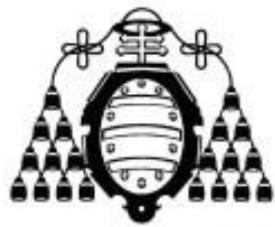


# UNIVERSIDAD DE OVIEDO



ESCUELA DE INGENIERÍA INFORMÁTICA

## TRABAJO FIN DE GRADO

“ULFG: Red social multiplataforma de videojuegos en Xamarin”

**DIRECTOR:** Pablo Javier Tuya González

**AUTOR:** Daniel Alba Muñiz

# Agradecimientos

---

Quisiera dedicar este proyecto a mi madre (Enedina), a mis abuelos (Aurelio y Elvira) y a Raúl por su apoyo durante mi formación académica.

Agradecer además a la Universidad de Oviedo y en concreto a mi tutor, Javier Tuya por su seguimiento y ayuda durante la elaboración del proyecto.

# Resumen

---

ULFG consiste en una red social multiplataforma para Android y Windows similar a otras ya existentes, como Twitter, pero más orientada a los videojuegos y a sus comunidades. Esta aplicación permitirá a un usuario publicar mensajes visibles para otros usuarios, así como ver los de los demás usuarios.

Los usuarios también podrán crear y gestionar grupos personalizados (gremios) a los que se podrán unir otros usuarios. Se podrán publicar mensajes que solo serán visibles para un grupo concreto.

Cada usuario tendrá un perfil propio y personalizable que se mostrará públicamente a todos los demás

# Palabras clave

---

- Red social
- SonarQube
- Multiplataforma
- Xamarin
- Visual Studio
- Azure
- Videojuegos

# Abstract

---

ULFG is a cross-platform social network for Android and Windows like those that already exists in the market, like Twitter. The difference is that this one is focused on games and their communities. This application will allow a user to publish messages that could be read by other users and see the ones others publish.

The users will be able to create and manage their own groups (guilds) as well as invite other users to them. It will be possible to publish messages that are only readable by the members of the group.

Each user will have a public profile that can be customized.

# Keywords

---

- Social Network
- SonarQube
- Cross-platform
- Xamarin
- Visual Studio
- Azure
- Videogames

# Índice General

---

<b>CAPÍTULO 1. MEMORIA DEL PROYECTO .....</b>	<b>14</b>
1.1 RESUMEN DE LA MOTIVACIÓN, OBJETIVOS Y ALCANCE DEL PROYECTO .....	14
1.2 RESUMEN DE TODOS LOS ASPECTOS .....	16
<b>CAPÍTULO 2. INTRODUCCIÓN .....</b>	<b>17</b>
2.1 JUSTIFICACIÓN DEL PROYECTO .....	17
2.2 OBJETIVOS DEL PROYECTO.....	18
2.3 ESTUDIO DE LA SITUACIÓN ACTUAL.....	19
2.3.1 <i>Evaluación de Sistemas Alternativos</i> .....	19
2.3.2 <i>Evaluación de Tecnologías Alternativas</i> .....	22
2.3.3 <i>Justificación de la decisión</i> .....	24
<b>CAPÍTULO 3. ASPECTOS TEÓRICOS .....</b>	<b>25</b>
3.1 CONCEPTOS .....	25
3.1.1 <i>Red Social</i> .....	25
3.1.2 <i>Jugador de videojuegos</i> .....	25
3.2 TECNOLOGÍAS.....	26
3.2.1 <i>.NET</i> .....	26
3.2.2 <i>Entity Framework</i> .....	27
3.2.3 <i>Base de Datos Relacional</i> .....	28
3.2.4 <i>Xamarin</i> .....	29
3.2.5 <i>Azure</i> .....	32
3.2.6 <i>Firebase</i> .....	33
3.3 HERRAMIENTAS .....	33
3.3.1 <i>Visual Studio Team Services</i> .....	33
3.3.2 <i>SonarQube</i> .....	34
3.4 METODOLOGÍAS.....	35
3.4.1 <i>Metricav3</i> .....	35
3.4.2 <i>Scrum</i> .....	36
3.5 PLATAFORMAS.....	38
3.5.1 <i>Android</i> .....	38
3.5.2 <i>UWP</i> .....	39
<b>CAPÍTULO 4. PLANIFICACIÓN DEL PROYECTO Y RESUMEN DE PRESUPUESTOS.....</b>	<b>41</b>
4.1 PLANIFICACIÓN Y SEGUIMIENTO.....	41
4.2 RESUMEN DEL PRESUPUESTO .....	45
<b>CAPÍTULO 5. ANÁLISIS .....</b>	<b>46</b>
5.1 DEFINICIÓN DEL SISTEMA .....	46
5.1.1 <i>Determinación del Alcance del Sistema</i> .....	46
5.2 REQUISITOS DEL SISTEMA .....	47
5.2.1 <i>Índice de requisitos (Story Mapping)</i> .....	47
5.2.2 <i>Identificación de Actores del Sistema</i> .....	50
5.3 IDENTIFICACIÓN DE LOS SUBSISTEMAS EN LA FASE DE ANÁLISIS.....	50
5.4 DIAGRAMA DE CLASES PRELIMINAR DEL ANÁLISIS.....	51
5.4.1 <i>Diagrama de Clases del dominio</i> .....	51

5.4.2	<i>Descripción de las Clases</i> .....	52
5.5	ANÁLISIS PRIORIZADO DE HISTORIAS DE USUARIO Y CRITERIOS DE ACEPTACIÓN .....	54
5.6	ANÁLISIS DE INTERFACES DE USUARIO .....	58
5.6.1	<i>Descripción de la estructura de la interfaz</i> .....	58
5.6.2	<i>Descripción del tema de color de la interfaz</i> .....	59
5.6.3	<i>Descripción del Comportamiento de la Interfaz</i> .....	59
5.6.4	<i>Diagrama de Navegabilidad</i> .....	61
5.7	ESPECIFICACIÓN DEL PLAN DE PRUEBAS .....	62
<b>CAPÍTULO 6. DISEÑO DEL SISTEMA .....</b>		<b>63</b>
6.1	ARQUITECTURA DEL SISTEMA .....	63
6.1.1	<i>Diagramas de Paquetes y Subpaquetes</i> .....	64
6.1.2	<i>Diagramas de Componentes</i> .....	65
6.1.3	<i>Diagramas de Despliegue</i> .....	66
6.2	DISEÑO DE CLASES.....	67
6.2.1	<i>Diagrama de Clases</i> .....	67
6.3	DIAGRAMAS DE INTERACCIÓN .....	81
6.3.1	<i>Enviar Notificación</i> .....	82
6.3.2	<i>Sincronización Offline</i> .....	82
6.4	DIAGRAMAS DE ACTIVIDADES.....	83
6.4.1	<i>Registro</i> .....	83
6.4.2	<i>Inicio de sesión</i> .....	85
6.4.3	<i>Editar campo</i> .....	86
6.4.4	<i>Seguir usuario</i> .....	86
6.4.5	<i>Enviar mensaje</i> .....	88
6.5	DISEÑO DE LA BASE DE DATOS.....	89
6.5.1	<i>Descripción del SGBD Usado</i> .....	89
6.5.2	<i>Integración del SGBD en el Sistema</i> .....	89
6.5.3	<i>Diagrama E-R</i> .....	90
6.6	DISEÑO DE LA INTERFAZ .....	91
6.6.1	<i>Login y Registro</i> .....	91
6.6.2	<i>Menú Principal</i> .....	91
6.6.3	<i>Pantalla Principal</i> .....	92
6.6.4	<i>Pantalla de Gremios</i> .....	95
6.6.5	<i>Otras pantallas</i> .....	97
6.7	ESPECIFICACIÓN TÉCNICA DEL PLAN DE PRUEBAS .....	97
6.7.1	<i>Pruebas Unitarias</i> .....	97
6.7.2	<i>Pruebas de Sistema</i> .....	99
6.7.3	<i>Pruebas de Usabilidad</i> .....	100
6.7.4	<i>Pruebas estáticas de Código</i> .....	103
<b>CAPÍTULO 7. IMPLEMENTACIÓN DEL SISTEMA .....</b>		<b>103</b>
7.1	ESTÁNDARES, NORMAS Y PATRONES SEGUIDOS.....	103
7.1.1	<i>Patrón MVVM</i> .....	103
7.1.2	<i>UML</i> .....	104
7.2	LENGUAJES DE PROGRAMACIÓN .....	105
7.2.1	<i>C#</i> .....	106
7.2.2	<i>SQL</i> .....	107
7.3	HERRAMIENTAS Y PROGRAMAS USADOS PARA EL DESARROLLO .....	109
7.3.1	<i>Visual Studio Enterprise</i> .....	109
7.3.2	<i>Postman</i> .....	109

7.3.3	<i>Visio Profesional 2016</i> .....	110
7.3.4	<i>Word 2016</i> .....	110
7.4	CREACIÓN DEL SISTEMA .....	111
7.4.1	<i>Problemas Encontrados</i> .....	111
7.4.2	<i>Descripción Detallada de las Clases</i> .....	112
<b>CAPÍTULO 8. DESARROLLO DE LAS PRUEBAS.....</b>		<b>113</b>
8.1	PRUEBAS UNITARIAS AUTOMATIZADAS .....	113
8.2	PRUEBAS DE SISTEMA.....	114
8.3	PRUEBAS DE USABILIDAD .....	114
8.3.1	<i>Cuestionarios de los usuarios</i> .....	114
8.3.2	<i>Cuestionario del responsable de las pruebas</i> .....	116
8.4	PRUEBAS ESTÁTICAS DE CÓDIGO .....	116
<b>CAPÍTULO 9. MANUALES DEL SISTEMA .....</b>		<b>118</b>
9.1	MANUAL DE INSTALACIÓN .....	118
9.1.1	<i>Android</i> .....	118
9.1.2	<i>Windows</i> .....	119
9.2	MANUAL DE EJECUCIÓN.....	121
9.2.1	<i>Creación de la Base de Datos</i> .....	122
9.2.2	<i>Crear y configurar Notification Hub (NH)</i> .....	123
9.2.3	<i>Preparar el backend</i> .....	128
9.2.4	<i>Consejos para la operación</i> .....	130
9.3	MANUAL DE USUARIO .....	132
9.3.1	<i>Registro</i> .....	132
9.3.2	<i>Login</i> .....	133
9.3.3	<i>Crear publicación</i> .....	133
9.3.4	<i>Buscar Usuarios</i> .....	134
9.3.5	<i>Editar perfil</i> .....	134
9.3.6	<i>Seguir y bloquear</i> .....	135
9.3.7	<i>Ver mensajes directos recibidos</i> .....	135
9.3.8	<i>Enviar mensaje directo</i> .....	136
9.3.9	<i>Buscar gremios</i> .....	137
9.3.10	<i>Crear gremio</i> .....	137
9.3.11	<i>Editar gremio</i> .....	138
9.3.12	<i>Invitar usuario a gremio</i> .....	139
9.3.13	<i>Abandonar gremio</i> .....	139
9.3.14	<i>Expulsar usuario de gremio</i> .....	140
9.3.15	<i>Enviar mensaje grupal al gremio</i> .....	140
9.3.16	<i>Cerrar sesión</i> .....	141
9.4	MANUAL DEL PROGRAMADOR.....	141
<b>CAPÍTULO 10. CONCLUSIONES Y AMPLIACIONES.....</b>		<b>143</b>
10.1	CONCLUSIONES.....	143
10.2	AMPLIACIONES .....	143
<b>CAPÍTULO 11. PRESUPUESTO .....</b>		<b>145</b>
11.1	COSTES DIRECTOS .....	145
11.1.1	<i>Costes básicos</i> .....	145
11.1.2	<i>Costes unitarios</i> .....	145
11.1.3	<i>Costes software</i> .....	148
11.2	COSTES INDIRECTOS .....	148

11.2.1	<i>Gastos indirectos del proyecto</i> .....	148
11.2.2	<i>Gastos indirectos del entorno de desarrollo</i> .....	149
11.3	PRESUPUESTO COMPLETO .....	150
<b>CAPÍTULO 12.</b>	<b>REFERENCIAS BIBLIOGRÁFICAS</b> .....	<b>152</b>
12.1	LIBROS Y ARTÍCULOS .....	152
12.2	REFERENCIAS EN INTERNET .....	153
<b>CAPÍTULO 13.</b>	<b>APÉNDICES</b> .....	<b>155</b>
13.1	GLOSARIO Y DICCIONARIO DE DATOS .....	155
13.2	ANEXOS .....	156

# Índice de Figuras

Figura 1.1.1. Torneo de eSports .....	15
Figura 2.1 Pantalla de Twitter .....	19
Figura 2.2 Pantalla de Discord.....	20
Figura 2.3 Pantalla de Facebook .....	21
Figura 3.1 Arquitectura del ecosistema .NET .....	26
Figura 3.2 Tabla de compatibilidades de .NET Standard .....	27
Figura 3.3 Diagrama de Arquitectura de Entity Framework.....	28
Figura 3.4 Arquitectura de una aplicación Xamarin Nativa .....	30
Figura 3.5 Arquitectura de una aplicación Xamarin.Forms .....	30
Figura 3.6 Arquitectura de una aplicación Xamarin.Forms con Custom Renderers .....	31
Figura 3.7 Servicios ofrecidos por Azure .....	32
Figura 3.8 Consola de Firebase.....	33
Figura 3.9 Ejemplo de un análisis de código en SonarQube .....	35
Figura 3.10 Fases de Scrum .....	37
Figura 3.11 Arquitectura del sistema Android .....	39
Figura 3.12 Arquitectura de UWP .....	40
Figura 4.1 Diagrama de Gantt con los Sprints .....	41
Figura 4.2 Burndown Chart Sprint 1.....	42
Figura 4.3 Burndown Chart Sprint 2 .....	42
Figura 4.4 Burndown Chart Sprint 3 .....	43
Figura 4.5 Burndown Chart Sprint 4 .....	43
Figura 4.6 Burndown Chart Sprint 5 .....	44
Figura 5.1 Boceto de la interfaz .....	58
Figura 5.2 Paletas de color empleadas.....	59
Figura 5.3 Error campo vacío.....	60
Figura 5.4 Error campo incorrecto .....	60
Figura 5.5 Error sin conexión a Internet .....	60
Figura 5.6 Diagrama de Navegabilidad .....	61
Figura 6.1 Arquitectura del Sistema Completo .....	63
Figura 6.2 Diagrama de Paquetes y Subpaquetes .....	64
Figura 6.3 Diagrama de Componentes del Dispositivo Cliente.....	65
Figura 6.4 Diagrama de Componentes de Azure .....	66
Figura 6.5 Diagrama de Despliegue.....	66
Figura 6.6 Diagrama del paquete ULFG.Android y sus subpaquetes .....	67
Figura 6.7 Diagrama del paquete ULFG.UWP y sus subpaquetes.....	68
Figura 6.8 Diagrama del subpaquete Logic .....	68
Figura 6.9 Diagrama del subpaquete Item .....	69
Figura 6.10 Diagrama del subpaquete ItemManager I .....	70
Figura 6.11 Diagrama del subpaquete ItemManager II .....	71
Figura 6.12 Diagrama del subpaquete ItemManager III .....	72
Figura 6.13 Diagrama del subpaquete Shared .....	73
Figura 6.14 Diagrama del subpaquete Main .....	73
Figura 6.15 Diagrama del subpaquete Behaviors .....	74
Figura 6.16 Diagrama del subpaquete PlatformInterfaces.....	74
Figura 6.17 Diagrama el subpaquete Publications .....	75
Figura 6.18 Diagrama del subpaquete Profiles .....	76

Figura 6.19 Diagrama del subpaquete PrivateChat .....	77
Figura 6.20 Diagrama del paquete Network .....	77
Figura 6.21 Diagrama del subpaquete Login .....	78
Figura 6.22 Diagrama del subpaquete Guilds I .....	78
Figura 6.23 Diagrama del subpaquete Guilds II .....	79
Figura 6.24 Diagrama del subpaquete Guilds III .....	79
Figura 6.25 Diagrama del subpaquete DataObjects .....	80
Figura 6.26 Diagrama del subpaquete Controllers I .....	80
Figura 6.27 Diagrama del subpaquete Controllers II .....	81
Figura 6.28 Diagrama de los subpaquetes Service y Helpers .....	81
Figura 6.29 Diagrama de interacción de Enviar Notificación al enviar un mensaje directo .....	82
Figura 6.30 Diagrama de interacción de la Sincronización Offline usando la lista de publicaciones como ejemplo .....	83
Figura 6.31 Diagrama de actividad del Registro .....	84
Figura 6.32 Diagrama de actividad de Inicio de Sesión .....	85
Figura 6.33 Diagrama de actividad de Editar un campo .....	86
Figura 6.34 Diagrama de actividad de Seguir usuario .....	87
Figura 6.35 Diagrama de actividad de Enviar un nuevo mensaje .....	88
Figura 6.36 Diagrama Entidad-Relación .....	90
Figura 6.37 Pantalla de Login .....	91
Figura 6.38 Pantalla de Registro .....	91
Figura 6.39 Pantalla del Menú principal .....	92
Figura 6.40 Pantalla de la Lista de publicaciones .....	92
Figura 6.41 Pantalla de Detalle de adjunto .....	92
Figura 6.42 Pantalla de Crear publicación .....	92
Figura 6.43 Pantalla de Ver Mensajes privados .....	93
Figura 6.44 Pantalla de Chat privado .....	93
Figura 6.45 Pantalla de Elegir destinatario .....	93
Figura 6.46 Pantalla de Crear mensaje .....	93
Figura 6.47 Pantalla de Búsqueda de usuarios .....	93
Figura 6.48 Pantalla de Perfil de Usuario .....	93
Figura 6.49 Pantalla de Gremios propios .....	95
Figura 6.50 Pantalla de Búsqueda de gremios .....	95
Figura 6.51 Pantalla de Detalle de gremio .....	95
Figura 6.52 Pantalla de Ver miembros .....	95
Figura 6.53 Pantalla de Invitar usuario .....	95
Figura 6.54 Pantalla de Editar gremio .....	95
Figura 6.55 Pantalla de Chat de gremio .....	96
Figura 6.56 Pantalla de Crear gremio .....	96
Figura 6.57 Pantalla Editar Perfil .....	97
Figura 6.58 Pantalla de Mi red .....	97
Figura 6.59 Pantalla de Acerca De .....	97
Figura 6.60 Diagrama de interacción de Envío de Notificación con caminos .....	99
Figura 6.61 Diagrama de interacción de Sincronización Offline con caminos .....	100
Figura 6.62 QualityGate de SonarQube para las pruebas de Código .....	103
Figura 7.1 Patrón MVVM .....	104
Figura 7.2 Collage de diferentes tipos de diagrama UML .....	105
Figura 7.3 Visual Studio Enterprise 2017 .....	109
Figura 7.4 Postman .....	110
Figura 7.5 Visio 2016 .....	110
Figura 7.6 Word 2016 .....	111

Figura 8.1 Ejecución de las pruebas unitarias .....	113
Figura 8.2 Resumen de las pruebas en SonarQube .....	116
Figura 8.3 Resultado final de las pruebas en SonarQube.....	117
Figura 8.4 Evolución del número de Issues a lo largo del desarrollo .....	117
Figura 8.5 Evolución del código duplicado a lo largo del desarrollo.....	118
Figura 9.1 Instalación de la apk en Android .....	118
Figura 9.2 Captura de instalación de certificado I .....	119
Figura 9.3 Captura de instalación de certificado II .....	120
Figura 9.4 Cambiar el modo de seguridad de Windows .....	120
Figura 9.5 Captura de instalación de la aplicación para Windows .....	121
Figura 9.6 Captura de App Installer en la Windows Store.....	121
Figura 9.7 Crear recurso de Azure .....	122
Figura 9.8 Crear SQL Database .....	123
Figura 9.9 Creación de Notification Hub .....	124
Figura 9.10 Localización de la configuración de los PNS.....	124
Figura 9.11Creación de un proyecto en Firebase .....	125
Figura 9.12 Integrar Firebase con la aplicación para Android .....	125
Figura 9.13 Acceder a los datos de la mensajería en la nube de Firebase .....	126
Figura 9.14 Clave del servidor de notificaciones Firebase.....	126
Figura 9.15 Asociar aplicación UWP con la tienda .....	127
Figura 9.16 Página principal del portal de desarrollador de Windows.....	127
Figura 9.17 Como acceder a los datos de la aplicación UWP .....	128
Figura 9.18 Clave de seguridad .....	128
Figura 9.19 Identificador de la aplicación .....	128
Figura 9.20 Localización de la cadena de conexión en Azure .....	129
Figura 9.21 Donde colocar la cadena de conexión en ULFGService.....	129
Figura 9.22 Selección del tipo de perfil de publicación .....	130
Figura 9.23 Selección del host para la publicación .....	130
Figura 9.24 Lista de recursos en Azure .....	130
Figura 9.25 Lista de recursos de Azure en VS .....	131
Figura 9.26 Lista de alertas configuradas .....	131
Figura 9.27 Creación de una alerta.....	132
Figura 9.28 Ubicación del Registro .....	133
Figura 9.29 Ubicación de la creación de publicaciones .....	133
Figura 9.30 Ubicación de la búsqueda de usuarios .....	134
Figura 9.31 Ubicación de la edición del perfil.....	135
Figura 9.32 Ubicación de los seguimientos y bloqueos .....	135
Figura 9.33 Ubicación de los mensajes directos .....	136
Figura 9.34 Ubicación de envío de mensaje I .....	136
Figura 9.35 Ubicación de envío de mensaje II .....	136
Figura 9.36 Ubicación de la búsqueda de gremios .....	137
Figura 9.37 Ubicación de la creación de gremios .....	138
Figura 9.38 Ubicación de la edición de un gremio.....	138
Figura 9.39 Ubicación de invitar a un usuario .....	139
Figura 9.40 Ubicación de abandonar gremio .....	139
Figura 9.41 Ubicación del chat grupal de gremio .....	141
Figura 9.42 Ubicación del cierre de sesión .....	141

# Capítulo 1. Memoria del Proyecto

## 1.1 Resumen de la Motivación, Objetivos y Alcance del Proyecto

Hoy en día las redes sociales son una parte integral de nuestra vida diaria, nos acompañan a todos lados y nos permiten estar en contacto con la gente que nos importa o estar siempre al día con los últimos acontecimientos. Existen muchos tipos de redes sociales: algunas son específicas de un único sector (LinkedIn), mientras que otras intentan ser mucho más genéricas (Twitter).

El sector de los videojuegos ha ido cobrando mucha importancia en los últimos años y todo indica que seguirá creciendo en el futuro. Desde la aparición de los eSports este sector intenta calar cada vez más en la sociedad y sustituir a otros deportes tradicionales. Durante el último año 2017 varios informes económicos han determinado que los eSports han recaudado aproximadamente 600 millones de euros y se espera que esta cifra alcance los 900 millones de euros en 2018. Cada vez más gente se siente atraída por este nuevo tipo de deporte, se espera que en el próximo año el número de espectadores alcance los 380 millones.

Los eSports o “deporte electrónico” son competiciones de videojuegos profesionalizadas, generalmente multijugador que requieren una gran dedicación e inversión de tiempo. Sus participantes asisten a entrenamientos constantes y se dedican profesionalmente a ello como cualquier otro deporte tradicional. Se compite sentado delante de una pantalla y está considerado un “deporte mental” que requiere buenos reflejos y capacidad de reacción. Es bastante común competir en equipos, aunque también se puede de forma individual en algunos juegos que lo permitan. La mayoría de los competidores se encuentran en un rango de edad entre 18 y 34 años.

Hoy en día es muy frecuente que se disponga de un dispositivo móvil con conexión a internet, lo que permite permanecer conectado en casi cualquier lugar debido a la gran facilidad de las redes móviles (wifi, datos). Toda red social que se precie debe hacer uso de esto para permitir a los usuarios mantener el contacto incluso fuera de sus lugares de juego. Un acercamiento multiplataforma sería el más adecuado ya que permitiría crear una aplicación móvil y otra para el ordenador de forma sencilla y sin repetir apenas código.

Este proyecto presenta una aplicación multiplataforma para Windows y Android que pretende dar solución al problema de la búsqueda de grupos para jugar y facilitar la comunicación entre jugadores de cualquier videojuego. Cada jugador podrá también compartir sus experiencias de juego y visualizar las de otros jugadores que le interesen.

Como TFG se pretende aplicar los conocimientos adquiridos durante el grado y ampliarlos con otros adicionales que complementen la formación.



*Figura 1.1.1. Torneo de eSports*

## 1.2 Resumen de Todos los Aspectos

A continuación, se presenta un resumen de todos los capítulos que contiene este documento para dar una visión general de lo que se va a exponer:

**Capítulo 2. Introducción:** Se justificará la elaboración del proyecto y sus objetivos. Se realizará un estudio de la situación actual del mercado y las diferentes alternativas existentes al proyecto desarrollado.

**Capítulo 3. Aspectos Teóricos:** Se explicarán brevemente los conceptos, herramientas y tecnologías que se usarán en el proyecto.

**Capítulo 4. Planificación del proyecto:** Se detallará la planificación seguida para la realización del proyecto y un resumen de su presupuesto.

**Capítulo 5. Análisis:** Incluye el estudio de los requisitos iniciales, de las interfaces y de las pruebas que servirá de base para la elaboración del diseño.

**Capítulo 6. Diseño del Sistema:** Se detalla en profundidad todo el diseño del sistema desarrollado a partir del análisis.

**Capítulo 7. Implementación del Sistema:** Se definen todas las normas y estándares utilizados en el desarrollo junto a los problemas encontrados y una descripción detallada del contenido y sus tecnologías.

**Capítulo 8. Desarrollo de las pruebas:** Se detalla en profundidad todo lo referente a las pruebas realizadas.

**Capítulo 9. Manuales del sistema:** Contiene guías y tutoriales de uso del sistema para los usuarios

**Capítulo 10. Conclusiones y Ampliaciones:** Se resume el sistema final elaborado y se definen posibles ampliaciones para el futuro.

**Capítulo 11. Presupuesto:** Describe de forma detallada el presupuesto del proyecto.

**Capítulo 12. Referencias Bibliográficas:** Se incluyen todas las referencias consultadas y utilizadas en la realización del proyecto.

**Capítulo 13. Apéndices:** Incluye un glosario de términos técnicos empleados en el documento y una descripción detallada del contenido que se ha entregado adjunto a este documento.

# Capítulo 2. Introducción

## 2.1 Justificación del Proyecto

La mayoría de los videojuegos actuales están evolucionando hacia el juego en equipo debido a los eSports. Los ingresos y el renombre que un videojuego obtiene de estos deportes no son despreciables y por ello, todas las empresas del sector intentan buscar su nicho en la competición electrónica.

Esto obliga a los jugadores a formar grupos para jugar, incluso si no compiten. Normalmente los juegos no proporcionan ninguna herramienta de búsqueda de grupo y en su lugar poseen un sistema de búsqueda aleatorio para cada partida. La única forma que suelen tener los jugadores para juntarse es tener la suerte de conocer gente en este emparejamiento aleatorio, algo que generalmente no se da, dificultando la creación de grupos no aleatorios o “premades” para jugar de forma un poco más seria.

Existen juegos híbridos más tradicionales que tienen parte individual y parte en grupo que han visto su población reducida por el auge de los eSports. Este es el caso por ejemplo de los MMORPG (Juegos de rol multijugador online masivo). Este tipo de juegos no suelen participar en los eSports, a excepción de algunos casos aislados. La pérdida de población hace que cada vez sea más difícil encontrar gente para el contenido grupal de este tipo de videojuegos y las técnicas que emplean los desarrolladores para paliar la pérdida de población no hacen más que complicar aún más la formación de grupos.

Hoy en día es muy frecuente que se disponga de un dispositivo móvil con conexión a internet, lo que permite permanecer conectado en casi cualquier lugar debido a la gran facilidad de las redes móviles (wifi, datos). Esto se puede aprovechar permitir a los usuarios mantener el contacto incluso fuera de sus lugares de juego. Por ello se ha decidido realizar una aplicación multiplataforma que pueda ejecutarse tanto en ordenadores como en dispositivos móviles.

Por todo se considera que es necesaria una herramienta universal para cualquier juego que sirva para ayudar en la búsqueda de grupos en cualquier juego y que además tenga funciones de red social para que los jugadores puedan compartir sus experiencias. Este proyecto pretende dar solución al problema descrito anteriormente.

## 2.2 Objetivos del Proyecto

Este proyecto intenta lograr los siguientes objetivos:

Objetivos del Proyecto a desarrollar

- Facilitar la creación y búsqueda de grupo con el que jugar.
- Permitir a los jugadores comunicarse y mantener el contacto.
- Crear una red social que permita a los jugadores compartir sus experiencias.
- Crear una herramienta multiplataforma que pueda usarse en dispositivos móviles y en ordenadores.

Objetivos del Trabajo Fin de Grado

- Aprender a realizar aplicaciones móviles multiplataforma
- Aprender a realizar interfaces de usuario
- Aprender a utilizar la arquitectura cliente-servidor

## 2.3 Estudio de la Situación Actual

En esta sección se identificarán y se describirán brevemente sistemas similares al que se va a desarrollar con sus correspondientes ventajas e inconvenientes. También se hará un estudio de las diferentes tecnologías que se podrían utilizar para desarrollar el sistema y se justificará la opción elegida.

### 2.3.1 Evaluación de Sistemas Alternativos

#### 2.3.1.1 Twitter

##### 2.3.1.1.1 Descripción

Twitter es una red social basada en la publicación de textos cortos que pueden contener adjuntos de diversos tipos como imágenes o videos. Los usuarios pueden seguir a otros usuarios para estar enterados de las últimas novedades y mandarse mensajes entre ellos.

No está centrada en un tema o sector concreto, sino que en ella podemos encontrar publicaciones de todo tipo.



Figura 2.1 Pantalla de Twitter

##### 2.3.1.1.2 Ventajas

Se puede encontrar información de cualquier tema y muy actualizada. Las principales personas públicas y periódicos están en esta red. El límite de caracteres de las publicaciones ayuda a que se centren en el tema que tratan y no incluyan información innecesaria.

### 2.3.1.1.3 Inconvenientes

El hecho de que no esté centrada puede ser un problema ya que resulta complicado encontrar a otros usuarios que comparten los mismos gustos. No existe un “filtro” de usuarios por tema ni sector y todos son tratados por igual.

### 2.3.1.2 Discord

#### 2.3.1.2.1 Descripción

Discord es un chat centrado en la comunidad de jugadores. Permite crear canales que actúan como chats grupales para la comunicación de sus miembros. Los canales pueden estar destinados a un juego concreto o pertenecer a gremios o comunidades de jugadores.

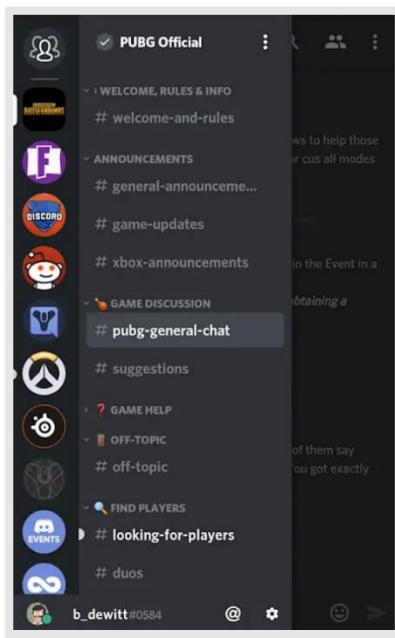


Figura 2.2 Pantalla de Discord

#### 2.3.1.2.2 Ventajas

Permite chats de voz y compartición de pantallas. Su interfaz es sencilla y coherente con el sector en el que se centra. Permite a las comunidades de jugadores tener un lugar en el que reunirse y comunicarse

#### 2.3.1.2.3 Inconvenientes

Para unirse a los canales hacen falta invitación y no hay forma de ver los canales ni los usuarios existentes. Este sistema no sirve para encontrar gente nueva, sino para mejorar la comunicación con gente que ya se ha conocido con otros medios.

### 2.3.1.3 Facebook

#### 2.3.1.3.1 Descripción

Facebook es una red social para todo tipo de usuarios en la que se pueden compartir con otros usuarios fotos e información personal. También incluye un chat para poder comunicarte de manera privada con una o más personas.

Esta red social es, hoy en día, la más conocida a nivel mundial y cuenta con casi 2000 millones de usuarios.



Figura 2.3 Pantalla de Facebook

#### 2.3.1.3.2 Ventajas

Comunicación sencilla y completa entre usuarios. El sistema puede sugerir usuarios basándose en los contactos de correo electrónico o los contactos de usuarios ya conocidos.

#### 2.3.1.3.3 Inconvenientes

Esta más centrada en personas que se conocen en la vida real o que han tenido algún tipo de contacto entre ellas. Ha tenido en el pasado muchos problemas de privacidad y tratamiento de datos personales de los usuarios sin su permiso.

## 2.3.2 Evaluación de Tecnologías Alternativas

En esta sección se explican las diferentes tecnologías que permiten realizar aplicaciones multiplataforma para dispositivos móviles y ordenadores. No se tienen en cuenta aquellas que solo permiten trabajar con una única plataforma porque serían muy costosas y obligarían a aprender y realizar una aplicación independiente para cada plataforma soportada.

### 2.3.2.1 Xamarin

#### 2.3.2.1.1 Descripción

Xamarin es una herramienta de desarrollo de aplicaciones multiplataforma (Windows, iOS, Android) en C# que permite reutilizar gran cantidad de código. Existen dos acercamientos a la hora de utilizar esta herramienta: Xamarin.Forms y Xamarin.Android / Xamarin.iOS

#### 2.3.2.1.2 Xamarin.Android/Xamarin.iOS

##### 2.3.2.1.2.1 Descripción

Permite compartir la lógica de negocio de una aplicación entre varias plataformas, pero la interfaz de usuario debe recrearse para cada plataforma

##### 2.3.2.1.2.2 Ventajas

Se puede compartir toda la lógica de negocio entre plataformas. Obtiene un rendimiento similar al nativo

##### 2.3.2.1.2.3 Inconvenientes

Debe crearse la interfaz de usuario en cada plataforma. Necesita conocimientos de programación nativa de cada plataforma

#### 2.3.2.1.3 Xamarin.Forms

##### 2.3.2.1.3.1 Descripción

Permite compartir tanto la lógica de negocio como la interfaz de usuario entre varias plataformas. Existen partes de código nativas que no se pueden compartir y deben implementarse en cada plataforma. Con este acercamiento se pueden lograr unos porcentajes muy elevados de código compartido.

##### 2.3.2.1.3.2 Ventajas

Es posible aprovechar prácticamente todo el código entre plataformas. Los componentes visuales de Forms se convierten automáticamente al equivalente en cada plataforma. Es posible editar y crear componentes nuevos que se muestren igual en todas ellas. Con solo aprender Xamarin.Forms sería suficiente para crear aplicaciones para todas las plataformas soportadas.

#### 2.3.2.1.3.3 Inconvenientes

Requiere un esfuerzo adicional para obtener un rendimiento aceptable. Es necesario aprender Xamarin.Forms, que es diferente a las plataformas nativas. No todos los componentes visuales están disponibles por defecto.

### 2.3.2.2 Angular JS

#### 2.3.2.2.1 Descripción

Angular JS es un framework Javascript para el desarrollo de aplicaciones web de una sola página para lograr un efecto similar a una aplicación de escritorio

#### 2.3.2.2.2 Ventajas

Sirve para cualquier dispositivo que tengo acceso a un navegador web

#### 2.3.2.2.3 Desventajas

Requiere un esfuerzo adicional para lograr que la interfaz sea fluida debido a la actualización en tiempo real de las vistas. No crea aplicaciones móviles sino aplicaciones web, por lo que no se pueden incluir en las tiendas de las plataformas.

### 2.3.2.3 Ionic

#### 2.3.2.3.1 Descripción

Ionic es un framework de desarrollo de aplicaciones móviles híbridas que usa HTML, CSS y Javascript.

#### 2.3.2.3.2 Ventajas

Rendimiento cercano al nativo usando conocimientos web muy extendidos. Elevada reutilización de código entre plataformas. Permite crear aplicaciones para Android e iOS y ejecutarse en cualquier dispositivo adicional que tenga acceso a un navegador web. Es posible convertir una aplicación web existente en híbrida de forma sencilla.

#### 2.3.2.3.3 Inconvenientes

El único inconveniente sería el uso del lenguaje de programación Javascript, que puede ser complicado para aquellos no familiarizados con él.

### 2.3.2.4 Aplicación Móvil Nativa

#### 2.3.2.4.1 Descripción

Consiste en crear cada aplicación utilizando código puramente nativo. En el caso de Android sería Java y en iOS, Objective-C. Para la aplicación de escritorio de Windows se podría utilizar cualquier lenguaje que permita crear este tipo de aplicaciones.

#### 2.3.2.4.2 Ventajas

El mejor rendimiento posible

#### 2.3.2.4.3 Desventajas

Cada aplicación es completamente diferente y no se puede reutilizar nada. Requiere aprender cada una de las plataformas por separado.

## 2.3.3 Justificación de la decisión

El sistema debe poderse incluir en las tiendas de las plataformas móviles objetivo (Google Play y App Store), por lo que Angular JS no serviría. En el caso de Windows sería opcional que fuese compatible con la Store, por lo que no es un factor determinante en la decisión.

Esto deja dos alternativas: Xamarin e Ionic.

La diferencia fundamental entre ambas alternativas es fundamentalmente el lenguaje de programación:

- **Javascript** es un lenguaje dinámico e interpretado que puede ejecutarse directamente desde un navegador web sin necesidad de configurar nada. Carece de una estructura clara, lo que obliga a conocer perfectamente todo el código y puede resultar difícil encontrar errores debido a que se ejecuta directamente sin compilarse y los errores suelen ser muy genéricos con poca información y varían entre distintos navegadores o plataformas.
- **C#** es un lenguaje orientado a objetos fuertemente tipado, estructurado y compilado. Posee un IDE propio muy completo pero que puede ser algo complicado de instalar y configurar. Puede detectar errores de tipo y sintaxis según se escribe y los errores que surgen durante la ejecución de la aplicación se describen detalladamente, facilitando la labor del programador. Es más complejo de aprender que Javascript, pero a la vez más completo en cuanto a funcionalidad. Este lenguaje se explicará más detalladamente en el apartado [7.2.1. C#](#).

Se ha decidido emplear C#, y por ende Xamarin, debido a las facilidades que proporciona a los programadores y que permiten agilizar el proceso de desarrollo.

Se ha decidido maximizar la multiplataforma y la reutilización de código por lo que la opción de Xamarin usada es Xamarin.Forms.

# Capítulo 3. Aspectos Teóricos

Esta sección está destinada a describir brevemente aquellos conceptos, herramientas y tecnologías existentes que vamos a usar en nuestro proyecto, describiendo por qué y para qué usamos cada uno en nuestro proyecto, pero sin hacer descripciones muy grandes de características y similares.

## 3.1 Conceptos

### 3.1.1 Red Social

El origen de las redes sociales se remonta a 1995, cuando Randy Conrads crea el sitio web classmates.com. Con esta red social se pretendía que la gente pudiese recuperar o mantener el contacto con antiguos compañeros del colegio, instituto o universidad.

A partir del 2000 y con la aparición de la [Web 2.0](#) las redes sociales comienzan a ganar fuerza y a multiplicarse. En 2004 se crea una de las redes sociales más grandes de nuestro tiempo: Facebook

Las redes sociales en Internet son comunidades virtuales donde sus usuarios interactúan con personas de todo el mundo. Pretende simular las redes de contactos y amigos tradicionales que se crean en el mundo real de una forma más amplia, permitiendo incluir gente de todo el mundo. Las más complejas están basadas en la teoría de los grafos y son capaces de sugerir contactos en función de los intereses o la pertenencia a otros grupos

No existe una clasificación concreta a la hora de definir las redes sociales, por lo que en algunos sitios se maneja la tipología que se utilizó para dividir los portales.

- Horizontales: dirigidas a un tipo de usuario genérico y sin una temática definida. Buscan proporcionar herramientas para la interrelación en general. Ejemplos de este tipo son Facebook y Google+.
- Verticales:
  - Por tipo de actividad: dirigidas a promover una actividad concreta. De este tipo son redes como Youtube o Twitter
  - Por tipo de usuario: dirigidas a unos usuarios específicos. Uno de los ejemplos más conocidos de este tipo es la red profesional Linkedin.

La red social a construir se clasificaría como una red de tipo vertical por tipo de usuario.

### 3.1.2 Jugador de videojuegos

Se considera jugador de videojuegos aquella persona que juega o haya jugado alguna vez a un juego electrónico independientemente de la plataforma. Entre las plataformas se incluyen tanto

consolas (Nintendo, PlayStation, Xbox) como dispositivos móviles (Android, iOS) y por supuesto el PC.

## 3.2 Tecnologías

### 3.2.1 .NET

Es un ecosistema de desarrollo de software compuesto por .NET Core, .Net Framework y Xamarin. Las aplicaciones de .NET se ejecutan sobre una máquina virtual que proporciona una seguridad mejorada y es compatible con varios lenguajes de programación. En los últimos años se ha desarrollado una librería común para los tres componentes conocida como .NET Standard.

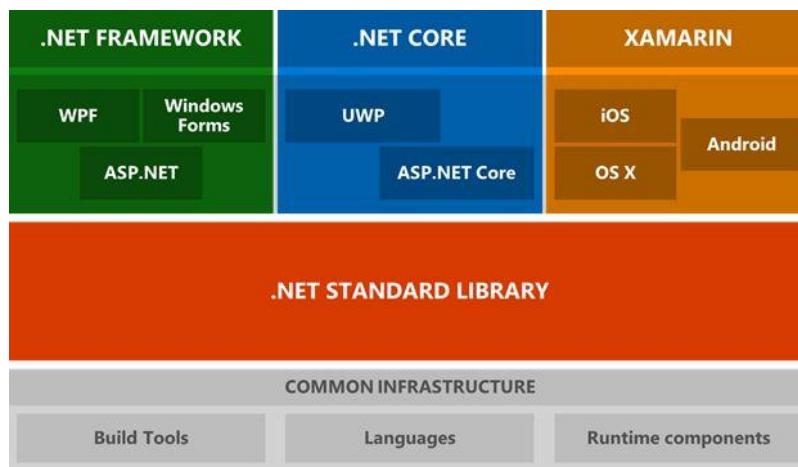


Figura 3.1 Arquitectura del ecosistema .NET

#### 3.2.1.1 .NET Framework

Es un framework software para el desarrollo de aplicaciones para la plataforma Windows. Fue el origen del ecosistema .NET, pero actualmente se pretende que sea sustituido por .NET Core, que es mucho más universal. Se usa la versión 4.6 en ULFGService, un proyecto de ASP.NET.

#### 3.2.1.2 .NET Core

Es un framework software opensource que permite la creación de aplicaciones multiplataforma en gran cantidad de lenguajes. Es el futuro del ecosistema .NET. No se emplea en el proyecto.

#### 3.2.1.3 .NET Standard

Es una especificación formal de las APIs de .NET que pretende estandarizar el ecosistema .NET de manera que se pueda emplear una única API para todas las implementaciones vistas anteriormente. Es el sustituto de las antiguas Bibliotecas de clases portátiles (PCL) cuya funcionalidad estaba muy limitada.

Según Microsoft, estas son las mejoras que proporciona:

- Define un conjunto uniforme de API de BCL para todas las implementaciones de .NET que se van a implementar, independientemente de la carga de trabajo.
- Permite a los desarrolladores generar bibliotecas portátiles que se pueden usar en las implementaciones de .NET con este mismo conjunto de API.
- Reduce o incluso elimina la compilación condicional de código fuente compartido debido a las API de .NET, solo para API de sistema operativo.

En el sistema se emplea .Net Estándar 2.0 para todos los paquetes que se despliegan en el dispositivo del cliente.

.NET Standard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0
Núcleo de .NET	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0
.NET Framework <sup>1</sup>	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1	4.6.1	4.6.1
Mono	4.6	4.6	4.6	4.6	4.6	4.6	4.6	5.4
Xamarin.iOS	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.14
Xamarin.Mac	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.8
Xamarin.Android	7.0	7.0	7.0	7.0	7.0	7.0	7.0	8.0
Plataforma universal de Windows	10.0	10.0	10.0	10.0	10.0	10.0.16299	10.0.16299	10.0.16299
Windows	8.0	8.0	8.1					
Windows Phone	8.1	8.1	8.1					
Windows Phone Silverlight	8.0							

*Figura 3.2 Tabla de compatibilidades de .NET Standard*

### 3.2.2 Entity Framework

Entity Framework es un conjunto de tecnologías para ayudar en el desarrollo de aplicaciones orientadas a datos. Permite la creación de los objetos de datos en forma de objetos específicos del dominio que luego se mapean a tablas de una base de datos. La primera versión fue incluida con .NET Framework el 11 de agosto de 2008.

Actualmente existen dos versiones del framework:

Entity Framework 6: Es la evolución natural del framework que funciona con .NET Framework. Se pretende que caiga en desuso al igual que la versión de .NET que utiliza.

Entity Framework Core: Es una versión nueva más ligera, extensible y multiplataforma preparada para funcionar en .NET Core, el futuro de .NET.

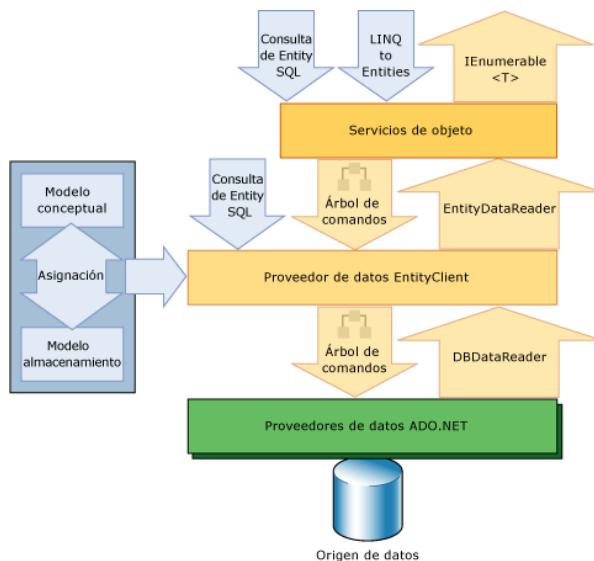


Figura 3.3 Diagrama de Arquitectura de Entity Framework

### 3.2.3 Base de Datos Relacional

Una base de datos relacional es una estructura para el almacenamiento de datos basada en el modelo relacional. En este modelo los datos se organizan en tablas con filas y columnas. Cada tabla debe tener una columna o conjunto de columnas que represente un valor único y sirva para diferenciar una fila de la tabla de las demás. Es posible relacionar los datos de distintas tablas entre sí mediante claves foráneas. Existen tres tipos de relaciones de clave foránea:

- One-to-One: Una fila de una tabla se relaciona con una única fila de otra
- Many-to-One: Una fila de una tabla puede relacionarse con varias filas de otra tabla
- Many-to-Many: Cada fila de las dos tablas implicadas puede relacionarse con varias filas de la otra.

Existe además la posibilidad de automatizar tareas cada vez que se añade, elimina o actualiza una fila mediante procedimientos almacenados.

Permiten realizar todo tipo de consultas complejas sobre los datos usando el lenguaje SQL. Este lenguaje se describirá en profundidad en [7.2.2. SQL](#).

Este tipo de BBDD es el más usado hoy en día ya que el modelo relacional puede modelar situaciones reales de forma eficiente. Se suelen representar mediante diagramas Entidad-Relación.

Existen otro tipo de BBDD conocidas como “NoSQL” debido a que no utilizan el lenguaje SQL para la realización de consultas. Dentro de este grupo podemos encontrar:

- Bases de datos Documentales
- Bases de datos en grafo
- Bases de datos clave / valor
- Bases de datos multivalor
- Bases de datos orientadas a objetos
- Bases de datos tabular

- Bases de datos de arrays

Se puede encontrar más información sobre los tipos de BD NoSQL en el siguiente enlace: [https://en.wikipedia.org/wiki/NoSQL#Types\\_and\\_examples\\_of\\_NoSQL\\_databases](https://en.wikipedia.org/wiki/NoSQL#Types_and_examples_of_NoSQL_databases).

Se ha decidido emplear las BD relacionales debido a que permiten modelar de forma efectiva el sistema a desarrollar y además realizar consultas más complejas para manejar los datos.

### 3.2.4 Xamarin

Xamarin es una plataforma de desarrollo de aplicaciones móviles multiplataforma en C# para dispositivos Android, IOS y Windows. Su objetivo es maximizar la reutilización del código entre plataformas manteniendo un rendimiento cercano al nativo.

La plataforma fue creada por la empresa Xamarin inc. fundada en 2011 por Nat Friedman y Miguel de Icaza. Fue adquirida por Microsoft en 2016.

En los últimos años Xamarin ha ganado gran popularidad debido al crecimiento de la tendencia de los desarrolladores hacia la multiplataforma y el número de aplicaciones construidas en Xamarin no hace más que aumentar.

Además de Android e iOS, las últimas versiones de Xamarin permiten reutilizar el código para la creación de aplicaciones en la Plataforma Universal de Windows.

Permite realizar aplicaciones de tres maneras diferentes:

- Compartiendo solo la lógica de negocio y usando una interfaz de usuario nativa para cada plataforma mediante Xamarin.Android y Xamarin.iOS. Esto se emplea para lograr un rendimiento semejante al nativo
- Compartiendo tanto la interfaz de usuario como la lógica de negocio usando Xamarin.Forms
- Compartiendo parte de la interfaz de usuario y toda la lógica de negocio usando Xamarin.Forms y los llamados "Custom Renderers"

Un Custom Renderer consiste en sustituir un componente de Xamarin.Forms por una representación nativa de dicho componente. Esto permite personalizarlo y obtener un rendimiento nativo. Puede aplicarse sobre una única plataforma de manera que para las que no se haya aplicado se representará el componente por defecto de Xamarin.Forms. Para más información acerca de esta característica consultar el siguiente enlace: <https://docs.microsoft.com/es-es/xamarin/xamarin-forms/app-fundamentals/custom-renderer/>.

Otra de las características de Xamarin es la inyección de dependencias, un principio de diseño creado para ayudar a los desarrolladores a incluir trozos de código nativo en una aplicación de Xamarin.Forms. Es similar a los Custom Renderers, pero en este caso no se centra en un componente sino en una funcionalidad concreta, por ejemplo, una tarea en segundo plano, el manejo de archivos o el uso de sensores del dispositivo. Para más información acerca de esta característica consultar el siguiente enlace <https://docs.microsoft.com/es-es/xamarin/xamarin-forms/app-fundamentals/dependency-service/>.

Otra característica importante es la posibilidad de añadir o sobrescribir la funcionalidad de un determinado componente y sus eventos mediante Behaviors (<https://docs.microsoft.com/es-es/xamarin/xamarin-forms/app-fundamentals/behaviors/introduction>)

También dispone de la característica “MesagingCenter (<https://docs.microsoft.com/es-es/xamarin/xamarin-forms/app-fundamentals/messaging-center>)” que permite a un componente interactuar con otro aunque este en vistas diferentes e incluso si uno de ellos no es visible en el momento de la interacción. El único requisito es que ambos componentes se encuentren en memoria en el momento de la comunicación.

Xamarin cuenta con gran cantidad de librerías creadas por la comunidad con funcionalidad muy variada, aunque la mayoría se centran en ofrecer una funcionalidad nativa avanzada mediante el uso de inyección de dependencias o custom renderers.

Las aplicaciones de Xamarin siguen una estructura determinada centrada en la reutilización de código que varía en función del tipo de aplicación que se vaya a construir. Las siguientes imágenes muestran la arquitectura de cada tipo de aplicación.



Figura 3.4 Arquitectura de una aplicación Xamarin Nativa

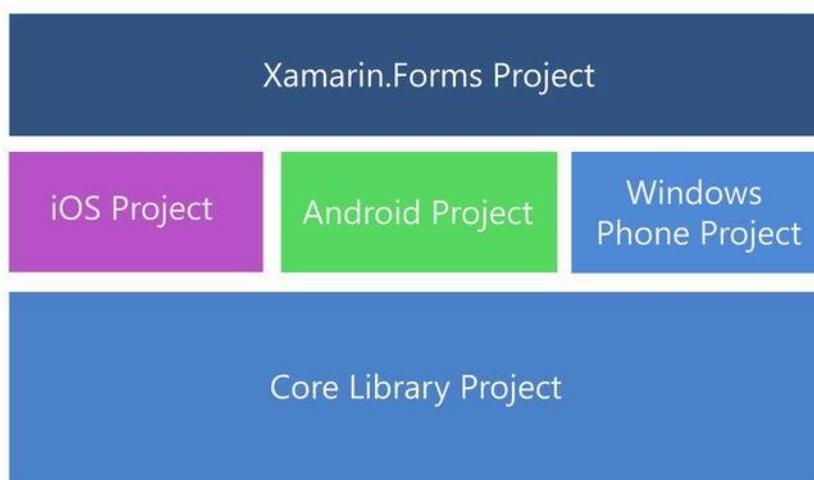
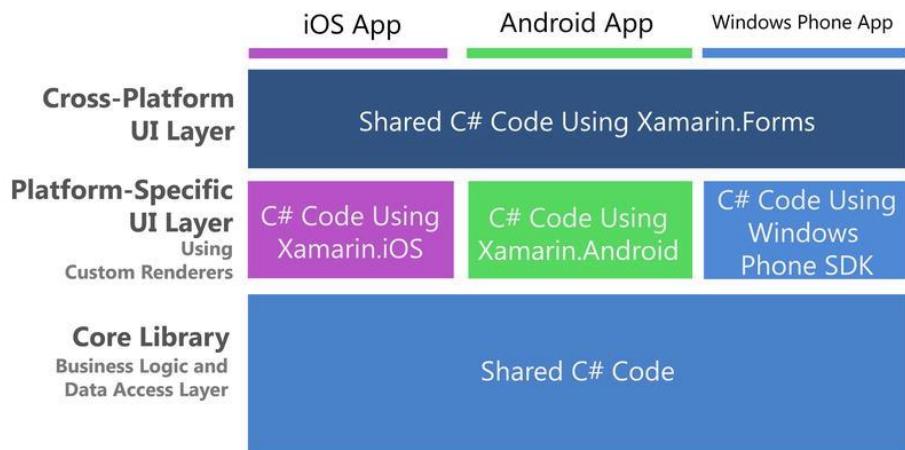


Figura 3.5 Arquitectura de una aplicación Xamarin.Forms

En esta arquitectura sin custom renderers los proyectos de cada plataforma no contienen interfaz de usuario. Toda ella está contenida en el proyecto de Forms y es compartida por todas las plataformas.



*Figura 3.6 Arquitectura de una aplicación Xamarin.Forms con Custom Renderers*

La interfaz de usuario en esta arquitectura está partida y tiene parte en Forms y parte en cada proyecto de forma nativa.

En este proyecto se construirá una aplicación de tipo Xamarin.Forms con la estructura indicada en la imagen anterior correspondiente. La versión de la plataforma utilizada es 3.0.0.446417.

### 3.2.5 Azure

Azure es una plataforma proveedor de servicios en la nube en constante expansión para ayudar a las organizaciones a satisfacer sus necesidades comerciales. Contiene gran cantidad de servicios que van desde el almacenamiento de archivos hasta Machine Learning pasando por el IoT (Internet of Things). Además, proporciona todo lo necesario para la creación y mantenimiento de aplicaciones de cualquier tipo en la nube (web, móvil). Es una de las plataformas en la nube más completas hoy en día.

Fue creado por Microsoft en 2010 bajo el nombre “Windows Azure”. A lo largo de los años ha ido incorporando nuevas funcionalidades hasta llegar a convertirse en lo que es hoy. En 2014 pasa a llamarse Microsoft Azure.

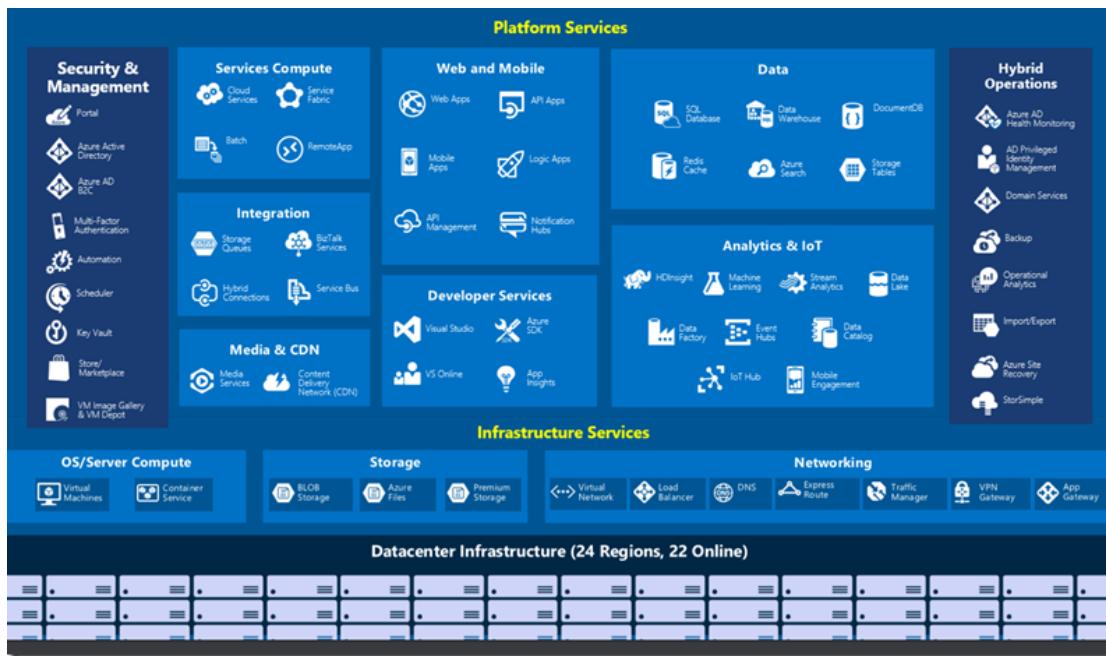


Figura 3.7 Servicios ofrecidos por Azure

Se pude encontrar más información acerca de cada servicio en el siguiente enlace <https://azure.microsoft.com/es-es/services/>.

En este proyecto se utilizarán los siguientes servicios:

- App Service (antiguo Mobile Apps): Sirve para crear un backend para una aplicación móvil o web.
- SQL Database: Es una base de datos relacional.
- Notification Hub: Sirve para centralizar las notificaciones push de todas las plataformas desde una única API.

## 3.2.6 Firebase

Firebase es una plataforma de desarrollo de aplicaciones móviles y web en la nube creada por James Tamplin y Andrew Lee en 2011 y adquirida por Google en 2014. En 2016 Google unificó todos sus servicios de desarrollo móvil usando la plataforma Firebase.

Proporciona un servicio para analizar el uso de las aplicaciones por parte de los usuarios y una serie de servicios orientados al desarrollo como almacenamiento de archivos, servicios de autenticación o notificaciones push.

El único servicio que vamos a usar en este proyecto es Firebase Cloud Messaging para el envío y recepción de notificaciones push en dispositivos Android. Se ha elegido porque es el estándar de Google para dicha función y es de uso gratuito para el uso que le vamos a dar en el proyecto. Para usos comerciales que requieran más recursos habría que adquirir una licencia.

Podemos encontrar una lista de los servicios ofrecidos en <https://firebase.google.com/products/>

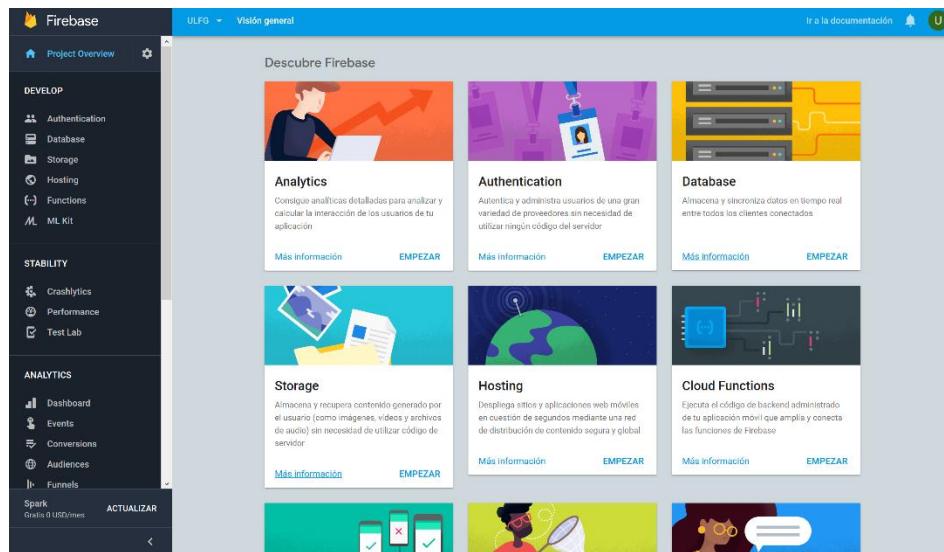


Figura 3.8 Consola de Firebase

## 3.3 Herramientas

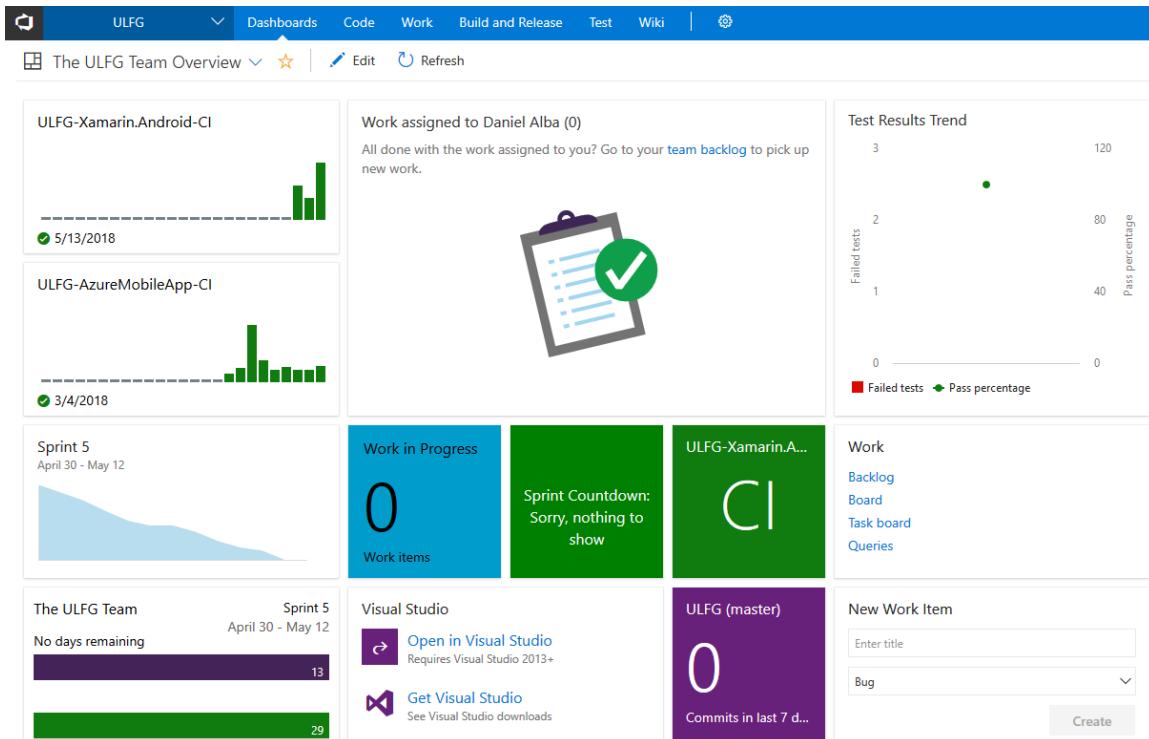
### 3.3.1 Visual Studio Team Services

Visual Studio Team Services (VSTS) es una herramienta de gestión de código en la nube creada por Microsoft. Está integrada completamente con Visual Studio facilitando su uso con aplicaciones del framework .NET, pero funciona en cualquier IDE y soporta cualquier lenguaje.

Es posible descargar una versión de escritorio para ejecutar de forma privada en un ordenador local, conocida como Team Foundation Server (TFS). TFS fue creado en 2005 y fue el origen de VSTS, que no es más que un TFS en la nube.

Proporciona herramientas para realizar un seguimiento de los proyectos mediante metodologías agiles como Scrum o Kanban, un control de versiones basado en Git o xx, integración continua y gestión de plan de pruebas.

Se ha decidido usarlo en este proyecto debido a que concentra todo lo necesario para realizar el desarrollo usando la metodología ágil scrum sin necesitar de herramientas adicionales.



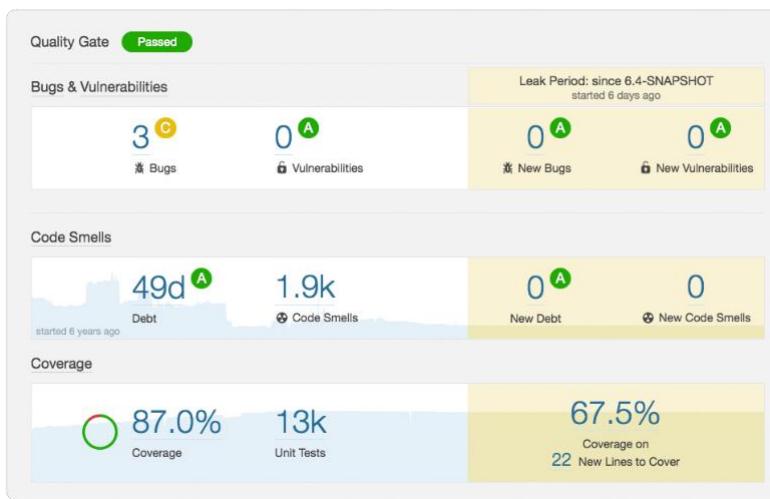
### 3.3.2 SonarQube

Es una herramienta para evaluar código fuente mediante análisis estático. Esto se hace para comprobar y mantener la calidad del código o encontrar potenciales errores en el mismo. La última versión liberada es la 7.1, pero existe una versión LTS (Long-Therm Support) 6.7.4

Hoy en día SonarQube es capaz de informar sobre código duplicado, estándares de codificación, pruebas unitarias, cobertura de código, complejidad ciclomática, potenciales errores, comentarios y diseño del software.

Se integra perfectamente con herramientas de CI como Jenkins o la usada en este proyecto, Visual Studio Team Services.

Soporta más de 20 lenguajes de programación diferentes, entre ellos el que será usado en el proyecto, C#



*Figura 3.9 Ejemplo de un análisis de código en SonarQube*

Dispone de una versión para descargar que puede ejecutarse en local o desplegarse en un servidor web propio y de otra que se ejecuta en la nube. Esta versión en la nube recibe el nombre de SonarCloud. Ambas versiones utilizan el mismo código fuente.

La herramienta es OpenSource, su código está disponible para todo el que quiera verlo en GitHub. La versión descargable es completamente gratuita mientras que SonarCloud cuenta con algunas restricciones. Si se desea emplear SonarCloud para analizar un proyecto de código privado se deberá pagar una licencia de renovación mensual. La condición para analizar un proyecto de forma gratuita es que su código sea visible para todo el mundo, es decir, sea OpenSource.

Para más información acerca de esta herramienta consultar el siguiente enlace: <https://www.sonarqube.org/>.

En este proyecto se utilizará el SonarQube del Grupo de Investigación en Ingeniería del Software (GIIS) de la Universidad de Oviedo proporcionado por el director del proyecto con la versión 6.7.3 de SonarQube.

## 3.4 Metodologías

### 3.4.1 Metricav3

MÉTRICA es una metodología de planificación, desarrollo y mantenimiento de sistemas de información, promovida por el Ministerio de Hacienda y Función Pública del Gobierno de España para la sistematización de actividades del ciclo de vida de los proyectos software en el ámbito de las administraciones públicas. Esta metodología está basada en el modelo de procesos del ciclo de vida de desarrollo ISO/IEC 12207 (Information Technology - Software Life Cycle Processes) así como en la norma ISO/IEC 15504 SPICE (Software Process Improvement And Assurance Standards Capability Determination).

Es una metodología orientada a procesos y en su última versión se contemplan los siguientes:

- Planificación de Sistemas de Información (PSI)
- Desarrollo de Sistemas de Información (DSI)
  - Estudio de Viabilidad del Sistemas (EVS)
  - Análisis del Sistema de Información (ASI)
  - Diseño del Sistemas de Información (DSI)
  - Construcción del Sistema de Información (CSI)
  - Implementación y Aceptación del Sistema (IAS)
- Mantenimiento de sistemas de información (MSI)

Se definen además un conjunto de actividades de tipo organizativo o de soporte al proceso de desarrollo y a los productos que pueden ser necesarias para lograr un sistema seguro y de calidad. Estas actividades se conocen como interfaces.

- Gestión de proyectos (GP)
- Seguridad (SEG)
- Aseguramiento de la Calidad (CAL)
- Gestión de la Configuración (GC)

La plantilla de esta documentación está basada en Metricav3, pero ha sido modificada con ayuda del director del proyecto para servir para la metodología empleada en el desarrollo, Scrum, que se explicará más adelante.

### 3.4.2 Scrum

Scrum es una metodología de tipo ágil de desarrollo de software basada en entregas parciales y regulares del producto final en base al valor que ofrecen a los clientes. Es ideal para situaciones en los que se tienen que desarrollar sistemas complejos en entornos dinámicos cuyos requisitos pueden cambiar con el tiempo y que exigen cierta rapidez en los resultados. De esta manera Scrum se centra en maximizar la capacidad del equipo de desarrollo para entregar rápidamente bloques funcionales del sistema y responder a requisitos emergentes de forma flexible. Esto permite ir adaptando el proyecto en cada entrega a lo que realmente quiere el cliente y asegurarse de que queda satisfecho con la entrega final.

Se contemplarán una serie de roles fundamentales

- **Scrum Team:** Es el equipo encargado de desarrollar y entregar el producto. Debe estar formado por miembros multidisciplinares y ser capaz de auto gestionarse a sí mismo de forma eficiente. Esto implica garantizar una comunicación constante entre los diferentes miembros y que éstos sean capaces de trabajar en equipo de forma horizontal.
- **Scrum Master:** Es una figura experimentada y con conocimientos avanzados de la metodología Scrum. Tiene como objetivo apoyar al equipo de desarrollo y eliminar los obstáculos que puedan surgir durante el proceso.
- **Product Owner:** Como su nombre indica es el “dueño del producto”. Es el que decide qué funcionalidades se incorporan al sistema y cuáles no. Es el encargado de reunirse con los diferentes clientes o stakeholders y de resolver los posibles conflictos que existan entre las funcionalidades que propongan. Este rol elabora una lista priorizada de requisitos y define los criterios de aceptación para cada uno.

- **Stakeholders:** Son aquellos que pueden influir directa o indirectamente en el desarrollo del producto. El cliente que contrata es el stakeholder principal y el destinatario final del valor entregado. Este rol no es específico de Scrum, sino que es un concepto general del desarrollo de proyectos.

El desarrollo de un proyecto Scrum se divide en Sprints. Un Sprint es un periodo de tiempo corto (2-4) semanas que empieza con una reunión inicial con el Product Owner y termina con una demostración de la funcionalidad desarrollada. Durante un Sprint no pueden modificarse los requisitos que se desarrollarán, pero el alcance sí que podría negociarse.

Cada sprint cuenta con una serie de reuniones características:

- Planificación del Sprint:** El equipo se reúne con el Product Owner y se compromete con la funcionalidad a desarrollar durante el sprint que se incluirá en un Sprint Backlog. Consta de dos partes: una en la que el equipo decide exactamente lo que va a hacer y otra en la que decide como va a distribuir y organizar el trabajo.
- Scrum diario:** Es una reunión diaria cuyo objetivo es que todos los miembros informen a los demás de su progreso y se cree un plan para las próximas 24h. El equipo evalúa el progreso de la funcionalidad que se desarrolla en el Sprint y se solucionan los problemas que puedan haber surgido. Sirve además para mejorar la comunicación entre los miembros. Se repite todos los días durante el desarrollo del Sprint hasta la revisión.
- Revisión del Sprint:** Tiene lugar al final del Sprint y es cuando el equipo presenta el producto desarrollado al Product Owner para que lo revise. Este puede aceptar o rechazar la funcionalidad desarrollada y actualizar el Product Backlog si fuera necesario. En esta reunión el equipo planifica el siguiente sprint y se vuelve a empezar desde el punto 1.
- Retrospectiva del Sprint:** El equipo se reúne para discutir lo que ha ido bien durante el Sprint y lo que sería necesario mejorar respecto al siguiente. Se identifican las posibles mejoras y se elabora un plan para llevarlas a cabo.

Si existen varios equipos de Scrum para el mismo proyecto existe otra reunión conocida como **“Scrum de Scrum”** en el que todos los equipos se reúnen para presentar su progreso. Suele realizarse después del Scrum diario.

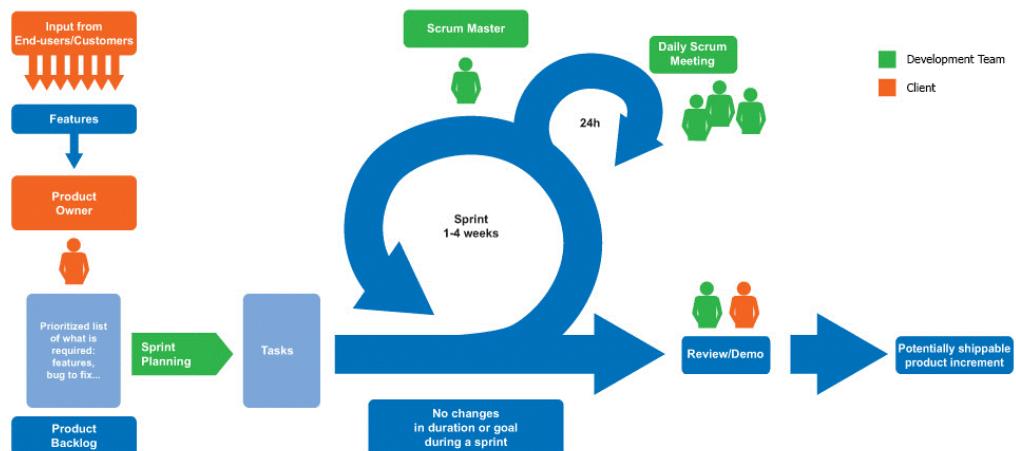


Figura 3.10 Fases de Scrum

Los elementos de documentación más importantes son:

- El Product Backlog es una lista priorizada de todos los requisitos funcionales de lo que se quiere desarrollar a lo largo de todo el proyecto. Las funcionalidades se priorizan en función de su retorno sobre la inversión (ROI). Es gestionado por el Product Owner y solo él puede realizar modificaciones durante la Revisión de cada Sprint.
- El Sprint Backlog es una lista priorizada de la funcionalidad específica a desarrollar durante el sprint. Es creada por el equipo de desarrollo durante la planificación del sprint y no se puede modificar durante el mismo. En ella el equipo define claramente cómo se va a implementar cada requisito. Se incluyen una serie de criterios de aceptación y se estima el tiempo que llevaría desarrollar cada requisito, además del miembro del equipo encargado de él.
- Un Burndown chart es un tipo de grafica que muestra la funcionalidad desarrollada frente al tiempo restante durante el desarrollo. Sirve para tener un control sobre los requisitos pendientes al comienzo de cada sprint y dar una idea global del nivel de finalización del proyecto.

Se emplea esta metodología para desarrollar el sistema porque es una metodología ágil que permite adaptarse a los cambios que puedan surgir durante el desarrollo y realizar entregas parciales con valor que permitan recibir retroalimentación por parte del director de proyecto.

## 3.5 Plataformas

### 3.5.1 Android

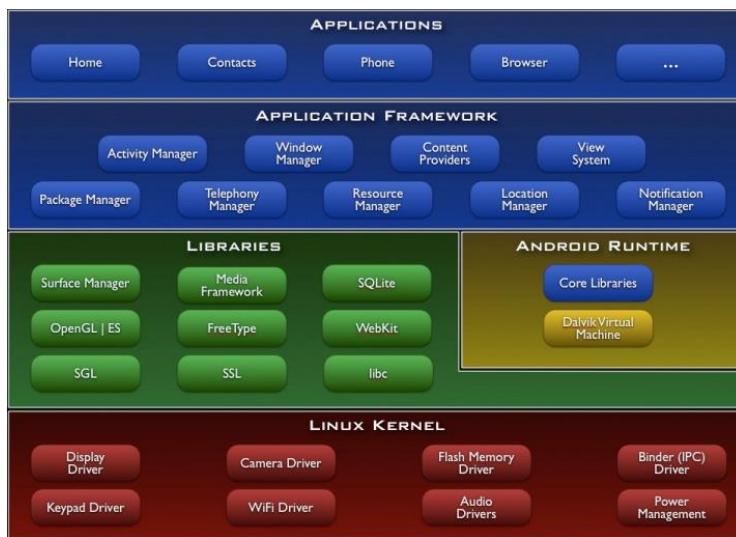
Android es un sistema operativo opensource para dispositivos móviles basado en Linux que fue creado por Google en 2007. Domina el mercado de los dispositivos con más del 80% de la cuota total. Tiene una interfaz sencilla y atractiva con gran cantidad de funciones. Debido a su situación de opensource cualquier persona con conocimientos, normalmente fabricantes pueden personalizar su versión de Android para diferenciarla de la competencia incluyendo programas adicionales y cambios de apariencia como módulos adicionales. Google mantiene una versión de Android sin modificaciones en sus dispositivos propios.

Suele tener cierta controversia con el tema de las actualizaciones ya que muchos fabricantes se niegan a actualizar sus módulos propios con las nuevas versiones, lo que provoca que los dispositivos nunca se actualicen ni reciban parches de seguridad. Los de Google, en cambio, se actualizan casi a diario.

Cada actualización suele venir con cambios en los patrones de diseño de la interfaz de usuario y en determinados módulos del núcleo de Android que obligan a los desarrolladores a actualizarse demasiado para cada versión nueva.

Una de las plataformas en las que se podrá ejecutar el sistema a construir será Android debido a su gran popularidad y uso.

La versión de Android empleada para construir el sistema es la 8.1 (Oreo), pero será compatible con todas las versiones a partir de 6.0 (Marshmallow).



*Figura 3.11 Arquitectura del sistema Android*

### 3.5.2 UWP

UWP, también conocida como Plataforma Universal de Windows es una plataforma de desarrollo de aplicaciones para dispositivos Windows 10 y Xbox. Surgió como una evolución de una plataforma de código compartido creada por Microsoft para los entornos Windows Phone 8.1 y Windows Desktop 8.1. En un principio permitía crear aplicaciones para Windows 10, Windows 10 Mobile y Xbox usando el mismo código fuente, pero en 2017 se anuncia la finalización del proyecto Windows 10 Mobile y el abandono del mercado móvil, lo que impide que las aplicaciones hechas con versiones posteriores de UWP funcionen en dispositivos móviles, ya que no recibirán nuevas actualizaciones.

Estas aplicaciones se distribuyen mediante una tienda conocida como Windows Store y deben cumplir determinadas condiciones para poder publicarse con un determinado uso máximo de memoria. Como Windows es un sistema operativo que permite descargar e instalar aplicaciones de casi cualquier fuente esta tienda no goza de mucha popularidad lo que lleva al poco uso de este tipo de aplicaciones. Las nuevas generaciones se están acostumbrando a instalar todas las aplicaciones mediante tiendas como Google Play en Android, por lo que es posible que en un futuro esta tienda se vaya empezando a usar poco a poco. Hoy en día tiene pocas aplicaciones, pero está evolucionando y se nota el interés de la empresa por sacarla adelante.

La ventaja principal de estas aplicaciones es la seguridad: Se ejecutan en un entorno virtualizado sobre el sistema operativo e impide cualquier acceso al mismo que no sea explícitamente autorizado por el usuario.

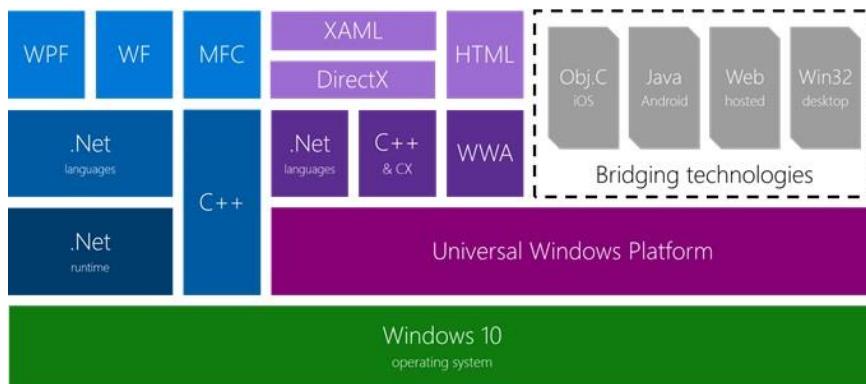
También incluyen funciones para facilitar la reescritura de aplicaciones en otros lenguajes como Objective-C (iOS) o Java (Android). Estas funciones se conocen como “Bridging Technologies”.

A pesar de su poca popularidad se ha decidido incluir esta plataforma por varios motivos

- Crear una aplicación de escritorio aprovechando la multiplataforma

- Tener un primer acercamiento a la plataforma.
- Curiosidad sobre la plataforma.

La versión de Windows 10 usada para construir el sistema es 1803 (April 2018 Anniversary Update). No funcionará en ninguna versión anterior de Windows 10.



*Figura 3.12 Arquitectura de UWP*

# Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos

## 4.1 Planificación y Seguimiento

El desarrollo del proyecto se ha dividido en cinco Sprints de diferente longitud y carga de trabajo. Las tareas asignadas podrán verse más adelante en la descripción de las historias de usuario en el apartado de Análisis. En este apartado se detallará la planificación y el seguimiento del desarrollo del proyecto representando de forma gráfica el seguimiento los diferentes Sprint mediante Diagramas Burndown obtenidos de VSTS y un pequeño resumen de estos apoyado por un Diagrama de Gantt.

Sprint	Intervalo	Días	Horas diarias	Horas totales	Resumen
1	15/2 – 27/2	13	3	39	Mensajes Directos
2	4/3 – 19/3	16	2	32	Gremios
3	3/4 - 16/4	14	2	28	Seguimientos, Bloqueos y Perfil
4	17/4 – 26/4	10	4	40	Autenticación, Imágenes y uso sin conexión
5	30/4 – 12/5	13	4	52	Notificaciones y sesión

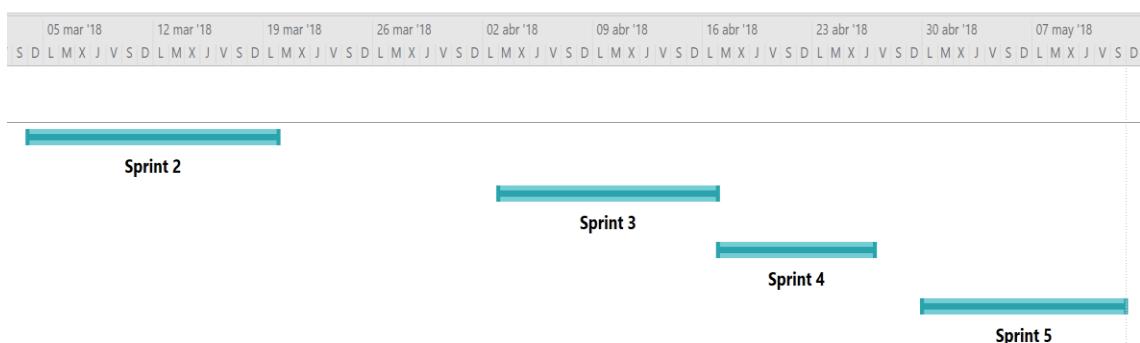


Figura 4.1 Diagrama de Gantt con los Sprints

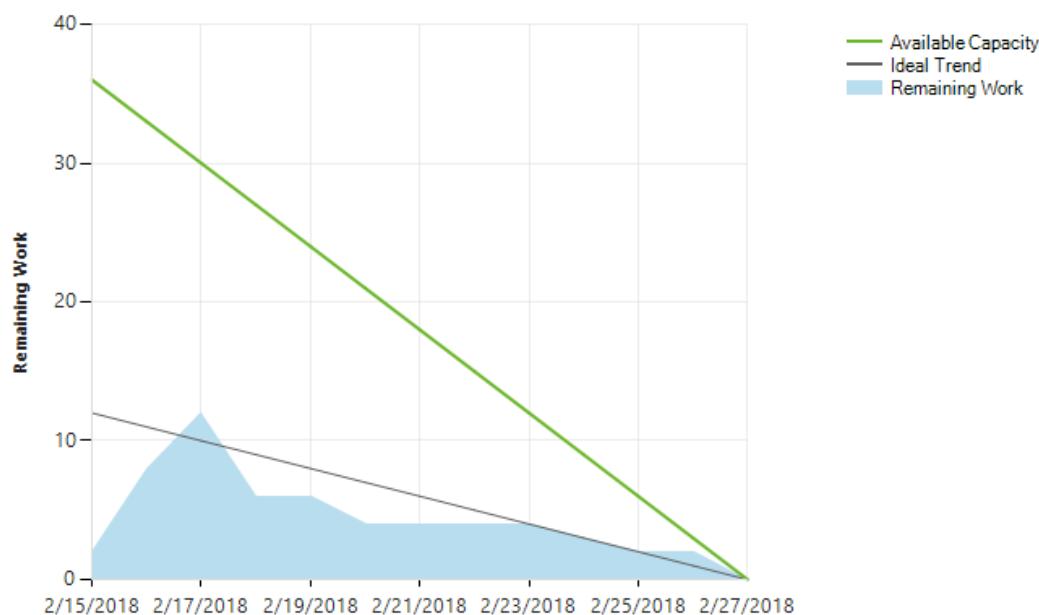


Figura 4.2 Burndown Chart Sprint 1

\*Nota: En este Sprint no se entendía del todo en que se basaba VSTS para crear el diagrama y por eso sale tan pequeño, pero el avance del trabajo restante es como se indica en el mismo. Este problema se describirá más adelante en [7.4.1 Problemas Encontrados](#).



Figura 4.3 Burndown Chart Sprint 2

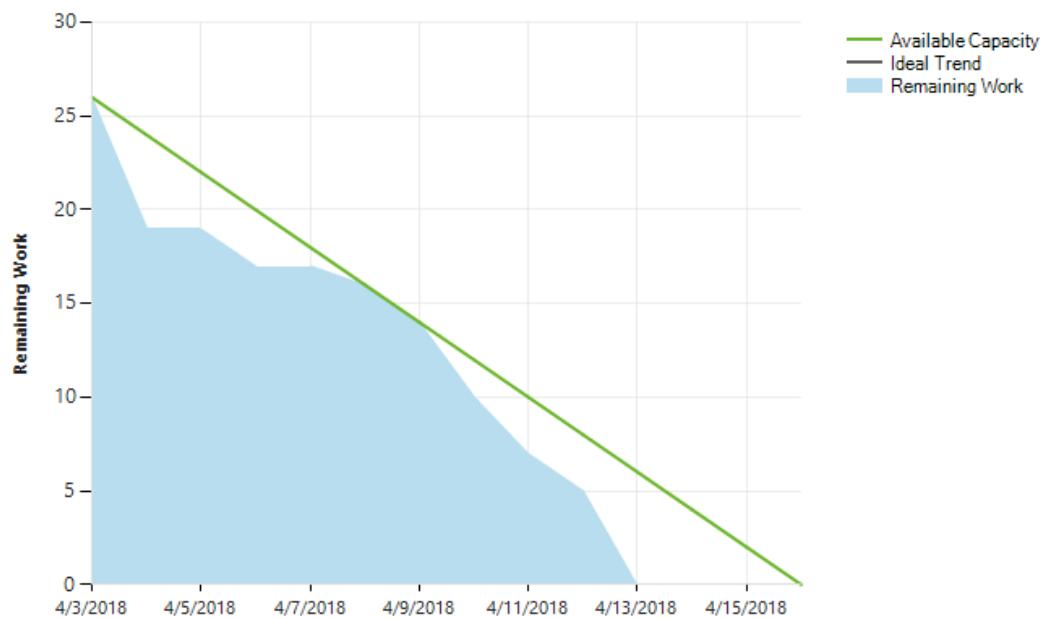


Figura 4.4 Burndown Chart Sprint 3

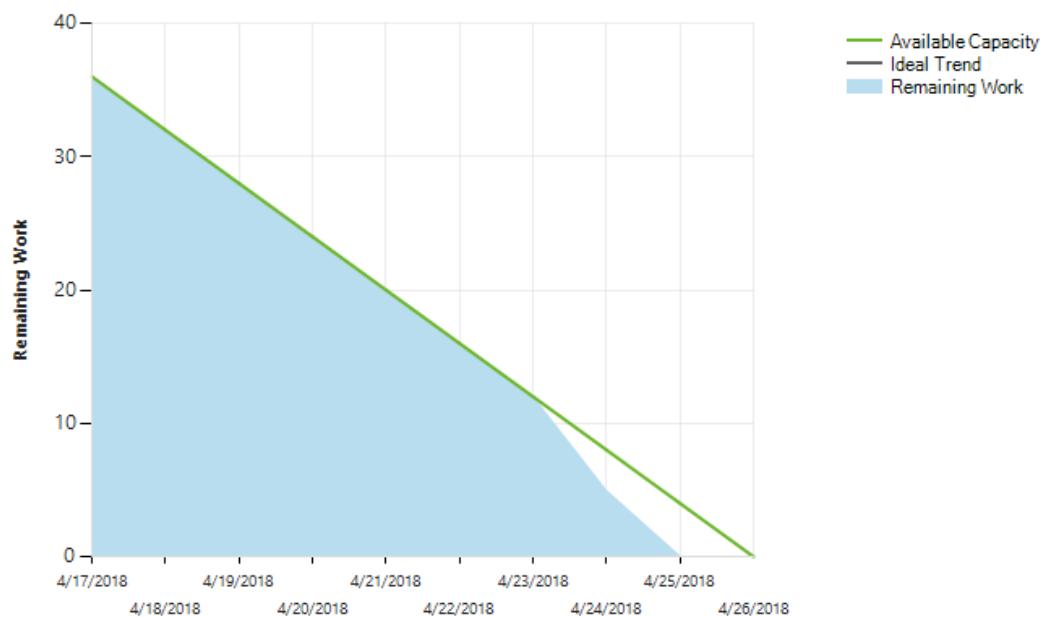
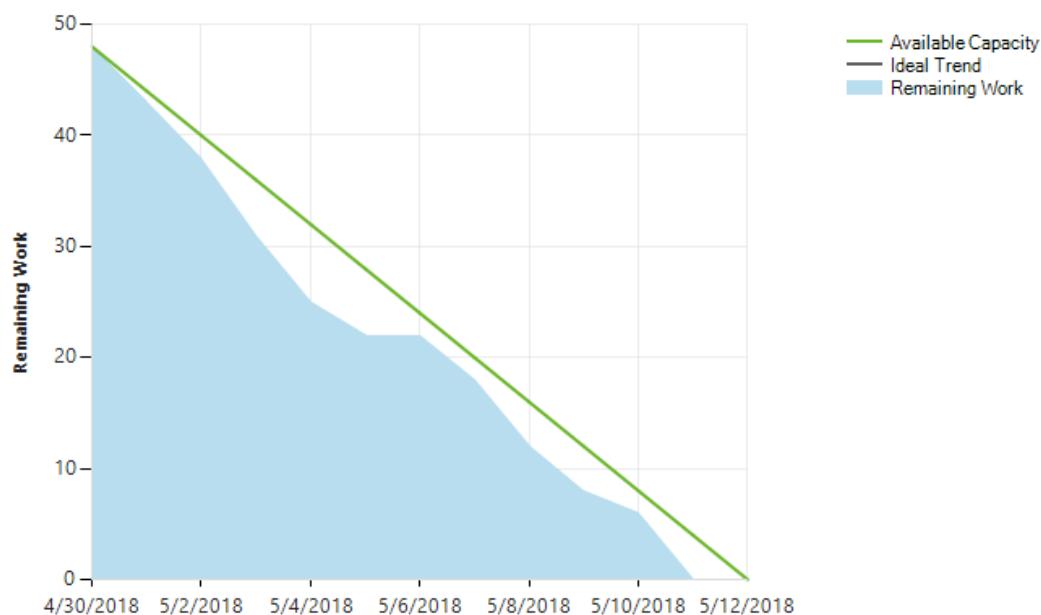


Figura 4.5 Burndown Chart Sprint 4



**Figura 4.6 Burndown Chart Sprint 5**

## 4.2 Resumen del Presupuesto

En esta sección se muestra un resumen del presupuesto total del proyecto para ser entregado al cliente. Las diferentes tareas se han adaptado para que sean entendidas más fácilmente por el cliente. Se puede encontrar un desglose más detallado de este presupuesto en el [Capítulo 11. Presupuesto](#).

Ítem	Concepto	Cantidad	Precio unitario	Total
0	Estudios iniciales	1,00	1.365 €	1.365 €
1	Arranque	1,00	1.365 €	1.365 €
2	Análisis, Diseño, Implementación y Pruebas del Sistema	1,00	15.429,65 €	15.429,65 €
3	Documentación detallada	1,00	8.190 €	8.190 €
Subtotal			26.349,65 €	
IVA (21%)			5.532,76 €	
TOTAL			31.882,47 €	

# Capítulo 5. Análisis

## 5.1 Definición del Sistema

### 5.1.1 Determinación del Alcance del Sistema

ULFG será un proyecto realizado en Xamarin.Forms centrado en la comunidad de jugadores online que funcionará como una red social con los siguientes requisitos:

#### Funcionales:

1. *Gestión de usuarios*
  - 1.1. Sistema de usuarios con registro y login
  - 1.2. Buscar usuarios
  - 1.3. Perfiles personalizables
2. *Mensajes*
  - 2.1. Mensajes directos individuales
  - 2.2. Mensajes grupales de gremio
3. *Publicaciones*
  - 3.1. Crear publicaciones visibles por los seguidores
  - 3.2. Posibilidad de borrar publicaciones ya creadas
  - 3.3. Adjuntar imágenes a las publicaciones
4. *Interacciones*
  - 4.1. Lista de seguidores, seguidos y bloqueados
  - 4.2. Seguir y Dejar de Seguir usuario
  - 4.3. Bloquear y Desbloquear usuario
5. *Gremios*
  - 5.1. Creación de gremios personalizables
  - 5.2. Buscar gremios
  - 5.3. Invitar usuarios
  - 5.4. Expulsar usuarios
  - 5.5. Borrar gremios
6. *Otros*
  - 6.1. Notificaciones push
  - 6.2. Uso sin conexión
  - 6.3. Sesión en el dispositivo

#### No Funcionales:

1. *Debe funcionar en dispositivos Windows 10 y Android*
2. *Debe estar hecha de manera que se permita en un futuro crear una aplicación para iOS de forma sencilla reduciendo la duplicidad del código*

## 5.2 Requisitos del Sistema

### 5.2.1 Índice de requisitos (Story Mapping)

El proceso de desarrollo del sistema se basa en la técnica de historias de usuario, que es muy común en metodologías ágiles como la que usamos en este proyecto (Scrum). Las historias de usuario representan la funcionalidad a desarrollar y sustituyen a los casos de uso de Métrica cuando van acompañadas de criterios de aceptación, de esta manera se evita tener información redundante y se sigue el principio [Lean](#). Deben ser Independientes, Negociables, Valiosas, Estimables, Pequeñas y Testeables (que se puedan probar).

Esta técnica puede provocar que se pierda la visión global de lo que realmente estamos desarrollando porque la funcionalidad se fragmenta demasiado y para ello es muy habitual incluir un índice de requisitos en forma de “Story Mapping”. Sabemos que cada historia de usuario representa una funcionalidad, pero resulta complicado relacionar los conjuntos de historias con la funcionalidad final que se pretende conseguir con ellos.

Un Story Mapping es una vista del backlog organizada en dos dimensiones que permite agrupar las historias de usuario por funcionalidad. En su eje horizontal se colocan las funcionalidades y en el vertical, las historias, de manera que cada historia pertenezca a una funcionalidad.

Utilizando este elemento para representar nuestro proyecto, conseguimos un mapa de todo el conjunto del producto, muy visual y directo.

- Con solo un vistazo podemos entender lo que hace nuestro sistema, tenemos todo en una única foto, tanto la idea general como las más específicas.
- No solo podemos ordenar las historias de forma más cómoda y acertada, sino que también podemos detectar alguna que falte, establecer dependencias entre ellas e incluso identificar mejoras.
- En este caso nuestro Story Mapping no incluye estimaciones, pero podría contenerlas para saber cuánto vamos a tardar en elaborar el producto. Al tenerlas todas juntas y ordenadas es mucho más fácil relacionarlas entre ellas y realizar la estimación.

A continuación, se incluye el Story Mapping del sistema a construir. Los números entre paréntesis indican el número de historia de usuario para facilitar la trazabilidad del índice con el análisis de las historias y sus criterios. Las historias se detallarán más adelante en el apartado [5.5. Análisis Priorizado de Historias de Usuario y Criterios de Aceptación](#).

	Gestión de usuarios	Gremios	Interacciones	Gestión del perfil	Chats individuales	Publicaciones	Otros
Sprint 1					Enviar un mensaje directo a un usuario (1) Listar mensajes directos recibidos (2) Responder un mensaje directo recibido (3)		
Sprint 2		Crear gremio (4) Listar gremios (5) Ver detalle de gremio (6) Unirse a gremio (7) Abandonar gremio (8) Listar mensajes grupales del gremio (9) Enviar un mensaje grupal a todos los miembros del gremio (10) Listar miembros de un gremio (11) Invitar a gremio (12)					
Sprint 3	Editar nombre (20) Editar biografía (21) Editar email (22)	Editar nombre (24) Editar descripción (23) Editar visibilidad (publico/privado) (25)		Ver perfil de otro usuario (13) Seguir a un usuario (14) Dejar de seguir a un usuario (17) Bloquear a un usuario (18)	Crear publicación (15) Ver publicaciones (16)		

	Gestión de usuarios	Gremios	Interacciones	Gestión del perfil	Chats individuales	Publicaciones	Otros
				Desbloquear a un usuario (22) Listar seguidores de un usuario (20) Listar seguidos de un usuario (19) Listar usuarios bloqueados (21)			
Sprint 4	Cambiar foto de perfil (30)	Expulsar miembro (29)				Borrar publicación (32)	
	Registrar usuario (33)	Cambiar imagen de gremio (31)					
	Iniciar sesión con las credenciales (34)						
	Cambiar contraseña (35)						
Sprint 5		Enviar mensaje al líder (36) Borrar gremio (37) Incluir un mensaje fijado para todos los miembros (38)				Adjuntar imagen a publicación (39)	Notificaciones push (42) Uso sin conexión (41) Mantener una sesión (40) Cerrar sesión (43)

## 5.2.2 Identificación de Actores del Sistema

Existen dos tipos de actores en el sistema:

- **Usuario anónimo:** Son los usuarios que no se han identificado en la aplicación
- **Usuario identificado:** Son aquellos usuarios que han accedido a la aplicación introduciendo sus credenciales

## 5.3 Identificación de los Subsistemas en la Fase de Análisis

La descomposición del sistema en subsistemas, según Metricav3, debe estar principalmente orientada a los procesos de negocio, aunque también es posible adoptar otros criterios lógicos como la homogeneidad de procesos o la afinidad de requisitos.

En este caso el sistema no puede diferenciarse en subsistemas ya que no existe una separación clara de procesos de negocio. Por lo tanto, consideraremos un único subsistema a nivel de análisis.

## 5.4 Diagrama de Clases Preliminar del Análisis

### 5.4.1 Diagrama de Clases del dominio

A continuación, se muestra el diagrama preliminar con las diferentes clases de dominio que contendría nuestro subsistema. Este diagrama pretende mostrar una idea aproximada del sistema que se va a construir.

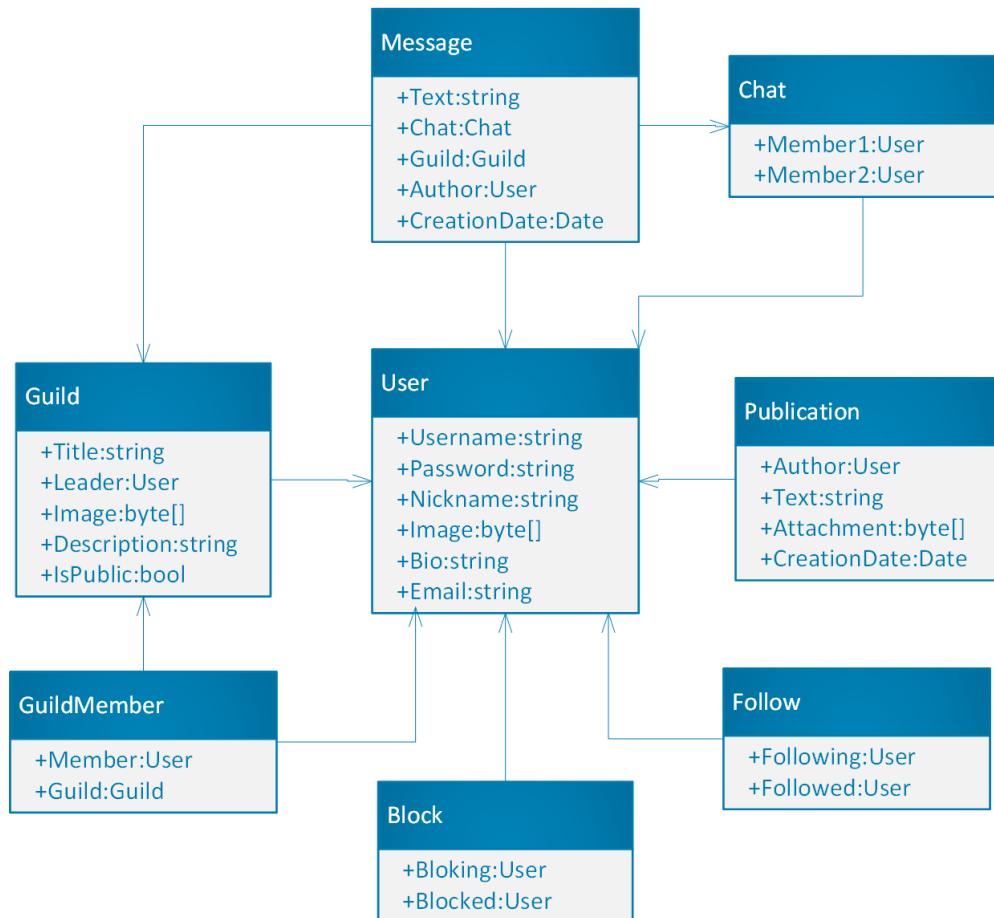


Figura 5.4. Diagrama de clases preliminar del dominio

## 5.4.2 Descripción de las Clases

A continuación, se explican de forma más detalladas las clases mostradas en el diagrama anterior.

<u>Nombre de la Clase</u>
Guild
Descripción
Modela un grupo de usuarios personalizable en el sistema
Atributos Propuestos
<b>Title:</b> Nombre del gremio <b>Leader:</b> Líder y creador del gremio <b>Image:</b> Imagen o ícono del gremio <b>Descripcion:</b> Texto descriptivo del gremio <b>Message:</b> Mensaje fijado visible solo para los miembros <b>IsPublic:</b> Indica si todos pueden unirse al gremio o si solo se entra por invitación

<u>Nombre de la Clase</u>
Message
Descripción
Modela un mensaje individual o grupal en el sistema
Atributos Propuestos
<b>Text:</b> Contenido del mensaje <b>Chat:</b> Chat individual al que pertenece <b>Guild:</b> Chat grupal de gremio al que pertenece <b>Author:</b> Usuario creador del mensaje <b>CreationDate:</b> Fecha de creación del mensaje

<u>Nombre de la Clase</u>
Chat
Descripción
Modela una agrupación de mensajes individuales entre usuarios en el sistema
Atributos Propuestos
<b>Member1:</b> Usuario miembro del chat <b>Member2:</b> El otro usuario miembro del chat

<u>Nombre de la Clase</u>
Publication
Descripción
Modela un texto corto que puede contener una imagen adjunta en el sistema
Atributos Propuestos
<b>Author:</b> Usuario creador de la publicación <b>Text:</b> Contenido de la publicación <b>Attachment:</b> Imagen adjuntada a la publicación <b>CreationDate:</b> Fecha de creación de la publicación

<b>Nombre de la Clase</b>
GuildMember
<b>Descripción</b>
Modela los usuarios que forman parte de un grupo en el sistema
<b>Atributos Propuestos</b>
<b>Member:</b> Usuario miembro
<b>Guild:</b> Gremio al que pertenece

<b>Nombre de la Clase</b>
Follow
<b>Descripción</b>
Modela un seguimiento en el sistema
<b>Atributos Propuestos</b>
<b>Following:</b> Usuario que sigue
<b>Followed:</b> Usuario seguido

<b>Nombre de la Clase</b>
Block
<b>Descripción</b>
Modela un bloqueo en el sistema
<b>Atributos Propuestos</b>
<b>Blocking:</b> Usuario que bloquea
<b>Blocked:</b> Usuario bloqueado

<b>Nombre de la Clase</b>
User
<b>Descripción</b>
Modela un usuario en el sistema
<b>Atributos Propuestos</b>
<b>Username:</b> Nombre de usuario único
<b>Nickname:</b> Apodo
<b>Image:</b> Imagen de perfil
<b>Bio:</b> Biografía o presentación
<b>Email:</b> Correo electrónico
<b>Password:</b> Contraseña

## 5.5 Análisis Priorizado de Historias de Usuario y Criterios de Aceptación

A continuación, se muestra una lista detallada de las historias de usuario y sus criterios de aceptación priorizadas por el valor que proporcionan al cliente. Esto sustituiría al análisis de casos de uso de Métrica.

Número	Historia de Usuario	Criterios de Aceptación
1	<b>Como</b> usuario <b>Quiero</b> mandar un mensaje a otro usuario	El usuario receptor debe existir. El mensaje se mandará usando el nombre de usuario del destinatario.
2	<b>Como</b> usuario <b>Quiero</b> ver los mensajes recibidos de otro usuario	Se debe mostrar un listado con los últimos mensajes recibidos de cada usuario. Para cada mensaje se mostrará el emisor y la fecha de envío, además del contenido. Solo deben ser visibles aquellos mensajes destinados al usuario actual. Al hacer clic en un mensaje se mostrará una lista con todos los mensajes anteriores intercambiados con ese usuario, ordenados por fecha de creación de forma ascendente y desplazada hacia el último elemento.
3	<b>Como</b> usuario <b>Quiero</b> responder a un mensaje recibido	Solo se podrá responder a un usuario a la vez. El historial de mensajes se actualizará al momento del envío con el nuevo mensaje.
4	<b>Como</b> usuario <b>Quiero</b> crear un gremio	Se deberá exigir un nombre, una descripción e indicar si el gremio es público o privado
5	<b>Como</b> usuario <b>Quiero</b> ver todos los gremios existentes	Se mostrará una lista con todos los gremios existentes. Se podrán filtrar los gremios por un texto contenido en su nombre o en su descripción. Deberá poderse filtrar para mostrar solo los gremios a los que pertenece el usuario actual.
6	<b>Como</b> usuario <b>Quiero</b> ver el detalle de un gremio	Al hacer clic en un gremio de la lista del que se es miembro, se mostrará información detallada del mismo. Esto incluirá imagen, número de miembros, fecha de creación y nombre del líder.
7	<b>Como</b> usuario <b>Quiero</b> unirme a un gremio	Solo será posible unirse a gremios a públicos y el usuario no debe ser miembro del gremio en cuestión.
8	<b>Como</b> usuario <b>Quiero</b> abandonar un gremio	Se podrá abandonar un gremio en cualquier momento. El creador del gremio no podrá abandonar.
9	<b>Como</b> usuario <b>Quiero</b> ver los mensajes de otros miembros del gremio	Cada gremio dispondrá de una lista de mensajes que podrán ver todos los miembros. Para cada mensaje se mostrará el emisor, la fecha de envío y su contenido. Estarán ordenados por fecha de creación de forma ascendente y la lista estará desplazada hacia el último elemento.

<b>10</b>	<b>Como</b> usuario <b>Quiero</b> enviar un mensaje a otros miembros del gremio	Se añadirá un mensaje a la lista de mensajes del gremio.
<b>11</b>	<b>Como</b> usuario <b>Quiero</b> ver todos los miembros de un gremio	Solo se podrá ver la lista de miembros si se es miembro de dicho gremio.
<b>12</b>	<b>Como</b> usuario <b>Quiero</b> invitar a otros usuarios a un gremio	Solo se podrá invitar a otros usuarios si se es miembro del gremio en cuestión.
<b>13</b>	<b>Como</b> usuario <b>Quiero</b> ver el perfil de otro usuario	Se debe mostrar la imagen, el apodo, el nombre de usuario y la descripción.
<b>14</b>	<b>Como</b> usuario <b>Quiero</b> seguir a otro usuario	No se puede seguir a usuarios bloqueados o que están bloqueando al usuario que sigue. No se puede volver a seguir a un usuario al que ya se sigue.
<b>15</b>	<b>Como</b> usuario <b>Quiero</b> publicar un mensaje para mis seguidores	El mensaje será visto sólo por los seguidores del usuario que publica.
<b>16</b>	<b>Como</b> usuario <b>Quiero</b> ver las publicaciones de los usuarios a los que sigo	Se mostrará una lista con todas las publicaciones de los usuarios a los que sigo y las propias. Para cada publicación se mostrará: imagen, nombre y apodo del creador junto con el contenido
<b>17</b>	<b>Como</b> usuario <b>Quiero</b> dejar de seguir a otro usuario	Se podrá dejar de seguir en cualquier momento a cualquier usuario al que se esté siguiendo. Al dejar de seguir desaparecerán las publicaciones de dicho usuario de la lista.
<b>18</b>	<b>Como</b> usuario <b>Quiero</b> bloquear a otro usuario	Al bloquear un usuario se cancelarán todos los seguimientos entre ellos. Los usuarios bloqueados no se pueden mandar mensajes entre ellos ni seguirse. Esta limitación no afecta a los chats grupales de los gremios. Las publicaciones del usuario bloqueado desaparecerán de la lista.
<b>19</b>	<b>Como</b> usuario <b>Quiero</b> ver a quienes sigo	Se mostrará un listado con los usuarios a los que sigue el usuario actual
<b>20</b>	<b>Como</b> usuario <b>Quiero</b> ver mis seguidores	Se mostrará un listado con los seguidores del usuario actual
<b>21</b>	<b>Como</b> usuario <b>Quiero</b> ver a quien bloquee	Se mostrará un listado con los usuarios a los que bloquea el usuario actual
<b>22</b>	<b>Como</b> usuario <b>Quiero</b> desbloquear a otro usuario	Tras el desbloqueo se permitirán nuevos seguimientos y será posible el envío de mensajes entre los usuarios.
<b>23</b>	<b>Como</b> usuario <b>Quiero</b> poder cambiar mi apodo	Debe de confirmarse el cambio.
<b>24</b>	<b>Como</b> usuario <b>Quiero</b> poder cambiar mi descripción	Debe de confirmarse el cambio.
<b>25</b>	<b>Como</b> usuario <b>Quiero</b> poder cambiar mi email	Debe de confirmarse el cambio.

26	<b>Como</b> usuario <b>Quiero</b> cambiar la descripción de un gremio	Solo el creador del gremio puede modificar los datos de este. Debe confirmarse el cambio.
27	<b>Como</b> usuario <b>Quiero</b> cambiar el nombre de un gremio	Solo el creador del gremio puede modificar los datos de este. Debe confirmarse el cambio.
28	<b>Como</b> usuario <b>Quiero</b> cambiar si mi gremio es público o privado	Solo el creador del gremio puede modificar los datos de este. Debe confirmarse el cambio.
29	<b>Como</b> usuario <b>Quiero</b> expulsar miembros de un gremio	Solo el creador del gremio puede expulsar miembros y lo podrá hacer en cualquier momento.
30	<b>Como</b> usuario <b>Quiero</b> cambiar mi foto de perfil	Solo se permitirá seleccionar imágenes de la galería del dispositivo.
31	<b>Como</b> usuario <b>Quiero</b> cambiar la imagen de un gremio	Solo se permitirá seleccionar imágenes de la galería del dispositivo.
32	<b>Como</b> usuario <b>Quiero</b> borrar una publicación	Solo se pueden borrar publicaciones propias.
33	<b>Como</b> usuario <b>Quiero</b> registrarme en la aplicación	Se introducirá el nombre de usuario (texto), el apodo (texto), el email (texto con formato de email) y la contraseña (texto). Se pedirá introducir la contraseña una segunda vez para confirmarla. Todos los campos son obligatorios. El nombre de usuario debe ser único, si ya hay un usuario registrado con ese mismo nombre, el registro no se podrá realizar. La contraseña debe guardarse encriptada.
34	<b>Como</b> usuario <b>Quiero</b> entrar en la aplicación introduciendo mis credenciales	Debe introducirse el nombre de usuario y la contraseña.
35	<b>Como</b> usuario <b>Quiero</b> poder cambiar mi contraseña	Se pedirá introducir la vieja contraseña 1 vez y la nueva 2 veces.
36	<b>Como</b> usuario <b>Quiero</b> contactar con el líder de un gremio	Se enviará un mensaje individual al usuario creador del gremio. Esto debe poder hacerse desde el propio gremio.
37	<b>Como</b> usuario <b>Quiero</b> borrar un gremio	Solo el creador del gremio podrá borrarlo. Se expulsará de forma automática a todos los miembros antes del borrado.
38	<b>Como</b> usuario <b>Quiero</b> fijar un mensaje para todos los miembros de un gremio	Solo el creador del gremio puede modificar los datos de este. Debe confirmarse el cambio.
39	<b>Como</b> usuario <b>Quiero</b> adjuntar una imagen a una publicación antes de enviarla	Solo se podrán seleccionar imágenes de la galería. Solo se podrá adjuntar como máximo una imagen por publicación. Se podrá cambiar y borrar el adjunto todas las veces que se quiera antes de confirmar el envío. La imagen adjunta se mostrará para la publicación en la lista de publicaciones a tamaño reducido y al hacer clic sobre ella se podrá ver a tamaño completo.

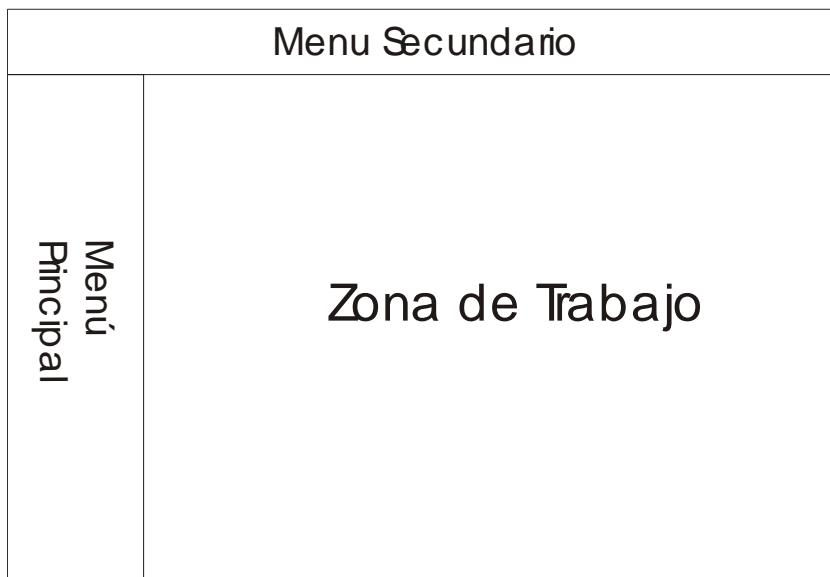
40	<b>Como</b> usuario <b>Quiero</b> mantener una sesión en el dispositivo <b>Para</b> no tener que introducir mis credenciales cada vez que abro la aplicación	Cuando se inicie sesión por primera vez se guardará el estado y no se volverán a pedir, accediendo directamente a la página principal al abrir la misma. Debe existir un botón de cerrar de sesión que permita borrar el estado para volver a introducir las credenciales.
41	<b>Como</b> usuario <b>Quiero</b> usar la aplicación sin conexión a Internet	La aplicación funcionará igual y los datos se sincronizarán cuando haya conexión. No se permitirá iniciar sesión ni registrarse, pero si entrar en la aplicación si existe una sesión iniciada.
42	<b>Como</b> usuario <b>Quiero</b> recibir notificaciones <b>Para</b> estar enterado de los eventos relevantes	Se deberán enviar notificaciones para indicar: <ul style="list-style-type: none"> <li>➤ Nuevo mensaje individual</li> <li>➤ Nuevo mensaje grupal de gremio</li> <li>➤ Nuevo seguidor</li> <li>➤ Expulsión de gremio</li> </ul>
43	<b>Como</b> usuario <b>Quiero</b> cerrar sesión en la aplicación <b>Para</b> entrar con otras credenciales	Se borrará la sesión del dispositivo y se volverá a la pantalla de login para introducir credenciales

## 5.6 Análisis de Interfaces de Usuario

A la hora de diseñar un interfaz de usuario, debemos cumplir con las normas de comunicación persona-máquina existentes, procurando que el interfaz sea usable, permita manejar el programa de manera eficiente y que no sea propenso a provocar errores en los usuarios. Por ello, normalmente se siguen guías de estilo o estándares generalizados para el tipo de aplicación que se pretende construir. En este caso se han intentado aplicar las [directrices de Material Design](#) en la medida de lo que permite Xamarin.Forms.

### 5.6.1 Descripción de la estructura de la interfaz

Todas las pantallas de la aplicación siguen la misma estructura que se muestra a continuación:



*Figura 5.1 Boceto de la interfaz*

A continuación, se procede a describir cada una de las zonas del boceto:

**Zona de Trabajo:** En esta zona se encuentra la vista donde se muestra el contenido y con la que el usuario puede interactuar. Aquí podemos encontrar por ejemplo listas con varios elementos, imágenes o botones.

**Menú Secundario:** Incluye un conjunto de acciones relacionadas con la vista actual pero que no interfieren en ella directamente. Las acciones se representan mediante imágenes que muestran la funcionalidad en Android, y mediante imágenes y texto descriptivo en Windows 10.

**Menú Principal:** Esto representa el menú de navegación. Desde él se puede cambiar de vista y acceder a las distintas partes de la aplicación. Este menú incluirá un pequeño resumen del usuario que está usando la aplicación y un conjunto de acciones con una imagen y un texto descriptivo.

## 5.6.2 Descripción del tema de color de la interfaz

Para crear una combinación de colores adecuada y que sea coherente con el logo de la aplicación se ha creado un tema de color personalizado compuesto por dos paletas basadas en los colores primario y secundario elegidos. Para crear dichas paletas se ha empleado la herramienta Material Palette Generator de Material Design disponible en <https://material.io/design/color/#tools-for-picking-colors>.

Se puede encontrar más información acerca de cómo crear temas personalizados de color en el siguiente [enlace](#).

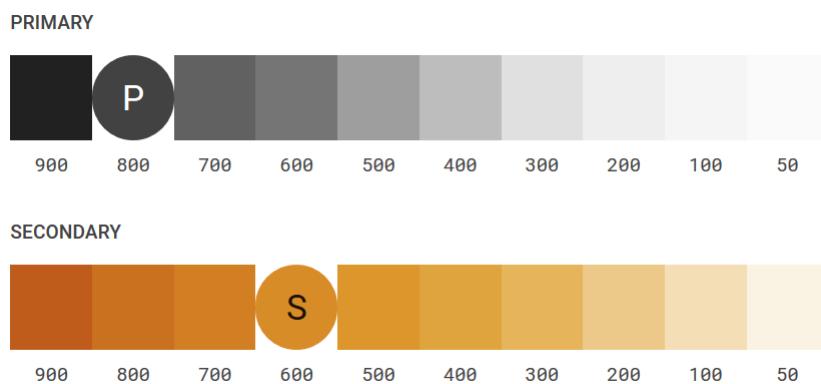


Figura 5.2 Paletas de color empleadas

## 5.6.3 Descripción del Comportamiento de la Interfaz

En la aplicación todos los errores relacionados con formularios incompletos o con un formato o valor incorrecto se indican mediante un diálogo flotante indicando el error ocurrido. La interfaz también avisará de esta manera cuando se necesite conexión a internet para realizar una determinada acción y no se disponga de ella. Las siguientes capturas muestran algunos de estos diálogos flotantes en la plataforma Android (el resultado es equivalente en Windows):



Figura 5.3 Error campo vacío

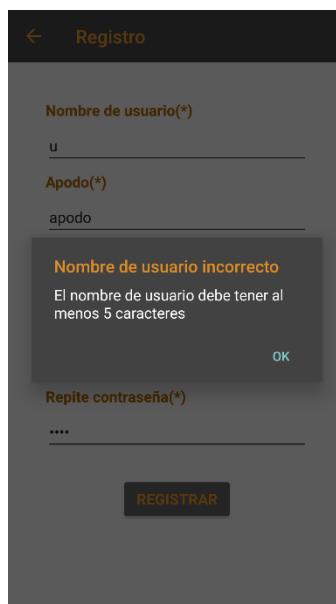


Figura 5.4 Error campo incorrecto



Figura 5.5 Error sin conexión a Internet

## 5.6.4 Diagrama de Navegabilidad

En esta sección se incluye un diagrama que muestra la navegación entre las diferentes pantallas de la aplicación. Las flechas rojas indican acciones sobre un menú (principal o secundario) y las flechas verdes indican acciones sobre la zona de trabajo.

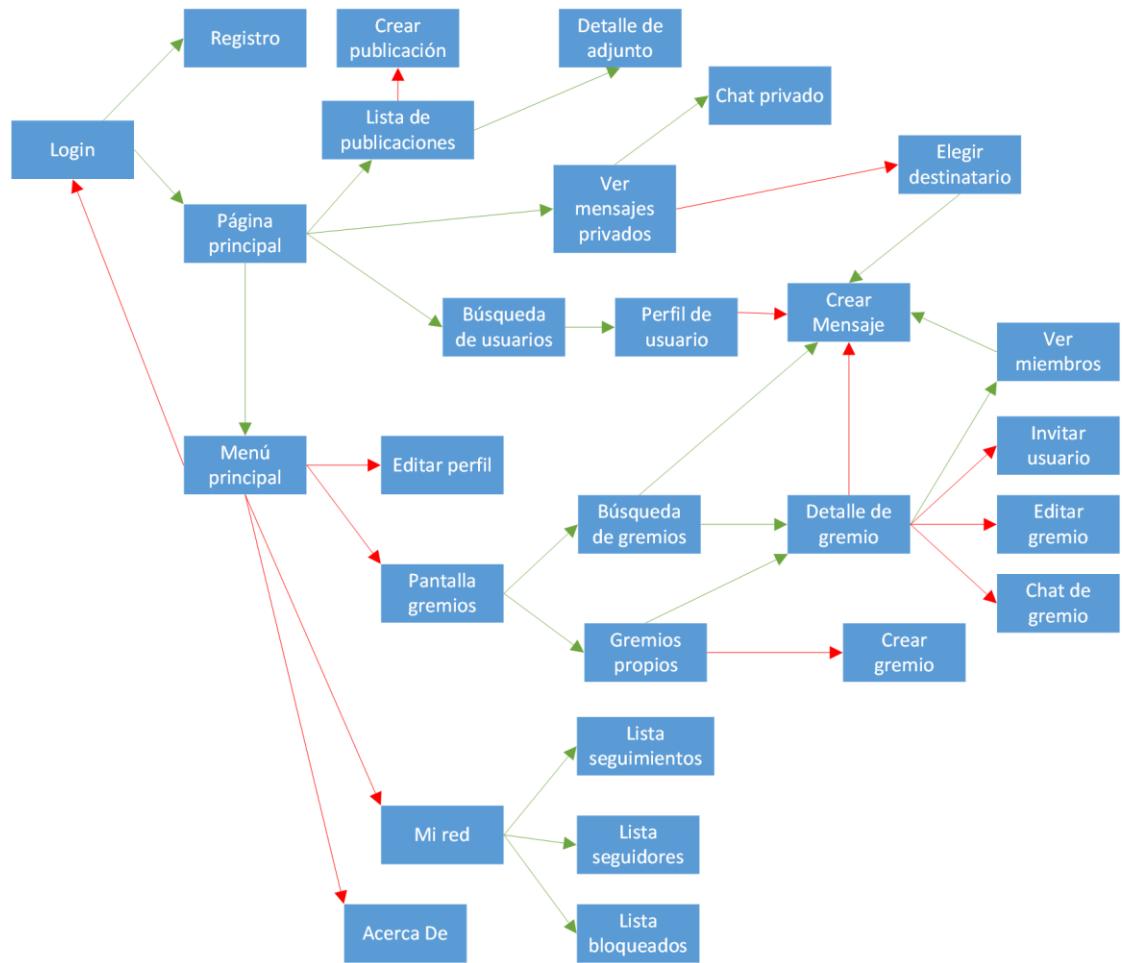


Figura 5.6 Diagrama de Navegabilidad

## 5.7 Especificación del Plan de Pruebas

En esta sección crearemos y diseñaremos el plan de pruebas de la aplicación y sus funciones, así como todos los mecanismos que utilizaremos para detectar defectos y corregirlos ya en la fase de implementación.

Las pruebas contemplarán aspectos tanto de funcionalidad de la aplicación como de aspectos de los usuarios o clientes de esta.

Se contemplarán cuatro tipos/niveles de pruebas:

- **Pruebas Unitarias:** Una prueba unitaria es una forma de probar el funcionamiento de una clase individual que cumple con una función concreta. Esto sirve para comprobar que cada clase funcione correctamente por separado. Las clases con simples operaciones CRUD no se incluyen en las pruebas unitarias. Se realizarán tres tipos de pruebas unitarias:
  - Automatizadas usando MS Test en Visual Studio para cubrir todas las clases de la lógica de negocio.
  - Manuales para probar el funcionamiento del servicio web usando [Postman](#).
  - Manuales para probar la funcionalidad de la interfaz sobre algunas pantallas concretas.
- **Pruebas de Sistema:** Estas pruebas abarcan la funcionalidad en la que interviene el sistema completo. En el caso de este sistema, las notificaciones y la sincronización. Se realizarán de forma manual.
- **Pruebas de Usabilidad:** Indican como de sencilla y usable es la interfaz de usuario. Se realizan pruebas con varios usuarios para determinar estas características.
- **Pruebas de Accesibilidad:** Servirán para determinar si los colores y tamaños de letra empleados en los textos son legibles para la mayoría de las personas. Se realizarán de forma manual con la herramienta Color Tool de Material Design sobre todos los textos de la aplicación. La herramienta está disponible en:  
<https://material.io/tools/color/#/?view.left=1&view.right=0>
- **Pruebas estáticas de Código:** Para determinar la existencia de código duplicado y posibles bugs mediante análisis estático. Se usará SonarQube.

# Capítulo 6. Diseño del Sistema

## 6.1 Arquitectura del Sistema

Para introducir este apartado se incluye un diagrama del sistema completo con todos sus elementos, que serán detallados y explicados en diagramas posteriores.

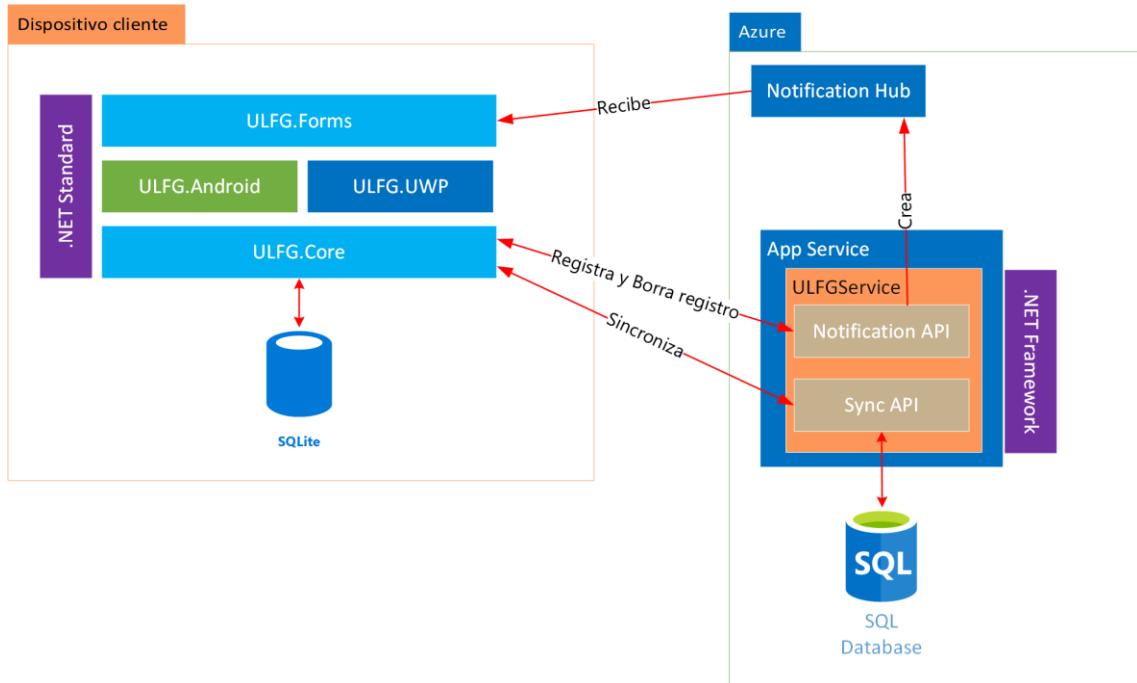


Figura 6.1 Arquitectura del Sistema Completo

## 6.1.1 Diagramas de Paquetes y Subpaquetes

En el siguiente diagrama se muestran los paquetes y subpaquetes desarrollados sin incluir código externo.



Figura 6.2 Diagrama de Paquetes y Subpaquetes

### 6.1.1.1 **ULFG.Android**

Es un paquete exclusivo de la plataforma Android. Contiene los subpaquetes para las notificaciones push (Firebase) y las implementaciones de código nativo (PlatformImpl).

### 6.1.1.2 **ULFG.UWP**

Es un paquete exclusivo de la plataforma UWP. Contiene un único paquete con las implementaciones de código nativo.

### 6.1.1.3 *ULFG.Forms*

Este paquete incluye toda la interfaz de usuario y la lógica de presentación divididas en subpaquetes según su funcionalidad.

### 6.1.1.4 *ULFG.Core*

Este paquete incluye la capa de persistencia (Data) y la lógica de negocio (Logic) separadas en subpaquetes.

Dentro de la capa de persistencia se pueden encontrar otros dos subpaquetes:

- Item: Contiene las clases del modelo
- ItemManager: Gestiona las acciones sobre las tablas de la base de datos

### 6.1.1.5 *ULFGService*

Contiene el mapeo de la base de datos y los controllers que componen la API con la que interactúa el cliente.

## 6.1.2 Diagramas de Componentes

Los diagramas de componentes muestran los diferentes componentes de un sistema y sus dependencias. Está destinado a detallar las relaciones del diagrama de arquitectura mostrado anteriormente. Se incluye un diagrama por cada bloque.

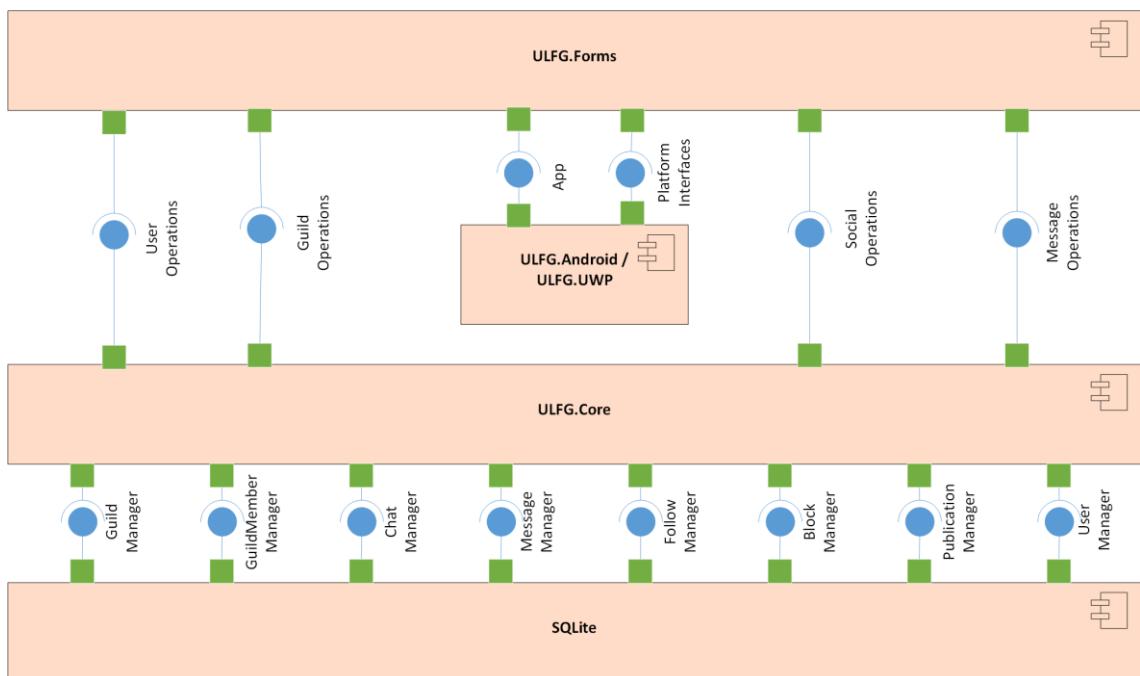


Figura 6.3 Diagrama de Componentes del Dispositivo Cliente

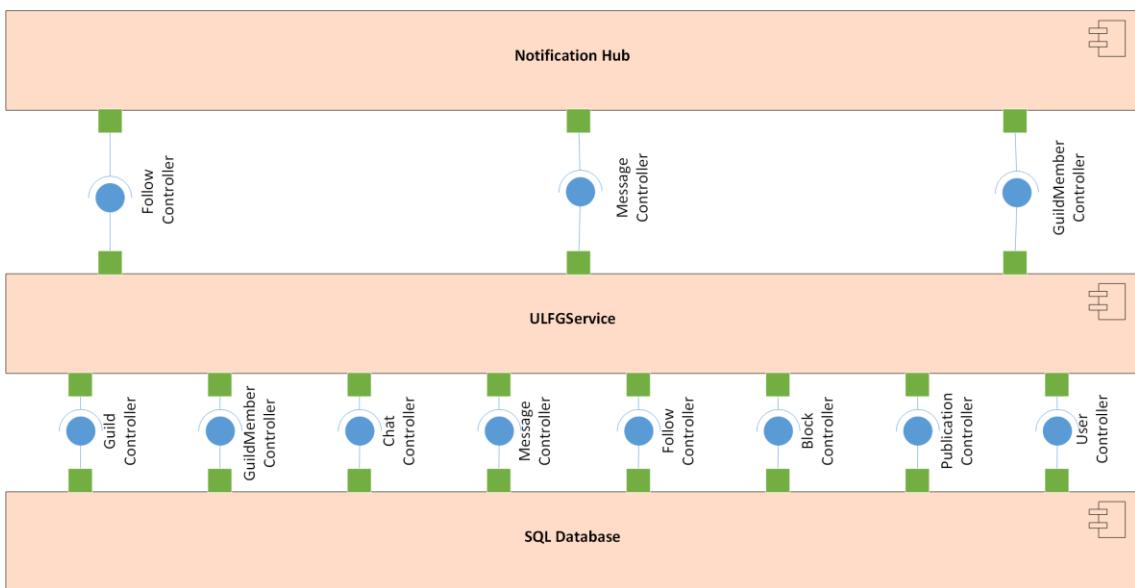


Figura 6.4 Diagrama de Componentes de Azure

### 6.1.3 Diagramas de Despliegue

En esta sección se mostrará un diagrama con el despliegue de la aplicación y posteriormente se explicará cada uno de los elementos.

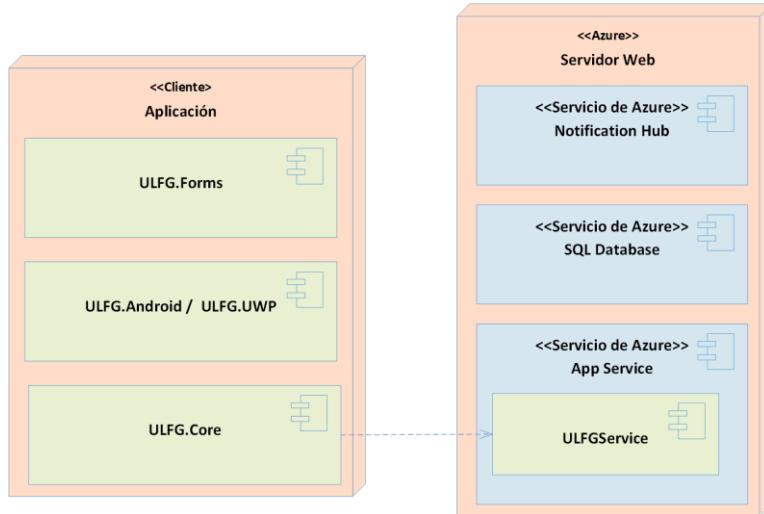


Figura 6.5 Diagrama de Despliegue

#### 6.1.3.1 Cliente

Representa el dispositivo del cliente donde se ejecuta la aplicación (Android o Windows).

#### 6.1.3.2 Azure

Contiene los servicios de Azure utilizados. Dentro del servicio AppService se despliega la API con la que interactúa el cliente.

## 6.2 Diseño de Clases

En esta sección representaremos diagramas que muestren los paquetes y las clases que formarán parte de la implementación final del sistema. En el apartado 7.4.2. se podrá encontrar una descripción detallada de cada una de las clases.

### 6.2.1 Diagrama de Clases

El diagrama de clases del sistema se dividirá en los paquetes y subpaquetes indicados en el diagrama del apartado 6.1.1. debido al tamaño de este. Todos los diagramas se han generado utilizando la herramienta de “Class Diagram” de Visual Studio. El sistema se ha construido empleando el patrón MVVM descrito en [7.1.1. Patrón MVVM](#).

#### 6.2.1.1 *ULFG.Android*

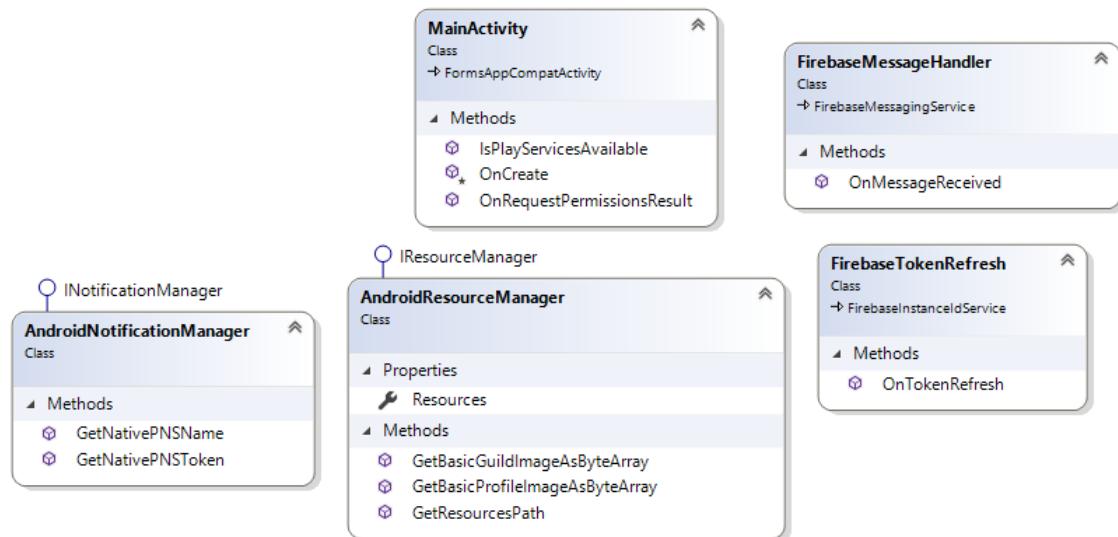


Figura 6.6 Diagrama del paquete *ULFG.Android* y sus subpaquetes

### 6.2.1.2 ULFG.UWP

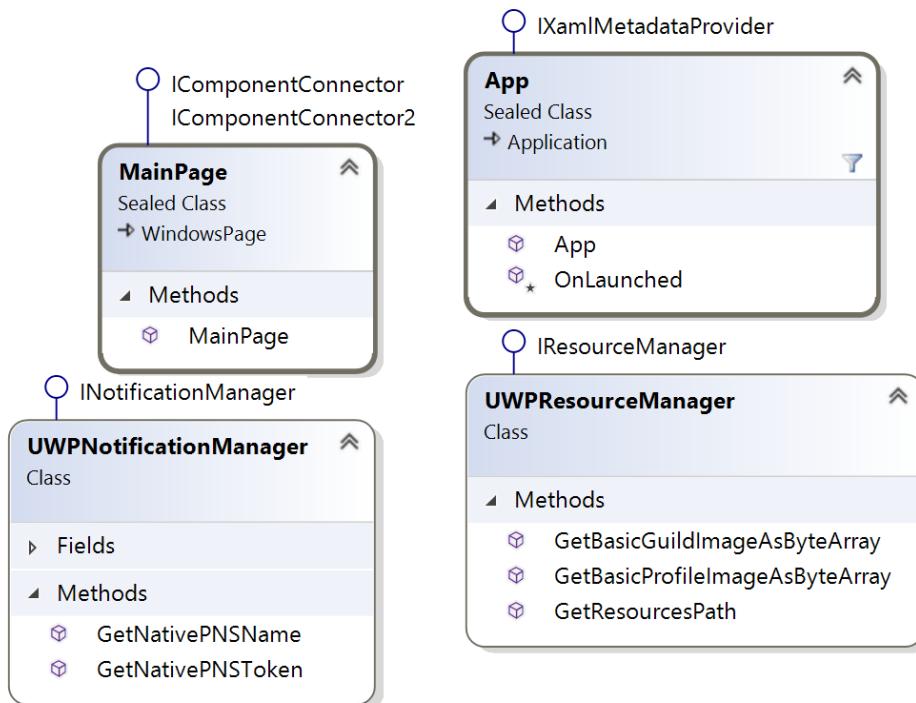


Figura 6.7 Diagrama del paquete ULFG.UWP y sus subpaquetes

### 6.2.1.3 ULFG.Core

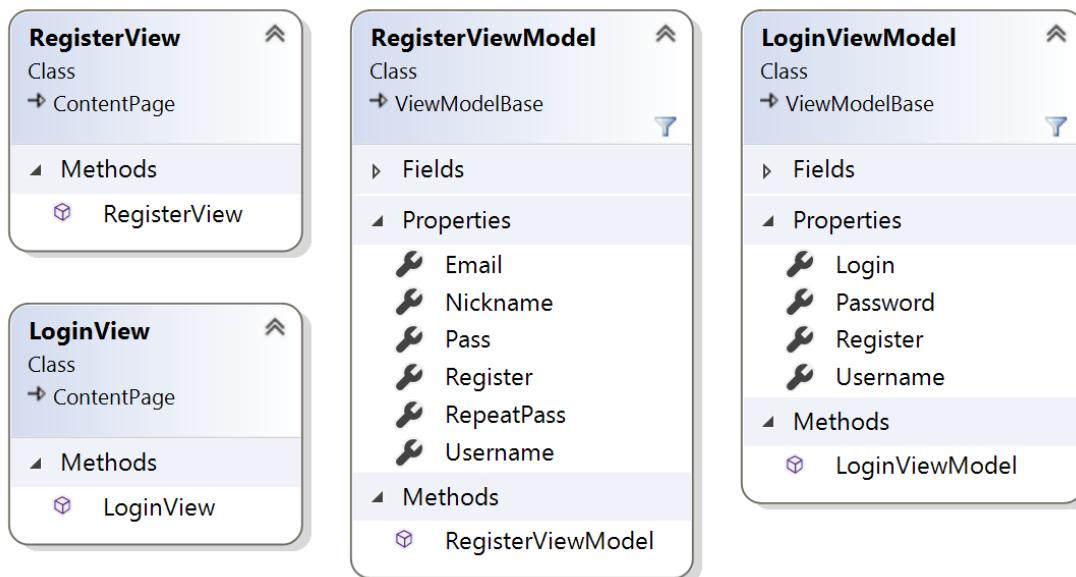


Figura 6.8 Diagrama del subpaquete Logic

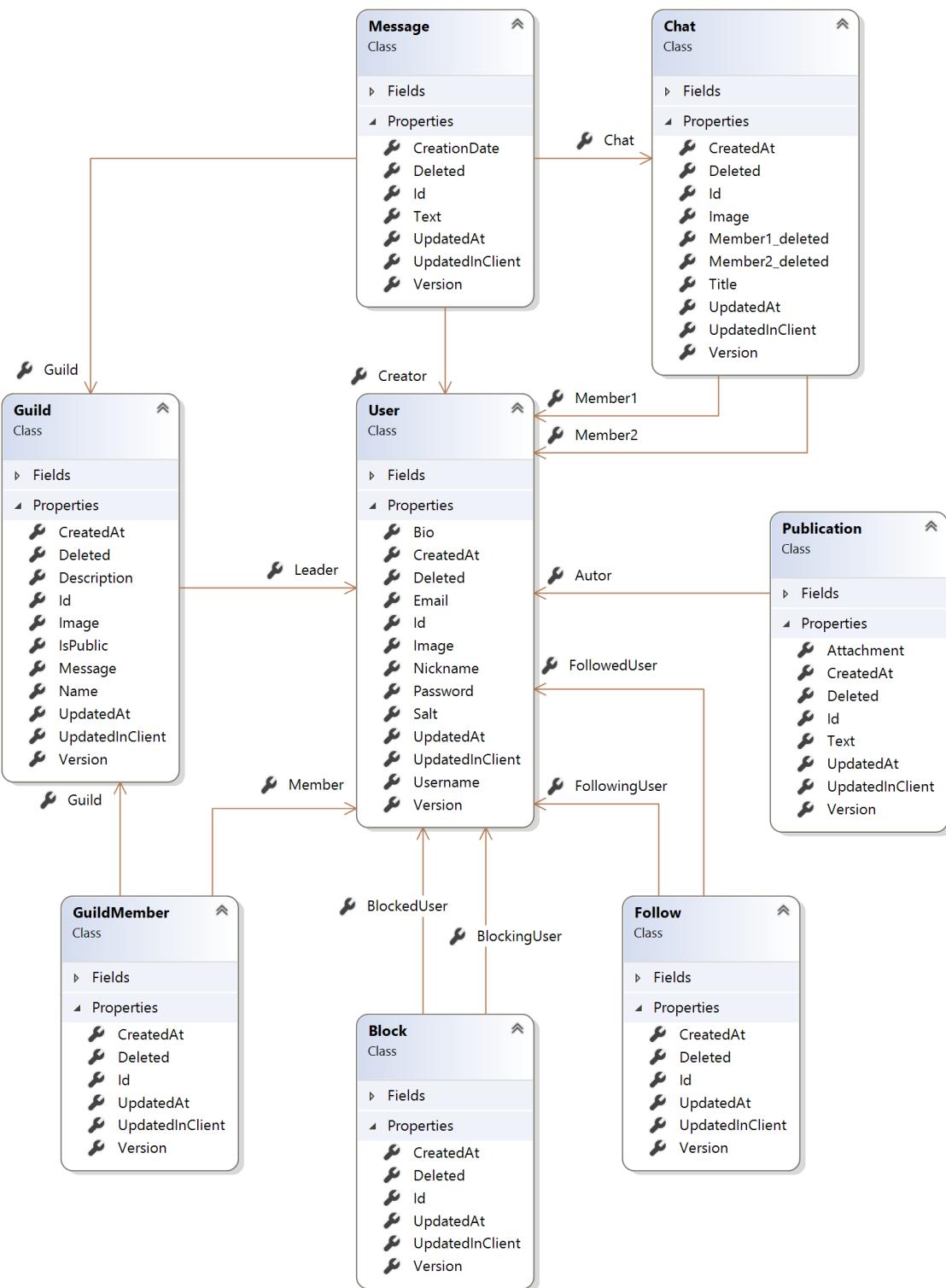


Figura 6.9 Diagrama del subpaquete Item

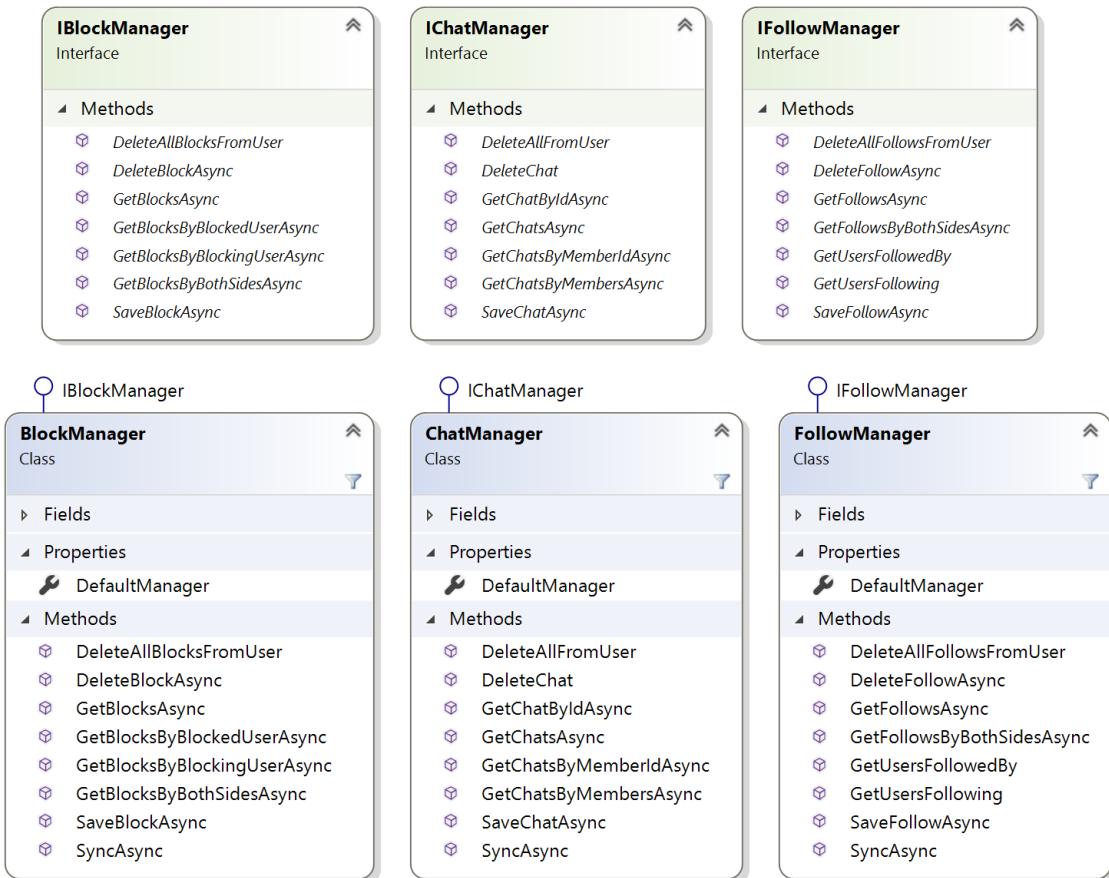


Figura 6.10 Diagrama del subpaquete *ItemManager I*

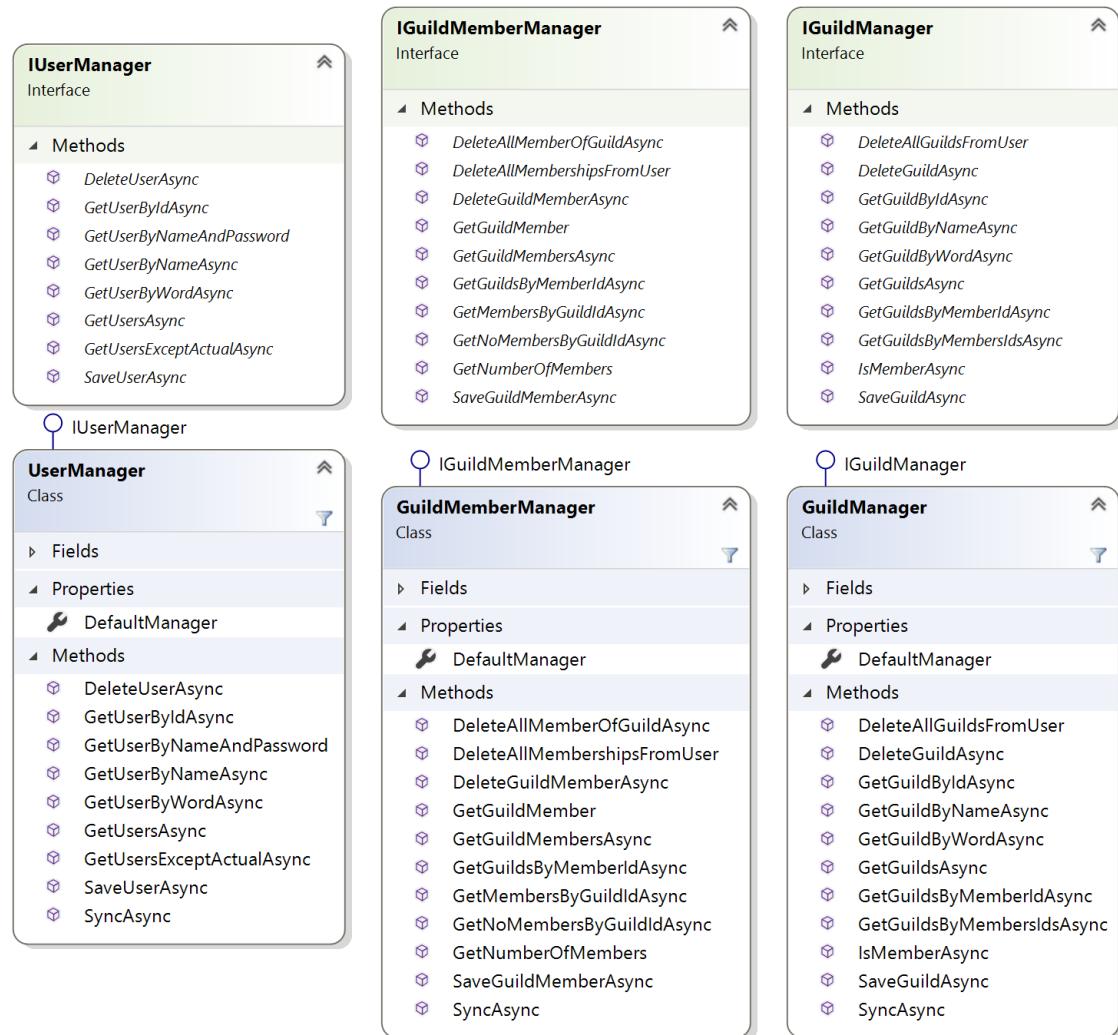


Figura 6.11 Diagrama del subpaquete *ItemManager II*

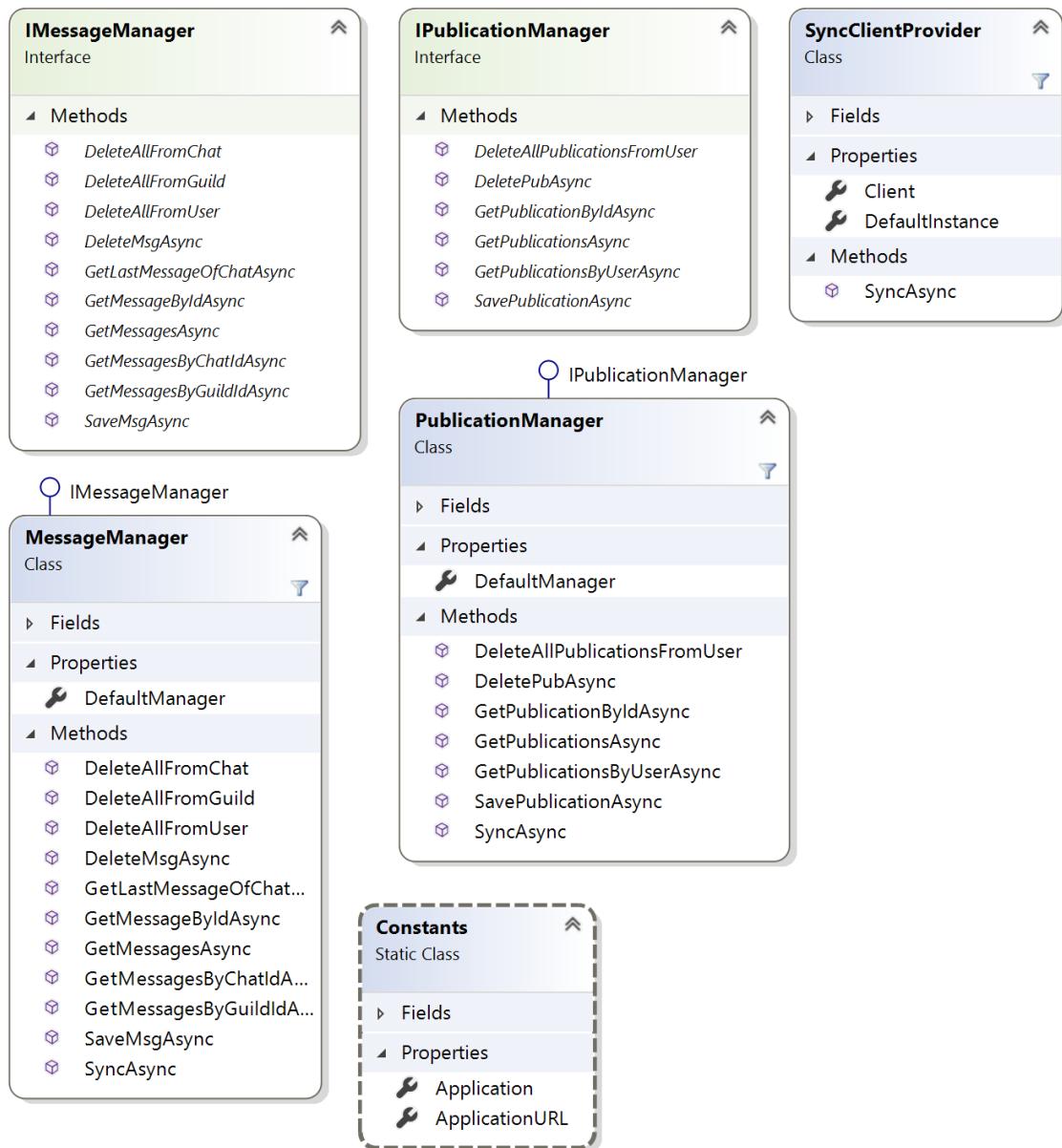


Figura 6.12 Diagrama del subpaquete *ItemManager III*

### 6.2.1.4 ULFG.Forms

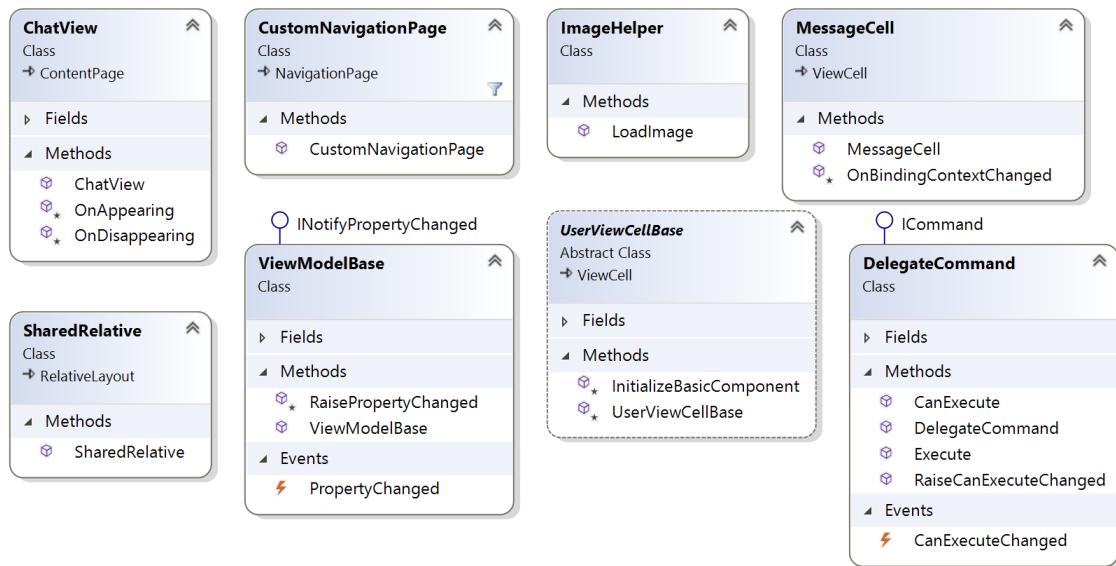


Figura 6.13 Diagrama del subpaquete Shared

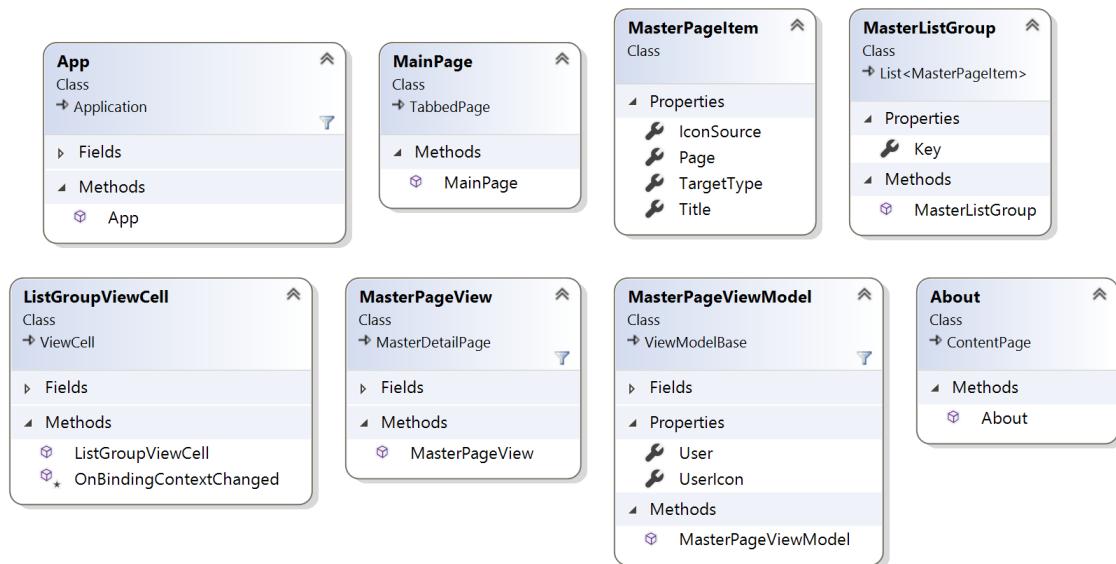


Figura 6.14 Diagrama del subpaquete Main

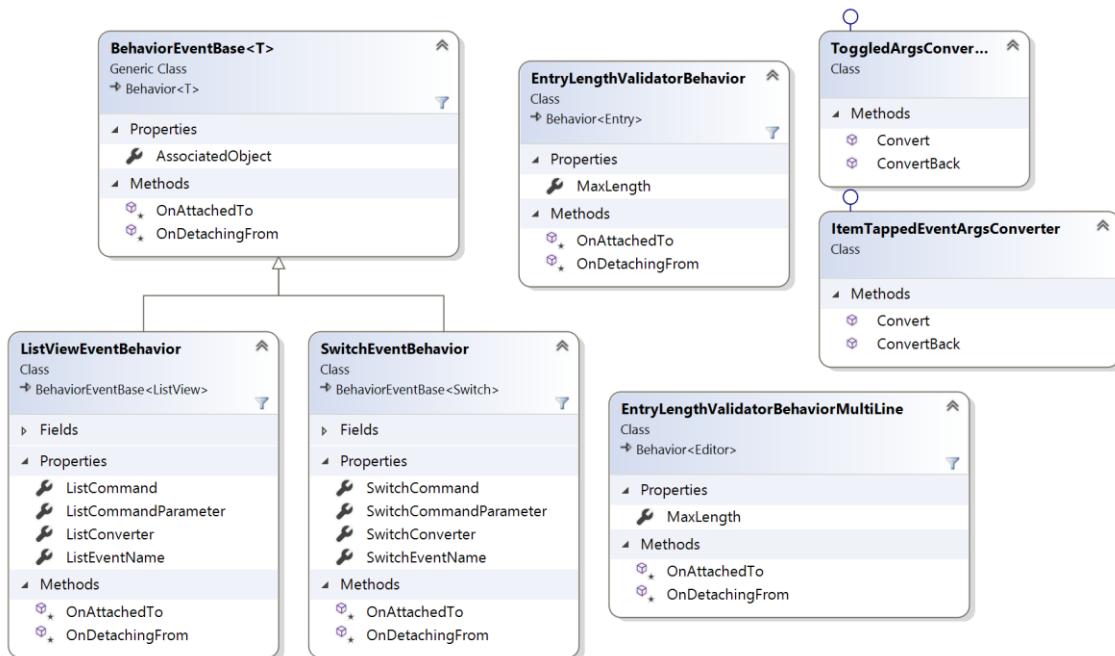


Figura 6.15 Diagrama del subpaquete Behaviors

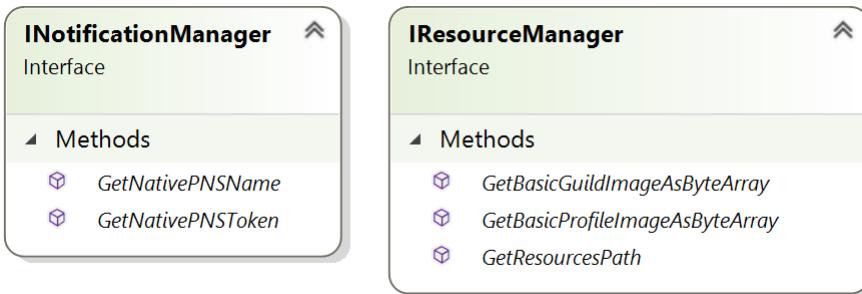


Figura 6.16 Diagrama del subpaquete PlatformInterfaces

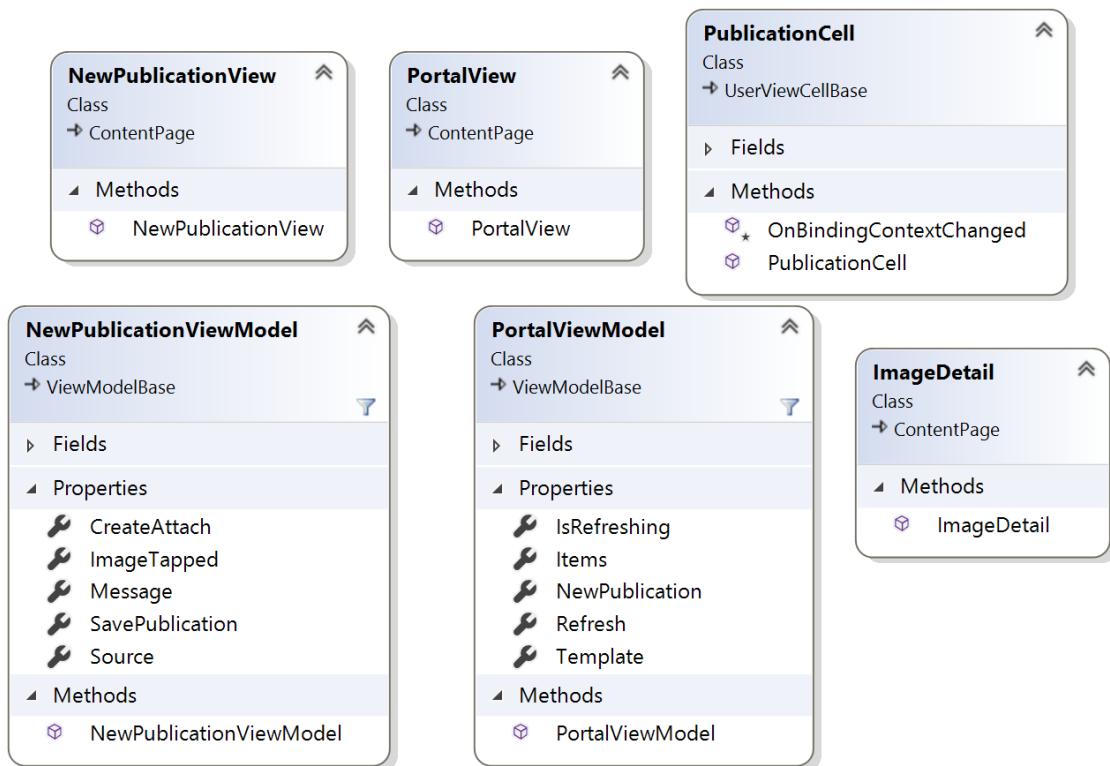


Figura 6.17 Diagrama el subpaquete Publications

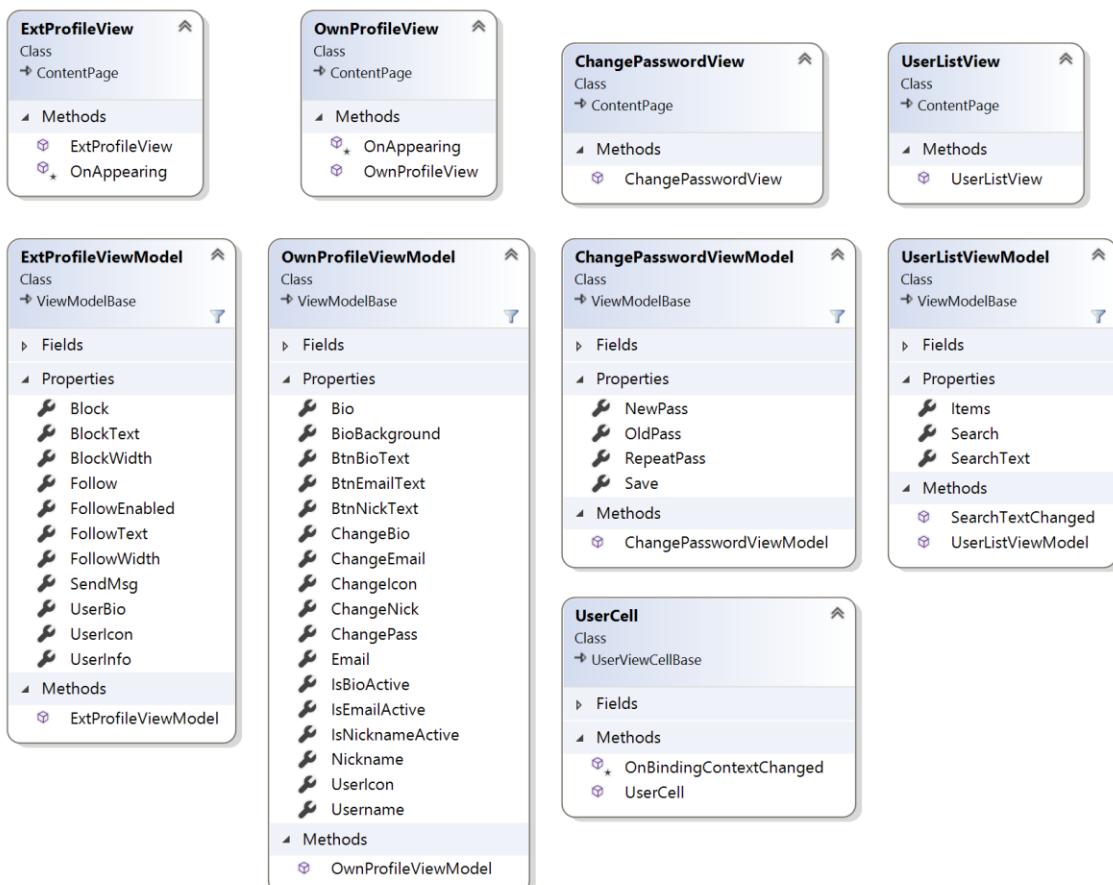


Figura 6.18 Diagrama del subpaquete Profiles

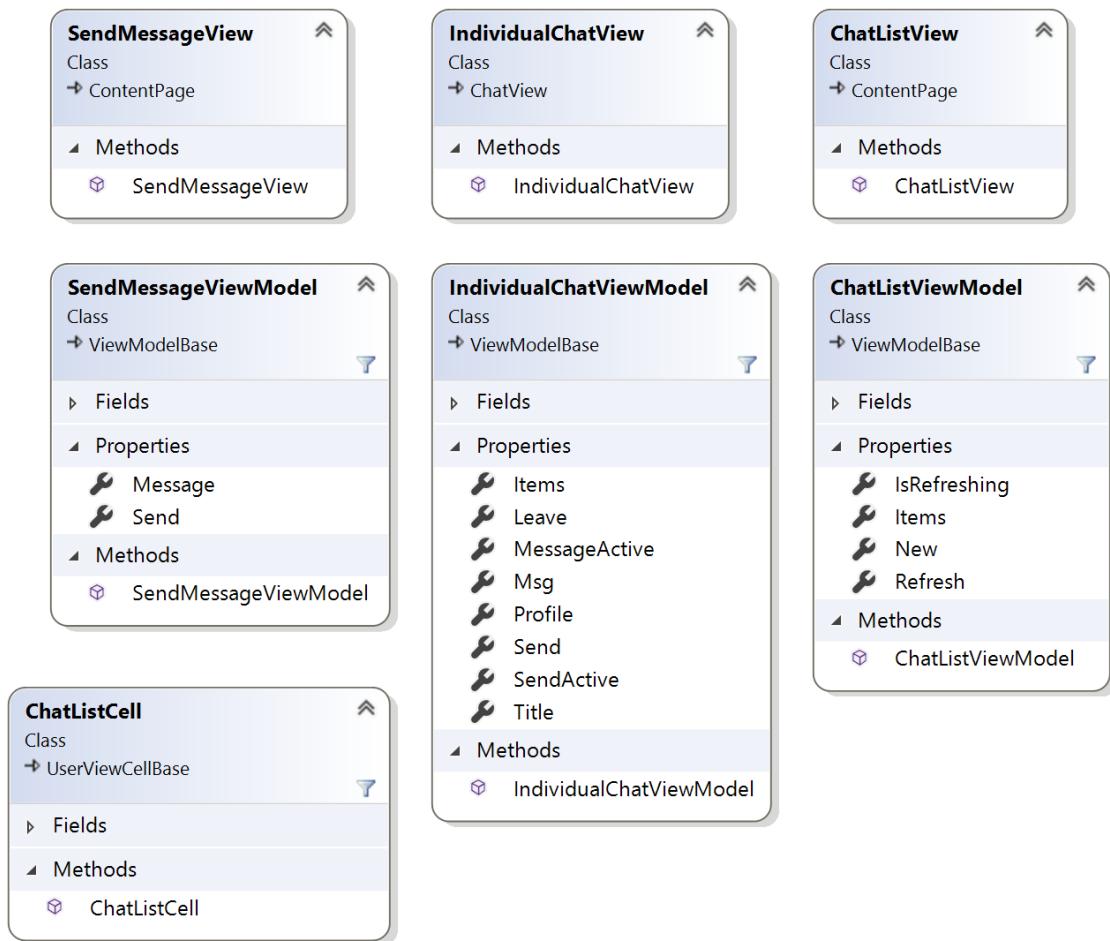


Figura 6.19 Diagrama del subpaquete PrivateChat

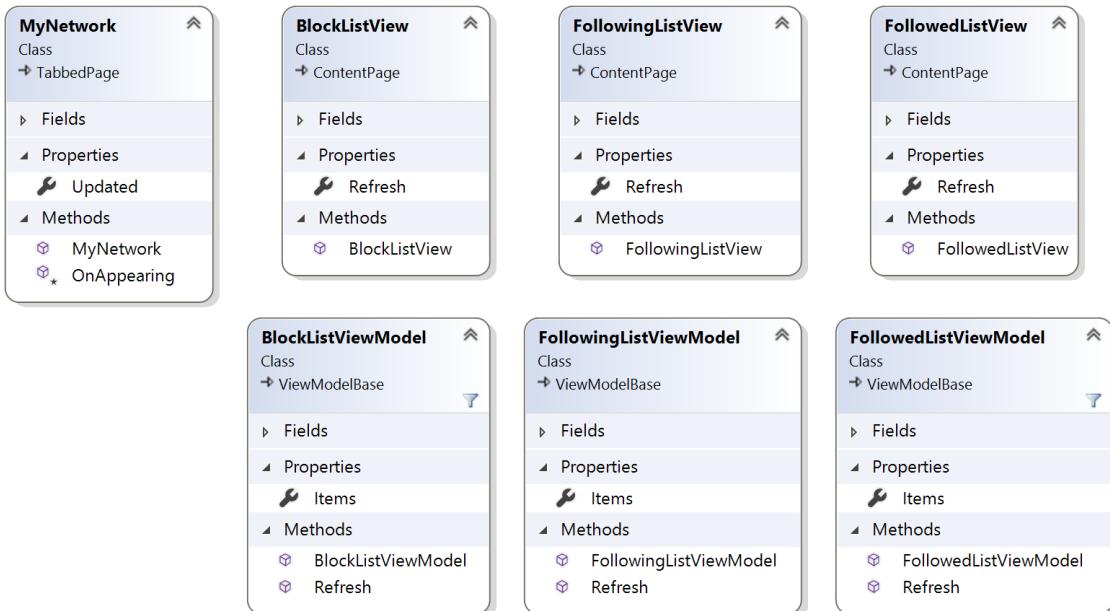


Figura 6.20 Diagrama del paquete Network

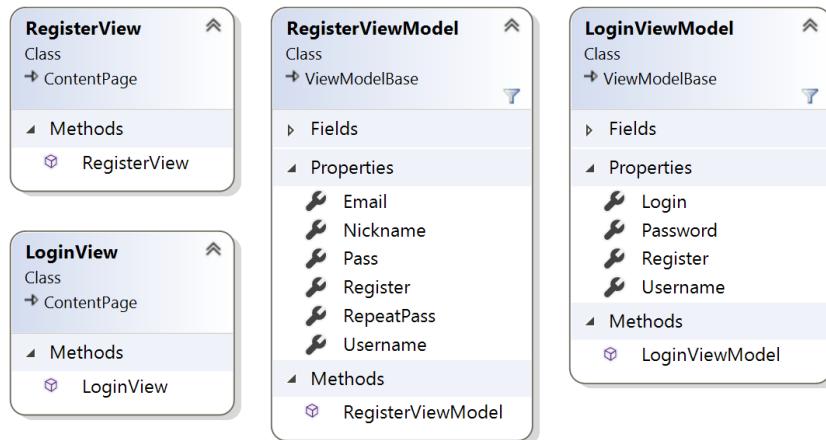


Figura 6.21 Diagrama del subpaquete Login

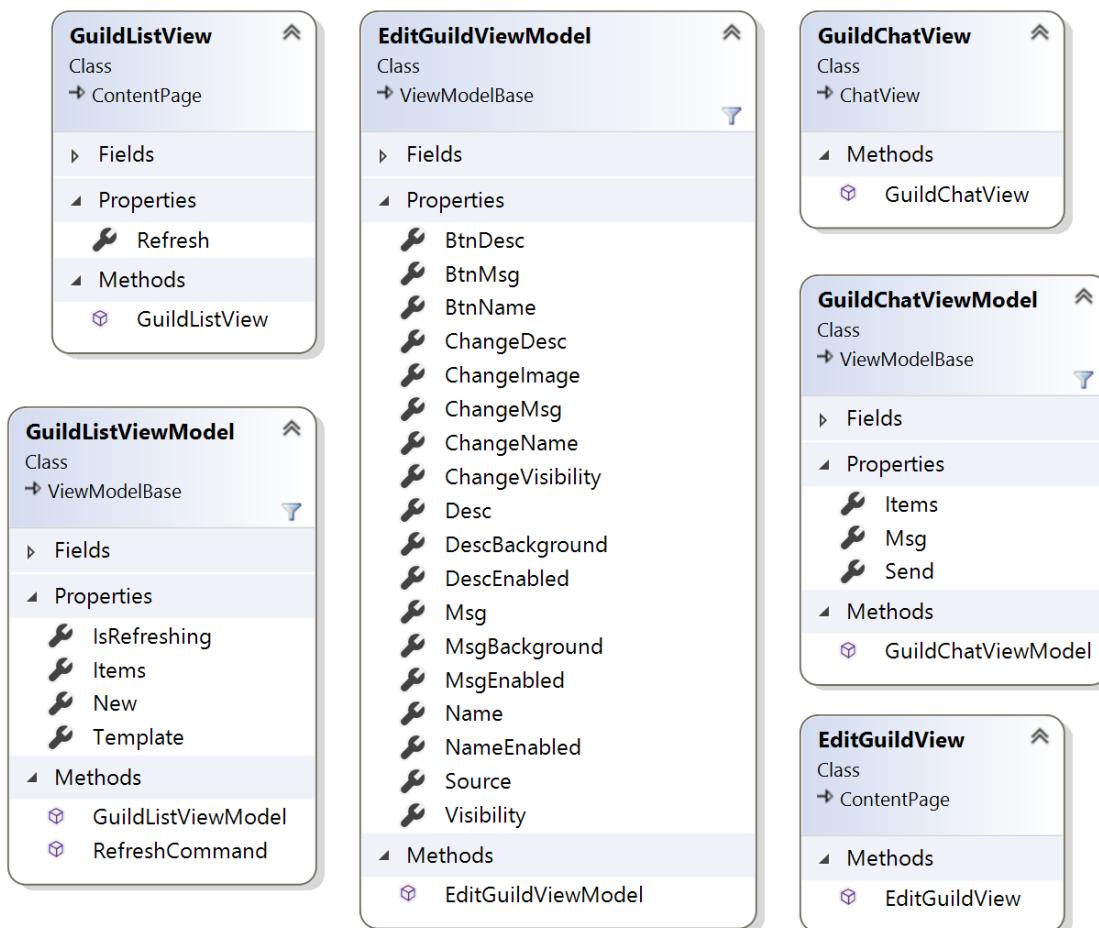


Figura 6.22 Diagrama del subpaquete Guilds

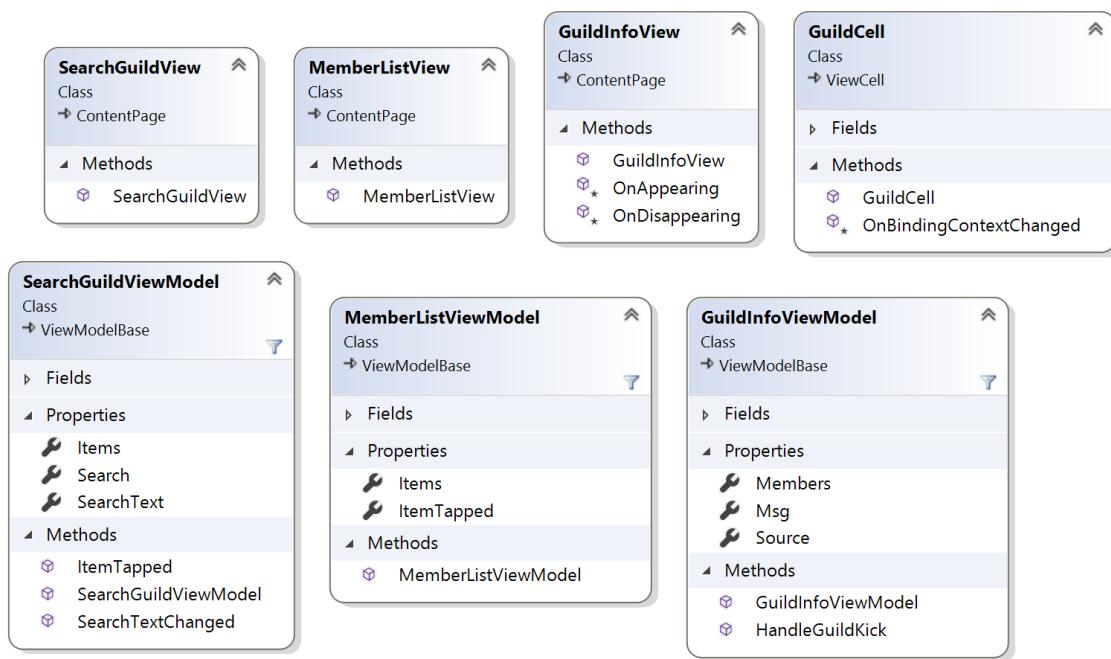


Figura 6.23 Diagrama del subpaquete **Guilds II**

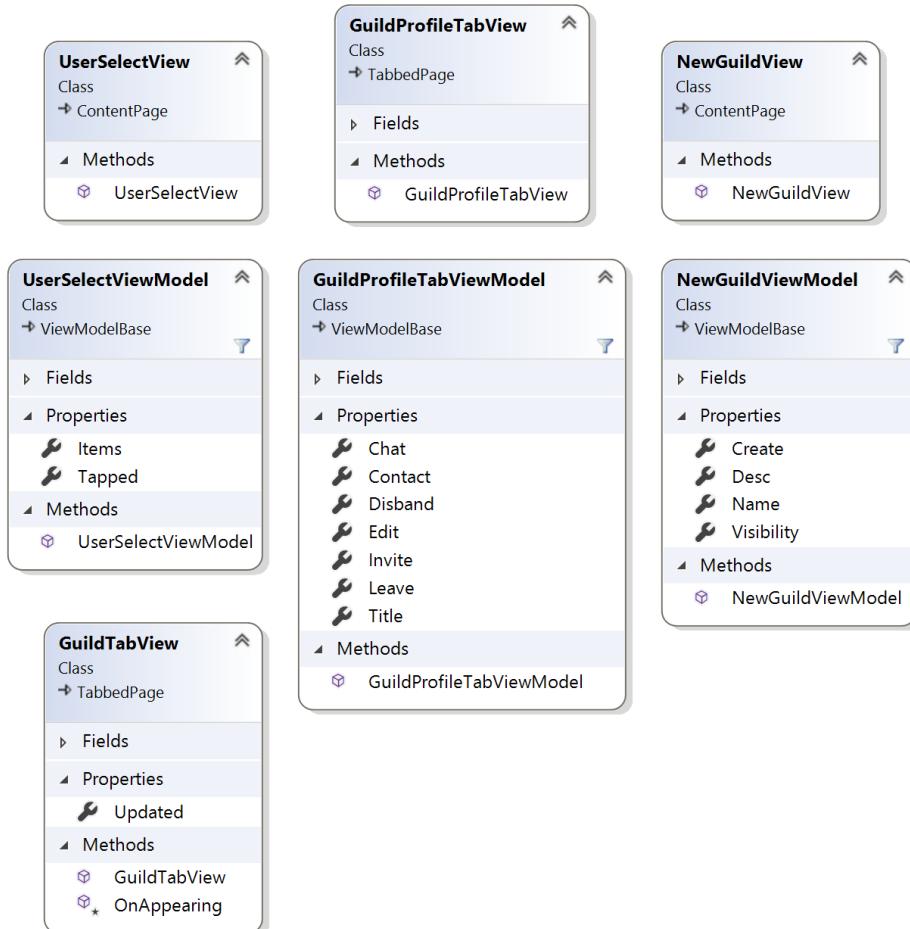


Figura 6.24 Diagrama del subpaquete **Guilds III**

### 6.2.1.5 ULGService

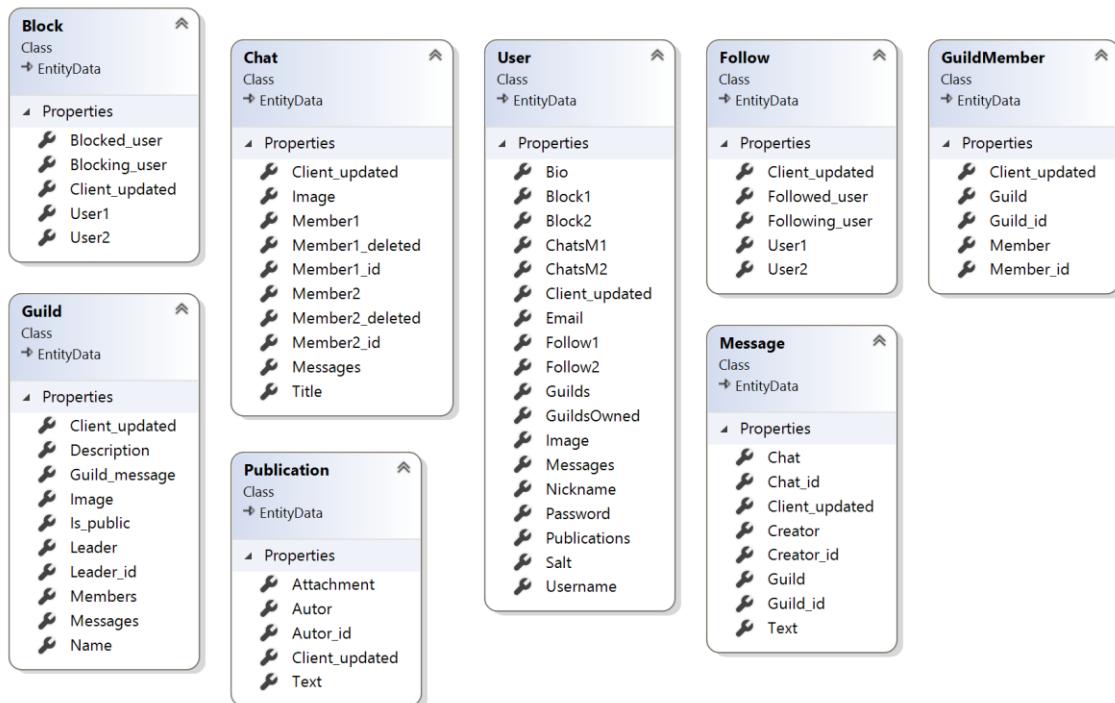


Figura 6.25 Diagrama del subpaquete DataObjects

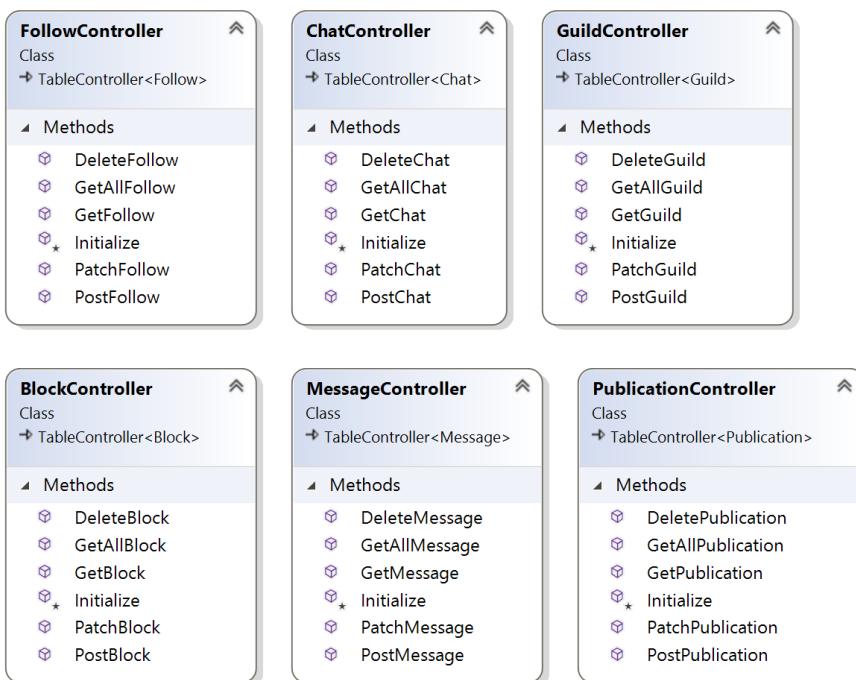


Figura 6.26 Diagrama del subpaquete Controllers I

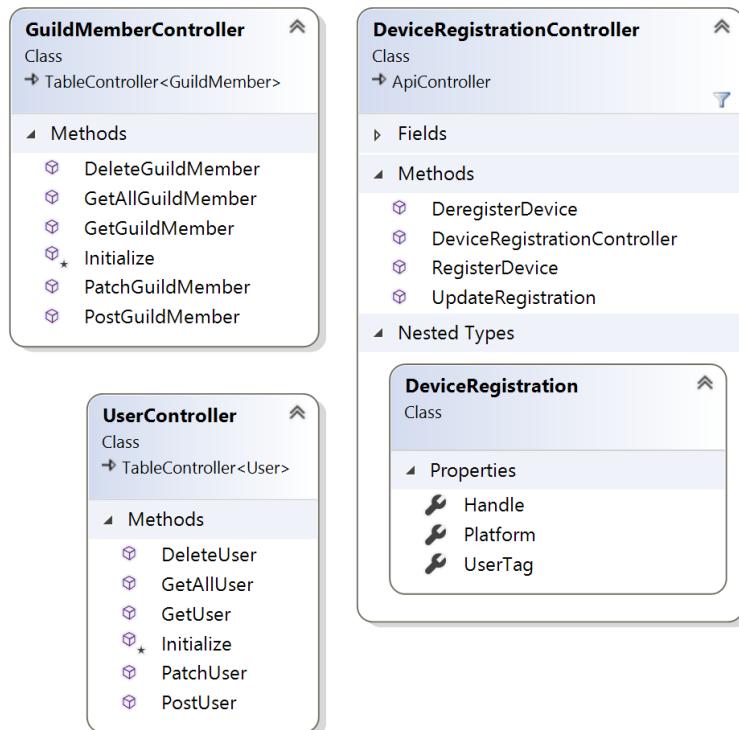


Figura 6.27 Diagrama del subpaquete **Controllers II**

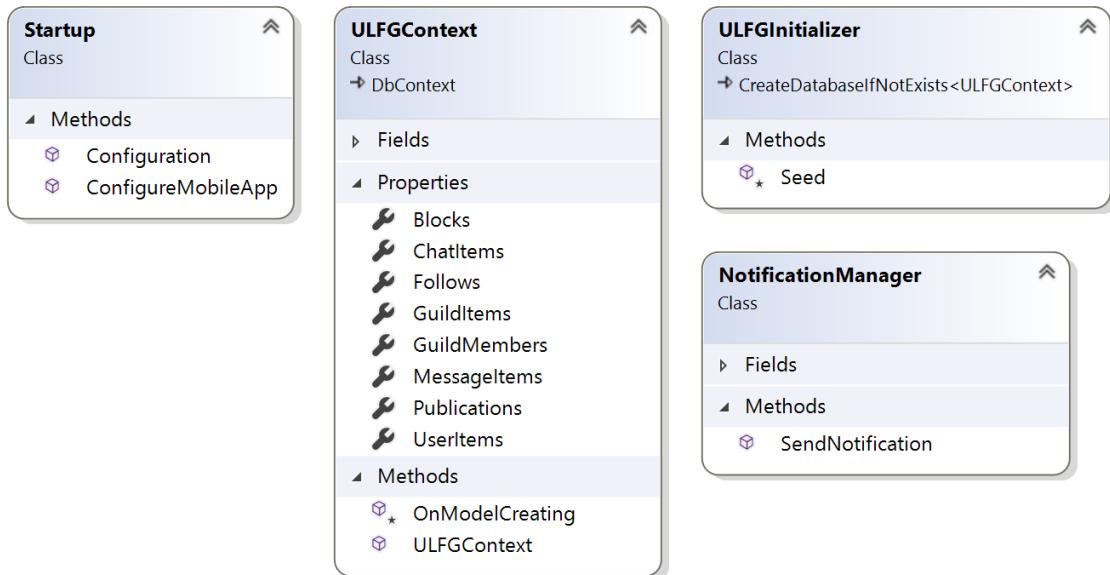


Figura 6.28 Diagrama de los subpaquetes **Service y Helpers**

## 6.3 Diagramas de Interacción

En este apartado se incluye un diagrama que muestra el proceso de envío de un mensaje privado y su correspondiente notificación al receptor.

### 6.3.1 Enviar Notificación

El siguiente diagrama muestra el proceso de enviar una notificación a un usuario cuando recibe un nuevo mensaje privado. Las demás notificaciones siguen un proceso similar. Si el emisor no dispone de conexión a internet el proceso de envío de la notificación se iniciará la siguiente vez que el emisor sincronice con la nube. Si el receptor no tiene conexión a internet la notificación quedará en espera hasta que se conecte.

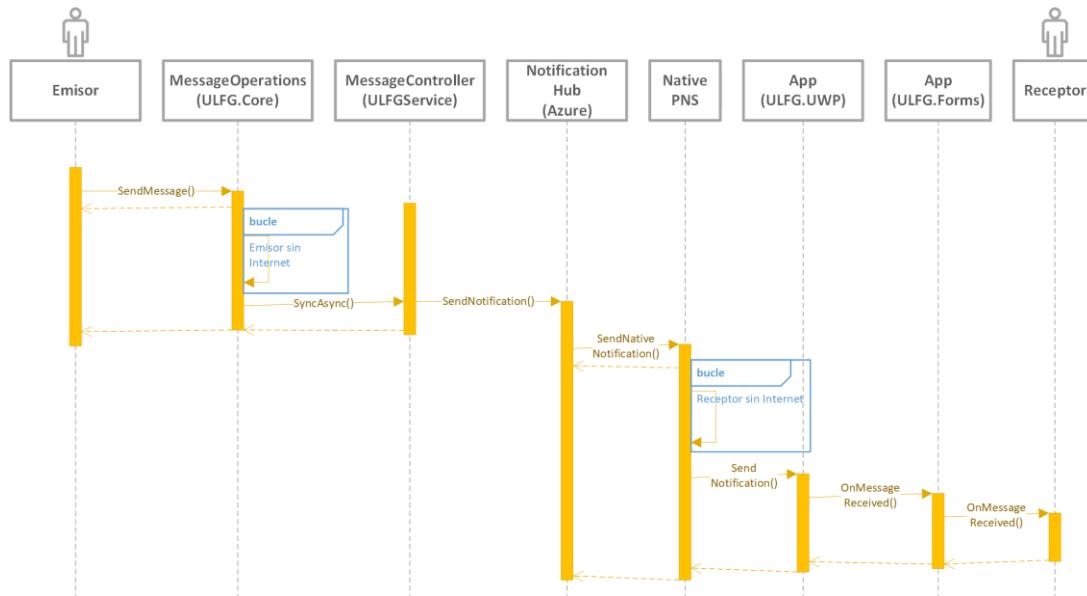
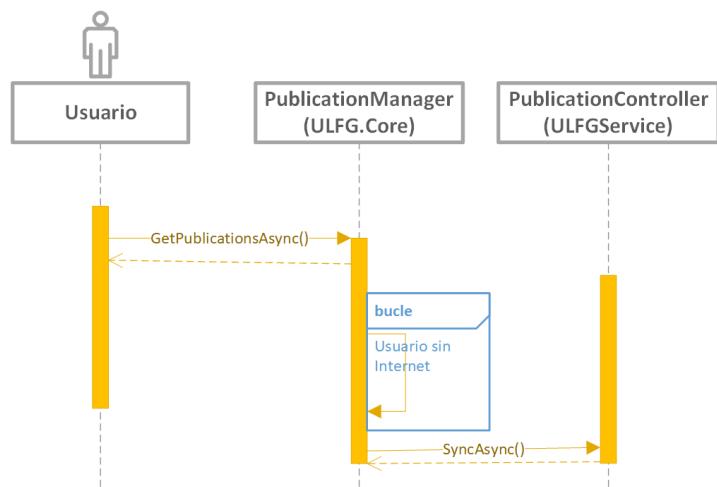


Figura 6.29 Diagrama de interacción de Enviar Notificación al enviar un mensaje directo

### 6.3.2 Sincronización Offline

Es muy común en dispositivos móviles que el usuario pueda continuar usando la aplicación aun sin conexión a Internet. En una red social el usuario podría leer y crear contenido en cualquier momento sabiendo que se enviará al servidor cuando posea conexión a Internet.

El siguiente diagrama muestra el proceso de sincronización utilizando como ejemplo la obtención de la lista de todas las publicaciones de un usuario. Si se dispone de conexión a internet se realiza una sincronización de los datos locales con la nube, en caso contrario se usan los datos locales sin actualizar.



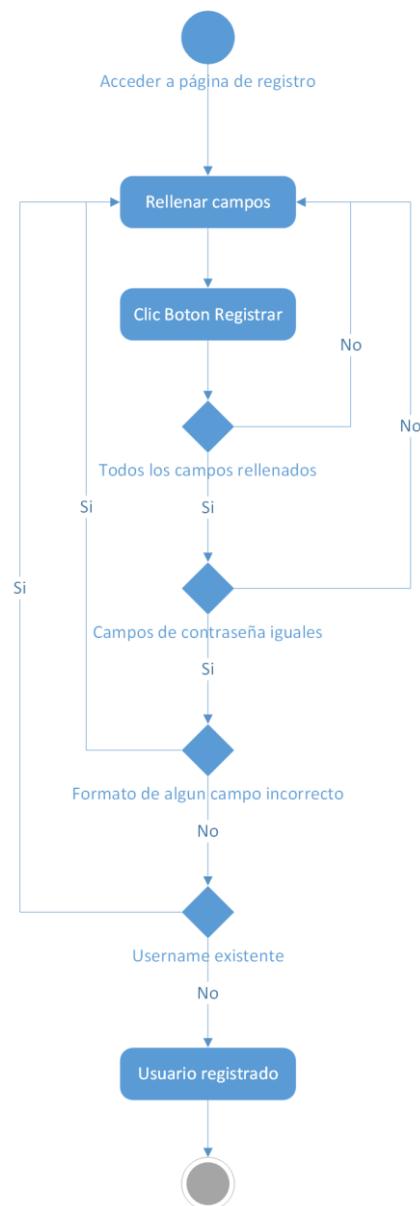
*Figura 6.30 Diagrama de interacción de la Sincronización Offline usando la lista de publicaciones como ejemplo*

## 6.4 Diagramas de Actividades

En este apartado se mostrarán una serie de diagramas de actividades para ayudar a comprender mejor algunas funcionalidades:

### 6.4.1 Registro

Este diagrama muestra el proceso para realizar un registro de un nuevo usuario en el sistema. Se comprueba que todos los campos estén rellenos, que los formatos sean correctos y que la contraseña se haya introducido bien dos veces. Si todo esto es correcto se comprueba si hay algún usuario registrado con el nombre introducido. En caso negativo se efectúa el registro.



*Figura 6.31 Diagrama de actividad del Registro*

## 6.4.2 Inicio de sesión

Al abrir la aplicación se comprueba si hay una sesión guardada en el dispositivo. En caso afirmativo se accede a la pantalla principal y en caso contrario al login para pedir credenciales.

En el login se comprueba que las credenciales sean correctas y si hay conexión a internet.

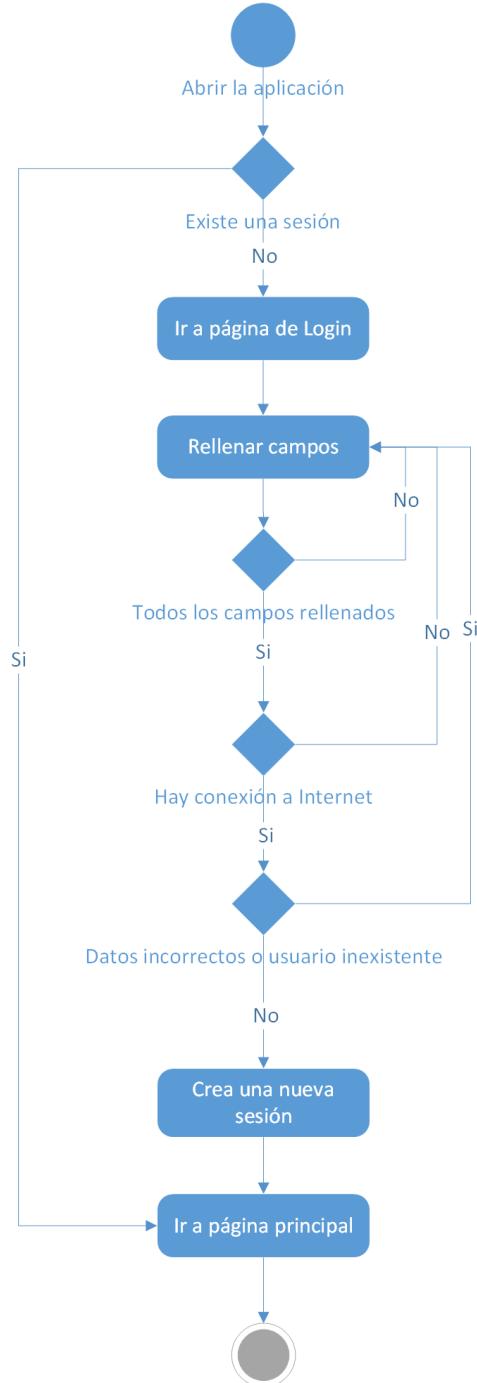


Figura 6.32 Diagrama de actividad de Inicio de Sesión

## 6.4.3 Editar campo

Cuando se edita un campo se comprueba que cumpla con el formato establecido para poder actualizarse.

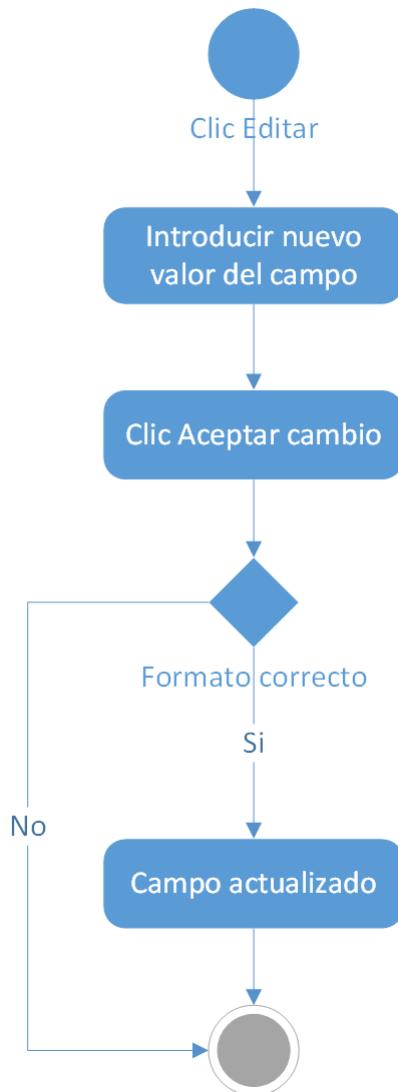


Figura 6.33 Diagrama de actividad de Editar un campo

## 6.4.4 Seguir usuario

Al seguir a un usuario se comprueba que no exista ningún bloqueo entre las partes involucradas.

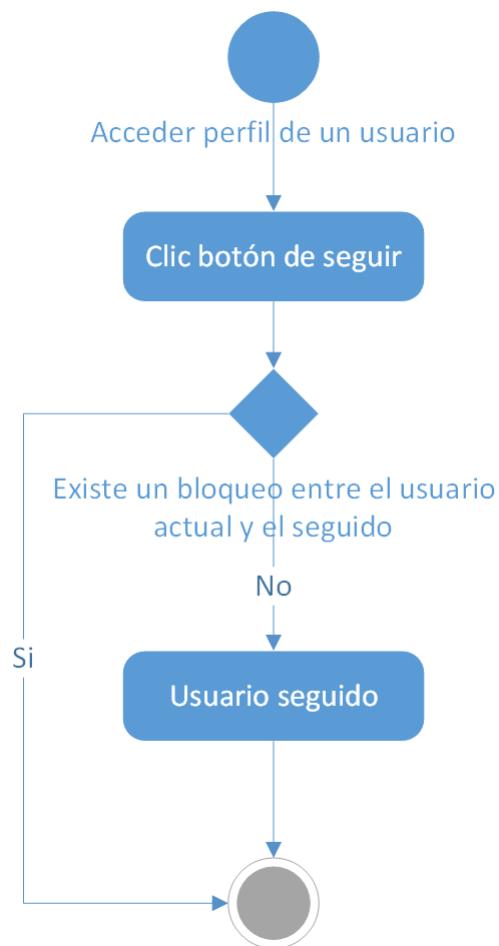


Figura 6.34 Diagrama de actividad de Seguir usuario

## 6.4.5 Enviar mensaje

Al enviar mensaje se revisa si ya existe un chat abierto entre los dos usuarios que intervienen. Si ya existe, añade el nuevo mensaje al chat y en caso contrario crea un chat nuevo y le añade el mensaje.



Figura 6.35 Diagrama de actividad de Enviar un nuevo mensaje

## 6.5 Diseño de la Base de Datos

El sistema utiliza un servicio en la nube de Azure como SGDB conocido como “Azure SQL Database”

### 6.5.1 Descripción del SGBD Usado

Azure SQL Database es un servicio de base de datos relacional inteligente en la nube totalmente administrado, multihilo y multiusuario basado en SQL Server. Ofrece un rendimiento predecible en varios niveles de servicio que proporciona escalabilidad dinámica sin tiempo de inactividad, optimización inteligente integrada, escalabilidad y disponibilidad globales, y opciones de seguridad avanzadas que incluyen copias de seguridad constantes para reducir los esfuerzos de desarrollo.

Posee unas extensas funciones de supervisión y de alerta completamente configurables que permiten monitorizar en todo momento el funcionamiento de la base de datos

Ofrece un contrato a nivel de servicio con una disponibilidad del 99,99%.

Para realizar operaciones sobre la base de datos se han utilizado únicamente el portal de Azure y Visual Studio

### 6.5.2 Integración del SGBD en el Sistema

La integración del SGDB en el sistema consta de dos partes: Cliente y Servicio web

#### 6.5.2.1 *Cliente*

La integración dentro de la parte cliente está representada por el subpaquete Data dentro de [ULFG.Core](#), visto anteriormente en el Diagrama de Clases.

#### 6.5.2.2 *Servicio web*

El servicio web hace uso de [Entity Framework](#) para mapear la base de datos. Esta formado para una serie de controladores API de tabla que actúan como un punto intermedio entre el cliente y el servidor de base de datos. Está representado por [ULFGService](#).

## 6.5.3 Diagrama E-R

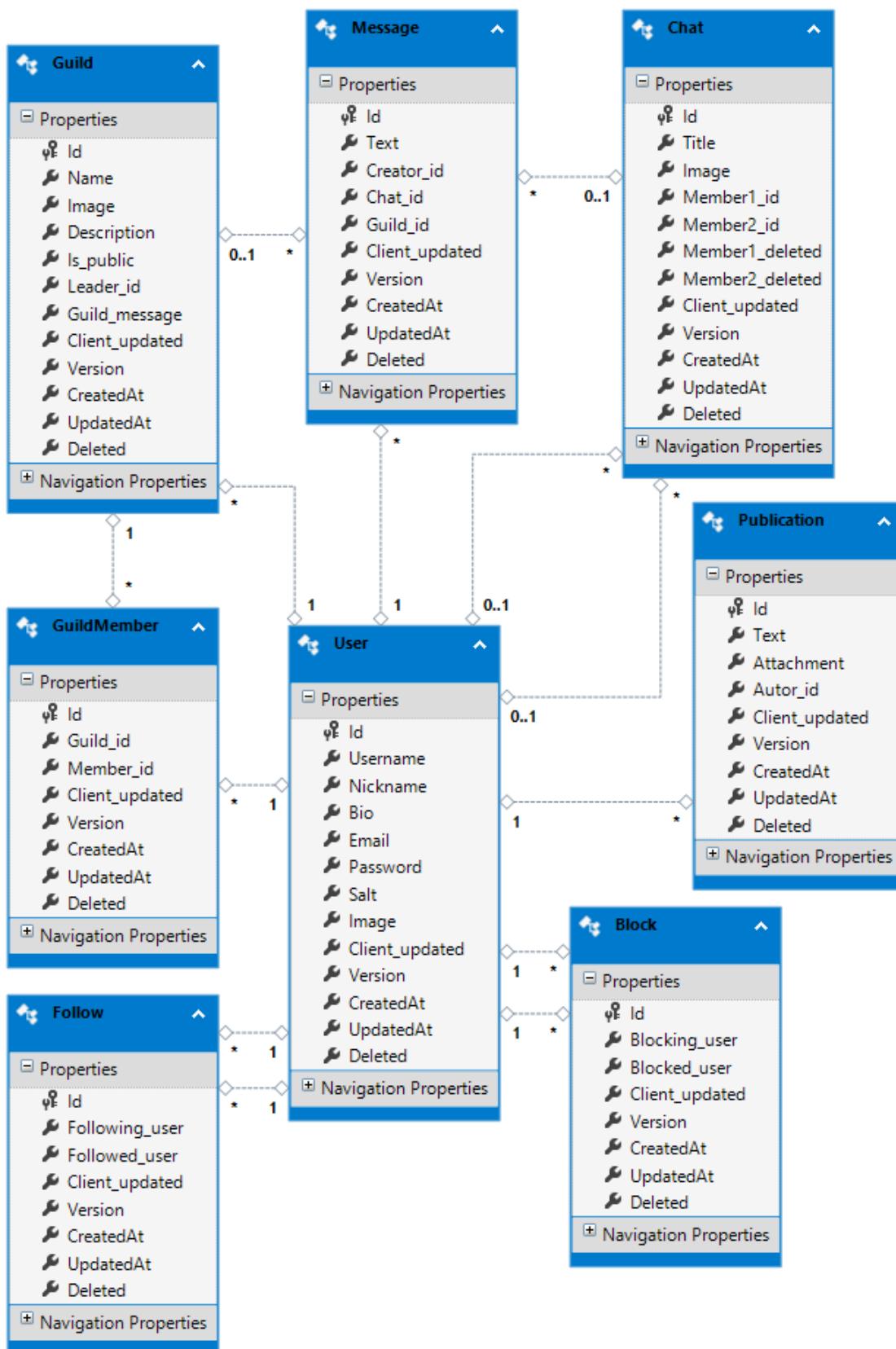


Figura 6.36 Diagrama Entidad-Relación

## 6.6 Diseño de la Interfaz

Esta sección incluye capturas de los diseños finales de las pantallas de la aplicación que siguen la estructura vista en el Análisis. Para simplificar se mostrarán solo las capturas de la aplicación para Android, ya que ambas comparten la misma interfaz y las mismas pantallas.

### 6.6.1 Login y Registro

Desde el Login se puede acceder al Registro (Figura 6.38 Pantalla de Registro ) y al Menú Principal (Figura 6.39 Pantalla del Menú principal).



Figura 6.37 Pantalla de Login



Figura 6.38 Pantalla de Registro

### 6.6.2 Menú Principal

Desde este menú se puede acceder a todos los bloques de la aplicación. Está disponible en todas las páginas realizando un movimiento con el dedo de izquierda a derecha y en la pantalla principal tocando el ícono de las tres barras (hamburguesa).

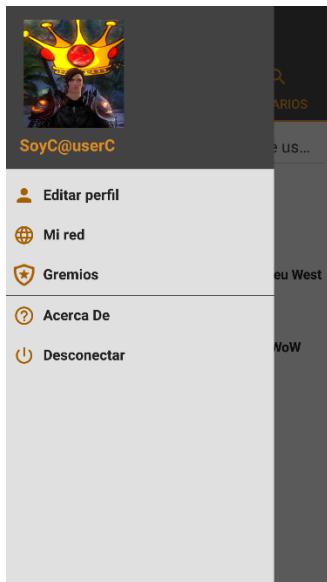


Figura 6.39 Pantalla del Menú principal

### 6.6.3 Pantalla Principal



Figura 6.40 Pantalla de la Lista de publicaciones



Figura 6.41 Pantalla de Detalle de adjunto

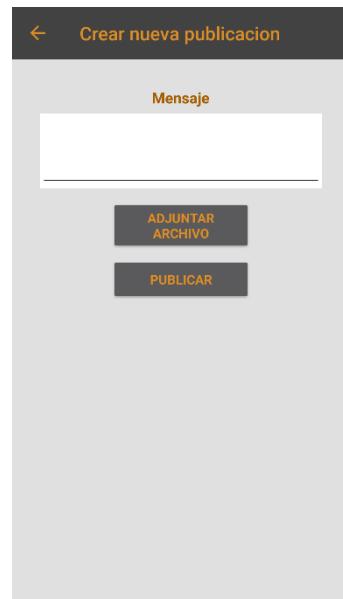


Figura 6.42 Pantalla de Crear publicación

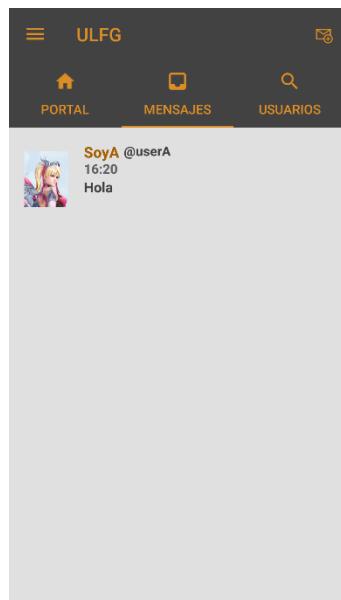


Figura 6.43 Pantalla de Ver Mensajes privados



Figura 6.44 Pantalla de Chat privado

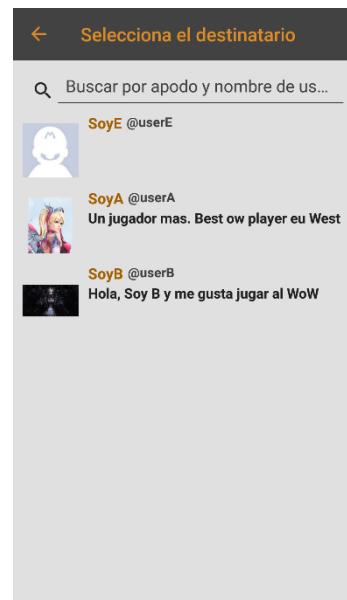


Figura 6.45 Pantalla de Elegir destinatario

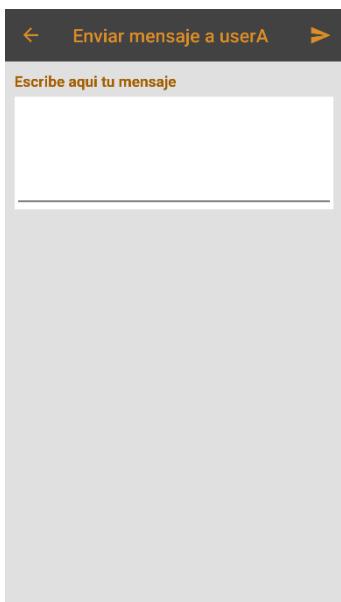


Figura 6.46 Pantalla de Crear mensaje

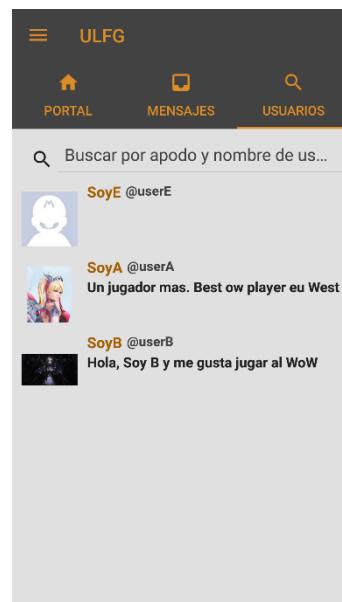


Figura 6.47 Pantalla de Búsqueda de usuarios



Figura 6.48 Pantalla de Perfil de Usuario

La página que contiene las pestañas es la conocida “Página Principal” y permite cambiar entre las páginas de Lista de Publicaciones (Figura 6.40 Pantalla de la Lista de publicaciones), Lista de Mensajes (Figura 6.43 Pantalla de ) y Búsqueda de Usuarios (Figura 6.47 Pantalla de ). Cada página dispone de diversas acciones en el Menú Secundario superior que permiten acceder a otras páginas.

**Lista de publicaciones** (Figura 6.40 Pantalla de la Lista de publicaciones):

- Al realizar una pulsación larga en una publicación propia sale una opción para borrarla.
- Al tocar la imagen adjunta de una publicación se abre un detalle de esta (Figura 6.41 Pantalla de Detalle de )

- Los botones del menú secundario permiten refrescar la lista y acceder a la página de crear publicaciones (Figura 6.42 Pantalla de Crear publicación)

**Lista de Chats (Figura 6.43 Pantalla de ):**

- Al tocar un chat de la lista se abre un chat con todos los mensajes enviados y recibidos de ese usuario (Figura 6.44 Pantalla de Chat )
- Desde el menú secundario se puede acceder a la pantalla de Elegir destinatario (Figura 6.45 Pantalla de Elegir destinatario), y desde ésta a Enviar mensaje (Figura 6.46 Pantalla de Crear mensaje) tocando un usuario.

**Lista de Usuarios (Figura 6.47 Pantalla de ):**

- Se puede filtrar por nombre de usuario, apodo y descripción escribiendo un texto en la barra de búsqueda
- Al tocar un usuario de la lista se abre su perfil (Figura 6.48 Pantalla de Perfil de Usuario)

## 6.6.4 Pantalla de Gremios

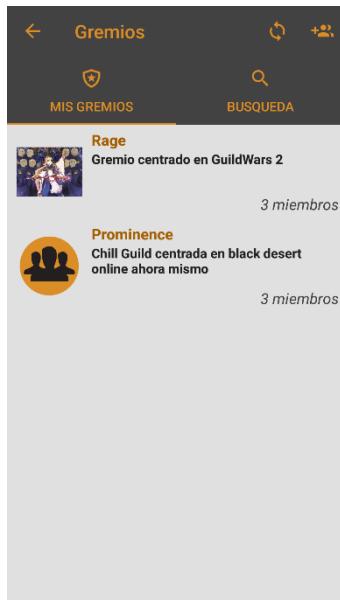


Figura 6.49 Pantalla de Gremios propios



Figura 6.50 Pantalla de Búsqueda de gremios

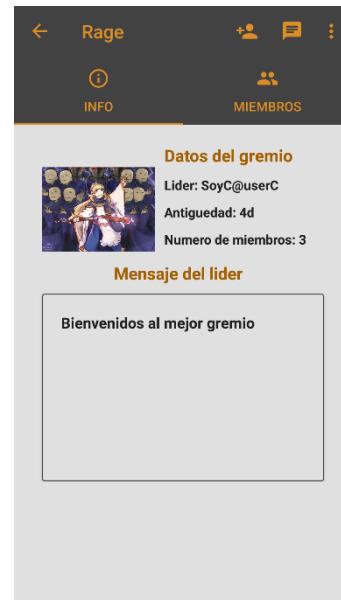


Figura 6.51 Pantalla de Detalle de gremio



Figura 6.52 Pantalla de Ver miembros

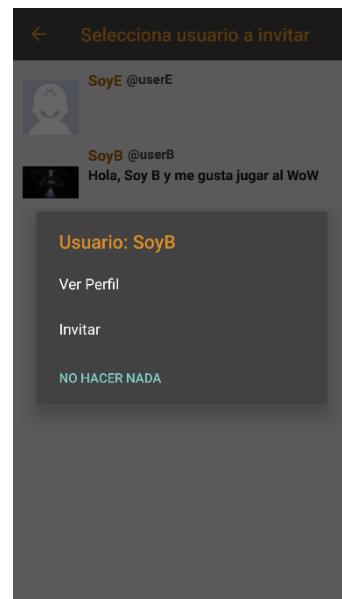


Figura 6.53 Pantalla de Invitar usuario



Figura 6.54 Pantalla de Editar gremio



Figura 6.55 Pantalla de Chat de gremio



Figura 6.56 Pantalla de Crear gremio

## Página principal de gremios

Usando el menú Principal accedemos a (Figura 6.49 Pantalla ) que muestra los gremios a los que pertenece el usuario. En la otra pestaña tenemos Figura 6.50 Pantalla de Búsqueda de gremio que permite buscar un gremio por su nombre o por algún texto en su descripción. Desde los botones del menú superior de (Figura 6.49 Pantalla ) se puede refrescar la lista y acceder a (Figura 6.56 Pantalla de Crear gremio).

Tocando un gremio del que se es miembro en cualquiera de las dos pestañas permite acceder a (Figura 6.51 Pantalla de Detalle de gremio).

## Detalle de gremio

La primera pestaña es puramente informativa (Figura 6.51 Pantalla de Detalle de gremio) y la pestaña de miembros (Figura 6.52 Pantalla de ) permite al líder del gremio expulsar tocando un usuario de la lista.

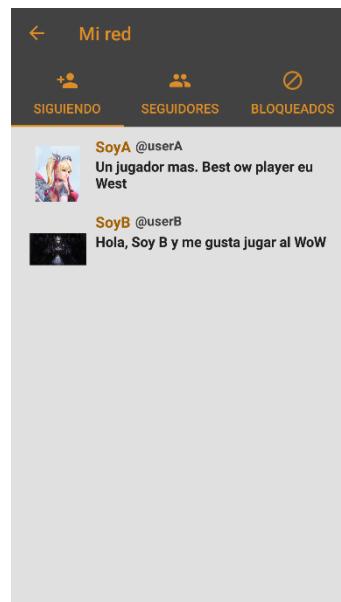
Las acciones del menú superior permiten:

- Invitar a un nuevo miembro, accediendo a (Figura 6.53 Pantalla de Invitar usuario)
- Enviar un mensaje al chat de gremio, accediendo a (Figura 6.55 Pantalla de Chat de gremio)
- Abandonar gremio. Si el usuario es el líder esta opción será para Deshacer el gremio
- Si el usuario actual es el líder tendrá una opción adicional para editar los datos del gremio, que accede a (Figura 6.54 Pantalla de Editar gremio)

## 6.6.5 Otras pantallas



**Figura 6.57 Pantalla Editar Perfil**



**Figura 6.58 Pantalla de Mi red**



**Figura 6.59 Pantalla de Acerca De**

La Figura 6.57 Pantalla Editar Perfil permite editar los datos del perfil del usuario y funciona exactamente igual que la de Editar gremio (Figura 6.54 Pantalla de Editar gremio).

La Figura 6.58 Pantalla de Mi red muestra la pantalla de Mi red con la lista seguimientos. Las otras dos páginas (seguidores y bloqueados) son idénticas y por ello se omiten.

La Figura 6.59 Pantalla de Acerca De muestra información acerca de la aplicación.

A todas estas pantallas se accede directamente a través del Menú Principal (Figura 6.39 Pantalla del Menú principal).

## 6.7 Especificación Técnica del Plan de Pruebas

En este apartado se detallan los tipos y niveles de pruebas descritos en el Análisis indicando sobre qué se aplicarán y como.

### 6.7.1 Pruebas Unitarias

#### 6.7.1.1 Manuales

##### 6.7.1.1.1 Pruebas del Servicio Web

Estas pruebas se irán ejecutando de forma manual durante el desarrollo para realizar peticiones HTTP que permitan comprobar el correcto funcionamiento del servicio web usando la herramienta Postman.

### 6.7.1.1.2 Pruebas de funcionalidad de la interfaz

Se realizarán algunas pruebas manuales en algunas pantallas de la interfaz que poseen una cierta complejidad. Las demás pantallas no gozan de ninguna funcionalidad destacable y no se incluirán en las pruebas unitarias.

- Pantalla Editar perfil
  - Campo editar apodo
  - Campo editar biografía
  - Campo editar email
- Pantalla Editar gremio
  - Campo editar nombre
  - Campo editar mensaje fijado
  - Campo editar descripción
  - Campo visibilidad
- Pantalla Perfil usuario
  - Botón Seguir / Dejar de seguir
  - Botón Bloquear/ Desbloquear

### 6.7.1.2 Automatizadas

En esta sección se detallan las pruebas unitarias automatizadas que se realizarán en el sistema. Se empleará la herramienta MSTest, un framework de automatización de pruebas unitarias integrado con Visual Studio para crear pruebas multihilo. Estas pruebas se ejecutarán automáticamente cada vez que se envíen cambios a VSTS como parte de la integración continua. Sólo se han realizado pruebas de las clases que contienen una cierta complejidad lógica. Las clases con simples operaciones CRUD no se incluyen en las pruebas unitarias.

A continuación, se representan las situaciones a cubrir por las pruebas. Para facilitar la trazabilidad con la implementación de las pruebas en MSTest que se incluye en el anexo “[Descripción detallada de las clases](#)” se indica entre paréntesis el método con el que se corresponden:

- Operaciones relativas a los usuarios
  - Registro del usuario (UserOperations.RegisterUser)
    - No existe un usuario con los datos introducidos
    - Existe un usuario con los datos introducidos
  - Acceso del usuario (UserOperations.LoginUser)
    - Login correcto
    - Login incorrecto
    - No existe este usuario
- Operaciones relativas a las interacciones entre usuarios
  - Bloquear usuario (SocialOperations.BlockUser)
    - Bloqueo correcto
  - Seguimiento del usuario (SocialOperations.FollowUser)
    - Existe un bloqueo propio
    - Existe un bloqueo del otro usuario

- No existe bloqueo
- Dejar de seguir al usuario (SocialOperations.UnFollowUser)
  - Usuario seguido
  - Usuario no seguido
- Desbloqueo del usuario (SocialOperations.UnBlockUser)
  - Usuario bloqueado
  - Usuario no bloqueado
- Operaciones relativas a la mensajería
  - Enviar mensaje (MessageOperations.SendMessage)
    - Existe ya un chat entre usuarios
    - No existe un chat entre usuarios
- Operaciones relativas a los gremios
  - Crear gremio (GuildOperations.CreateGuild)
    - Creación correcta
  - Abandonar gremio (GuildOperations.LeaveGuild)
    - Usuario no es miembro del gremio
    - Usuario es miembro del gremio

## 6.7.2 Pruebas de Sistema

Estas pruebas están basadas en los diagramas de interacción vistos en 6.3 y se han realizado utilizando la técnica de caminos y transiciones. A continuación, se incluyen los diagramas de interacción con sus caminos identificados y posteriormente el diseño de las pruebas a realizar.

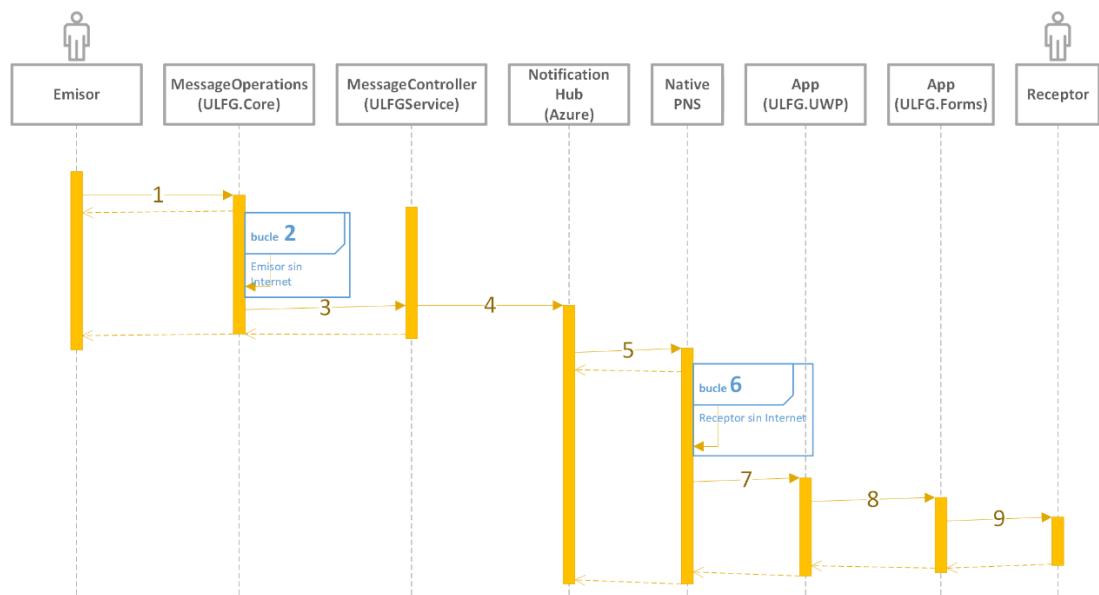
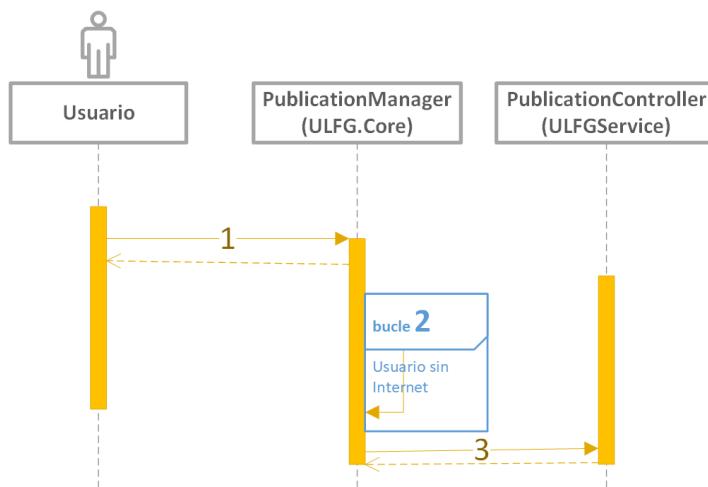


Figura 6.60 Diagrama de interacción de Envío de Notificación con caminos



**Figura 6.61 Diagrama de interacción de Sincronización Offline con caminos**

- Envío de Notificación
  - 1. Emisor sin conexión a internet (1,2)
  - 2. Emisor con conexión a internet y receptor sin ella (1,3,4,5,6)
  - 3. Emisor y receptor con conexión a internet (1,3,4,5,7,8,9)
- Sincronización Offline
  - 4. Usuario con conexión a internet (1,2)
  - 5. Usuario sin conexión a internet (1,3)

## 6.7.3 Pruebas de Usabilidad

Las pruebas de usabilidad tratan de evaluar la aplicación en su funcionamiento real y medir como de difícil o sencillo es su uso por parte de los usuarios finales.

Los objetivos son de estas pruebas son:

- Mejorar la calidad de la interacción de los usuarios con nuestra aplicación.
- Reducir el tiempo necesario para hacer las distintas tareas en la aplicación y de esta forma aumentar la productividad.

Se realizarán pruebas en disantos tipos de usuarios y se valorará su satisfacción con el producto mediante el uso de los cuestionarios que serán descritos a continuación.

### 6.7.3.1 Actividades de las Pruebas de Usabilidad

#### 6.7.3.1.1 Preguntas de carácter general

A continuación, se incluye el cuestionario con las preguntas de carácter general que se usarán para determinar el tipo de usuario. Algunas admiten más de una respuesta.

¿Usa un ordenador frecuentemente?
1. Todos los días 2. Varias veces a la semana

3. Ocasionalmente
4. Nunca o casi nunca

#### ¿Qué tipo de actividades realiza con el ordenador?

1. Es parte de mi trabajo o profesión
2. Para ocio y entretenimiento
3. Solo empleo aplicaciones estilo Office
4. Únicamente leo el correo y navego ocasionalmente

#### ¿Con que frecuencia utiliza redes sociales?

1. Todos los días
2. Varias veces a la semana
3. Ocasionalmente
4. Nunca o casi nunca

#### ¿Qué tipo de uso le suele dar a su dispositivo móvil además de para llamar y enviar mensajes? (Respuesta múltiple)

1. Sólo para llamar y enviar mensajes
2. Hago uso de aplicaciones varias, como redes sociales y mensajería
3. Ocio y entretenimiento
4. No tengo móvil

#### ¿Sabe lo que son los eSports?

1. Si, he visto o ido a algún torneo
2. Si, pero nunca he visto ningún torneo
3. No, aunque he escuchado algo sobre ellos
4. No, nunca he oido hablar de ellos

#### ¿En qué dispositivos suele jugar a videojuegos? (Respuesta múltiple)

1. Ordenador
2. Móvil
3. Consolas
4. No juego a videojuegos

### 6.7.3.1.2 Actividades guiadas

A continuación, se describen las diferentes actividades que realizarán los usuarios con la aplicación durante las pruebas:

- Registro en la aplicación
- Inicio de sesión
- Crear Publicación
- Editar datos de perfil
- Seguir usuario
- Bloquear usuario
- Enviar mensaje a un usuario

- Unirse a un gremio
- Crear un gremio
- Abandonar un gremio
- Editar datos de un gremio
- Enviar un mensaje por chat de gremio
- Salir de la aplicación

### 6.7.3.1.3 Preguntas Cortas sobre la Aplicación y Observaciones

A continuación, se muestra cuestionario de preguntas cortas que realizarán los usuarios cuando prueben la aplicación.

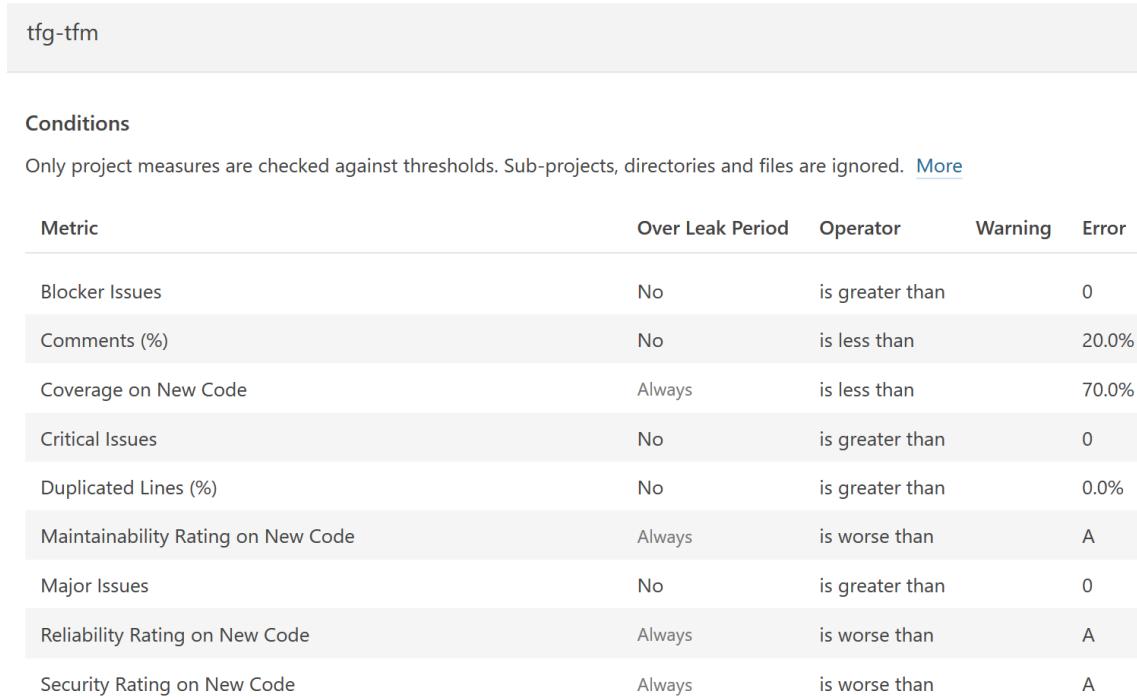
Facilidad de Uso	Siempre	Frecuentemente	Ocasionalmente	Nunca
¿Sabe dónde está dentro de la aplicación?				
¿Sabe localizar cada funcionalidad dentro de la aplicación?				
¿Le resulta sencillo el uso de la aplicación?				
Funcionalidad	Siempre	Frecuentemente	Ocasionalmente	Nunca
¿Funciona cada tarea como usted espera?				
¿El tiempo de respuesta de la aplicación es muy grande?				
Calidad del Interfaz				
Aspectos gráficos	Muy Adecuado	Adecuado	Poco Adecuado	Nada Adecuado
El tipo y tamaño de letra es				
Los iconos e imágenes usados son				
Los colores empleados son				
Diseño de la Interfaz	Si	No	A veces	
¿Le resulta fácil de usar?				
¿El diseño de las pantallas es claro y atractivo?				
¿Cree que el programa está bien estructurado?				
Observaciones				
Cualquier comentario del usuario				

### 6.7.3.1.4 Cuestionario para el responsable de las pruebas

Aspecto Observado	Notas
El usuario se adapta rápidamente a la aplicación y realiza las tareas de forma eficiente	
Tiempo en realizar cada tarea	
Errores leves cometidos	
Errores graves cometidos	
Satisfacción con la interfaz de usuario	

## 6.7.4 Pruebas estáticas de Código

A continuación, se describen las características de la QualityGate establecida en SonarQube que se emplea para realizar las pruebas estáticas de código.



The screenshot shows the 'Conditions' section of a QualityGate configuration in SonarQube. It lists various metrics and their thresholds:

Metric	Over Leak Period	Operator	Warning	Error
Blocker Issues	No	is greater than	0	
Comments (%)	No	is less than	20.0%	
Coverage on New Code	Always	is less than	70.0%	
Critical Issues	No	is greater than	0	
Duplicated Lines (%)	No	is greater than	0.0%	
Maintainability Rating on New Code	Always	is worse than	A	
Major Issues	No	is greater than	0	
Reliability Rating on New Code	Always	is worse than	A	
Security Rating on New Code	Always	is worse than	A	

*Figura 6.62 QualityGate de SonarQube para las pruebas de Código*

# Capítulo 7. Implementación del Sistema

## 7.1 Estándares, Normas y Patrones Seguidos

En este apartado se describen los estándares, normas y patrones empleados en la construcción del sistema

### 7.1.1 Patrón MVVM

La finalidad principal del patrón MVVM (Modelo Vista-Vista-Modelo) es tratar de desacoplar lo máximo posible la interfaz de usuario de la lógica de la aplicación. Tiene tres componentes diferenciados:

- **View (Vista):** La vista representa la interfaz de usuario con sus componentes visuales. Las vistas en MVVM son activas, contienen comportamientos, eventos y enlaces a datos que necesitan tener conocimiento del modelo subyacente.
- **Model (Modelo):** Representa la capa de datos y/o la lógica de negocio.

- **ViewModel (Modelo de la Vista):** El ViewModel (modelo de vista) es un actor intermediario entre el modelo y la vista, contiene toda la lógica de presentación y se comporta como una abstracción de la interfaz. La comunicación entre la vista y el viewmodel se realiza por medio los enlaces de datos (bindings).



*Figura 7.1 Patrón MVVM*

La principal diferencia con el conocido patrón MVC es que las vistas se actualizan automáticamente cuando hay cambios en modelo, mientras que en MVC deben actualizarse manualmente. Esto se consigue mediante “Bindings” o enlaces de datos entre el modelo y la vista gestionados por el ViewModel.

## 7.1.2 UML

Es un lenguaje gráfico de modelado para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un sistema software incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Es el lenguaje de modelado más conocido y usado en la industria del Software.

Fue publicado por primera vez en 1997 en su versión 1.0 y en 2005 recibió una actualización mayor y pasó a la versión 2.0. Actualmente se emplea la versión 2.5 liberada en 2015.

Los tipos de diagrama existentes en UML 2.5 pueden agruparse en tres bloques:

### Estructurales

Muestran la estructura estática de los objetos del sistema

- De clases
- De componentes
- De despliegue
- De Objetos
- De paquetes
- De perfiles
- De estructura compuesta

### De comportamiento

Muestran el comportamiento dinámico de los objetos en el sistema.

- De actividades
- De casos de uso
- De máquina de estados

#### De interacción

Muestran las relaciones entre los objetos del sistema

- Global de interacciones
- De comunicación
- De secuencia
- De tiempos

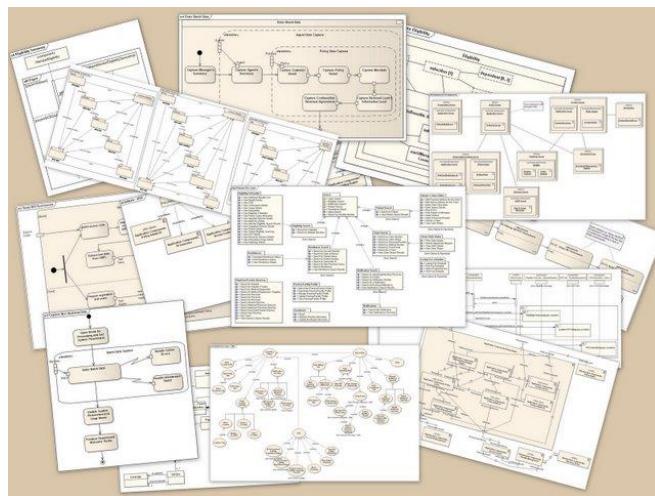


Figura 7.2 Collage de diferentes tipos de diagrama UML<sup>1</sup>

Puede encontrarse una descripción de todos los tipos diagramas en el siguiente [https://es.wikipedia.org/wiki/Lenguaje unificado de modelado#Tipos de diagramas en UML 2.5](https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado#Tipos_de_diagramas_en_UML_2.5).

## 7.2 Lenguajes de Programación

Descripción de los lenguajes de programación usados, su versión y distribución, los módulos o complementos de los mismos empleados, etc.

---

<sup>1</sup> Imagen creada por [Kishorekumar 62](#) y distribuida mediante una licencia [CC BY-SA 3.0](#). La imagen original puede encontrarse [aquí](#)

## 7.2.1 C#

### 7.2.1.1 Descripción

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.

Es multiparadigma, estructurado, imperativo, orientado a objetos, dirigido por eventos, funcional, genérico y reflexivo

La versión de C# empleada es la 7.3 liberada el 7 de mayo de 2018.

### 7.2.1.2 Módulos / Dependencias

A continuación, se describen todos los módulos empleados para el sistema basados en el lenguaje de programación C#. Todos ellos se obtienen empleando la herramienta NuGet, un gestor de paquetes para .NET similar a Maven en Java.

#### 7.2.1.2.1 NSubstitute

Plugin para crear un Mockup de una clase determinada para realizar pruebas. Permite especificar los retornos de los métodos de la clase simulada para un cada conjunto de parámetros y escuchar si se ha llamado a un determinado método, entre otras cosas. La versión utilizada es 3.1.0.

Se emplea para simular la Base de Datos durante las pruebas unitarias

Se puede descargar de NuGet y su página principal es <http://nsubstitute.github.io/>

#### 7.2.1.2.2 Newtonsoft Json

Plugin para facilitar la creación y gestión de los Json. Se emplea para enviar información con las peticiones HTTP. La versión utilizada es 11.0.2.

Se puede descargar de NuGet y su página principal es <https://www.newtonsoft.com/json>

#### 7.2.1.2.3 UserDialogs

Plugin para crear diálogos de forma unificada para todas las plataformas. Se emplea en Xamarin.Forms para crear diálogos que funcionen tanto en Android como en UWP. La versión utilizada es 7.0.1.

Se puede descargar de NuGet y su código fuente puede encontrarse en  
<https://github.com/aritchie/userdialogs>

#### **7.2.1.2.4 FFImageLoading**

Plugin para optimizar la carga de las imágenes. Se emplea en todas las imágenes de la aplicación. La versión utilizada es 2.3.6.

Se puede descargar de NuGet y su código fuente puede encontrarse en  
<https://github.com/luberda-molinet/FFImageLoading>

#### **7.2.1.2.5 Media Plugin**

Plugin para acceder y seleccionar imágenes y videos de la galería nativa de cada plataforma. Permite además sacar fotos y grabar videos en el momento. Se emplea para modificar las imágenes de los perfiles de usuario y los gremios y para los adjuntos de las publicaciones. La versión utilizada es 3.1.3.

Se puede descargar de NuGet y su código fuente puede encontrarse en  
<https://github.com/jamesmontemagno/MediaPlugin>

#### **7.2.1.2.6 Settings Plugin**

Plugin para almacenar preferencias de forma nativa en cada plataforma de forma unificada. Se emplea para almacenar la sesión del usuario en el dispositivo. La versión utilizada es 3.1.1.

imágenes de los perfiles de usuario y los gremios y para los adjuntos de las publicaciones.

Se puede descargar de NuGet y su código fuente puede encontrarse en  
<https://github.com/jamesmontemagno/SettingsPlugin>

#### **7.2.1.2.7 Connectivity Plugin**

Plugin para comprobar el estado de la red en cada plataforma de forma unificada. Se emplea para todas las comprobaciones de red. La versión utilizada es 3.1.1.

Se puede descargar de NuGet y su código fuente puede encontrarse en  
<https://github.com/jamesmontemagno/ConnectivityPlugin>

## **7.2.2 SQL**

SQL es un lenguaje estructurado específico del dominio para definir una base de datos y manejar los datos guardados en ella. La última versión data de 2012 y será la usada en el sistema para el manejo de la base de datos.

Tiene las siguientes características:

- **Lenguaje de definición de datos:** El LDD de SQL proporciona comandos para la definición de esquemas de relación, borrado de relaciones y modificaciones de los esquemas de relación (CREATE, ALTER, TRUNCATE, DROP).
- **Lenguaje interactivo de manipulación de datos:** El LMD de SQL incluye lenguajes de consultas basado tanto en álgebra relacional como en cálculo relacional de tuplas.

Permite realizar consultas(SELECT), insertar(INSERT), actualizar(UPDATE) y borrar(DELETE) datos.

- **Integridad:** El LDD de SQL incluye comandos para especificar las restricciones de integridad que deben cumplir los datos almacenados en la base de datos.
- **Definición de vistas:** El LDD incluye comandos para definir las vistas.
- **Control de transacciones:** SQL tiene comandos para especificar el comienzo y el final de una transacción.
- **SQL incorporado y dinámico:** Esto quiere decir que se pueden incorporar instrucciones de SQL en lenguajes de programación como: C++, C, Java, PHP, Cobol, Pascal y Fortran.
- **Autorización:** El LDD incluye comandos para especificar los derechos de acceso a las relaciones y a las vistas.

## 7.3 Herramientas y Programas Usados para el Desarrollo

Este apartado incluye una descripción de todas las herramientas de software utilizadas para el desarrollo del sistema

### 7.3.1 Visual Studio Enterprise

Es un entorno de desarrollo integrado (IDE) para Android, iOS, Windows, la Web y la nube para tecnologías Microsoft. Además, también incluye soporte para Python, Node.js, R y Javascript.

Dispone de varias versiones:

- **Community:** Versión gratuita para desarrolladores individuales
- **Professional:** Versión de pago que proporciona herramientas para facilitar la colaboración de grupos pequeños.
- **Enterprise:** Versión de pago centrada en empresas grandes que incluye facilidades para escalar, controlar el rendimiento y mejorar la calidad de las aplicaciones.

Se ha decidido emplear la versión Enterprise porque se disponía de una licencia proporcionada por Microsoft Imagine para Estudiantes y porque dispone de más facilidades a la hora de realizar diagramas que la versión gratuita. La versión es VS empleada es 15.7.2.

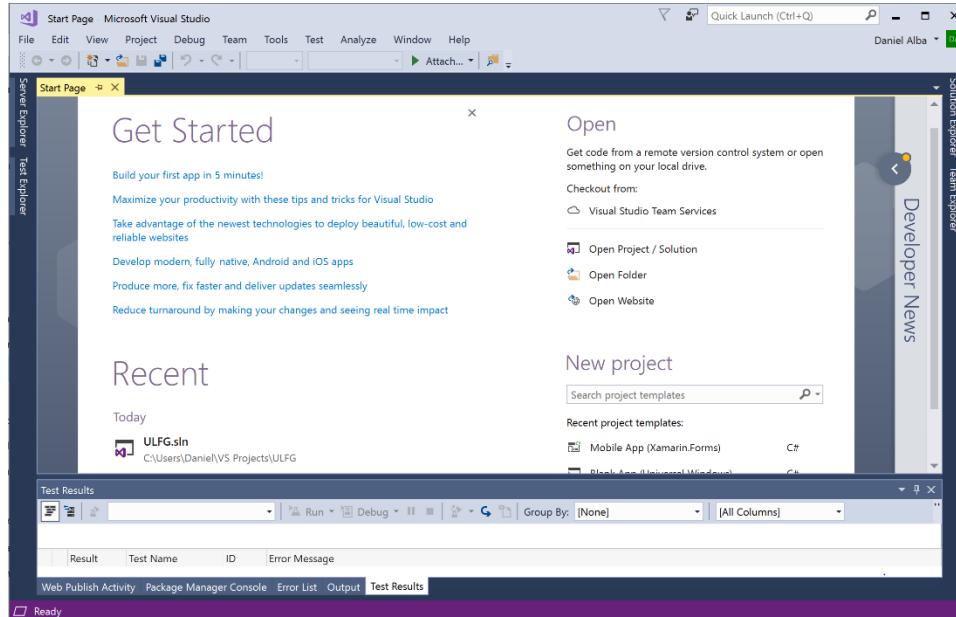
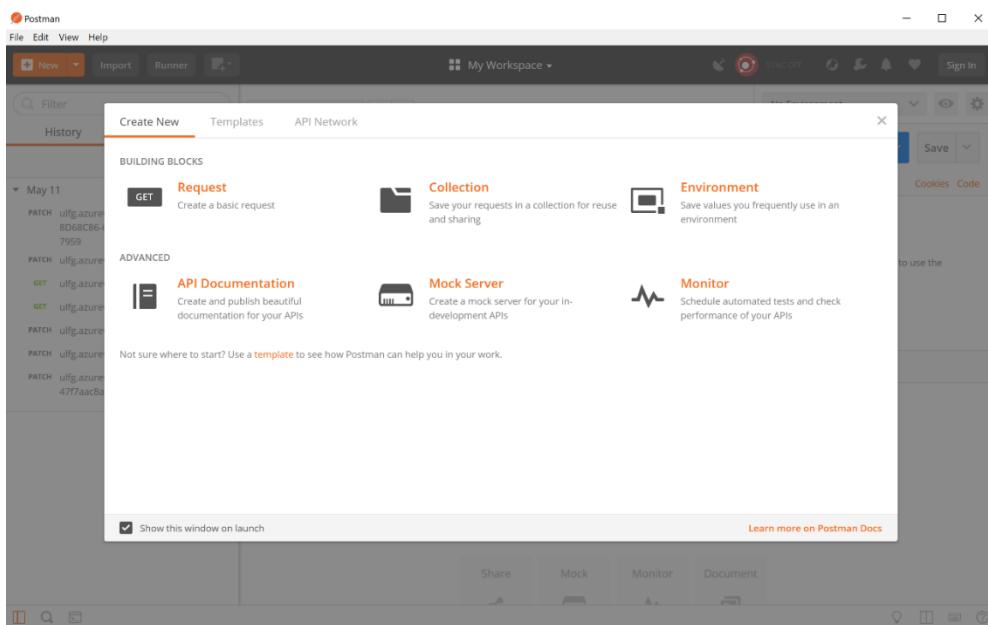


Figura 7.3 Visual Studio Enterprise 2017

### 7.3.2 Postman

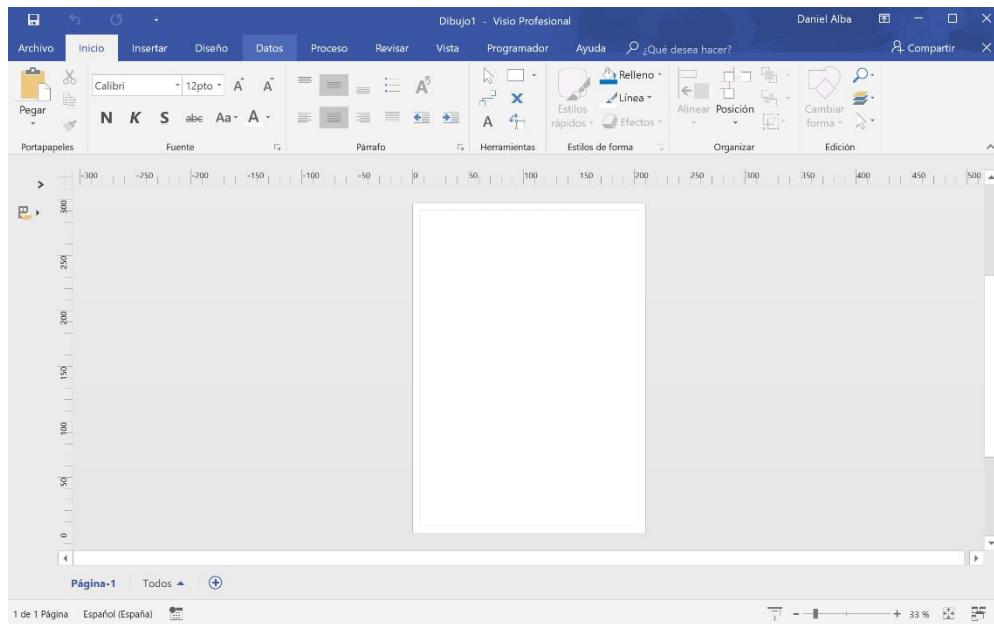
Es un entorno de desarrollo avanzado para APIs web (ADE) que comprende todo el ciclo de desarrollo. Sólo se ha utilizado para realizar pruebas sobre el servicio web. La versión empleada es 6.0.10.



**Figura 7.4 Postman**

### 7.3.3 Visio Profesional 2016

Es una herramienta para la creación de diagramas propietaria de Microsoft. Se utiliza para elaborar todos los diagramas que aparecen en la documentación a excepción de los de clases, que son generados por VS. La versión empleada es 16.0.9330.2124.



**Figura 7.5 Visio 2016**

### 7.3.4 Word 2016

Es un procesador de textos propietario de Microsoft. Se ha utilizado para la elaboración de esta documentación. La versión empleada es 16.0.9330.2124.

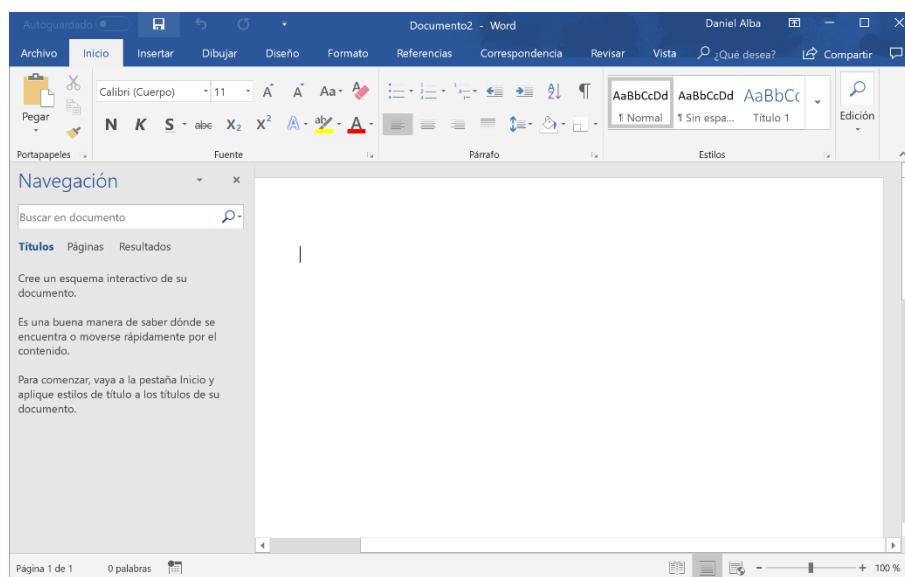


Figura 7.6 Word 2016

## 7.4 Creación del Sistema

En este apartado se detallan los problemas encontrados durante el desarrollo y las clases vistas anteriormente en el Diseño

### 7.4.1 Problemas Encontrados

A continuación, se enumeran los problemas encontrados en el desarrollo junto a la solución que se les ha dado:

- Diagramas BurntDown de VSTS:** Se ha tenido dificultades para entender cómo funcionan este tipo de diagramas y en qué se basa VSTS para elaborarlos. Tras hacer pruebas durante el primer Sprint se ha encontrado una forma de crear correctamente los diagramas creando tareas para cada ítem del backlog y estimando estas tareas. Esto era porque VSTS no organizaba las historias de forma plana sino con cierta jerarquía empleando las tareas asignadas a cada ítem del backlog.
- Estimación de las tareas:** Al ser todo tecnologías nuevas se ha tenido problemas al estimar cada funcionalidad. A lo largo de los Sprints esto ha ido mejorando por sí solo a base de la experiencia obtenida.
- Análisis de los proyectos .NET Standard en SonarQube:** No se detectaban los proyectos basados en .NET Standard en SonarQube. Para solucionar esto se ha editado cada uno de los proyectos y añadido un identificador único (UUID). Tras esto SonarQube detectó los proyectos.
- Problemas con los bloqueos de la interfaz de usuario:** Todas las operaciones que no involucren cambios en la vista deben ejecutarse en un hilo diferente por temas de arquitectura de las plataformas. Esto provoca muchas veces bloqueos en la interfaz cuyo

origen era difícil de encontrar. Se han definido tres formas de realizar dichas tareas en otro hilo y cuando ocurre un bloqueo se van probando todas hasta encontrar una que no bloquee. De esta forma se agiliza el desarrollo. Las tres formas son:

1. Async / await
  2. Task.Run / Wait
  3. Device.BeginInvokeOnMainThread
5. **Envío de notificaciones push y datos asociados:** Requería configurar gran cantidad de parámetros en cada sistema de notificaciones y crear plantillas para cada plataforma. Además, UWP no permitía enviar datos extra en las notificaciones push por lo que se ha tenido que realizar un esfuerzo adicional para gestionar las notificaciones en UWP.
6. **Sincronización de datos sin conexión:** Esto incluye la decisión de cuando sincronizar con la nube y como gestionar los conflictos:
  - Se ha decidido sincronizar siempre que haya conexión a internet ya que la BD es relativamente rápida y las consultas son sencillas. Hay algunas excepciones que sólo se sincronizan al crear la vista.
  - Para gestionar los conflictos se ha utilizado el siguiente algoritmo: Si la fecha de actualización del dato en el cliente es posterior a la fecha de actualización de ese dato en la base de datos, se fuerza la versión del cliente al servidor.
7. **Lentitud de la aplicación de Xamarin.Forms:** Las aplicaciones de Xamarin.Forms son más lentas que una nativa, en algunos tanto que resulta inviable. Su principal problema es la creación de nuevas páginas. Para ello se ha realizado una precarga de todas las páginas al abrir la aplicación mediante hilos concurrentes y se han almacenado en una estructura de datos de manera que cuando se necesita cambiar de página no se crea una nueva, sino que se reutiliza la que está almacenada. Se incluye lógica adicional para “simular” que se crea una página cada vez cuando en realidad se está reutilizando la misma. Esto ha acelerado la aplicación enormemente.

## 7.4.2 Descripción Detallada de las Clases

En esta sección se describen de forma detallada todas las clases vistas en el diagrama del Diseño junto a las que contienen las pruebas unitarias. Para generar esta documentación se han volcado los comentarios de código mediante la herramienta Docfx (<https://dotnet.github.io/docfx/>). Esta documentación se encuentra adjunta en formato HTML.

# Capítulo 8. Desarrollo de las Pruebas

## 8.1 Pruebas Unitarias automatizadas

En este apartado se adjunta una captura con los resultados de la última ejecución de las pruebas unitarias automatizadas en la integración continua de VSTS. Los nombres son descriptivos y trazables con el diseño de las pruebas realizado en [6.7.1. Pruebas Unitarias](#).

Outcome	Test Case Title	Priority	Duration
Passed	TestLoginNoExiste		0:00:00.003
Passed	TestSeguirConBloqueoExterno		0:00:00.003
Passed	TestAbandonarExpulsarDeGremioNoExiste		0:00:00.003
Passed	TestSeguirConBloqueoPropio		0:00:00.004
Passed	TestDejarDeSeguirNoExiste		0:00:00.003
Passed	TestLoginClaveIncorrecta		0:00:00.004
Passed	TestDesbloquearExiste		0:00:00.000
Passed	CrearChatExiste		0:00:00.000
Passed	TestCambiarPasswordCorrecta		0:00:00.003
Passed	TestCrearGremio		0:00:00.040
Passed	TestBloquear		0:00:00.033
Passed	TestCambiarPasswordIncorrecta		0:00:00.000
Passed	TestCambiarPasswordNoExisteUser		0:00:00.000
Passed	TestLoginCorrecto		0:00:00.717
Passed	TestRegistroDeUsuarioExistente		0:00:00.000
Passed	TestDejarDeSeguirExiste		0:00:00.004
Passed	CrearChatNoExiste		0:00:00.040
Passed	TestDesbloquearNoExiste		0:00:00.003
Passed	TestSeguirSinBloqueo		0:00:00.000
Passed	TestAbandonarExpulsarDeGremioExiste		0:00:00.004
Passed	TestRegistroDeUsuarioCorrecto		0:00:00.467

*Figura 8.1 Ejecución de las pruebas unitarias*

## 8.2 Pruebas de Sistema

En este apartado se describen las pruebas manuales del sistema diseñadas anteriormente junto a su número y como se procede a su realización.

N.º Prueba	Desarrollo	Resultado
1	Se abren dos emuladores Android 8.1 Se entra en sesión como “userA” en uno y como “userB” en otro Se activa el modo avión en el de “userA” Se envía un mensaje a “userB”	“userB” no visualiza ninguna notificación
2	Se abren dos emuladores Android 8.0 Se entra en sesión como “userA” en uno y como “userB” en otro Se activa el modo avión en el de “userB” Se envía un mensaje a userB Se espera 2 minutos Se quita el modo avión de “userB”	“userB” recibe la notificación tras quitar el modo avión
3	Se abren dos emuladores Android 8.1 Se entra en sesión como “userA” en uno y como “userB” en otro Se envía un mensaje a userB	“userB” recibe la notificación al momento del envío del mensaje
4	Se abren dos emuladores Android 8.1 Se entra en sesión como “userA” en uno y como “userB” en otro Crear una publicación con “userB” Crear un seguimiento de “userA” a “userB”	“userA” visualiza nuevas publicaciones al acceder al portal tras seguir
5	Se abren dos emuladores Android 8.1 Se entra en sesión como “userA” en uno y como “userB” en otro Se verifica que no existe seguimiento de “userA” a “userB” Se crea una publicación con “userB” Se pone modo avión a “userA” Se crea un seguimiento de “userA” a “userB”	“userA” no visualiza nuevas publicaciones al acceder al portal tras seguir

## 8.3 Pruebas de Usabilidad

A partir de los cuestionarios que se diseñaron anteriormente y de los procedimientos explicados, mostramos aquí el resultado de aplicarlos sobre todos los usuarios que hayamos empleado para estas pruebas. Se ha distribuido la aplicación de Android a los usuarios porque no requiere ningún tipo de configuración adicional y es más sencilla de instalar.

### 8.3.1 Cuestionarios de los usuarios

¿Usa un ordenador frecuentemente?
5. Todos los días U2 U4

<p>6. Varias veces a la semana U1      7. Ocasionalmente U3      8. Nunca o casi nunca</p>	
<b>¿Qué tipo de actividades realiza con el ordenador?</b>	
<p>5. Es parte de mi trabajo o profesión U4      6. Para ocio y entretenimiento U2      7. Solo empleo aplicaciones estilo Office      8. Únicamente leo el correo y navego ocasionalmente U1 U3</p>	
<b>¿Con que frecuencia utiliza redes sociales?</b>	
<p>5. Todos los días U1 U4      6. Varias veces a la semana U2      7. Ocasionalmente U3      8. Nunca o casi nunca</p>	
<b>¿Cuál es el uso que más suele darle a su dispositivo móvil?</b>	
<p>5. Llamar y enviar SMS U3      6. Uso de aplicaciones varias, como redes sociales y mensajería U1 U4      7. Ocio y entretenimiento U2      8. No tengo móvil</p>	
<b>¿Sabe lo que son los eSports?</b>	
<p>5. Si, he visto o ido a algún torneo U2      6. Si, pero nunca he visto ningún torneo      7. No, aunque he escuchado algo sobre ellos U1 U3 U4      8. No, nunca he oido hablar de ellos</p>	
<b>¿En qué dispositivo suele jugar principalmente a videojuegos?</b>	
<p>5. Ordenador U2      6. Móvil      7. Consolas U4      8. No juego a videojuegos U1 U3</p>	

Facilidad de Uso	Siempre	Frecuentemente	Ocasionalmente	Nunca
¿Sabe dónde está dentro de la aplicación?	U1 U4	U2	U3	
¿Sabe localizar cada funcionalidad dentro de la aplicación?	U1 U4	U2 U3		
¿Le resulta sencillo el uso de la aplicación?	U1 U4	U2 U3		
Funcionalidad	Siempre	Frecuentemente	Ocasionalmente	Nunca
¿Funciona cada tarea como usted espera?	U1 U2 U4	U3		
¿El tiempo de respuesta de la aplicación es muy grande?			U1 U2 U3 U4	

Calidad del Interfaz				
Aspectos gráficos	Muy Adecuado	Adecuado	Poco Adecuado	Nada Adecuado
<i>El tipo y tamaño de letra es</i>	U1 U2 U4	U3		
<i>Los iconos e imágenes usados son</i>		U1 U2 U3 U4		
<i>Los colores empleados son</i>		U1 U2 U3 U4		
Diseño de la Interfaz	Si	No	A veces	
<i>¿Le resulta fácil de usar?</i>	U1 U2 U4			U3
<i>¿El diseño de las pantallas es claro y atractivo?</i>	U1 U2 U3 U4			
<i>¿Cree que el programa está bien estructurado?</i>	U1 U2 U3 U4			
Observaciones				
Ningún usuario ha comentado nada				

### 8.3.2 Cuestionario del responsable de las pruebas

Aspecto Observado	Notas
<i>El usuario se adapta rápidamente a la aplicación y realiza las tareas de forma eficiente</i>	Todos los usuarios se adaptan en un tiempo asequible
<i>Tiempo en realizar cada tarea</i>	Normalmente es rápido, puede verse afectado por algunas lentitudes de la interfaz
<i>Errores leves cometidos</i>	Ninguno
<i>Errores graves cometidos</i>	Ninguno
<i>Satisfacción con la interfaz de usuario</i>	En general todos los usuarios han reaccionado positivamente al diseño de la interfaz

### 8.4 Pruebas estáticas de Código

En este apartado se adjunta una captura del resultado del análisis de SonarQube. La cobertura aparece como 0% debido a que las pruebas unitarias están hechas de forma concurrente y SonarQube no las detecta debido a que es una herramienta de terceros y su integración con VSTS no es perfecta. Otro aspecto que destacar es el 0% de duplicación de código.



Figura 8.2 Resumen de las pruebas en SonarQube

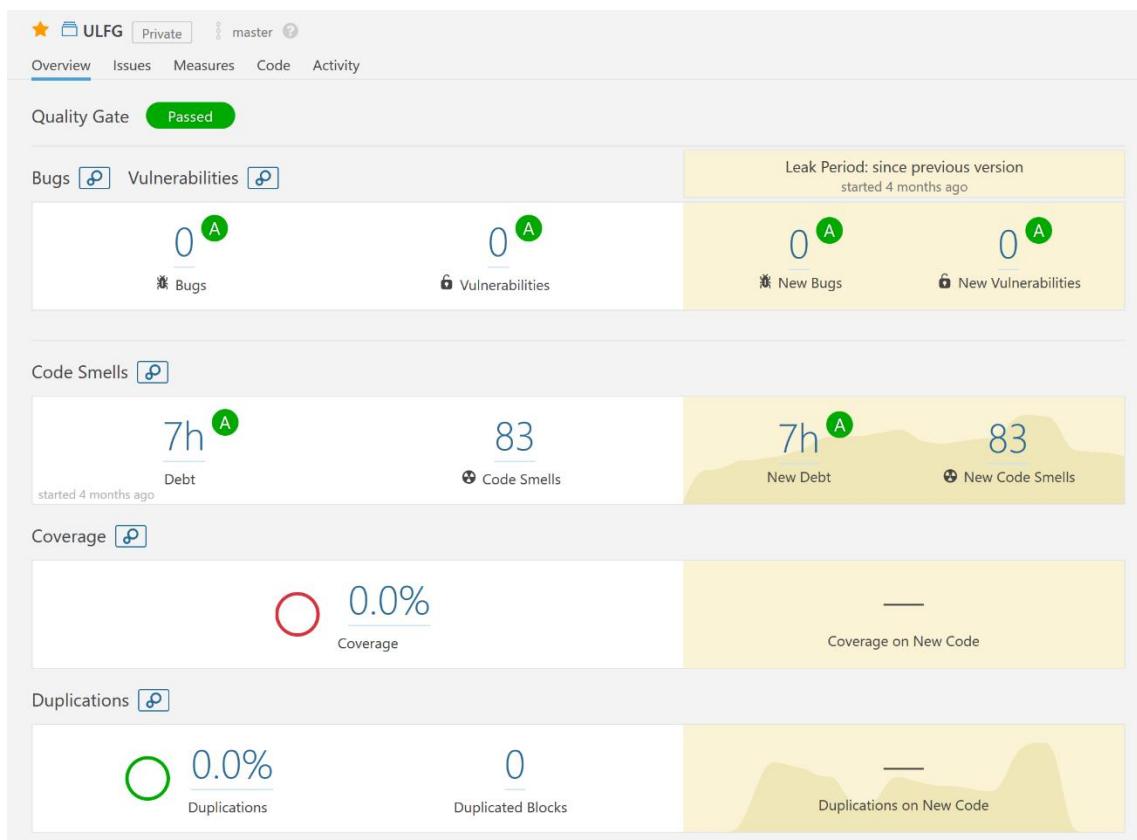


Figura 8.3 Resultado final de las pruebas en SonarQube

A continuación, se incluyen dos diagramas que muestran la evolución de la calidad del código a lo largo del desarrollo midiendo las Issues y las líneas de código duplicadas. En ellos se puede apreciar que se ha mantenido un nivel aceptable durante todo el desarrollo y se han solucionado casi todos los problemas antes de la Release, quedando sólo algunos Code Smells sin importancia catalogados como “Minor”.

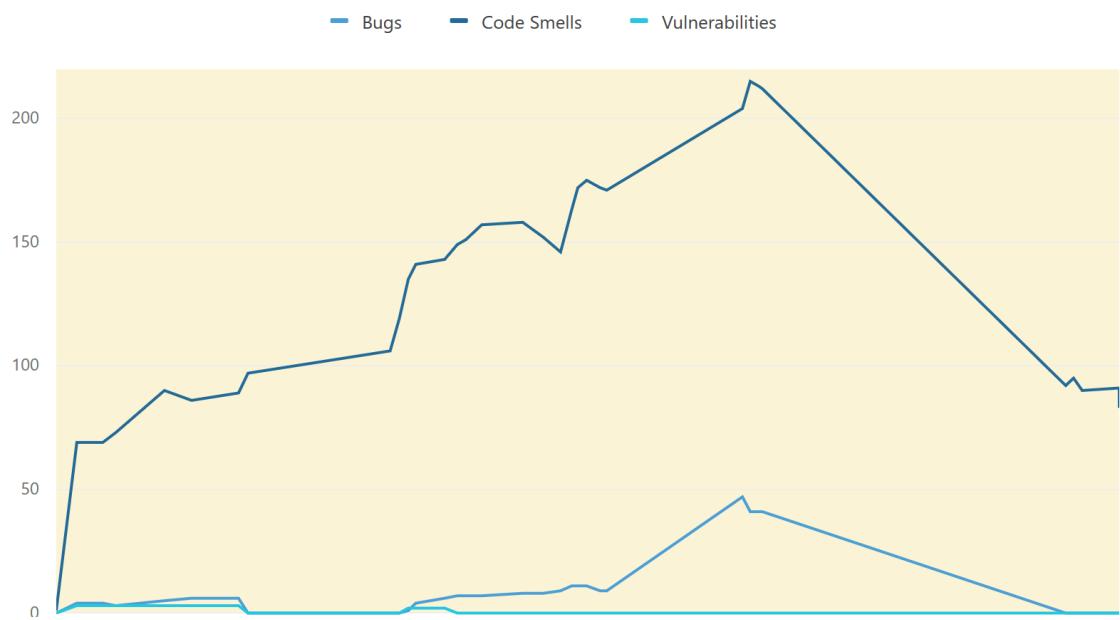


Figura 8.4 Evolución del número de Issues a lo largo del desarrollo

— Lines of Code    — Duplicated Lines

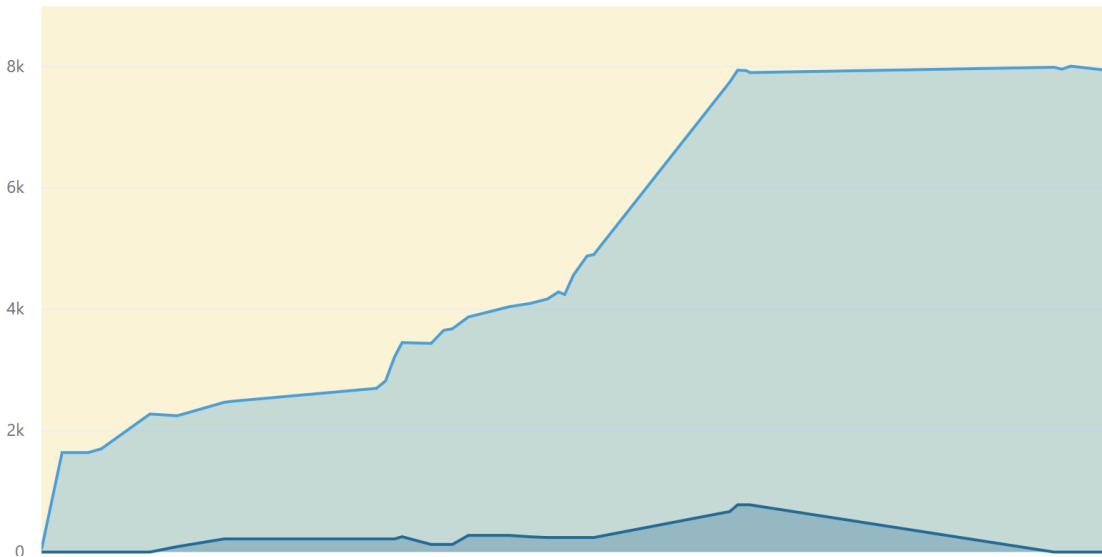


Figura 8.5 Evolución del código duplicado a lo largo del desarrollo

## Capítulo 9. Manuales del Sistema

### 9.1 Manual de Instalación

A continuación, se explican todos los pasos necesarios para instalar el sistema a partir de sus respectivos instaladores.

#### 9.1.1 Android

Se dispone de un fichero con la extensión .apk que debe transferirse a un dispositivo móvil con Android 6.0+ y ejecutarse. No requiere ninguna configuración adicional.

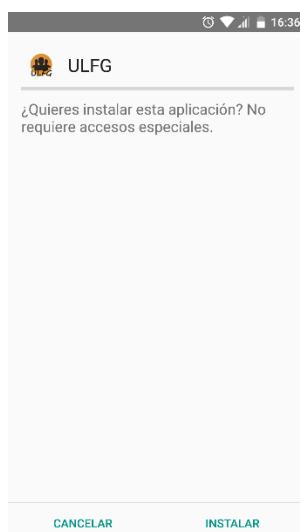


Figura 9.1 Instalación de la apk en Android

## 9.1.2 Windows

Se proporciona un fichero con la extensión .appxbundle y un certificado digital. La aplicación no puede instalarse si no se tiene instalado el certificado en la raíz de confianza. Se debe instalar primero el certificado haciendo doble clic en el mismo y eligiendo como ubicación de instalación “Entidades de certificación de raíz de confianza” cuando el instalador pregunte acerca de la ubicación de almacenamiento.

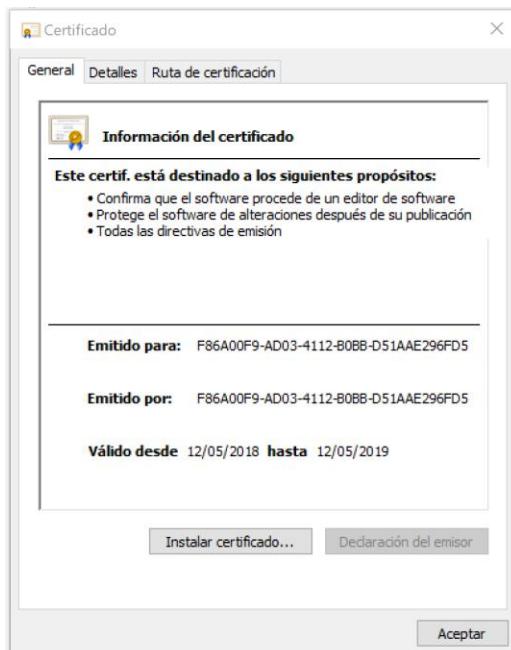
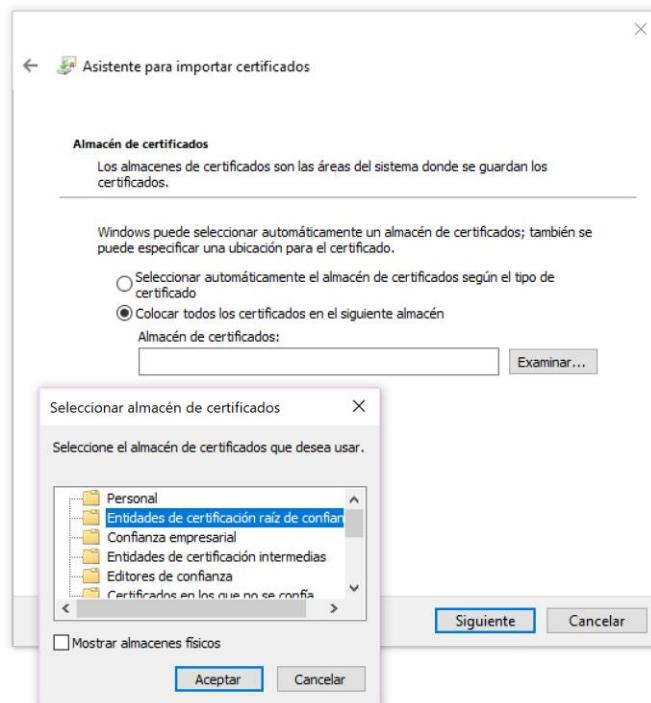


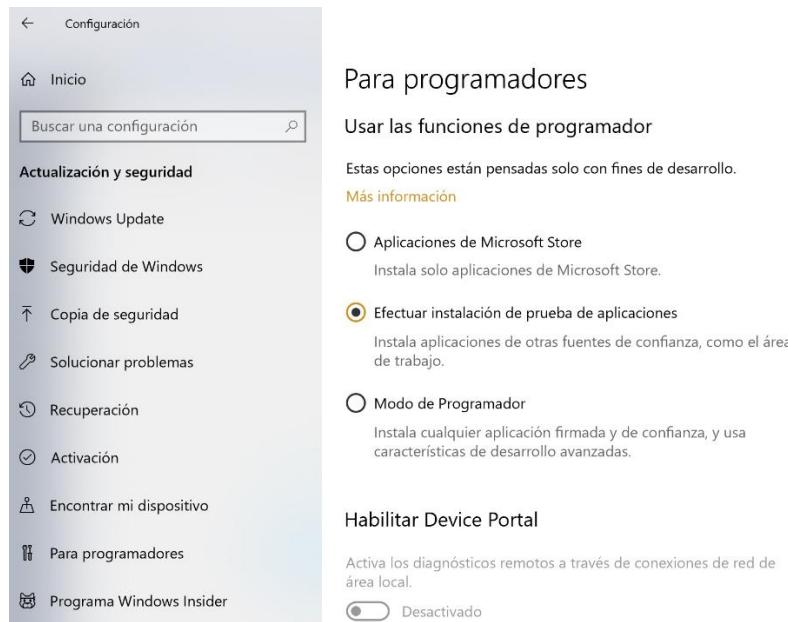
Figura 9.2 Captura de instalación de certificado I



**Figura 9.3 Captura de instalación de certificado II**

Es necesario que el modo de seguridad de Windows sea “Efectuar instalación de prueba de aplicaciones” o “Modo de programador” para poder instalar aplicaciones UWP desde fuera de la tienda. Para ello se debe acceder a:

Configuración > Actualización y seguridad > Para programadores



**Figura 9.4 Cambiar el modo de seguridad de Windows**

Tras tener instalado el certificado y Windows configurado adecuadamente se puede hacer doble clic en el fichero .appxbundle para instalar la aplicación. El instalador detectará e instalará la versión correcta para el sistema operativo (x86 o x64).

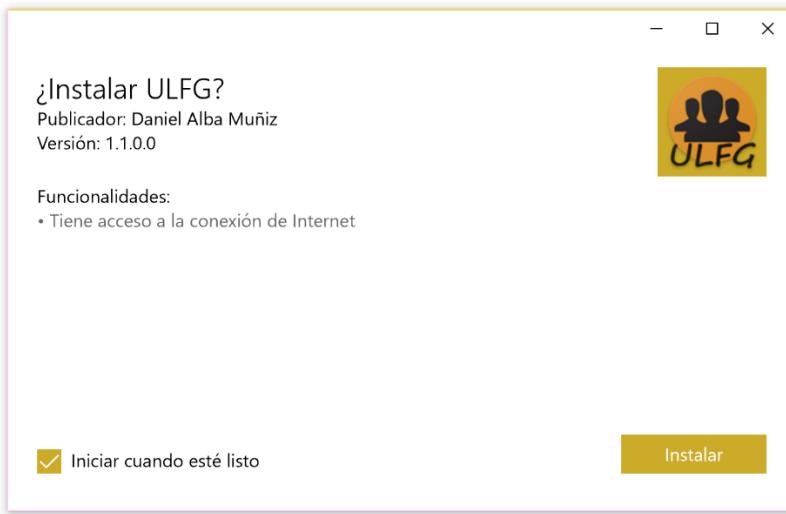


Figura 9.5 Captura de instalación de la aplicación para Windows

Si el sistema no pudiese abrir el fichero .appxbundle se debe ir a la Windows Store y descargar la aplicación “App Installer”.

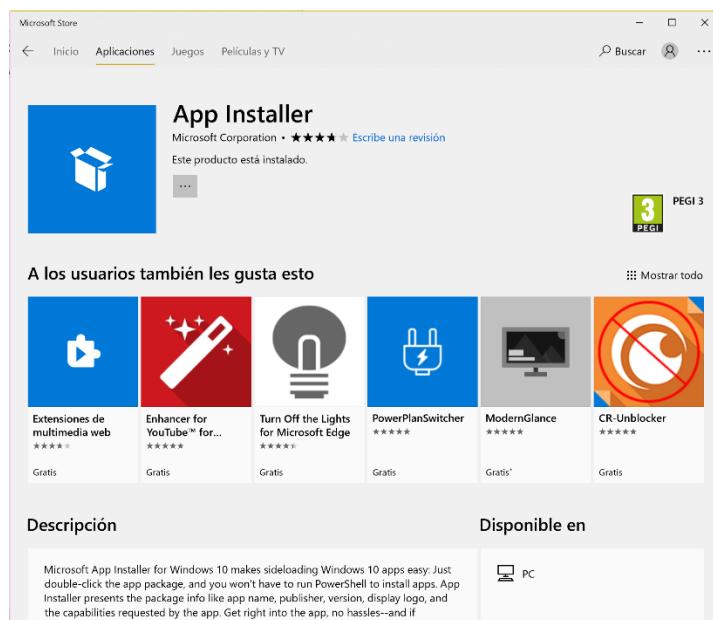


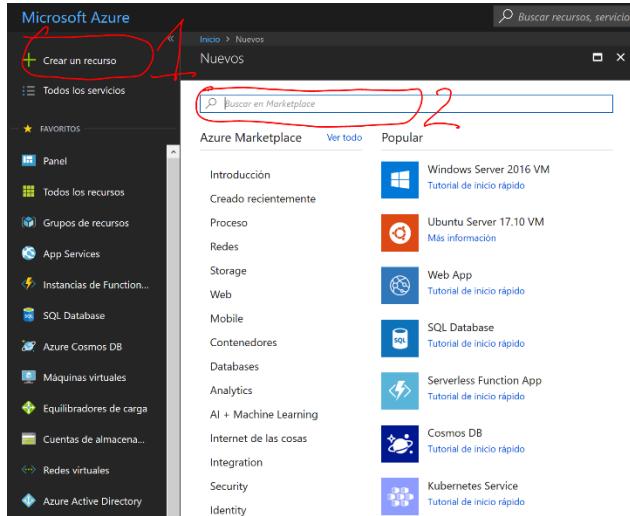
Figura 9.6 Captura de App Installer en la Windows Store

## 9.2 Manual de Ejecución

En este manual se detallan los pasos necesarios para arrancar el sistema a partir del código fuente y se indicarán algunos consejos para la gestión de la operación de este. Se da por hecho la posesión de una licencia de Azure.

## 9.2.1 Creación de la Base de Datos

Comenzamos accediendo al portal de Azure (<https://portal.azure.com/>) e iniciando sesión. En el menú lateral que aparece a la izquierda seleccionamos “Crear un nuevo recurso” y en la barra de búsqueda escribimos “SQL Database”.



*Figura 9.7 Crear recurso de Azure*

Le damos a crear y rellenamos los campos que nos pide:

- Un nombre cualquiera para identificarla.
- Nuestra licencia de Azure.
- Grupo de recursos para almacenar recursos y servicios relacionados. Creamos uno si no lo tenemos ya.
- El servidor donde se alojará la base de datos. Debemos darle una url, una contraseña y seleccionar la región.
- El Plan de tarifa que mejor nos convenga en función de precio y prestaciones
- Los demás campos los podemos dejar por defecto.

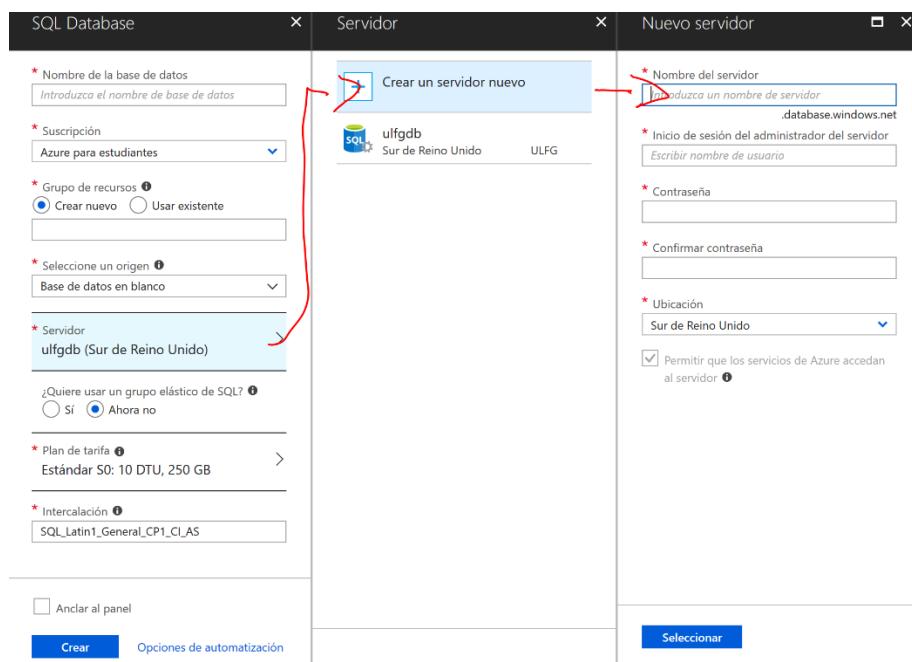


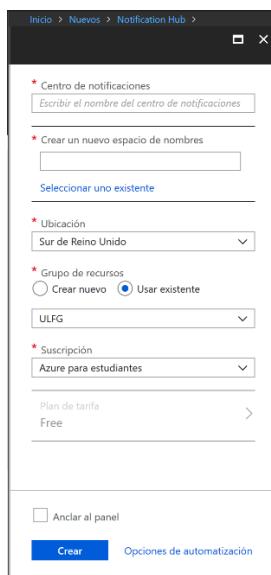
Figura 9.8 Crear SQL Database

Con esto ya tenemos creada la BD.

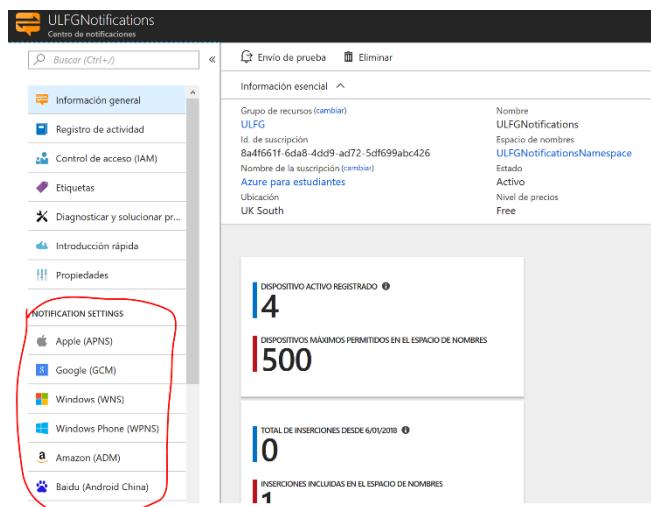
## 9.2.2 Crear y configurar Notification Hub (NH)

Lo primero que debemos hacer es crear un nuevo recurso (similar al apartado anterior) pero esta vez escribimos “Notification” Hub en la barra de búsqueda y creamos el recurso rellenando los campos que se solicitan:

- Un nombre para identificarlo.
- Espacio de nombres para almacenar los dispositivos conectados al hub. Se recomienda crear uno por cada hub.
- Ubicación donde estará.
- Utilizar el mismo grupo de recursos que para la base de datos.
- La suscripción de Azure.
- El plan de tarifa que se desee.

**Figura 9.9 Creación de Notification Hub**

Para configurar el hub con los diferentes PNS nativos de cada plataforma debemos acceder al recurso creado y añadir una serie de datos adicionales para cada PNS.

**Figura 9.10 Localización de la configuración de los PNS**

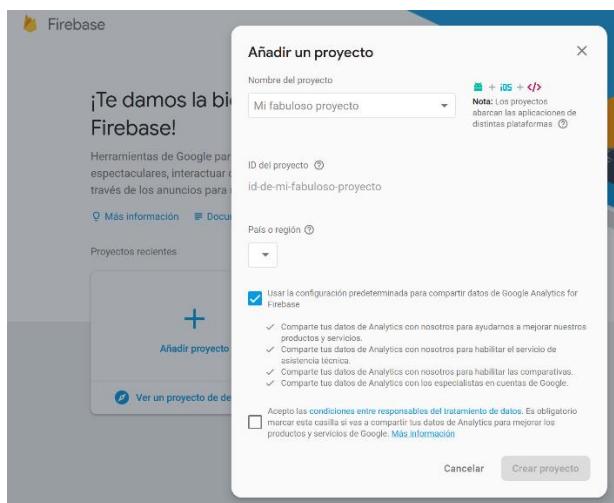
Ahora pasaremos a explicar cómo obtener estos datos para Android y Windows.

### 9.2.2.1 Configuración PNS Firebase (Android)

Antes de continuar es necesario disponer de una cuenta de Google, puede conseguirse una aquí: <https://accounts.google.com/SignUp?hl=es>.

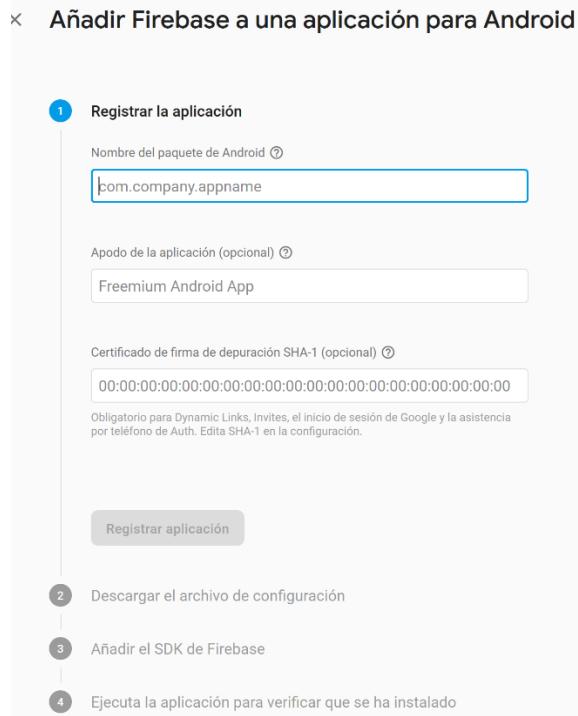
Accedemos a la consola de Firebase introduciendo nuestra cuenta de Google <https://console.firebaseio.google.com/>.

En la página de inicio seleccionamos "Crear nuevo proyecto" e introducimos los datos que solicitados.



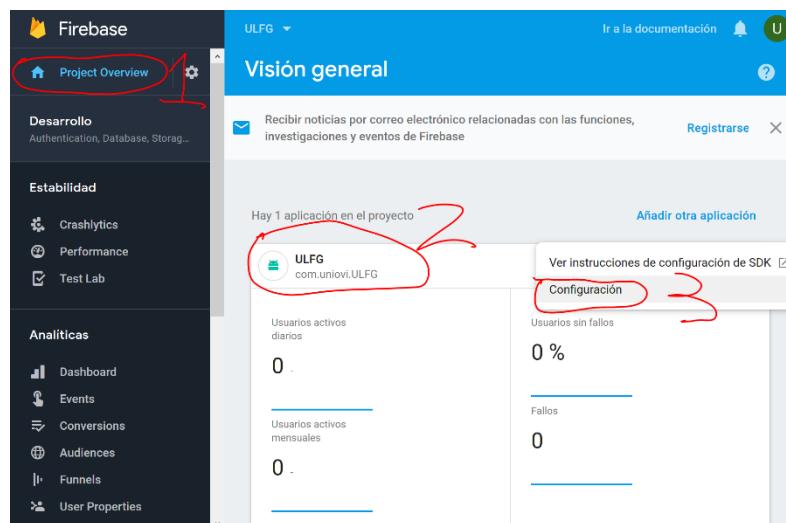
**Figura 9.11 Creación de un proyecto en Firebase**

Entramos al nuevo proyecto y buscamos la opción “Añadir aplicación” y seguimos las instrucciones que proporciona la plataforma para integrarse con nuestra aplicación para Android.



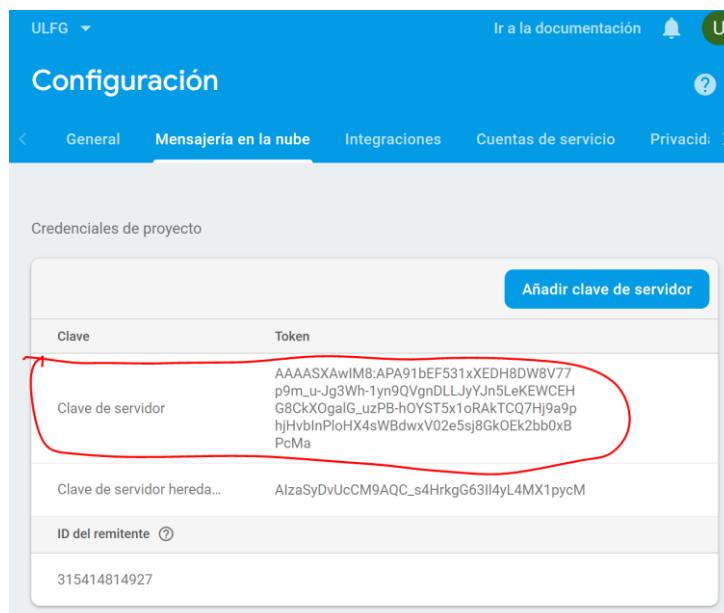
**Figura 9.12 Integrar Firebase con la aplicación para Android**

Una vez integrada ya tenemos disponibles todos los servicios de Firebase, aunque solo usaremos Cloud Messaging para las notificaciones. Ahora debemos obtener una clave para introducirla en Notificacion Hub. Para ello vamos a configuración de nuestra recién vinculada aplicación disponible desde “Project Overview”.



**Figura 9.13 Acceder a los datos de la mensajería en la nube de Firebase**

Dentro de la configuración vamos a la pestaña “Mensajería en la nube” y seleccionamos el campo “clave de servidor”, esta es la clave que necesita NH. Regresamos a Azure y la introducimos en la configuración del PNS vista anteriormente.

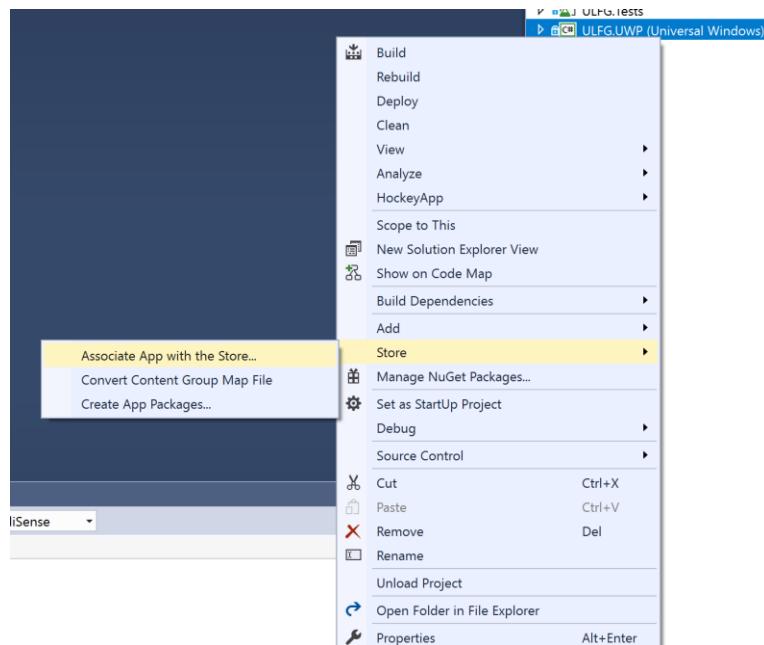


**Figura 9.14 Clave del servidor de notificaciones Firebase**

## 9.2.2.2 Configuración PNS Windows Notification System (WNS)

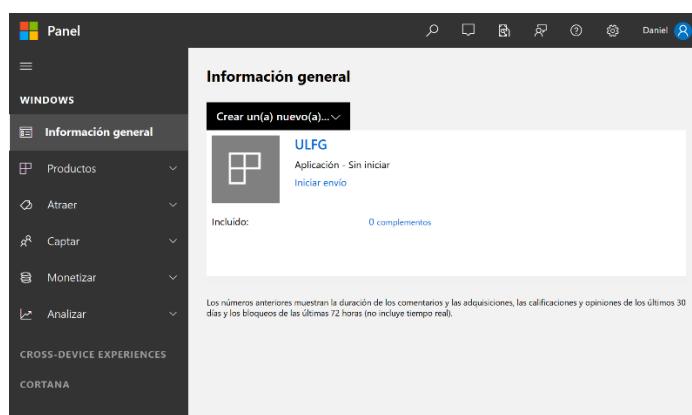
Antes de continuar debe disponerse de una cuenta de Microsoft con licencia de desarrollador: <https://developer.microsoft.com/es-es/store/register>.

Lo primero que debemos hacer es abrir el proyecto con VS y hacer clic derecho en el proyecto de UWP > Store > Associate App with the Store. Nos aseguramos de que la cuenta que se muestra en el asistente es la que tiene la licencia de desarrollador y elegimos un nombre para la aplicación.



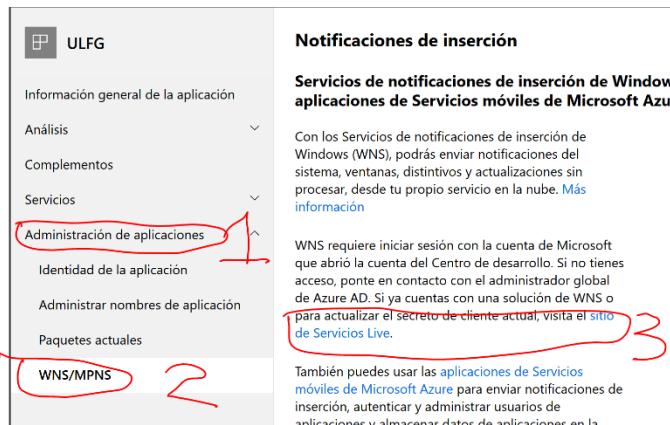
*Figura 9.15 Asociar aplicación UWP con la tienda*

Ahora nos dirigimos al portal del desarrollador de Windows <https://developer.microsoft.com/es-es/dashboard/windows/overview>. Debería aparecer la aplicación que hemos vinculado con anterioridad, le hacemos clic.



*Figura 9.16 Página principal del portal de desarrollador de Windows*

Desplegamos el menú “Administración de aplicaciones” y le damos a “WNS/MPNS”. Una vez dentro buscamos el enlace “sitio de Servicios Live”.



**Figura 9.17** Como acceder a los datos de la aplicación UWP

En esta página tenemos todo lo que necesita NH.

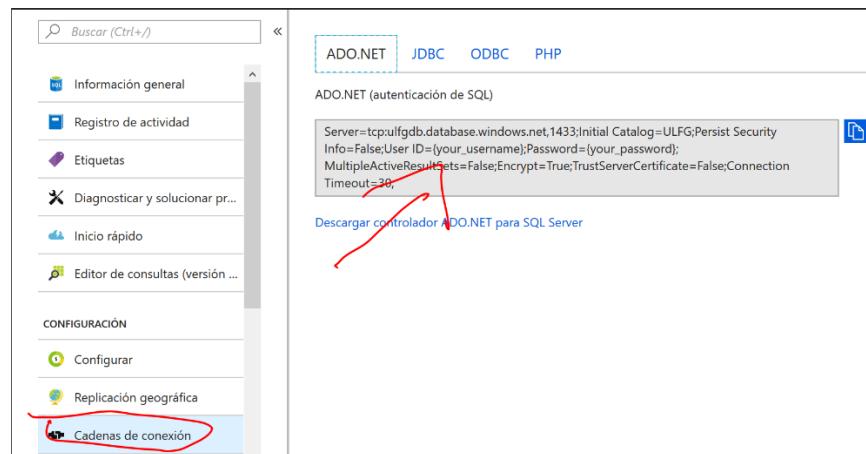
**Figura 9.18** Clave de seguridad

**Figura 9.19** Identificador de la aplicación

### 9.2.3 Preparar el backend

Debemos crear otro recurso de Azure de la misma manera que los demás, pero esta vez buscamos “Mobile App Quickstart” e introducimos los campos solicitados similares a los de otros servicios.

Una vez creado vamos a VS y editamos el fichero web.config en la raíz de ULFGService. Debemos añadir la cadena de conexión de la base de datos creada. Esto podemos obtenerlo en Azure accediendo al servicio SQL Database.

**Figura 9.20 Localización de la cadena de conexión en Azure**

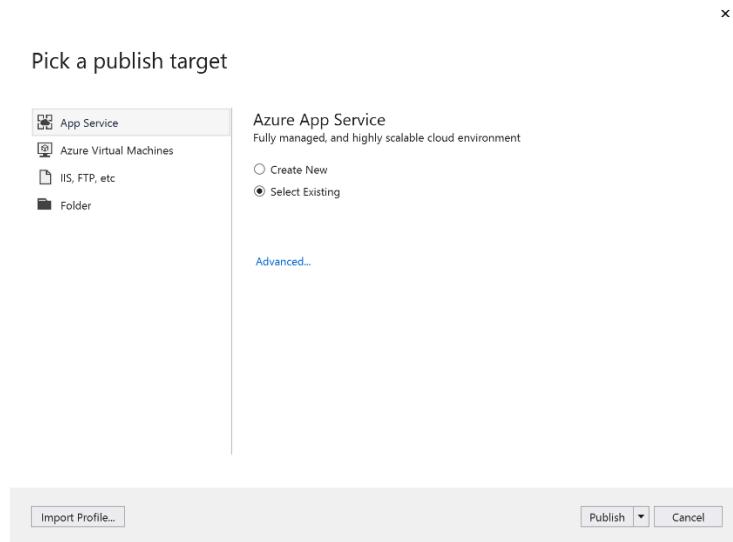
```

<?xml version="1.0" encoding="utf-8"?>
-!--
|   For more information on how to configure your ASP.NET application, please visit
|   http://go.microsoft.com/fwlink/?LinkId=169433
|   -->
-<configuration>
-  <configSections>
-    <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b71d1fba2e7bb96a">
-      <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkId=169433-->
-    </section>
-  </configSections>
-  <connectionStrings>
-    <add name="ULFG_ConnectionString" connectionString="Server=tcp:ulfgdb.database.windows.net,1433;Initial Catalog=ULFG;Persist Security Info=False;User ID=(your_username);Password=(your_password);MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;" />
-      <!-- For Visual Studio 2013 / SQL Express Local DB 2012 replace the above with the following connection string -->
-      <add name="MS_TableConnectionString" connectionString="Data Source=(localdb)\v11.5;Initial Catalog=ULFG;Integrated Security=True;MultipleActiveResultSets=False;Connect Timeout=30;" providerName="System.Data.SqlClient" />
-    </connectionStrings>

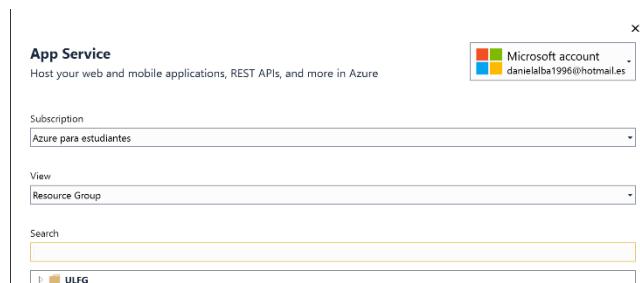
```

**Figura 9.21 Donde colocar la cadena de conexión en ULFGService**

y publicamos el proyecto ULFGService en el App Service. Para ello debemos crear un nuevo perfil de publicación de tipo App Service. Buscamos el servicio de App Service creado dentro del grupo de recursos y lo seleccionamos como objetivo para la publicación. Pedirá credenciales, que serán las mismas de Azure.



**Figura 9.22 Selección del tipo de perfil de publicación**

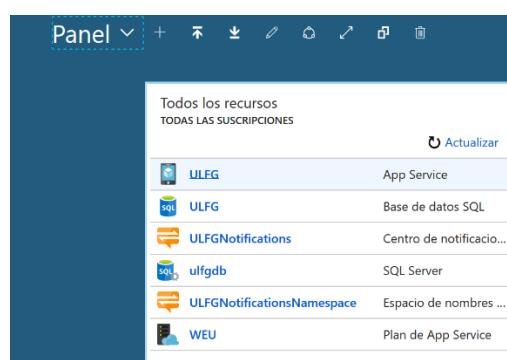


**Figura 9.23 Selección del host para la publicación**

Una vez creado el perfil realizamos la publicación. Debemos incluir la url del backend en la clase Constants de ULFG.Core.Data para que el cliente sepa a donde conectarse.

## 9.2.4 Consejos para la operación

Para gestionar los servicios de Azure se puede usar tanto el portal como VS. En VS se debe abrir la ventana “Server Explorer” y desplegar Azure. En Azure se visualiza una lista de todos los recursos en el panel principal.



**Figura 9.24 Lista de recursos en Azure**

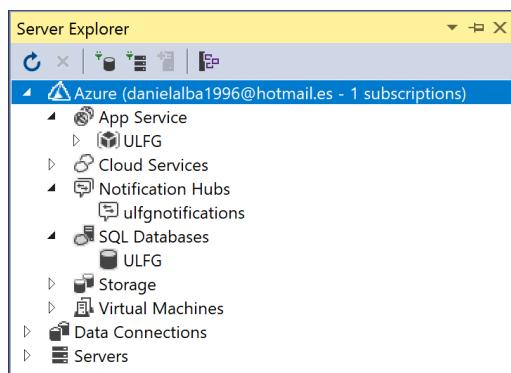


Figura 9.25 Lista de recursos de Azure en VS

En el portal de Azure se pueden configurar todo tipo de alertas para todos los servicios. Primero se debe acceder al servicio y luego seleccionar “Alertas”.

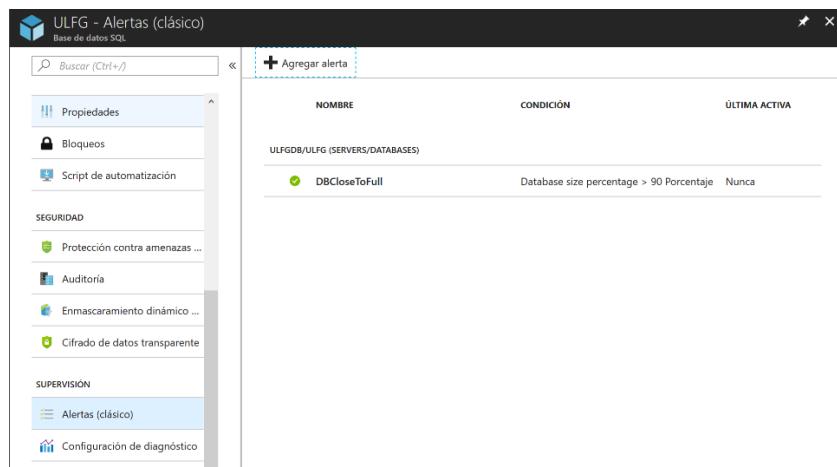


Figura 9.26 Lista de alertas configuradas



Figura 9.27 Creación de una alerta

## 9.3 Manual de Usuario

Las aplicaciones móviles suelen ser bastante intuitivas y no poseer un manual de usuario. En este manual se explicarán brevemente todas las operaciones que es posible realizar con la aplicación:

### 9.3.1 Registro

Se puede acceder al abrir la aplicación mediante un botón en la pantalla de Login. Permite registrar un usuario en el sistema. Requiere conexión a Internet. Los formatos de los campos son los siguientes (todos obligatorios):

- Nombre de usuario: Entre 5 y 16 caracteres.
- Apodo: Entre 3 y 16 caracteres.
- Correo electrónico: Máximo de 24 caracteres y estructura del tipo “nombre@dominio”.
- Contraseña (y su repetición): Entre 8 y 16 caracteres con al menos una letra mayúscula, una minúscula y un número.



Figura 9.28 Ubicación del Registro

### 9.3.2 Login

Es la pantalla que se muestra al abrir la aplicación, se debe introducir un nombre de usuario y una contraseña de un usuario existente y disponer de conexión a internet.

### 9.3.3 Crear publicación

Se accede desde la lista de publicaciones en la página principal. Se debe tener en cuenta:

- El texto debe tener entre 1 y 80.
- La imagen adjuntada no debe superar un tamaño de 850kb.



Figura 9.29 Ubicación de la creación de publicaciones

### 9.3.4 Buscar Usuarios

En la página principal tras haber iniciado sesión. Permite filtrar los usuarios existentes por un texto en su descripción o nombre de usuario.

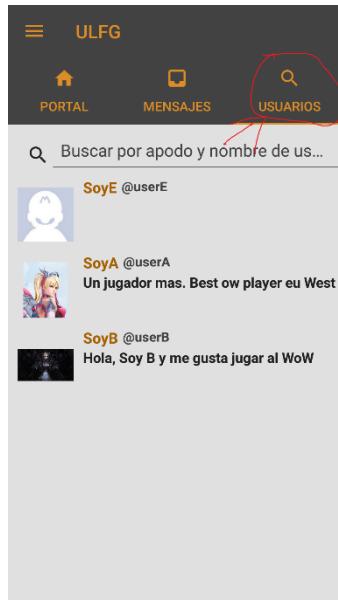


Figura 9.30 Ubicación de la búsqueda de usuarios

### 9.3.5 Editar perfil

Se accede desde el menú desplegable de navegación y permite modificar los datos del usuario actual. Los campos deben cumplir el siguiente formato.

- Apodo: Entre 3 y 16 caracteres.
- Correo electrónico: Máximo de 24 caracteres y estructura del tipo "nombre@dominio".
- Contraseña (y su repetición): Entre 8 y 16 caracteres con al menos una letra mayúscula, una minúscula y un número.
- Biografía: Entre 0 y 240 caracteres.
- Imagen: Inferior a 850kb.

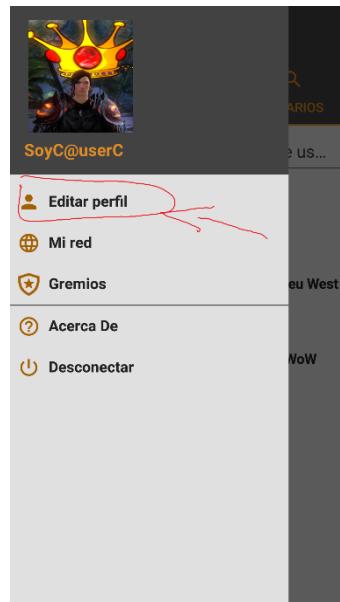


Figura 9.31 Ubicación de la edición del perfil

### 9.3.6 Seguir y bloquear

Desde el perfil de un usuario se puede acceder a los botones de Seguir y Bloquear.

- Bloquear significa que no se podrán recibir mensajes ni seguimientos de un usuario. Todos los seguimientos existentes se eliminan.
- Seguir significa que las publicaciones de un usuario aparecerán en el portal.



Figura 9.32 Ubicación de los seguimientos y bloqueos

### 9.3.7 Ver mensajes directos recibidos

Se pueden enviar mensajes desde el perfil de un usuario. El contenido debe tener entre 1 y 240 caracteres.

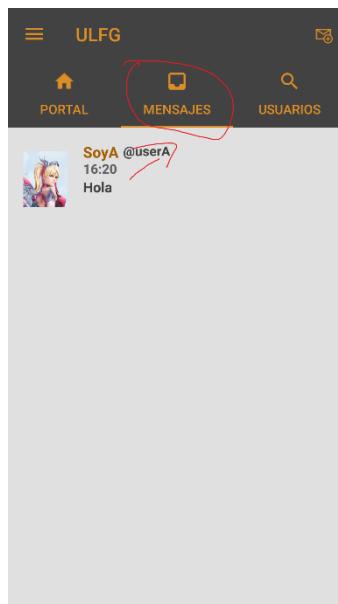


Figura 9.33 Ubicación de los mensajes directos

### 9.3.8 Enviar mensaje directo

Si ya se han intercambiado mensajes con un usuario, es posible continuar la conversación desde la pantalla de mensajes directos tocando el chat en cuestión. Si se desea iniciar una conversación se debe acceder al perfil del usuario y usar la acción del menú secundario superior.

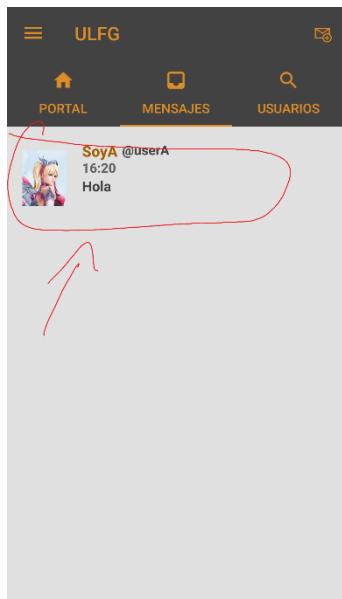


Figura 9.34 Ubicación de envío de mensaje I   Figura 9.35 Ubicación de envío de mensaje II

### 9.3.9 Buscar gremios

Se puede acceder desde el menú principal de navegación. Pulsando en un gremio se puede acceder a sus detalles si se es miembro. Si no se es miembro aparecerá la opción de unirse (si es público) y de contactar con el líder.

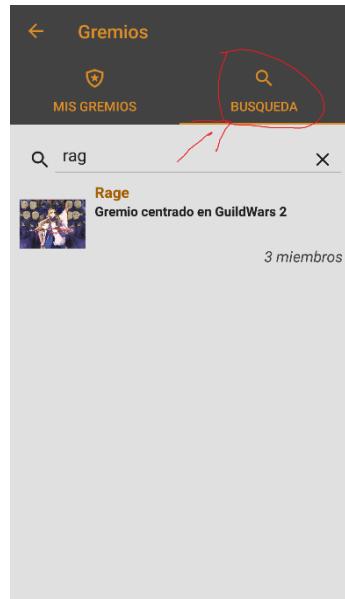


Figura 9.36 Ubicación de la búsqueda de gremios

### 9.3.10 Crear gremio

Se puede acceder desde el menú secundario superior en la página de búsqueda de gremios. Se deberán introducir los siguientes campos:

- Nombre: Entre 3 y 24 caracteres.
- Descripción: Entre 0 y 80 caracteres.
- Visibilidad: Público (Derecha) o Privado (Izquierda).

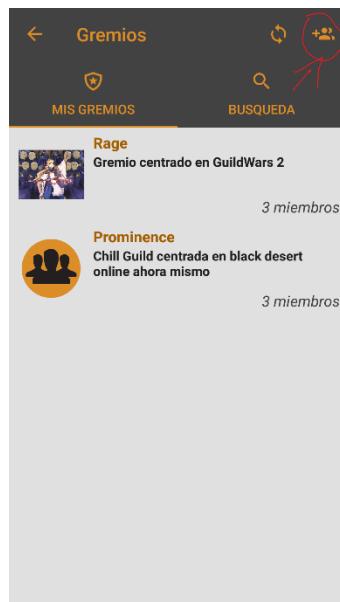


Figura 9.37 Ubicación de la creación de gremios

### 9.3.11 Editar gremio

Se puede acceder desde el menú secundario superior del detalle de un gremio. Permite modificar los siguientes campos:

- Nombre: Entre 3 y 24 caracteres.
- Descripción: Entre 0 y 80 caracteres.
- Visibilidad: Público (Derecha) o Privado (Izquierda).
- Mensaje fijado: Entre 0 y 240 caracteres.
- Imagen: Inferior a 320kb.



Figura 9.38 Ubicación de la edición de un gremio

### 9.3.12 Invitar usuario a gremio

En el menú secundario superior del detalle de un gremio se puede encontrar esta acción. Se abrirá una página para seleccionar el usuario a invitar.



Figura 9.39 Ubicación de invitar a un usuario

### 9.3.13 Abandonar gremio

En el menú secundario superior del detalle de un gremio se puede encontrar esta acción.

**NOTA:** Si el que abandona es el líder del gremio, esta función expulsará a todos los miembros y borrará el gremio.



Figura 9.40 Ubicación de abandonar gremio

### 9.3.14 Expulsar usuario de gremio

Desde la lista de miembros haciendo una pulsación larga en un usuario se puede acceder a la opción de expulsar.



Figura Ubicación de la expulsión del gremio

### 9.3.15 Enviar mensaje grupal al gremio

Esta acción se encuentra en el menú secundario superior de pantalla de detalle de gremio. Permite crear mensajes que podrán ser leídos por todos los usuarios que formen parte del gremio.



Figura 9.41 Ubicación del chat grupal de gremio

### 9.3.16 Cerrar sesión

Esta opción se encuentra en el menú principal de navegación y permite volver a la pantalla de login.

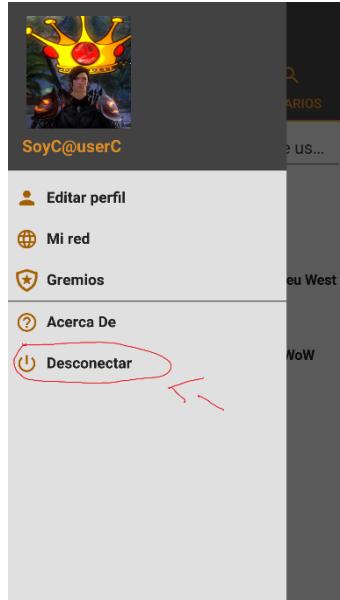


Figura 9.42 Ubicación del cierre de sesión

## 9.4 Manual del Programador

Este manual está destinado a ayudar a otros programadores a ampliar, modificar o entender aspectos de la construcción del sistema.

### Añadir nuevas pantallas

Para añadir nuevas pantallas a la interfaz de usuario se debe seguir el patrón visto en el apartado [7.1.1 Patrón MVVM](#) y ser incluidas en el paquete ULFG.Forms. Los colores de interfaz deben intentar definirse en la medida de lo posible en el fichero App.xaml de ULFG.Forms. Aquellos componentes específicos de una plataforma que no se puedan tratar aquí deben ser configurados en el proyecto nativo.

### Añadir nueva funcionalidad

Las nuevas funcionalidades que deban implementarse deberán incluirse dentro del subpaquete Logic de ULFG.Core. Aquellas funcionalidades que requieran realizarse de forma nativa se deberán implementar utilizando inyección de dependencias (puede consultarse más en el siguiente enlace: <https://xamarinhelp.com/xamarin-forms-dependency-injection>) dentro del subpaquete PlatformInterfaces de ULFG.Forms y los subpaquetes PlatformImpl en los proyectos nativos. Es necesario que todas las plataformas tengan su implementación, en caso contrario el sistema lanzará una excepción.

### Añadir nuevas clases al modelo de dominio

Este proceso debe llevarse a cabo en dos partes:

1) *En ULFG.Core:*

- a) Crear una clase que modele el nuevo elemento similar a las ya existentes en el subpaquete Item.
- b) Crear una interfaz para un nuevo manager y su respectiva implementación dentro del subpaquete ItemManager para gestionar las operaciones sobre el nuevo elemento. Debe utilizarse cualquiera de los managers existentes como plantilla y emplear el cliente común de base de datos que está representado por la clase SyncClientProvider.
- c) En SyncClientProvider se debe añadir una nueva tabla en el constructor para almacenar el nuevo elemento del dominio.

2) *En ULFGService:*

- a) Se debe crear una nueva tabla para el elemento del dominio que se desea añadir. Para ello ha de usarse [EntityFramework](#), definiendo el modelo en el subpaquete DataObjects y las restricciones dentro de la clase ULFGContext
- b) Realizar una nueva migración con los cambios al modelo
- c) Publicar el backend en App Service

### Añadir nuevos servicios a la API

Los nuevos servicios de la API deberán incluirse en forma de controladores HTTP dentro del subpaquete Controllers de ULFGService.

# Capítulo 10. Conclusiones y Ampliaciones

## 10.1 Conclusiones

Este proyecto surgió a raíz de uno similar realizado para la asignatura “Software para Dispositivos Móviles” que consistía en una simple herramienta para Android que permitía buscar grupo para jugar a un determinado juego. Esta herramienta ha evolucionado para convertirse en una red social con la multiplataforma como idea fundamental. Aunque se disponía de una idea general de lo que se quería construir desde el principio, he tenido dificultades a la hora de decidir la funcionalidad que se iba a implementar y su prioridad.

A pesar de que se ha tenido que realizar un esfuerzo adicional para lograr una fluidez aceptable en la interfaz de usuario, este esfuerzo no es nada comparado con el de tener que elaborar una aplicación diferente para cada plataforma.

Considero que Xamarin.Forms aún tiene que mejorar un poco para poder ser usado de forma general para el desarrollo de aplicaciones: Las builds tardan cerca de 12 minutos en terminar y al hacer un despliegue en un dispositivo real se tarda cerca de otros 12 minutos. Además, he sufrido cuelgues y ralentizaciones en mi dispositivo móvil durante tareas cotidianas tras un despliegue en modo depuración.

En general estoy satisfecho con el proyecto desarrollado, he logrado una sincronización offline relativamente transparente y una interfaz de usuario bastante sencilla.

Durante el desarrollo de la documentación he aprendido que lo importante no son las páginas, sino incluir la mayor cantidad de información útil en el menor volumen posible y que las imágenes sean lo suficientemente grandes y no haya problemas para visualizarlas correctamente.

Respecto a las pruebas del software, he aprendido que es más importante un diseño eficiente de las situaciones a cubrir en las que se incluyan las funcionalidades que tengan cierta complejidad frente a un conjunto de casos de prueba inefectivos que prueben todos y cada uno de los métodos de la aplicación.

Otra de las cosas que he aprendido durante el desarrollo de este proyecto es a elaborar y discutir diferentes tipos de diagramas UML para reflejar el funcionamiento y la arquitectura de la aplicación.

## 10.2 Ampliaciones

A continuación, se incluyen las posibles ampliaciones que se podrían realizar en el sistema. No se han realizado por falta de tiempo y por ser menos prioritarias.

1. Borrar cuenta (esto sería necesario para poder distribuir la aplicación, debido al “Derecho al Olvido” del RGPD).
2. Enviar más tipos de adjuntos en las publicaciones como videos y gifs.
3. Permitir comentar y valorar publicaciones. Incluir un campo en el perfil de cada usuario basado en las valoraciones de sus publicaciones que represente la reputación del usuario.
4. Integración con Discord para importar contactos, gremios o el historial de juego. Dispone de una API pública a la que se puede consultar.
5. Integración con la plataforma Battle.Net. Al igual que Discord posee una API pública.
6. Posibilidad de configurar las notificaciones que se desea recibir.
7. Recordar contraseña mediante el envío de un mensaje de confirmación al correo electrónico del usuario.
8. Añadir un calendario a los gremios que permita al líder crear y planificar eventos.
9. Añadir llamadas VoIP entre usuarios.

# Capítulo 11. Presupuesto

En este apartado se incluye una estimación del presupuesto para el desarrollo del sistema ULFG estimado en una duración de 4 meses y 363 horas de trabajo. Para la realización del presupuesto se han contemplado las siguientes unidades de trabajo:

1. Estudio de alternativas
2. Sistema de control de versiones e integración continua
3. Diseño de la base de datos
4. Configuración inicial y arranque del sistema
5. Sprint 1
6. Sprint 2
7. Sprint 3
8. Sprint 4
9. Sprint 5
10. Documentación detallada

## 11.1 Costes directos

### 11.1.1 Costes básicos

#### 11.1.1.1 Recursos de trabajo

COD	UD	Descripción	Precio
RT001	h	Analista programador	50 €

#### 11.1.1.2 Recursos materiales

COD	UD	Descripción	Precio
RM001	h	Ordenador portátil de desarrollo	2,5 €

### 11.1.2 Costes unitarios

COD	UD	Descripción	Precio	Hora	Subtotal	Importe
ANAPROG		Coste de Analista programador				
RT001	h	Analista programador	50	1	50	€

<b>RM001</b>	h	Ordenador portátil de desarrollo	2,5	1	2,5 €
<b>Precio por hora</b>				<b>52,5 €</b>	

### 11.1.2.1 Precios de unidades de trabajo

<b>Precio Estudio de alternativas nº1</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>20</b>	Analista programador	52,5 €	1.050
<b>TOTAL</b>			<b>1.050 €</b>

<b>Precio Sistema de control de versiones nº2</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>10</b>	Analista programador	52,5 €	525
<b>TOTAL</b>			<b>525 €</b>

<b>Precio Diseño de base de datos nº3</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>12</b>	Analista programador	52,5 €	630
<b>TOTAL</b>			<b>630 €</b>

<b>Precio Configuración inicial y arranque del sistema nº4</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>10</b>	Analista programador	52,5 €	525
<b>TOTAL</b>			<b>525 €</b>

<b>Precio Sprint 1 nº5</b>			
<b>Medición</b>	Concepto	Coste unitario	Total

<b>39</b>	Analista programador	52,5	2.047,50
		€	
		<b>TOTAL</b>	<b>2.047,50</b>
		€	

<b>Precio Sprint 2</b>			
<b>nº6</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>32</b>	Analista programador	52,5	1.680
		€	
		<b>TOTAL</b>	<b>1.680</b>
		€	

<b>Precio Sprint 3</b>			
<b>nº7</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>28</b>	Analista programador	52,5	1.470
		€	
		<b>TOTAL</b>	<b>1.470</b>
		€	

<b>Precio Sprint 4</b>			
<b>nº8</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>40</b>	Analista programador	52,5	2.100 €
		<b>TOTAL</b>	<b>2.100 €</b>

<b>Precio Sprint 5</b>			
<b>nº9</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>52</b>	Analista programador	52,5	2.730
		€	
		<b>TOTAL</b>	<b>2.730</b>
		€	

<b>Precio Documentación detallada</b>			
<b>nº10</b>			
<b>Medición</b>	Concepto	Coste unitario	Total
<b>120</b>	Analista programador	52,5	6.300
		€	
		<b>TOTAL</b>	<b>6.300</b>
		€	

## 11.1.3 Costes software

Medición	UD	Descripción	Coste unitario	Total
4	mes	Azure SQL Database	5,25 €	21 €

## 11.2 Costes indirectos

### 11.2.1 Gastos indirectos del proyecto

COD	UD	Descripción	Precio
GIP001	Año	Office 365 Personal	69 €
GIP002	Usuario/Mes	Visio Professional	12,70 €
GIP003	Mes	Visual Studio Enterprise	250 €
GIP004	usuario	Licencia individual de desarrollador de Windows	19 €

#### 11.2.1.1 Gastos indirectos del proyecto unitarios

COD	UD	Descripción	Precio
OFF365		Coste de Microsoft Office 365 Personal	
GIP001	Año	Microsoft Office 365 Personal	69 €
		Coste Mensual	5,75 €
		Coste total (x4)	23 €

COD	UD	Descripción	Precio
VP		Coste de Visio Professional	
GIP002	Usuario/Mes	Visio Professional	12,70 €

<b>Coste Mensual</b>	<b>12,70 €</b>
<b>Coste total (x4)</b>	<b>50,8 €</b>

COD	UD	Descripción	Precio
<b>VSE</b>		Coste de Visual Studio Enterprise	
<b>GIP003</b>	Mes	Visual Studio Enterprise	250 €
		<b>Coste Mensual</b>	<b>250 €</b>
		<b>Coste total (x4)</b>	<b>1.000 €</b>

COD	UD	Descripción	Precio
<b>WL</b>		Coste Licencia individual de desarrollador de Windows	
<b>GIP004</b>	usuario	Licencia individual de desarrollador de Windows	19 €
		<b>Coste por proyecto</b>	<b>3,8 €</b>
		<b>Coste total</b>	<b>3,8 €</b>

## 11.2.2 Gastos indirectos del entorno de desarrollo

COD	UD	Descripción	Precio
<b>GIED001</b>	mes	Sistema de comunicación por vía telefónica e internet	55 €
<b>GIED002</b>	ud	Gastos de mobiliario	250 €
<b>GIED003</b>	€/m3	Agua	0,11 €
<b>GIED004</b>	€	Luz kW/h	1,71 €

### 11.2.2.1 Gastos indirectos del entorno de desarrollo unitarios

COD	UD	Descripción	Precio
-----	----	-------------	--------

SCTI	Coste de Sistema de comunicación por vía telefónica e internet
<b>GIED001</b>	mes Sistema de comunicación por vía telefónica e internet
	<b>Coste 55 € Mensual</b>
	<b>Coste total 220 € (x4)</b>

COD	UD	Descripción	Precio
GM		Coste de Gastos de mobiliario	
<b>GIED002</b>	ud	Gastos de mobiliario	250 €
		<b>Coste 250 € Mensual</b>	
		<b>Coste total 250 €</b>	

COD	UD	Descripción	Precio
AGU		Coste de Agua	
<b>GIED003</b>	€/m3	Agua	0,11 €
		<b>Coste hora 0,11 €</b>	
		<b>Coste total 39,93 € (x363)</b>	

COD	UD	Descripción	Precio
LUZ		Coste de Luz	
<b>GIED004</b>	€	Luz	1,71 € kW/h
		<b>Coste hora 0,128 €</b>	
		<b>Coste total 46,46€ (x363)</b>	

## 11.3 Presupuesto completo

Medición	Concepto	Coste unitario	Total
<b>1</b>	Costes de software	21 €	21 €
<b>1</b>	Precio nº1: Estudio de alternativas	1.050 €	1.050 €
<b>1</b>	Precio nº2: Sistema de control de versiones e integración continua	525 €	525 €
<b>1</b>	Precio nº3: Diseño de la base de datos	630 €	630 €
<b>1</b>	Precio nº4: Configuración inicial y arranque del sistema	525 €	525 €
<b>1</b>	Precio nº5: Sprint 1	2.047,50 €	2.047,50 €
<b>1</b>	Precio nº6: Sprint 2	1.680 €	1.680 €
<b>1</b>	Precio nº7: Sprint 3	1.470 €	1.470 €
<b>1</b>	Precio nº8: Sprint 4	2.100 €	2.100 €
<b>1</b>	Precio nº9: Sprint 5	2.730 €	2.730 €
<b>1</b>	Precio nº10: Documentación Detallada	6.300 €	6.300 €
		<b>SUMA</b>	19.078,50 €
		<b>Costes indirectos</b>	1.547,6 €
		<b>Imprevistos (10%)</b>	1.907,85 €
		<b>Beneficios (20%)</b>	3.815,7 €
<b>PRESUPUESTO DE EJECUCIÓN</b>			26.349,65 €

# Capítulo 12. Referencias Bibliográficas

## 12.1 Libros y Artículos

Libros y artículos usados de alguna forma durante el desarrollo del proyecto o su documentación.

**[Hermes, Dan (2015)]** “Xamarin Mobile Application Development”. Apress. ISBN: 978-1-4842-0215-9.

## 12.2 Referencias en Internet

En este apartado se incluyen todas las páginas Web consultadas para cualquier aspecto relacionado con el desarrollo del sistema o su documentación:

**[Microsoft]** “Documentación de App Service Mobile Apps”: <https://azure.microsoft.com/es-es/documentation/learning-paths/appservice-mobileapps/> [Consultado en 2018].

**[Microsoft]** “Incorporación de notificaciones push a la aplicación de Xamarin.Forms”: <https://docs.microsoft.com/es-es/azure/app-service-mobile/app-service-mobile-xamarin-forms-get-started-push#update-the-server-project-to-send-push-notifications> [Consultado en 2018].

**[Microsoft]** “Enviar notificaciones de inserción desde aplicaciones móviles de Azure”: <https://docs.microsoft.com/es-es/xamarin/xamarin-forms/data-cloud/push-notifications/azure> [Consultado en 2018].

**[Microsoft]** “Track progress by creating status and trend query-based charts”: <https://docs.microsoft.com/es-es/vsts/report/dashboards/charts?view=vsts> [Consultado en 2018].

**[Microsoft]** “Introduction to Continuous Integration with Xamarin”: <https://docs.microsoft.com/en-us/xamarin/tools/ci/intro-to-ci> [Consultado en 2018].

**[Microsoft]** “Build your Xamarin app”: <https://docs.microsoft.com/en-us/vsts/pipelines/apps/mobile/xamarin?view=vsts&tabs=vsts> [Consultado en 2018].

**[Microsoft]** “DocFx User Manual”: [https://dotnet.github.io/docfx/tutorial/docfx.exe\\_user\\_manual.html](https://dotnet.github.io/docfx/tutorial/docfx.exe_user_manual.html) [Consultado en 2018].

**[Ritchie, Allan]** “UserDialogs”: <https://github.com/aritchie/userdialogs> [Consultado en 2018].

**[Google]** “Material Icons”: <https://material.io/tools/icons/?style=baseline> [Consultado en 2018].

**[Google]** “Tools for picking colors”: <https://material.io/design/color/#tools-for-picking-colors> [Consultado en 2018].

**[Google]** “Color Tool”: <https://material.io/tools/color/#!/> [Consultado en 2018].

**[Synergix]** “Modelo de Dominio”: <https://synergix.wordpress.com/2008/07/10/modelo-de-dominio/> [Consultado en 2018].

**[Dotneco]** “Sending Xamarin Forms Push Notification to single recipient”: <https://dotnetco.de/sending-xamarin-forms-push-notification-single-recipient/> [Consultado en 2018].

**[Opensource]** “NSubstitute. Docs and getting help”: <http://nsubstitute.github.io/help.html> [Consultado en 2018].

**[NewtonSoft]** “Json .NET”: <https://www.newtonsoft.com/json> [Consultado en 2018].

**[Luberda, Daniel]** “FFImageLoading”: <https://github.com/luberda-molinet/FFImageLoading> [Consultado en 2018].

**[Impressum]** “ImageColorPicker”: <https://imagecolorpicker.com/> [Consultado en 2018].

**[ShareIcon]** “Share Icon”: <https://www.shareicon.net/> [Consultado en 2018].

**[Nurik, Roman]** “Android Asset Studio”:

<http://romannurik.github.io/AndroidAssetStudio/index.html> [Consultado en 2018].

**[SBSBlogers]** “Installing a Self-Signed Certificate as a Trusted Root CA in Windows Vista”:

<https://blogs.technet.microsoft.com/sbs/2008/05/08/installing-a-self-signed-certificate-as-a-trusted-root-ca-in-windows-vista/> [Consultado en 2018].

**[Xamarin]** “Xamarin: Mobile App Development & App Creation Software”:

<https://www.xamarin.com/> [Consultado en 2018].

**[lucasg]** “Creating an Appx package without losing it's mind”:

<https://lucasg.github.io/2018/01/15/Creating-an-appx-package/> [Consultado en 2018].

**[EntityFrameworkTutorial]** “What is Code-First”:

<http://www.entityframeworktutorial.net/code-first/what-is-code-first.aspx> [Consultado en 2018].

**[Rupert, René]** “Azure Push Notification Hub and Xamarin.Forms - René Ruppert - Xamarin University Lightning Lecture”: <https://www.youtube.com/watch?v=le2IDY22xwM> [Consultado en 2018].

**[Wikipedia]** “Lenguaje unificado de modelado”:

[https://es.wikipedia.org/wiki/Lenguaje\\_unificado\\_de\\_modelado#Tipos\\_de\\_diagramas\\_en\\_UM\\_L\\_2.5](https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado#Tipos_de_diagramas_en_UM_L_2.5) [Consultado en 2018].

**[Gómez, Miguel Ángel]** “El patrón MVVM en Xamarin Forms”:

<https://miguelgomez.io/xamarin/patron-mvvm-xamarin-forms/> [Consultado en 2018].

**[Gobierno de España]** “Métrica v3”:

[http://administracionelectronica.gob.es/pae/Home/pae\\_Documentacion/pae\\_Metodolog/pae\\_Metrica\\_v3.html.2018](http://administracionelectronica.gob.es/pae/Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3.html.2018) [Consultado en 2018].

**[RGPD]** Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, de Protección de Datos (D.O.U.E. nº.119, de 4 de mayo de 2016) .

# Capítulo 13. Apéndices

## 13.1 Glosario y Diccionario de Datos

En este apartado se incluyen por orden alfabético todos los términos que se consideran importantes con una descripción breve de su significado dentro de la aplicación:

- **Servicio Web / API:** “Application Programming Interface”. Componente que se encuentra en un servidor y que proporciona una funcionalidad a una aplicación instalada en un dispositivo.
- **PNS:** “Platform Notification System”, el sistema de notificaciones de una plataforma.
- **RGPD:** “Reglamento General de Protección de Datos”, que entró en vigor el 25 de mayo de 2018.
- **Software:** Programa para un dispositivo informático.
- **Análisis estático:** Análisis del código fuente de la aplicación.
- **Certificado digital:** Fichero que permite verificar la procedencia de un programa.
- **Fichero .apk:** Instalador de aplicaciones en Android.
- **Fichero .appxbundle:** Instalador de una aplicación UWP para Windows 10.
- **Encriptación:** Ocultar un texto usando algún tipo de estrategia.

## 13.2 Anexos

Directorio	Contenido
<i>Ejecutables</i>	Contiene los instaladores de las aplicaciones Android y Windows. Incluye también el certificado necesario para instalar la aplicación para Windows.
<i>Descripción detallada de las clases</i>	Contiene la descripción detallada de las clases como un sitio web en html.
<i>Código fuente</i>	Contiene el código fuente de la aplicación que puede ser abierto con Visual Studio.
<i>Diagramas</i>	Contiene los diagramas utilizados en la documentación en formato vsdx y png