

APLICACIÓN CENTRO FORMACIÓN EDI ESCUELA DE INFORMÁTICA E INGLÉS

Proyecto Fin de ciclo

David Piñuel bosque



4 DE JUNIO DE 2023

Justificación y Objetivos

Teniendo en cuenta la gran importancia que tiene la evolución de la enseñanza en la sociedad actual, y el refuerzo que supone la evolución tecnológica, se plantea la tesisura de reforzar con un registro todo lo relacionado con el centro de enseñanza mediante el personal de dicho centro, para fortalecer el crecimiento personal de los estudiantes, por ello se desarrolla esta aplicación, una en Android para el personal del centro.

En este proyecto se desarrolla una aplicación, una que permite al administrador unas funciones para añadir, modificar, eliminar, consultar y asignar diferentes alumnos, profesores, cursos y aulas además de otras opciones como un mapa donde se ve su geolocalización y una información tanto del centro de formación como del diseño de las aulas.

Resumen

The final end-of-cycle Project Will be carried out for a training center EDI School of Informatics and English. It will be an application for the Android operating system with authentication where they can manage and maintain a database for students who Will take courses in the center's classrooms that will be taught by teachers, as well as providing information about the center and its location, as well as a detailed report of the classroom design.

On the other hand, it will have an administrator user who will be able to perform operations on the database and other types of users, in which they will not have access to the database but will have access to the center's information such as its location and the detailed report on the classroom design.

Finally, I want to mention that the application will have a main menu with different options. When you start you will not have access to the options except to log in. Once logged in, it will depend on whether you are an administrator user or a generic user, some options or others will be enabled. It will also have a secondary menu with only one option, sign out.

Índice general

Justificación y Objetivos	1
Resumen	1
1. Introducción	6
2. Objetivos	7
2.1. Objetivos específicos	7
3. Ejemplos Aplicaciones Comunes	8
3.1. Ítaca	8
3.2. Alexia	9
3.3. Additio App	9
3.4. Gescola	10
3.5. Conclusiones	10
4. Metodología	11
4.1. Sistema de control de versiones.	11
5. Tecnología	13
5.1. Aplicaciones de diseño	13
5.1.1. Balsamiq Wireframes	13
5.1.2. Draw.io	15
5.1.3. Visio Profesional	16
5.2. Aplicación Android	17
5.2.1. Android Studio	17
5.2.2. Kotlin/Android	19
5.2.3. Data Binding	20
5.2.4. GitHub	20
6. Diseño	21
6.1. Aplicación Android	21
6.2. Estructuración Aplicación Android	23
6.2.1. Entidad-relación	23
6.2.2. Modelo relacional	25
6.2.2.1. Aulas	25
6.2.2.2. Usuario	25
6.2.2.3. Cursos	26
6.2.2.4. Profesor	26
6.2.2.5. Alumno	27
6.3. Diagrama de clases	27
6.4. Diagramas de casos de uso	28
6.4.1. Administrador	29
6.4.2. Otros usuarios	30
7. Arquitectura	32
7.1. Android	32
8. Implementación	35
8.1. Android	35
8.1.1. Interfaz	35
8.1.2. Navegación entre pantallas	37
8.1.3. Data Binding	38
8.1.4. Modelo	42
8.1.5. Controlador	43
8.1.6. Vista	45
9. Funcionamiento de la aplicación	46
10. Análisis de Negocio	59
11. Generar App Android .APK	62
12. Conclusiones	66
13. Posibles evoluciones de los proyectos	67
14. Bibliografía	68
15. Anexo A	69
16. Anexo B	70

Índice de Ilustraciones

Ilustración 1: Creaciones y fusión de ramas	11
Ilustración 2: Proyecto Android.....	13
Ilustración 3: Herramienta de diagramas Draw.io para escritorio.	14
Ilustración 4: Visio Profesional.	15
Ilustración 5: Entorno de Desarrollo Integrado Android Studio.	17
Ilustración 6: Distribución acumulada por versión de Android.	18
Ilustración 7: Menú principal y menú lateral con opciones.	21
Ilustración 8: Diagrama de base de datos.	23
Ilustración 9: Tabla Aulas.....	24
Ilustración 10: Tabla Users.....	24
Ilustración 11: Tabla Cursos.....	25
Ilustración 12: Tabla Profesor	25
Ilustración 13: Tabla Alumnos	26
Ilustración 14: Diagrama de clases.	27
Ilustración 15: Diagrama Caso de Uso Para El Administrador	29
Ilustración 16: Arquitectura Model-View-ViewModel.	33
Ilustración 17:Herramienta gráfica de Android Studio.....	35
Ilustración 18:Herramienta gráfica de Android Studio.....	35
Ilustración 19:Herramienta gráfica de navegación entre vistas de Android Studio	36
Ilustración 20:Asignación de Acción Botón Menú Principal Parte 1.	37
Ilustración 21:Asignación de Acción Botón Menú Principal Parte 2.	38
Ilustración 22:Asignación de Acción Botón Menú Principal Parte 3.	38
Ilustración 23: Código Adaptador Menú Principal Parte 1.....	39
Ilustración 24: Código Adaptador Menú Principal Parte 2.....	39
Ilustración 25: Código Adaptador Menú Principal Parte 3.....	40
Ilustración 26: Modelo Tabla Aulas.	41
Ilustración 27: Modelo Tabla Users.....	41
Ilustración 28: Modelo Tabla Cursos.....	41
Ilustración 29: Modelo Tabla Profesores.....	41
Ilustración 30: Modelo Tabla Alumnos.	41
Ilustración 31: Métodos de las Tablas BBDD Parte 1.	42
Ilustración 32: Métodos de las Tablas BBDD Parte 2.	42
Ilustración 33: Métodos de las Tablas BBDD Parte 3.	43
Ilustración 34: Vista Plantilla Alumnos.....	44
Ilustración 35: Funcionalidad Alumnos.....	46
Ilustración 36: Funcionalidad Profesores	47
Ilustración 37: Funcionalidad Cursos	48
Ilustración 38: Funcionalidad Aulas	49
Ilustración 39: Funcionalidad Menú Principal	50
Ilustración 40: Funcionalidad Inicio Sesión.....	51
Ilustración 41: Funcionalidad Contactos.....	52
Ilustración 42: Funcionalidad Diseño Aulas Parte 1.....	53
Ilustración 43: Funcionalidad Diseño Aulas Parte2	54
Ilustración 44: Funcionalidad Geolocalización Parte 1	55
Ilustración 45: Funcionalidad Geolocalización Parte 2	56
Ilustración 46: Funcionalidad Menú Secundario Cerrar Sesión.....	57
Ilustración 47: Generar APK Parte 1	61
Ilustración 48: Generar APK Parte 2	62
Ilustración 49: Generar APK Parte 3	62
Ilustración 50: Generar APK Parte 4	63
Ilustración 51: Generar APK Parte 5	63
Ilustración 52: Generar APK Parte 6	64
Ilustración 53: Generar APK Parte 7	64

1. Introducción

El desarrollo de los estudiantes de un centro está englobado desde la infancia hasta avanzada la adolescencia, donde dicho estudiante se enfrenta ciertos retos como, por ejemplo, el aprendizaje del castellano, la adquisición de conocimientos matemáticos, físicos o químicos, históricos, el desarrollo del entendimiento y razonamiento. Para demostrar el nivel de conocimiento adquirido, los estudiantes se ven puestos a prueba con exámenes, trabajos de investigación o los deberes diarios para fortalecer dichos conocimientos.

Para consolidar la responsabilidad del centro, realizan un registro con una peculiar relación para almacenar que aulas están ocupadas en el centro y que cursos imparten en ellas, tanto así como los alumnos y profesores que realizan ese curso.

Uno de los aspectos que más podría preocupar al personal de un centro educativo, en lo referente a este tipo de aplicaciones es que se podría considerar que se está privando al centro la posibilidad desarrollar la responsabilidad de estructurarlo en papel y que esto afecte a su desarrollo. No obstante, esta aplicación debe ser considerada una herramienta de refuerzo, no una forma de sustitución de los métodos más tradicionales.

Por ello se ha desarrollado una aplicación con dos distintos inicios, cada una enfocada a un tipo distinto de usuario, para la aplicación se utiliza el sistema operativo Android¹ en su versión 5 o superior, la decisión de elegir esta tecnología es la continua evolución del lenguaje y la comunidad de desarrolladores, donde se puede encontrar gran cantidad de guías, documentación y librería para ayudar en el desarrollo de una aplicación, ya sean de fuentes oficiales o creadas por la extensa comunidad que hay. Por otra parte, Android cuenta con una gran tasa de implantación en el mercado que ronda el 84,1% mientras que para iOS sólo es un 15,9% en el año 2021², además se parte de una base de conocimientos, tanto de Java en su versión 1.8 como en la arquitectura de las aplicaciones Android.

https://www.android.com/intl/es_es/
[Cuota de mercado de los sistemas operativos para smartphone en España](#)

2. Objetivos

En este proyecto se afrontan una serie de retos, por un lado, diseñar y desarrollar una aplicación en Android, aunque se parte de una base de conocimientos en esta tecnología la investigación será necesaria para poderla llevar a cabo, por otro lado diseñar y desarrollar una base de datos robusta donde guardar toda la información del centro, los alumnos, los docentes, aunque se tienen conocimientos en la tecnología, la investigación será necesaria debido a la continua evolución que tiene la tecnología.

El objetivo global es crear la aplicación con dos distintos inicios, cada una enfocada a un grupo de usuarios, que forman el total de los usuarios finales.

2.1. Objetivos específicos

Con respecto a los desarrollos:

Desarrollar un código limpio, modular y reutilizable para ambas aplicaciones.

Hacer uso de las versiones más recientes de librerías y Frameworks.

Con respecto a la funcionalidad:

Desarrollar una aplicación Android en Kotlin para los administradores de un centro educativo que les permita: añadir, modificar, eliminar, consultar y asignar los respectivos alumnos, profesores, cursos y aulas. Y para el otro grupo de usuarios no tendrán acceso a cualquier mantenimiento que tenga que ver con la base de datos del centro ya sea añadir, modificar, eliminar, consultar y asignar alumnos, profesores, cursos y aulas, no obstante, si tendrán acceso a la información del centro y aulas y su geolocalización.

Desarrollo de pantallas básicas en dicha tecnología para añadir, modificar, eliminar, consultar y asignar los respectivos alumnos, profesores, curso y aulas.

Desarrollo de algoritmos para poder ver la geolocalización del centro.

Diseño e implementación de una base de datos común para dicha aplicación, donde se pueda guardar la información del centro educativo.

3. Estado del arte

A continuación, se va a realizar un análisis de aplicaciones actuales, tanto móviles como aplicaciones en línea, que se encuentran en el mercado, y que puedan tener similitud con el proyecto.

Posteriormente se expone una conclusión de cómo se debe afrontar el desarrollo de las aplicaciones para cubrir las posibles necesidades de los usuarios finales, de modo que ambas aplicaciones puedan ser diferenciadas del resto, añadiendo mejoras en la medida de lo posible.

3.1. Ítaca

ITACA⁴ (Innovación Tecnológica Administrativa de Centros y Alumnado) es una aplicación fomentada por la Consejería de Educación, Cultura y Deporte de la Comunidad Valenciana y que se puso en marcha en 2010.

Esta plataforma cuenta con dos aplicaciones, una en línea para que las familias puedan hacer un seguimiento de la evolución de sus hijas e hijos durante el curso académico, y una aplicación de escritorio que permite al centro educativo y a los docentes gestionar el curso académico.

Por una parte, la aplicación Web destinada a las familias permite consultar las notas de sus hijos e hijas, ver un registro de asistencia a clase, el calendario de evaluaciones y las diferentes actividades extraescolares que se puedan desempeñar en el centro. Además, permite el envío de mensajes al personal docente del centro.

Por otra parte, la aplicación de escritorio Ítaca 3 permite al personal de centros públicos y concertados del sistema educativo valenciano acceder a un sistema centralizado de información, todo usuario tiene un perfil que determina el nivel de acceso a la información del sistema. Aunque la información que hay sobre esta aplicación para los centros educativos es reducida, se puede intuir que el personal docente puede publicar notas de pruebas individuales, así como las evaluaciones finales por trimestre o por curso académico, también puede llevar un registro de asistencia a clase, incluidos los retrasos, y permite enviar y recibir mensajes de las familias. En cuanto a la administración permite gestionar el calendario de evaluaciones y actividades extraescolares que se deseen ofertar a las familias.

<https://portal.edu.gva.es/itaca/es/inicio>

3.2. Alexia

Alexia⁵ es una plataforma de gestión integral de alojamiento en la nube, que engloba tanto los aspectos administrativos, educativos como de comunicación, al cual se puede acceder tanto desde dispositivos móviles como desde cualquier navegador de un ordenador. Al estar fundamentada en la tecnología de almacenamiento en la nube permite el acceso a la información de forma permanente, siempre que se disponga de una conexión a internet. Además, cuenta con herramientas como Microsoft 365, Moodle o G Suite entre otras.

Los usuarios finales de esta aplicación son:

El personal del centro educativo, permitiendo una gestión administrativa como control de asistencia, la comunicación con las familias y la gestión de las evaluaciones.

Los docentes, permitiéndoles realizar un seguimiento más cercano del aprendizaje adquirido por los estudiantes, pudiendo orientarles mejor en el desarrollo de su crecimiento personal de cara a cursar estudios superiores o universitarios.

Las familias de los estudiantes del centro, quienes pueden tener un mejor seguimiento de la educación de sus hijas e hijos, comprobar la asistencia y consultar fechas de eventos o exámenes a lo largo del curso académico.

3.3. Additio App

Additio App⁶ es un sistema de gestión de centros educativos, con dos aplicaciones, una enfocada a la gestión de horarios a lo largo de un curso académico, que permite a un docente tener un control de asistencia de los alumnos y hacer un seguimiento de las calificaciones que los estudiantes puedan tener en las pruebas a las que se enfrenten, y otra que permite la gestión administrativa de las clases de un curso académico y la comunicación, fomentando el aprendizaje autodidacta de los estudiantes y la participación de las familias en el desarrollo de sus hijas e hijos.

Los usuarios finales de esta aplicación son:

El personal del centro.

El personal docente.

3.4. Gescola

Gescola⁷ es un sistema de gestión, con versión móvil y versión en línea, que permite tener un control de asistencia y comportamiento de los estudiantes, también permite realizar el envío de notificaciones a las familias, mediante la mensajería interna de la que dispone, con el envío por copia mediante un correo electrónico o por SMS. Además, permite la gestión de evaluaciones, horarios y tutorías, las salidas que se puedan realizar en forma de excursiones, incluyendo el pago de estas a través de la aplicación, incidencias del material del centro y la gestión de préstamos de la biblioteca de la que disponga el centro.

Los usuarios finales de esta aplicación son:

El personal del centro.

El personal docente.

Las familias de los estudiantes del centro.

3.5. Conclusiones

Como se puede apreciar ya existen aplicaciones que ofrecen una gran cantidad de servicios, muy parecidas a la que se plantea para este proyecto, para poder generar una diferenciación con estas, y con las demás que se encuentran en el mercado, se plantea crear interfaces más amigables para ambas aplicaciones.

<https://educaria.com.ar/alexia/>
<https://www.additioapp.com/es>

4. Metodología

Un desarrollo de calidad ayuda en la reducción de costes del desarrollo y permite adaptarse de una forma ágil a la hora de afrontar cambios en las especificaciones o si se detecta algún tipo de error en el sistema.

La metodología debe ajustarse a las características del diseño y del desarrollo del proyecto, en este caso se opta por realizar un desarrollo ágil que permita poder apreciar la evolución de este, permitiendo a la vez que los requisitos evolucionen según las necesidades que puedan darse en la fase de desarrollo.

Entre las más destacables metodologías ágiles tenemos Scrum y Kanban, pero dado que el proyecto está desarrollado por una sola persona y en Scrum se deben establecer roles de distintos tipos, entre los diferentes miembros del equipo de desarrollo, y se deben realizar Scrum diario o Daily Standup de forma diaria para tratar cualquier tema relacionado con las tareas que se estén realizando, se opta por la utilización de la metodología Kanban.

Kanban es un sistema de gestión de proceso visual, que permite diseñar tareas cortas a lo largo del desarrollo de cualquier tipo de proyecto, estas tareas se representan con tarjetas que representan elementos específicos e independientes del desarrollo, lo que permite adecuar el cambio incremental y evolutivo.

4.1. Sistema de control de versiones.

Para el sistema de control de versiones se ha utilizado Git⁹, mediante el uso de diferentes ramas se ha podido realizar desarrollos ordenados e independientes con el uso de varias ramas: “evo_administrador”, “evo_usuarios”, una vez finalizado cada desarrollo de forma independiente se ha usado una rama de desarrollo, que ha sido nombrada como “develop”, para realizar las pruebas de aceptación, y una vez finalizadas se ha usado la rama maestra, que recibe el nombre de “master”, para finalmente subir el desarrollo a GitHub¹⁰. Esta práctica se ha realizado para ambas aplicaciones, manteniendo las ramas en desarrollo en el entorno local y actualizando la versión final o estable en el repositorio en línea.

Las ramas de los distintos desarrollos se crean desde la rama maestra, para asegurar que el código desde el que se inicia el desarrollo es estable y ha sido previamente validado. En caso de detectarse un error en la rama maestra, se ha creado una rama desde esta para subsanar dicho error y, tras validar dicha corrección en la rama de desarrollo finalmente se guardan los cambios en la rama maestra.

En la figura 4.4 se puede observar cómo se crearían las ramas de distintos evolutivos y como cada uno sería independiente del resto.

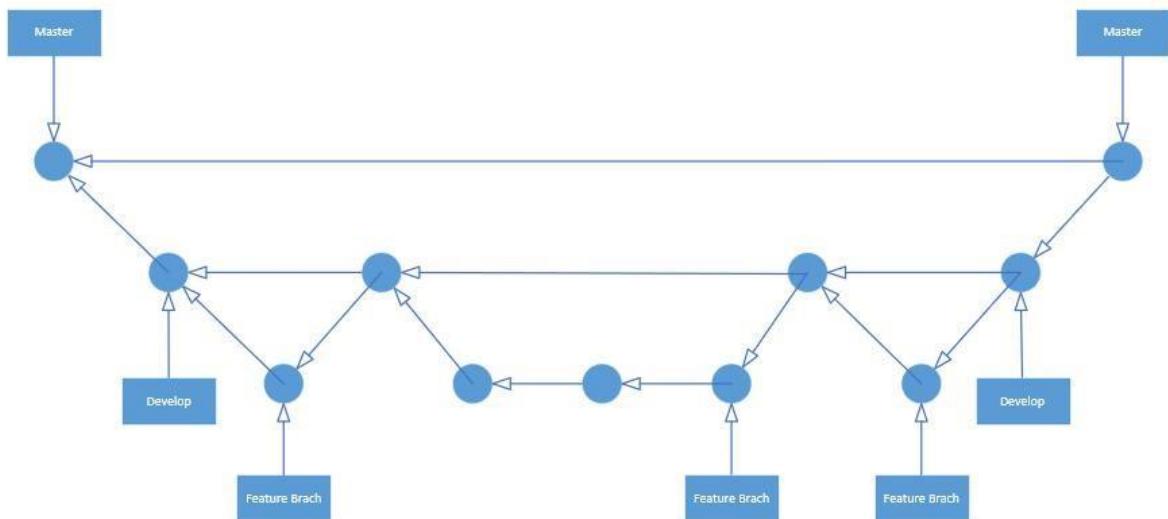


Ilustración 1: Creaciones y fusión de ramas

5. Tecnología

En esta sección se presentan las tecnologías y aplicaciones usadas durante la fase de diseño y desarrollo de ambas aplicaciones que forman parte del proyecto. En primer lugar, se describen las aplicaciones usadas en la fase de diseño, en segundo lugar, se profundizará en las que solo afectan a la aplicación Android, seguida de las que se han usado en la implementación de la aplicación Laravel y, para finalizar, se hará hincapié en las que comparten ambas aplicaciones.

5.1. Aplicaciones de diseño

En la fase de diseño, donde se han realizado los prototipos de las aplicaciones, el diseño de la base de datos, el diagrama de clases y los diagramas de casos de uso, se ha optado por utilizar herramientas que han sido utilizadas a lo largo del Grado. A continuación, se presenta cada una de ellas.

5.1.1. Balsamiq Wireframes

Balsamiq Wireframes¹¹ es una potente herramienta destinada a la creación de prototipos o bocetos, utilizada habitualmente para el diseño temprano de aplicaciones móviles y para páginas webs, dispone tanto de versión de escritorio como versión en línea.

Estos bocetos proporcionan a un desarrollador, o a un equipo de desarrolladores, una visión global de cómo será el producto final esperado, lo que permite tener una mayor agilidad a la hora de dividir las posibles tareas a realizar, y llevarlas a cabo.

La aplicación también permite que estos prototipos puedan usarse para realizar un estudio de aceptación y realizar algunas correcciones previas al inicio del desarrollo. Este estudio podría hacerse sobre un proyecto creado con Balsamiq, ya que se permite configurar los diferentes elementos de cada boceto para que interactúen con el resto de los que tenga el proyecto y, de forma dinámica, el grupo objetivo del estudio puede valorar la usabilidad de lo que acabará siendo el producto final.

<https://balsamiq.com/>
<https://git-scm.com/>
<https://github.com/>

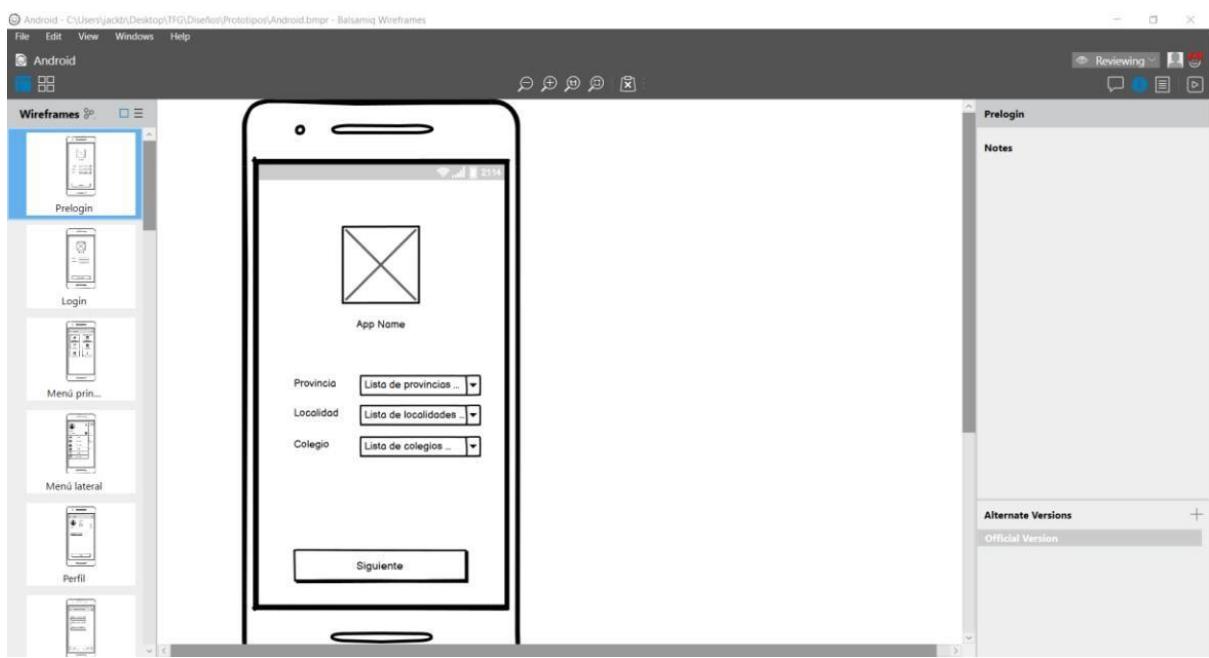


Ilustración 2: Proyecto Android.

5.1.2. Draw.io

Draw.io¹² es una aplicación, con versión de escritorio y versión en línea, que permite diseñar diagramas de forma sencilla y rápida, ya que dispone de una amplia galería de figuras, también permite la posibilidad de poder realizar diseños de forma colaborativa, además la versión en línea permite almacenar los diseños en distintos servicios de almacenamiento de archivos.

Para el desarrollo del proyecto se ha utilizado esta aplicación, en su versión de escritorio, para diseñar las tablas de la base de datos de ambas aplicaciones.

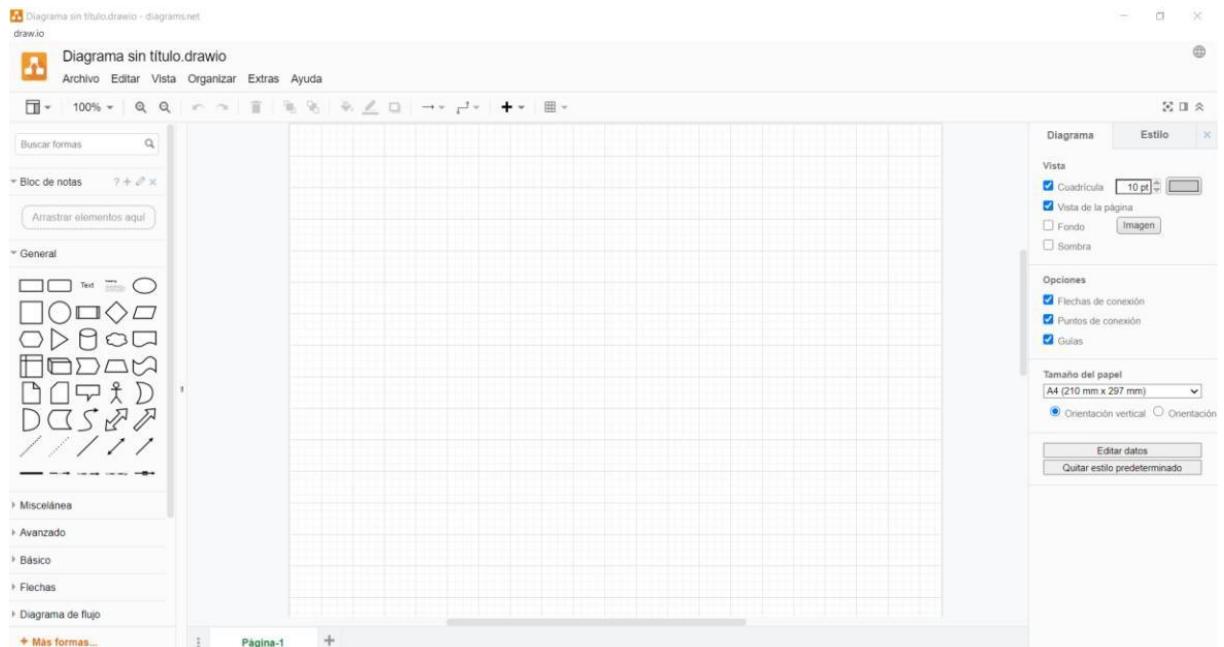


Ilustración 3: Herramienta de diagramas Draw.io para escritorio.

5.1.3. Visio Profesional

Visio 2016 es una herramienta desarrollada por Microsoft¹³ cuyo uso está destinado para dibujar una amplia variedad de diagramas. Entre esto podemos encontrar diagramas de clases, diagramas de flujo, diagramas de casos de uso, diagramas de procesos, que están más enfocados al diseño de un proyecto Software. Además, permite diseñar diagramas usados en otros ámbitos como planos de construcción o planos de planta para la arquitectura.

Para el diseño de las aplicaciones del proyecto se ha optado por usar esta aplicación dado que permite realizar un diseño más profesional que el que podría dar Draw.io, que se ha usado para hacer sólo un diagrama más específico.

Para usar la aplicación se necesita obtener una licencia, para el caso, la licencia utilizada es la que proporciona el servicio OnTheHub¹⁴, gracias al convenio entre la Escuela Politécnica de la Universidad de Alicante y la compañía Microsoft.

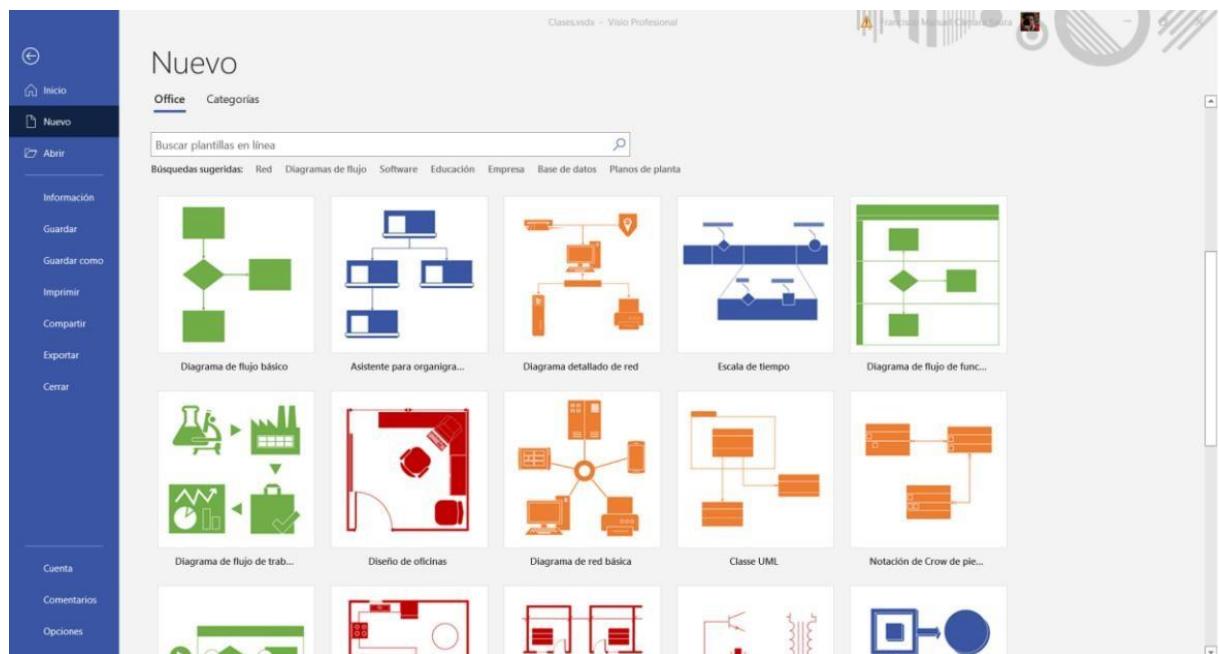


Ilustración 4: Visio Profesional.

5.2. Aplicación Android

A la hora de plantear este proyecto, y las tecnologías que se usarían en el desarrollo del producto final, la elección de Android podría considerarse como un prerequisito, ya que había trabajado previamente con esta tecnología y había experimentado la potencia y versatilidad que presenta.

Android es una tecnología joven y en constante evolución, con millones de usuarios en todo el mundo, y con la intención de mejorar y reforzar los conocimientos de esta tecnología la mejor forma era mediante un proyecto de estas dimensiones.

El desarrollo se lleva a cabo en el sistema operativo Windows 10, se crea el escenario perfecto para simular un entorno de producción, teniendo que realizar las pruebas a través de una red previamente configurada.

A continuación, se procede a explicar el entorno de programación utilizado para el desarrollo, el lenguaje de programación empleado en él, y las distintas librerías, componentes específicos y Frameworks utilizado a lo largo de la implementación de la aplicación Android.

5.2.1. Android Studio

Como entorno de programación se ha utilizado el IDE (Integrated Development Environment, es castellano, Entorno de Desarrollo Integrado) Android Studio¹⁵, en la versión Artic Fox 2020.3.1, ya que se trata del entorno oficial, desarrollado y gestionado por Google, para el desarrollo de aplicaciones para dispositivos con el sistema operativo Android.

Android está basado en el IDE IntelliJ IDEA¹⁶, uno de los entornos más utilizados para el desarrollo de aplicaciones Java. IntelliJ IDEA incluye una amplia variedad de herramientas que sirven de ayuda a la hora de realizar la implementación del código de un

<https://developer.android.com/studio>
<https://www.jetbrains.com/es-es/lp/idea-extended-trial/>

proyecto. Además, el entorno es personalizable, por lo que se puede modificar en función de las necesidades de un desarrollador.

Google modifica el entorno de IntelliJ IDEA para añadirle herramientas adicionales como la ejecución de una aplicación sin necesidad de un dispositivo físico gracias a los emuladores que proporciona, el diseño de vistas que formarán la interfaz de la aplicación, el explorador de archivos y un gestor de la base de datos de la aplicación que se esté testando, ya sea en un dispositivo físico o en uno emulado.

También, el propio entorno permite la vinculación de proyectos con gestores de repositorios, como podría ser GitHub, otra característica remarcable podría ser la posibilidad de firmar una aplicación, esto se hace de cara a ponerla en producción, permitiendo que Google pueda comprobar que la persona que la pública es el desarrollador, un grupo de desarrolladores o una empresa, certificando así que la aplicación es legítima.

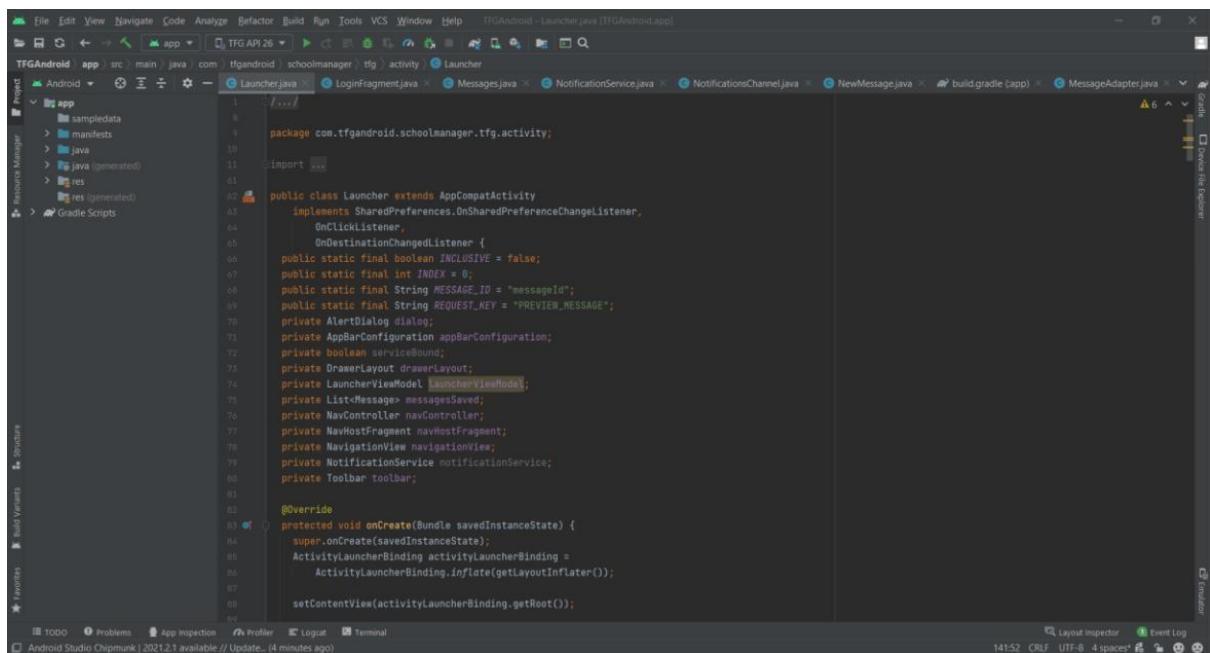


Ilustración 5: Entorno de Desarrollo Integrado Android Studio.

5.2.2. Kotlin/Android

El desarrollo de la aplicación se ha realizado utilizando el lenguaje Kotlin, en su versión 8, la cual se usa para el sistema operativo Android desde la versión 5. Android es el sistema operativo más utilizado en la actualidad, alrededor de 3.000 millones de usuarios con dispositivos activos en 2021, aunque realmente los informes al respecto varían en el porcentaje de usabilidad, en todos coinciden en esta aproximación.

Para el desarrollo del proyecto se ha utilizado la versión de Android 7.0 “Nougat” o superior, ya que permite llegar al 91,7% de los usuarios de este sistema operativo, tal y como se muestra a continuación en la figura obtenida de Android Studio.

El sistema operativo Android tiene una extensa documentación, entre la oficial¹⁷ y la amplia comunidad de la que dispone, permitiendo a cualquier desarrollador sin mucho conocimiento en esta tecnología desarrollar una aplicación funcional.

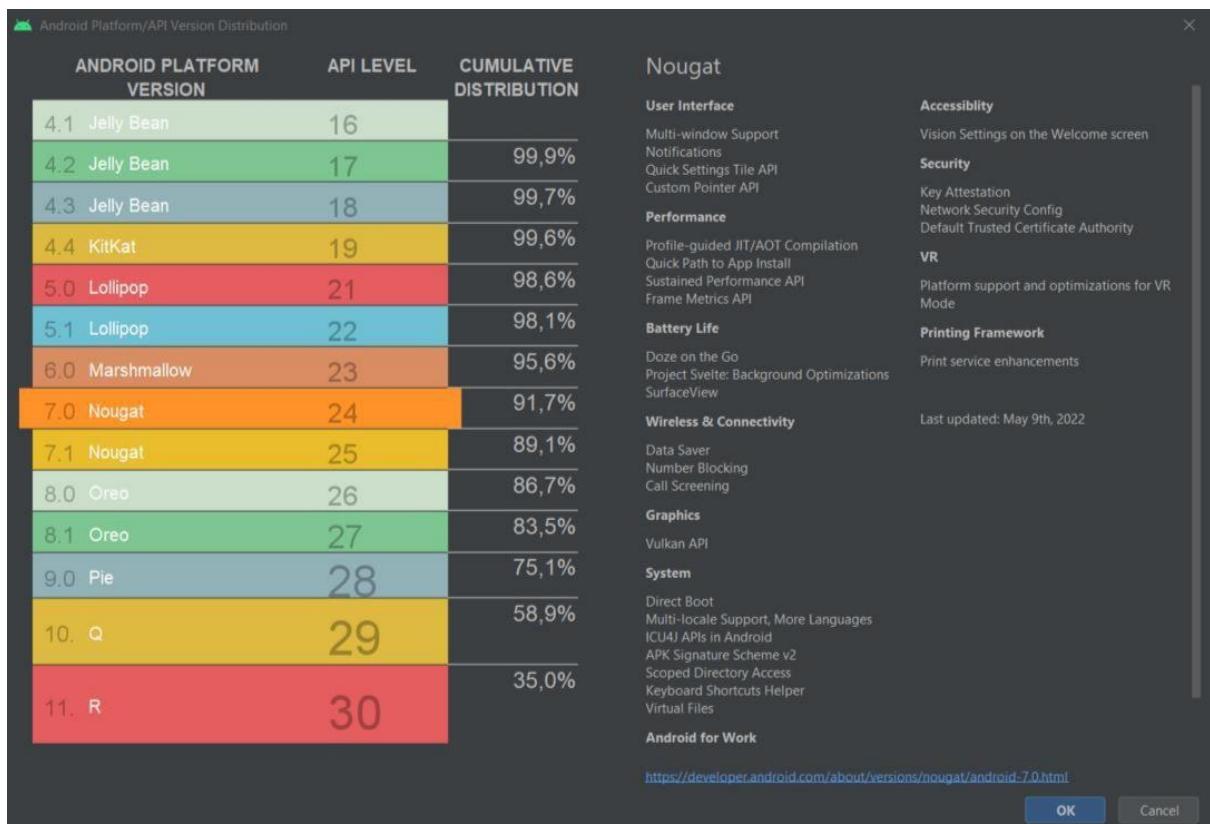


Ilustración 6: Distribución acumulada por versión de Android.

<https://developer.android.com/docs>

5.2.3. Data Binding

Data Binding²¹ es otra de las librerías de Android Jetpack, esta permite vincular los componentes de la interfaz de usuario diseñada en Android Studio con la lógica de negocio de la implementación, donde se establece el comportamiento de las pantallas que verá el usuario al usar la aplicación, esto se hace mediante un formato declarativo. Esto permite a un desarrollador ser más ágil a la hora de desarrollar una aplicación.

5.2.4. GitHub

Tras varios años usando este repositorio en diferentes asignaturas, parece una obviedad por qué se ha optado por su utilización, GitHub es un repositorio de sencillo manejo, con una gran comunidad que proporciona ayuda en forma de documentación, guías o soporte de dudas, con el podemos mantener copias de seguridad de cualquier proyecto de forma remota, ya sea un desarrollo individual o se trate de desarrollos colaborativos. Además, GitHub proporciona la posibilidad de crear repositorios privados ilimitados de forma gratuita.

Para el desarrollo de ambas aplicaciones del proyecto se han creado dos repositorios de forma independiente, en cada uno se encuentra el código fuente de la aplicación respectiva.

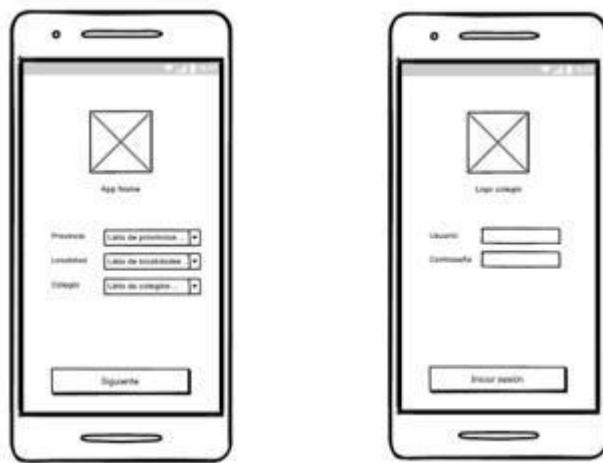
<https://developer.android.com/jetpack/androidx/releases/databinding?hl=es-419>

6. Diseño

Para realizar el diseño temprano se ha utilizado la aplicación Balsamiq Wireframes, en su versión de escritorio. Estos diseños de las pantallas permiten tener una visión genérica de las interfaces de ambas aplicaciones, lo que permite ver un boceto de cómo serán las aplicaciones finales y así analizar los requisitos de cada una de ellas.

6.1. Aplicación Android

Al iniciar la aplicación, un tutor legal se encontrará con las vistas previa autenticación y autenticación, donde deberá seleccionar un centro de estudios indicando provincia y población y a continuación deberá identificarse en el sistema.



Una vez identificado, y si los valores introducidos son correctos, el usuario accederá al resto de funcionalidades de la aplicación, que son accesibles desde el menú principal. Además, mediante un menú lateral se podrá acceder a una serie de secciones que sólo se puede acceder desde aquí.

Por un lado, tenemos el perfil, donde el usuario podrá modificar distintos datos, según proceda.

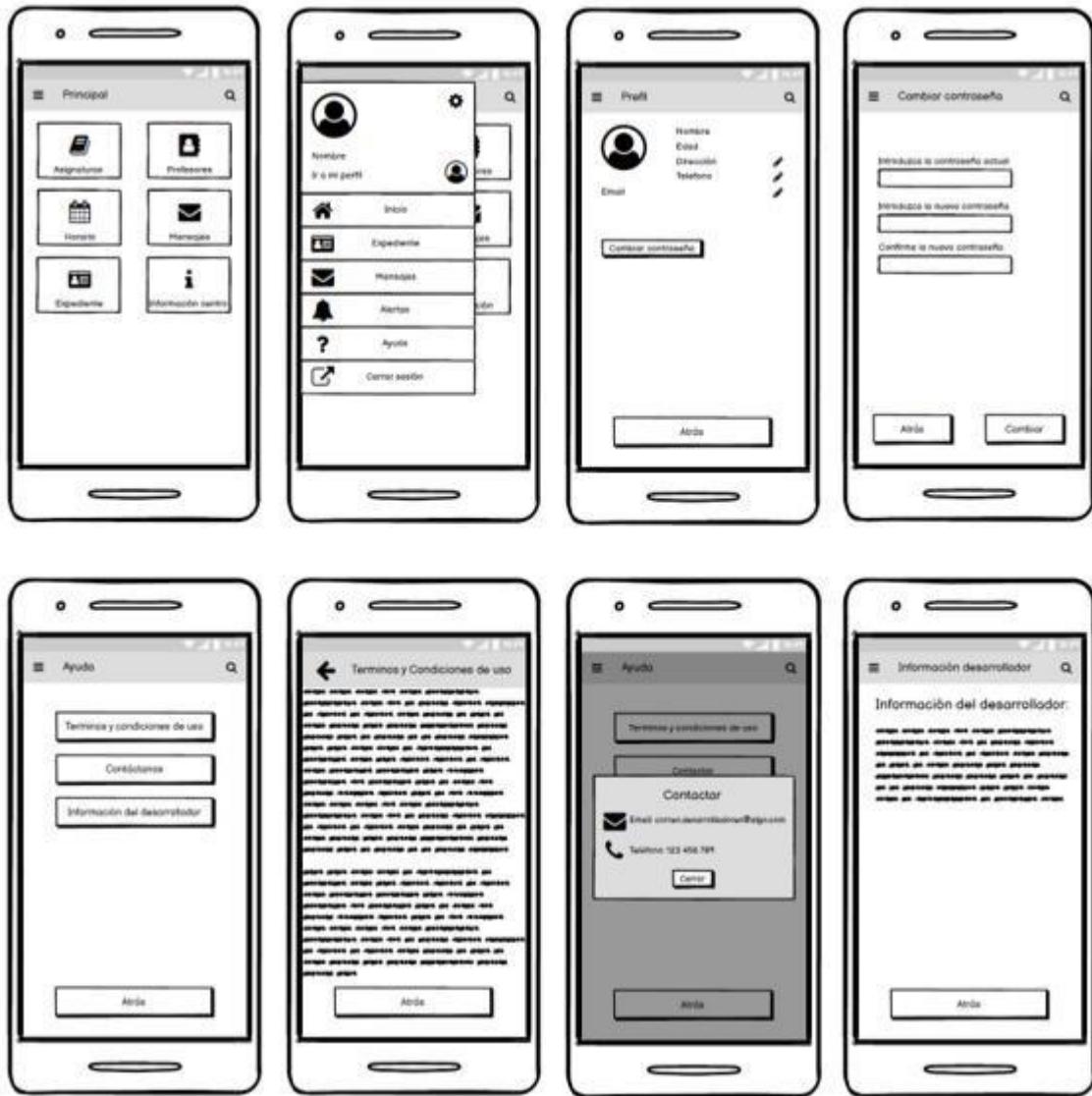


Ilustración 7: Menú principal y menú lateral con opciones.

Un usuario, si es administrador desde el menú puede añadir, modificar, eliminar, consultar y asignar tanto cursos, como alumnos, profesores y aulas. En la sección de cursos solo puedes asignar aulas al curso por sus IDs correspondientes. En la sección de profesores puedes asignar cursos al profesor por sus IDs correspondientes. En la sección de alumnos puedes asignar cursos y profesores al alumno por sus respectivos IDs. Aparte de tener secciones en el menú para consultar información del centro y su geolocalización, tanto así que hay un apartado de un listado de aulas que puedes apreciar al presionar sobre ellas un video de como son las aulas y sus instalaciones con un listado de sus materiales.

6.2. Estructuración Aplicación Android

6.2.1. Entidad-relación

En primer lugar, tenemos que un centro tiene unas aulas donde tiene muchos pares de cursos que sólo pertenecen a un centro, para cada curso tenemos un profesor, pero un profesor sólo pertenece a un curso y para cada profesor hay alumnos que sólo pertenecen a este profesor.

Para cada alumno tenemos un profesor, que puede ser profesor de varios alumnos, estos profesores y un curso que puede ser cursado por varios alumnos.

Para cada profesor tenemos un curso, que puede ser impartido por un solo profesor.

Para cada curso tenemos un aula, que puede ser realizado en una sola aula.

Por último, el centro también, tiene personal administrativo que, al igual que los profesores y alumno, forman parte de la generalización de la tabla usuario, por lo que son los únicos que pueden modificar dicha base de datos.

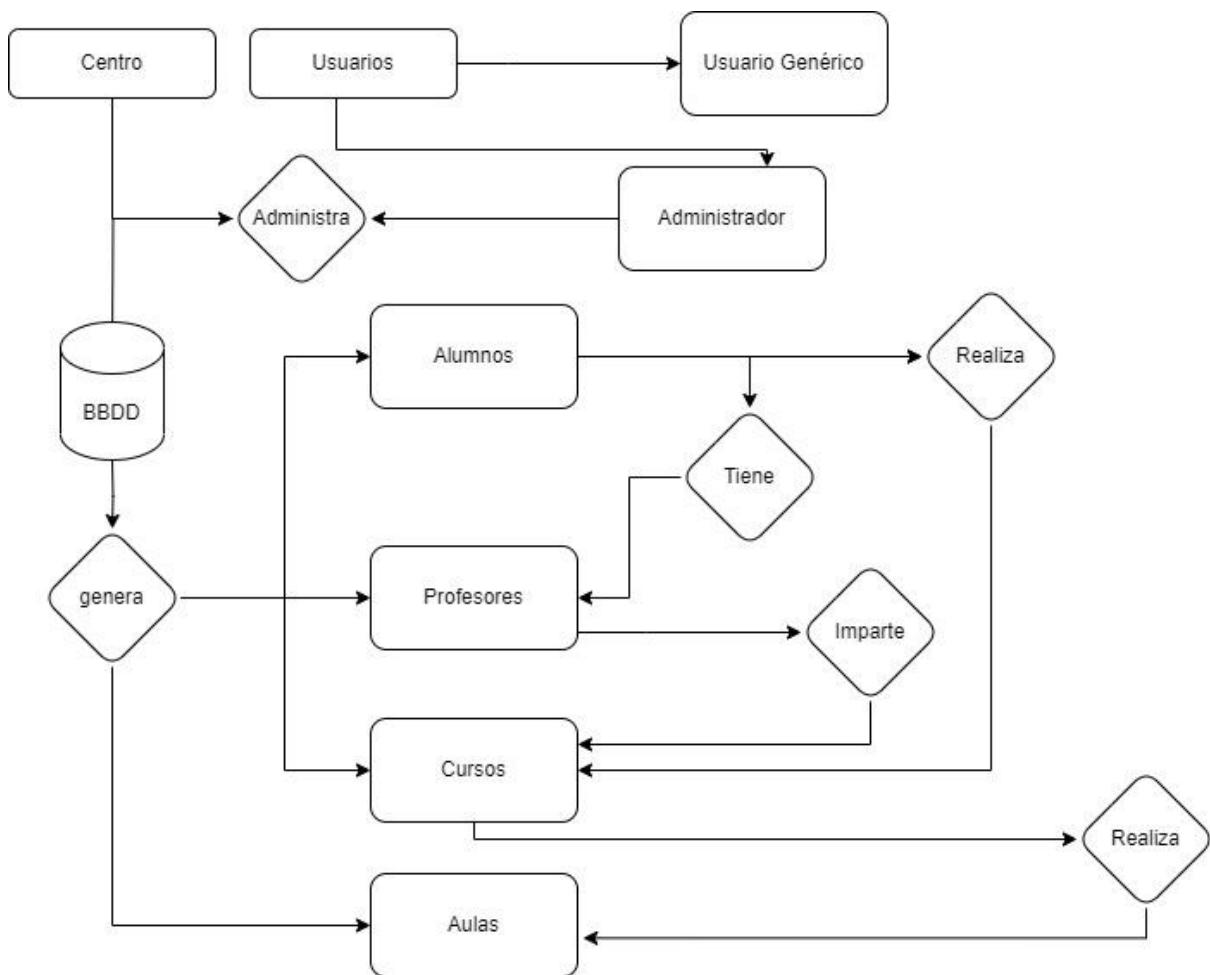


Ilustración 8: Diagrama de base de datos.

6.2.2. Modelo relacional

Para simplificar este punto, se exponen las tablas de forma independiente, para cada tabla se detallan los campos que contienen y se añade la representación relacional correspondiente.

6.2.2.1. Aulas

Un aula tiene un identificador que lo define como único, también tiene un nombre.

```
val createTableAulas = "CREATE TABLE $TABLA_AULAS" +  
    "($COLUMN_ID_AULAS INTEGER PRIMARY KEY AUTOINCREMENT, " +  
    "$COLUMN_AULAS_AULAS TEXT)"
```

Ilustración 9: Tabla Aulas

6.2.2.2. Usuario

Un usuario es una generalización de usuarios, esto se hace para evitar repetir campos idénticos en diferentes tablas, los atributos de esta tabla serían un identificador único que lo define, nombre o Nick de usuario y una contraseña.

```
val createTableUsers = "CREATE TABLE $TABLA_USERS" +  
    "($COLUMN_ID_USERS INTEGER PRIMARY KEY AUTOINCREMENT, " +  
    "$COLUMN_NOMBRE_USERS TEXT, " +  
    "$COLUMN_PASSWORD_USERS TEXT)"
```

Ilustración 10: Tabla Users

6.2.2.3. Cursos

Un curso, es un conjunto de módulos o asignaturas que deben impartirse en el centro, esta tabla tiene atributos como un identificador que lo define como único, un nombre, un código el cual se identifica el centro y una referencia de la tabla aula para que ese curso se asigne a un aula y pueda impartirse.

```
val createTableCursos = "CREATE TABLE $TABLA_CURSOS" +  
    "($COLUMN_ID_CURSO INTEGER PRIMARY KEY AUTOINCREMENT, " +  
    "$COLUMN_NOMBRE_CURSO TEXT, " +  
    "$COLUMN_CODIGO_CURSO TEXT, " +|  
    "$COLUMN_ID_AULAS INTEGER, " +  
    "FOREIGN KEY($COLUMN_ID_AULAS) REFERENCES $TABLA_AULAS($COLUMN_ID_AULAS))"
```

Ilustración 11: Tabla Cursos

6.2.2.4. Profesor

Un profesor tiene un identificador único que lo diferencia, además de un nombre y apellidos, aparte también una referencia de la tabla curso para que dicho profesor se le asigne un curso al que pueda impartir.

```
val createTableProfesores = "CREATE TABLE $TABLA_PROFESORES" +  
    "($COLUMN_ID_PROFESOR INTEGER PRIMARY KEY AUTOINCREMENT, " +  
    "$COLUMN_NOMBRE_PROFESOR TEXT, " +  
    "$COLUMN_APELLIDOS_PROFESOR TEXT, " +  
    "$COLUMN_ID_CURSO INTEGER, " +  
    "FOREIGN KEY($COLUMN_ID_CURSO) REFERENCES $TABLA_CURSOS($COLUMN_ID_CURSO))"
```

Ilustración 12: Tabla Profesor

6.2.2.5. Alumno

Un alumno tiene como identificador único, un nombre y apellidos, dos referencias, la primera referencia es de la tabla profesores; donde se le asignara un profesor al alumno para que pueda realizar la enseñanza y la segunda referencia es la tabla cursos puesto que se le asigna un curso que ha de realizar.

```
val createTableAlumnos = "CREATE TABLE $TABLA_ALUMNOS " +  
    "[$COLUMNNA_ID_ALUMNO INTEGER PRIMARY KEY AUTOINCREMENT, " +  
    "$COLUMNNA_NOMBRE_ALUMNO TEXT, " +  
    "$COLUMNNA_APELLIDOS_ALUMNO TEXT, " +  
    "$COLUMNNA_ID_PROFESOR INTEGER, " +  
    "$COLUMNNA_ID_CURSO INTEGER, " +  
    "FOREIGN KEY($COLUMNNA_ID_PROFESOR) REFERENCES $TABLA_PROFESORES($COLUMNNA_ID_PROFESOR), " +  
    "FOREIGN KEY($COLUMNNA_ID_CURSO) REFERENCES $TABLA_CURSOS($COLUMNNA_ID_CURSO))"
```

Ilustración 13: Tabla Alumnos

6.3. Diagrama de clases

Un diagrama de clases UML (Unified Modeling Language, en castellano Lenguaje Unificado de Modelado) describe la estructura de un sistema, mostrando las clases que lo componen, sus atributos, operaciones y la relación entre estos objetos, como los atributos son los que se han expuesto en el apartado anterior, se omiten de las tablas del diagrama de clases para mejorar su comprensión.

Para la aplicación de este proyecto el diagrama de clases es el mismo, aunque la implementación se realiza en Kotlin.

El diagrama de clases es una adaptación del diagrama de base de datos, que mantiene las mismas restricciones que tiene la base de datos expuesta en el apartado anterior, y aporta un valor positivo a un desarrollador o desarrolladores de un proyecto, permitiendo tener una mayor visión del desarrollo global.

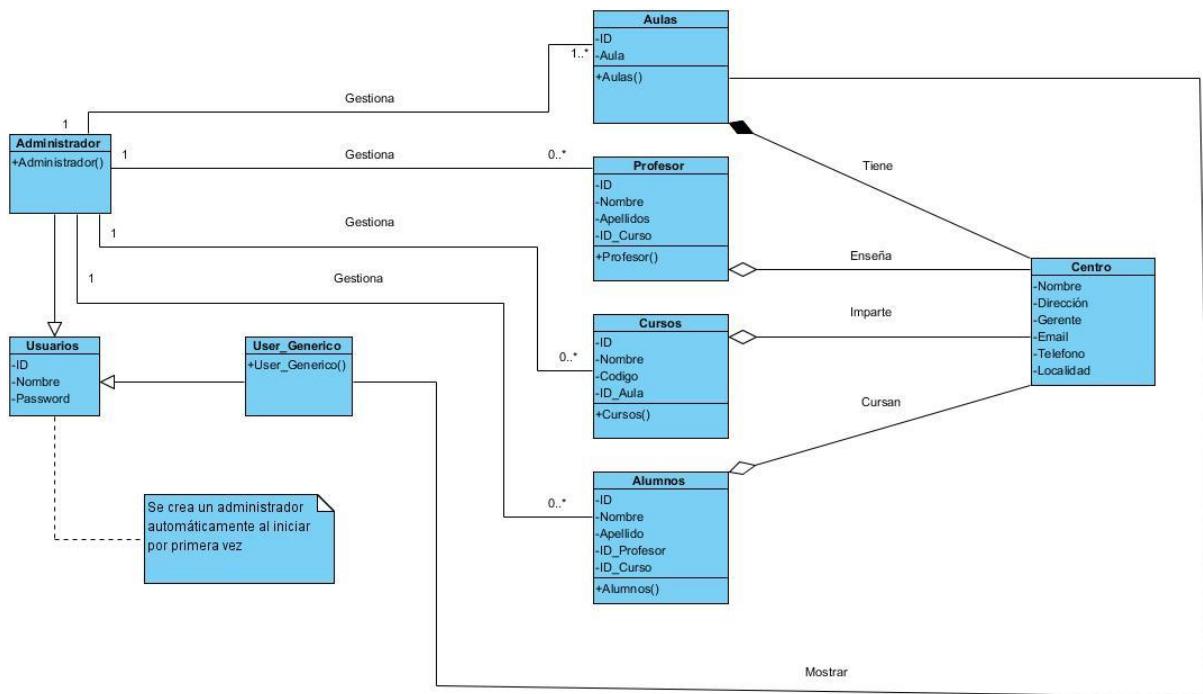


Ilustración 14: Diagrama de clases.

6.4. Diagramas de casos de uso

Un diagrama de clases es otro tipo de UML que permite representar de forma gráfica las funcionalidades para cada usuario de una aplicación, estas funcionalidades son el resultado de la captura de requisitos realizada durante el análisis del proyecto. Por una parte, se abordan las funcionalidades del administrador en la aplicación Android y, por otra parte, se abordan las funcionalidades de los demás usuarios en la aplicación Android.

6.4.1. Administrador

Se han diseñado diversas funcionalidades con las que el administrador pueda realizar dichas acciones.

Una vez autenticado en la aplicación, el administrador podrá:

Añadir a un alumno en la base de datos.

Modificar a un alumno su información en la base de datos.

Eliminar a un alumno en la base de datos.

Añadir un curso en la base de datos.

Modificar la información de un curso en la base de datos.

Eliminar un curso en la base de datos.

Añadir un profesor a la base de datos.

Modificar la información de un profesor en la base de datos.

Eliminar un profesor de la base de datos.

Añadir un aula a la base de datos.

Modificar la información de la base de datos.

Eliminar un aula de la base de datos.

Consultar listado de Alumnos.

Consultar listado de Cursos.

Consultar listado de Profesores.

Consultar listado de Aulas.

Registrar un nuevo usuario.

Mostrar información del centro, geolocalización y diseño de las diferentes aulas.

Asignar cursos a alumnos.

Asignar profesores a alumnos.

Asignar cursos a profesores.

Asignar aulas a cursos.

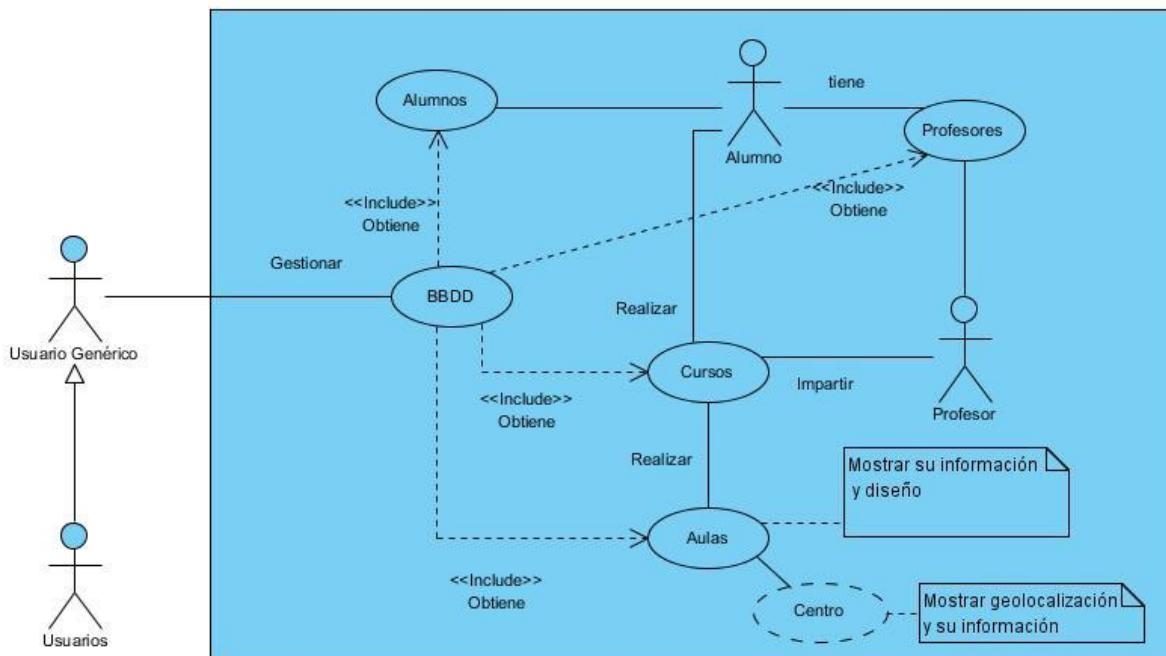


Ilustración 15: Diagrama Caso de Uso Para El Administrador

6.4.2. Otros usuarios

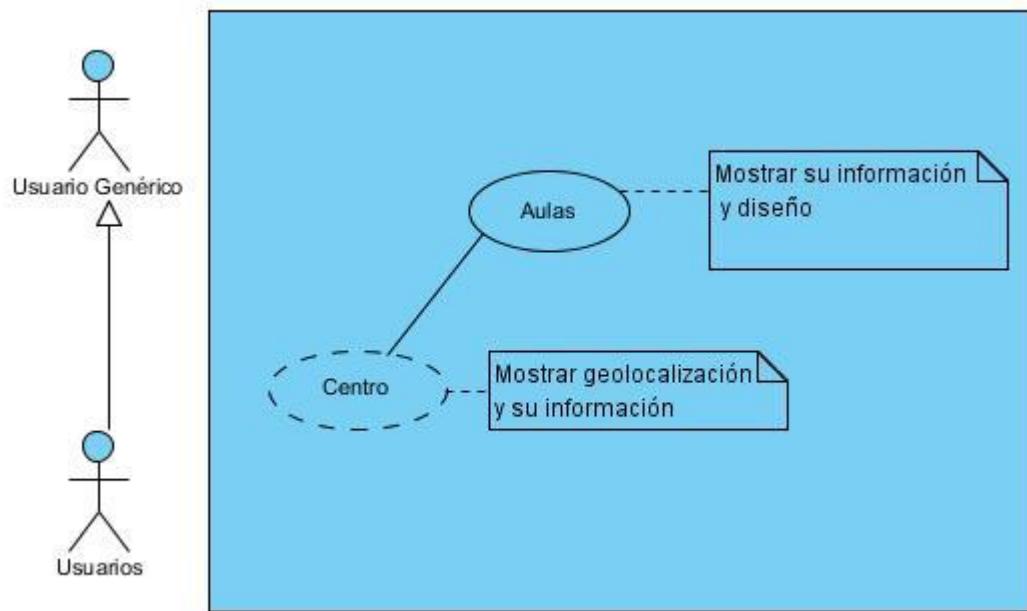
Los demás usuarios que no son administradores se han diseñado una serie de funciones con las que dichos usuarios puedan realizar en la aplicación:

Una vez autenticado en la aplicación, el usuario podrá:

Mostrar información del centro.

Mostrar información y diseño de las diferentes aulas que hay en el centro.

Mostrar la geolocalización del centro.



7. Arquitectura

A la hora de realizar el desarrollo de un proyecto uno de los aspectos más importantes es la arquitectura de los datos de la aplicación, esto es importante ya que permite a un desarrollador o equipo de desarrolladores ser más ágiles a la hora de realizar los desarrollos que se vayan planteando.

Además, puede suponer un ahorro de tiempo del desarrollo, evita los posibles errores que podrían surgir de no haber una arquitectura claramente definida, posibilita una más fácil refactorización, si en un futuro fuese necesaria, y ayuda a evitar cualquier duplicidad de código.

A continuación, se exponen las arquitecturas elegidas para cada una de las aplicaciones. Por una parte, la aplicación Android, que no tiene una clara arquitectura definida, lo cual da una mayor libertad a los desarrolladores para usar la que más les haga sentir cómodos. Por otra parte, la aplicación Laravel, que tiene una arquitectura definida en el Framework utilizado, lo que condiciona su uso.

7.1. Android

Dado que para Android no hay una arquitectura predefinida, la toma de decisión a la hora de elegir alguna de las que existen actualmente no es sencilla, se debe valorar que separe el modelo de la vista, para facilitar un desarrollo limpio; también, se tiene que tener en cuenta que el modelo debe abstraerse de realizar las consultas a la API REST alojada en el servidor y que la gestión de los datos obtenidos de la base de datos proporcionados por la API REST deben ser tratados de forma independiente al control de la vista que hace el modelo.

Actualmente la arquitectura que engloba estas condiciones es MVVM (Model-View-ViewModel, en castellano Modelo-Vista-Modelo de Vista), por ello es la elegida para el desarrollo.

Los elementos de esta arquitectura son:

- I. Modelo: En el que gracias a la librería de ROOM se realiza el acceso a la base de datos de la aplicación móvil a través de interfaces DAO.
- II. Vista: Se trata de los Fragments (en castellano Fragmentos) que pueda tener la aplicación, no debe confundirse con un archivo XML donde está el diseño de una pantalla. Un Fragment es donde se implementa toda la lógica de negocio de la interfaz e interactúa de forma directa y asíncrona, mediante objetos LiveData, con el ViewModel.
- III. Modelo de la vista: un ViewModel está directamente vinculado a un Fragment y es quien tiene la responsabilidad de procesar las peticiones realizadas desde la vista y devolver los datos obtenidos a la vista que los haya solicitado.
- IV. Repositorio: Los ViewModels de cada vista realizan peticiones al repositorio para obtener los datos que han sido solicitados desde la vista, ya sea de la base de datos de la aplicación móvil o al servidor mediante la API REST.
- V. Fuente de datos remota: Donde gracias a la librería de Retrofit se hace una petición a la API REST.

En la siguiente figura se puede observar cómo se estructuran estos elementos expuestos y así tener una mejor comprensión de cómo funciona la arquitectura.

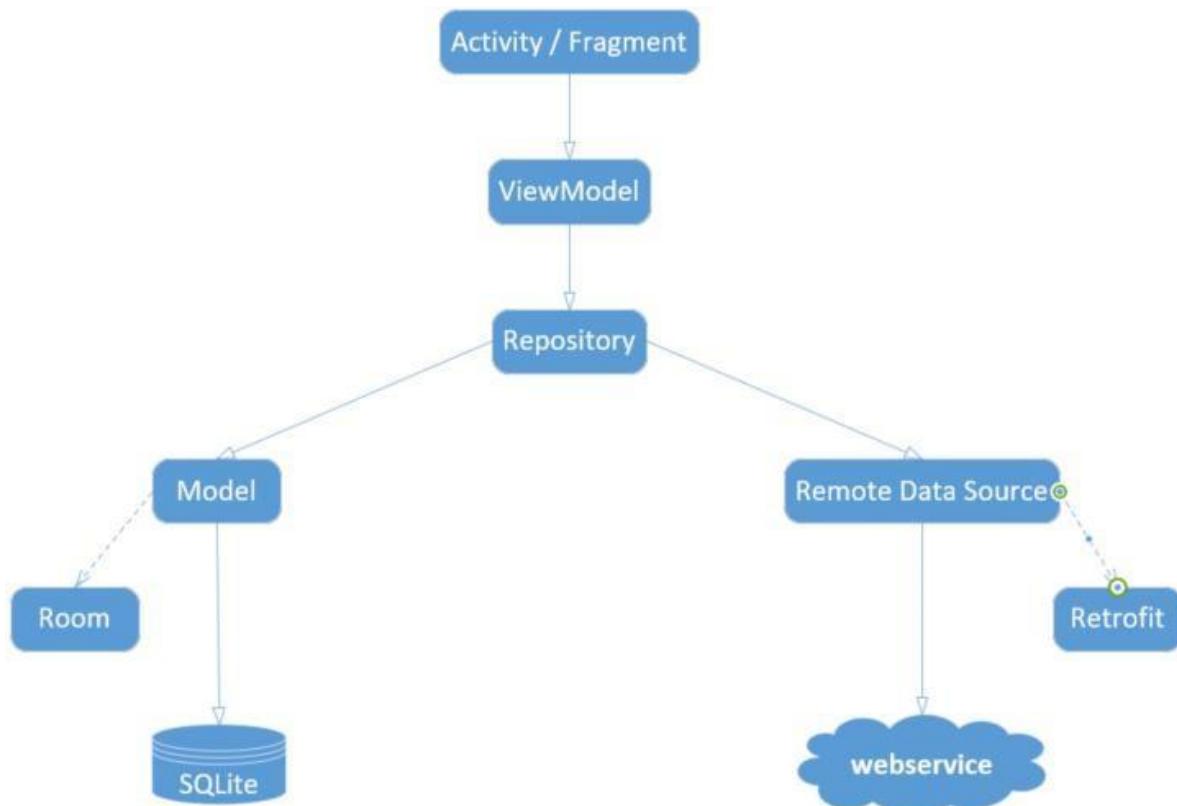


Ilustración 16: Arquitectura Model-View-ViewModel.

8. Implementación

A continuación, se procede a explicar los componentes más específicos del desarrollo llevado a cabo en la implementación de la aplicación, de forma independiente, como forma de refuerzo visual, en cada apartado se han incluido figuras o listas que ayudan a la comprensión del trabajo realizado y de la descripción realizada.

8.1. Android

Dado que Android no tiene establecido ningún Framework, hay una mayor libertad a la hora de realizar un desarrollo, permitiendo adaptarse a cualesquiera que sean las circunstancias que vayan surgiendo.

8.1.1. Interfaz

Uno de los puntos más importantes a la hora de realizar un desarrollo es pensando en la usabilidad, pero también en el aspecto amigable del producto final, por este motivo la interfaz debe ser sencilla e intuitiva.

Para el diseño e implementación de estas interfaces se ha utilizado la herramienta de diseño que incluye Android Studio, este editor gráfico permite añadir elementos de forma sencilla e intuitiva, aunque también permite la codificación del código XML resultante de generar la vista. Posteriormente esta interfaz se carga en un Fragment con la utilización de Data Binding, donde se incluye la funcionalidad de cada elemento.

En las siguientes figuras se pueden observar ejemplos del diseño gráfico y del diseño mediante el archivo XML.

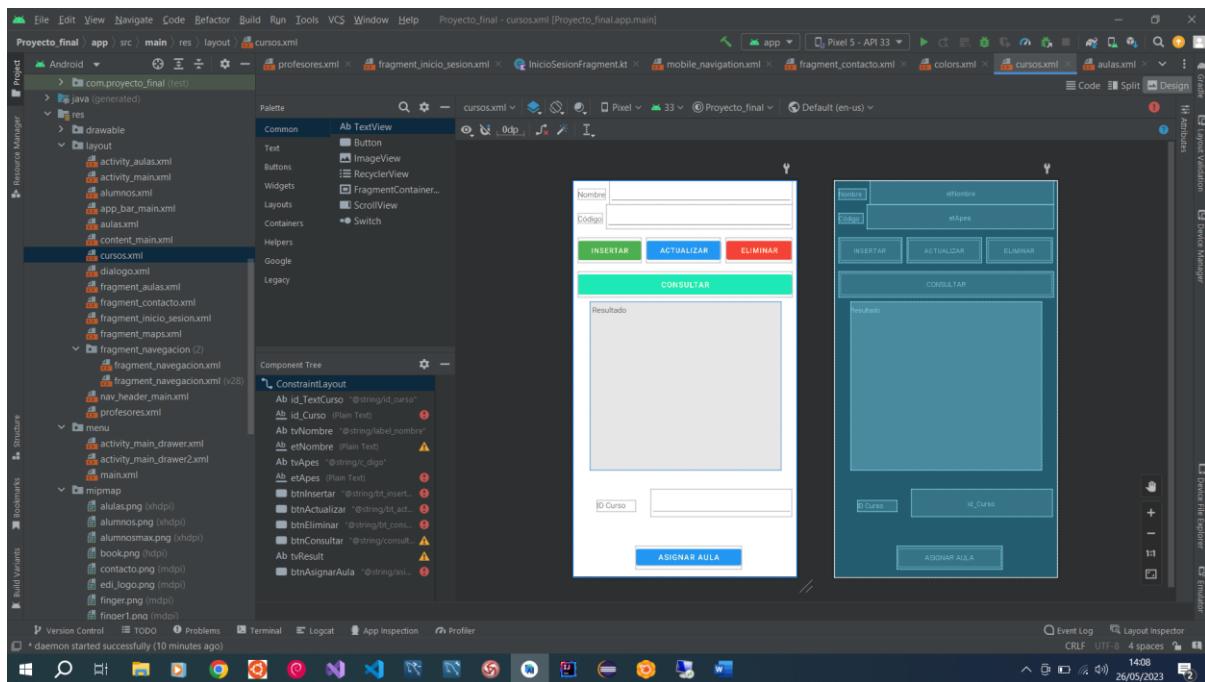


Ilustración 17:Herramienta gráfica de Android Studio.

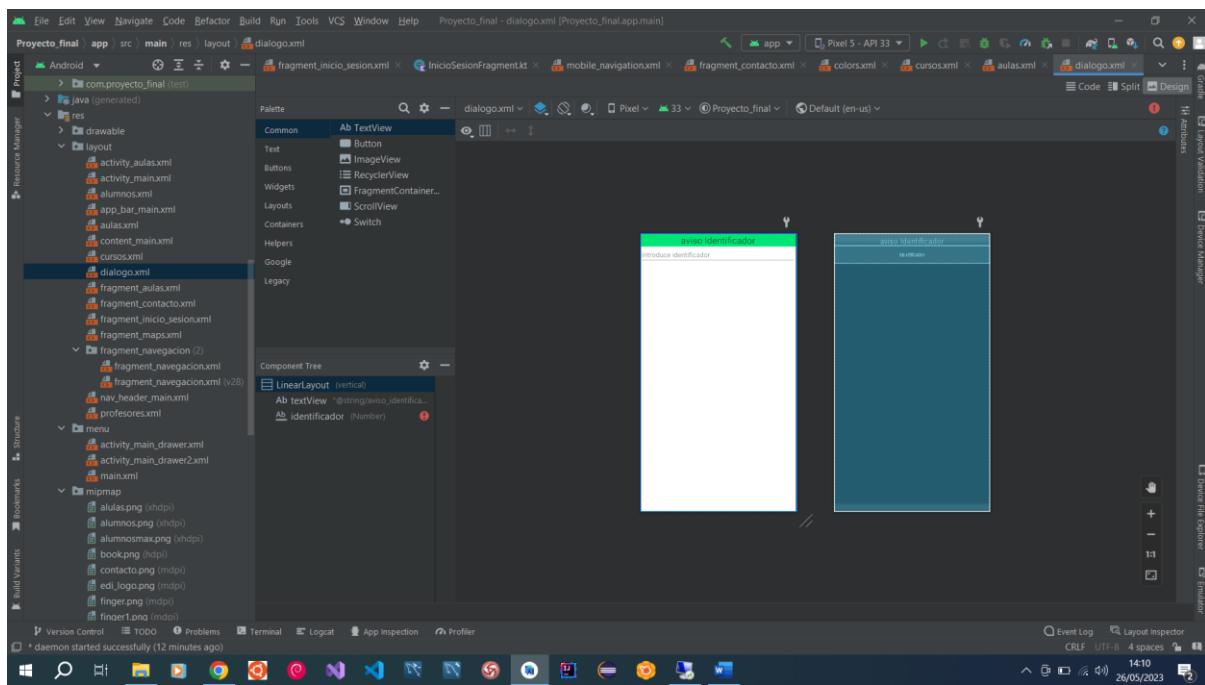


Ilustración 18:Herramienta gráfica de Android Studio.

8.1.2. Navegación entre pantallas

Para poder realizar las transiciones entre pantallas se ha optado por tener una única Activity (en castellano Actividad) y sobre la que cargar los diferentes Fragments que se han diseñado para las distintas funcionalidades de la aplicación, para poder hacer esto se ha utilizado la librería Navigation³⁷ de Android Jetpack, que permite una transición sencilla después de realizar la configuración pertinente en el documento XML que tiene asociado.

Para poder manipular esta configuración de una manera más ágil se añadió en Android Studio una herramienta similar a la comentada en el punto anterior, tanto en modo gráfico como texto.

Cada Fragment, así como cada una de las acciones que conectan a unas vistas con otras, deben tener un identificador único para poder acceder a ellas desde cualquier punto de la aplicación Android.

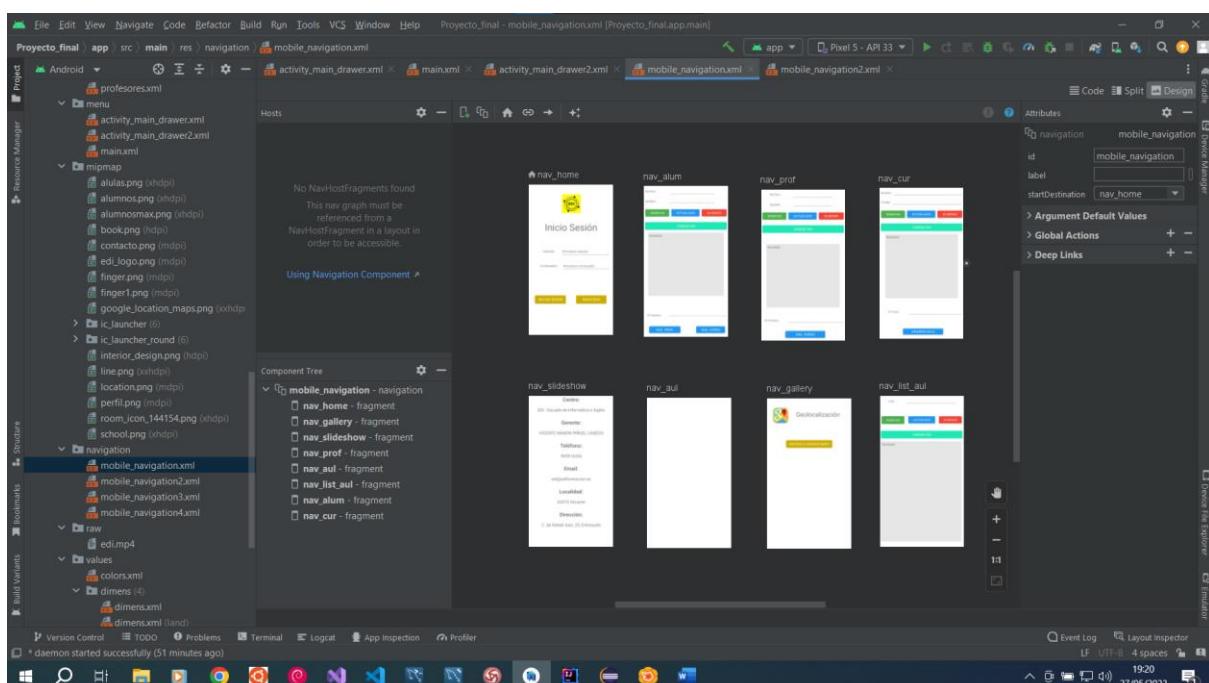


Ilustración 19:Herramienta gráfica de navegación entre vistas de Android Studio

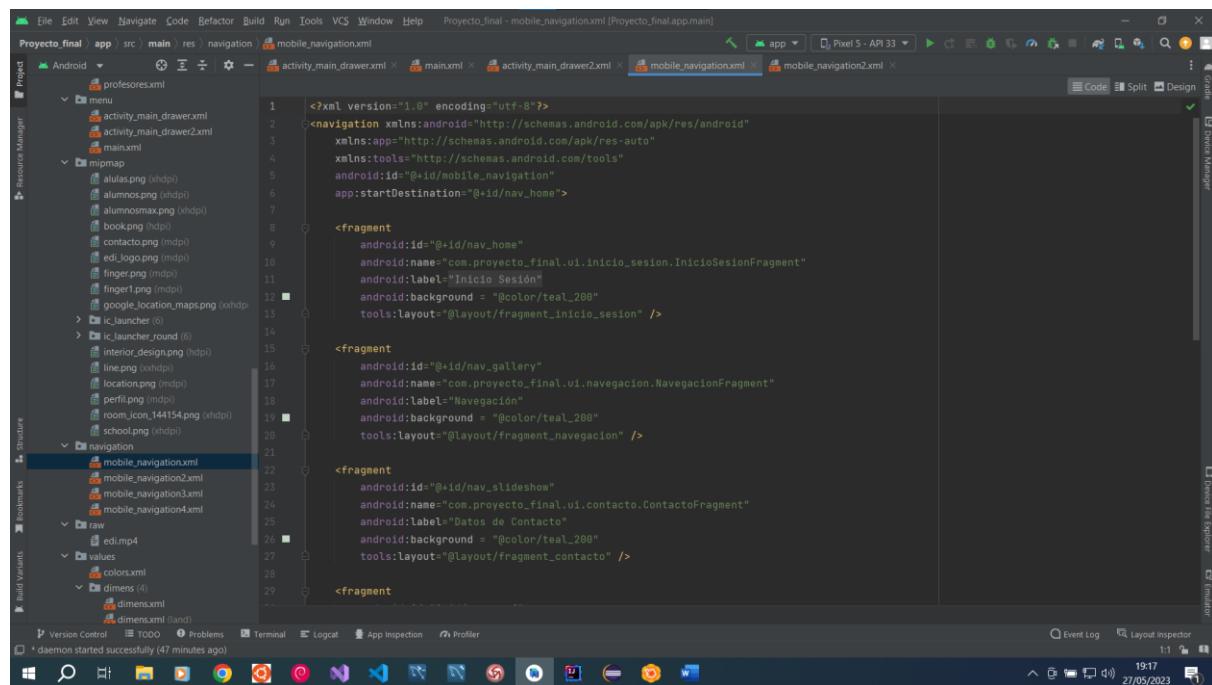
- <https://developer.android.com/guide/navigation>

8.1.3. Data Binding

Data Binding se utiliza para poder simplificar la implementación a la hora de identificar elementos de una vista en un Fragment. Al usar esta librería, Android Studio genera unos archivos con la notación del nombre de la vista y la terminación FragmentBinding, por ejemplo, para el menú principal sería MainMenuFragmentBinding.

En estos archivos generados se guardan las referencias de los elementos de la vista, por lo que sólo hay que crear una instancia de esta clase informando la instancia vista del Fragment, así poder acceder a ellos para asignarles un comportamiento.

A continuación, se muestra la asignación de acción para cada botón del menú principal



```
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_home">

    <fragment
        android:id="@+id/nav_home"
        android:name="com.proyecto_final.ui.inicio_sesion.InicioSesionFragment"
        android:label="Inicio Sesión"
        android:background = "@color/teal_200"
        tools:layout="@layout/fragment_inicio_sesion" />

    <fragment
        android:id="@+id/nav_gallery"
        android:name="com.proyecto_final.ui.navegacion.NavegacionFragment"
        android:label="Navegación"
        android:background = "@color/teal_200"
        tools:layout="@layout/fragment_navegacion" />

    <fragment
        android:id="@+id/nav_slideshow"
        android:name="com.proyecto_final.ui.contacto.ContactoFragment"
        android:label="Datos de Contacto"
        android:background = "@color/teal_200"
        tools:layout="@layout/fragment_contacto" />

</navigation>
```

Ilustración 20:Asignación de Acción Botón Menú Principal Parte 1.

```
<fragment
    android:id="@+id/nav_slideshow"
    android:name="com.proyecto_final.ui.contacto.ContactoFragment"
    android:label="Datos de Contacto"
    android:background="@color/teal_200"
    tools:layout="@layout/fragment_contacto" />

<fragment
    android:id="@+id/nav_prof"
    android:name="com.proyecto_final.ui.profesores.ProfesoresFragment"
    android:label="Profesores"
    android:background="@color/teal_200"
    tools:layout="@layout/profesores" />

<fragment
    android:id="@+id/nav_aula"
    android:name="com.proyecto_final.ui.aulas.AulasFragment"
    android:label="Diseño Aulas del Centro"
    android:background="@color/teal_200"
    tools:layout="@layout/fragment_aulas" />

<fragment
    android:id="@+id/nav_list_aula"
    android:name="com.proyecto_final.ui.aulas.ListadoAulasFragment"
    android:label="Aulas del Centro"
    android:background="@color/teal_200"
    tools:layout="@layout/aulas" />

<fragment
    android:id="@+id/nav_alum"
    android:name="com.proyecto_final.ui.alumnos.AlumnosFragment"
    android:label="Alumnos"
    android:background="@color/teal_200"
    tools:layout="@layout/alumnos" />
```

Ilustración 21:Asignación de Acción Botón Menú Principal Parte 2.

```
<fragment
    android:id="@+id/nav_aula"
    android:name="com.proyecto_final.ui.aulas.AulasFragment"
    android:label="Diseño Aulas del Centro"
    android:background="@color/teal_200"
    tools:layout="@layout/fragment_aulas" />

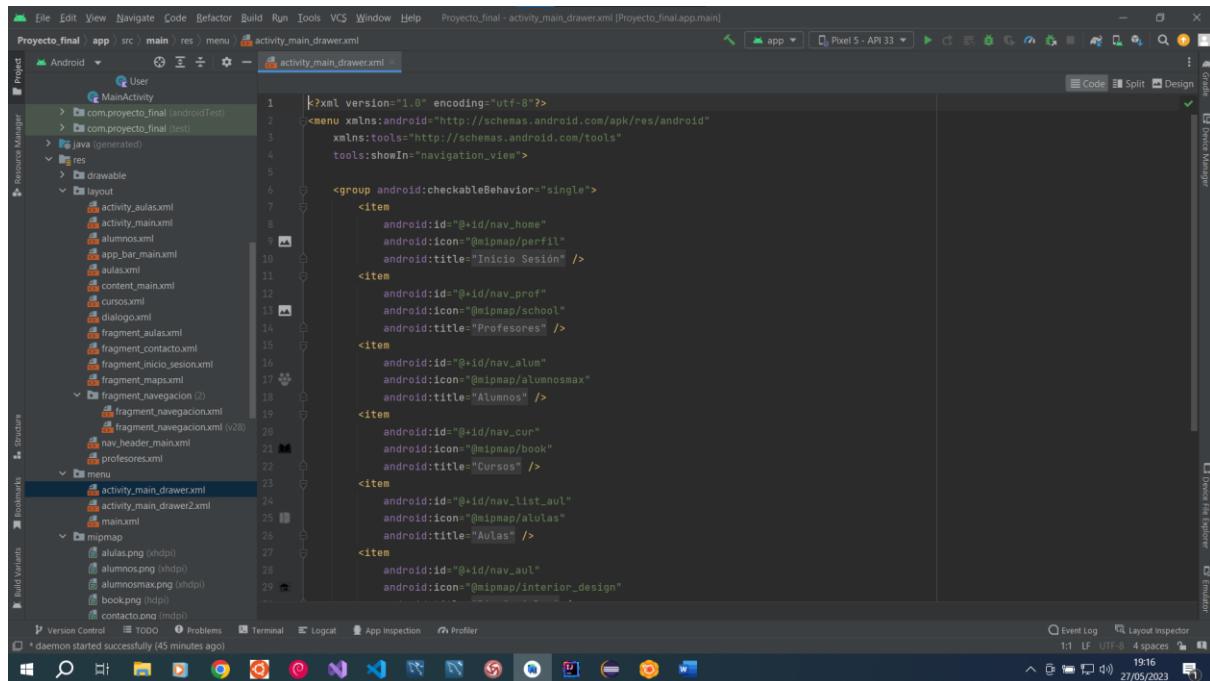
<fragment
    android:id="@+id/nav_list_aula"
    android:name="com.proyecto_final.ui.aulas.ListadoAulasFragment"
    android:label="Aulas del Centro"
    android:background="@color/teal_200"
    tools:layout="@layout/aulas" />

<fragment
    android:id="@+id/nav_alum"
    android:name="com.proyecto_final.ui.alumnos.AlumnosFragment"
    android:label="Alumnos"
    android:background="@color/teal_200"
    tools:layout="@layout/alumnos" />

<fragment
    android:id="@+id/nav_curs"
    android:name="com.proyecto_final.ui.curso.CursosFragment"
    android:label="Cursos"
    android:background="@color/teal_200"
    tools:layout="@layout/cursos" />
```

Ilustración 22:Asignación de Acción Botón Menú Principal Parte 3.

Una vez visto el menú principal, se muestra el código del adaptador para el menú principal.

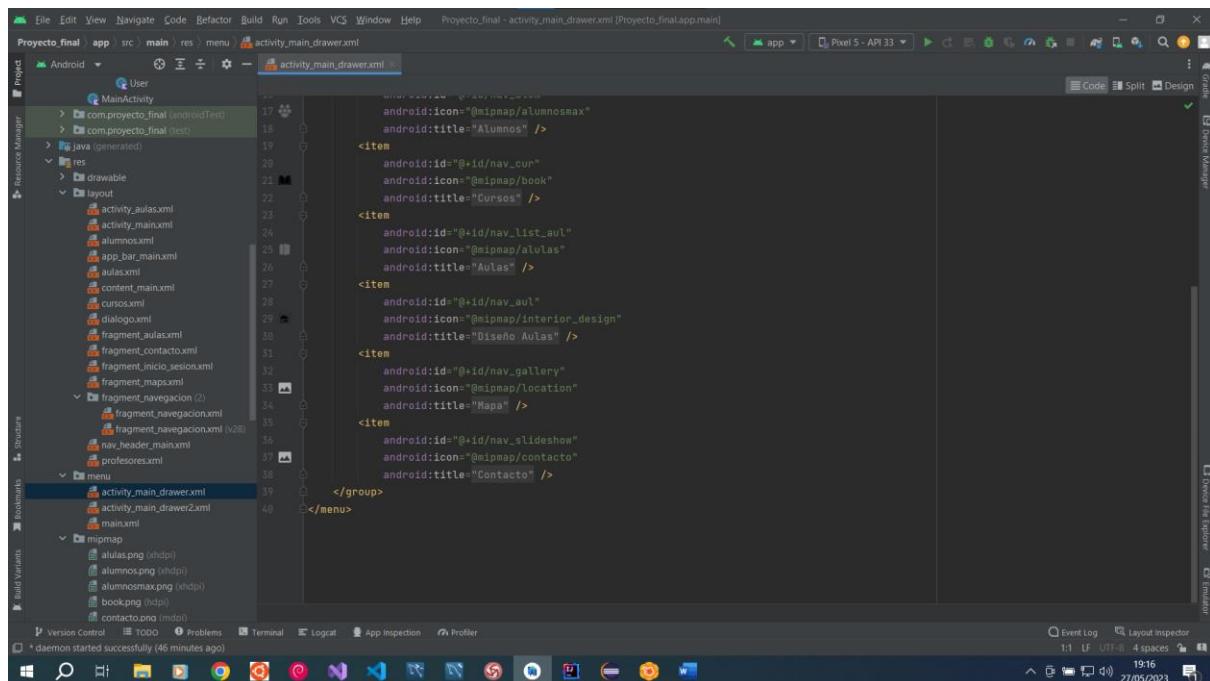


The screenshot shows the Android Studio interface with the code editor open for the file `activity_main_drawer.xml`. The code defines a navigation drawer menu with various items for navigating between different screens like Home, Profesores, Alumnos, Cursos, Aulas, and Contacto.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_home"
            android:icon="@mipmap/perfil"
            android:title="Inicio Sesión" />
        <item
            android:id="@+id/nav_prof"
            android:icon="@mipmap/school"
            android:title="Profesores" />
        <item
            android:id="@+id/nav_alum"
            android:icon="@mipmap/alumnosmax"
            android:title="Alumnos" />
        <item
            android:id="@+id/nav_curs"
            android:icon="@mipmap/book"
            android:title="Cursos" />
        <item
            android:id="@+id/nav_list_aul"
            android:icon="@mipmap/aulas"
            android:title="Aulas" />
        <item
            android:id="@+id/nav_aul"
            android:icon="@mipmap/interior_design"
            android:title="Diseño Aulas" />
    </group>
</menu>
```

Ilustración 23: Código Adaptador Menú Principal Parte 1.



The screenshot shows the continuation of the code for `activity_main_drawer.xml` in the Android Studio code editor. It includes additional items for navigating to the gallery, slideshows, and contact information.

```
        <item
            android:id="@+id/nav_gallery"
            android:icon="@mipmap/location"
            android:title="Mapa" />
        <item
            android:id="@+id/nav_slideshow"
            android:icon="@mipmap/contacto"
            android:title="Contacto" />
    </group>
</menu>
```

Ilustración 24: Código Adaptador Menú Principal Parte 2.

```
private lateinit var appBarConfiguration: AppBarConfiguration
private lateinit var binding: ActivityMainBinding

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)

    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)

    setSupportActionBar(binding.appBarMain.toolbar)

    binding.appBarMain.toolbar.setOnClickListener { view ->
        Snackbar.make(view, "EDI (Escuela de Informática e Ingles)", Snackbar.LENGTH_LONG)
            .setAction("Action", null).show()
    }

    binding.appBarMain.toolbar.app_bar_main
    val drawerLayout: DrawerLayout = binding.drawerLayout
    val navView: NavigationView = binding.navView
    val navController = findNavController(R.id.nav_host_fragment_main)
    // Passing each menu ID as a set of Ids because each
    // menu should be considered as top level destinations.
    appBarConfiguration = AppBarConfiguration(setOf(
        R.id.nav_home, R.id.nav_gallery, R.id.nav_slideshow, R.id.nav_alum, R.id.nav_list_alum, R.id.nav_prof, R.id.nav_cur), drawerLayout)
    setupActionBarWithNavController(navController, appBarConfiguration)
    navView.setupWithNavController(navController)
}
```

Ilustración 25: Código Adaptador Menú Principal Parte 3.

8.1.4. Modelo

El modelo se encarga de realizar una representación en forma de objeto, de una de las entidades de la base de datos y permiten al desarrollador poder trabajar con los datos de una, o varias, filas de la tabla de una forma ágil.

Cada modelo tiene una serie de campos, uno que debe referenciar al nombre de la tabla, otro que refiera al identificador único si lo tuviese y otro que refiera al resto de columnas. Además, puede contener métodos que refieren a otros objetos de la base de datos o que sirvan como secuencia de búsqueda.

A continuación, se presenta el modelo de la tabla aulas.

```
data class Aulas(val id: Int, val aula: String)
```

Ilustración 26: Modelo Tabla Aulas.

Ahora el modelo de la tabla usuarios.

```
data class User(val id: Int, val user: String, val password:String)
```

Ilustración 27: Modelo Tabla Users.

Ahora el modelo de la tabla Cursos.

```
data class Curso(val id: Int, val nombre: String, val apellido: String, val id_aula: Int)
```

Ilustración 28: Modelo Tabla Cursos.

Ahora el modelo de la tabla Profesores.

```
data class Profesor(val id: Int, val nombre: String, val apellido: String, val id_curso: Int)
```

Ilustración 29: Modelo Tabla Profesores.

Y, por último, el modelo de la tabla Alumnos.

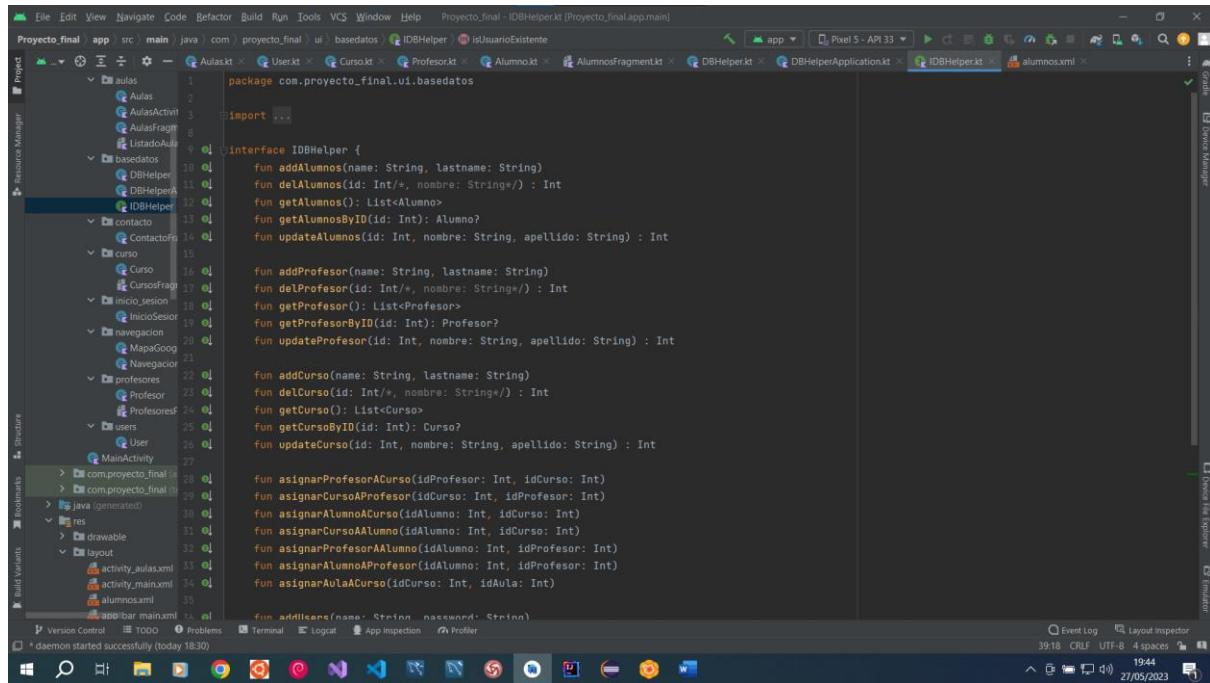
```
data class Alumno(val id: Int, val nombre: String, val apellido: String, val id_profesor: Int, val id_curso: Int)
```

Ilustración 30: Modelo Tabla Alumnos.

8.1.5. Controlador

El controlador se encarga de gestionar las peticiones realizadas a las APIs y en función de la información recibida proporciona una respuesta, ya sea con datos o con un error controlado.

A continuación, se muestra un ejemplo de uno de los métodos del controlador de las tablas.



```
package com.proyecto_final.ui.basedatos

import ...

interface IDBHelper {
    fun addAlumnos(name: String, lastname: String)
    fun delAlumnos(id: Int, nombre: String) : Int
    fun getAlumnos(): List<Alumno>
    fun getAlumnosByID(id: Int): Alumno?
    fun updateAlumnos(id: Int, nombre: String, apellido: String) : Int

    fun addProfesor(name: String, lastname: String)
    fun delProfesor(id: Int, nombre: String) : Int
    fun getProfesor(): List<Profesor>
    fun getProfesorByID(id: Int): Profesor?
    fun updateProfesor(id: Int, nombre: String, apellido: String) : Int

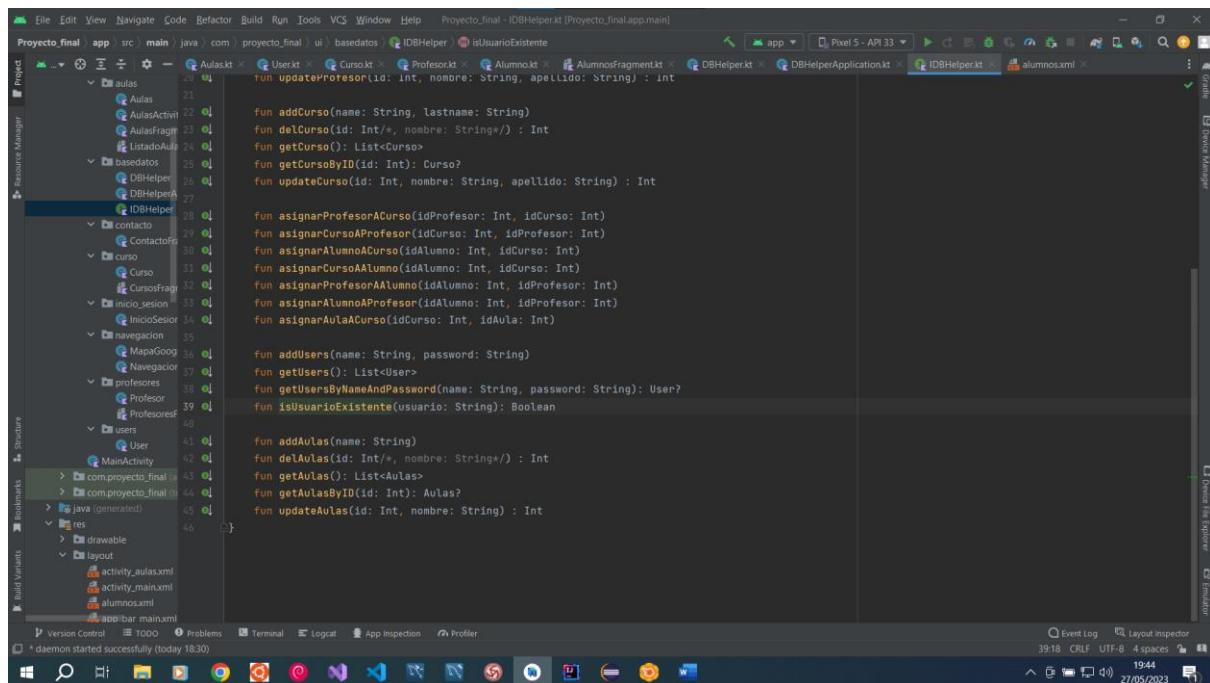
    fun addCurso(name: String, lastname: String)
    fun delCurso(id: Int, nombre: String) : Int
    fun getCurso(): List<Curso>
    fun getCursoByID(id: Int): Curso?
    fun updateCurso(id: Int, nombre: String, apellido: String) : Int

    fun asignarProfesorACurso(idProfesor: Int, idCurso: Int)
    fun asignarCursoProfesor(idCurso: Int, idProfesor: Int)
    fun asignarAlumnoACurso(idAlumno: Int, idCurso: Int)
    fun asignarCursoAlumno(idAlumno: Int, idCurso: Int)
    fun asignarProfesorAAAlumno(idAlumno: Int, idProfesor: Int)
    fun asignarAlumnoAProfesor(idAlumno: Int, idProfesor: Int)
    fun asignarAulaACurso(idCurso: Int, idAula: Int)

    fun addUsers(name: String, password: String)
    fun getUsers(): List<User>
    fun getUsersByNameAndPassword(name: String, password: String): User?
    fun isUsuarioExiste(usuario: String): Boolean
}

fun updateAulas(id: Int, nombre: String) : Int
```

Ilustración 31: Métodos de las Tablas BBDD Parte 1.



```
package com.proyecto_final.ui.basedatos

import ...

interface IDBHelper {
    fun updateProfesor(id: Int, nombre: String, apellido: String) : Int
    fun addCurso(name: String, lastname: String)
    fun delCurso(id: Int, nombre: String) : Int
    fun getCurso(): List<Curso>
    fun getCursoByID(id: Int): Curso?
    fun updateCurso(id: Int, nombre: String, apellido: String) : Int

    fun asignarProfesorACurso(idProfesor: Int, idCurso: Int)
    fun asignarCursoProfesor(idCurso: Int, idProfesor: Int)
    fun asignarAlumnoACurso(idAlumno: Int, idCurso: Int)
    fun asignarCursoAlumno(idAlumno: Int, idCurso: Int)
    fun asignarProfesorAAAlumno(idAlumno: Int, idProfesor: Int)
    fun asignarAlumnoAProfesor(idAlumno: Int, idProfesor: Int)
    fun asignarAulaACurso(idCurso: Int, idAula: Int)

    fun addUsers(name: String, password: String)
    fun getUsers(): List<User>
    fun getUsersByNameAndPassword(name: String, password: String): User?
    fun isUsuarioExiste(usuario: String): Boolean
}

fun updateAulas(id: Int, nombre: String) : Int
```

Ilustración 32: Métodos de las Tablas BBDD Parte 2.

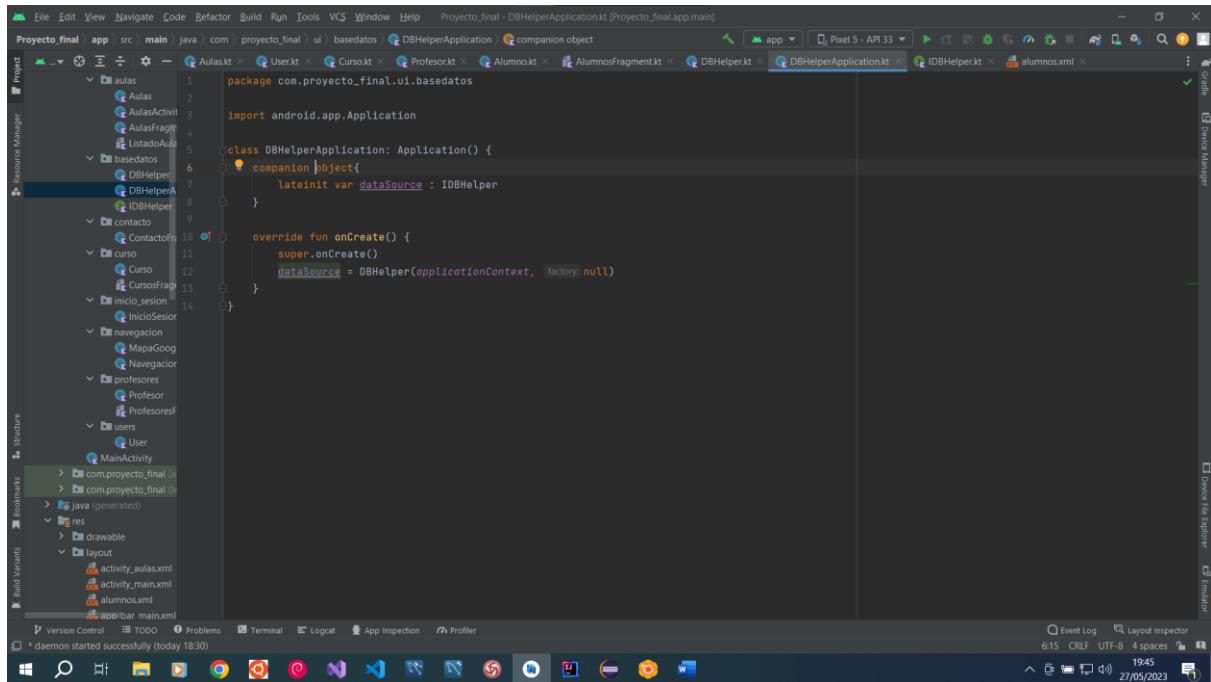


Ilustración 33: Métodos de las Tablas BBDD Parte 3.

8.1.6. Vista

La vista es una representación de los datos que un usuario puede ver en pantalla, para poder realizar esto se han usado plantillas Blade que se van añadiendo a una plantilla maestra que incluye una cabecera y un pie de página.

A continuación, se muestra un ejemplo de la plantilla de la vista alumnos.

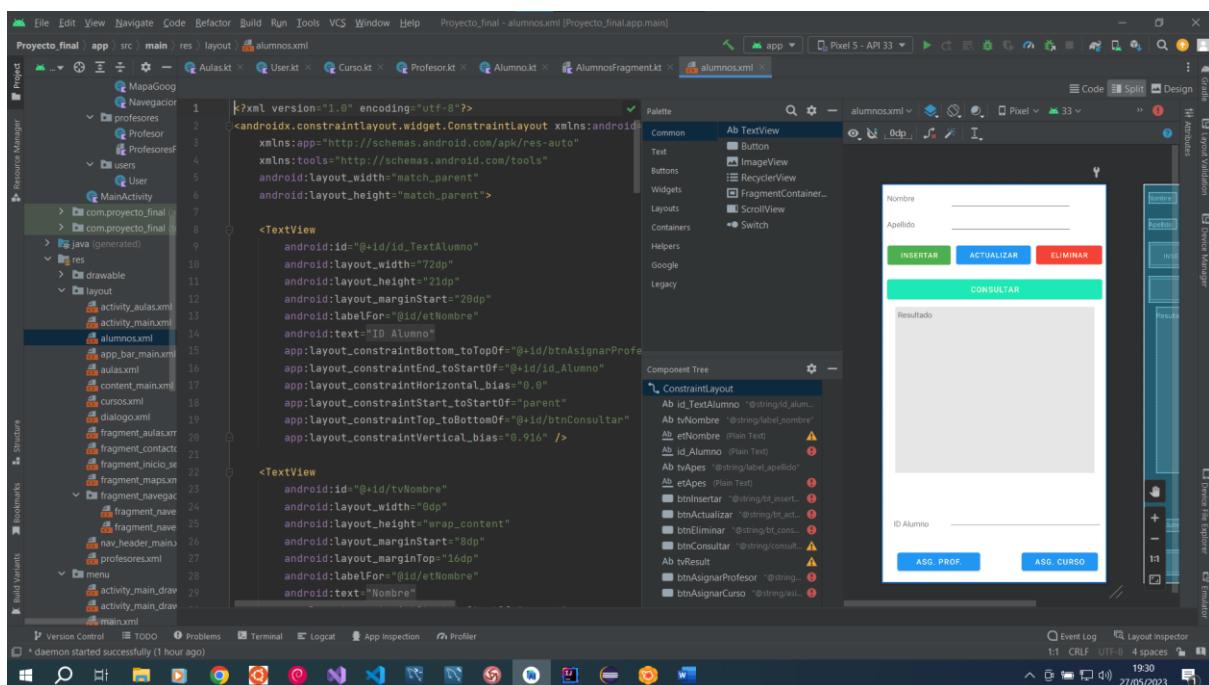


Ilustración 34: Vista Plantilla Alumnos.

9. Funcionamiento de la aplicación

Menú principal:

- Inicio Sesión: Poder acceder a las opciones del menú principal dependiendo de si eres administrador o un usuario genérico. Deberás Autenticarte, es decir, iniciando sesión como administrador, te dará acceso a todas las opciones del menú principal. Iniciando sesión como usuario genérico, te dará acceso solamente a tres opciones que son la información del centro, la geolocalización y el diseño de las aulas. También es posible no autentificarse, esto sería sin ningún usuario, solo tendrás acceso a la opción de inicio de sesión.
- Contactos: Ver la información del centro EDI Escuela de Informática e Ingles.
- Diseño aulas: Ver el diseño e información de cada aula del centro EDI Escuela de Informática e Ingles.
- Geolocalización: Mostrar la localización en un mapa a través de google Maps con un key autenticador validado para su autorización inserción en aplicación.
- Aulas: Administrar la base de datos de la tabla aulas. Podrá añadir, modificar, eliminar, consultar y asignar aulas a la aplicación.
- Cursos: Administrar la base de datos de la tabla cursos. Podrá añadir, modificar, eliminar, consultar y asignar cursos a la aplicación.
- Profesores: Administrar la base de datos de la tabla Profesores. Podrá añadir, modificar, eliminar, consultar y asignar Profesores a la aplicación.
- Alumnos: Administrar la base de datos de la tabla Alumnos. Podrá añadir, modificar, eliminar, consultar y asignar alumnos a la aplicación.

- ❖ Alumnos: Deberá introducir un nombre y un apellido. Para asignar un profesor o un curso se tendrá en cuenta el ID del alumno que queremos. Para asignar tendremos que saber los ID correspondientes del profesor o curso.

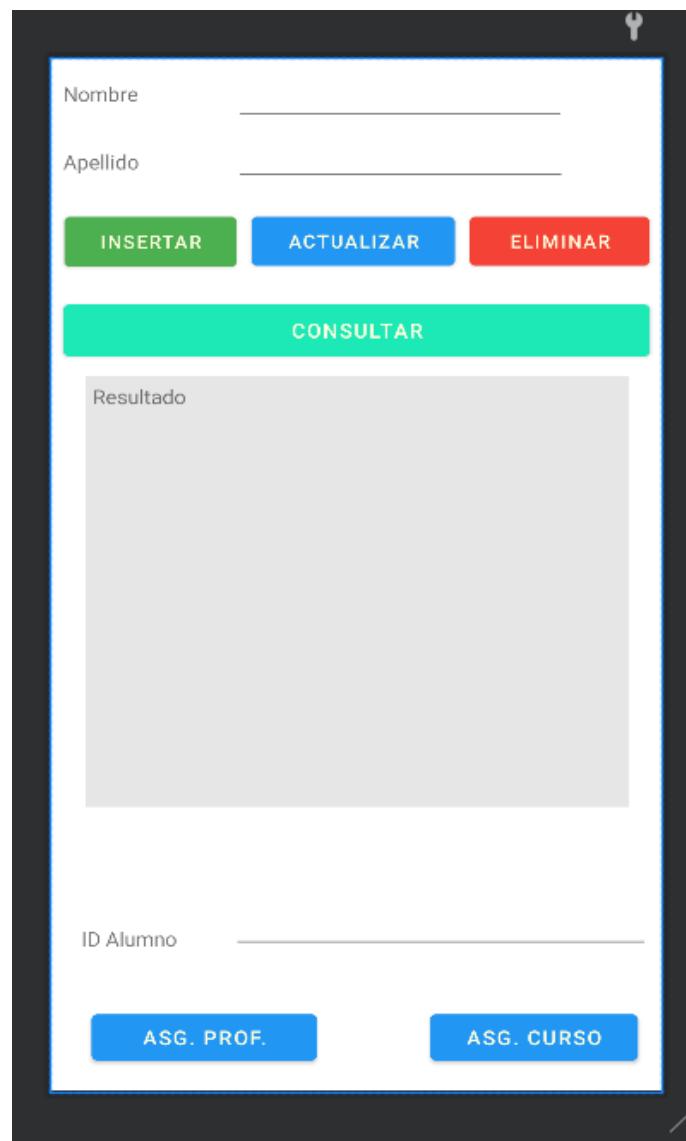


Ilustración 35: Funcionalidad Alumnos

- ❖ Profesores: Deberá introducir un nombre y un apellido. Para asignar un curso se tendrá en cuenta el ID del profesor que queremos. Para asignar tendremos que saber el ID correspondiente del curso.

The diagram illustrates a user interface for managing professors. At the top, there are two input fields: 'Nombre' (Name) and 'Apellido' (Last Name), each with a corresponding text input box. Below these are three buttons: 'INSERTAR' (Insert) in green, 'ACTUALIZAR' (Update) in blue, and 'ELIMINAR' (Delete) in red. A large green button labeled 'CONSULTAR' (Consult) is positioned below the update and delete buttons. A large gray rectangular area labeled 'Resultado' (Result) is located in the center. At the bottom, there is another input field labeled 'ID Profesor' (Professor ID) with a text input box, and a blue button labeled 'ASG. CURSO' (Assign Course).

Ilustración 36: Funcionalidad Profesores

- ❖ Cursos: Deberá introducir un nombre y un código. Para asignar un aula se tendrá en cuenta el ID del curso que queremos. Para asignar tendremos que saber el ID correspondiente del aula.

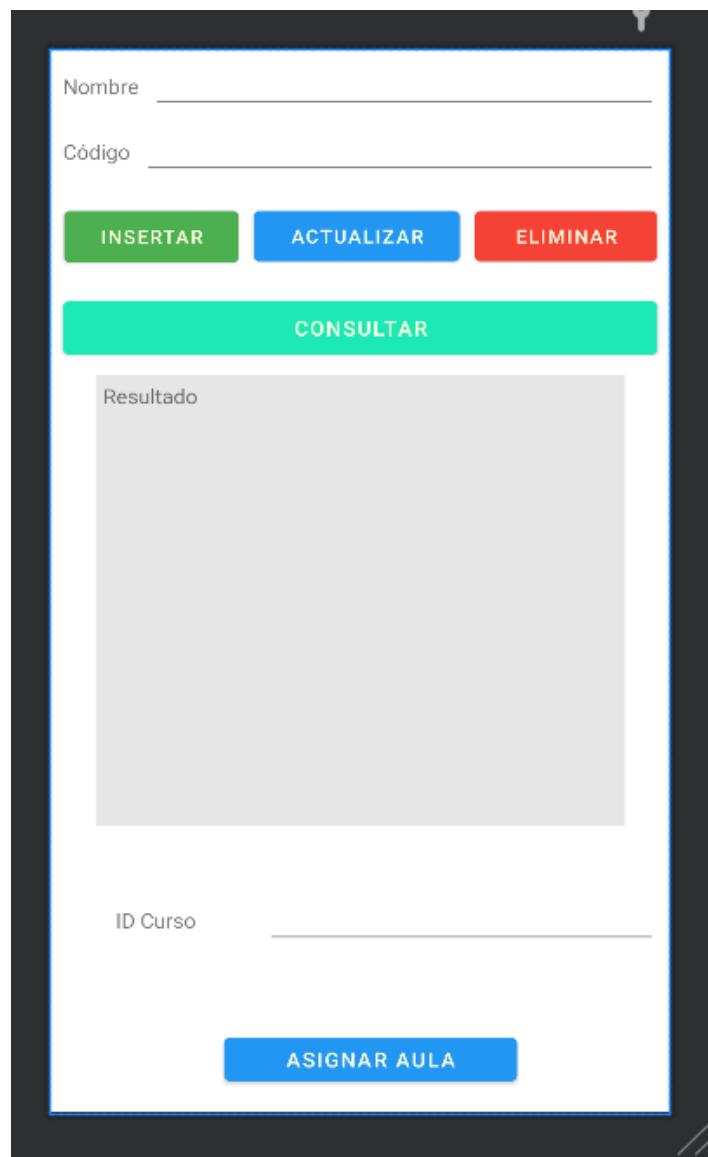


Ilustración 37: Funcionalidad Cursos



Aulas: Deberá introducir un nombre de aula.

The diagram illustrates a user interface for managing classrooms. At the top left is a label "Aula" next to an input field. Below this are three buttons: "INSERTAR" (green), "ACTUALIZAR" (blue), and "ELIMINAR" (red). A large green button labeled "CONSULTAR" is positioned centrally below the update buttons. The bottom section is a large gray area labeled "Resultado" (Result) where output would be displayed.

Ilustración 38: Funcionalidad Aulas

- ❖ Como se verá el menú principal:

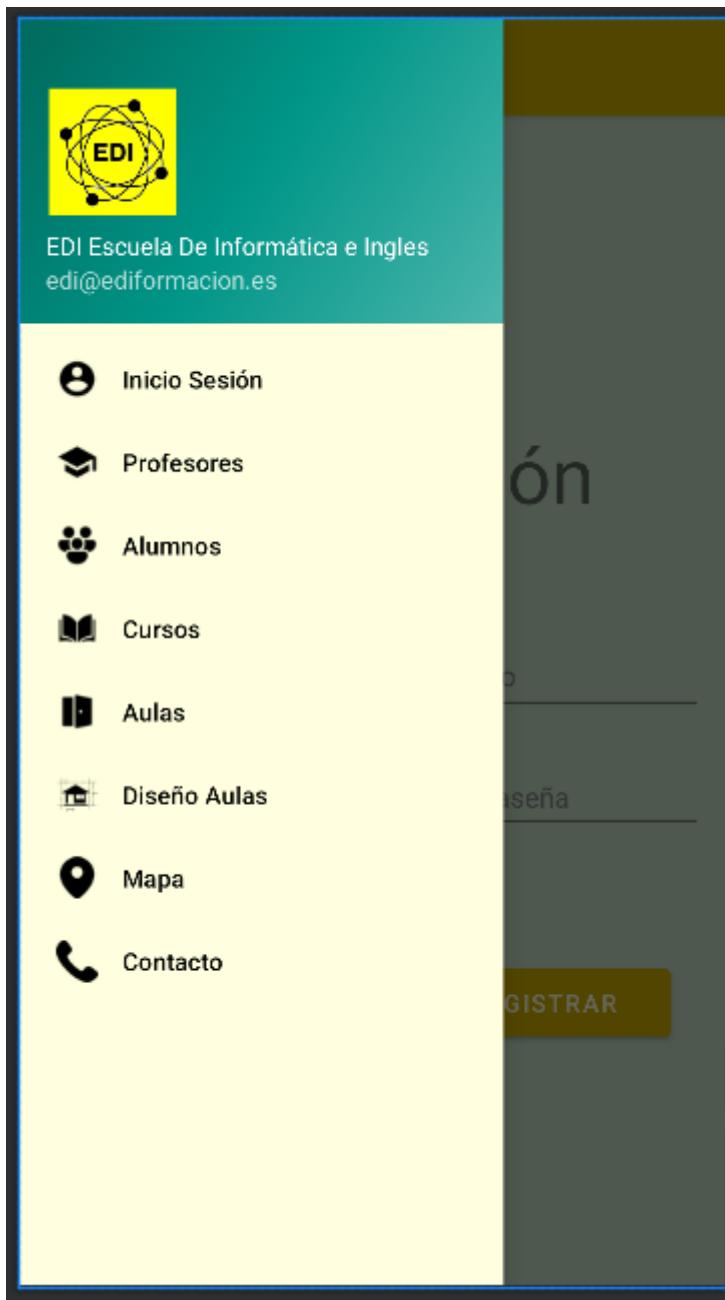


Ilustración 39: Funcionalidad Menú Principal

- ❖ Inicio sesión: Deberás introducir el usuario y contraseña y darle a iniciar sesión si ya tienes un usuario para autenticarte o registrarte para añadir ese usuario y contraseña a la base de datos para su posterior inicio de sesión.

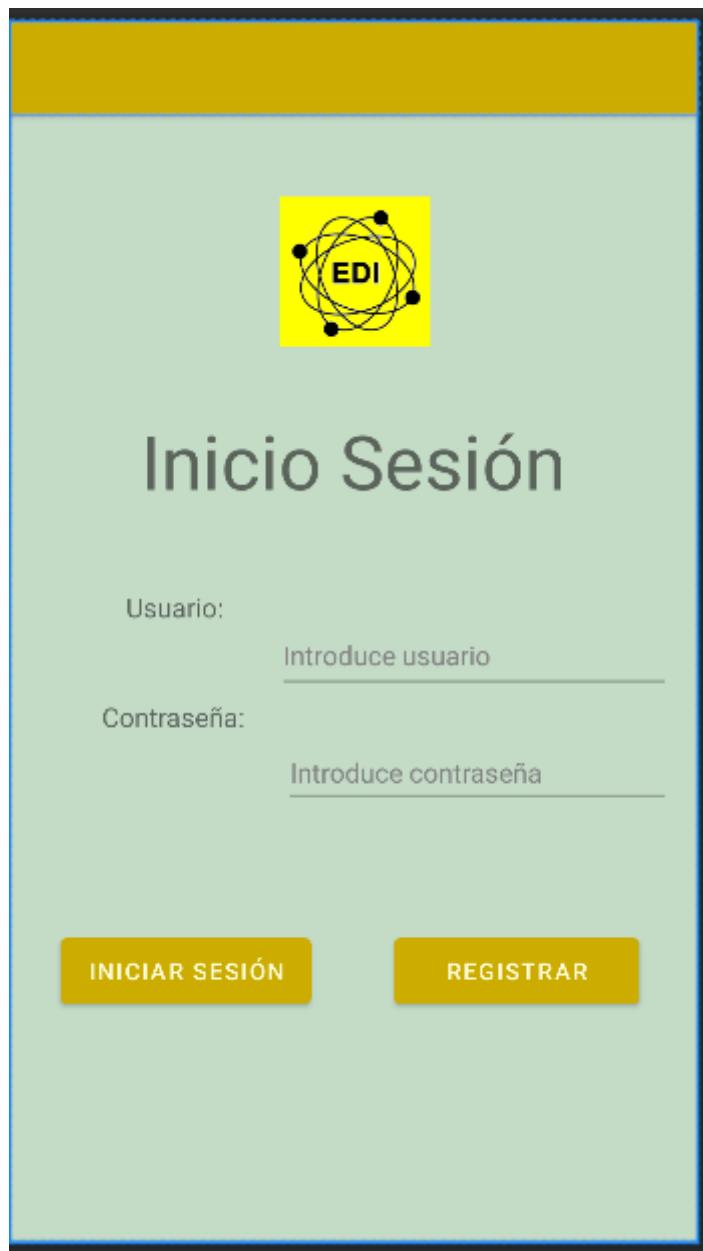


Ilustración 40: Funcionalidad Inicio Sesión

- ❖ Contactos: Esta opción es solo información del centro. Se muestra dicha información y se podrá visualizar.

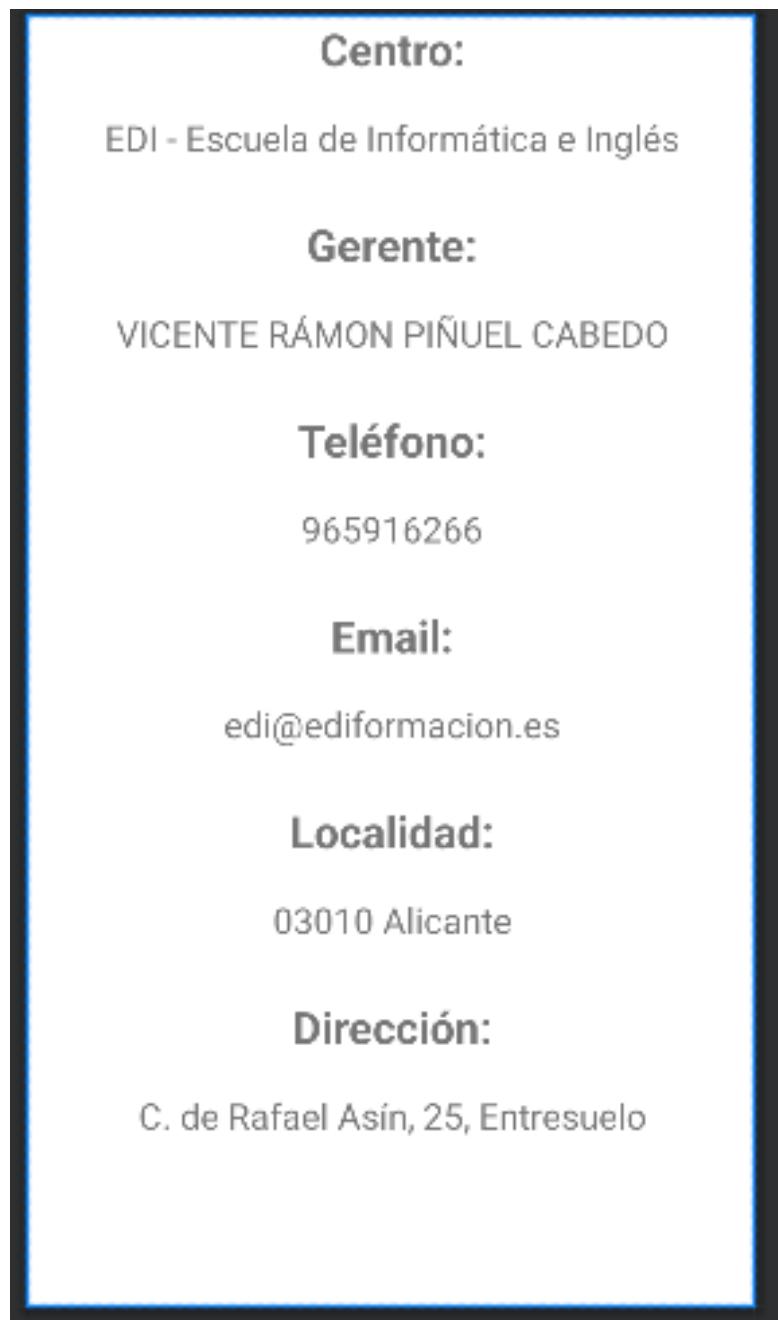


Ilustración 41: Funcionalidad Contactos

- ❖ Diseño de aulas: Podrás visualizar un listado de las aulas del centro y al pinchar sobre ellas, se cargará un video de las aulas del centro y una descripción de lo que contiene esa aula.

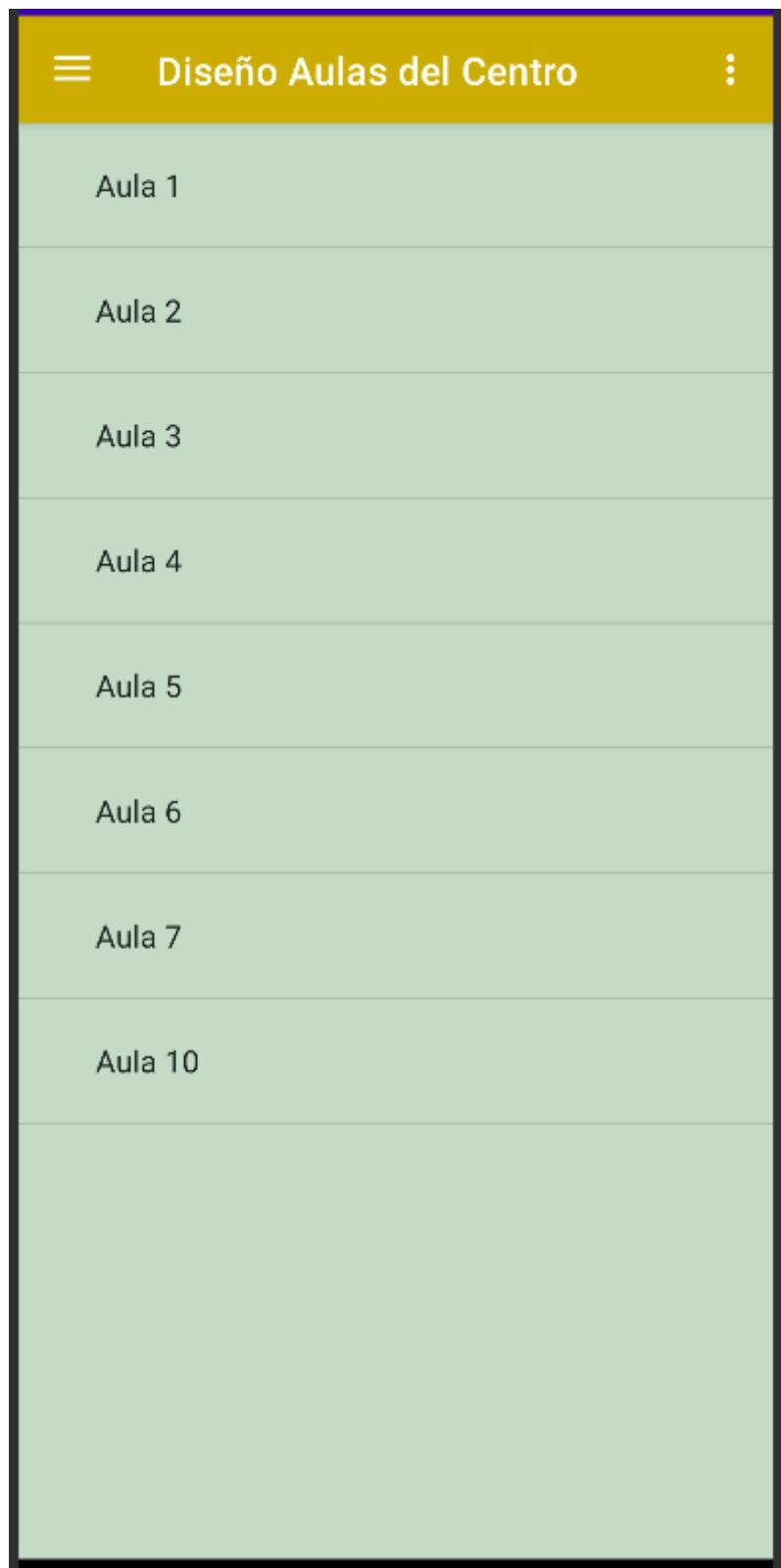
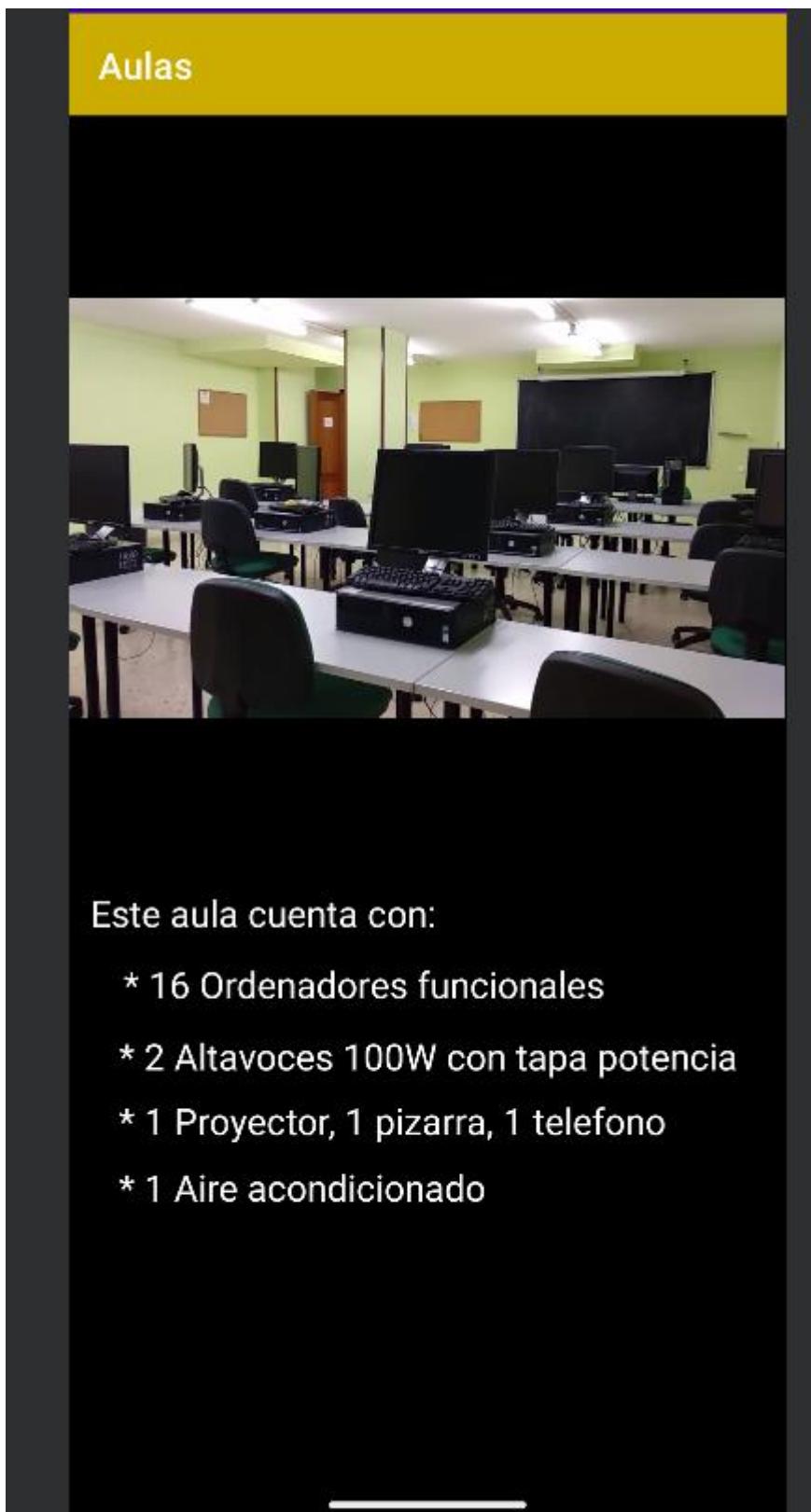


Ilustración 42: Funcionalidad Diseño Aulas Parte 1



Este aula cuenta con:

- * 16 Ordenadores funcionales
- * 2 Altavoces 100W con tapa potencia
- * 1 Proyector, 1 pizarra, 1 telefono
- * 1 Aire acondicionado

Ilustración 43: Funcionalidad Diseño Aulas Parte2

- ❖ Geolocalización: Podrás acceder a la localización del centro mediante la autenticación con una key de google maps.

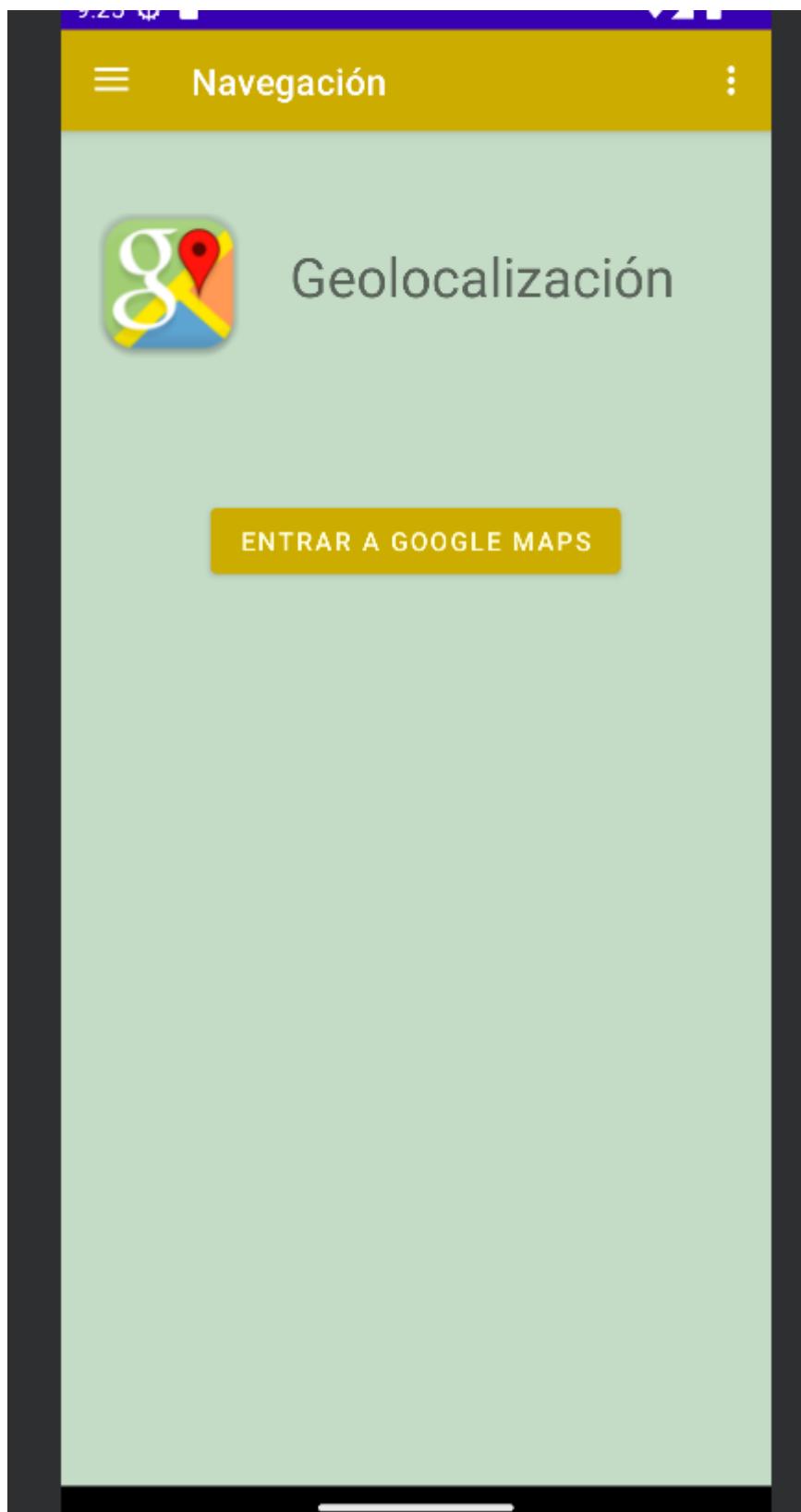


Ilustración 44: Funcionalidad Geolocalización Parte 1

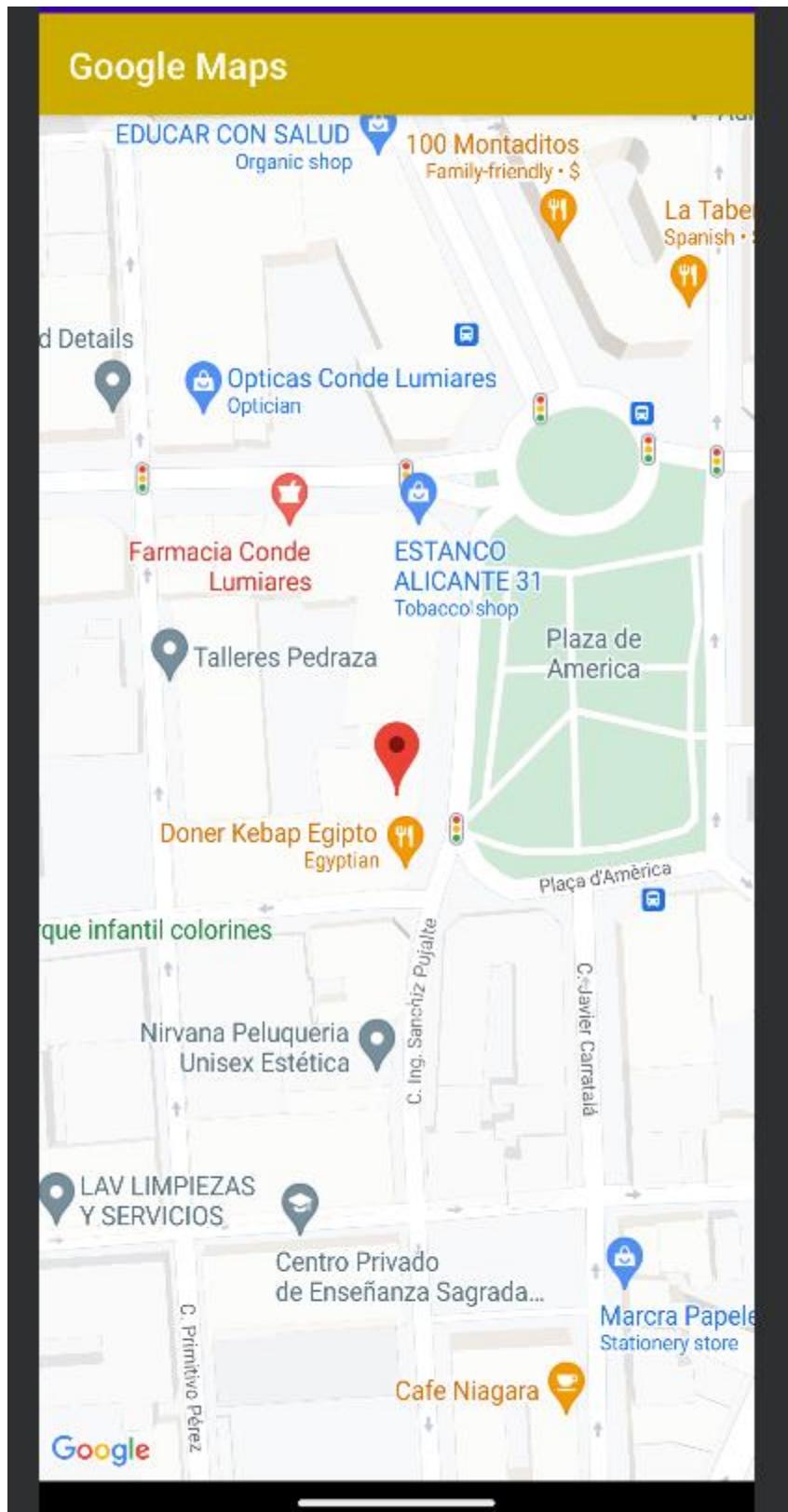


Ilustración 45: Funcionalidad Geolocalización Parte 2

Menú secundario:

- Cerrar sesión: Una vez se haya iniciado una sesión, bien sea de un usuario genérico o un usuario administrador. Está habilitada una opción en un menú secundario para cerrar la sesión abierta y volver al comienzo de la aplicación.

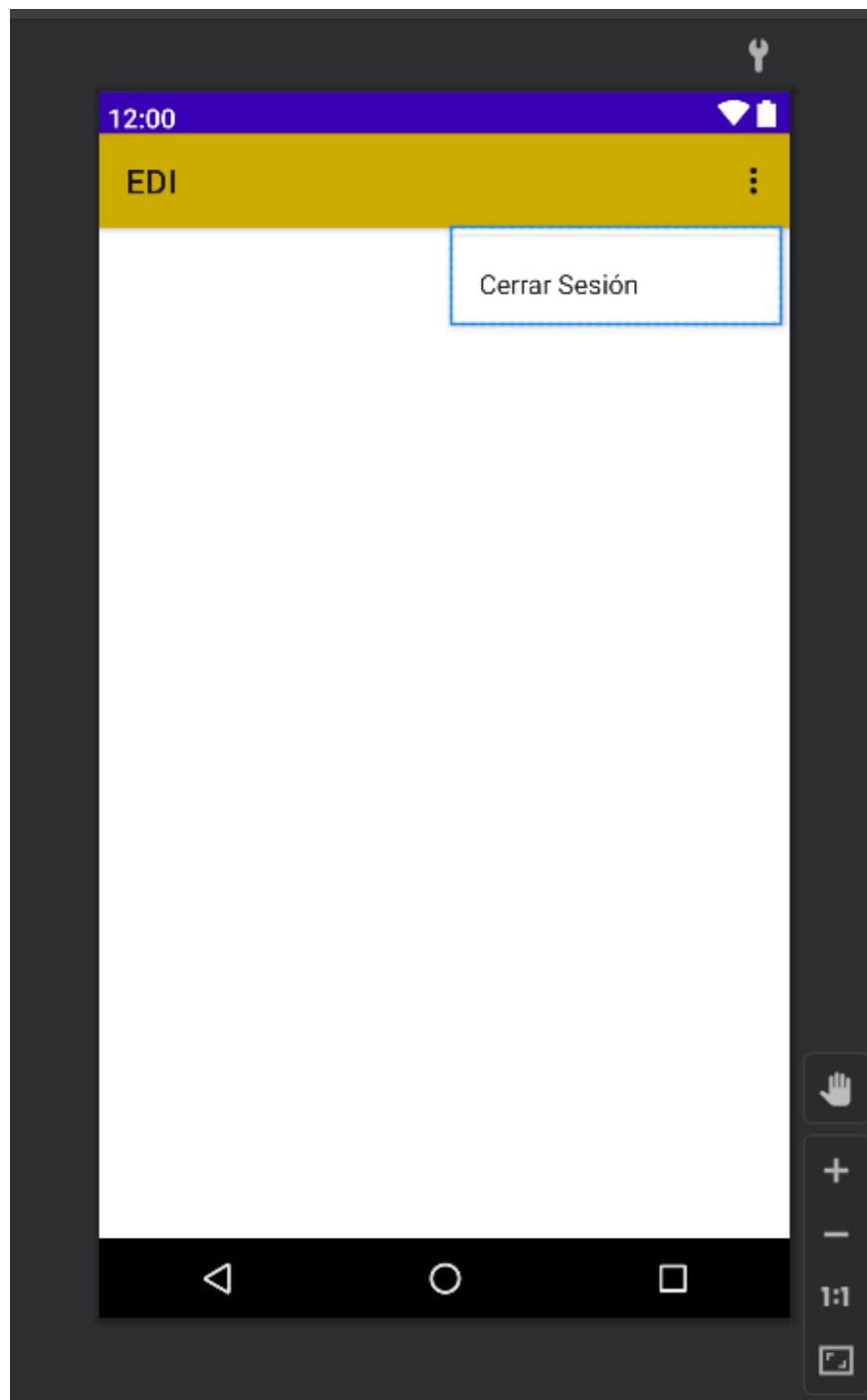


Ilustración 46: Funcionalidad Menú Secundario Cerrar Sesión

10. Análisis de Negocio

- ❖ **¿QUÉ VENDO?** Explicación de que productos o servicio vamos a vender ahora o en el futuro, pero no una explicación técnica sino una explicación de índole más comercial en la que se resalten las bondades y cualidades de nuestro producto o servicio. ¿Por qué voy a comprarlo?, ¿Qué necesidad cubre mi producto o servicio?

La actividad principal de la empresa va a consistir en la realización de aplicaciones móviles. Solamente se realizarán aplicaciones para sistema Android. Queremos cubrir la necesidad de implantar los programas habituales de centros educativos o centros de formación en las nuevas tecnologías como son el teléfono móvil inteligente.

- ❖ **¿A QUIEN SE LO VENDO?** Cuál es mi público objetivo, ¿quién son ese grupo de personas o empresas que van a necesitar mi producto o servicio.?

El público objetivo de la empresa serán la mayoría de centros dedicados a la educación tanto colegios, institutos, centro de formación o centros colaboradores. Esta empresa va a diferenciarse en este aspecto puesto que va a ofrecer servicios y productos dirigidos también al público infantil.

- ❖ **¿CÓMO SE LO VENDO?** Una vez que se haya determinado el público objetivo, ¿cómo voy a llegar a ellos ?, ¿de qué manera se van a enterar del producto o servicio que ofrezco?

Mensaje

El mensaje que se quiere transmitir es el de una empresa moderna que está a la vanguardia de las últimas tecnologías y que ofrece un proceso de mantenimiento de los programas para traspasarlos a un sistema novedoso y a disponibilidad en cualquier momento como lo son los teléfonos móviles inteligentes.

Estrategia

Las estrategias de comunicación de la empresa van a ser las siguientes:

- Comunicación corporativa: La filosofía de la empresa se basa en realizar aplicaciones móviles en un sistema operativo llamado Android. El diseño se centrará en transmitir modernidad y simplicidad a la hora de realizar las tareas cotidianas.
- Comunicación externa: La comunicación externa se abordará mediante marketing de captación, marketing directo, marketing indirecto y marketing relacional.

❖ ANALISIS DAFO – CAME

DEBILIDADES	AMENAZAS
<ul style="list-style-type: none"> - Solvencia técnica. - La empresa es desconocida. - Estructura aún poco sólida. - Bajos precios de productos importados. 	<ul style="list-style-type: none"> - Constante crecimiento del número de empresas dedicadas a la realización de aplicaciones móviles. - Fuerte competencia ejercida por grandes empresas del sector tecnológico.
FORTALEZAS	OPORTUNIDADES
<ul style="list-style-type: none"> - Empresa joven y entusiasta. - Calidad del servicio - Profundo conocimiento del sector. - Contactos profesionales. 	<ul style="list-style-type: none"> - Crecimiento de las nuevas tecnologías. - Redes sociales y aplicaciones en dispositivos móviles. - Abaratamiento del equipamiento tecnológico doméstico.

CORREGIR LAS DEBILIDADES

- Solvencia técnica. Al ser una empresa emergente, y no disponer aún de una solvencia técnica que la acrechte frente a clientes/as e inversores/as, encontrará dificultades a la hora de posicionarse en el mercado y conseguir financiación. Para corregirlo, habrá que buscar financiación en diferentes fuentes e iniciar una campaña de comunicación importante.
- La empresa es desconocida. Por la misma razón que en el punto anterior, la empresa aún es poco conocida por la mayoría de clientes/as potenciales. El Plan de Comunicación adquiere especial protagonismo.
- Estructura aún poco sólida. Al ser una empresa que está comenzando aún no está consolidada en el mundo laboral y puede verse afectado por esto. Para corregirlo, habrá que buscar la manera de consolidar la empresa en el mundo laboral, realizando un buen trabajo.
- Bajos precios de productos importados. Al realizar aplicaciones Android, habrá centros o empresas que no dispongan de un dispositivo móvil inteligente y se han tenido que bajar los precios para su venta. Para solventarlo, habrá que mirar al por mayor para sacar una mayor rentabilidad a los productos.

11. Generar App Android .APK

Para generar el APK correspondiente del proyecto primero debemos saber que existen dos formas de realizarlo:

La primera es sin firma, esto es debido a que no necesitamos tener unas credenciales de Google y es completamente generada sin autor de la aplicación y al instalarla, nos comunicará que la app no tiene derechos de autor que si queremos seguir instalando y que debemos activar la opción de instalar desconocido en nuestros teléfonos móviles.

La segunda opción, es firmando la aplicación, pues que necesitamos unas credenciales para la aplicación se deben generar para que podamos extraer el archivo APK a nuestro repositorio.

Hemos elegido la segunda opción puesto que vamos a continuar explicando su extracción y generación.

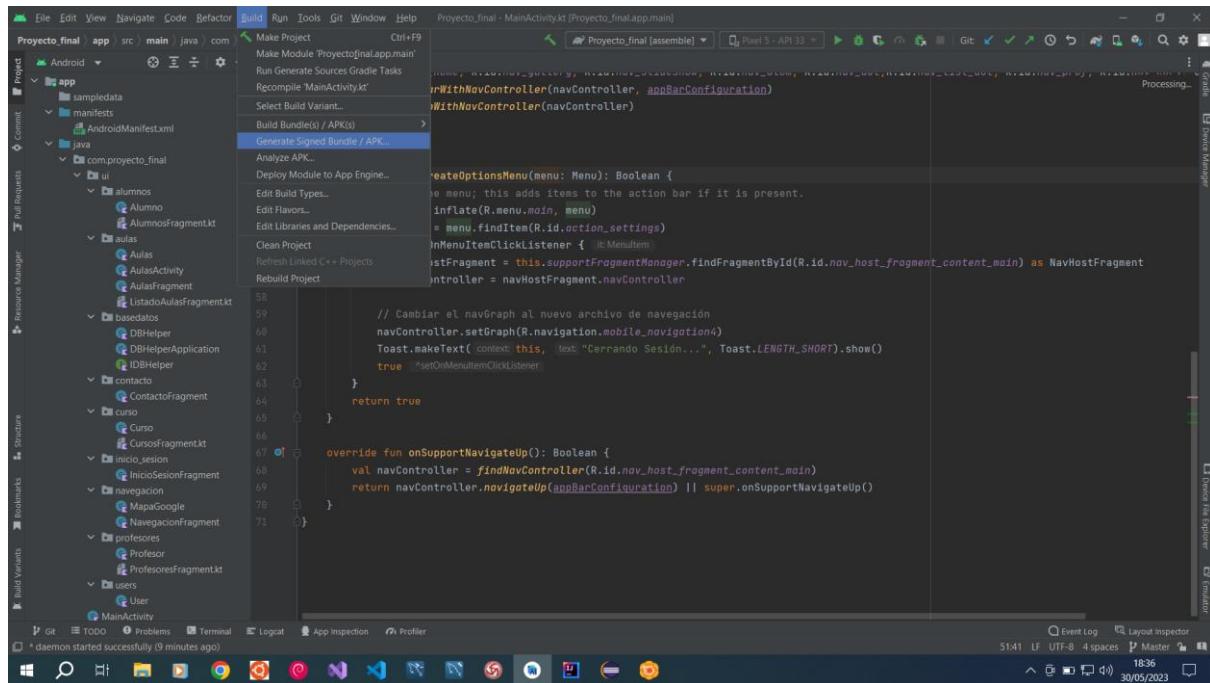


Ilustración 47: Generar APK Parte 1

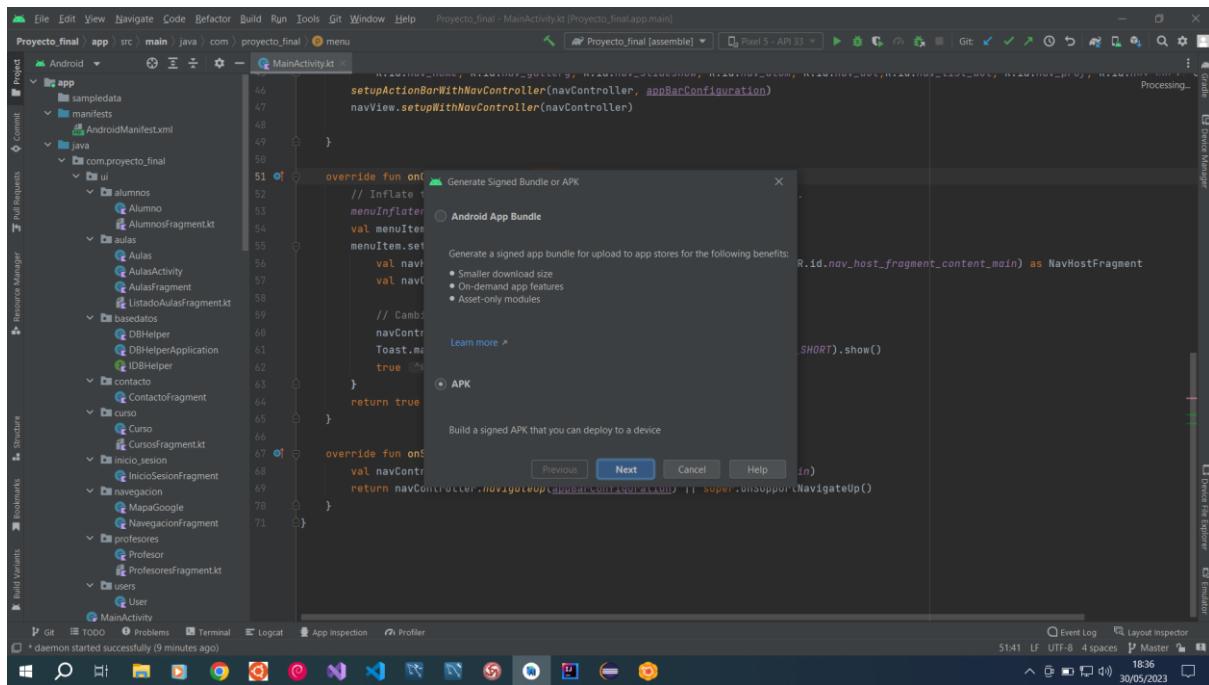


Ilustración 48: Generar APK Parte 2

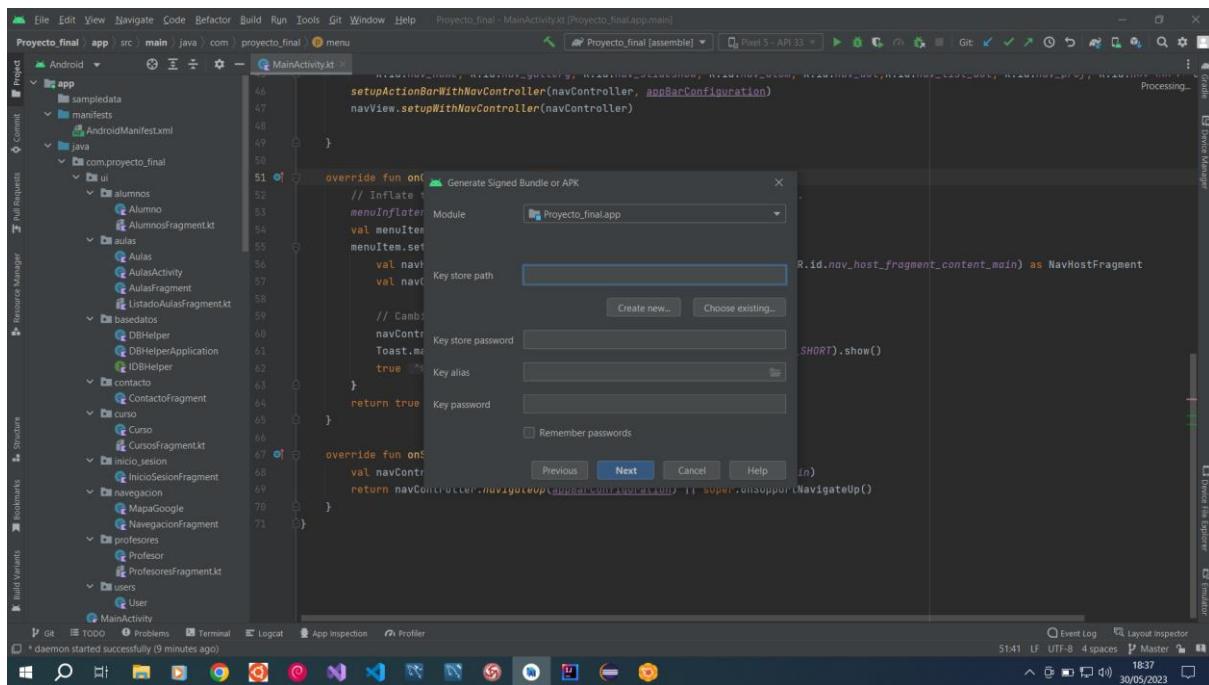


Ilustración 49: Generar APK Parte 3

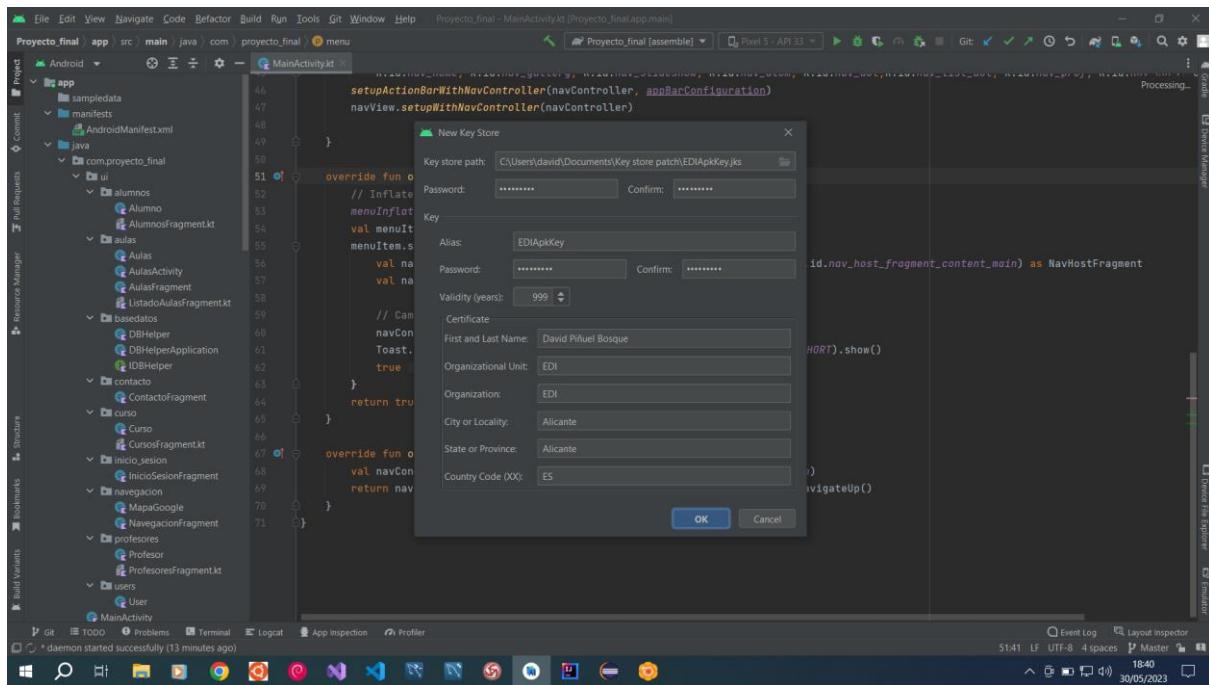


Ilustración 50: Generar APK Parte 4

Llegados a este punto, creamos el fichero donde están las credenciales que generamos para su posterior firma.

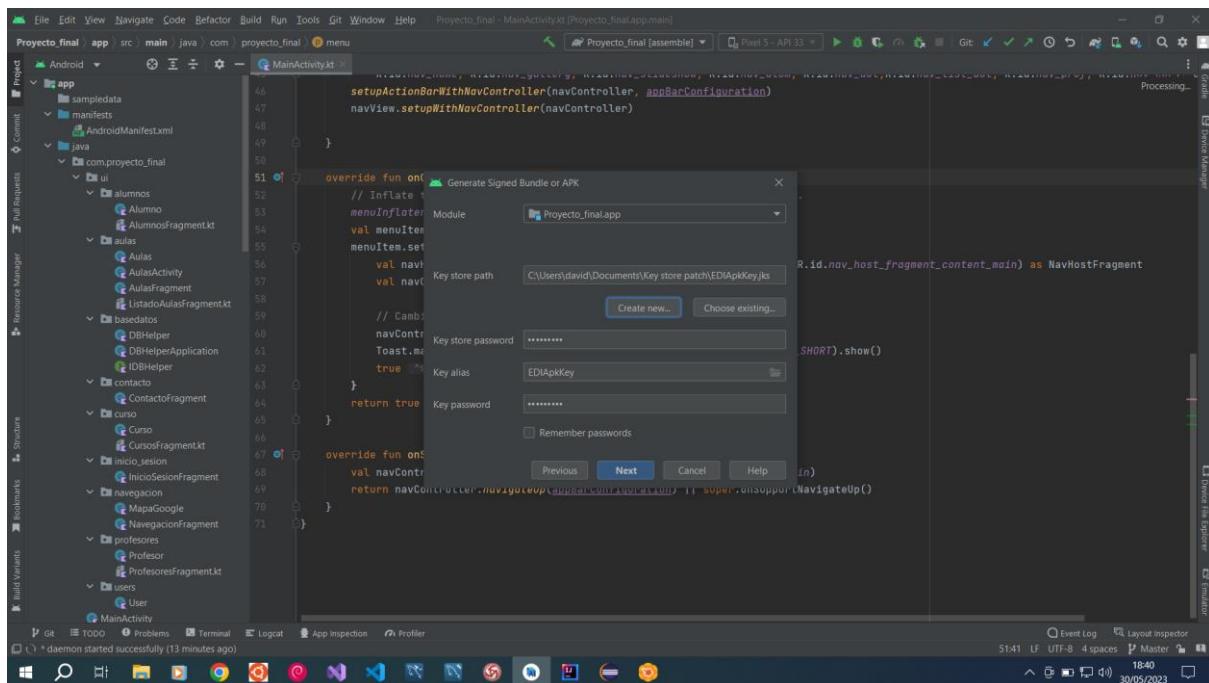


Ilustración 51: Generar APK Parte 5

Rellenamos con los datos de la aplicación y quien la realizó y aplicamos su generación.

```

    setupActionBarWithNavController(navController, appBarConfiguration)
    navView.setupWithNavController(navController)

    override fun onCreateOptionsMenu(menu: Menu): Boolean {
        // Inflate the menu; this adds items to the action bar if it is present.
        menuInflater.inflate(R.menu.main, menu)
        val menuItem = menu.findItem(R.id.action_settings)
        menuItem.setOnMenuItemClickListener {
            val navHostFragment = this.supportFragmentManager.findFragmentById(R.id.nav_host_fragment_content_main) as NavHostFragment
            val navController = navHostFragment.navController
            // Cambiar el navGraph al nuevo archivo de navegación
            navController.setGraph(R.navigation.mobile_navigation)
            Toast.makeText(context, "Cerrando Sesión...", Toast.LENGTH_SHORT).show()
            true //setOnMenuItemClickListener
        }
        return true
    }

    override fun onSupportNavigateUp(): Boolean {
        val navController = findNavController(R.id.nav_host_fragment_content_main)
        return navController.navigateUp(appBarConfiguration) || super.onSupportNavigateUp()
    }
}

```

Generate Signed APK
APK(s) generated successfully for module 'Proyecto_final.app' with 1 build variant:
Build variant 'release': locate or analyze the APK.

Ilustración 52: Generar APK Parte 6

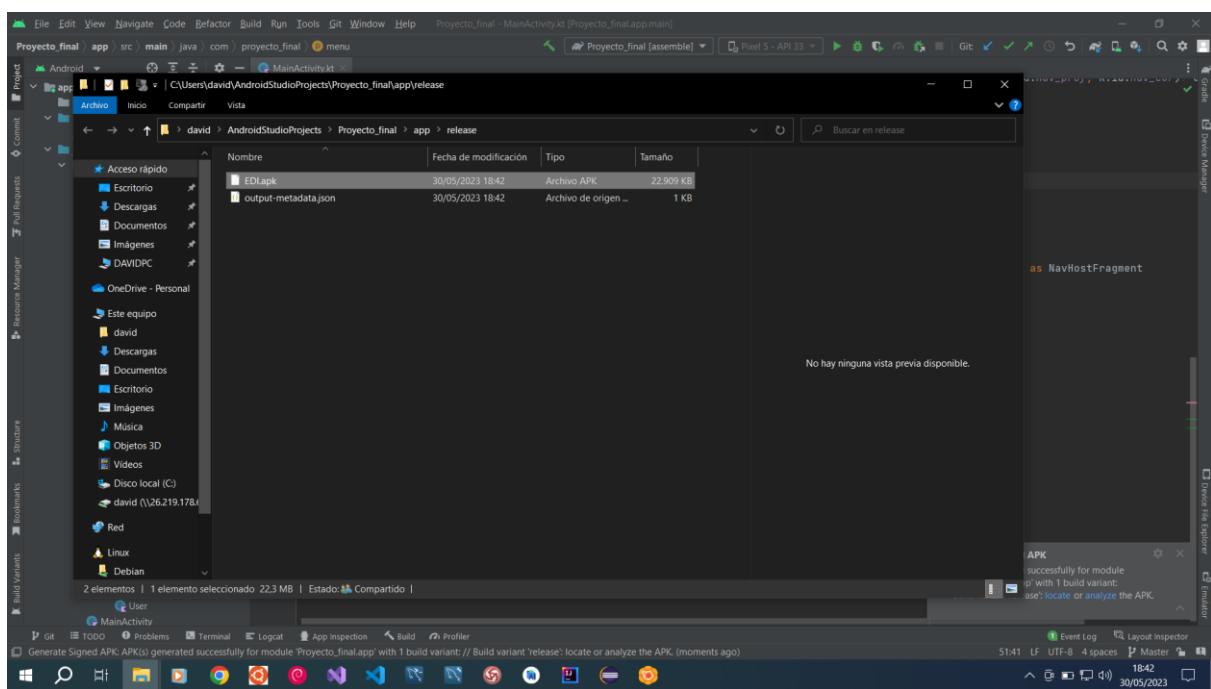


Ilustración 53: Generar APK Parte 7

12. Conclusiones

Llegados a este punto, y para concluir con todo el trabajo realizado, se procede a realizar un repaso y valoración de todo el proyecto.

Dada la gran envergadura de dicha aplicación, y que el desarrollo de ambas no ha podido ser constante a lo largo del tiempo, la primera impresión que obtengo es que el proyecto era muy ambicioso y, que haberlo realizado individualmente ha supuesto un gran reto, esto se ve traducido en toda la experiencia y conocimientos obtenidos durante las distintas fases de este.

Además, el poder aplicar tanto los conocimientos adquiridos en las distintas asignaturas del grado, como la experiencia adquirida los últimos años en el mundo laboral, han dado como resultado esta aplicación funcional que podría ser competente en el mercado actual.

En lo referente a los objetivos marcados en la propuesta, del trabajo de fin de grado, considero que se han cumplido en la medida de lo posible, esto ha permitido que haya podido realizar otras líneas de desarrollo para la mejora de la experiencia de usuario.

13. Posibles evoluciones de los proyectos

Aunque en este proyecto la aplicación ha sido desarrollada para obtener unas funcionalidades amplias, sería ambicioso tratar de expandir aún más sus funcionalidades, ya que dicha aplicación podría continuar creciendo, y evolucionando, de forma exponencial.

A continuación, se exponen una serie de puntos que hace referencia a estas posibles continuaciones de ambas aplicaciones.

13.1. Aumentar Menú con Opción a Enlaces Externos

Uno de los puntos que quiero aclarar primero, es que en el centro EDI Escuela de Informática e inglés usan bastantes enlaces a entidades externas como son LABORA y el Servef.

Añadiendo enlaces a diversas actividades donde pueden ir directamente para realizar su mantenimiento, como son tales del COPPRA para la asistencia de alumnos del Servef y documentación referente al centro para LABORA.

13.2. Aumentar la documentación al Alumno

Otra de las posibles mejoras para la aplicación podría ser la de pedir más información referente al alumno que se va a registrar en el sistema y así tenemos más datos sobre dicha persona.

14. Bibliografía

- **Android Developer.** Documentación propia de Android.
 - <https://developer.android.com/>

15. Anexo A

En este anexo se exponen los archivos que se entregan junto a esta memoria:

- I. Proyecto Android Studio
- II. Archivo Aplicación Android con extensión .APK.
- III. PowerPoint para la presentación en la defensa.

16. Anexo B

En este anexo se añaden los enlaces de los repositorios GIT, donde se encuentran las versiones más estables de la rama maestra, así como los enlaces a ambas aplicaciones, por un lado, a la Web de la aplicación Laravel y, por otro lado, al directorio compartido de Google Drive desde donde se puede descargar el archivo APK, firmado por el desarrollador, de la aplicación Android.

A continuación, se añaden las direcciones del repositorio GIT:

- https://github.com/davipibos31/PFC_FP_DAM_ANDROID

En la siguiente ruta se puede acceder al APK de la aplicación Android:

- [https://drive.google.com/file/d/1hREXnVnXMqX0Tkt8TUynC3KgPSKMsOnK/view
?usp=sharing](https://drive.google.com/file/d/1hREXnVnXMqX0Tkt8TUynC3KgPSKMsOnK/view?usp=sharing)

Para poder acceder a la aplicación Android se proporcionan las siguientes credenciales:

- Administrador:
 - Usuario: EDI
 - Contraseña: admin

El usuario es en mayúscula las 3 letras y la contraseña es en minúscula las 5 letras.