fidedigna cada una de las pruebas será realizada seis veces se expondrán las capturas de la muestras con mayores y menores índices de almacenamiento.

8.3.1.1 Pruebas Tiempos De Apertura

Las pruebas de Tiempos de apertura tienen como fin determinar cuan rápida es la aplicación a la hora de cargar los componentes que necesita para su correcto y total funcionamiento. La Tabla N°23 muestra los datos obtenido luego de 6 pruebas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Apertura	2 S.					

Tabla N° 23: Pruebas Tiempos de Apertura

8.3.1.2 Pruebas Login

Luego de 6 Pruebas con el Login las Figuras 28 y 29 presentan las pruebas con más altos índices con respecto a la memoria utilizada.

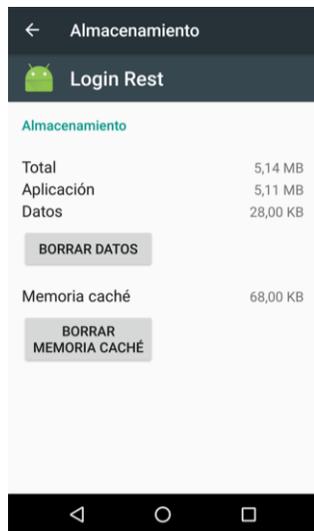


Figura N° 27: Login Menor Almacenamiento



Figura N° 26: Login Mayor Almacenamiento

Y las figuras 30 y 31 Muestran el mayor y menor consumo hecho por la consulta login al servidor.



Figura N° 29 Login Menor Consumo



Figura N° 28: Login Menor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 24

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Login_data	28 KB					
Login_cache	68 KB	76 KB	74 KB	68 KB	70 KB	72 KB
Login_Total	5,14 MB					
Login_Datos Moviles	1,2 KB	1,3 KB	1,2 KB	1,2 KB	1,2 KB	1,2 KB

Tabla N° 24 Pruebas Login

Además del análisis con respecto al peso y el consumo de datos se debe tomar en cuenta el tiempo de reacción de la aplicación a la hora de realizar las consultas, la Tabla N° 25 muestra los tiempos de respuesta que se obtuvieron con cada una de las pruebas realizadas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Login	1 S.	1 S.	1,5 S.	1 S.	1 S.	1,5 S.

Tabla N° 25: Tiempos de respuesta Login

8.3.1.3 Pruebas Consulta Horarios

Luego de 6 pruebas utilizando la consulta de horarios las Figuras 32 y 33 muestran los datos de almacenamiento utilizados por la aplicación.

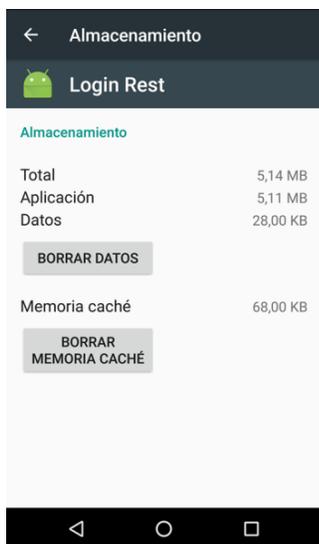


Figura N° 30: Horarios Menor Almacenamiento

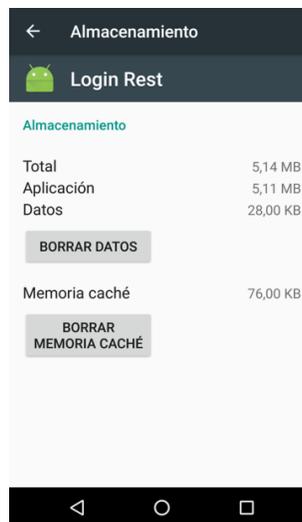


Figura N° 31: Horarios Mayor Almacenamiento

Las Figuras 34 y 35 Muestran el mayor y menor consumo hecho por la consulta Horarios al servidor.



Figura N° 33: Horarios Menor Consumo



Figura N° 32 Horarios mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 26

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Horario_data	28 KB					
Horario_cache	72 KB	70 KB	74 KB	68 KB	76 KB	72 KB
Horario_Total	5,14 MB					
Horario_Datos Moviles	1,6 KB	1,6 KB	1,6 KB	1,2 KB	1,7 KB	1,2 KB

Tabla N° 26: Pruebas Consulta horarios

Luego de presentar los resultados de las pruebas de almacenamiento se procede a exponer las pruebas con respecto a los tiempos de respuesta que tuvo la aplicación en la consulta de Horarios, estos se muestran en la Tabla N° 27

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Horarios	1 S.					

Tabla N° 27: Tiempos de respuesta Horarios

8.3.1.4 Pruebas Consulta Notas

Luego de 6 pruebas utilizando la consulta de Notas las Figuras 36 y 37 muestran los datos de almacenamiento utilizados por la aplicación.

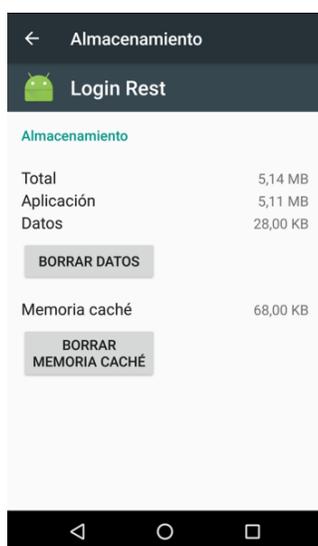


Figura N° 35: Notas Menor Almacenamiento

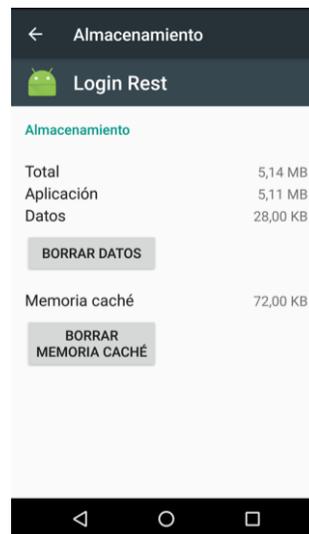


Figura N° 34: Notas Mayor Almacenamiento

Las Figuras 38 y 39 Muestran el mayor y menor consumo hecho por la consulta Notas al servidor.



Figura N° 37: Notas menor Consumo



Figura N° 36: Notas Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 28

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Notas_data	28 KB					
Notas_cache	70 KB	70 KB	72 KB	70 KB	68 KB	72 KB
Notas_Total	5,14 MB					
Notas_Datos Móviles	1,7 KB	1,6 KB	1,6 KB	1,3 KB	1,3 KB	1,6 KB

Tabla N° 28: Pruebas Consulta Notas

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 29

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Notas	1 S.	1 S.	1 S.	1,5 S.	1 S.	1 S.

Tabla N° 29: Tiempo Respuestas Pruebas Notas

8.3.1.5 Pruebas Consulta Inscripcion_1

Luego de 6 pruebas utilizando la consulta de Inscripcion_1 las Figuras 40 y 41 muestran los datos de almacenamiento utilizados por la aplicación.

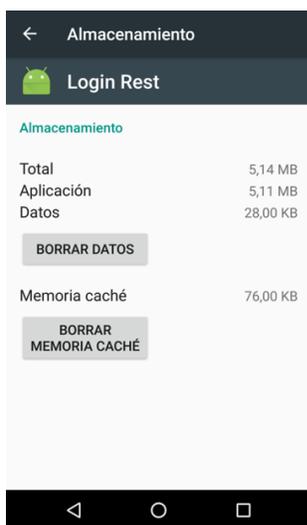


Figura N° 39: Inscripcion_1 Menor Peso



Figura N° 38: Inscripcion_1 Mayor Peso

Las Figuras 42 y 43 Muestran el mayor y menor consumo hecho por la consulta Inscripcion_1 al servidor.



Figura N° 41: Inscripcion_1 Menor Consumo



Figura N° 40: Inscripcion_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 30

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Inscripcion_1_data	28 KB					
Inscripcion_1_cache	76 KB	76 KB	84 KB	76 KB	82 KB	80 KB
Inscripcion_1_Total	5,14 MB					
Inscripcion_1_Datos Móviles	1,6 KB	1,6 KB	2,5 KB	1,6 KB	2,5 KB	2,1 KB

Tabla N° 30: Pruebas Consulta Inscripcion_1

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 31

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_1	1 S.	1,5 S.	1 S.	1 S.	1,5 S.	1 S.

Tabla N° 31: Tiempo Respuestas Pruebas Inscripcion_1

8.3.1.6 Pruebas Consulta Inscripcion_2

Luego de 6 pruebas utilizando la consulta de Inscripcion_2 las Figuras 44 y 45 muestran los datos de almacenamiento utilizados por la aplicación.

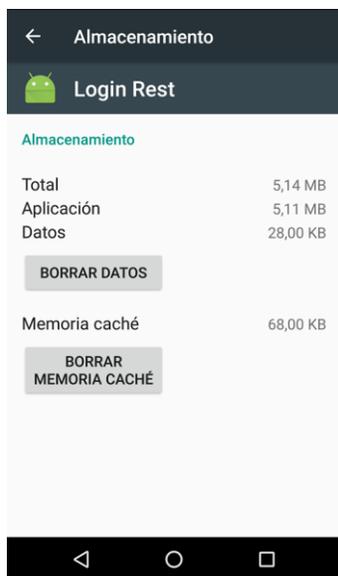


Figura N° 43: Inscripcion_2 Menor Peso

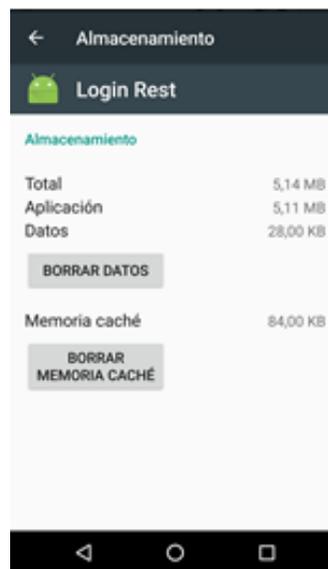


Figura N° 42: Inscripcion_2 Mayor Peso

Las Figuras 46 y 47 Muestran el mayor y menor consumo hecho por la consulta Inscripcion_2 al servidor.



Figura N° 44: Inscripcion_2 Menor Consumo



Figura N° 45: Inscripcion_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 32

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Inscripcion_2_data	28 KB					
Inscripcion_2_cache	68 KB	70 KB	70 KB	70 KB	70 KB	84 KB
Inscripcion_2_Total	5,14 MB					
Inscripcion_2_Datos	1,7 KB	1,7 KB	1,6 KB	1,6 KB	1,6 KB	1,6 KB
Moviles						

Tabla N° 32: Pruebas Consulta Inscripcion_2

Una vez mostrados los datos de almacenamiento, los datos de tiempos de respuestas se muestran en la Tabla N° 32.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1,5 S.	1,5 S.	1 S.	1,5 S.	1 S.

Tabla N° 33: Tiempo Respuestas Pruebas Inscripcion_2

8.3.1.7 Pruebas Consulta Profesor_1

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_1 las Figuras 48 y 49 muestran los datos de almacenamiento utilizados por la aplicación.

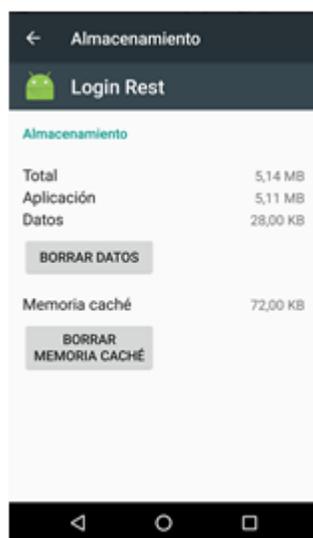


Figura N° 46: Profesor_1 Menor Peso

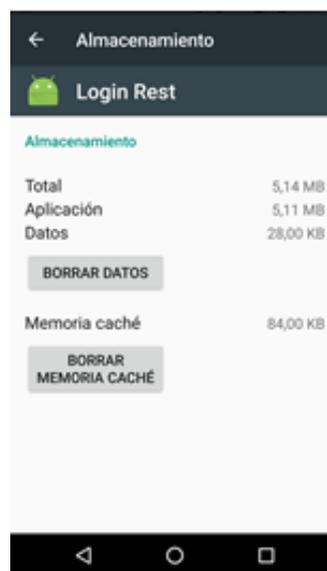


Figura N° 47: Profesor_1 mayor peso

Las Figuras 50 y 51 Muestran el mayor y menor consumo hecho por la consulta Profesor_1 al servidor.



Figura N° 49: Profesor_1 menor Consumo



Figura N° 48 Profesor_1 mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 34.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_1_data	28 KB					
Profesor_1_cache	72 KB	84 KB				
Profesor_1_Total	5,14 MB					
Profesor_1_Datos Móviles	1,3 KB	1,5 KB	1,6 KB	1,5 KB	1,5 KB	1,5 KB

Tabla N° 34: Pruebas Consulta Profesor_1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 35.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.					

Tabla N° 35: Tiempo Respuestas Pruebas Profesor_1

8.3.1.8 Pruebas Consulta Profesor_2

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_2 las Figuras 50 y 51 muestran los datos de almacenamiento utilizados por la aplicación.

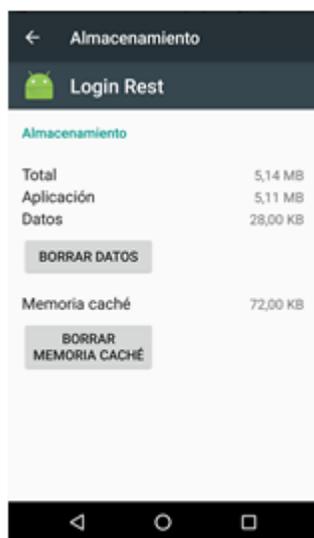


Figura N° 51: Profesor_2 Menor Peso

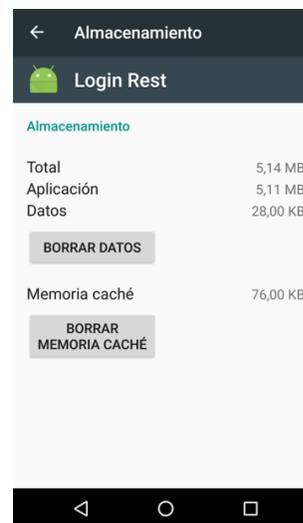


Figura N° 50: Profesor_2 Mayor Peso

Las Figuras 52 y 53 Muestran el mayor y menor consumo hecho por la consulta Profesor_2 al servidor.



**Figura N° 52: Profesor_2
menor Consumo**



**Figura N° 53: Profesor_2
Mayor Consumo**

Los resultados de las pruebas están expresados en la Tabla N° 36.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_2_data	28 KB					
Profesor_2_cache	72 KB	72 KB	72 KB	76 KB	76 KB	76 KB
Profesor_2_Total	5,14MB	5,14 MB				
Profesor_2_Datos Móviles	1,6 KB	1,6 KB	1,8 KB	1,2 KB	2,3 KB	1,6 KB

Tabla N° 36: Pruebas Consulta Profesor _2

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 37.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1,5 S.	1 S.	1 S.	1,5 S.	1 S.	1 S.

Tabla N° 37: Tiempo Respuestas Pruebas Profesor_2

8.3.1.9 Pruebas Consulta Listado_1

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_1 las Figuras 54 y 55 muestran los datos de almacenamiento utilizados por la aplicación.

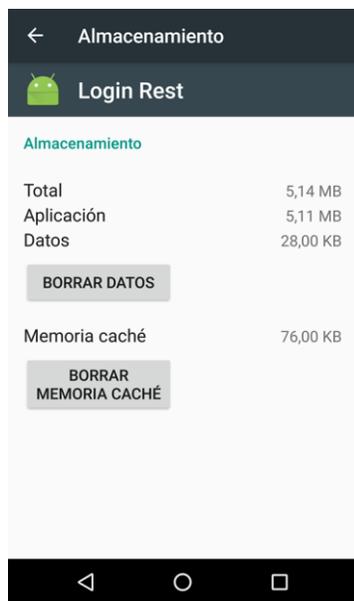


Figura N° 55: Listado_1 Menor Peso

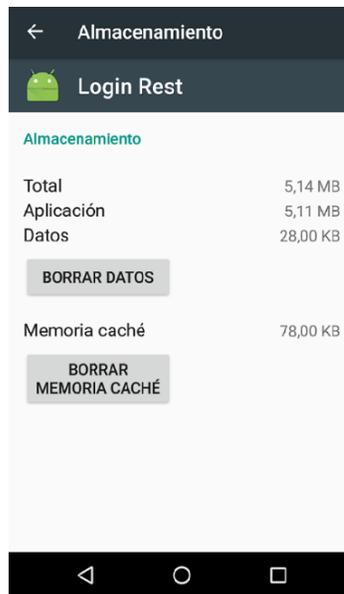


Figura N° 54: Listado_1 Mayor Peso

Las Figuras 57 y 56 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 57: Listado_1 Menor Consumo

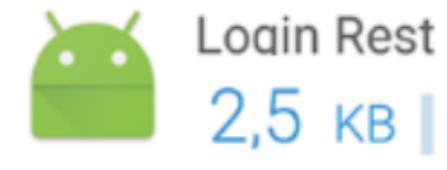


Figura N° 56: Listado_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 38.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_1_data	28 KB					
Lista_1_cache	78 KB	76 KB	78 KB	76 KB	78 KB	76 KB
Lista_1_Total	5,14 MB					
Lista_1_Datos Moviles	2,2 KB	1,9 KB	2,1 KB	2,2 KB	2,3 KB	2,5 KB

Tabla N° 38: Pruebas Consulta Listado _1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 39.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_1	2 S.	1,5 S.	2 S.	1,5 S.	2 S.	1,5 S.

Tabla N° 39: Tiempo Respuestas Pruebas Listado_1

8.3.1.10 Pruebas Consulta Listado_2

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_2 las Figuras 60 y 61 muestran los datos de almacenamiento utilizados por la aplicación.

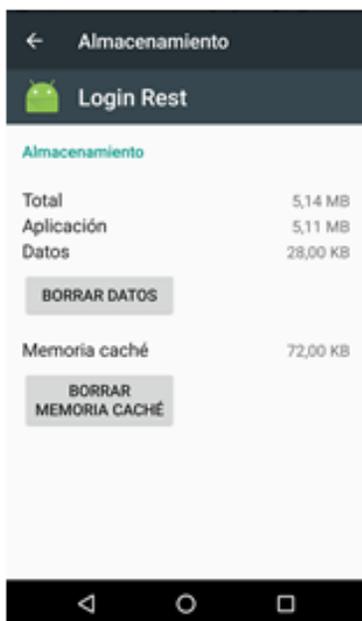


Figura N° 59: Listado_2 Menor Peso

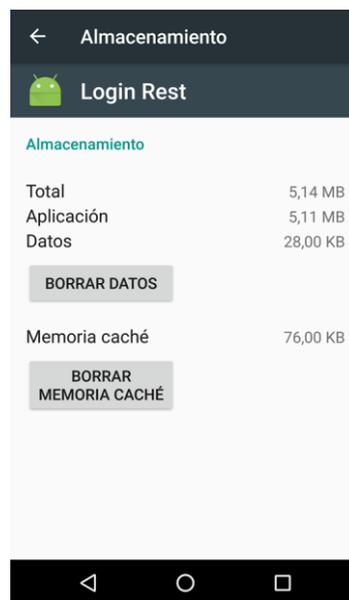


Figura N° 58: Listado_2 Menor Peso

Las Figuras 60 y 61 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 60: Listado_2 Menor Consumo



Figura N° 61: Listado_2 Menor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 40.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_2_data	28 KB					
Lista_2_cache	72 KB	76 KB	72 KB	76 KB	76 KB	76 KB
Lista_2_Total	5,14 MB					
Lista_2_Datos Moviles	2,3 KB	2,4 KB	2,5 KB	2,4 KB	2,4 KB	2,5 KB

Tabla N° 40: Pruebas Consulta Listado_1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 41.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_2	1,5 S.	2 S.	1,5 S.	2 S.	1,5 S.	2 S.

Tabla N° 41 : Tiempo Respuestas Pruebas Listado_1

8.4 Aplicación en Xamarin .Forms

Xamarin es una herramienta para los desarrolladores de aplicaciones móviles, y la novedad de esta herramienta, es la opción de escribir su aplicación en lenguaje C# y el mismo código será traducido para ejecutarse en iOS, Android y Windows Phone.

Para la presente investigación se desarrolla una aplicación llamada “Login_MIUBB.Andorid” en Visual Studio utilizando las herramientas que nos entrega Xamarin.

Al instalar la aplicación obtenemos los datos que se muestran en la Figura N° 62.

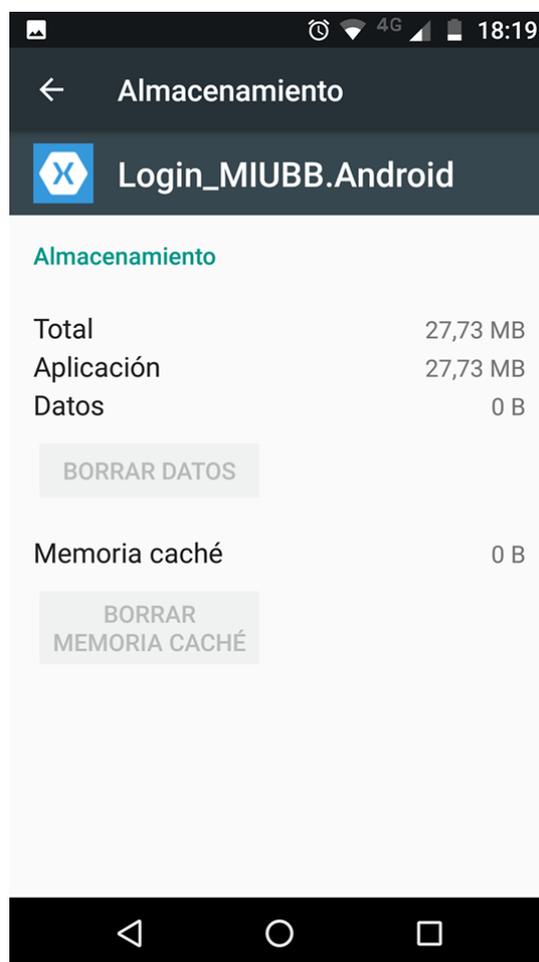


Figura N° 62: Aplicación Xamarin Sin Uso

De la figura anterior se obtienen los datos expresados en la Tabla N° 42, “Login_MIUBB.Andorid” presenta dentro del SmarthPhone un peso de 5.11 MB sin uso alguno.

APLICACIÓN RECIÉN INSTALADA	
APP	27,73 MB.
DATA	0 MB.
CACHE	0 MB.
Total	27,73 MB.

Tabla N° 42: Datos Aplicación Xamarin Recién Instalada

8.4.1 Pruebas

Luego de presentar las características de aplicación en Xamarin se realizan las pruebas de las consultas con acceso al servidor, con el fin de obtener una muestra fidedigna cada una de las pruebas será realizada seis veces, se expondrán las capturas de la muestras con mayor y menor índice de almacenamiento.

8.4.1.1 Pruebas Tiempos De Apertura

Las pruebas de Tiempos de apertura tienen como fin determinar cuan rápida es la aplicación a la hora de cargar los componentes que necesita para su correcto y total funcionamiento. La Tabla N°43 muestra los datos obtenido luego de 6 pruebas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Apertura	3 S.	3 S.	3 S.	3,5 S.	3 S.	3 S.

Tabla N° 43: Pruebas Tiempos de Apertura

8.4.1.2 Pruebas Login

Luego de 6 Pruebas con el Login las Figuras 66 y 67 presentan las pruebas con más altos índices con respecto a la memoria utilizada.

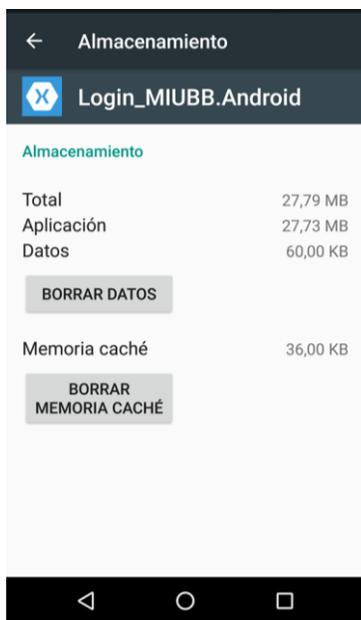


Figura N° 64: Login Menor Almacenamiento

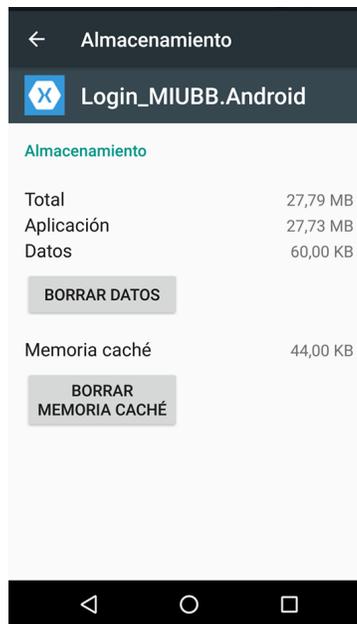


Figura N° 63: Login Mayor Almacenamiento

Y las figuras 67 y 68 muestran el mayor y menor consumo hecho por la consulta login al servidor.

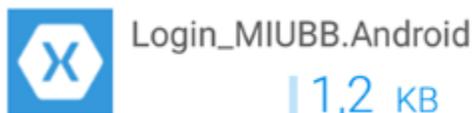


Figura N° 65 Login Menor Consumo



Figura N° 66: Login Menor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 44.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Login_data	60 KB					
Login_cache	44 KB	36KB	40 KB	36 KB	44KB	44 KB
Login_Total	27,79 MB					
Login_Datos Moviles	1,3 KB	1,2 KB	1,2 KB	1,3 KB	1,2 KB	1,2 KB

Tabla N° 44 Pruebas Login

Además del análisis con respecto al peso y el consumo de datos se debe tomar en cuenta el tiempo de reacción de la aplicación a la hora de realizar las consultas, la Tabla N° 45 muestra los tiempos de respuesta que se obtuvieron con cada una de las pruebas realizadas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Login	1 S.	1,5 S.	1 S.	1 S.	2 S.	1 S.

Tabla N° 45: Tiempos de respuesta Login

8.4.1.3 Pruebas Consulta Horarios

Luego de 6 pruebas utilizando la consulta de horarios las Figuras 69 y 70 muestran los datos de almacenamiento utilizados por la aplicación.

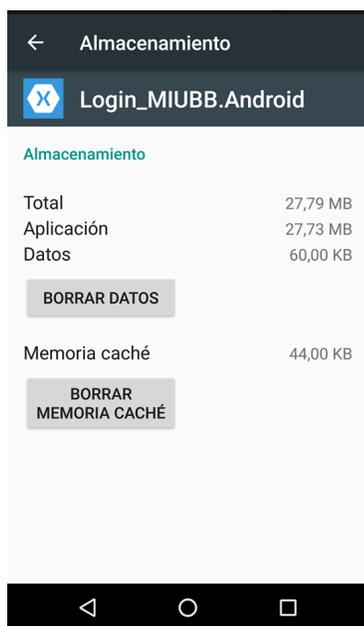


Figura N° 68: Horarios Menor Almacenamiento

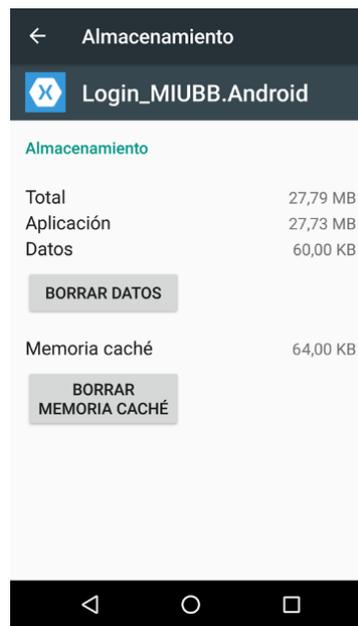


Figura N° 67: Horarios Mayor Almacenamiento

Las Figuras 34 y 35 Muestran el mayor y menor consumo hecho por la consulta Horarios al servidor.



Figura N° 70: Horarios Menor Consumo

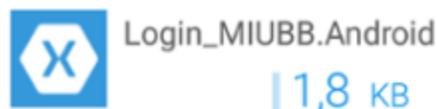


Figura N° 69 Horarios mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 46.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Horario_data	60 KB					
Horario_cache	44 KB	57 KB	40 KB	64 KB	44KB	44 KB
Horario_Total	27,79 MB					
Horario_Datos Moviles	1,5 KB	1,6 KB	1,8 KB	1,7 KB	1,6 KB	1,8 KB

Tabla N° 46: Pruebas Consulta horarios

Luego de presentar los resultados de las pruebas de almacenamiento se procede a exponer las pruebas con respecto a los tiempos de respuesta que tuvo la aplicación en la consulta de Horarios, estos se muestran en la Tabla N° 47.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Horarios	1 S.					

Tabla N° 47: Tiempos de respuesta Horarios

8.4.1.4 Pruebas Consulta Notas

Luego de 6 pruebas utilizando la consulta de Notas las Figuras 73 y 74 muestran los datos de almacenamiento utilizados por la aplicación.

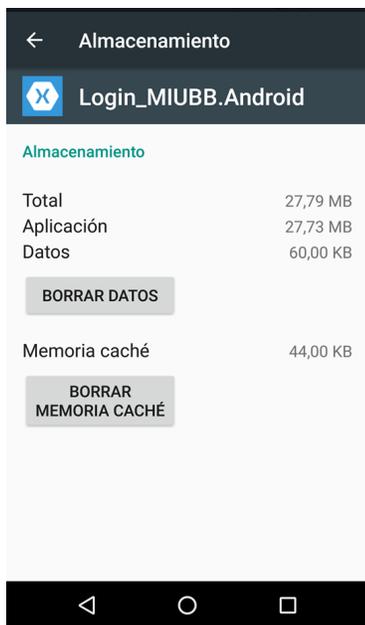


Figura N° 71: Notas Menor Almacenamiento

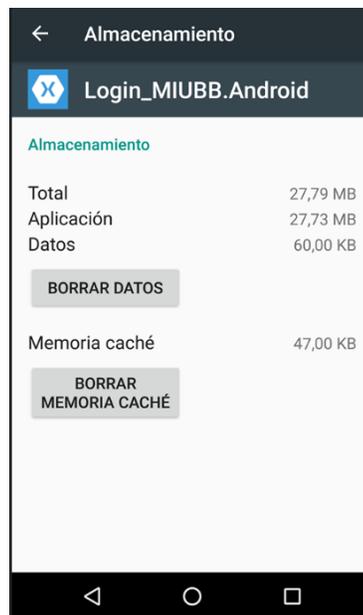


Figura N° 72: Notas Mayor Almacenamiento

Las Figuras 38 y 39 Muestran el mayor y menor consumo hecho por la consulta Notas al servidor.



Figura N° 74: Notas menor Consumo

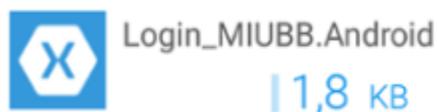


Figura N° 73: Notas Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 48

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Notas_data	60 KB					
Notas_cache	44 KB	44 KB	47 KB	44 KB	45 KB	44 KB
Notas_Total	27,79 MB					
Notas_Datos Móviles	1,6 KB	1,5 KB	1,7 KB	1,6 KB	1,8 KB	1,5 KB

Tabla N° 48: Pruebas Consulta Notas

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 49.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Notas	1 S.					

Tabla N° 49: Tiempo Respuestas Pruebas Notas

8.4.1.5 Pruebas Consulta Inscripcion_1

Luego de 6 pruebas utilizando la consulta de Inscripcion_1 las Figura 78 muestran los datos de almacenamiento utilizados por la aplicación en esta consulta en particular la memoria cache no varía en ninguna de las consultas realizadas.

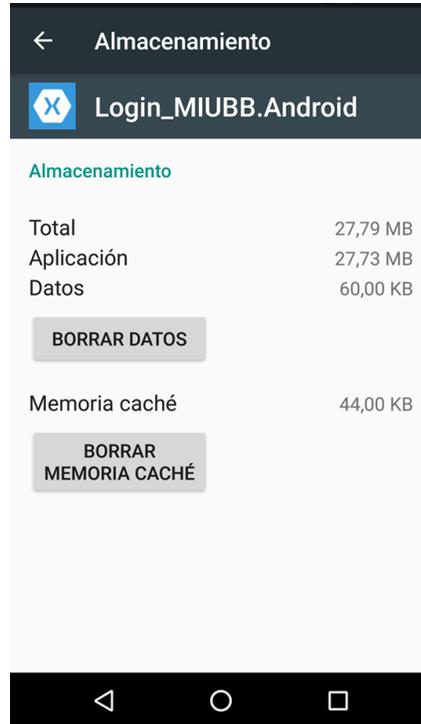


Figura N° 75: Inscripcion_1 Peso

Las Figuras 78 y 79 Muestran el mayor y menor consumo hecho por la consulta Incripcion_1 al servidor.



Figura N° 77: Incripcion_1 Menor Consumo



Figura N° 76: Incripcion_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 50.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Incripcion_1_data	60 KB					
Incripcion_1_cache	44 KB					
Incripcion_1_Total	27,79 MB					
Incripcion_1_Datos Móviles	2,3 KB	2,0 KB	2,1 KB	2,1 KB	2,1 KB	2,2 KB

Tabla N° 50: Pruebas Consulta Incripcion_1

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 51.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Incripcion_1	1 S.	1,5 S.	1 S.	1 S.	1,5 S.	1 S.

Tabla N° 51: Tiempo Respuestas Pruebas Incripcion_1

8.4.1.6 Pruebas Consulta Inscripcion_2

Luego de 6 pruebas utilizando la consulta de Inscripcion_2 las Figuras 80 y 81 muestran los datos de almacenamiento utilizados por la aplicación.

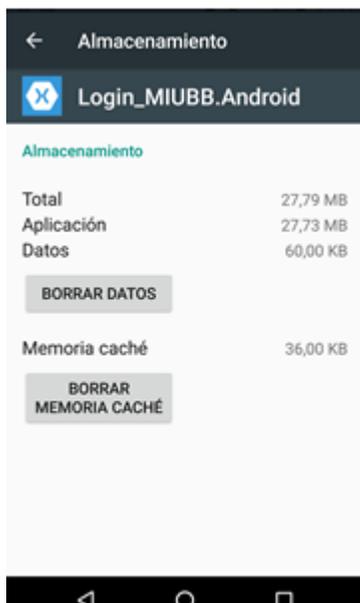


Figura N° 79: Inscripcion_2 Menor Peso

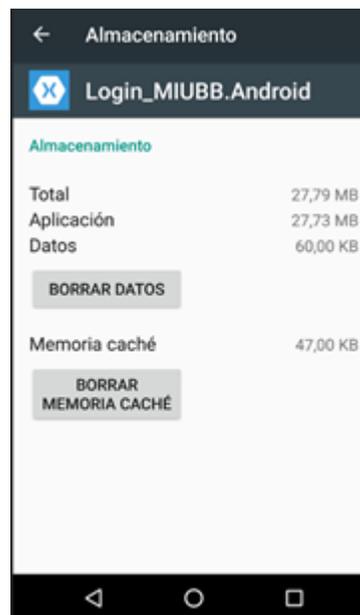


Figura N° 78: Inscripcion_2 Mayor Peso

Las Figuras 46 y 47 Muestran el mayor y menor consumo hecho por la consulta Inscripcion_2 al servidor.

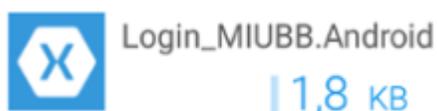


Figura N° 80: Inscripcion_2 Menor Consumo



Figura N° 81: Inscripcion_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 52.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Inscripcion_2_data	60 KB					
Inscripcion_2_cache	36 KB	44 KB	47 KB	44 KB	45 KB	44 KB
Inscripcion_2_Total	27,79 MB					
Inscripcion_2_Datos Moviles	1,9 KB	1,8 KB	1,9 KB	1,9 KB	1,8 KB	1,9 KB

Tabla N° 52: Pruebas Consulta Inscripcion_2

Una vez mostrados los datos de almacenamiento, los datos de tiempos de respuestas se muestran en la Tabla N° 53.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1 S.	1,5 S.	1 S.	1,5 S.	1 S.

Tabla N° 53: Tiempo Respuestas Pruebas Inscripcion_2

8.4.1.7 Pruebas Consulta Profesor_1

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_1 las Figuras 85 y 86 muestran los datos de almacenamiento utilizados por la aplicación.

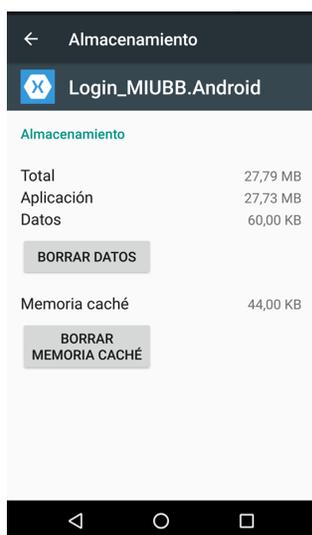


Figura N° 82: Profesor_1 Menor Peso

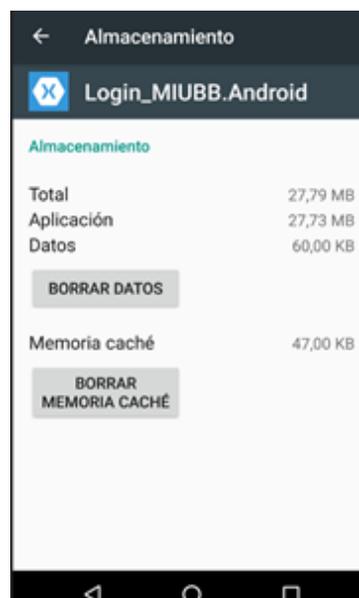


Figura N° 83: Profesor_1 mayor peso

Las Figuras 87 y 88 Muestran el mayor y menor consumo hecho por la consulta Profesor_1 al servidor.



Figura N° 85: Profesor_1 menor Consumo



Figura N° 84 Profesor_1 mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 54.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_1_data	60 KB					
Profesor_1_cache	44 KB	44 KB	47 KB	44 KB	45 KB	44 KB
Profesor_1_Total	27,79 MB					
Profesor_1_Datos Móviles	3,1 KB	3,2 KB	3,2 KB	3,0 KB	3,1 KB	3,2 KB

Tabla N° 54: Pruebas Consulta Profesor _1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 55.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1,5 S.	1 S.	1,5 S.	1 S.	1 S.

Tabla N° 55: Tiempo Respuestas Pruebas Profesor_1

8.4.1.8 Pruebas Consulta Profesor_2

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_2 las Figuras 52 y 53 muestran los datos de almacenamiento utilizados por la aplicación.

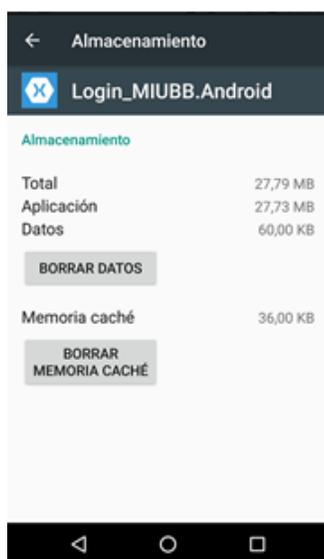


Figura N° 86: Profesor_2 Menor Peso



Figura N° 87: Profesor_2 Mayor Peso

Las Figuras 91 y 92 Muestran el mayor y menor consumo hecho por la consulta Profesor_2 al servidor.



**Figura N° 88: Profesor_2
menor Consumo**



**Figura N° 89: Profesor_2
Mayor Consumo**

Los resultados de las pruebas están expresados en la Tabla N° 56.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_2_data	60 KB					
Profesor_2_cache	47 KB	44 KB	45 KB	44 KB	47 KB	44 KB
Profesor_2_Total	27,79 MB					
Profesor_2_Datos Moviles	1,7 KB	1,4 KB	1,5 KB	1,6 KB	1,4 KB	1,8 KB

Tabla N° 56: Pruebas Consulta Profesor_2

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 57.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1 S.	1,5 S.	1 S.	1 S.	1 S.

Tabla N° 57: Tiempo Respuestas Pruebas Profesor_2

8.4.1.9 Pruebas Consulta Listado_1

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_1 las Figuras 90 y 91 muestran los datos de almacenamiento utilizados por la aplicación.

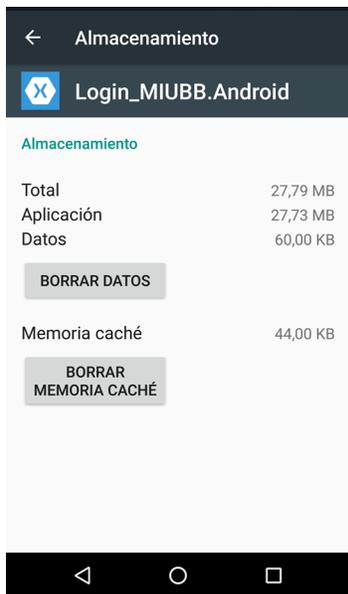


Figura N° 91: Listado_1 Menor Peso

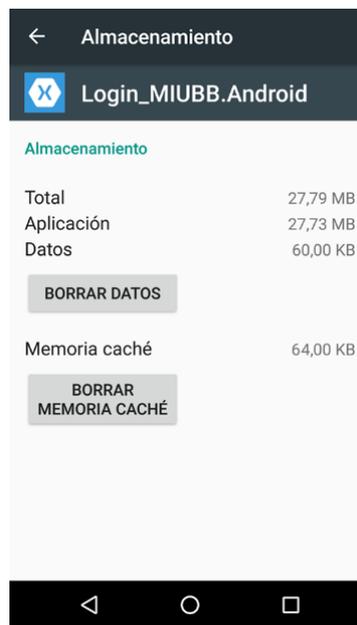


Figura N° 90: Listado_1 Mayor Peso

Las Figuras 92 y 93 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 93: Listado_1 Menor Consumo



Figura N° 92: Listado_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 58.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_1_data	60 KB					
Lista_1_cache	47 KB	44 KB	45 KB	44 KB	64 KB	44 KB
Lista_1_Total	27,79 MB					
Lista_1_Datos Moviles	10,1 KB	10,4 KB	10,5 KB	10,3 KB	10,5 KB	10,3 KB

Tabla N° 58: Pruebas Consulta Listado _1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 59.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_1	1,5 S.	1 S.	1 S.	1,5 S.	1 S.	1,5 S.

Tabla N° 59: Tiempo Respuestas Pruebas Listado_1

8.4.1.10 Pruebas Consulta Listado_2

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_2 las Figuras 97 y 98 muestran los datos de almacenamiento utilizados por la aplicación.

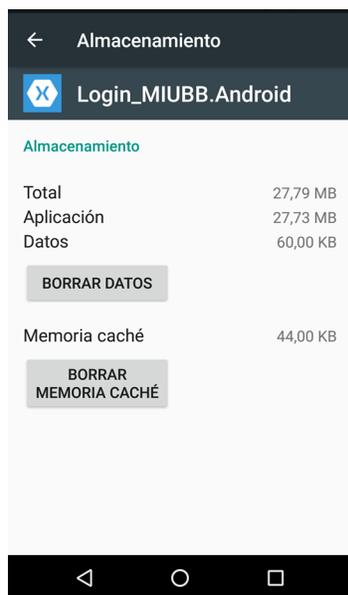


Figura N° 94: Listado_2 Menor Peso

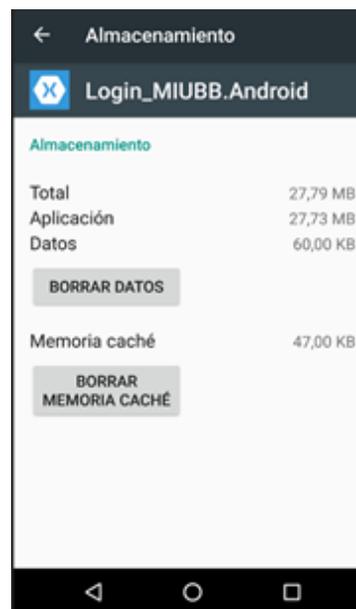


Figura N° 95: Listado_2 Menor Peso

Las Figuras 99 y 100 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 97: Listado_2 Menor Consumo



Figura N° 96: Listado_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 60.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_2_data	60 KB					
Lista_2_cache	47 KB	44 KB	45 KB	44 KB	47 KB	44 KB
Lista_2_Total	27,79 MB					
Lista_2_Datos Moviles	9,2 KB	9,2 KB	8,5 KB	8,5 KB	9 KB	8,8 KB

Tabla N° 60: Pruebas Consulta Listado_2

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 61.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_2	1 S.	1 S.	1,5 S.	1 S.	1,5 S.	2 S.

Tabla N° 61 : Tiempo Respuestas Pruebas Listado_2

8.5 Aplicación en Ionic

Ionic framework tiene muchas ventajas en su utilización para el desarrollo de una aplicación multiplataforma siendo utilizada por alguien que tenga conocimientos sobre desarrollo web, es decir, HTML, CSS y AngularJS, por lo que no debería suponer una gran dificultad la creación de aplicaciones en este Framework. Además su rendimiento y la documentación disponible nos posibilitan el desarrollo de aplicaciones rápidas y atractivas sin ser necesario tener conocimientos de múltiples lenguajes nativos.

Para la presente investigación se desarrolla una aplicación llamada “MyApp” implementada de forma web con el Framework Ionic, la que sin ser utilizada entrega los datos especificados en la Figura N° 100.

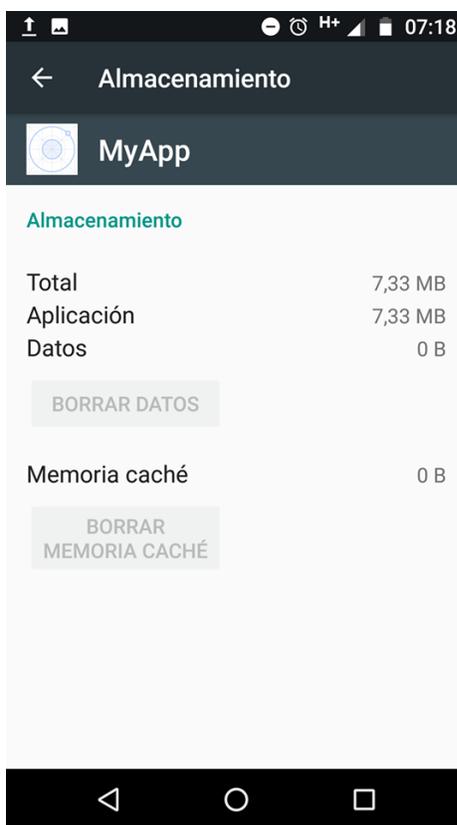


Figura N° 98: Aplicación Ionic Sin uso

Con lo anterior se rescató la siguiente información expresada en la Tabla N° 60 donde se puede apreciar que el peso de la aplicación es de 7.33MB.

APLICACIÓN “MyApp” RECIÉN INSTALADA	
APP	7,33 MB
DATA	0 B
CACHE	0 B
Total	7,33 MB

Tabla N° 62: Datos Aplicación Ionic Recién Instalada

8.5.1 Pruebas

Luego de presentar las características de aplicación en Ionic se realizan las pruebas de las consultas con acceso al servidor, con el fin de obtener una muestra fidedigna cada una de las pruebas será realizada seis veces, se expondrán las capturas de la muestras con mayor y menor índice de almacenamiento.

8.5.1.1 Pruebas Tiempos De Apertura

Las pruebas de Tiempos de apertura tienen como fin determinar cuan rápida es la aplicación a la hora de cargar los componentes que necesita para su correcto y total funcionamiento. La Tabla N°63 muestra los datos obtenido luego de 6 pruebas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Apertura	3 S.	3 S.	3 S.	3,5 S.	3 S.	3 S.

Tabla N° 63: Prueba Tiempos de Apertura

8.5.1.2 Pruebas Login

Luego de 6 Pruebas con el Login las Figuras 99 y 100 presentan las pruebas con más altos índices con respecto a la memoria utilizada.

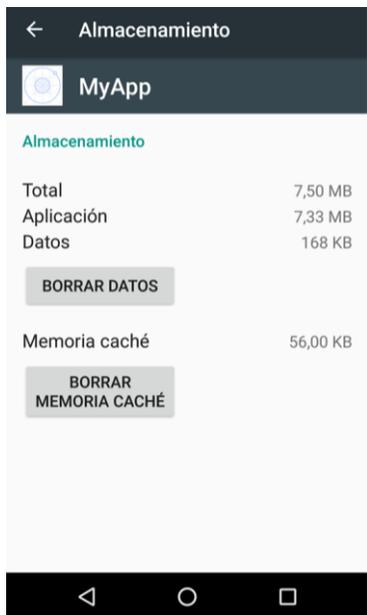


Figura N° 99: Login Menor Almacenamiento

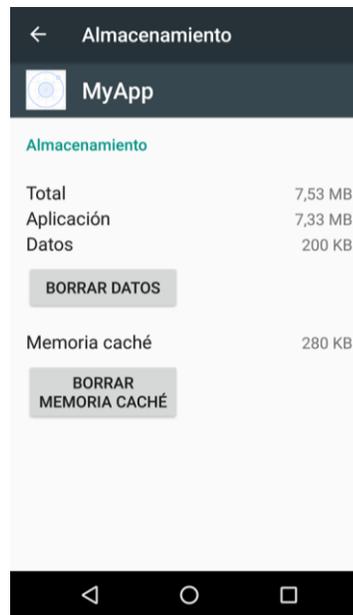


Figura N° 100: Login Mayor Almacenamiento

Y las figuras 101 y 102 muestran el mayor y menor consumo hecho por la consulta login al servidor.



Figura N° 101 Login Menor Consumo



Figura N° 102: Login Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 64.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Login_data	168 KB	200 KB				
Login_cache	56 KB	280 KB	56 KB	56 KB	210 KB	56 KB
Login_Total	7,5 MB	7,53 MB	7,5 MB	7,5 MB	7,5 MB	7,5 MB
Login_Datos Moviles	1,7 KB	1,9 KB	1,8 KB	1,6 KB	1,7 KB	1,6 KB

Tabla N° 64 Pruebas Login

Además del análisis con respecto al peso y el consumo de datos se debe tomar en cuenta el tiempo de reacción de la aplicación a la hora de realizar las consultas, la Tabla N° 65 muestra los tiempos de respuesta que se obtuvieron con cada una de las pruebas realizadas.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Login	1 S.	1 S.	1 S.	1,5 S.	1 S.	1 S.

Tabla N° 65: Tiempos de respuesta Login

8.5.1.3 Pruebas Consulta Horarios

Luego de 6 pruebas utilizando la consulta de horarios las Figuras 105 y 106 muestran los datos de almacenamiento utilizados por la aplicación.

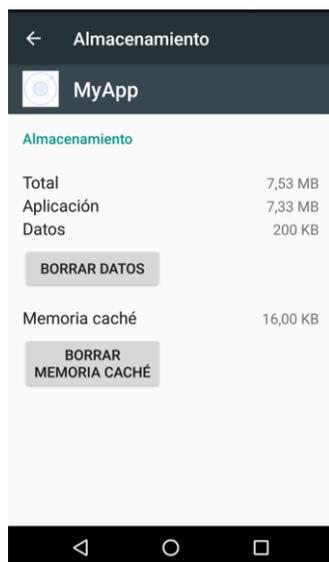


Figura N° 103: Horarios Menor Almacenamiento



Figura N° 104: Horarios Mayor Almacenamiento

Las Figuras 107 y 108 Muestran el mayor y menor consumo hecho por la consulta Horarios al servidor.



Figura N° 105 Horarios mayor Consumo

Figura N° 106: Horarios Menor Consumo
 Los resultados de las pruebas están expresados en la Tabla N° 66

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Horario_data	200 KB					
Horario_cache	16 KB	16 KB	28 KB	16 KB	28 KB	16 KB
Horario_Total	7,53 MB					
Horario_Datos Moviles	2,9 KB	2,8 KB	2,8 KB	3 KB	2,8 KB	2,9 KB

Tabla N° 66: Pruebas Consulta horarios

Luego de presentar los resultados de las pruebas de almacenamiento se procede a exponer las pruebas con respecto a los tiempos de respuesta que tuvo la aplicación en la consulta de Horarios, estos se muestran en la Tabla N° 67

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Horarios	1 S.					

Tabla N° 67: Tiempos de respuesta Horarios

8.5.1.4 Pruebas Consulta Notas

Luego de 6 pruebas utilizando la consulta de Notas las Figuras 109 y 110 muestran los datos de almacenamiento utilizados por la aplicación.

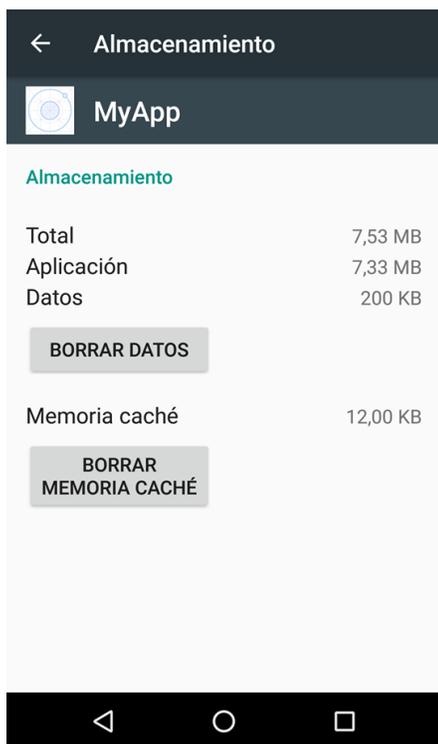


Figura N° 108: Notas Menor Almacenamiento

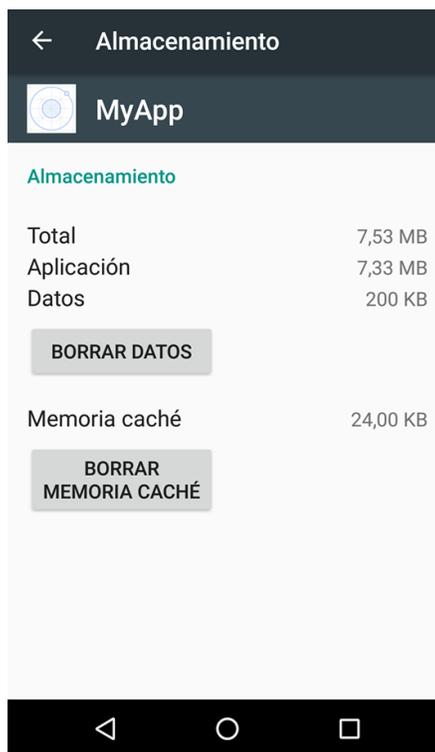


Figura N° 107: Notas Mayor Almacenamiento

Las Figuras 111 y 112 Muestran el mayor y menor consumo hecho por la consulta Notas al servidor.



Figura N° 110: Notas menor Consumo



Figura N° 109: Notas Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 68

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Notas_data	200 KB					
Notas_cache	24 KB	12 KB	16 KB	12 KB	20 KB	24 KB
Notas_Total	7,53 MB					
Notas_Datos Móviles	2,9 KB	3,1 KB	3,3 KB	3,7 KB	2,9 KB	3,3 KB

Tabla N° 68: Pruebas Consulta Notas

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 69

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Notas	1 S.	1,5 S.	1 S.	1 S.	1,5 S.	1 S.

Tabla N° 69: Tiempo Respuestas Pruebas Notas

8.5.1.5 Pruebas Consulta Inscripcion_1

Luego de 6 pruebas utilizando la consulta de Inscripcion_1 las Figuras 113 y 114 muestran los datos de almacenamiento utilizados por la aplicación en esta consulta en particular la memoria cache no varía en ninguna de las consultas realizadas.

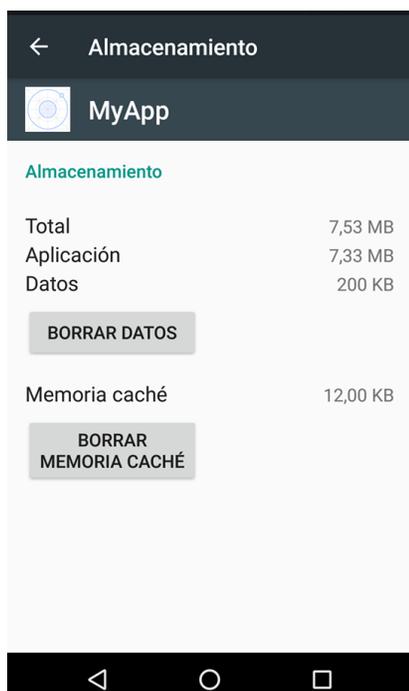


Figura N° 112: Inscripcion_1 Peso Menor

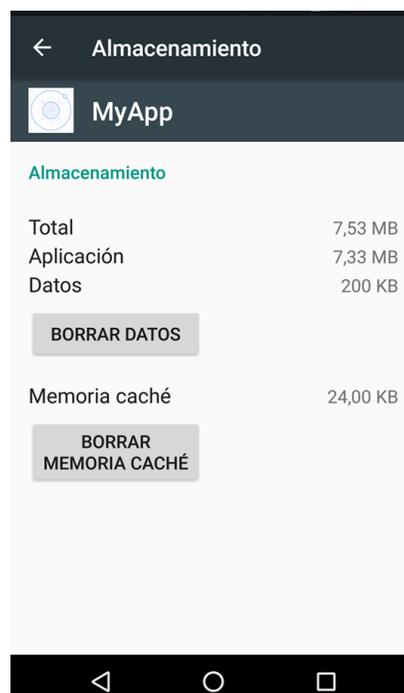


Figura N° 111: Inscripcion_1 Peso Mayor

Las Figuras 113 y 114 Muestran el mayor y menor consumo hecho por la consulta Incripcion_1 al servidor.



Figura N° 114: Incripcion_1 Menor Consumum



Figura N° 113: Incripcion_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 70

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Incripcion_1_data	200 KB					
Incripcion_1_cache	12 KB	12 KB	24 KB	24 KB	16 KB	24 KB
Incripcion_1_Total	7,53 MB					
Incripcion_1_Datos Móviles	1,6 KB	1,6 KB	1,9 KB	1,8 KB	1,5 KB	1,8 KB

Tabla N° 70: Pruebas Consulta Incripcion_1

Una vez mostrados los datos de almacenamiento se muestran los datos de tiempos de respuestas, estos se muestran en la Tabla N° 71

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Incripcion_1	1 S.					

Tabla N° 71: Tiempo Respuestas Pruebas Incripcion_1

8.5.1.6 Pruebas Consulta Incripcion_2

Luego de 6 pruebas utilizando la consulta de Incripcion_2 las Figuras 117 y 118 muestran los datos de almacenamiento utilizados por la aplicación.

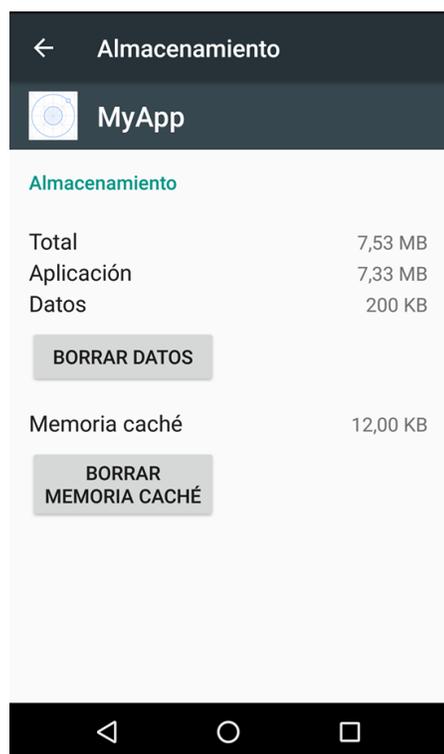


Figura N° 116: Incripcion_2 Menor Peso

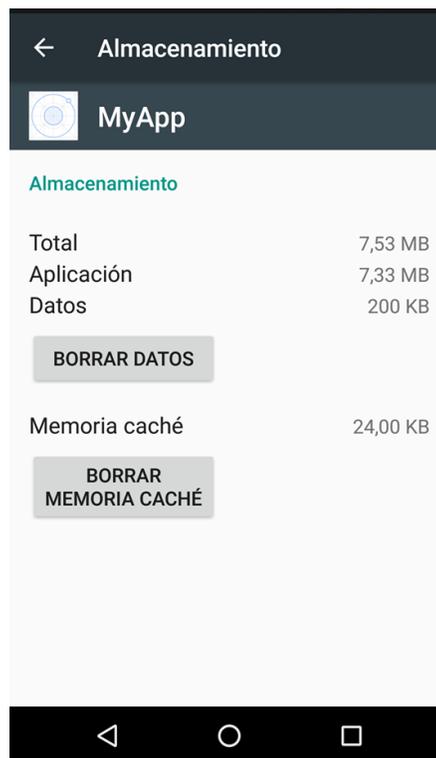


Figura N° 115: Incripcion_2 Mayor Peso

Las Figuras 119 y 120 Muestran el mayor y menor consumo hecho por la consulta Incripcion_2 al servidor.



Figura N° 117: Incripcion_2 Menor Consumo



Figura N° 118: Incripcion_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 72.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Inscripcion_2_data	200 KB					
Inscripcion_2_cache	24 KB	12 KB	12 KB	24 KB	24 KB	12 KB
Inscripcion_2_Total	7,53 MB					
Inscripcion_2_Datos Moviles	1,7 KB	1,6 KB	1,6 KB	2,0 KB	2,0 KB	1,6 KB

Tabla N° 72: Pruebas Consulta Inscripcion_2

Una vez mostrados los datos de almacenamiento, los datos de tiempos de respuestas se muestran en la Tabla N° 73.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1 S.	1 S.	1,5 S.	1 S.	1 S.

Tabla N° 73: Tiempo Respuestas Pruebas Inscripcion_2

8.5.1.7 Pruebas Consulta Profesor_1

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_1 las Figuras 121 y 122 muestran los datos de almacenamiento utilizados por la aplicación.

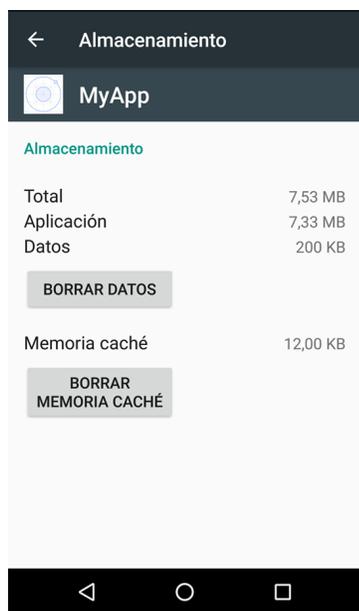


Figura N° 120: Profesor_1 Menor Peso

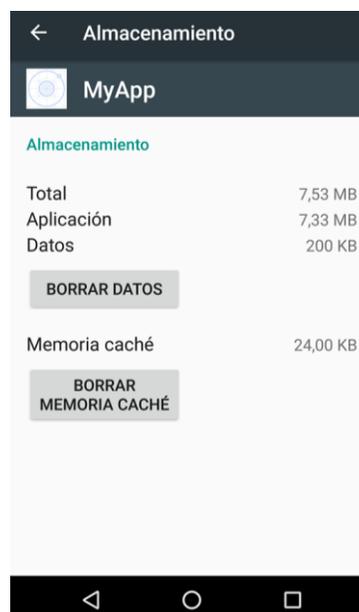


Figura N° 119: Profesor_1 mayor peso

Las Figuras 121 y 122 Muestran el mayor y menor consumo hecho por la consulta Profesor_1 al servidor.



Figura N° 122: Profesor_1 menor Consumo



Figura N° 121 Profesor_1 mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 74.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_1_data	200 KB					
Profesor_1_cache	24 KB	24 KB	24 KB	12 KB	24 KB	12 KB
Profesor_1_Total	7,53	7,53	7,53	7,53	7,53	7,53
Profesor_1_Datos Móviles	1,7 KB	1,7 KB	1,9 KB	1,7 KB	1,7 KB	1,4 KB

Tabla N° 74: Pruebas Consulta Profesor_1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 75.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1 S.	1 S.	1,5 S.	1 S.	1 S.	1,5 S.

Tabla N° 75: Tiempo Respuestas Pruebas Profesor_1

8.5.1.8 Pruebas Consulta Profesor_2

Una vez realizadas las 6 pruebas utilizando la consulta de Profesor_2 las Figuras 123 y 124 muestran los datos de almacenamiento utilizados por la aplicación.

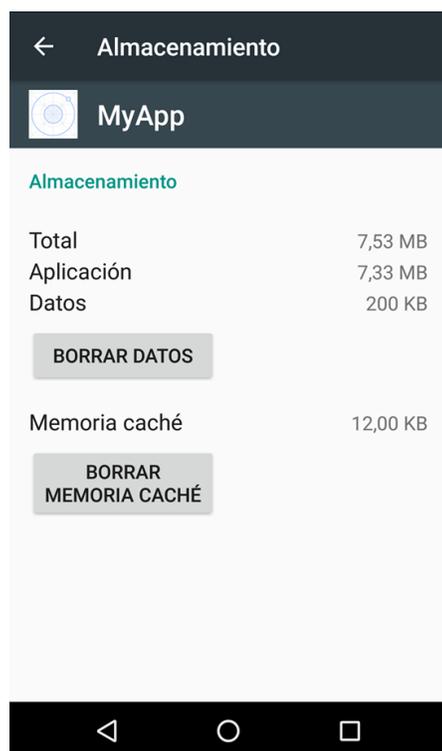


Figura N° 124: Profesor_2 Menor Peso

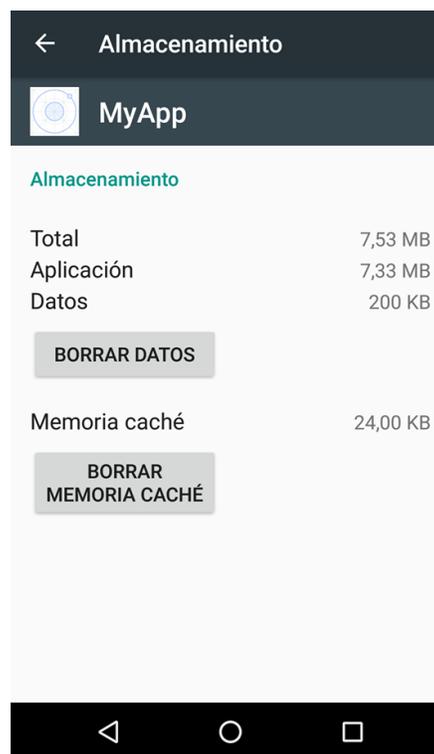


Figura N° 123: Profesor_2 Mayor Peso

Las Figuras 127 y 128 Muestran el mayor y menor consumo hecho por la consulta Profesor_2 al servidor.



Figura N° 125: Profesor_2 menor Consumo



Figura N° 126: Profesor_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 76.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Profesor_2_data	200 KB					
Profesor_2_cache	12 KB	12 KB	24 KB	24 KB	24 KB	12 KB
Profesor_2_Total	7,53	7,53	7,53	7,53	7,53	7,53
Profesor_2_Datos Móviles	1,4 KB	1,7 KB	1,8 KB	1,7 KB	1,7 KB	1,7 KB

Tabla N° 76: Pruebas Consulta Profesor_2

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 77.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Inscripcion_2	1,5 S.	1 S.	1 S.	1 S.	1 S.	1 S.

Tabla N° 77: Tiempo Respuestas Pruebas Profesor_2

8.5.1.9 Pruebas Consulta Listado_1

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_1 las Figuras 129 y 130 muestran los datos de almacenamiento utilizados por la aplicación.

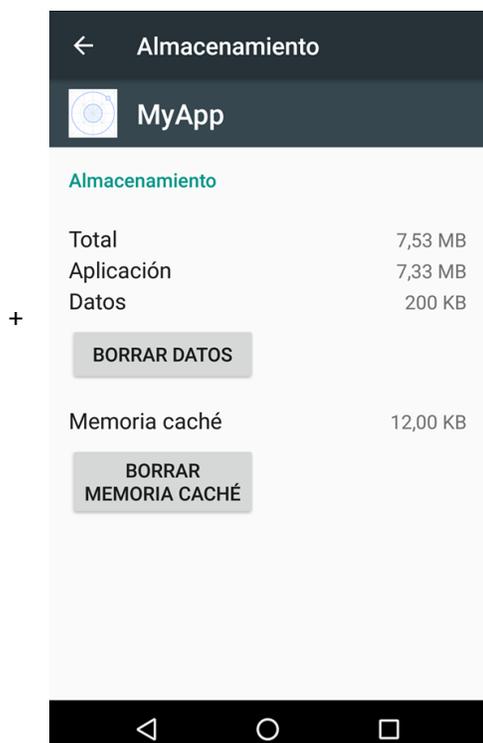


Figura N° 128: Listado_1 Menor Peso

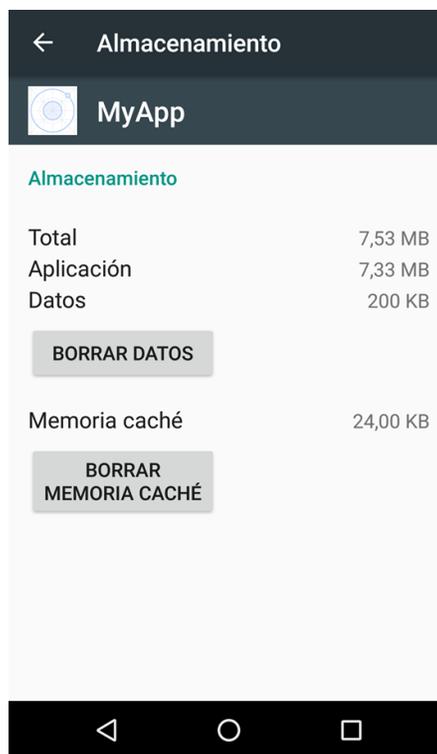


Figura N° 127: Listado_1 Mayor Peso

Las Figuras 129 y 130 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 130: Listado_1 Menor Consumo



Figura N° 129: Listado_1 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 78.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_1_data	200 KB					
Lista_1_cache	24 KB	24 KB	24 KB	24 KB	12 KB	24 KB
Lista_1_Total	7,53 MB					
Lista_1_Datos Moviles	2,2 KB	2,2 KB	3 KB	2,2 KB	2,1 KB	2,4 KB

Tabla N° 78: Pruebas Consulta Listado_1

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 79.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_1	1,5 S.	2 S.	1,5 S.	2,5 S.	2 S.	2 S.

Tabla N° 79: Tiempo Respuestas Pruebas Listado_1

8.5.1.10 Pruebas Consulta Listado_2

Una vez realizadas las 6 pruebas utilizando la consulta de Listado_2 las Figuras 133 y 134 muestran los datos de almacenamiento utilizados por la aplicación.

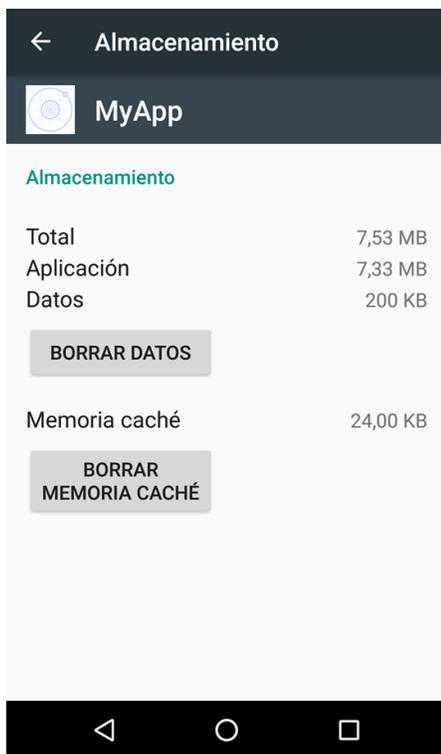


Figura N° 132: Listado_2 Menor Peso

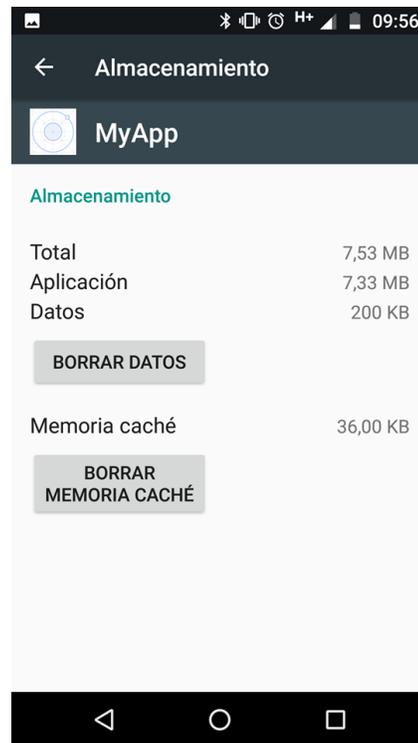


Figura N° 131: Listado_2 Mayor Peso

Las Figuras 135 y 136 Muestran el mayor y menor consumo hecho por la consulta Listado_2 al servidor.



Figura N° 133: Listado_2 Menor Consumo



Figura N° 134: Listado_2 Mayor Consumo

Los resultados de las pruebas están expresados en la Tabla N° 80.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Lista_2_data	200 KB					
Lista_2_cache	24 KB	24 KB	24 KB	36 KB	24 KB	36 KB
Lista_2_Total	7,53 MB					
Lista_2_Datos Moviles	2,4 KB	2,4 KB	2,5 KB	3,3 KB	2,5 KB	2,8 KB

Tabla N° 80: Pruebas Consulta Listado_2

Una vez mostrados los datos de almacenamiento se analizan los datos de tiempos de respuestas, estos se muestran en la Tabla N° 81.

	Prueba_1	Prueba_2	Prueba_3	Prueba_4	Prueba_5	Prueba_6
Tiempo Listado_2	2 S.	1,5 S.	1 S.	2 S.	1 S.	2 S.

Tabla N° 81 : Tiempo Respuestas Pruebas Listado_2

9 ANÁLISIS EMPÍRICO DE LOS RESULTADOS DE LAS PUEBAS EN LA CAPA DE APLICACIÓN

Del Capítulo anterior se rescató información expresadas en las tablas de la N° 22 a la N° 81, con lo anterior se procesa a realizar las comparaciones correspondientes al peso, memoria utilizada, datos consumidos, tiempos de apertura y tiempos de consulta entre las tecnologías de la capa de aplicación investigadas.

9.1 Peso Total Sin Uso

Una vez instaladas las aplicaciones los resultados rescatados con respecto al peso de cada una de las aplicaciones están expresadas en la Figura N° 137.

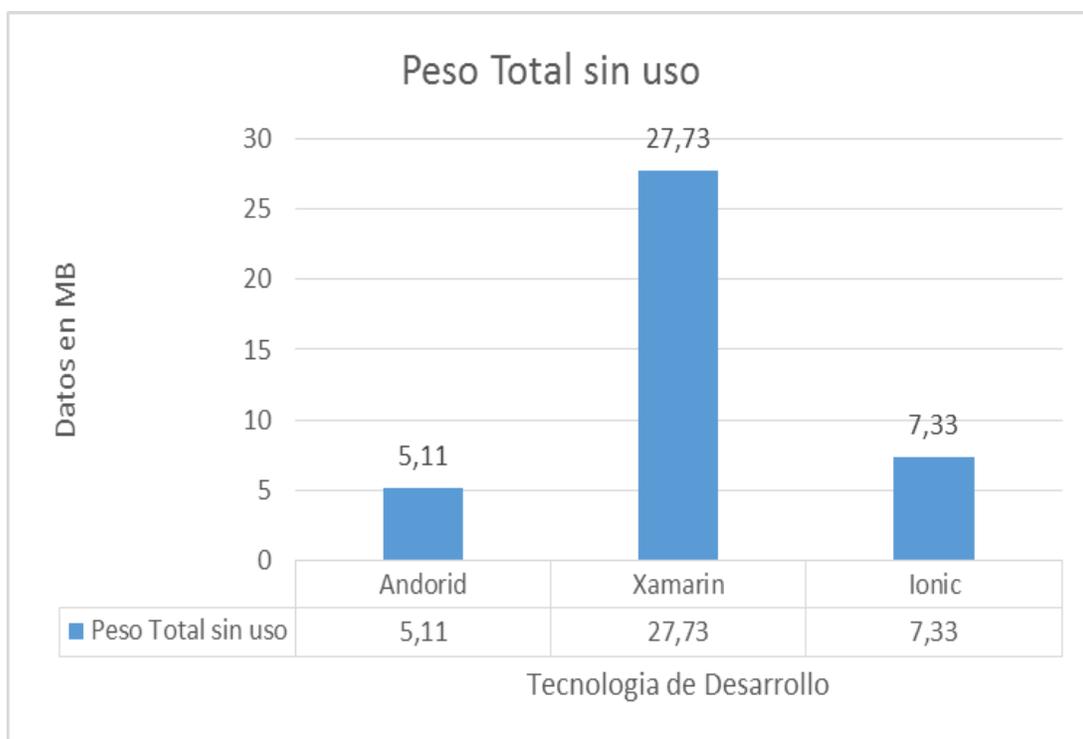


Figura N° 135: Comparativa Peso Aplicaciones

De la Figura N° 137 anteriormente expuesta podemos observar que la aplicación desarrollada de forma Nativa lleva la ventaja con respecto a las otras dos plataformas de desarrollo, ya que sin tener uso alguno su peso es el menor.

9.2 Comparación De Datos Guardados en las Consultas

Cuando son ejecutadas por primera vez estas utilizan un espacio de memoria para guardar datos, a medida que son utilizadas el valor de los datos puede variar en la Figura N° 138 se representan estos datos de manera gráfica.

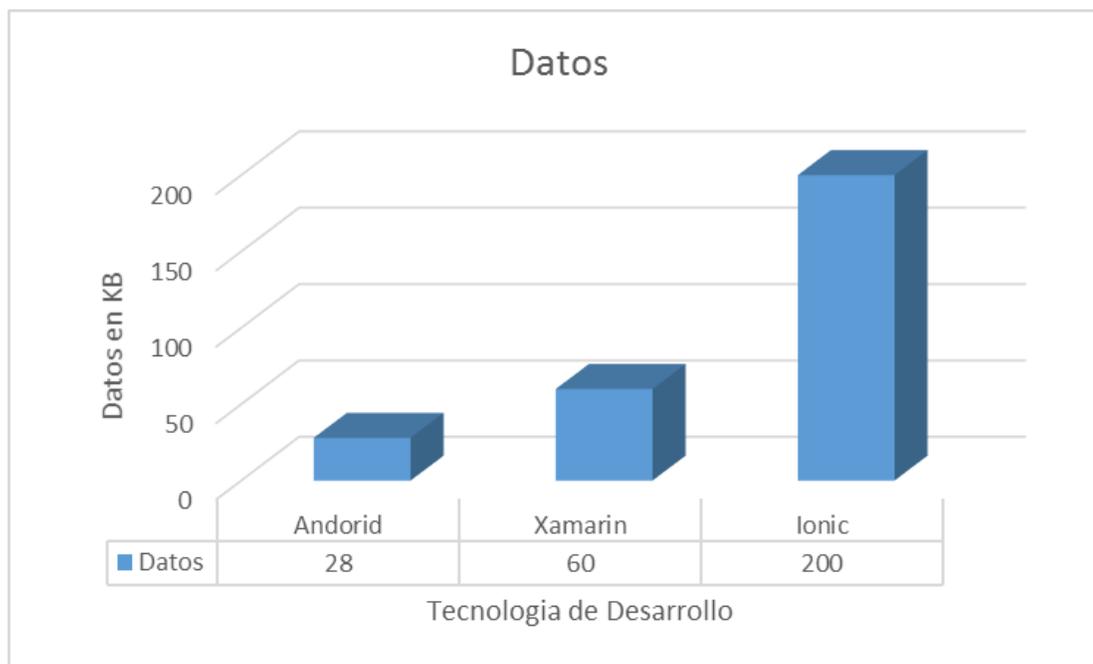


Figura N° 136: Comparativa de Memoria Cache y Datos Almacenados

De la figura N° 138 se observa como Ionic en comparación a las demás plataformas es quien más utiliza este recurso, mientras que Xamarin y Andorid Studio se mantienen constantes.

9.3 Peso Final Aplicaciones

Luego de analizar la diferencia de memoria cache y datos entre las aplicaciones a la hora de realizar las consultas, debemos evaluar su peso final es por esto que se hace necesario evaluar el peso final que estas tienen, la Figura N° 137.

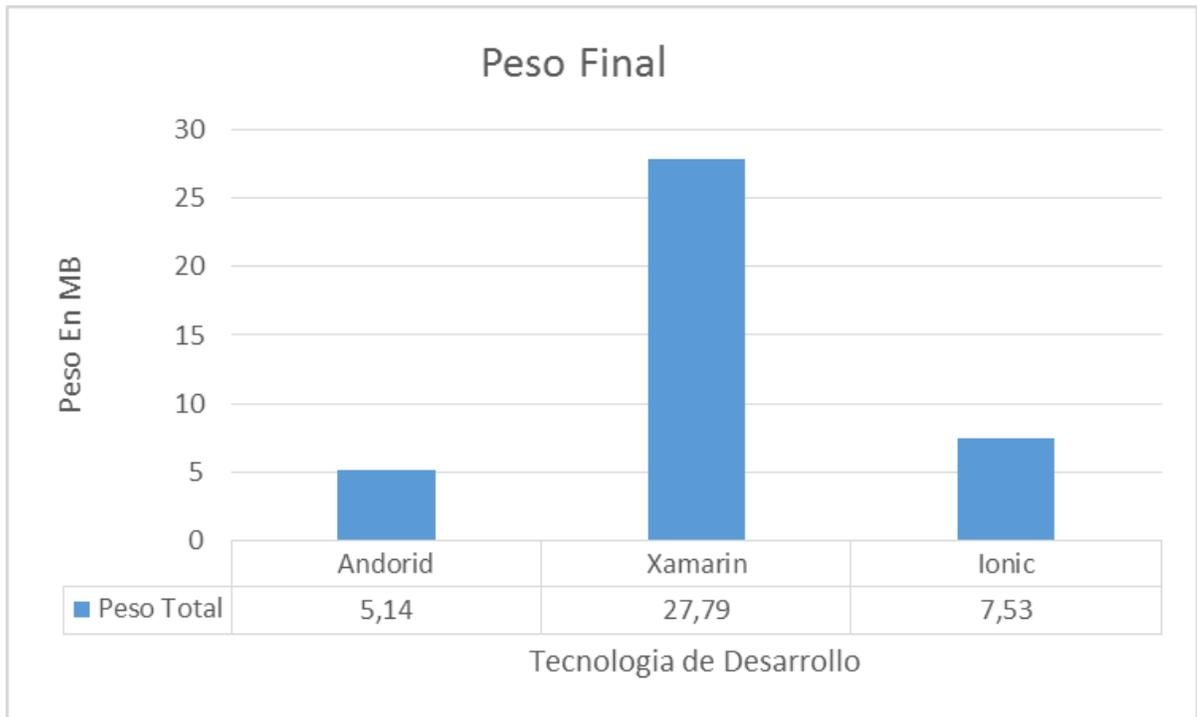


Figura N° 137: Peso Final

De la Figura anterior podemos observar que el peso de las aplicación no vario en casi nada, puesto que Android Studio una vez hechas las consultas vario su peso en un 0.6%, mientras que Xamarin lo hizo con un 0.22% y por ultimo Ionic termino con un 2.45% más de memoria utilizada al final de las pruebas

9.4 Análisis De Tiempos

Un aspecto importante a considerar a la hora de desarrollar aplicaciones móviles es que estas sean rápidas y aprovechen al máximo las capacidades del dispositivo en el cual se despliegan, es por esto que se hace necesario medir los tiempos de respuesta que cada una de estas tienen al momento de ser utilizadas.

Un sinónimo de rapidez dentro de las aplicaciones móviles es tener bajos tiempos de respuestas a la hora de realizar una consulta al servidor, ya sea para actualizar, modificar o solo consultar datos, un tiempo corto en la respuesta supone un menor costo a la hora de consumir datos móviles.

9.4.1 Tiempos De Apertura

La Figura N° 138 presenta el promedio de cada una de las aplicaciones la hora de ser ejecutadas.

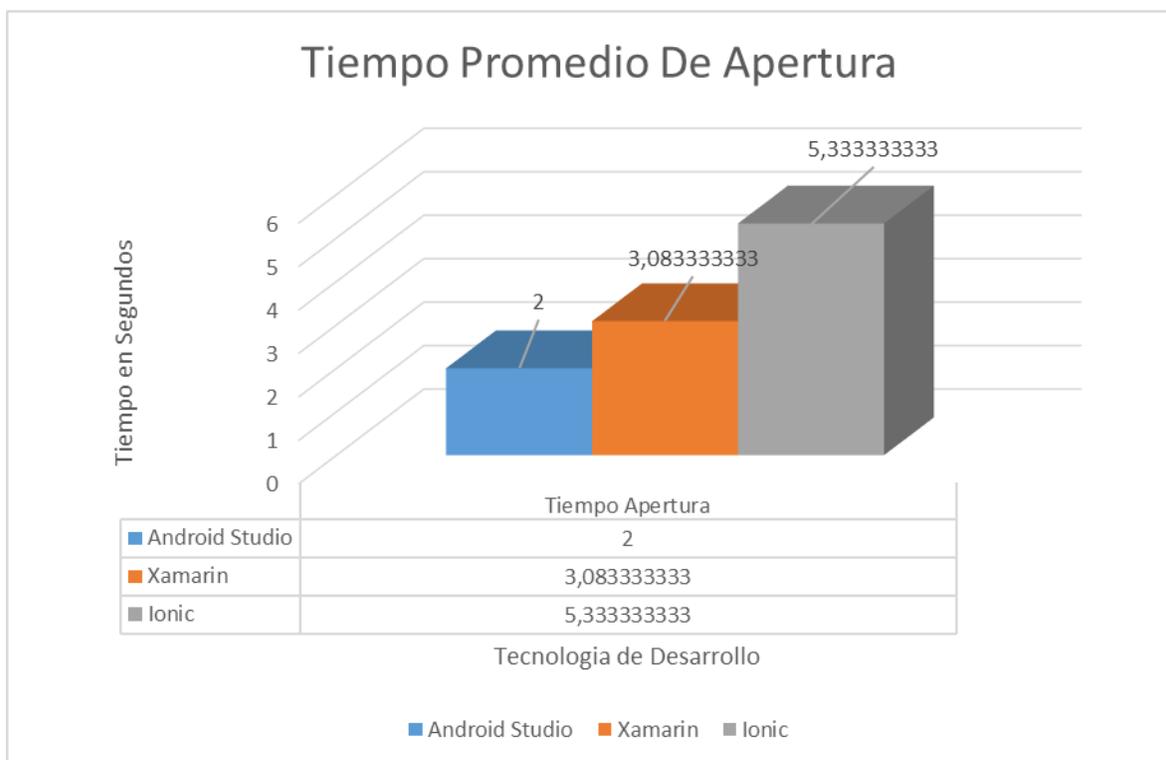


Figura N° 138: Comparativa Tiempos de Apertura

Como se puede apreciar en la Figura N° 138 la aplicación desarrollada en Android Studio es quien menos tiempo demora en mostrar la pantalla de Login con 2 Segundos, Xamarin en cambio tarda 1.5 veces más y por ultimo Ionic realiza esta tarea en un tiempo de 5.33 Segundos que representa 2.66 veces lo que tarda la aplicación Nativa de Android.

9.4.2 Tiempos De Respuesta Login

Adicionalmente medimos el tiempo que cada una de las aplicaciones a la hora de realizar la consulta de login al servidor.

Es por esto que la Figura N° 139 nos presenta de forma gráfica los tiempos que demora cada una de las aplicaciones en realizar la consulta de acceso.

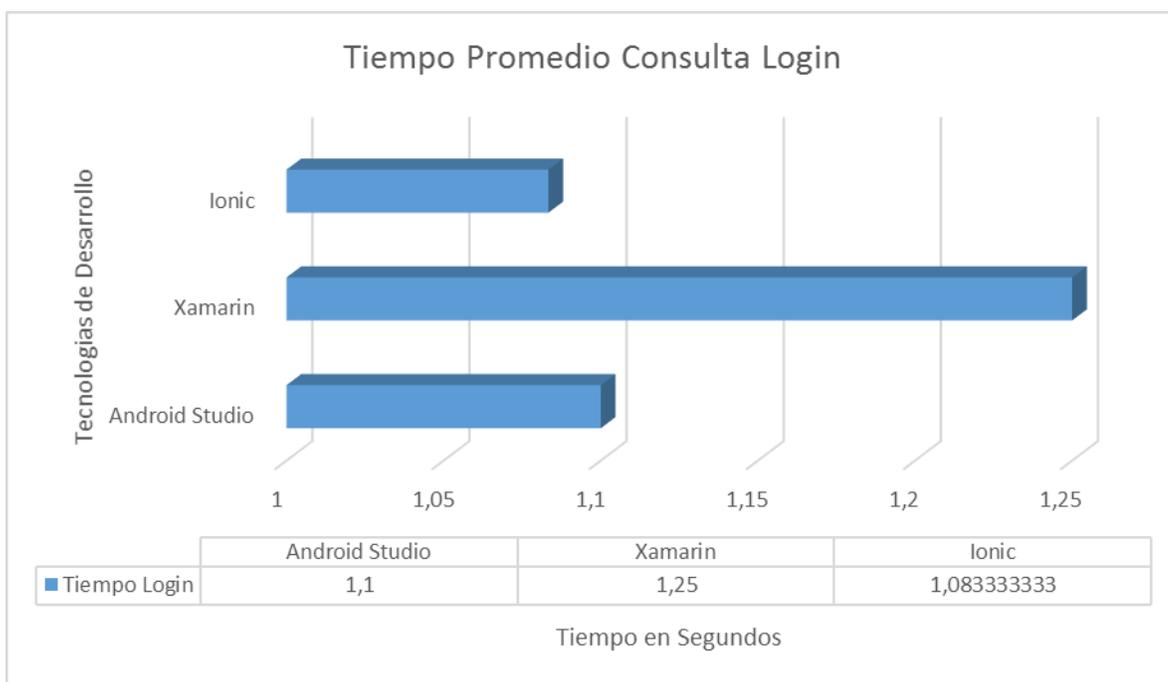


Figura N° 139: Comparativa Tiempos de Respuesta Login

Con respecto a la Figura N° 139 quien menos tiempo promedio tarda en consumir el servicio de Login es Ionic con 1.0833 Segundos promedio seguido de Android Studio con 1.1 Segundos y por ultimo Xamarin quien demora en promedio 1,25 Segundos.

9.4.3 Tiempos De Respuesta Consulta Horario

En todas las aplicaciones se midió el tiempo de respuesta al ahora de consultar Horarios, estos datos se grafican en la Figura N°140

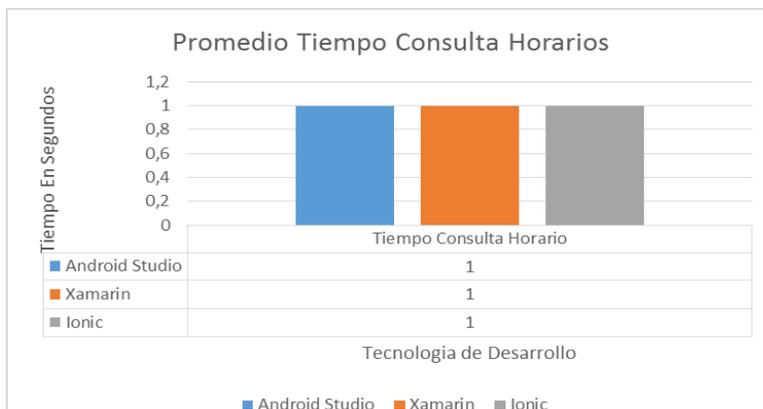


Figura N° 140: Comparativa Tiempos de Respuesta Horarios

De la Figura anterior podemos rescatar que en las tres aplicaciones el tiempo de respuesta fue el mismo.

9.4.4 Tiempos De Respuesta Consulta Notas

La Figura N° 141 presenta la comparación del tiempo promedio que las tres aplicaciones presentaron a la hora de realizar la consulta Notas.

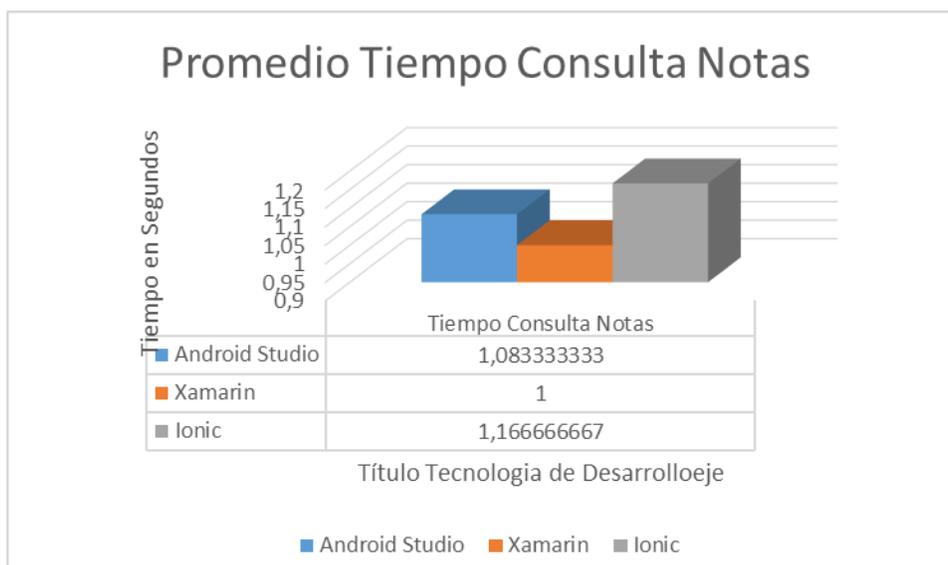


Figura N° 141: Comparativa Tiempos de Respuesta Notas

De la Figura anterior se puede observar que Xamarin es quien menos tiempo demora en consumir el servicio que entrega las notas.

9.4.5 Tiempos De Respuesta Consulta Inscripcion_1

La Figura N°142 presenta la comparación del tiempo promedio de cada una de las aplicaciones desarrolladas.

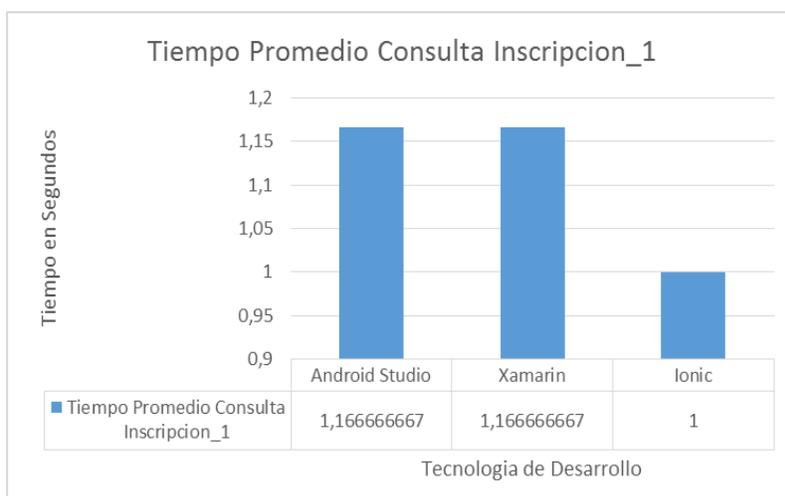


Figura N° 142: Comparativa Tiempos de Respuesta Inscripcion_1

En la Figura anterior se puede observar que Android Studio como Xamarin tienen el mismo tiempo de respuesta promedio, siendo Ionic quien menos demora en obtener los datos de la consulta.

9.4.6 Tiempos De Respuesta Consulta Inscripcion_2

La Figura N°143 presenta la comparación del tiempo promedio de cada una de las aplicaciones desarrolladas para la consulta Inscripcion_2.

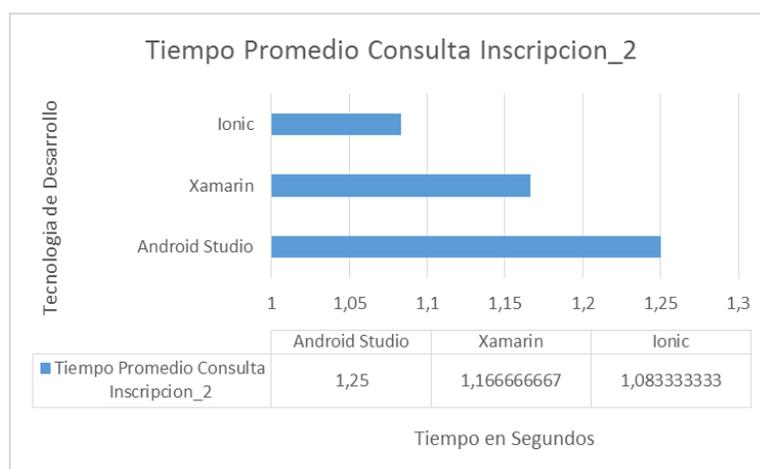


Figura N° 143: Comparativa Tiempos de Respuesta Inscripcion_2

Al igual que en la consulta anterior Ionic es quien mantiene un menor tiempo de respuesta

9.4.7 Tiempos De Respuesta Consulta Profesor_1

La Figura N°144 muestra la comparación de las tres aplicaciones a la hora de ejecutar la consulta Profesor_1.

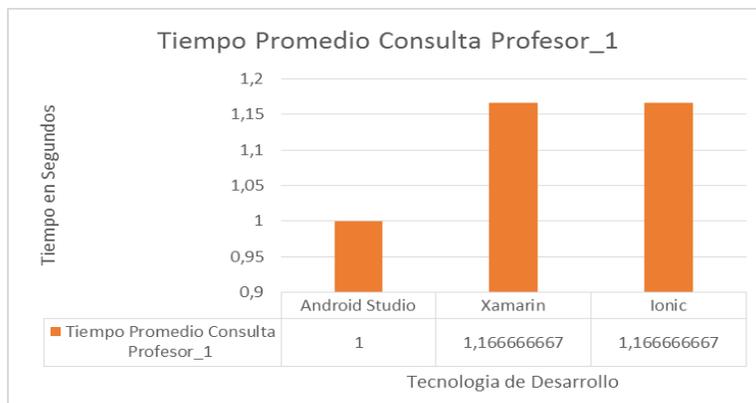


Figura N° 144: Comparativa Tiempos de Respuesta Profesor_1

En la Figura anterior se puede observar como Andorid Studio es quien menos tiempo demora en obtener los datos proporcionados por el servidor.

9.4.8 Tiempos De Respuesta Consulta Profesor_2

La Figura N°145 muestra la comparación de las tres aplicaciones a la hora de ejecutar la consulta Profesor_2.

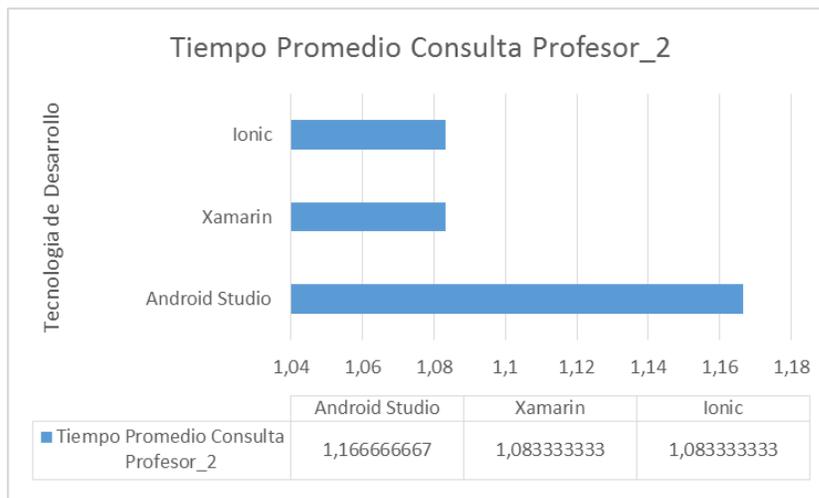


Figura N° 145: Comparativa Tiempos de Respuesta Profesor_2

La Figura N°145 muestra un resultado completamente diferente al descrito por el punto anterior, siendo en este caso Andorid Studio en más tiempo promedio demora en obtener los datos desde el servidor.

9.4.9 Tiempos De Respuesta Consulta Lista_1

La Figura N°146 muestra la comparación de las tres aplicaciones a la hora de ejecutar la consulta Lista_1.

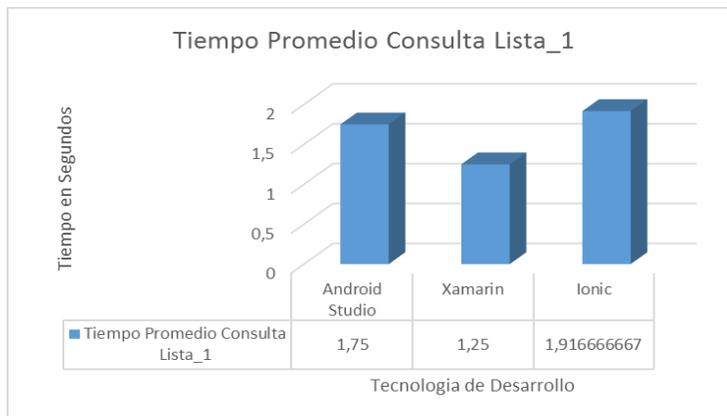


Figura N° 146: Comparativa Tiempos de Respuesta Lista_1

De la Figura anterior se puede deducir que tiende a procesar una lista de datos más rápido que las otras alternativas estudiadas.

9.4.10 Tiempos De Respuesta Consulta Lista_2

La Figura N°147 muestra la comparación de las tres aplicaciones a la hora de ejecutar la consulta Lista_1.

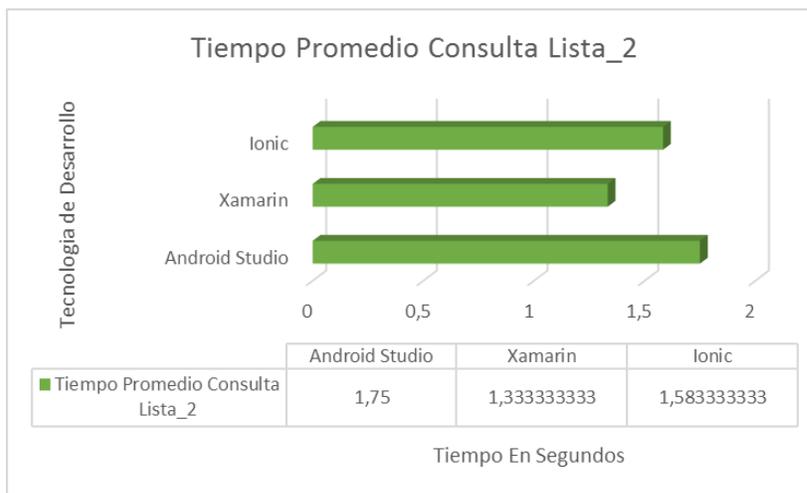


Figura N° 147: Comparativa Tiempos de Respuesta Lista_2

Al igual que la prueba anterior Xamarin es quien menos tiempo promedio demora en procesar la consulta Lista_2

9.5 Datos Consumidos

Con el fin de realizar una prueba comparable entre las 3 aplicaciones que se están presentando, todas realizan el proceso Login, Consulta Horario, Consulta Notas, Consulta Inscripcion_1, Consulta Inscripcion_2, Consulta Profesor_1, Consulta Profesor_2, Consulta Lista_1 y Consulta lista_2; Todas las aplicaciones apuntan a los mismos PHP en línea encargados de realizar las consulta por lo que las condiciones para las pruebas son las mismas para todas las aplicaciones.

Una vez explicado lo anterior procedemos a comparar la cantidad de Datos Móviles consumidos por cada una de las consultas realizadas por las aplicaciones.

Con respecto al consumo de datos realizado por las aplicaciones a la hora de ejecutar la consulta las consultas, se priorizaron 3 datos relevantes, el valor más bajo, el valor más alto y el promedio de las 6 pruebas realizadas

9.5.1 Comparación Consumo de Datos Login

En la Figura N° 148 se presentan los datos correspondientes a la ejecución de la consulta Login.

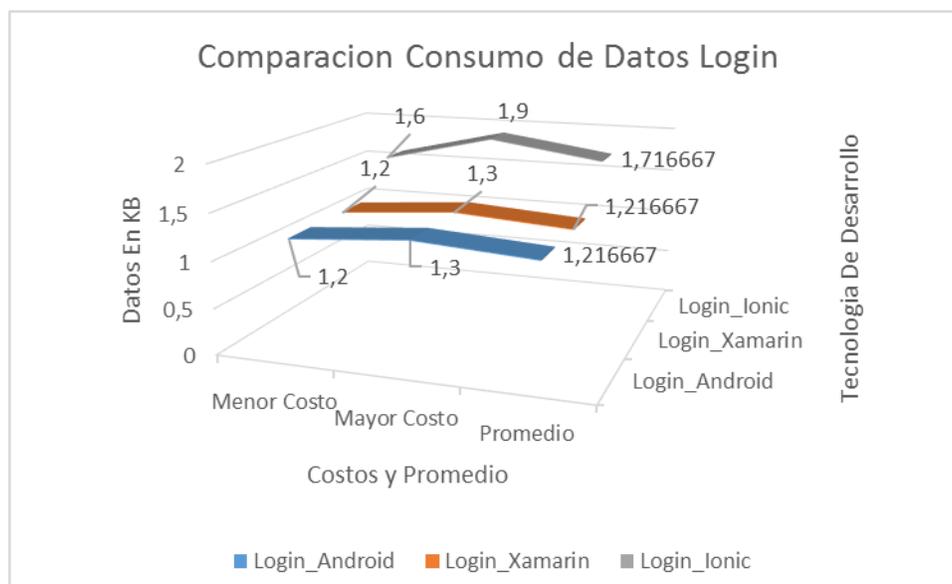


Figura N° 148: Comparación Consumo de Datos Login

9.5.2 Comparación Consumo de Datos Consulta Horarios

La Figura N° 149 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Horarios

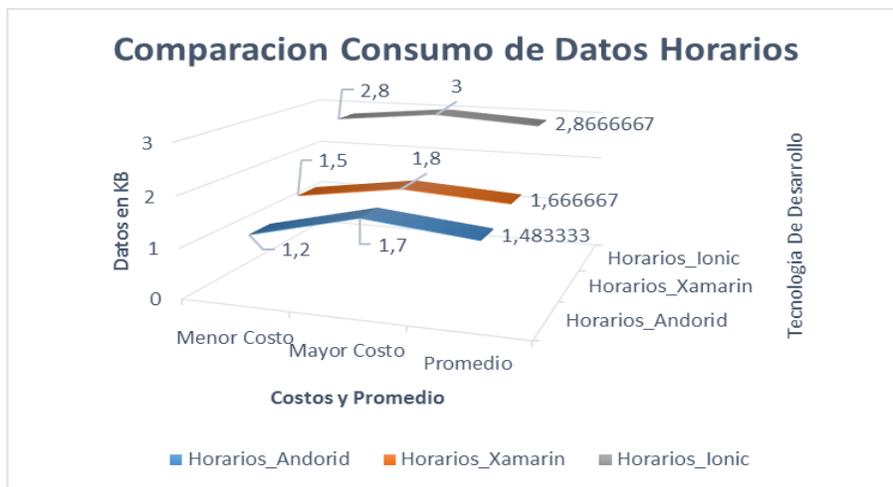


Figura N° 149: Comparación Consumo de Datos Consulta Horarios

9.5.3 Comparación Consumo de Datos Consulta Notas

La Figura N° 150 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Notas

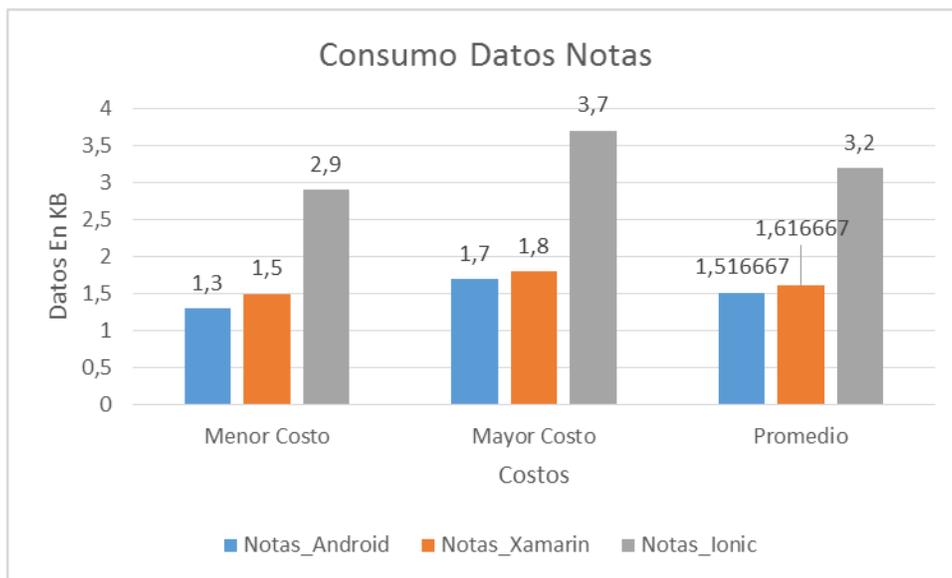


Figura N° 150: Comparación Consumo de Datos Consulta Notas

9.5.4 Comparación Consumo de Datos Consulta Inscripción_1

La Figura N° 151 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Inscripción_1.

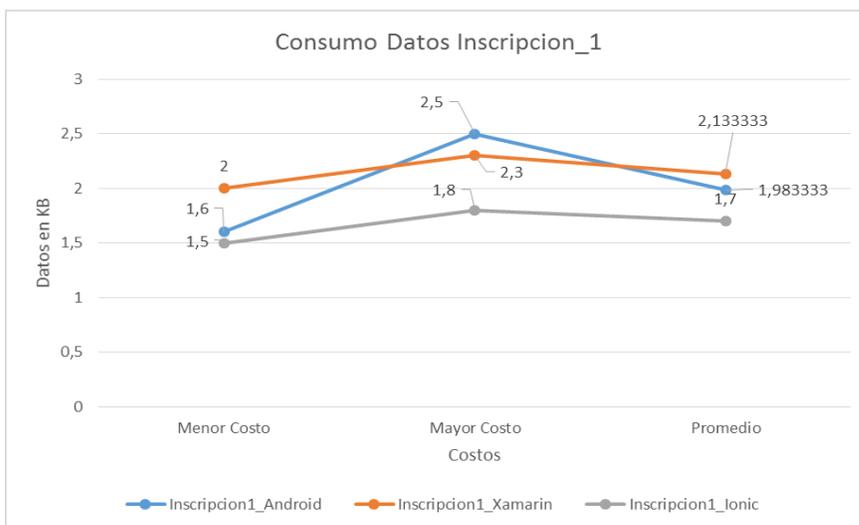


Figura N° 151: Comparación Consumo de Datos Consulta Inscripción_1

9.5.5 Comparación Consumo de Datos Consulta Inscripción_2

La Figura N° 152 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Inscripción_2

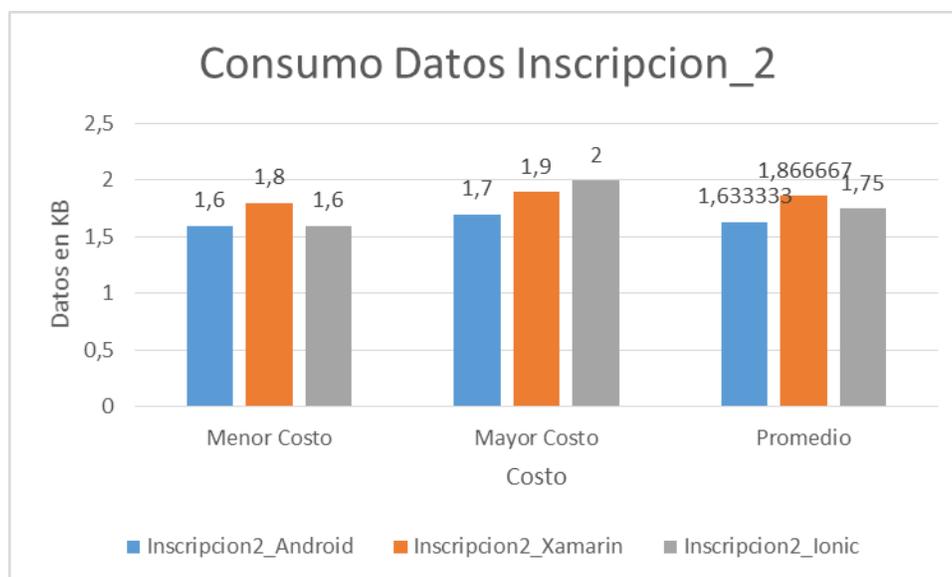


Figura N° 152: Comparación Consumo de Datos Consulta Inscripción_2

9.5.6 Comparación Consumo de Datos Consulta Profesor_1

La Figura N° 153 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Profesor_1

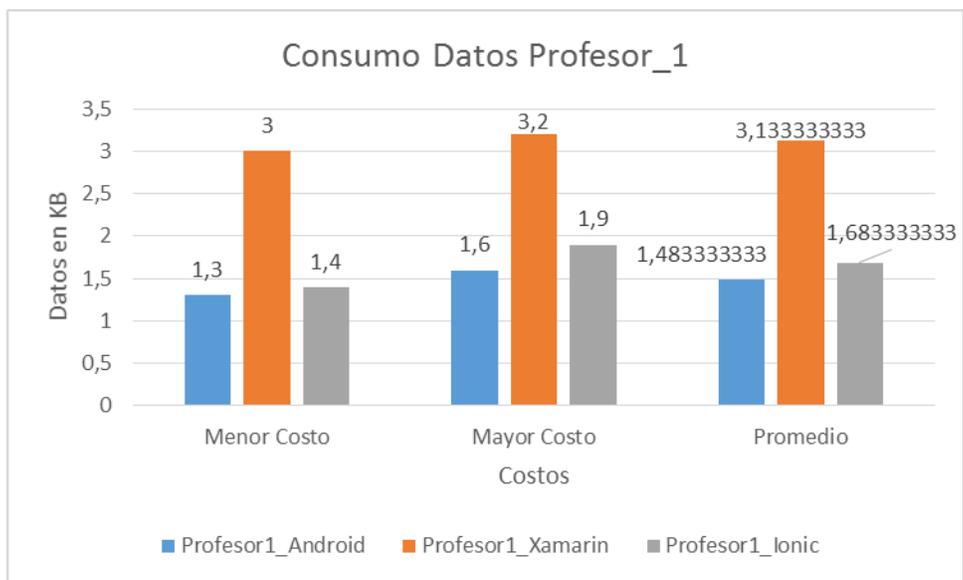


Figura N° 153: Comparación Consumo de Datos Consulta Profesor_1

9.5.7 Comparación Consumo de Datos Consulta Profesor_2

La Figura N° 154 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Profesor_2

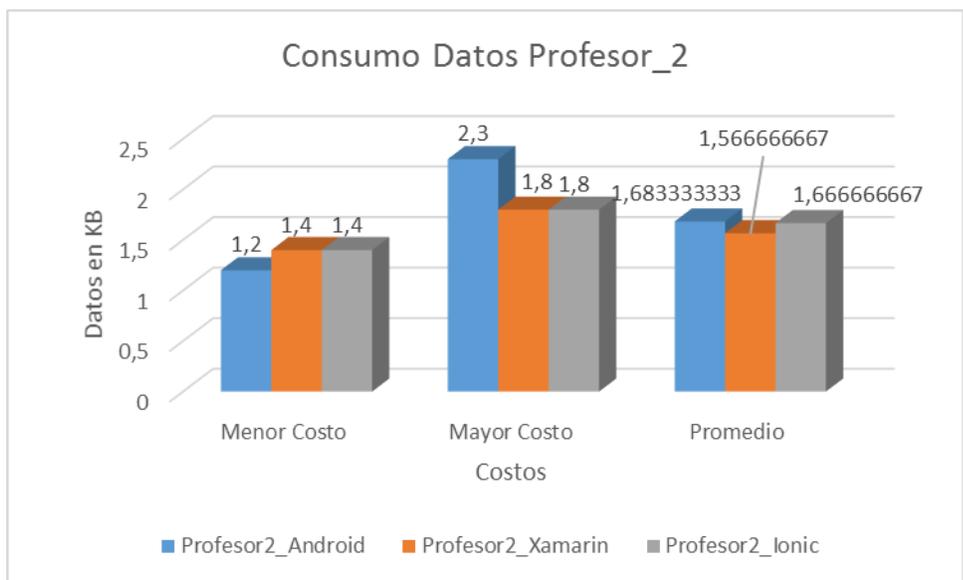


Figura N° 154: Comparación Consumo de Datos Consulta Profesor_2

9.5.8 Comparación Consumo de Datos Consulta Listado_1

La Figura N° 155 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Listado_1

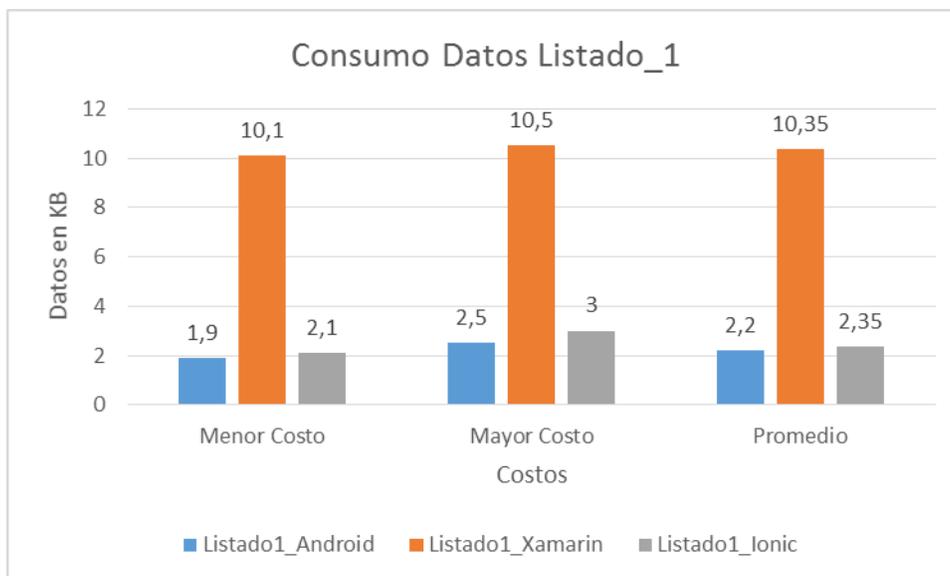


Figura N° 155: Comparación Consumo de Datos Consulta Listado_1

9.5.9 Comparación Consumo de Datos Consulta Listado_2

La Figura N° 156 presenta la comparación entre las aplicaciones a la hora de ejecutar la consulta de Listado_2

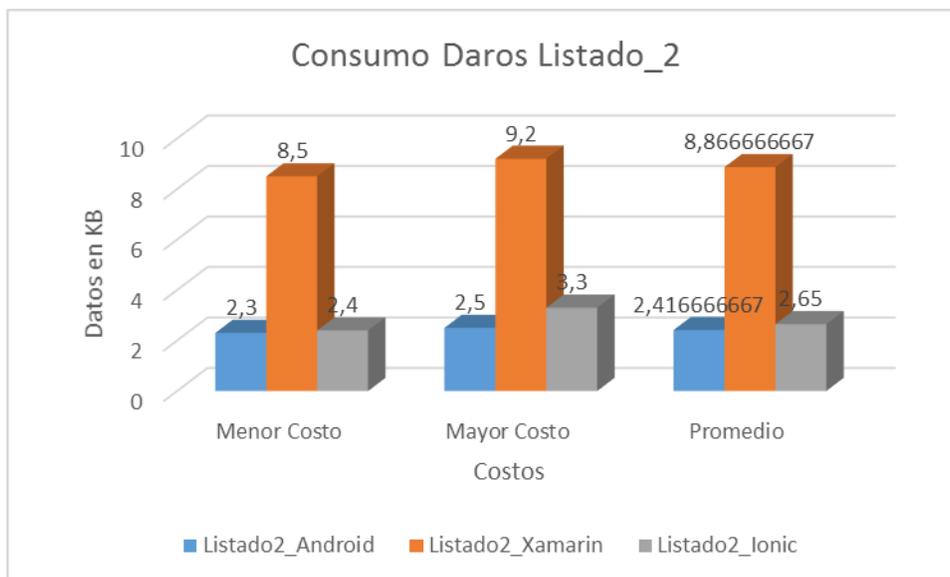


Figura N° 156: Comparación Consumo de Datos Consulta Listado_2

9.6 Conclusión de análisis capa de aplicación

Para realizar un estudio profundo en la Capa de Aplicación de las distintas propuestas de investigadas, se realizó un análisis comparativo de las distintas alternativas de desarrollo, tomando en cuenta seis puntos relevantes de análisis que son detallados a continuación:

- Peso total sin uso
- Memoria cache
- Datos almacenados
- Peso final de las aplicaciones
- Tiempos de apertura
- Tiempos de respuesta de Cada Una de las Consultas
- Consumo de datos Por Cada Una de las Consultas

Donde cada una de las soluciones sobresalió en algún aspecto, en algunos casos con diferencias mínimas pero marcando un diferencia a la hora de observar de forma macro el análisis, la Tabla N° 82 muestra quien es más eficiente en cada uno de los puntos estudiados.

	Ionic	Xamarin	Android Studio
Peso total sin uso			✓
Datos almacenados			✓
Memoria Cache		✓	
Peso final de las aplicaciones			✓

Tabla N° 82: Comparación de Puntos

De acuerdo a los datos presentados en la Tabla N°82 Android Studio es la tecnología de desarrollo más eficiente a la hora de desarrollar una aplicación móvil para Android ya que aprovecha de mejor forma los recursos del dispositivo en el que se despliega la solución; Sin embargo existen otro tipo de variables que se deben considerar a la hora de decidir por una alternativa al Desarrollo Móvil en la capa de Aplicación.

Considerando el estudio anterior y la comparación entre cada una de las consultas con respecto a los tiempos de respuesta que toman a la hora de ser ejecutadas se plantea el siguiente cuadro donde cada ✓ selecciona a la alternativa de desarrollo que fue más eficiente.

Como ejemplo para el estudio de Tiempos de apertura se obtuvieron los siguientes resultados promedios:

- **Andorid Studio:** 2 Segundos
- **Xamarin:** 3.08 Segundos
- **Ionic:** 5.3 Segundos

Por lo que a la hora de traspasar estos resultados a la tabla comparativa Android Studio es quien se lleva el ✓.La tabla N° 83 presenta la comparación de las tres alternativas esta vez considerando los Tiempos de Respuesta por cada una de las consultas.

	Android Studio	Xamarin	Ionic
Tiempo Apertura	✓		
Tiempo Login			✓
Tiempo Consulta Horario	✓	✓	✓
Tiempo Consulta Notas		✓	
Tiempo Consulta Inscripcion_1			✓
Tiempo Consulta Inscripcion_2			✓
Tiempo Consulta Profesor_1	✓		
Tiempo Consulta Profesor_2		✓	✓
Tiempo Consulta Lista_1		✓	
Tiempo Consulta Lista_2		✓	

Tabla N° 83: Comparación Tiempos de Respuesta

El mismo criterio empleado anteriormente para la tabla de resumen comparativa entre Tiempos de respuesta por cada una de las plataformas, fue considerado para la tabla comparativa del Consumo de Datos por consulta, que cada tipo de desarrollo obtuvo.

En este caso cada ✓ es dado a la plataforma de desarrollo que menos consumo de datos promedio obtuvo a medida que se realizaron las pruebas.

	Android Studio	Xamarin	Ionic
Login_ Datos Móviles	✓		
Horario_Datos_Móviles			✓
Notas_ Datos Móviles			✓
Inscripcion_1_Datos Móviles	✓		
Inscripcion_2 Datos Móviles			✓
Profesor_1_ Datos Móviles			✓
Profesor_2_ Datos Móviles		✓	
Lista_1_ Datos Móviles			✓
Lista_2_ Datos Móviles			✓

De la tabla anterior podemos concluir que Ionic es quien menos consumo de datos realizó a medida que se realizaron las pruebas.

10 FACTIBILIDAD DE LA PROPUESTA

A manera de introducción, comenzar destacando que en este punto se plantearán 3 ítems que son utilizados para realizar el cálculo de la factibilidad del proyecto propuesto.

1° Ítem: Factibilidad técnica.

La factibilidad técnica permite obtener la información necesaria respecto a, si existe o está al alcance la tecnología necesaria para el sistema a implementar, chequeando si se cuenta con los equipos y programas mínimos para la realización y utilización de éste.

2° Ítem: Factibilidad Operativa.

La factibilidad operativa se refiere a que debe existir personal capacitado para la realización del proyecto en desarrollo y así mismo, colocando mucho énfasis en el nivel de mejora que aportará el proyecto para la Dirección De Informática. Además, deben existir usuarios finales dispuestos a emplear los servicios generados por el sistema desarrollado.

3° Ítem: Factibilidad Económica.

La factibilidad económica determina si el sistema es factible para su realización respecto de los costos asociados al desarrollo y su utilización. Así mismo, se debe probar que los beneficios a obtener con la realización de este sistema son superiores a los costos en que se incurrirá al desarrollo e implementación.

10.1 Factibilidad técnica.

Para la solución propuesta a nivel de capa de aplicación nos encontramos con 3 soluciones propuestas por lo que analizaremos la factibilidad técnica para cada una de ellas.

10.1.1 Android Studio

Al tratarse de una solución basada en un IDE, es por esto que en términos de Hardware y Software, Android Studio tiene requisitos mínimos para ser utilizados:

Estos requisitos mínimos son:

- Windows 7/8/10 (32 o 64 bits).
- 2 GB de RAM (8 GB de RAM recomendado).
- 2 GB de espacio libre mínimo (4 GB recomendado).
- Resolución mínima de 1.280 x 800.
- Java 8.
- 64 bits y procesador Intel (emulador).

10.1.2 Xamarin

Al igual que Andorid Studio Xamarin está basada en un IDE, y utilizar código C# que necesita ser compilado para crear aplicaciones, los requisitos mínimos para crear y compilar una aplicación móvil tienen que ver con el sistema en el que se desplegará y el hardware que se utilizará, además de algunos requisitos adicionales que esta solución de desarrollo requiere.

A nivel de sistema operativo Xamarin requiere de las siguientes especificaciones:

- Windows 10 versión 1507 o superior: Hogar, Profesional, Educación y Empresa
- Windows Server 2016: estándar y centro de datos
- Windows 8.1: Core, Professional y Enterprise
- Windows Server 2012 R2: Essentials, Standard, Datacenter
- Windows 7 SP1: Home Premium, Professional, Enterprise, Ultimate.

Xamarin también tiene requisitos mínimos de Hardware para ser desplegado:

- Procesador de 1.8 GHz o más rápido. Dual-core o mejor recomendado
- 2 GB de RAM; Se recomiendan 4 GB de RAM (mínimo de 2,5 GB si se ejecuta en una máquina virtual)
- Espacio en el disco duro: hasta 130 GB de espacio disponible, dependiendo de las características instaladas; las instalaciones típicas requieren 20-50 GB de espacio libre.
- Tarjeta de video que admite una resolución de pantalla mínima de 720p (1280 por 720); Visual Studio funcionará mejor con una resolución de WXGA (1366 por 768) o superior.

Por último se requieren los siguientes requisitos adicionales para el desarrollo en este IDE:

- Se requieren derechos de administrador para instalar Visual Studio.
- Se requiere .NET Framework 4.5 para instalar Visual Studio. Visual Studio requiere .NET Framework 4.6.1, que se instalará durante la instalación.
- Windows 10 Enterprise LTSB edition y Windows 10 S no son compatibles para el desarrollo. Puede usar Visual Studio 2017 para crear aplicaciones que se ejecuten en Windows 10 LTSB y Windows 10 S.
- Internet Explorer 11 o Edge es necesario para escenarios relacionados con Internet. Algunas funciones podrían no funcionar a menos que estas, o una versión posterior, estén instaladas.
- Para compatibilidad con emuladores, se requieren ediciones de Windows 8.1 Pro o Enterprise (x64). También se requiere un procesador que admita Client Hyper-V y Second Level Address Translation (SLAT).
- Xamarin.Android requiere una edición de Windows de 64 bits y el Java Development Kit (JDK) de 64 bits.
- Se requiere PowerShell 3.0 o superior en Windows 7 SP1 para instalar Mobile Development con cargas de trabajo C ++, JavaScript o .NET.

10.1.3 Ionic

A diferencia de Xamarin, Ionic es un Framework de desarrollo basado en web, por lo que a la hora de desarrollar un proyecto es necesario tener instalado varios componentes que serán especificados a continuación.

Ya que Ionic se encuentra basado en Angular y Cordova estos deben estar instalados previamente, junto con ellos es necesario el software Node.JS que es un intérprete JavaScript que funciona del lado del servidor, que nos permite compilar las aplicaciones desarrolladas en el Framework IONIC y por último SDK de Android necesita los siguientes requisitos de sistema y hardware para ser utilizado:

- Windows 7/8/10 (32 o 64 bits).
- 2 GB de RAM (8 GB de RAM recomendado).
- 2 GB de espacio libre mínimo (4 GB recomendado).
- Resolución mínima de 1.280 x 800.

- Java 8.
- 64 bits y procesador Intel (emulador).

La Dirección de Informática cuenta con los equipos necesarios para la utilización de cualquiera de las tecnologías estudiadas.

10.2 Factibilidad operativa.

Para la Universidad del Bio-Bío y en específico para la Dirección de Informática, la necesidad de establecer una estructura de desarrollo móvil sostenible en el tiempo es un hecho que no puede esperar más.

Teniendo esta información expresada en reuniones y entrevistas con los encargados del departamento anteriormente mencionado, se ha planteado la investigación distintas plataformas de desarrollo móvil con el fin de establecer un nuevo estándar a la hora de implementar un solución móvil dentro de la universidad, lo cual generará aspectos positivos que son descritos a continuación:

- Optimización en tiempos de desarrollo
- Las personas a quienes se les construirá un software dentro de la universidad no tendrán limitación de sistema operativo móvil.

10.3 Factibilidad económica.

En este punto se plantea un caso de estudio real para el cálculo de la factibilidad del proyecto en desarrollo.

- **Caso real**

En este caso se estudiará la factibilidad, basándose en las condiciones actuales que presenta la Universidad.

10.3.1 Caso Real

10.3.1.1 Costos de Personal

La propuesta no estima que se deba realizar un gasto adicional en costos de personal.

10.3.1.2 Costos de Aprendizaje

El desarrollo estará a cargo del personal de la Universidad del Bio-Bío por lo que habrá que capacitarlos en base a cursos de programación para el lenguaje de la solución propuesta.

- **Costos Aprendizaje Android Studio**

En el caso de Android Studio, Google cuenta con cursos para aprender a desarrollar en esta tecnología, específicamente en la página <https://www.udacity.com/course/android-developer-nanodegree-by-google--nd801#>, donde ofrecen el curso que muestra la Figura N°163.



Figura N° 157: Costo Implementación Android Studio

De la Figura anterior obtenemos el valor del curso de Android Studio por 8 meses que asciende a \$1.199 Dólares, esto se traduce en un valor mensual de \$149.88 Dólares, lo que se traduce en un costo mensual de \$90.677 Pesos Chilenos mensuales.

- **Costos Aprendizaje Xamarin**

Xamarin University es la web oficial para aprender a usar este IDE de desarrollo, como lo muestra la Figura N° 164



The screenshot shows the Xamarin University website. On the left is the Xamarin logo, a white 'X' inside a shield with a laurel wreath. To the right, the text reads 'Xamarin University' in a large font, followed by 'Microsoft | Paid'. Below this is a promotional message: 'Go mobile and stay ahead of the competition with live, interactive mobile development training led by Xamarin experts on your schedule.' A green 'Get' button is centered below the text. At the bottom, there is a link: 'Already have a subscription? [Log in to Xamarin University](#)'.

Overview Pricing

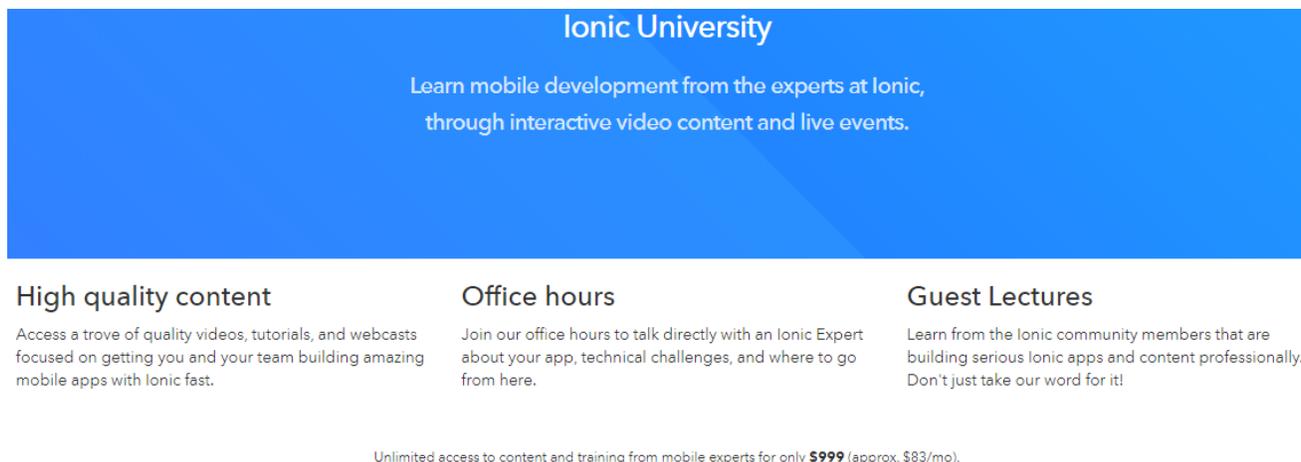
\$83.25/month per subscriber or sign up for a [free trial](#)

Figura N° 158: Costo aprendizaje Xamarin

De la Figura anterior podemos obtener el valor por mes del curso intensivo de Xamarin, que tiene un costo de \$83.24 Dólares que al precio actual se traduce en \$50.408 Pesos Chilenos mensuales. Xamarin entrega acceso a todos los cursos, por lo que el tiempo de duración del curso dependerá exclusivamente de la capacidad de aprendizaje del desarrollador.

- **Costos Aprendizaje Ionic**

Al igual que Xamarin Ionic cuenta con Ionic University que entrega todas las herramientas y conocimientos para utilizar este framework, la Figura N° 165 es una captura de la página oficial.



Ionic University

Learn mobile development from the experts at Ionic, through interactive video content and live events.

High quality content

Access a trove of quality videos, tutorials, and webcasts focused on getting you and your team building amazing mobile apps with Ionic fast.

Office hours

Join our office hours to talk directly with an Ionic Expert about your app, technical challenges, and where to go from here.

Guest Lectures

Learn from the Ionic community members that are building serious Ionic apps and content professionally. Don't just take our word for it!

Unlimited access to content and training from mobile experts for only **\$999** (approx. \$83/mo).

Figura N° 159: Costo aprendizaje Ionic

De la Figura anterior podemos rescatar el costo anual por acceder a todas las plataformas de aprendizaje que entrega Ionic, anualmente su costo es de \$999 Dólares, si lo vemos desde un valor mensual llega a la suma de \$83,25 Dólares que a valor actual del peso chileno son \$50.444 Pesos chilenos.

10.3.13 Costos de software

Teniendo en cuenta que los equipos con los que cuenta la Universidad poseen el hardware necesario y apropiado sin la necesidad de recurrir a gastos extras, pero para el Software es necesario la adquisición de algunas licencias.

Para desarrollar en Xamarin el IDE Visual Studio requiere de una licencia de pago que tiene un costo de \$453.219 por equipo, al contrario Ionic se desarrolla bajo licencia de software libre permisiva del MIT, por su parte Android Studio también cuenta con licencia libre, Lo que significa que puede ser utilizado tanto para su uso personal como para su uso comercial estos datos están representados en la Tabla N° 84.

Plataforma	Costo Licencia
Xamarin	\$453.219
Ionic	\$0
Andorid Studio	\$0

Tabla N° 84: Costos de Software

10.3.14 Costos de Publicar aplicación

En la plataforma Andorid, GooglePlay es quien regula la publicación de apps y esto tiene un costo asociado de \$25 Dólares, que ha precio actual del dólar son \$15.830 Pesos Chilenos, los cuales se pagan una sola vez.

En la plataforma IOS el costo de la licencia de desarrollador tiene un costo de \$99 Dólares, que ha precio actual asciende a unos \$62.685 Pesos Chilenos.

10.3.2 Costo total de implementación

El costo total del proyecto será determinado por la elección de plataforma para desarrollar y sus datos están expresados en la Tabla N°85.

	Xamarin	Andorid Studio	Ionic
Costo Aprendizaje (Costo Total Curso)	\$50.408	\$759.184	\$632.548
Duración	1 Mes	12 Meses	8 meses
Costo Software	\$453.219	\$0	\$0
Costo Total	\$582.142	\$837.699	\$711.063

Tabla N° 84: Costo Total De Implementación

Para realizar un estudio completo de la viabilidad del proyecto, se analizaron tres ítems sumamente importantes que nos entregan resultados efectivos y concretos para determinar dicha factibilidad.

Primero, se analizó la factibilidad técnica del proyecto, en donde se consideraron los requerimientos necesarios, tanto en el hardware como en el software. Los resultados obtenidos son positivos, ya que la Universidad cuenta con el software y hardware necesarios para la realización de los proyecto, teniendo que incurrir sólo en gastos correspondientes a la capacitación del personal y en donde la Universidad está dispuesta a cubrir dichos gastos.

Posteriormente, se analizó la factibilidad operacional del proyecto, en donde los resultados son gratamente esperados, ya que la administración en reuniones y entrevistas presentan

una buena aceptación con respecto a las alternativas de desarrollo y muchas expectativas con la nueva forma en que realizará su trabajo, en donde se explica que las soluciones propuestas son una nueva forma de enfrentar y dar solución al desarrollo de aplicaciones móviles.

Finalmente se analizó la factibilidad económica de la propuesta identificando cada uno de los gastos que la universidad tendría que costear a la hora de elegir una de las alternativas propuestas en este documento.

Concluir comentando que el estudio realizado para saber si el proyecto es factible de realizar nos entrega finalmente resultados positivos en los tres ítems, teniendo por conclusión que la *“Propuesta de Arquitectura de Sistemas para el desarrollo de Aplicaciones Móviles en la Dirección De Informática de la Universidad del Bío-Bío”*, cumple con los requisitos necesarios para su realización

11 ESTRUCTURA DE LA PROPUESTA DE ARQUITECTURA.

Considerando el estudio realizado en los capítulos anteriores, la selección para la estructura que compondrá la Arquitectura de Sistemas Para el Desarrollo de Aplicaciones Móviles ha sido establecida en base a las necesidades que se entregaron en durante la fase de entrevistas y conversaciones, la cual pretende dar una solución, que permita a los desarrolladores optimizar los tiempos de desarrollo a la hora de implementar una solución móvil para la Universidad.

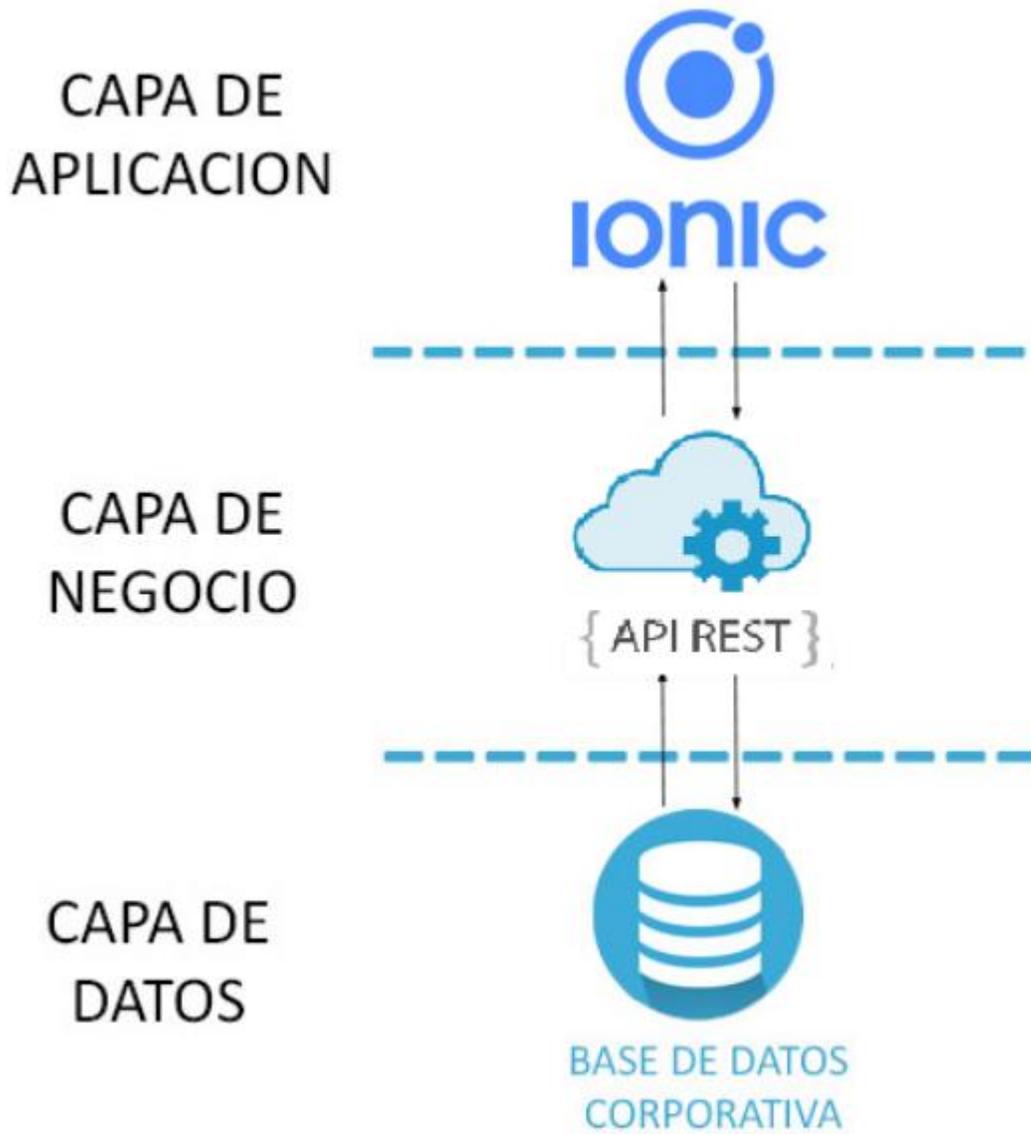
Por lo anterior la elección de una alternativa para la capa de aplicación, se basa en el estudio y comparación de resultados entregados por las pruebas que fueron realizadas durante el transcurso de esta investigación.

La problemática a afrontar en esta capa es la optimización de los tiempos de desarrollo ya que al utilizar un desarrollo nativo las aplicaciones deben ser desarrolladas, una por cada plataforma en la que se desea desplegar.

La solución elegida para la capa de aplicación es la implementación de Ionic Framework, quien si bien en las pruebas anteriores no fue quien más destaque, su rendimiento a la hora de desplegar datos provenientes desde un servidor es prácticamente equivalente a los resultados que se obtuvieron de la aplicación desarrollada nativamente, además al estar basado en un ambiente web, su curva de aprendizaje es mucho menor a la que se obtiene realizando un desarrollo nativo.

Finalmente en la capa Negocio y la Capa de Datos no nos adentraremos ya que la infraestructura con la que se trabaja contiene datos confidenciales a los que un alumno no puede acceder.

Como resultado del análisis anteriormente mencionado la Arquitectura propuesta para la Dirección de informática de la Universidad Del Bío-Bío se encuentra desplegada en la siguiente Figura.



12 CONCLUSIONES

La creación de una Propuesta de Arquitectura para el desarrollo de Aplicaciones Móviles en la Dirección De Informática de la Universidad del Bio-Bío para la cual se realizó este proyecto informático, ha sido un trabajo arduo y de constante estudio.

Se puede afirmar que se logró investigar las mejores prácticas desde el punto de vista de desarrollo; se desarrollaron distintas aplicaciones Móviles con el objetivo de comparar distintas tecnologías. Se estudiaron tres alternativas, la primera el desarrollo nativo que aprovecha al máximo los recursos de cada uno de los dispositivos en los que se despliegan ya que están escritos en el mismo lenguaje, y en segundo lugar Ionic y Xamarin que tienen como ventaja un desarrollo multiplataforma.

En lo personal trabajar con el Framework Ionic, con quien tuve una muy grata experiencia a la hora de desarrollar la aplicación, ya que al ser un Framework ofrece un código estructurado y una larga lista de características como su estructura basada en HTML CSS y JavaScript que hace más fácil el desarrollo.

Puedo afirmar que única desventaja que puede tener Ionic frente a las otras herramientas de desarrollo, es su ligera lentitud a la hora de iniciar y desplegarse en el dispositivo, sin embargo una vez abierta no tiene nada que envidiar a las aplicaciones que se desarrollan en forma nativa ya que su rendimiento va a la par con ellas. Particularmente en mi caso, el tiempo que me tomo desarrollar la aplicación en Ionic no supera una 1/6 parte del tiempo que me tomo desarrollar en Xamarin y Android Studio juntos. Es por esta razón que considero que la mejor opción para el desarrollo en la capa de aplicación es el “Framework Ionic”.

Con todo lo anterior se propone una arquitectura que permite el desarrollo de aplicaciones móviles; Esta es la Arquitectura de Tres capas con la que la universidad ya trabajaba, pero con variantes en su capa de aplicación, con el uso del Framework Ionic como plataforma de desarrollo.

En lo personal fue un gran desafío, ya que cada uno de los softwares desarrollados para las pruebas realizadas, fueron construidos en un principio con un total desconocimiento, debido a que durante mi formación nunca se había trabajado en aplicaciones móviles, lo que significa que todos conocimientos necesarios para crear estas fueron adquiridos en la medida que se investigaba cada una de las alternativas investigadas.

13 BIBLIOGRAFÍA

Alvarez, M. (2 de Marzo de 2017). *Desarrollo Web*. Obtenido de DesarrolloWeb.com: <https://desarrolloweb.com/articulos/que-es-ionic2.html>

Alvarez, M. A. (19 de 12 de 2014). *Desarrollo Web*. Obtenido de Ventajas e inconvenientes de API REST para el desarrollo: <https://desarrolloweb.com/articulos/ventajas-inconvenientes-apirest-desarrollo.html>

Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice, Second Edition*. Addison Wesley.

CA Technology. (2015). *Arquitectura y Estrategia de Una Api Un Enfoque Coordinado*. Obtenido de CA Technology: <https://www.ca.com/content/dam/ca/es/files/ebook/api-strategy-and-architecture-a-coordinated-approach.pdf>

Cruz, M. (03 de 08 de 2014). *QodeBlog*. Obtenido de ¿Qué es una App Nativa?: <http://Qode.pro/blog/que-es-una-app-nativa>

Fowler, M. (2002). *Patterns of Enterprise Application Architecture*. Addison Wesley.

Gallego, A. (15 de Enero de 2016). *Manual de introducción a Ionic*. Obtenido de Introducción a Ionic: <https://www.gitbook.com/book/ajgallego/ionic/details>

Microsoft. (2001). *Microsoft Application Architecture Guide*.

Xamarin Inc. (03 de Junio de 2017). *Xamarin*. Obtenido de Introduction to Mobile Development Xamarin: https://developer.xamarin.com/guides/cross-platform/getting_started/

Zamora, J. A. (12 de 05 de 2014). *El Androide Libre*. Obtenido de El Androide Libre: <https://elandoide Libre.espanol.com/2014/05/xamarin-la-api-para-crear-aplicaciones-multiplataforma-en-c-net.html>

14 ANEXO N°1: GUIA PARA EL DESARROLLO DE APLICACIONES

Actualmente la Dirección de Informática de la universidad del Bío-Bío no cuenta con una guía que establezca estándares a la hora de comenzar un desarrollo móvil. Este trabajo tiene como objetivo dar a conocer las distintas formas de cómo comenzar un proyecto de móvil según las habilidades del Desarrollador.

Además se explicara brevemente como ambientar el entorno de desarrollo, según el tipo de proyecto que se desea crear. Con esto estableceremos una breve guía para desarrollo de aplicaciones móviles.

Esta Guía está basada en el Proyecto de Título “Propuesta de Arquitectura para el desarrollo de Aplicaciones Móviles En La Dirección De Informática de la Universidad del Bío-Bío”.

14.1 Elección De tecnología de Desarrollo

Un punto importante a la hora de comenzar cualquier proyecto es conocer las fortalezas y debilidades que posee el equipo de trabajo, además establecer el tipo de desarrollo con el que se desea trabajar.

Antes de entrar en detalles de cómo realizar un desarrollo móvil debemos tener en consideración el tipo de tecnología sobre la cual se trabajara. Hoy en día existen muchas herramientas que permiten el desarrollo de aplicaciones. Concretamente se debe elegir entre 3:

- Desarrollo Nativo
- Desarrollo Multiplataforma Compilado a nativo
- Desarrollo Multiplataforma basado en Tecnologías Web

14.2 Desarrollo Nativo

Este tipo de desarrollo se realiza mediante lenguajes de programación específicos para cada plataforma en la que se desea desplegar la aplicación.

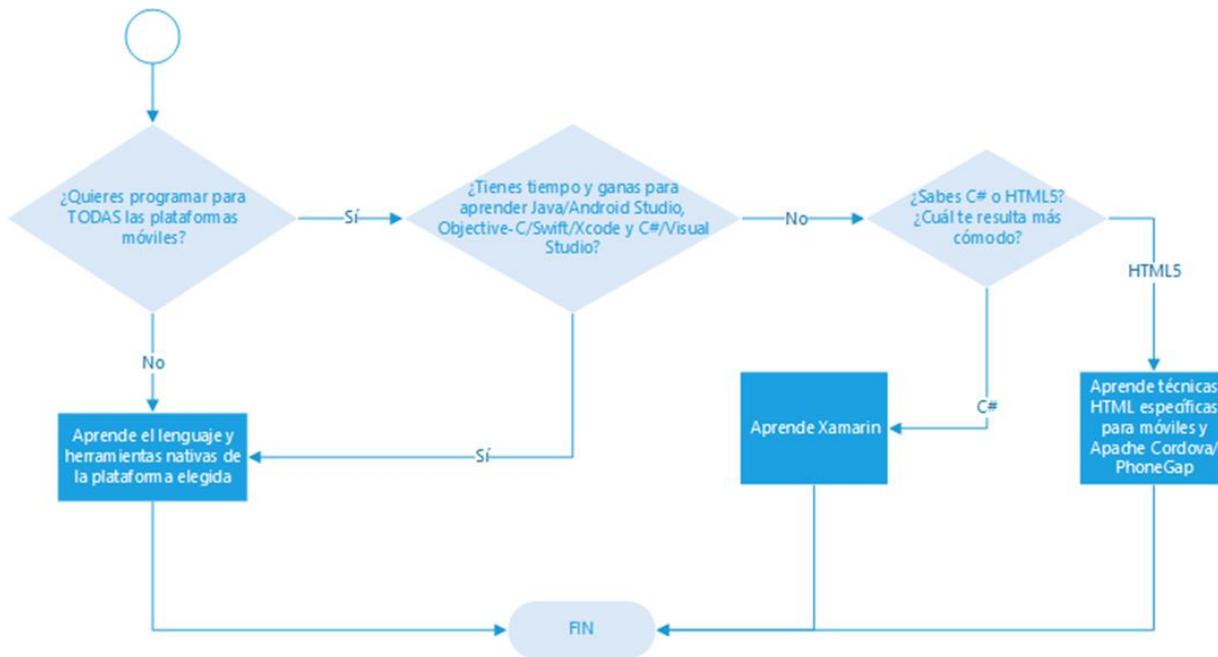
14.3 Desarrollo Multiplataforma Xamarin

Como se nombra anteriormente este tipo de desarrollo es multiplataforma y su desarrollo se realiza en código C#.

14.4 Desarrollo Multiplataforma Ionic Framework

Al igual que el punto anterior este desarrollo es multiplataforma y su desarrollo está basado en una plataforma Web.

Con estos puntos claros podemos elegir el tipo de tecnología de desarrollo con que se construirá la aplicación, una sugerencia para la elección de esta se encuentra en el siguiente diagrama⁹.



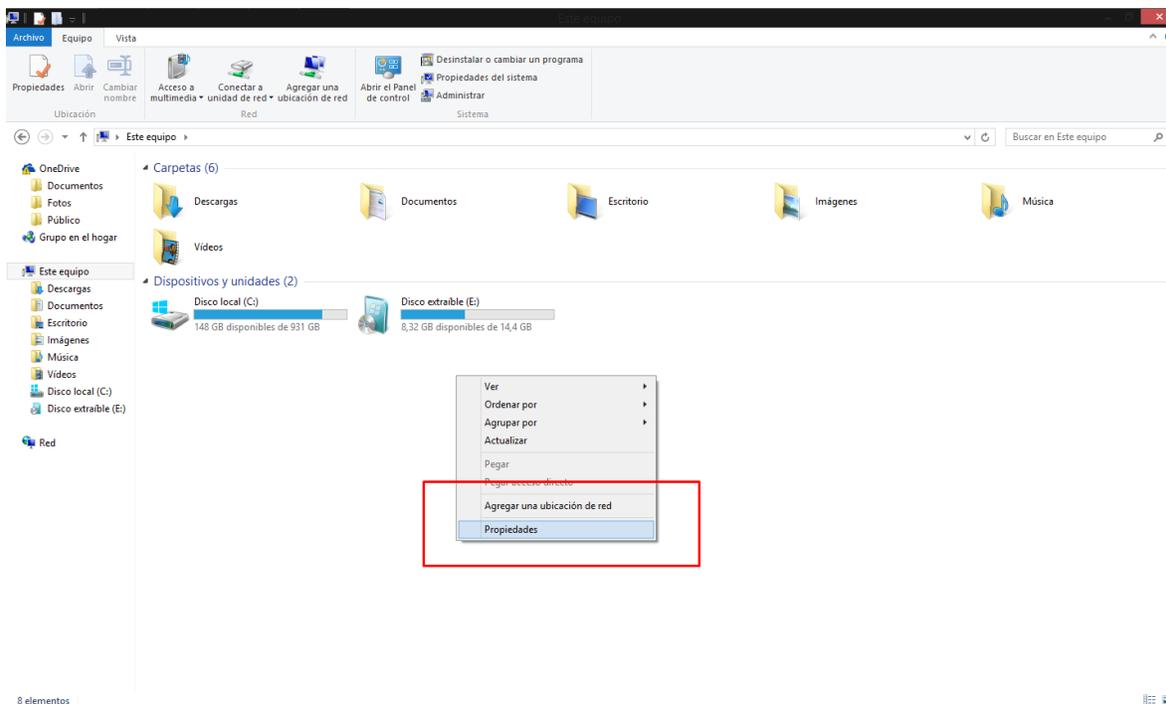
14.5 Configuración para Desarrollo Nativo Andorid

Para desarrollar nativamente en Andorid es necesario instalar Java, para esto nos dirigimos a la página <http://www.oracle.com/technetwork/es/java/javase/downloads/index.html> y seleccionamos la versión de java con la que trabajaremos.

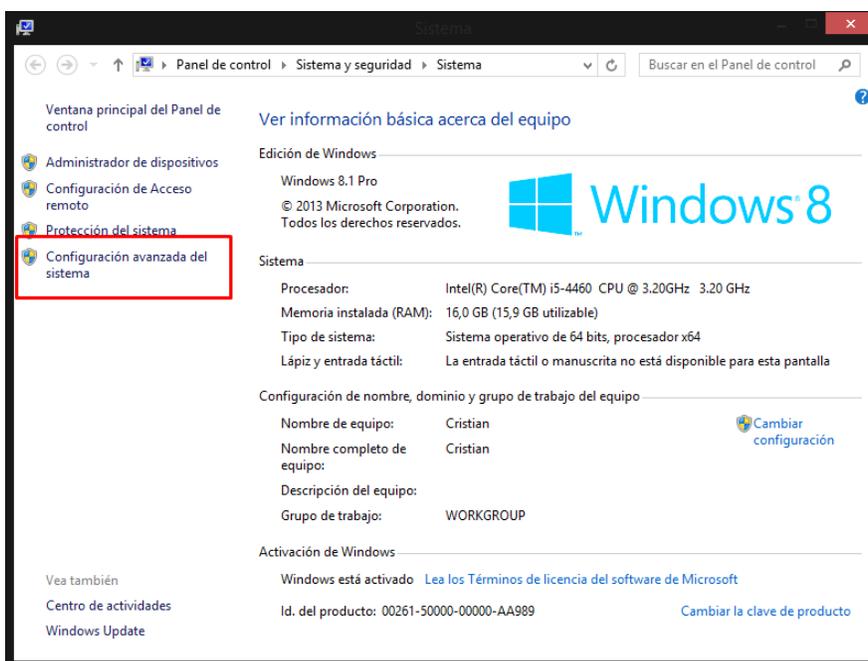


⁹ Diagrama obtenido de https://www.campusmv.es/recursos/image.axd?picture=Elegir-Entorno-Programacion-Movil_1.png

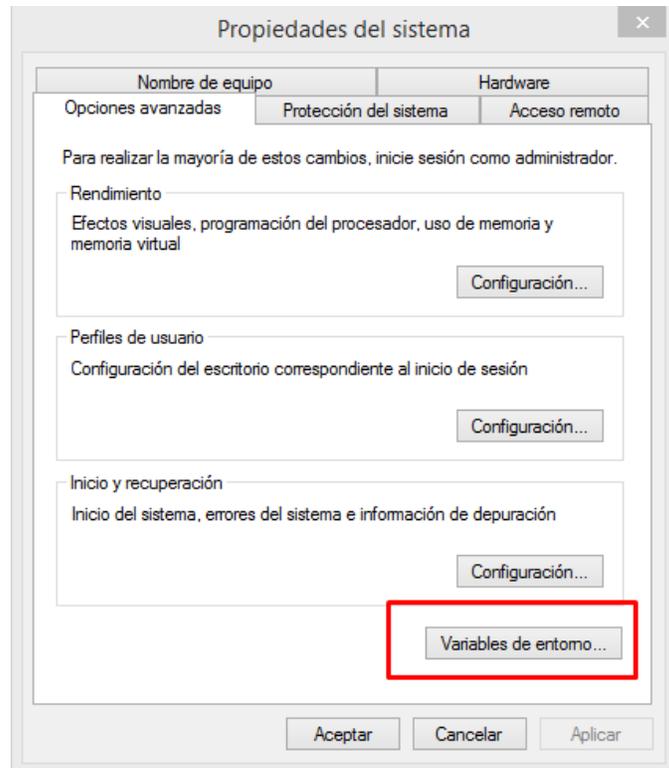
Una vez descargado e instalado, se procede a configurar la variables de entorno, para ello nos dirigimos al equipo, damos Click derecho y seleccionamos propiedades.



Esta opción nos desplegará una ventana, en la cual seleccionaremos “Configuración Avanzada del Sistema”.



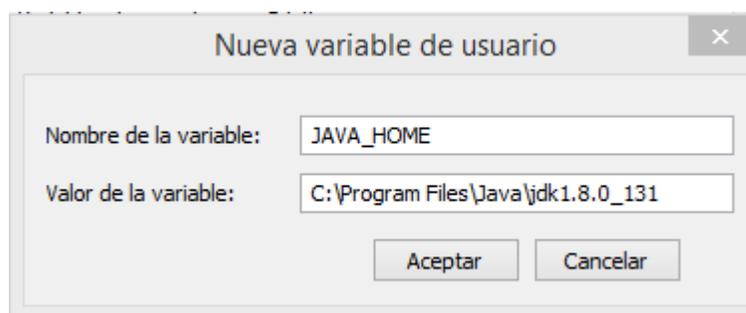
Donde se abra la siguiente pestaña y accederemos a la opción de Variables de Entorno.



Una vez dentro agregamos una nueva variable con los siguientes parámetros,

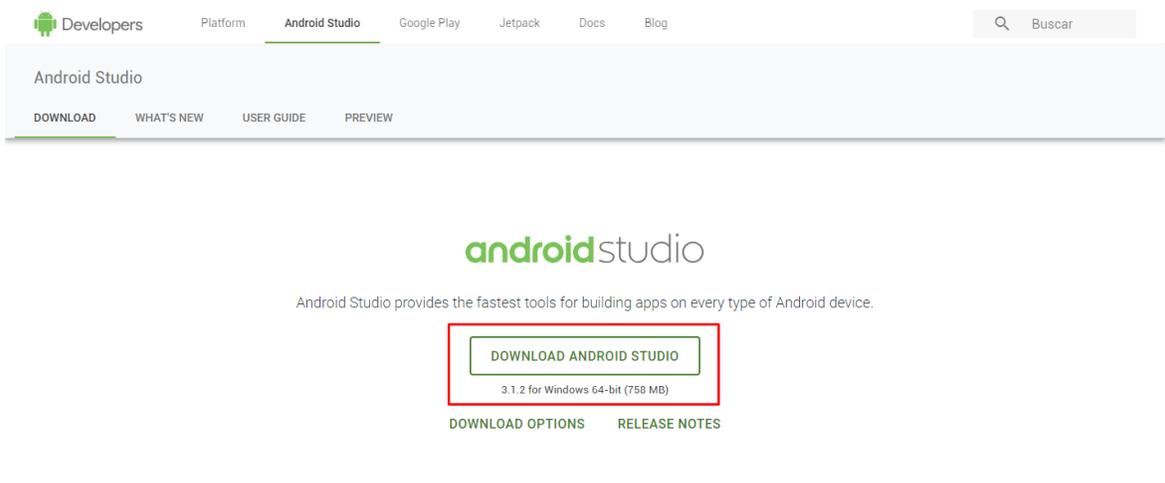
Nombre de la Variable: JAVA_HOME

Valor de la variable: “Direccion donde se Instaló el archivo Java anteriormente Descargado”

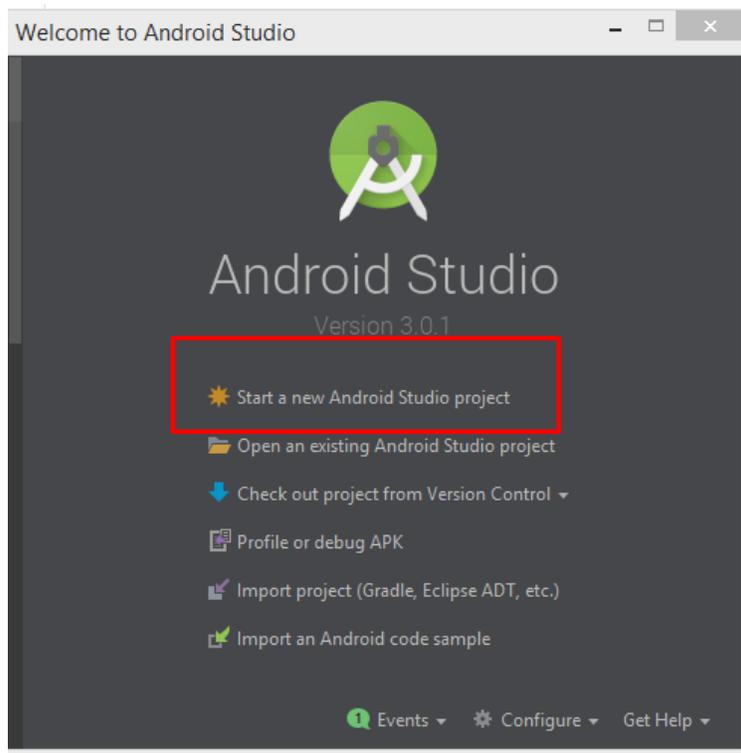


Damos aceptar en todas las pestañas que fueron desplegadas y con esto ya tenemos ambientado el entorno JAVA dentro de nuestra computadora.

Luego es esto debemos descargar el entorno de trabajo que en este caso en “AndoridStudio”, para ello nos dirigimos a la página <https://developer.android.com/studio/>.



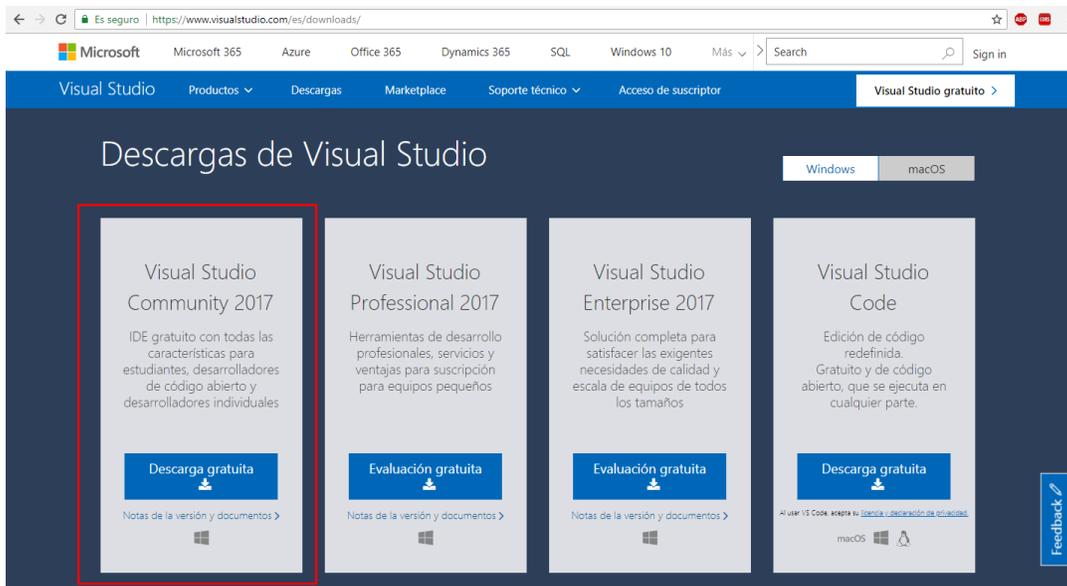
Una vez descargado procedemos a su Instalación. Una vez Instalado, iniciamos la aplicación y nos mostrara lo siguiente



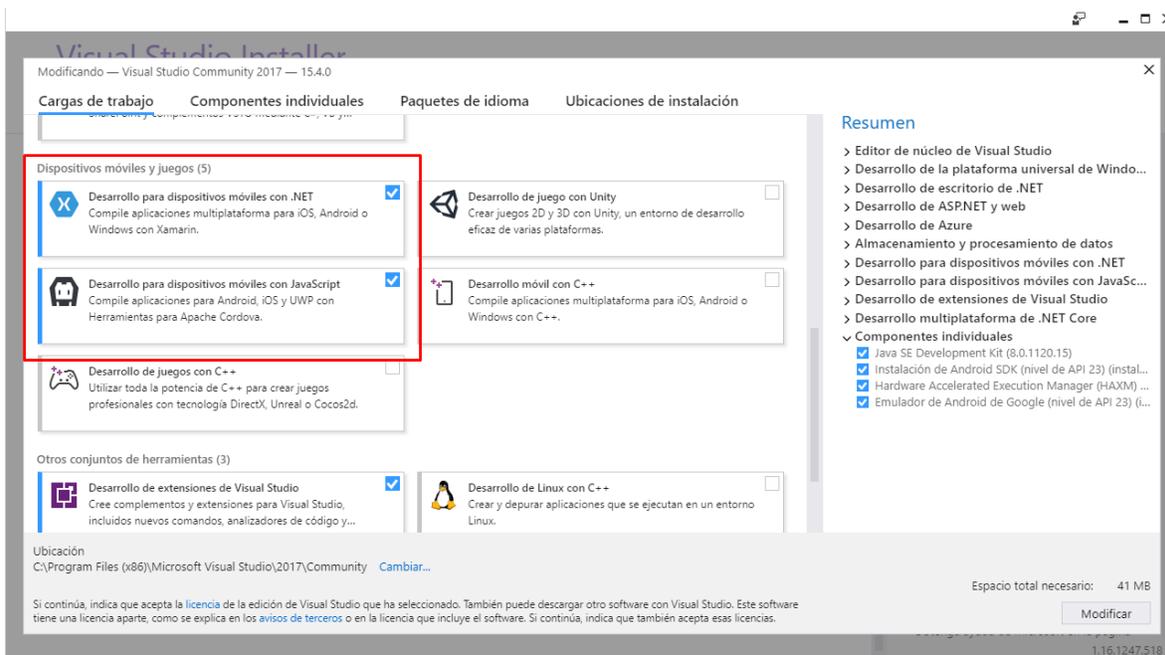
Damos Inicio a un Nuevo Proyecto Android y ya se puede comenzar a desarrollar aplicaciones nativas para esta plataforma.

14.6 Configuración para Desarrollo Multiplataforma con Xamarin

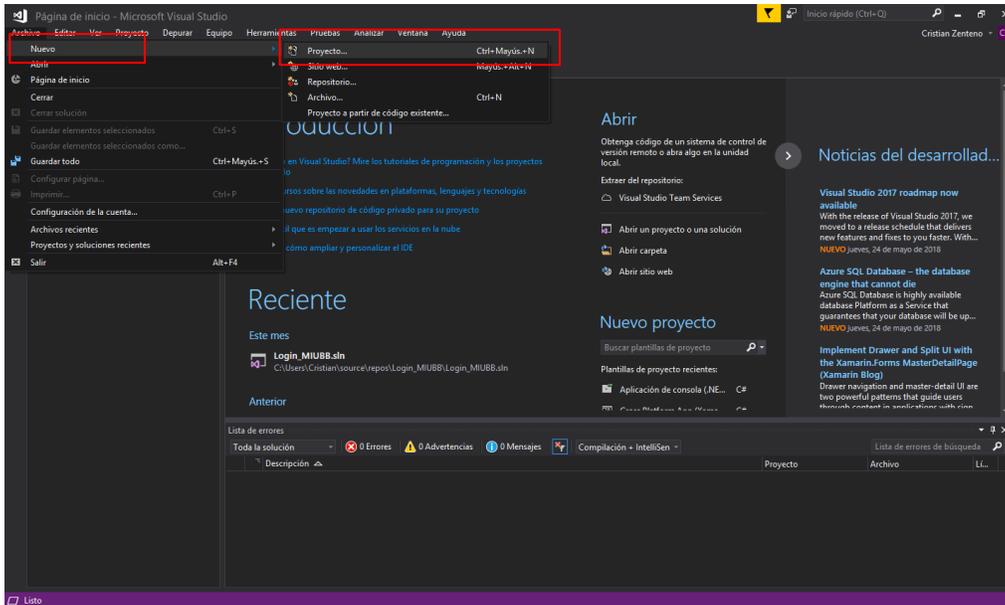
Para comenzar a utilizar Xamarin se debe instalar la herramienta Visual Studio que se descarga a través de este link <https://www.visualstudio.com/es/downloads/>



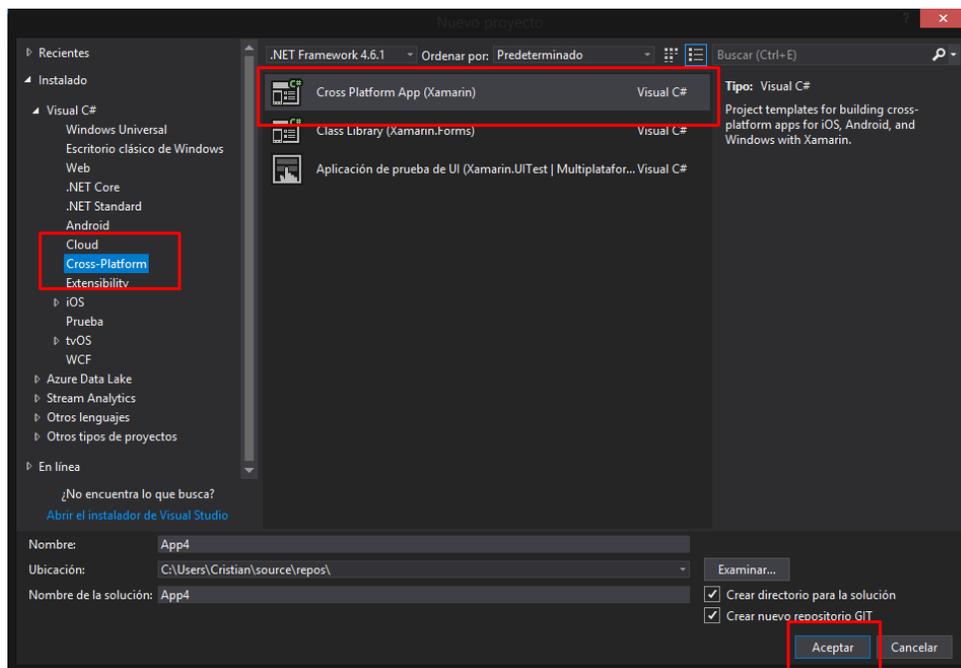
Una vez instalado se deben seleccionar las características necesarias para poder desarrollar con Xamarin.



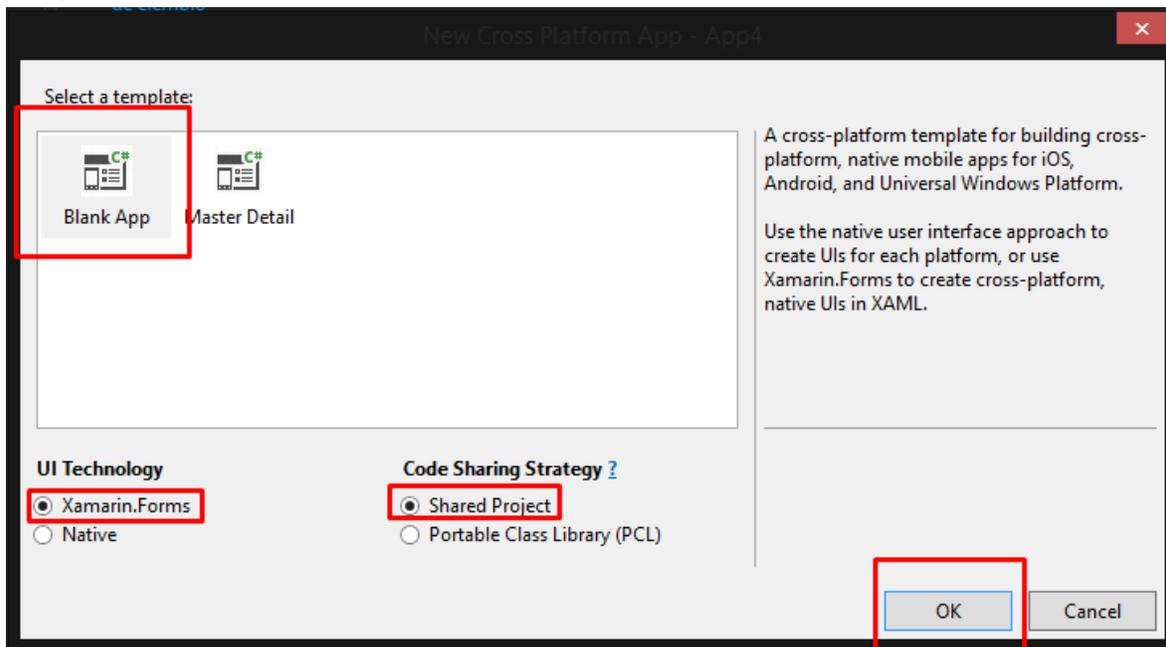
Para crear un nuevo proyecto en Xamarin se deben seleccionar las opciones especificadas en la siguiente imagen.



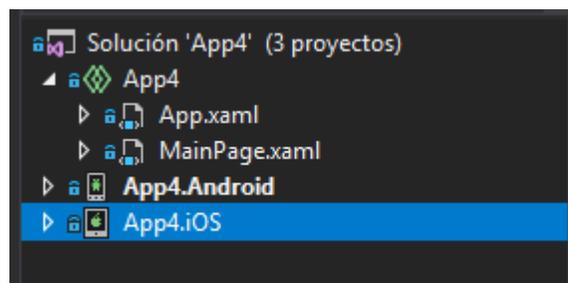
Se selecciona la forma en la que se desea programar, en este caso Multi Plataforma.



Una vez aceptado el paso anterior nos llevara a la siguiente ventana donde seleccionamos las siguientes alternativas y damos a OK.



Con esto se crearan las dependencias necesarias para el desarrollo multiplataforma. Donde obtendremos 3 proyectos simultáneos, para este ejemplo App4 es quien contiene el código compartido, App4.Android y App4.iOS son los proyectos que pueden ser compilados para probar el código desarrollado en la parte Multiplataforma del proyecto.

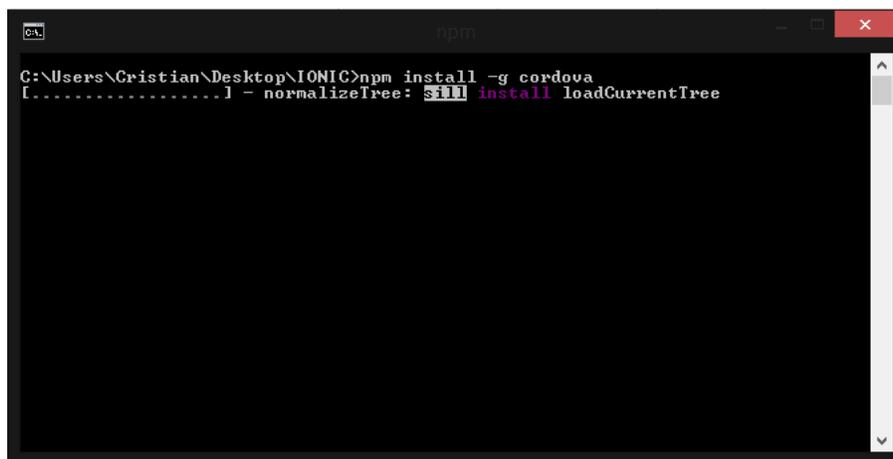


14.7 Configuración para Desarrollo Multiplataforma con IONIC

Para poder instalar Ionic debemos instalar Node JS y Cordova, Node JS para poder trabajar con su paquete de instalaciones \$npm,y Cordova para poder desplegar nuestras soluciones en los diferentes dispositivos móviles, por lo que debemos dirigirnos a la página oficial de NodeJS el <https://nodejs.org/es/>



Una vez instalado podremos instalar Frameworks a través de \$npm,para ello ingresamos a NodeJS e instalamos Cordova, mediante una línea de comandos \$npm install –g cordova



Una vez terminada la Instalación de Cordova deberemos tener algo como esto

```

C:\Windows\system32\cmd.exe

+-- aws4@1.7.0
+-- har-validator@2.0.6
+-- commander@2.15.1
+-- is-my-json-valid@2.17.2
+-- is-my-ip-valid@1.0.0
+-- http-signature@1.1.1
+-- sshpk@1.14.1
+-- mime-types@2.1.18
+-- mime-db@1.33.0
+-- stringstream@0.0.6
+-- underscore@1.8.3
+-- insight@0.8.4
+-- tough-cookie@2.3.4
+-- update-notifier@0.5.0
+-- latest-version@1.0.1
+-- package-json@1.2.0
+-- got@3.3.1
+-- duplexify@3.6.0
+-- lowercase-keys@1.0.1
+-- registry-url@3.1.0
+-- rc@1.2.7
+-- deep-extend@0.5.1

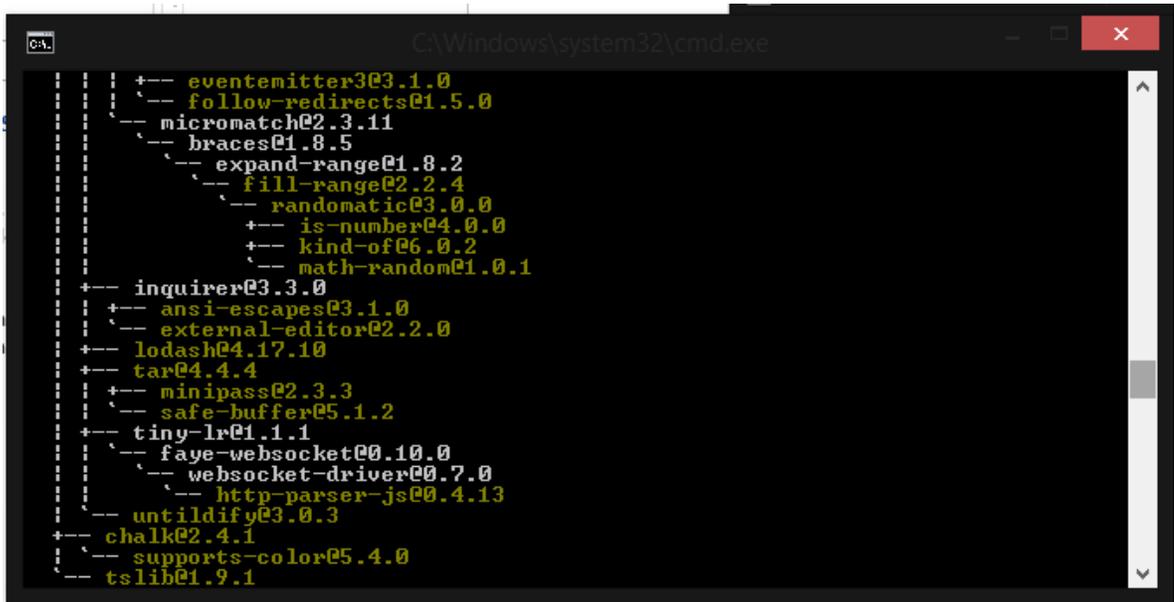
C:\Users\Cristian\Desktop\IONIC>
    
```

Como último paso de Instalación ejecutamos el comando \$npm install -g Ionic con lo que comenzara la instalación de Ionic.

```

C:\Users\Cristian\Desktop\IONIC>npm install -g ionic
[.....] \ fetchMetadata: sill mapToRegistry uri https://registry.n
    
```

Una vez finalizada la instalación nos debe arrojar una pantalla como la siguiente



```

C:\Windows\system32\cmd.exe

+-- eventemitter3@3.1.0
+-- follow-redirects@1.5.0
+-- micromatch@2.3.11
+-- braces@1.8.5
+-- expand-range@1.8.2
+-- fill-range@2.2.4
+-- randomatic@3.0.0
+-- is-number@4.0.0
+-- kind-of@6.0.2
+-- math-random@1.0.1
+-- inquirer@3.3.0
+-- ansi-escapes@3.1.0
+-- external-editor@2.2.0
+-- lodash@4.17.10
+-- tar@4.4.4
+-- minimpass@2.3.3
+-- safe-buffer@5.1.2
+-- tiny-lr@1.1.1
+-- faye-websocket@0.10.0
+-- websocket-driver@0.7.0
+-- http-parser-js@0.4.13
+-- untildify@3.0.3
+-- chalk@2.4.1
+-- supports-color@5.4.0
+-- tslib@1.9.1
  
```

Luego de todo este proceso de instalación se puede comenzar a utilizar este Framework, Ionic tiene la ventaja de tener todo muy bien documentado, es por esta razón que se deja la siguiente dirección <https://ionicframework.com/getting-started> donde se encuentran todas las instrucciones para poder crear una app con este Framework.

14.8 Conclusiones

Con respecto a las conclusiones de esta guía se puede afirmar que se especificaron todos los pasos necesario para poder comenzar el desarrollo de aplicaciones en alguna de las plataformas anteriormente mencionadas, según las fortalezas que cada desarrollador pueda tener, para esto se utilizó un diagrama de flujo que daba opciones de desarrollo según las respuestas que se le entregaban.

15 ANEXO N°2: ESTUDIO CAPA DE NEGOCIO

La capa de negocio es una parte fundamental dentro de la Arquitectura de 3 Capas pues es quien se encarga de comunicar la Capa de Aplicación con la Capa de Datos, GraphQL es un nuevo lenguaje de comunicación entre capas, y es una nueva solución a la hora de conectar el FrontEnd con el BackEnd.

15.1 GraphQL

15.1.1 ¿Qué es GraphQL?

Podemos definir a GraphQL como un lenguaje de consulta, fue creado por el equipo de aplicaciones móviles de Facebook, ellos necesitaban una API de extracción de datos que fuera lo suficientemente potente para describir todo el contenido que implicaba Facebook, pero que además fuera sencilla y fácil de aprender para los desarrolladores. Por tanto Desarrollaron GraphQL para suplir esta necesidad. En el año 2015 comenzaron el proceso para liberar el código mediante la redacción de una especificación, la liberación de una aplicación de referencia y la formación de una comunidad que gira en torno a esta API. Para simplificar un poco más, la figura N° 13 representa la interfaz de comunicación en base a una Api GraphQL

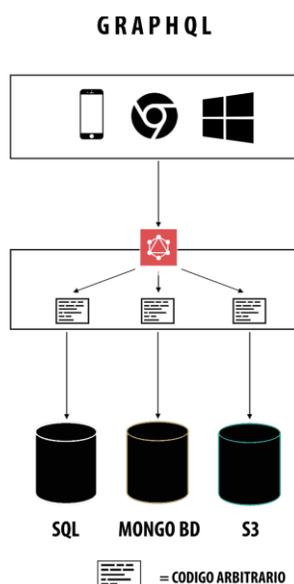


Figura N° 160: Arquitectura GraphQL

La Figura anterior ejemplifica el funcionamiento de GraphQL quien estructura las consultas al servidor, dando acceso directo a los datos guardados en los servidores.

Una Consulta en este formato de API es una cadena que se envía al servidor para que se interprete y ejecute, devolviendo un JSON al cliente. Podemos clarificarlo de mejor forma con el siguiente ejemplo representado en la Figura N° 14 que simula la solicitud de un amigo dentro del Api de Facebook.

Consulta

```
{
  usuario(id: 4802170) {
    id
    nombre
    Sona migos
    ImagenDePerfil (Tamaño:
50) {
    uri
    Ancho
    Alto
  }
  Amigos (Primeros: 1) {
    TotalAmigos
    Amigos {
      id
      Nombre
    }
  }
}
```

Respuesta

```
"Datos": {
  "Usuario": {
    "id": "4802170",
    "nombre": "Lee Byron",
    "son amigos": true,
    "ImagenDePerfil": {
      "uri":
"cdn://pic/4802170/50",
      "ancho": 50,
      "alto": 50
    },
  },
  "Amigos": {
    "TotalAmigos": 14,
    "Amigos": [
      {
        "id": "305249",
        "name": "Stephen Schwink"
      }
    ]
  }
}
```

Figura N° 161: Modelo Consulta GraphQL

15.1.2 Principales Características de GraphQL

Con GraphQL se puede acceder a distintos recursos en una única llamada. Se elimina el concepto de endpoints¹⁰, sustituyéndolo por objetos en los que se indica solo lo que se necesita. Se intenta eliminar el versionado de las API's pudiendo deprecarse el uso de atributos. GraphQL desacopla la dependencia con el acceso a las bases de datos.

• *Define Una Forma De Datos*

Las respuestas que entrega GraphQL son intuitivas, por lo que hace fácil predecir la forma con la que se retornaran los datos, esto facilita aprender y utilizar las herramientas que se entregan para el desarrollo

• *Jerárquico*

Se sigue la semántica de las relaciones entre los objetos de una base de datos, un servicio Rest se puede requerir múltiples viajes de ida y vuelta o una declaración compleja en SQL para obtener los datos, en cambio en GraphQL la jerarquía de datos está dada por la combinación y estructura que se le da a los datos

• *Fuertemente Tipada*

Cada nivel de consulta corresponde a un tipo particular de datos, cada uno de estos tipos describe un conjunto de campos disponibles, lo que lo hace similar a SQL, esto permite proporcionar mensajes descriptivos antes de ejecutar una consulta, lo que favorece su desarrollo en entornos nativos como lo es en el caso de JAVA. En resumidas cuentas, este lenguaje puede asegurar que la consulta sea sintácticamente correcta y válida, antes de ser ejecutada. Esto facilita el desarrollo de aplicaciones por parte del cliente.

¹⁰ Interface a través de la cual los sistemas externos pueden enviar y recibir mensajes al sistema interno

15.2 Implementación y prueba de Tecnologías de la Capa de Negocio

Para poder comparar las diferentes tecnologías de comunicación se crean dos aplicaciones la primera basada en API Rest y la segunda en la Api GraphQL con el fin de entregar una opinión con respecto a la utilización de nuevas tecnologías de comunicación entre capas.

15.2.1 Aplicación Rest

Para la aplicación desarrollada con Api Rest se utilizara un diagrama de caso de uso que nos permite conocer la actividad que realiza nuestro software

15.2.1.1 Caso de uso

La Figura N° 159 representa la función que la aplicación genera para el ingreso a un servidor

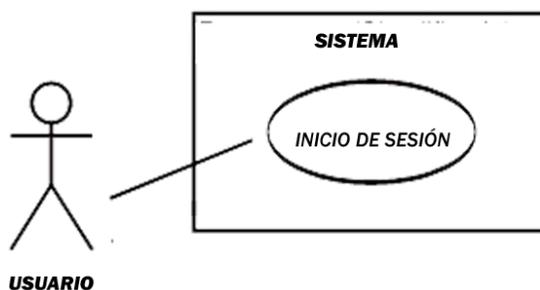


Figura N° 162: Inicio Sesión

15.2.1.2 Descripción de actores

Actor: USUARIO.

Rol: Puede tener acceso a la aplicación y acceder a las opciones que esta entrega.

Nivel de conocimiento técnico requerido: Conocimientos a nivel de usuario en aplicaciones funcionales de oficina y en específico navegación web y móvil.

15.2.13 Especificación de caso de uso

- **Caso de Uso: <Inicio de Sesión>**

Descripción: Descripción general del Caso de Uso

Pre-Condiciones: El usuario debe tener instalada y abierta la aplicación móvil

Flujo de Eventos Básicos:

Al actor	El sistema
1.- Este caso de uso comienza cuando el usuario ingresa al sistema (Aplicación móvil) a través de un dispositivo móvil	2.- El sistema muestra el formulario para el inicio de sesión
3.- El usuario ingresa los datos, Usuario y Contraseña, en el formulario y presiona el botón "Iniciar Sesión".	4.- El sistema Valida que los datos ingresados serán correctos
	5.- El sistema Muestra la pantalla principal para el usuario

Flujo de Eventos Alternativo:

Al actor	El sistema
3(a) El Usuario Ingresa datos de inicio de Sesión erróneos	4(a) El sistema Valida que los datos ingresados están correctos.
	4(b) El sistema no verifica los datos de la cuenta.
	4(c) El Sistema notifica que los datos para el inicio de sesión son erróneos y solicita su ingreso nuevamente.

15.2.14 Modelado de Datos

La Figura N° 160 representa el modelado de datos para la aplicación desarrollada

users	
Id	int
username	varchar(64)
password	text

Figura N° 163: Modelado de datos

15.2.2 Aplicación GraphQL

El desarrollo de esta aplicación se realizó en Android Studio con acceso a la capa de negocios que nos entrega Facebook con el fin de probar la interfaz de comunicación GraphQL a continuación se presenta el desarrollo del software en el que se implementó esta tecnología.

15.2.2.1 Caso de uso

La Figura N° 161 representa el caso de uso que se desarrolló para la comparación

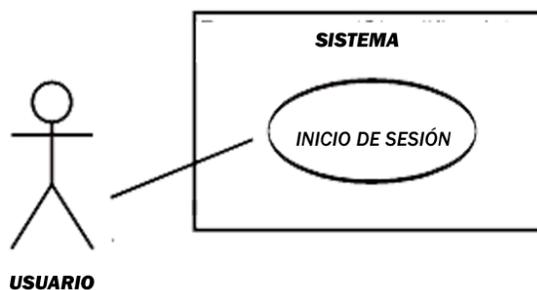


Figura N° 164: Inicio Sesión GraphQL

15.2.2.2 Descripción de actores

Actor: USUARIO.

Rol: Puede tener acceso a la aplicación y acceder a las opciones que esta entrega.

Nivel de conocimiento técnico requerido: Conocimientos a nivel de usuario en aplicaciones funcionales de oficina y en específico navegación web y móvil.

15.2.2.3 Especificación de caso de uso

- **Caso de Uso: <Inicio de Sesión>**

Al actor	El sistema
1.- Este caso de uso comienza cuando el usuario ingresa al sistema (Aplicación móvil) a través de un dispositivo móvil	2.- El sistema muestra el formulario para el inicio de sesión mediante Facebook
3.- El usuario ingresa los datos, Usuario y Contraseña, en el formulario y presiona el botón “Iniciar Sesión”.	4.- El sistema Valida que los datos ingresados serán correctos
	5.- El sistema Muestra la pantalla principal para el usuario

Flujo de Eventos Alternativo:

Al actor	El sistema
3(a) El Usuario Ingresa datos de inicio de Sesión erróneos	4(a) El sistema Valida que los datos ingresados están correctos.
	4(b) El sistema no verifica los datos de la cuenta.
	4(c) El Sistema notifican que los datos para el inicio de sesión son erróneos y solicita su ingreso nuevamente.

15.2.2.4 Modelado de Datos

La figura N° 162 muestra la tabla proporcionada por Facebook para la implementación de aplicaciones en base a su API GraphQL

Profile	
Id	int
Email	e-mail
password	text
Firstname	Varchar(32)
Lastname	Varchar(32)
Linkuri	Varchar(32)
Middlename	Varchar(32)
Name	Varchar(32)
ProfilePictureUri	Varchar(32)

Figura N° 165: Modelado de Datos

15.2.3 Conclusión en la capa de negocio

Con respecto al análisis de en la capa de negocio, se desarrolló una aplicación en Android que utiliza la API'S de los servidores de Facebook, trabajando con la forma en que se da acceso a los datos que están estructurados en GraphQL.

La diferencia entre GraphQL y Rest radica en que a la hora de realizar una consulta o consumir un servicio podemos especificar los datos que queremos que nos retorne, en cambio con Rest, traemos el total de los datos que están disponibles en el php en línea.

Concluir diciendo que GraphQL es una nueva tecnología desarrollada exclusivamente para facilitar el acceso a los datos a la hora de desarrollar aplicaciones móviles y que debe ser analizada por la Dirección de Informática como una alternativa al desarrollo que actualmente se realiza en la capa de Negocio dentro de la Arquitectura de la Universidad.

16 ANEXO N°2: DOCUMENTACIÓN WIKI

Bienvenido a la WIKI de la Dirección de Informática de la Universidad del Bio-Bio

La Dirección de informática de la Universidad del Bio-Bio tiene como propósito asegurar un alto nivel de integración en el desarrollo de los sistemas informáticos y proporcionar una experiencia satisfactoria a sus usuarios. Por lo anterior es necesario contar con especificaciones que sirvan de guía y faciliten en desarrollo informático ajustado a las exigencias técnicas de la institución.

Los objetivos de dicha estandarización son:

- Mejorar la usabilidad aumentando la satisfacción del usuario.
- Permitir que las soluciones elaboradas para una aplicación pueda ser reutilizadas fácilmente.
- Facilitar el mantenimiento de las aplicaciones informáticas.
- Desarrollar aplicaciones informáticas seguras.

La Dirección de Informática será la responsable de promover y verificar el cumplimiento de actividades y estándares necesarias para un buena implementación de sistemas.

Los principales destinatarios son los analistas, analistas programadores y programadores responsables del desarrollo y mantenimiento de las aplicaciones.

Si necesita acceso a las secciones de esta WIKI solicítelo a:

- Cristian Ortiz (cortizbr@ubiobio.cl)
- Joan Briones (jbriones@ubiobio.cl)

Editar

Sistemas Cliente Servidor

- [Estándares de Programación Power Builder](#)

Editar

Intranet Institucional

- [Estándares de Programación](#)

Editar

Aplicaciones Móviles

- [Estándares de Programación](#)

Editar

Manuales y Procedimientos DBA

- [Buenas Prácticas SQL SERVER](#)
- [Extracción Versiones Código SQL SERVER](#)

Editar

Desarrollos Externos

- [Actividades Proyecto de Desarrollo de SW](#)
- [Desarrollo WEB](#)

Editar

- ◊ [Bienvenido a la WIKI de la Dirección de Informática de la Universidad del Bio-Bio](#)
- ◊ [Sistemas Cliente Servidor](#)
- ◊ [Intranet Institucional](#)
- ◊ [Aplicaciones Móviles](#)
- ◊ [Manuales y Procedimientos DBA](#)
- ◊ [Desarrollos Externos](#)

17 ANEXO N°3: ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE PARA EL DESARROLLO DE MIUBB

17.1 Alcances

El software contempla la implementación de la intranet de la universidad a nivel móvil, dando una nueva opción a los alumnos de la Universidad de acceder a recursos que entrega el sistema.

17.2 Objetivo del Software

El objetivo del software es implementar una solución informática, que permita a los alumnos de la Universidad del Bio-Bío, optimizar y mejorar el proceso de acceder a las distintas opciones que entrega la intranet de la institución mediante una plataforma móvil.

17.3 Atributos del producto

Descripción global del producto

- **Usabilidad**

Que los mensajes de error sean claros para el usuario. Es decir que en el 100% de los mensajes de error, relacionados con el ingreso y procesamiento de datos, se indique claramente el error, la causa y la solución.

- **Eficiencia**

El sistema debe garantizar un tiempo de carga inferior a 2 segundos considerando una conexión de red de velocidad estándar de 4 Mbyte/sy 100 usuarios conectados.

- **Seguridad**

El sistema mantiene un control de acceso a la funcionalidad a través de login – password.

17.4 Interfaces de Comunicación

Se utilizará interfaces de comunicación de 3 capas, la capa de presentación, capa de aplicación y capa de datos.

El sistema donde se implementara el software utiliza un diseño Cliente – Servidor de 3 capas, la capa de presentación, capa de aplicación y capa de datos, ya que nuestro sistema realiza la totalidad de sus funcionalidades en un servidor web, por consiguiente, como se debe acceder desde cualquier Smartphone compatible con la versión y plataforma en la que se creara la aplicación, la que Figura N°167 es una representación de la arquitectura a utilizar.

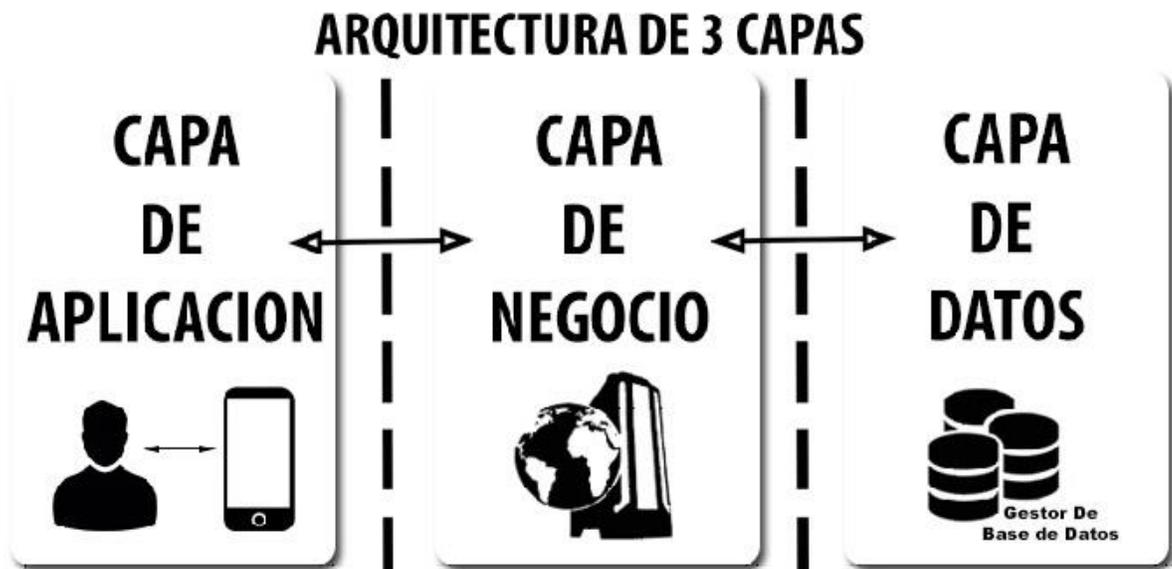


Figura N° 166: Arquitectura Utilizada

17.5 Requerimientos específicos

17.5.1 Requerimientos funcionales del sistema

ID	Nombre	Descripción
RF01	Visualizar Notas	La aplicación Móvil deberá permitir la visualización las notas de cada ramo inscrito por alumno
RF02	Visualizar Asistencia	La aplicación Móvil deberá permitir la visualización la asistencia en cada ramo inscrito por el alumno
RF03	Visualizar Horarios	La aplicación Móvil deberá permitir la visualización de los horarios y salas de los ramos inscritos
RF04	Visualizar Malla Curricular	La aplicación Móvil deberá permitir la visualización la malla curricular del alumno
RF05	Visualizar Cuentas	La aplicación Móvil deberá permitir la visualización cuentas del alumno
RF06	Reserva de Canchas	La aplicación Móvil deberá permitir la reserva de cachas dentro del campus donde se encuentre el alumno
RF07	Reserva de atención Asistente Social	La aplicación Móvil deberá permitir la reserva de horas de atención para la Asistente Social
RF08	Reserva de atención medico	La aplicación Móvil deberá permitir la reserva de horas de atención para los especialistas de la salud.

Interfaces de entrada

ID	Nombre Ítem	Detalle de Datos Contenidos en ítem
DA_01	Datos del Alumno	Rut, pass_alumno

17.5.2 Análisis

Diagrama de Casos de Uso

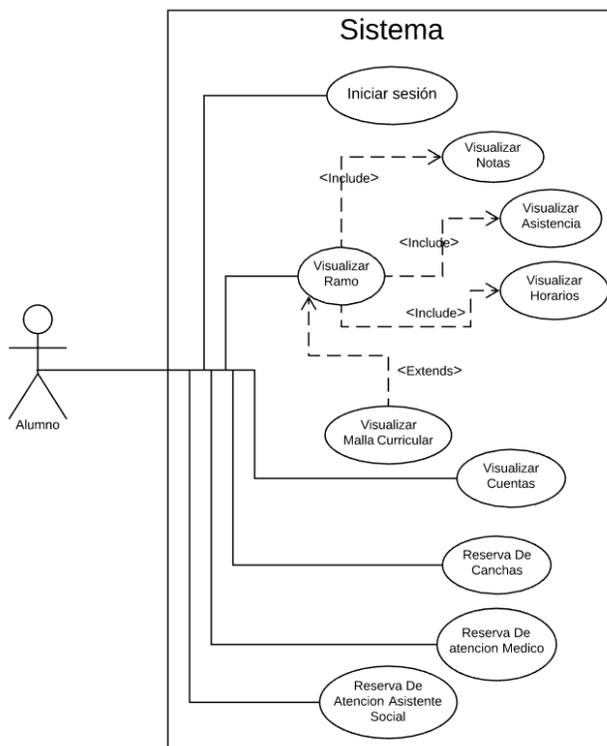


Figura N° 167: Diagrama Casos de Uso

Descripción de Actores

Para el diagrama de casos de uso se consideró solo un actor que es el alumno quien tendrá acceso a las distintas áreas entregadas por la aplicación

Actor: Alumno

Rol: Puede visualizar las notas, horarios, asistencia y malla curricular, también podrá realizar la reserva de cachas así como de horarios de atención para Profesionales de la salud y asistente social.

Nivel de conocimientos técnicos requeridos: Conocimientos a nivel de usuario en aplicaciones funcionales de oficina y en específico navegación móvil.

Nivel de privilegios del sistema: Visualización de la aplicación, visualización de notas, horarios, asistencia y malla curricular, reserva de cachas y horarios de atención para Profesionales de la salud y asistente social.

Especificación de Casos de Uso

C.U.01: Casos de Uso <Iniciar sesión: Alumno>

- **Actores:** Alumno
- **Descripción:** El alumno ingresa a la aplicación Móvil.
- **Precondiciones:** El alumno debe tener descargada e instalada la aplicación Móvil
- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno ingresa al sistema (Aplicación Móvil) a través de un navegador.	2.-) El Sistema muestra el formulario para inicio de sesión.
3.-) El Alumno ingresa los datos, Usuario y contraseña, en el formulario y presiona el Botón “Iniciar Sesión”.	4.-) El Sistema Valida que los datos ingresados están correctos.
	5.-) El Sistema Muestra el menú principal para el alumno.

- **Flujos de Eventos Alternativos:**

Actor	Sistema
3a.-) El alumno ingresa datos de inicio de Sesión erróneos.	4a.-) El Sistema Valida que los datos ingresados están correctos
	4b.-) El Sistema no verifica los datos de la cuenta.
	4c.-) El Sistema notifica que los datos para el Inicio de sesión son erróneos y solicita su ingreso nuevamente.

- **Post Condición:** El Sistema se encuentra cargado para realizar la acción que el alumno estime conveniente.

C.U.02: Casos de Uso <Visualizar Notas: Alumno>

- **Actores:** Alumno
- **Descripción:** El alumno ingresa a la sección de Notas, donde se despliegan los ramos con sus respectivas notas y ponderaciones.
- **Precondiciones:** Haber iniciado sesión y estar en el menú principal
- **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de notas dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos con sus respectivas ponderaciones y notas.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Notas no ingresadas al sistema	2.-) La aplicación no despliega los ramos ni las notas con sus ponderaciones
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

- **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

C.U.03: Casos de Uso <Visualizar Horario: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Horarios, donde se despliegan los ramos con sus respectivos horarios y salas donde se impartirá.

• **Precondiciones:** Haber iniciado sesión y estar en el menú principal

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Horario dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos con sus respectivos horarios y número de sala donde se impartirá el ramo.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Horario y Salas no ingresadas al sistema.	2.-) La aplicación no despliega la lista de ramo con sus salas ni horarios
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

C.U.04: Casos de Uso <Visualizar Asistencia: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Asistencia, donde se despliegan los ramos con su porcentaje de asistencia a cada uno de los ramos inscritos.

• **Precondiciones:** Haber iniciado sesión y estar en el menú principal

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Asistencia dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos con sus respectivos porcentajes de asistencia.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Asistencia no ingresadas al sistema.	2.-) La aplicación no despliega la lista de ramos con su respectiva asistencia
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

C.U.05: Casos de Uso <Visualizar Malla Curricular: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Malla Curricular, donde se despliegan los ramos correspondientes a la carrera.

• **Precondiciones:** Haber iniciado sesión y estar en el menú principal

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Malla Curricular dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de ramos correspondientes a la carrera que cursa el alumno.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Los ramos no están actualizados.	2.-) La aplicación no despliega la lista de ramos de la malla.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

C.U.06: Casos de Uso <Visualizar Cuentas: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Cuentas, donde se despliegan los datos asociados a la cuenta corriente del alumno.

• **Precondiciones:** Haber iniciado sesión y estar en el menú principal

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Cuentas dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de cuentas que el alumno tiene dentro de la universidad.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Los no presenta cuenta corriente dentro de la universidad.	2.-) La aplicación no despliega la cuenta corriente del alumno.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta

C.U.07: Casos de Uso <Reserva de Canchas: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Reserva de Canchas, donde se despliegan las opciones para reservar algún espacio deportivo.

• **Precondiciones:** Haber iniciado sesión y estar en el menú principal

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Reserva de Canchas dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de Canchas y horarios disponibles para la reserva.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) Todas las cachas utilizadas.	2.-) La aplicación no despliega la lista de cachas ni sus horarios.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta.

C.U.08: Casos de Uso <Reserva de atención Asistente Social: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Reserva de atención Asistente Social, donde se despliegan las opciones para la reserva de un horario de atención.

• **Precondiciones:**

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción de Reserva de atención Asistente Social dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de Fechas y horarios disponibles para la reserva de atención.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) No existe asistente social disponible.	2.-) La aplicación no despliega la lista de horarios para reserva de atención.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta.

C.U.09: Casos de Uso <Reserva de atención médico: Alumno>

• **Actores:** Alumno

• **Descripción:** El alumno ingresa a la sección de Reserva de atención médico, donde se despliegan las opciones para la reserva de un horario de atención.

• **Precondiciones:**

• **Flujos Básicos:**

Actor	Sistema
1.-) Este caso de uso comienza cuando el Alumno selecciona la opción Reserva de atención médico dentro del menú principal.	2.-) La aplicación móvil desplegará la lista de Fechas y horarios disponibles para la reserva de atención.
	3.-) La aplicación móvil permite al alumno volver al menú principal para poder visualizar las demás opciones.
4.-) El alumno puede seleccionar la opción de volver o simplemente cerrar la aplicación.	5.-) La aplicación móvil responderá a la solicitud del alumno.

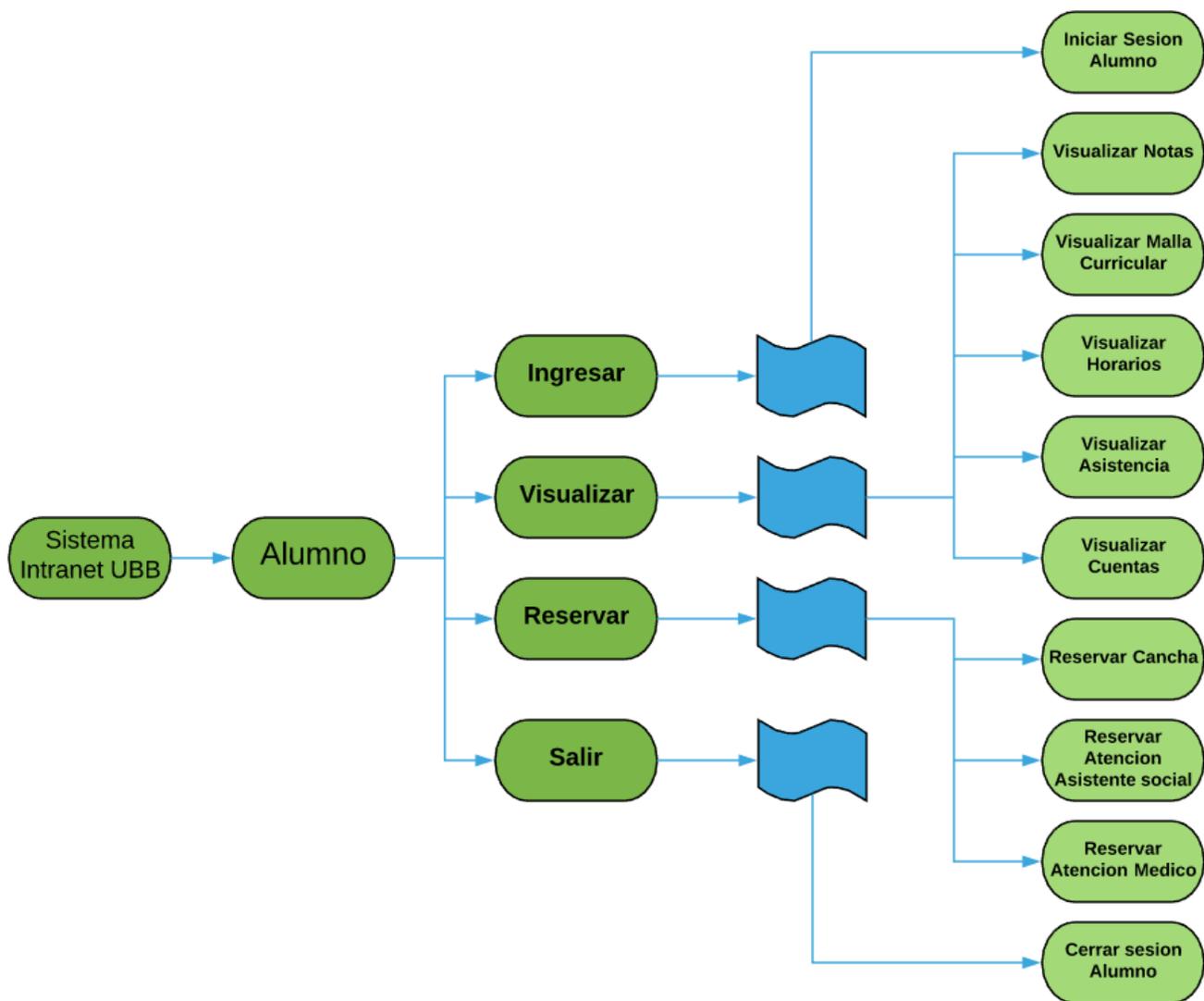
• **Flujos de Eventos Alternativos:**

Actor	Sistema
1.-) No existe médico disponible.	2.-) La aplicación no despliega la lista de horarios para reserva de atención.
	3.-) La aplicación no responde a las necesidades del cliente.
	4.-) Aplicación produce un cierre inesperado.

• **Post Condición:** El alumno podrá seguir visualizando la aplicación o si lo desea puede salir de esta.

17.5.3 Diseño de arquitectura Funcional

En el siguiente diagrama se muestra la arquitectura funcional del sistema, donde se especifican las relaciones entre los módulos que tendrán el sistema y las dependencias que existen entre ellos.



17.5.4 Diseño de interfaz y navegación

Platillas de Diseño

Logo: Segmento de la aplicación reservado para colocar una o varias imágenes que representen visualmente a la empresa.

Contenido: Es este espacio se desplegarán los formularios, textos informes, opciones de menú o sub-menú, botones, etc. que el usuario podrá utilizar

Wireframes de plataforma Móvil

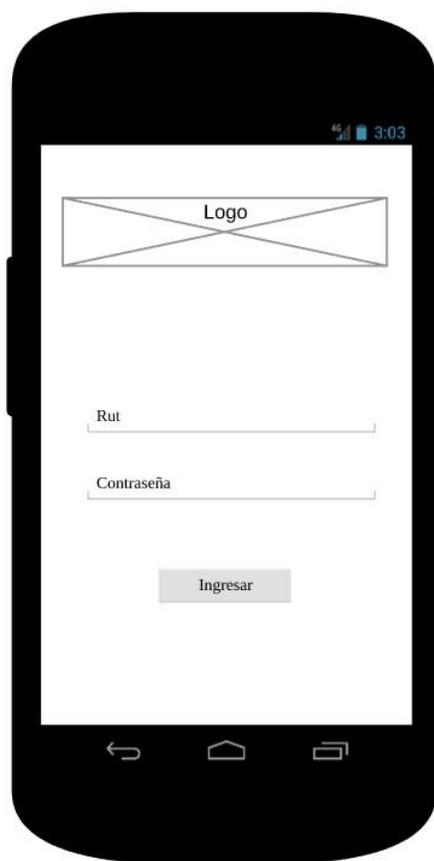


Figura N° 168: Wireframe inicio de sesión

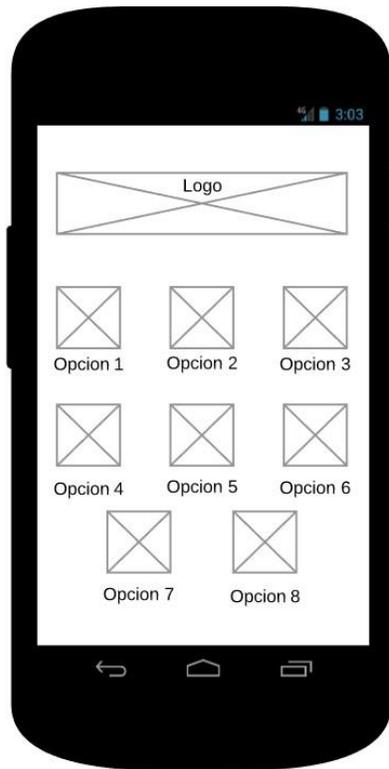


Figura N° 169: Wireframe Menu Principal

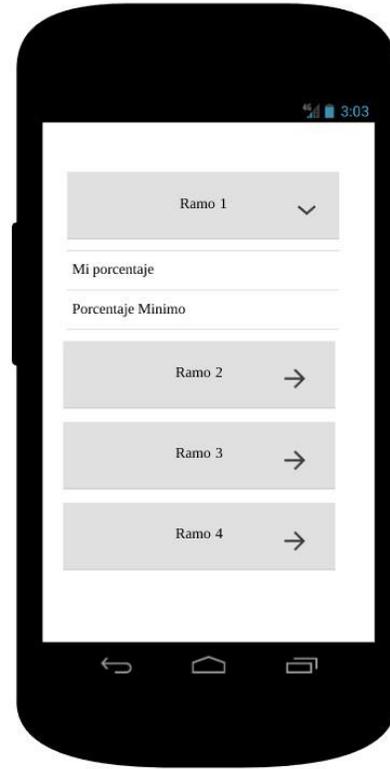


Figura N° 171: Wireframe Asistencia

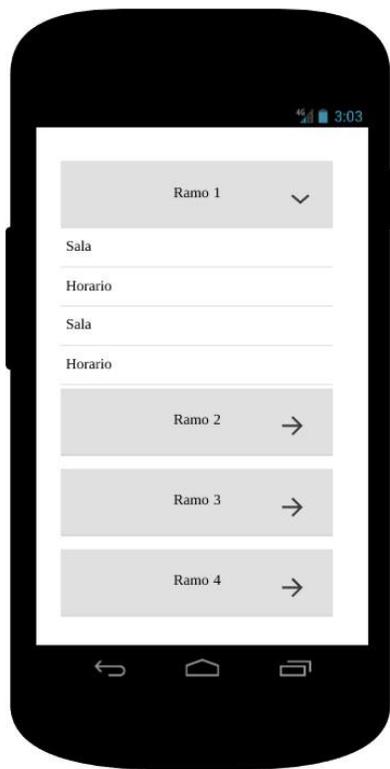


Figura N° 34: Wireframe Horario

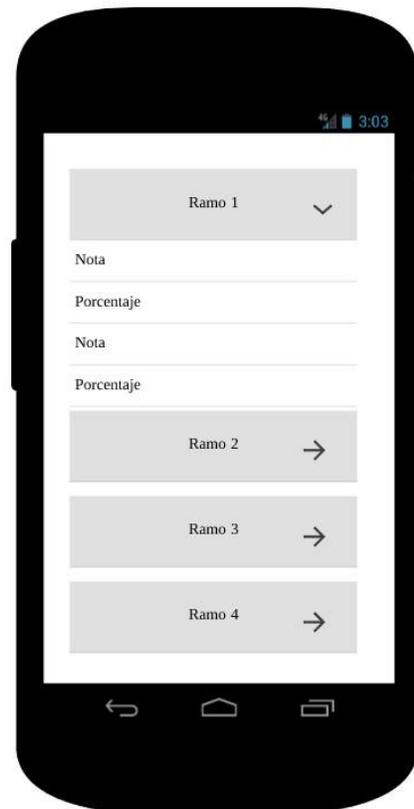


Figura N° 172: Wireframe Notas

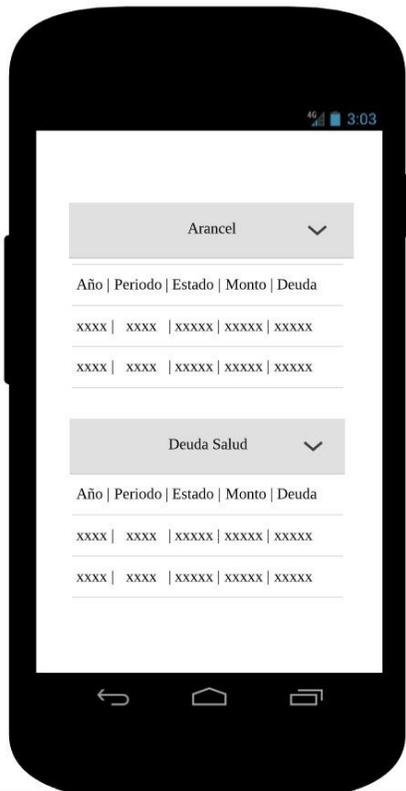


Figura N° 174: Wireframe Cuentas

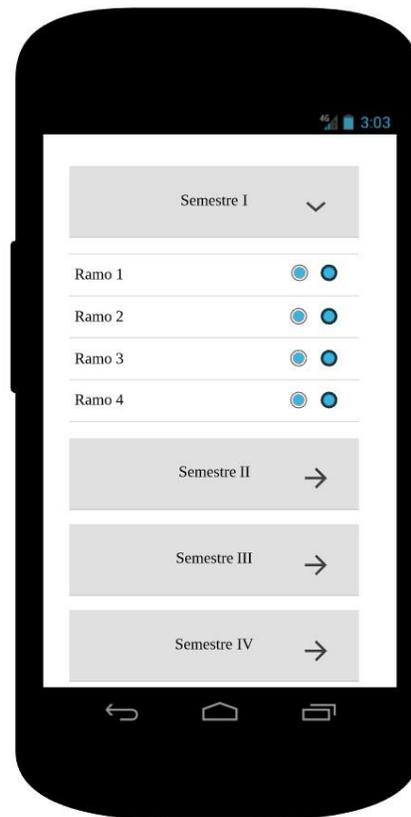


Figura N° 173: Wireframe Malla Curricular

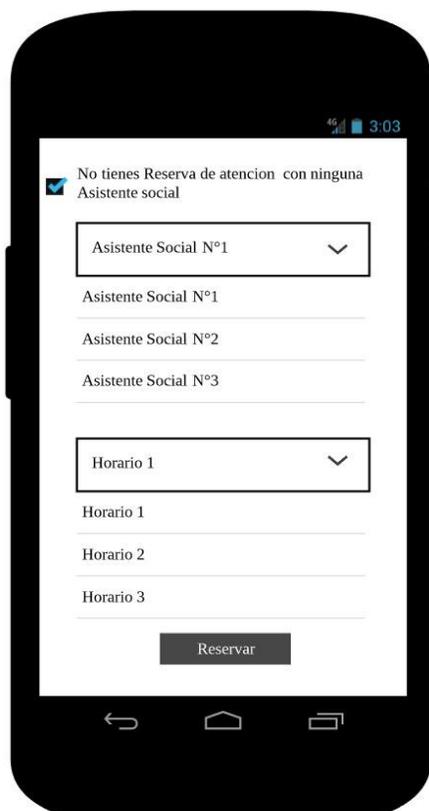


Figura N° 176: Wireframe Asistente Social

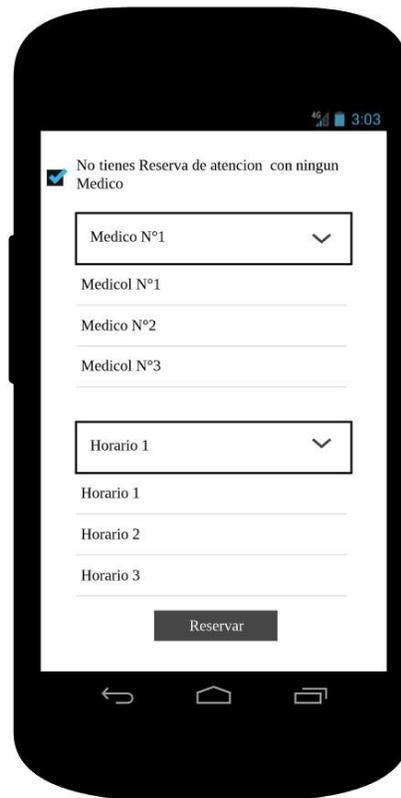


Figura N° 175: Wireframe Salud

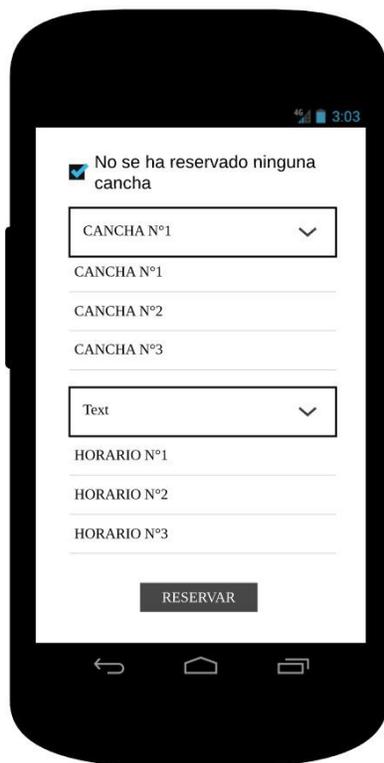


Figura N° 177: Wireframe Canchas

17.5.5 Especificación de módulos

N° Modulo: 1.0		Nombre Modulo: Ingresar	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Rut_Alumno	Varchar(250)	Confirmación	Boolean
Contraseña_Alumno	Varchar(250)		

Tabla N° 85: Especificación de Modulo

N° Modulo: 2.0		Nombre Modulo: Visualizar Horarios	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	ramo_alumno	Varchar(250)
		horario_ramo	Varchar(250)
		sala_ramo	Varchar(250)

Tabla N° 86: Especificación de Modulo

N° Modulo: 3.0		Nombre Modulo: Visualizar Notas	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	ramo_alumno	Varchar(250)
		nota_ramo	Varchar(250)
		ponderacion_ramo	Varchar(250)

Tabla N° 87: Especificación de Modulo

N° Modulo: 4.0		Nombre Modulo: Visualizar asistencia	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	ramo_alumno	Varchar(250)
		nota_ramo	Varchar(250)
		ponderacion_ramo	Varchar(250)

Tabla N° 88: Especificación de módulos

N° Modulo: 5.0		Nombre Modulo: Visualizar cuentas	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Año_arancel	Varchar(250)
		Semestre_arancel	Varchar(250)
		estado_arancel	Varchar(250)
		Año_deuda	Varchar(250)
		Semestre_deuda	Varchar(250)
		Monto_deuda	Varchar(250)
		Estado_deuda	Varchar(250)

Tabla N° 89: Especificación de Modulo

N° Modulo: 6.0		Nombre Modulo: Visualizar malla Curricular	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Nombre_ramo	Varchar(250)
		Periodo_ramo	Varchar(250)
		Ultimo_estado	Varchar(250)

Tabla N° 90: Especificación de Modulo

N° Modulo: 7.1		Nombre Modulo: visualizar horario atención asistente social	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Nombre_asistente	Varchar(250)
		dia_disponible_a	Varchar(250)
		horario_disponible_a	Varchar(250)

Tabla N° 91: Especificación de Modulo

N° Modulo: 7.2		Nombre Modulo: Ingresar horario atención asistente social	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Rut_Alumno	Varchar(250)	Confirmación	Boolean
Nombre_asistente	Varchar(250)		
dia_disponible_a	Varchar(250)		
horario_disponible_a	Varchar(250)		

Tabla N° 92: Especificación de Modulo

N° Modulo: 8.1		Nombre Modulo: visualizar horario atención medico	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Nombre_medico	Varchar(250)
		dia_disponible_m	Varchar(250)
		horario_disponible_m	Varchar(250)

Tabla N° 93: Especificación de Modulo

N° Modulo: 8.2		Nombre Modulo: Ingresar horario atención medico	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Rut_Alumno	Varchar(250)	Confirmación	Boolean
Nombre_medico	Varchar(250)		
dia_disponible_m	Varchar(250)		
horario_disponible_m	Varchar(250)		

Tabla N° 94: Especificación de Modulo

N° Modulo: 9.1		Nombre Modulo: visualizar horario canchas	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Confirmación	Boolean	Tipo_cancha	Varchar(250)
		Numero_cancha	Varchar(250)
		horario_disponible_m	Varchar(250)
		dia_disponible_C	Varchar(250)

Tabla N° 95: Especificación de Modulo

N° Modulo: 9.2		Nombre Modulo: Ingresar reserva cancha	
Parámetros de entrada		Parámetros de Salida	
Nombre:	Tipo de Dato:	Nombre	Tipo de Salida
Rut_Alumno	Varchar(250)	Confirmación	Boolean
Numero_cancha	Varchar(250)		
horario_disponible_m	Varchar(250)		
dia_disponible_C	Varchar(250)		

Tabla N° 96: Especificación de Modulo