



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



**TECNOLÓGICO
NACIONAL DE MÉXICO**

Instituto Tecnológico de Pabellón de Arteaga
Departamento de Ciencias Básicas

PROYECTO DE TITULACIÓN

**APLICACIÓN MÓVIL PARA EL CONTROL Y GESTIÓN DE NOTIFICACIONES ACADÉMICAS
EN EL AULA**

**PARA OBTENER EL TÍTULO DE
INGENIERA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES**

PRESENTA:

MAYGUALI GUADALUPE MARTÍNEZ CASILLAS

ASESOR:

BENITO RODRIGUEZ CABRERA

Mayo



2023
AÑO DE
**Francisco
VILA**

EL REVOLUCIONARIO DEL PUEBLO

Agradecimientos.

Quiero aprovechar este espacio para agradecer a todas las personas que me han apoyado durante mi formación profesional a lo largo de estos años.

En primer lugar, quiero agradecer a mis padres que siempre han estado a mi lado apoyándome, en especial a mi mamá que ha sabido inyectar moral en mis peores momentos, no solo los vividos a causa de mis estudios, sino como consecuencia de la vida, a mi papa por exigirme cada vez más para ser una mejor persona. A mis hermanos Francisco, Wendy y la pequeña Cami, los cuales son uno de mis motores que me impulsan a ser mejor cada día para que siempre se sientan orgullosos de mí y ser un ejemplo a seguir. A mi esposo por confiar en mí, tenerme paciencia y estar presente en el momento de alcanzar cada una de mis metas. Y por último quiero agradecer a mi bisabuelita Celia y abuelita Coco por ponerme en sus oraciones siempre, tíos, tías y primos en general por brindarme su amor, por apoyarme, por animarme para lograr ser mejor cada día.

Quiero mostrar mi más sincero agradecimiento mis maestros Benito e Israel y a la Dra. Nivia que compartieron conmigo de sus conocimientos, por su tiempo y dedicación.

Resumen

Existe una gran problemática en la comunicación dentro del aula debido a que cada uno de los maestros transmite más de una materia con grupos grandes de alumnos y tiene la gran responsabilidad de tener una comunicación constante con cada uno de ellos. Para solucionar este inconveniente se desarrolló la aplicación móvil hecha con sistema operativo Android, la cual es dinámica, de fácil manejo y acceso.

El presente proyecto se realizó debido al nivel de importancia que tiene la comunicación que se genera dentro del aula de clase entre alumnos y docentes, de tal manera a que la interacción entre alumnos y estudiantes es uno de los recursos más indispensables para la enseñanza.

Para su desarrollo se utilizó la herramienta de Flutter, el cual se utiliza para crear aplicaciones para dispositivos móviles. La metodología utilizada para el desarrollo de la aplicación fue una metodología ágil conocida como Prototyping, la cual tiene un proceso continuo, de tal manera que cada una de las etapas realizadas dentro del proyecto ha permitido obtener los resultados necesarios para cumplir con cada uno de los objetivos planteados y así lograr un prototipo completo y aceptable. La aplicación móvil que se desarrolló lleva como nombre “Aplicación móvil para el control y gestión de las notificaciones en el aula” y tiene como propósito oficializar notificaciones entre los alumnos y maestros de diferentes materias, de tal manera que el maestro tenga un mejor control de sus materias y pueda administrar mejor sus horarios brindando comodidad, facilidad y sin necesidad de tomarle mucho tiempo a la hora de notificar a sus alumnos.

Para concluir, el proyecto desarrollado también lleva de la mano la integración de la base de datos interna de Flutter, la cual almacena cada una de las cuentas de usuario que los alumnos y los docentes crean para que estos puedan acceder correctamente a aplicación y puedan hacer uso de ella, además, la aplicación también almacena cada una de las materias que son agregadas por los docentes con su respectiva información referente a los horarios, nombre de la materia, por mencionar algunos.

índice

Agradecimientos	2
Resumen	3
índice	4
Lista de Tablas	6
Lista de Ilustraciones	7
CAPÍTULO 1: GENERALIDADES DEL PROYECTO	8
1.- Introducción	8
2. Descripción de la empresa u organización y del puesto o área del trabajo del residente	9
2.1 Organigrama del Laboratorio de Iluminación Artificial (LIA)	12
3. Problemas a resolver, priorizándolos	13
4. Justificación	13
5. Objetivos (General y Específicos)	14
Objetivo general:	14
Objetivos específicos:	14
CAPÍTULO 2: MARCO TEÓRICO	14
6.1 Importancia del control académico entre estudiantes y docentes	15
6.1.1 La gestión y el contexto educativo	15
6.1.2 Áreas de gestión	16
6.1.3 Gestión académica	17
6.2 Términos básicos	17
6.2.1 Aplicación móvil	17
6.2.2 Front- End	17
6.2.3 Back –End	18
6.3 Herramientas para la aplicación móvil	18
6.3.1 Android Studio	18
6.3.2 Flutter	19
6.4 Requisitos para la base de datos	19
6.4.1 Base de datos	20
6.4.2 SQLite	20
6.4.3 MySQL	20

CAPÍTULO 3: DESARROLLO	20
7. Metodología del desarrollo	20
8. Procedimiento y descripción de las actividades realizadas.	22
1. Definición de los objetivos del prototipo.	23
2. Validación de los requerimientos a través de la funcionalidad del producto....	23
3. Diseñar y desarrollar el prototipo	24
3.1 Mockpus del diseño	24
3.2 Código del desarrollo	28
4. Evaluar el funcionamiento del prototipo	38
CAPÍTULO 4: RESULTADOS	41
9. Resultados	41
CAPÍTULO 5: CONCLUSIONES	51
10. Conclusiones del Proyecto	51
CAPÍTULO 6: COMPETENCIAS DESARROLLADAS	52
11. Competencias desarrolladas y/o aplicadas.	52
CAPÍTULO 7: FUENTES DE INFORMACIÓN	53
12. Fuentes de información.....	53
CAPÍTULO 8: ANEXOS	55
13. Anexos.....	55
13.1 Curso de Udemý	55
13.2 Instalaciones de herramientas para el desarrollo	56
14. Registros de Productos.....	57

Lista de Tablas

Tabla 1. Cronograma de actividades	22
Tabla 2. Requerimientos.....	23
Tabla 3. Requisitos funcionales de la aplicación	45
Tabla 4. Prueba de inicio de sesión	47
Tabla 5. Prueba de materias agregadas y BDD	48
Tabla 6. Pruebas de notificaciones	49

Lista de Ilustraciones

Ilustración 1. Vista satelital del Instituto Tecnológico de Pabellón de Arteaga, Aguascalientes (22° 4' N y 102° 16' 10' O). (Fuente: Google Earth.)	9
Ilustración 2. Organigrama del Laboratorio de Iluminación Artificial (LIA)	12
Ilustración 3. Ciclo de vida del prototipo fuente propia del autor	21
Ilustración 4. Mockup_login	25
Ilustración 5. Mockup para registrar correo y contraseña.....	26
Ilustración 6. Mockup_ inicio y acceso a la clase	26
Ilustración 7. Mockup_Agregar Materias.....	27
Ilustración 8. Mocukps_notificaciones.....	27
Ilustración 9. Código de la pantalla del login_ formulario	28
Ilustración 10. Código de la pantalla del login_ validación del correo y contraseña	29
Ilustración 11. Código de la pantalla del login_ botón para ingresar y mostrar el error en pantalla	30
Ilustración 12. Código del servicio para registrar usuarios	31
Ilustración 13. Código de la pantalla para agregar materias_diseño de la pantalla	32
Ilustración 14. Código de la pantalla agregar materias_ formulario para agregar materias	33
Ilustración 15. Código de la pantalla agregar materias_ border del formulario	34
Ilustración 16. Código de conexión a la base de datos interna	35
Ilustración 17. Código para la pantalla de notificaciones.....	36
Ilustración 18. Código del servicio para recibir las notificaciones	37
Ilustración 19. Diagrama de flujo_Login	38
Ilustración 20. Diagrama de flujo_agregarmaterias	39
Ilustración 21. Diagrama de flujo_notificación.....	40
Ilustración 22. Pantalla de LOGIN.....	42
Ilustración 23. Pantalla principal.....	42
Ilustración 24. Pantalla para agregar materia	43
Ilustración 25. Pantalla de notificaciones	43
Ilustración 26. Menú para cerrar sesión	44
Ilustración 27. Pantalla con el correo y contraseña a registrar.....	46
Ilustración 28. Información almacenada en la BDD	46
Ilustración 29. Prueba de validación caso#1.....	47
Ilustración 30. Prueba de validación caso #2.....	48
Ilustración 31. Prueba de mostrar materias agregadas a la aplicación y a la base de datos	49
Ilustración 32. Prueba de enviar y recibir notificaciones.....	50
Ilustración 33. Curso de Flutter	55
Ilustración 34. Herramientas de desarrollo	56

CAPÍTULO 1: GENERALIDADES DEL PROYECTO

1.- Introducción

Actualmente las aplicaciones móviles son muy utilizadas gracias a las facilidades de acceso a internet, así como los avances tecnológicos de teléfonos inteligentes, los cuales nos permiten estar conectados como una sociedad digital y a través de las aplicaciones brindar u obtener cualquier servicio desde cualquier parte del mundo, estos cuentan con sistemas operativos que facilitan desarrollar aplicaciones gratuitas que se puede instalar en un dispositivo móvil sin ningún problema.

Existen un sin fin de aplicaciones que permiten realizar procesos en el ámbito comercial, médico, educativo y para muchas otras áreas. Se planea a través del presente proyecto desarrollar una aplicación institucional, la cual ayudará a que tanto los alumnos y maestros tengan una vía formal de comunicación. en la cual se oficialicen las notificaciones de las diferentes materias que tienen registradas en su horario escolar.

La estructura del proyecto está dividida en cuatro secciones principales, las cuales se describen a continuación.

El primer capítulo contiene una descripción de los problemas a resolver con el desarrollo de la aplicación, así como también la justificación del tema, además se mencionan los objetivos tanto como el general y los objetivos específicos los cuales determinan el alcance del proyecto.

En el segundo capítulo se desarrolla el marco teórico, el cual engloba la redacción de conceptos teóricos de todos los términos que se utilizan para una mayor comprensión técnica del proyecto, además, se describen las herramientas necesarias para el desarrollo de la aplicación tanto el lenguaje de programación utilizado, sistema operativo utilizado, entre otros.

En el tercer capítulo describe cada una de las actividades realizadas para el desarrollo de la aplicación utilizando la metodología definida para el proyecto.

Finalmente, el último capítulo contiene los resultados que se obtuvieron desde los objetivitos tanto el general y los objetivos específicos en los cuales se incorporaron algunas pruebas para poder determinar la funcionalidad de la aplicación.

2. Descripción de la empresa u organización y del puesto o área del trabajo del residente

El lugar donde se realizaron las residencias profesionales fue en el Tecnológico Nacional de México campus Pabellón de Arteaga (Ilustración 1) en el Laboratorio de Iluminación Artificial (LIA), el tecnológico fue fundado el 01 de septiembre de 2008, su ubicación es Carretera a la Estación de Rincón km 1, C.P. 20670 Pabellón de Arteaga, del estado de Aguascalientes.



Ilustración 1. Vista satelital del Instituto Tecnológico de Pabellón de Arteaga, Aguascalientes (22° 4' N y 102° 16' 10' O). (Fuente: Google Earth.)

Durante el 2017 se creó el Laboratorio de Iluminación Artificial (LIA) dentro del Tecnológico Nacional de México campus Pabellón de Arteaga, con el objetivo profundizar y clarificar los efectos de la luz (continua y pulsada) en plantas y alimentos como su caracterización biológica con herramientas tecnológicas actuales LEDs, siendo un centro capaz de poseer el nivel de competitividad que otros laboratorios existentes en el mundo.

La misión de LIA es proyectar e impulsar el desarrollo de nuevas actividades en el sector agroindustrial a través de procedimientos que ayuden a mejorar la calidad de vida alrededor del mundo, generando productos agroalimentarios con valor agregado. Así mismo, LIA cuenta con la visión de ser un centro capaz de impulsar, desarrollar y fortalecer iniciativas que incentiven la productividad y rentabilidad agroalimentaria; generando en el sector agroindustrial nuevas filosofías de competitividad y calidad.

LIA cuenta con cuatro proyectos de investigación y desarrollo tecnológico enfocados a la línea de investigación de Biosistemas Mecatrónicas y Ambientes Artificiales. A continuación, se describe brevemente cada uno:

1. Diseño, construcción e implementación de sistemas de producción multinivel en espacios cerrados con luz artificial LED, que permiten potenciar el contenido nutrimental de cultivos como Microgreens, lechuga, lenteja, espinaca, albahaca, alfalfa, entre otras, a través de recetas de luz (diferentes combinaciones de color).
2. Desarrollo de Sistemas de desinfección de alimentos con radiación ultravioleta LED tipo A, B y C aplicados a productos agroindustriales y en fresco.
3. Implementar estrategias tecnológicas con luz artificial tipo LED para preservar, extender y/o acelerar el tiempo de vida en almacén de frutas y verduras, además de analizar el efecto sobre la biosíntesis de compuestos (licopeno, capsaicina, entre otros.) durante su estancia en anaquel.
4. Investigación, integración y aplicación de la agricultura de precisión con el uso de tecnología aérea no tripulada (VANTs) para el monitoreo y detección de plagas en diferentes cultivos con el objetivo de evitar pérdidas en la producción.

La infraestructura que se encuentra en LIA son 3 cámaras de crecimiento (descripción Temperatura 12 °C a 30 °C.

- Iluminación de área de trabajo.
- Humedad: Ambiente (+5 % a 90 %). Programación día/noche de temperatura.
- Racks y accesorios para producción de cultivos comerciales.

- Sistema de caracterización de la luz (reflexión, absorción y transmisión) que incluye dos espectrofotómetros Ocean, esfera de medición de esparcimiento de luz e integradora (0.5 a 1 m) con accesorios para la manipulación de objetos en el interior.
- Un espectro radiómetro ILT950, fluorómetro WALZ, SPAD 502, sensores de radiación fotosintéticamente activa, sensores climatológicos (humedad, temperatura, CO2 y radiación).
- Controladores Lógicos Programables
- Sistema de radiación artificial tipo LED RGBW (rojo, azul, verde y blanco) de 0 a 1000 μmol de intensidad, con canal programable independiente para el color.
- Frecuencia de pulsado de 1 a 500 kHz. Con software y control independiente. Sistemas de radiación artificial equipados con UV-LED e Infrarrojo.

El lugar en el que se desarrolló el residente dentro del Laboratorio de Iluminación Artificial (LIA), fue dentro del área de desarrollo de aplicaciones móviles, teniendo la capacidad y las habilidades de razonamiento lógico utilizando los diferentes softwares, también aplicó sentido de responsabilidad, compromiso con el desarrollo. Incrementó sus habilidades para la programación a través de sus conocimientos tecnológicos y profesionales existentes para el proceso de creación y diseño de apps, con base a las reglas, estándares e indicaciones específicas establecidas por la empresa para poder realizar preparación, diseño en Flutter y la programación en el lenguaje de Dart.

2.1 Organigrama del Laboratorio de Iluminación Artificial (LIA)

El LIA cuenta con un organigrama general, el cual está organizado y estructurado jerárquicamente por puestos según su función. En la Ilustración 2 se muestra un diseño claro y fácil de comprender.

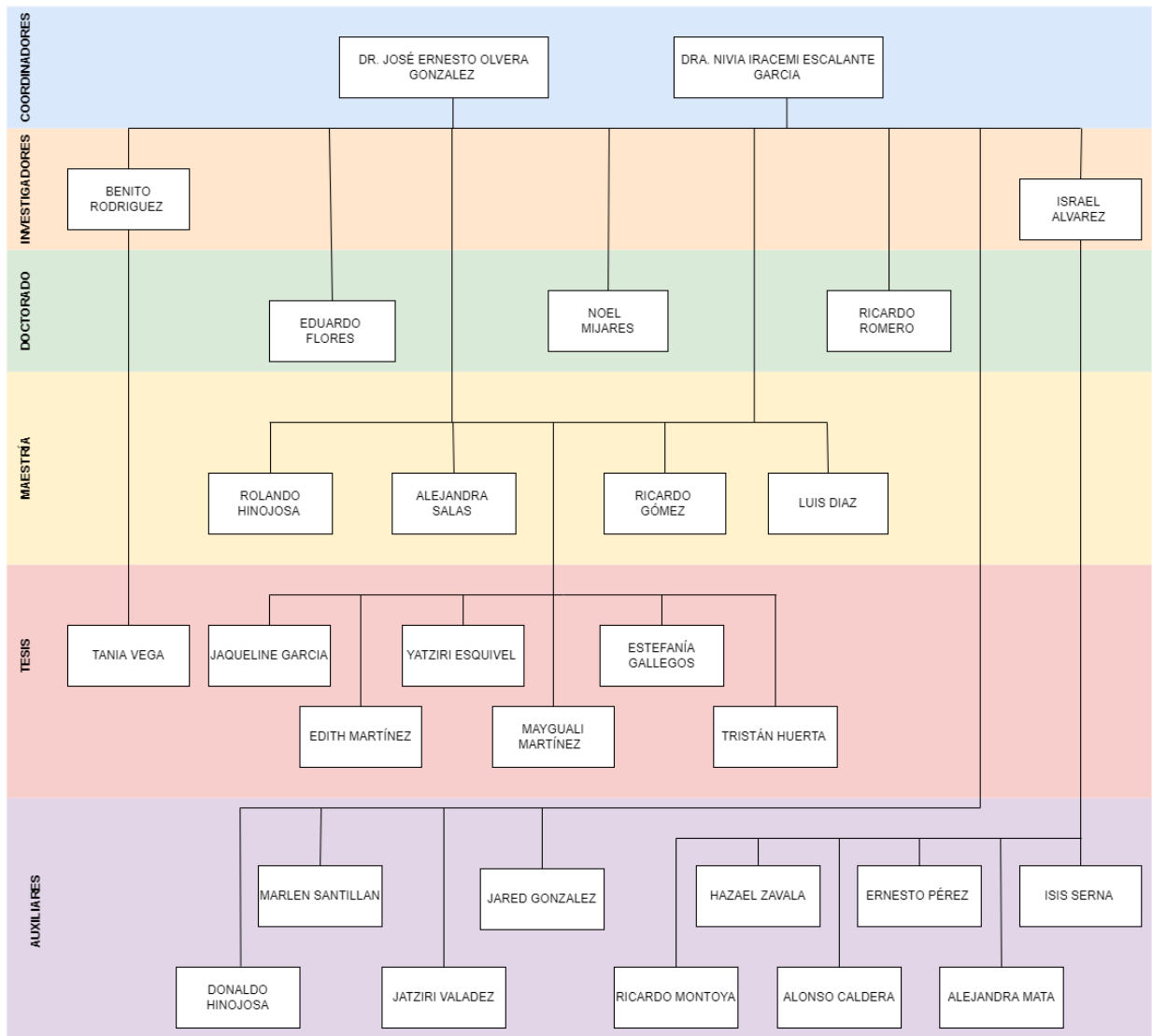


Ilustración 2. Organigrama del Laboratorio de Iluminación Artificial (LIA)

3. Problemas a resolver, priorizándolos

En la actualidad existe una gran problemática en la comunicación entre maestro y alumnos. Se tiene el conocimiento de que nuestros maestros imparten más de una materia de diversas carreras y en distintos horarios a lo largo del semestre. Por esta razón, el maestro es quien tiene la más grande responsabilidad de tener comunicación con cada uno de los grupos, educar, guiar y corregir de tal manera que para el maestro es difícil tener una comunicación directa y constante con cada uno de los alumnos a los cuales les imparte clase.

La implementación de nuevas estrategias tecnológicas que permita contribuir a mejorar la comunicación entre el profesor y los alumnos de manera que tengan una vía formal para oficializar avisos de las diferentes materias a través de la aplicación por medio de notificaciones, y que lleguen de manera instantánea. Con el fin, de evitar la necesidad de estar interactuando con la propia aplicación y dejar de estar accediendo a la web para verificar si nuestros maestros han dejado algún mensaje, asignado tareas o actividades, cambios de horarios, por mencionar algunos.

4. Justificación

En la actualidad nos encontramos con distintas formas de comunicarnos con nuestros dispositivos, así como el correo electrónico, mensajería SMS, llamadas telefónicas, por mencionar algunos. Esta aplicación móvil institucional ayuda que tanto alumnos y maestros tengan una vía formal de comunicación en la que se oficialicen las notificaciones entre los alumnos y maestros de diferentes materias a través de la aplicación, puesto que la manera más rápida de enviar y recibir notificaciones es mediante el celular. Lo que beneficiaría tanto al maestro para llevar un mejor control de sus materias, administrar sus horarios, y enviar avisos académicos de manera general hacia los alumnos a través de las notificaciones, y así los alumnos podrán recibir sus notificaciones de manera instantánea a través de su celular.

5. Objetivos (General y Específicos)

Objetivo general:

Desarrollar una herramienta de software institucional que facilite la interacción entre alumno y maestro de manera formal dentro de las instituciones, tal que ayude a notificar sobre sucesos que se ameriten en sesiones de clase o en asesorías de cualquier tipo.

Objetivos específicos:

1. Generar el conjunto de requisitos funcionales que permitan diseñar la plataforma de interacción académica entre profesor y alumno.
2. Diseñar la aplicación utilizando la plataforma de Flutter y al mismo tiempo hacer uso de la base de datos interna en lenguaje de consulta (SQL) y el término coloquial Lite por sus siglas en inglés SQLite.
3. Integrar los componentes funcionales, así como el diseño de arquitectura, la base de datos por mencionar algunos de tal manera que permitan su despliegue para su posterior evaluación.
4. Crear una base de datos en Lenguaje de Consulta Estructurado por sus siglas en inglés MySQL que nos permita almacenar y acceder los datos fiscales de los alumnos y los maestros, además de reservar cada una de las notificaciones.

CAPÍTULO 2: MARCO TEÓRICO

En este apartado se presentan los diferentes conceptos que llevaron al diseño, desarrollo e implementación del proyecto. Así como una breve introducción sobre la importancia de llevar un canal de comunicación adecuado entre el estudiante y docente.

Uno de los aspectos relevantes es la comunicación que existe entre el estudiante y docente, es decir, de qué forma el docente le da retroalimentación a cada uno de los estudiantes. Para determinar, si un alumno va con evaluaciones adecuadas para lograr acreditar sus materias. Se enlistan las características que deben ser consideradas para la evaluación.

Así mismo, el rol del docente impacta en la gran diversidad de procesos y aspectos relacionados con la mejora educativa, siendo el aprendizaje de los estudiantes, también son responsables de planificar, desarrollar e implementar estrategias didácticas e investigación educativa (Tacca, D., Tacca, A. y Alva, M. (2019); Hallinger, P., Heck, R. y Murphy, J. (2014)).

6.1 Importancia del control académico entre estudiantes y docentes

La relación entre estudiantes y docentes beneficia el aprendizaje del alumnado incrementando su rendimiento académico, es por eso que hoy en día los maestros juegan un papel importante en la educación, por este motivo, la mayor parte de su tiempo se encuentran ocupados debido a la gran cantidad de alumnos con los que este interactúa, de tal manera que necesitan de herramientas tecnológicas que les permitan ahorrar trabajo y comunicarse con rapidez con todos sus alumnos (Allcott & Gentzkow, 2017).

6.1.1 La gestión y el contexto educativo

La gestión y el contexto educativo buscan reforzar el desempeño de las escuelas por medio de distintas técnicas, elementos y conocimientos para asegurar un buen desarrollo académico de los estudiantes, estos cambian y se adaptan de acuerdo a los requerimientos de la humanidad, es por eso que actualmente con el crecimiento de los recursos en la internet, permiten crear ambientes virtuales de aprendizaje que favorecen

tanto a los alumnos como a los docentes ya que esto les facilita su manera de aprender debido a que es más flexible, abierto, innovado y cambiante de acuerdo a los nuevos requerimientos de la humanidad (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.2 Áreas de gestión

La gestión educativa está constituida por cuatro áreas de gestión: área de gestión directiva, área de gestión pedagógica y académica, área de gestión de la comunidad y área de gestión administrativa y financiera (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.2.1 Área de gestión directiva:

Es capacitado para el mejoramiento de una institución educativa, es la encargada de ejercer el liderazgo y control administrativo del plantel, a la par que la actualización y la revisión de las dinámicas recreativas, sociales y psicológicas del establecimiento (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.2.2 Área de gestión pedagógica y académica

Está centrada en los procesos de enseñanza y aprendizaje, así como la relación entre docente y alumno. Se busca que el docente tenga estrategias y métodos para la enseñanza de los contenidos a través de la comunicación, mientras que en el estudiante lo que se pretende es que desarrolle habilidades y obtenga conocimientos teóricos, que todo en suma sean aprendizajes que lo ayuden en el mercado laboral y otros ámbitos de su vida (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.2.3 Área de gestión de la comunidad

Esta área está relacionada con la parte social del acto educativo, mediante la promoción de la participación ciudadana, los planes de intercambio, de estímulo al investigador, el fomento del crecimiento profesional y otros aspectos relacionados con la comunidad educativa (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.2.4 Área de gestión administrativa y financiera

Tiene a su cargo todo lo relacionado con los sistemas de información y registro de los estudiantes, el manejo de los recursos financieros, la administración de los recursos humanos y físicos y la presentación de los servicios complementarios (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.1.3 Gestión académica

Se encarga de los procesos orientados a mejorar los proyectos educativos, para lograr que los estudiantes aprendan y desarrollen las competencias necesarias para su desarrollo personal, social y profesional garantizando una educación de calidad (Cabrera Farfán, M. T., & Reyes Adan, 2017).

6.2 Términos básicos

Para poder trabajar y entender mejor el desarrollo de la aplicación móvil es necesario comprender algunos conceptos básicos de tal manera que se conozca que ambas tienen roles diferentes y que trabajan de forma conjunta.

6.2.1 Aplicación móvil

El término de aplicación móvil tiene como propósito ejecutarse desde un dispositivo móvil, el cual el usuario puede descargar y una vez descargado puede acceder directamente. Toda aplicación móvil se compone de dos elementos, tanto el Front-End como el Back-End ninguno de estos elementos es más importante que el otro de hecho se necesita mutuamente para que las aplicaciones sean funcionales (Cuello & Vittote, 2013).

6.2.2 Front- End

Una aplicación móvil no sólo debe verse bien, sino que tiene que funcionar correctamente, por eso es que la capa de presentación Front-End se encarga de los detalles visuales tales como colores, fondos, tamaños, animaciones, imágenes, videos, efectos y de esta manera hace que el usuario pueda interactuar con nuestra aplicación o sistema (Graciela et al., 2021).

6.2.3 Back –End

Básicamente, cuando hablamos de "detrás de escena", es decir, el servidor y la base de datos que ayudan a entregar información solicitada por el usuario desde una interfaz, hablamos del back-end. Este se encarga de la manipulación de los datos, es el responsable de que la información que llega desde el front-end, sea almacenada a una base de datos (Graciela et al., 2021).

6.3 Herramientas para la aplicación móvil

Para el desarrollo de la aplicación móvil son necesarias las herramientas de desarrollo para Android además de las de Flutter. Android Studio se creó con el fin de ayudar a compilar apps de la más alta calidad y el método recomendado para la instalación de Android Studio es a través del entorno de desarrollo integrado y es compatible con tres sistemas operativos Android, Linux y OIS.

6.3.1 Android Studio

Android Studio contiene un asistente que permite instalar las herramientas del paquete de desarrollo necesarias dentro del cual se utiliza un dispositivo virtual (AVD), que tiene las mismas características de un teléfono o una Tablet Android que deseas simular en Android Emulador para poder tener una visión más real del resultado final (Studio, 2022).

6.3.2 Flutter

También se requiere del marco de trabajo Flutter, el cual se utiliza para crear aplicaciones para dispositivos móviles multiplataforma, lo que permite con un único código fuente tener una aplicación corriendo en iOS y Android. Flutter no utiliza componentes nativos, sino que viene con sus propios componentes, llamados *widgets*, los widgets son componentes que son utilizados para crear el interfaz de usuario de una aplicación y estos están anidados unos dentro de otros, por lo que la misma aplicación se verá igual en cualquier dispositivo, independientemente de su sistema operativo o la versión (InLab, 2021).

6.3.2.1 Dart

Flutter utiliza Dart como lenguaje de programación, el objetivo de Dart es ser un lenguaje de programación orientado al frontend con optimizaciones en tiempo de compilación y con soporte multiplataforma, siendo compatible con distintas plataformas como ARM, x86 o incluso la transpilación a JavaScript, siendo compatible con miles de paquetes o bibliotecas disponibles en su ecosistema, es gratuito y de código abierto (InLab, 2021).

6.4 Requisitos para la base de datos

Otra de las partes de las que está compuesta el desarrollo de aplicaciones son las bases de datos. Se requiere de una base de datos para poder guardar toda la información, modificarla, borrarla y se puede acceder a ella a través de una consulta de una forma rápida y simple desde un ordenador. Los datos que se almacenan se gestionan por un sistema de datos que suelen estar organizados por filas y columnas, dentro de una serie de tablas que se pueden relacionar unas con otra, para la base de datos interna se ha escogido utilizar SQLite y para la base de datos externa MySQL.

6.4.1 Base de datos

Una base de datos es una colección de información organizada de tal modo que sea fácilmente accesible, gestionada y actualizada. En una sola vista, las bases de datos pueden ser clasificadas de acuerdo con los tipos de contenido: bibliográfico, de puro texto, numéricas y de imágenes (Felipe, 2021).

6.4.2 SQLite

Las aplicaciones de Flutter hacen uso de las bases de datos SQLite. SQLite es una de las formas más populares de almacenar datos localmente a través del sqflite, uno de los paquetes más utilizados y actualizados para conectarse a dicha base de datos en Flutter, mostrando los conceptos básicos para insertar, leer, actualizar y eliminar datos (Flutter, 2019).

6.4.3 MySQL

El servidor de MySQL controla el acceso a los datos para garantizar el uso simultáneo de varios usuarios, para proporcionar acceso a dichos datos y para asegurarse de que solo obtienen acceso a ellos los usuarios con autorización. Por lo tanto, MySQL es un servidor multiusuario y de subprocesamiento múltiple. Utiliza SQL (del inglés Structured Query Language, Lenguaje de consulta estructurado), el lenguaje estándar para la consulta de bases de datos utilizado en todo el mundo (Palomo-Duarte et al., 2021)

CAPÍTULO 3: DESARROLLO

7. Metodología del desarrollo

Para el desarrollo de este sistema se optó por una metodología ágil conocida como Prototyping. Esta metodología se define como un método de desarrollo de sistemas en

el que un prototipo (una aproximación temprana de un sistema final o producto) es construido, probado y luego reinventado según sea necesario. El proceso continuo hasta que finalmente se logre un prototipo aceptable a partir del cual el sistema o producto completo puede ser ahora desarrollado [Rouse, 2005]. El uso de prototipos es una herramienta útil para aplicarse en casi todas las actividades de diseño y creación de software.

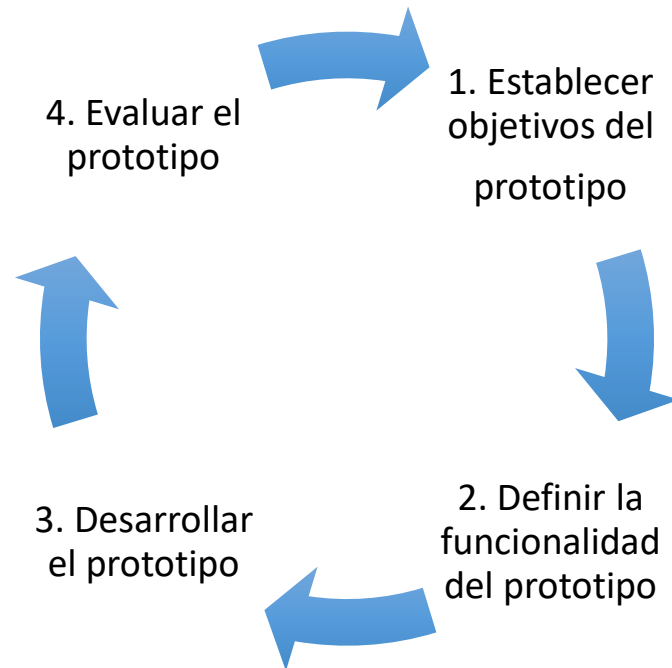


Ilustración 3. Ciclo de vida del prototipo fuente propia del autor

Para el caso de las cuestiones tácticas del presente trabajo, el prototipo diseñado se basará en las especificaciones de los requerimientos bajo los supuestos de:

1. Definición de los objetivos del prototipo.

En esta etapa se realizó un análisis el cual se pide al usuario describir cada uno de los requerimientos con los que debe cumplir el sistema, puesto que a estos se tiene la capacidad necesaria para resolver un problema o alcanzar el objetivo.

2. Validación de los requerimientos a través de la funcionalidad del producto.

En esta etapa se realizó la documentación correspondiente al análisis de la funcionalidad de los requerimientos a través de una tabla de requerimientos.

3. Diseñar y desarrollar el prototipo.

En esta etapa se utilizó el software de Draw.io para el diseño de las pantallas de la aplicación, además Visual Studio Code para la construcción de la aplicación.

La funcionalidad será programada a través de la plataforma Flutter, MySQL y al mismo tiempo hacer uso de la base de datos interna SQLite.

4. Evaluar el funcionamiento del prototipo.

En esta etapa se realizaron pruebas necesarias para determinar la funcionalidad de un sistema, en donde se comprueba si este cumple con todos los requerimientos que el usuario solicitó, si cumple su función, si su manejo es correcto, etc. Se diseñaron diagramas de flujo los cuales nos ayudaron a determinar la funcionalidad de la aplicación.

8. Procedimiento y descripción de las actividades realizadas.

En este apartado se dan a conocer el desglose de actividades que se llevaron a cabo para diseño de la aplicación móvil para el control y gestión de las notificaciones académicas en el aula, cada una de estas actividades nos ayudaron a lograr un buen desarrollo del proyecto, estas actividades se realizaron paso a paso tal y como se muestra en la siguiente Tabla 1.

Tabla 1. Cronograma de actividades

Actividades	Agosto	Septiembre	Octubre	Noviembre	Diciembre
1. Definición de los objetivos del prototipo.					

2. Validación de los requerimientos a través de la funcionalidad del producto.					
3. Diseñar y desarrollar el prototipo.					
4. Evaluar el funcionamiento del prototipo.					

Tabla 1. Cronograma de actividades

1. Definición de los objetivos del prototipo.

Uno de los elementos clave en el desarrollo de un proyecto es determinar cuáles son las metas que queremos conseguir con dicho proyecto, es por ello que se definen los objetivos para que nos sirvan de guía durante todo el desarrollo del proyecto. Para la definición de los objetivos del prototipo se elaboró un objetivo general y enseguida los objetivos específicos, para el objetivo general, se plasmó la idea central de cómo es que se va a diseñar la aplicación para el control y gestión de notificaciones en el aula y de acuerdo al objetivo general, se desglosaron los objetivos específicos en forma de lista en donde se determinaron tareas muy específicas las cuales nos van ayudar a cumplir con el objetivo general mostrado en el apartado 5. Objetivos (General y Específicos).

2. Validación de los requerimientos a través de la funcionalidad del producto.

Para la recopilación de los requerimientos del desarrollo de la aplicación móvil para el control y gestión de notificaciones en el aula, se programó una video llamada, en la cual se trataron aspectos muy importantes como lo es la definición de los que requerimientos para determinar la funcionalidad del prototipo los cuales son mostrados en la Tabla 2.

Tabla 2. Requerimientos

Modulo	Identificador	Descripción	Prioridad
Aplicación para el control	RF-1	El docente creará su cuenta (Correo y contraseña)	Alta
	RF-2	El docente iniciará sesión (Correo y Contraseña)	Alta

	RF-3	El docente dará de alta las materias	Alta
	RF-4	El docente enviará notificaciones a los alumnos	Alta
	RF-5	El docente recibirá notificaciones de sus alumnos	Alta
	RF-6	El alumno creará su cuenta (Correo y contraseña)	Alta
	RF-8	El alumno iniciará sesión (Correo y Contraseña)	Alta
	RF-9	El alumno deberá inscribirse a la materia	Alta
	RF-10	El alumno recibirá notificaciones de su docente	Alta
	RF-11	El alumno enviará notificaciones a su docente	Alta

3. Diseñar y desarrollar el prototipo

Con el propósito de aprender el lenguaje de programación Dart de cero y las otras plataformas seleccionadas hacia el desarrollo de la aplicación para el control y la gestión de notificaciones en el aula se tomó un curso de Udemy llamado “Flutter: Tu guía completa de desarrollo para IOS y Android” ver Ilustración 33, este curso tuvo como objetivo enseñar todas las herramientas básicas para poder crear aplicaciones funcionales aplicables a la vida real. Enseguida se realizó la instalación de los programas a utilizar Ilustración 34 para el desarrollo del prototipo.

3.1 Mockpus del diseño

El diseño de la aplicación se generó a través de los requerimientos solicitados y mencionados anteriormente Tabla 2. Requerimientos, por medio del software Draw.io.

En la Ilustración 4 se observa la pantalla del Login en donde los usuarios podrán acceder a la pantalla de inicio de la aplicación.



Ilustración 4. Mockup_login

Si el usuario aún no tiene una cuenta para poder acceder a la pantalla de inicio de la aplicación tendrá que dar clic en crear una nueva cuenta y aparecerá la pantalla de la Ilustración 5, en la cual tendrá que introducir su correo y asignar una contraseña, para después poder hacer el Login correctamente.

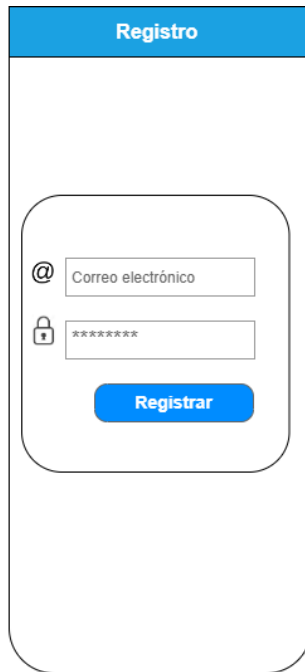


Ilustración 5. Mockup para registrar correo y contraseña

Cuando el usuario hace Login correctamente, se dirigirá a la pantalla de inicio Ilustración 6 en la cual se puede observar las materias agregadas por el maestro y así mismo tendrá que dar clic para unirse a la clase deseada.

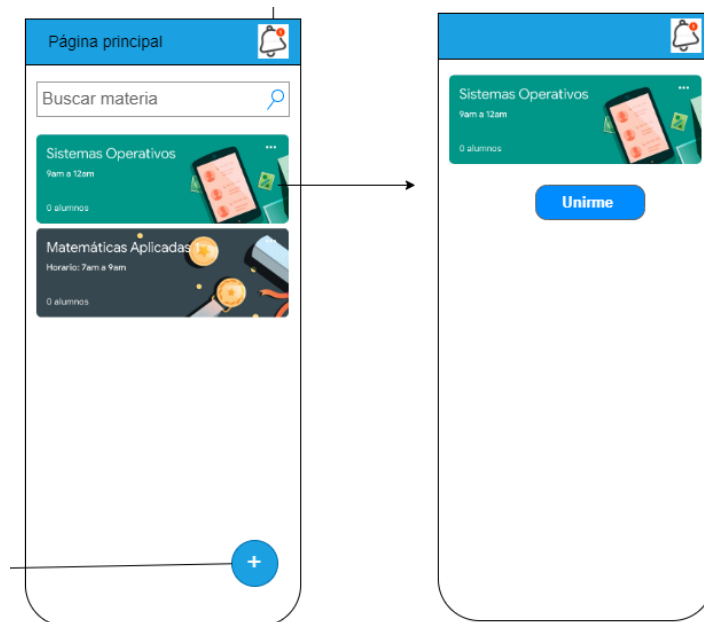


Ilustración 6. Mockup_ inicio y acceso a la clase

Para agregar las materias se elaboró un formulario el cual deberá ser llenado por el docente ver Ilustración 7.

The mockup shows a mobile application screen with a blue header containing the text 'Añadir una materia'. Below the header, there are three text input fields. The first is labeled 'Nombre de la materia' and contains the placeholder text 'Nombre de la materia'. The second is labeled 'Horario' and contains the placeholder text 'Horario'. The third is labeled 'Nombre del maestro' and contains the placeholder text 'Nombre del maestro'. At the bottom center of the form is a blue button with the white text 'Añadir'.

Ilustración 7. Mockup_Agregar Materias

Así mismo para recibir y enviar las notificaciones se crearon las siguientes pantallas ver Ilustración 8.

The illustration shows two mobile application screens. The left screen has a blue header with the text 'Notificaciones' and a notification bell icon. Below the header is a list of five items, each labeled 'Notificación 1' through 'Notificación 5'. The right screen has a blue header and contains a form for sending a notification. It includes a text input field for 'Titulo de la notificación' with the placeholder 'Titulo de la notificación', a larger text input field for 'Texto' with the placeholder 'Texto', and two text input fields for 'Nombre de la materia' and 'Nombre del maestro', both with the placeholder 'Nombre de la materia'. A blue button with the white text 'Enviar' is located at the bottom center. An arrow points from the notification bell icon on the left screen to the right screen. A small upward-pointing arrow is located at the bottom center of the left screen.

Ilustración 8. Mocukps_notificaciones

3.2 Código del desarrollo

Para el desarrollo de la aplicación se utilizó Visual Studio Code con el fin de escribir el código de cada una de las pantallas de la aplicación para después mostrarlas en el dispositivo virtual de Android Studio.

Para poder desarrollar y diseñar la pantalla de LOGIN, se escribieron las siguientes líneas de código ver Ilustración 9, Ilustración 10, e Ilustración 11 las cuales nos ayudaron a diseñar el formulario para que el usuario pudiera iniciar sesión y colocar las validaciones del correo y contraseña para que el usuario pudiera acceder correctamente a la pantalla principal aplicación.

```
1 import 'package:app_notificaciones/providers/login_form.providers.dart';
2 import 'package:app_notificaciones/services/services.dart';
3 import 'package:app_notificaciones/ui/input_decorations.dart';
4 import 'package:app_notificaciones/widgets/widgets.dart';
5 import 'package:flutter/material.dart';
6 import 'package:provider/provider.dart';
7
8 class LoginScreen extends StatelessWidget {
9   @override
10  Widget build(BuildContext context) {
11    return Scaffold(
12      body: AuthBackground(
13        child: SingleChildScrollView(
14          child: Column(children: [
15            SizedBox(height: 300),
16
17            ///////////////////////////////////////////////////AQUÍ EMPIEZA LA TARJETA DEL FORMULARIO
18            CardContainer(
19              child: Column(
20                children: [
21                  SizedBox(height: 10),
22                  Text('LOGIN', style: Theme.of(context).textTheme.headline4),
23                  SizedBox(height: 30),
24                  ChangeNotifierProvider(
25                    create: (_) => LoginFormProvider(), child: _LoginForm() // ChangeNotifierProvider
26                  ],
27                ), // Column // CardContainer
28              ),
29              SizedBox(height: 100),
30              TextButton(
31                onPressed: () => Navigator.pushReplacementNamed(context, 'register'),
32                child: Text('Crear una nueva cuenta',
33                  style: TextStyle(color: Colors.black, fontSize: 20)), // Text
34              ), // TextButton
35
36              SizedBox(height: 100)
37            ], // Column
38          )); // SingleChildScrollView // AuthBackground // Scaffold
39    }
40  }
```

Ilustración 9. Código de la pantalla del login_formulario

```

40
41 class _LoginForm extends StatelessWidget {
42   @override
43   Widget build(BuildContext context) {
44     final loginForm = Provider.of<LoginFormProvider>(context);
45     return Container(
46       child: Form(
47         key: loginForm.formKey,
48         //TODO: mantener la referencia al KEY
49
50         ///
51         autovalidateMode: AutovalidateMode.onUserInteraction,
52
53         ///
54         child: Column(
55           children: [
56             TextFormField(
57               autocorrect: false,
58               keyboardType: TextInputType.emailAddress,
59               decoration: InputDecorations.authInputDecoration(
60                 hintText: 'correo123@gmail.com',
61                 labelText: 'Correo electrónico',
62                 prefixIcon: Icons.alternate_email_rounded),
63               onChanged: (value) => loginForm.email = value,
64               validator: (value) {
65                 String pattern =
66                   r'^(([^<>()[\]\\\.,;:\s@\\"]+(\.[^<>()[\]\\\.,;:\s@\\"]+)*)(\.[^<>()[\]\\\.,;:\s@\\"]+))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\)|((\[[a-zA-Z]{1,3}\.|\.[a-zA-Z]{2,})$)';
67                 RegExp regExp = new RegExp(pattern);
68
69                 return regExp.hasMatch(value ?? '')
70                   ? null
71                   : 'El valor ingresado no luce como un correo';
72               },
73             ), // TextFormField
74             SizedBox(height: 20),
75             TextFormField(
76               autocorrect: false,
77               obscureText: true,
78               keyboardType: TextInputType.emailAddress,

```

Ilustración 10. Código de la pantalla del login_ validación del correo y contraseña

```
92     shape: RoundedRectangleBorder(  
93       borderRadius: BorderRadius.circular(10)), // RoundedRectangleBorder  
94     disabledColor: Colors.grey,  
95     elevation: 0,  
96     color: Colors.blue,  
97     child: Container(  
98       padding:  
99         EdgeInsets.symmetric(horizontal: 80, vertical: 15),  
100       child: Text(  
101         loginForm.isLoading ? 'Espere...' : 'Ingresar',  
102         style: TextStyle(color: Colors.white, fontSize: 18),  
103       )), // Text // Container  
104     onPressed: loginForm.isLoading  
105       ? null  
106       : () async {  
107         //TODO: Login form  
108         FocusScope.of(context).unfocus();  
109         final authService =  
110           Provider.of<AuthService>(context, listen: false);  
111         if (!loginForm.isValidForm()) return;  
112         loginForm.isLoading = true;  
113  
114         //validar si el login es correcto (FALTA  
115         final String? errorMessage = await authService.login(  
116           loginForm.email, loginForm.password);  
117  
118         if (errorMessage == null) {  
119           Navigator.pushReplacementNamed(context, 'inicio');  
120         } else {  
121           // TODO: MOSTRAR ERROR EN PANTALLA  
122           print(errorMessage);  
123           NotificationsService.showSnackBar(errorMessage);  
124           loginForm.isLoading = false;  
125         }  
126       }) // MaterialButton  
127     ],  
128   )), // Column // Form  
129 ); // Container  
130 }  
131 }
```

Ilustración 11. Código de la pantalla del login_ botón para ingresar y mostrar el error en pantalla

Para poder desarrollar y diseñar la pantalla de crear una nueva cuenta, se reutilizó el mismo código del LOGIN, solo se enlazó con un servicio Ilustración 12 el cual permite que los usuarios que se registren se guarden en la base de datos interna.

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:flutter_secure_storage/flutter_secure_storage.dart';
import 'package:http/http.dart' as http;

class AuthService extends ChangeNotifier {
  final String _baseUrl = 'identitytoolkit.googleapis.com';
  final String _firebaseToken = 'AIzaSyAJABL1Bfr6TK2RcXSaaR2d8w_umK8SP9Y';

  final storage = new FlutterSecureStorage();

  // si retornamos algo es un error Si no todo bien

  Future<String?> createUser(String email, String password) async {
    final Map<String, dynamic> authData = {
      'email': email,
      'password': password,
    };

    final url =
      Uri.https(_baseUrl, '/v1/accounts:signUp', {'key': _firebaseToken});
    final resp = await http.post(url, body: json.encode(authData));
    final Map<String, dynamic> decodedResp = json.decode(resp.body);
    if (decodedResp.containsKey('idToken')) {
      await storage.write(key: 'token', value: decodedResp['idToken']);
      // Token hay que guardarlo en un lugar seguro
      return null;
    } else {
      return decodedResp['error']['message'];
    }
  }

  Future<String?> login(String email, String password) async {
    final Map<String, dynamic> authData = {
      'email': email,
      'password': password,
    };

    final url = Uri.https(
      _baseUrl, '/v1/accounts:signInWithPassword', {'key': _firebaseToken});
```

Ilustración 12. Código del servicio para registrar usuarios

Para poder desarrollar y diseñar la pantalla de para agregar las materias, se escribieron las siguientes líneas de código Ilustración 13, Ilustración 14 e Ilustración 15 las cuales nos ayudaron a diseñar el formulario para que el docente pueda agregar cada una de sus materias que impartirá.

```

1 import 'package:flutter/material.dart';
2 import 'package:provider/provider.dart';
3 import 'package:app_notificaciones/providers/materia_form_provider.dart';
4 import 'package:app_notificaciones/services/services.dart';
5 import 'package:app_notificaciones/ui/input_decorations.dart';
6
7 class agregarmateriasScreen extends StatelessWidget {
8   @override
9   Widget build(BuildContext context) {
10     final materiaServices = Provider.of<MateriasService>(context);
11     return ChangeNotifierProvider(
12       create: (_) => MateriaFormProvider(materiaServices.selectedMateria),
13       child: _AgregarmateriasScreenBody(materiaServices: materiaServices),
14     ); // ChangeNotifierProvider
15   }
16 }
17
18 class _AgregarmateriasScreenBody extends StatelessWidget {
19   const _AgregarmateriasScreenBody({
20     Key? key,
21     required this.materiaServices,
22   }) : super(key: key);
23
24   final MateriasService materiaServices;
25
26   Widget build(BuildContext context) {
27     final materiaForm = Provider.of<MateriaFormProvider>(context);
28     return Scaffold(
29       appBar: AppBar(
30         elevation: 10,
31         // texto de la barra
32         title: const Text('Agregar Materias',
33           style: TextStyle(color: Colors.black, fontSize: 24)), // Text
34         centerTitle: true,
35         backgroundColor: Color.fromARGB(255, 110, 236, 255),
36       ), // AppBar
37       body: SingleChildScrollView(
38         keyboardDismissBehavior: ScrollViewKeyboardDismissBehavior.onDrag,
39         child: Column(
40           children: [
41             SizedBox(
42               height: 300,
43             ), // SizedBox
44             Stack(
45               children: [
46                 Stack(
47                   children: [
48                     _MateriaForm(),
49                     SizedBox(height: 100),
50                   ],
51                 ), // Column
52             ), // SingleChildScrollView
53             floatingActionButton: FloatingActionButton(
54               child: Icon(Icons.save_outlined),
55               onPressed: () async {
56                 if (!materiaForm.isValidForm()) return;
57                 await materiaServices.saveOrCreateMateria(materiaForm.materia);
58               },
59             ), // FloatingActionButton
60           ]; // Scaffold

```

Ilustración 13. Código de la pantalla para agregar materias_diseño de la pantalla


```

64 class _MateriaForm extends StatelessWidget {
65   @override
66   Widget build(BuildContext context) {
67     final materiaForm = Provider.of<MateriaFormProvider>(context);
68     final materia = materiaForm.materia;
69     return Padding(
70       padding: EdgeInsets.symmetric(horizontal: 10),
71       child: Container(
72         padding: EdgeInsets.symmetric(horizontal: 20),
73         width: double.infinity,
74         decoration: _buildBoxDecoration(),
75         child: Form(
76           key: materiaForm.formKey,
77           autovalidateMode: AutovalidateMode.onUserInteraction,
78           child: Column(
79             children: [
80               SizedBox(
81                 height: 10,
82               ), // SizedBox
83               TextFormField(
84                 initialValue: materia.materia,
85                 onChanged: (value) => materia.materia = value,
86                 validator: (value) {
87                   if (value == null || value.length < 1)
88                     return 'El nombre de la materia es bbligatorio';
89                 },
90                 style: TextStyle(color: Colors.black, fontSize: 20),
91                 decoration: InputDecorations.authInputDecoration(
92                   hintText: 'Nombre de la materia',
93                   labelText: 'Nombre de la materia:'), // TextFormField
94               SizedBox(height: 30),
95               TextFormField(
96                 initialValue: materia.maestro,
97                 onChanged: (value) => materia.maestro = value,
98                 validator: (value) {
99                   if (value == null || value.length < 1)
100                     return 'El nombre del maestro es obligatorio';
101                 },
102                 style: TextStyle(color: Colors.black, fontSize: 20),
103                 decoration: InputDecorations.authInputDecoration(
104                   hintText: 'Nombre del maestro',
105                   labelText: 'Nombre del maestro:'), // TextFormField
106               SizedBox(
107                 height: 30,
108               ), // SizedBox
109               TextFormField(
110                 initialValue: materia.horario,
111                 onChanged: (value) => materia.horario = value,
112                 validator: (value) {
113                   if (value == null || value.length < 1)
114                     return 'El horario de la clase es obligatorio';
115                 },
116                 style: TextStyle(color: Colors.black, fontSize: 20),
117                 decoration: InputDecorations.authInputDecoration(
118                   hintText: 'Horario', labelText: 'Horario de la clase:'), // TextFormField
119               SizedBox(
120                 height: 30,
121               ), // SizedBox
122               SwitchListTile.adaptive(
123                 value: materia.habilitado,
124                 title: Text('Activa para inscribirte'),
125                 activeColor: Colors.green,
126                 onChanged: materiaForm.updateAvailability), // SwitchListTile.adaptive
127             ],
128           ), // Column // Form
129         ), // Container
130       ); // Padding
131   }
132 }

```

Ilustración 14. Código de la pantalla agregar materias_formulario para agregar materias

```
132
133   BoxDecoration _buildBoxDecoration() => BoxDecoration(
134     color: Colors.white,
135     borderRadius: BorderRadius.only(
136       bottomLeft: Radius.circular(25),
137       bottomRight: Radius.circular(25)), // BorderRadius.only
138     boxShadow: [
139       BoxShadow(
140         color: Colors.black.withOpacity(0.05),
141         offset: Offset(0, 5),
142         blurRadius: 5,
143       ) // BoxShadow
144     ]); // BoxDecoration
145 }
146
```

Ilustración 15. Código de la pantalla agregar materias_border del formulario

Una vez que se hizo el diseño y el desarrollo de la pantalla para agregar las materias se realizó la conexión con la base de datos interna, en la cual se escribieron las siguientes líneas de código Ilustración 16 que nos ayudaron a que las materias registradas por el docente se almacenarán de forma segura en la base de datos.

```
5 import 'dart:convert';
6
7 class Materias {
8   Materias(
9     {required this.habilitado,
10    required this.horario,
11    required this.maestro,
12    required this.materia,
13    this.id});
14
15   bool habilitado;
16   String horario;
17   String maestro;
18   String materia;
19   String? id;
20
21   factory Materias.fromJson(String str) => Materias.fromMap(json.decode(str));
22
23   String toJson() => json.encode(toMap());
24
25   factory Materias.fromMap(Map<String, dynamic> json) => Materias(
26     habilitado: json["habilitado"],
27     horario: json["horario"],
28     maestro: json["maestro"],
29     materia: json["materia"],
30   );
31
32   Map<String, dynamic> toMap() => {
33     "habilitado": habilitado,
34     "horario": horario,
35     "maestro": maestro,
36     "maestro": maestro,
37     "materia": materia,
38   };
39
40   Materias copy() => Materias(
41     habilitado: this.habilitado,
42     horario: this.horario,
43     maestro: this.maestro,
44     materia: this.materia,
45     id: this.id,
46   );
47 }
```

Ilustración 16. Código de conexión a la base de datos interna

Y por último para poder desarrollar y diseñar la pantalla de recibir notificaciones, se escribieron las siguientes líneas de código Ilustración 17 las cuales nos ayudaron a crear un espacio para que las notificaciones puedan mostrarse en dicha pantalla

```
1 import 'package:flutter/material.dart';
2
3 class notificacionesScreen extends StatelessWidget {
4   const notificacionesScreen({Key? key}) : super(key: key)
5   @override
6   Widget build(BuildContext context) {
7     final args = ModalRoute.of(context)?.settings.arguments ?? 'No data ';
8     return Scaffold(
9       appBar: AppBar(
10        elevation: 10,
11        title: const Text('Notificaciones',
12          style: TextStyle(color: Colors.black, fontSize: 24)),
13        centerTitle: true,
14        backgroundColor: Color.fromARGB(255, 110, 180, 236),
15        actions: <Widget>[
16          IconButton(
17            icon: Icon(
18              Icons.notification_add,
19              size: 40,
20            ),
21            onPressed: () {
22              {
23                Navigator.pushNamed(context, 'redactarnoti');
24              }
25            }], ],
26        body: Center(
27          child: Text(
28            '$args',
29            style: TextStyle(fontSize: 24),
30          ),
31        ),
32      );
33    }
34  }
35 }
```

Ilustración 17. Código para la pantalla de notificaciones

Una vez que se diseñó la pantalla de notificaciones se enlazó con un servicio Ilustración 18 el cual nos ayudó a que nuestras notificaciones fueran mostradas de tal manera que la aplicación estuviese abierta, cerrada o en manipulación.

```
7
8 class PushNotificationService {
9     static FirebaseMessaging messaging = FirebaseMessaging.instance;
10    static String? token;
11    static StreamController<String> _messageStream =
12    new StreamController.broadcast();
13
14    static Stream<String> get messagesStream => _messageStream.stream;
15
16    static Future _backgroundHandler(RemoteMessage message) async {
17        //print('onBackground Handler ${message.messageId}');
18        print(message.data);
19        _messageStream.add(message.data['mensaje'] ?? 'Sin mensajes');
20    }
21
22    static Future _onMessageHandler(RemoteMessage message) async {
23        //print('onMessage Handler ${message.messageId}');
24        print(message.data);
25        _messageStream.add(message.data['mensaje'] ?? 'Sin mensajes');
26    }
27
28    static Future _onMessageOpenApp(RemoteMessage message) async {
29        //print('onMessageOpenApp Handler ${message.messageId}');
30        print(message.data);
31        _messageStream.add(message.data['mensaje'] ?? 'Sin mensajes');
32    }
33
34    static Future initializeApp() async { // Push Notifications
35        await Firebase.initializeApp();
36        token = await FirebaseMessaging.instance.getToken();
37        print('Token: $token');
38        FirebaseMessaging.onBackgroundMessage(_backgroundHandler); //Handlers
39        FirebaseMessaging.onMessage.listen(_onMessageHandler);
40        FirebaseMessaging.onMessageOpenedApp.listen(_onMessageOpenApp);
41        // Local Notifications
42    }
43
44    static closeStreams() {
45        _messageStream.close();
46    }
47 }
```

Ilustración 18. Código del servicio para recibir las notificaciones

4. Evaluar el funcionamiento del prototipo

Para evaluar el funcionamiento del prototipo se diseñaron diagramas de flujo con la finalidad de tener una mejor visión del seguimiento de cada una de las acciones a realizar dentro de cada una de las pantallas y estos diagramas se muestran a continuación:

En el primer diagrama de flujo Ilustración 19 se describe el ingreso del usuario a la aplicación escribiendo su correo e introduciendo su contraseña y en el otro caso se describe cómo es que el usuario tiene que crear su cuenta de tal manera esté pueda ingresar a la aplicación.

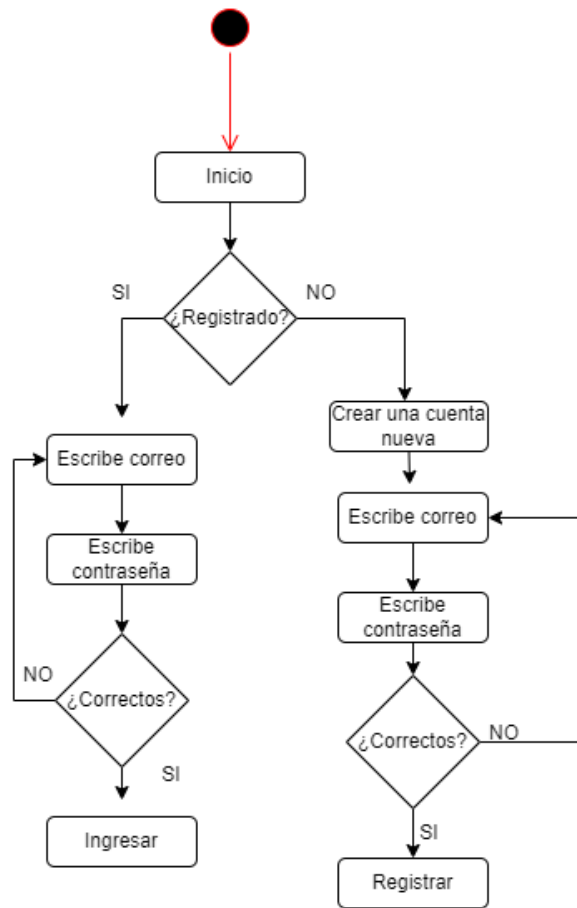


Ilustración 19. Diagrama de flujo_Login

En el segundo diagrama de flujo Ilustración 20 permite ingresar los datos de la materia que se está integrando a la aplicación de tal manera que los alumnos puedan seleccionar para inscribirse.

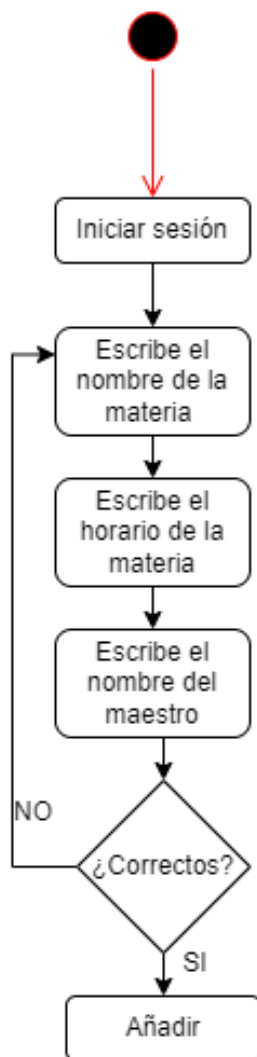


Ilustración 20. Diagrama de flujo_agregarmaterias

El tercer diagrama de flujo Ilustración 21 permite redactar la notificación que será enviada para que esta pueda ser mostrada en la pantalla de notificaciones de la aplicación.

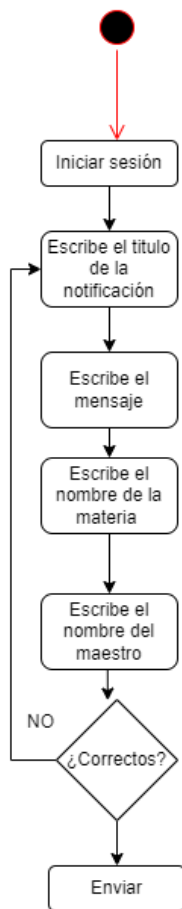


Ilustración 21. Diagrama de flujo_notificación

CAPÍTULO 4: RESULTADOS

9. Resultados

Los resultados obtenidos fueron tomados de cada uno de los objetivos específicos, de tal manera que estos fueran cumpliendo con el objetivo general. Y los resultados alcanzados del objetivo general se muestran cada una de las pantallas que forman parte de la aplicación.

9.1 Resultados del objetivo general

La Ilustración 22 tiene como objetivo realizar el ingreso al sistema para que los usuarios registren sus datos como correo y contraseña, la Ilustración 23 se diseñó con el fin de tener un punto de inicio para acceder a todas las funcionalidades de la aplicación, la Ilustración 24 tiene como objetivo agregar las materias a la pantalla principal para que los docentes registren los datos de la materia como el horario, el nombre de las materias, por mencionar algunos, la Ilustración 25 se diseñó con la intención de tener una pantalla para que las notificaciones recibidas puedan ser vistas y por último en la pantalla principal se insertó un menú con el icono de salir para cerrar la sesión Ilustración 26.



Ilustración 22. Pantalla de LOGIN

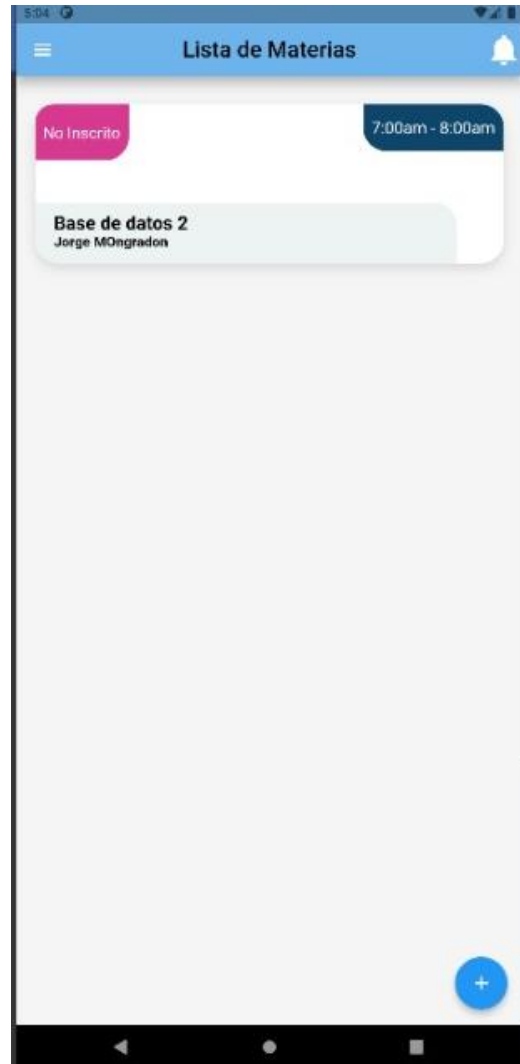


Ilustración 23. Pantalla principal

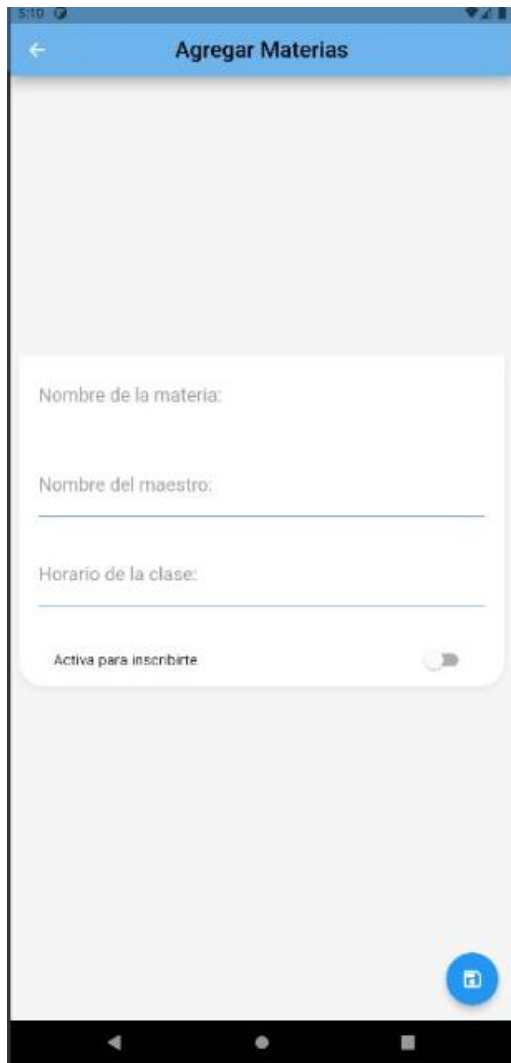


Ilustración 24. Pantalla para agregar materia

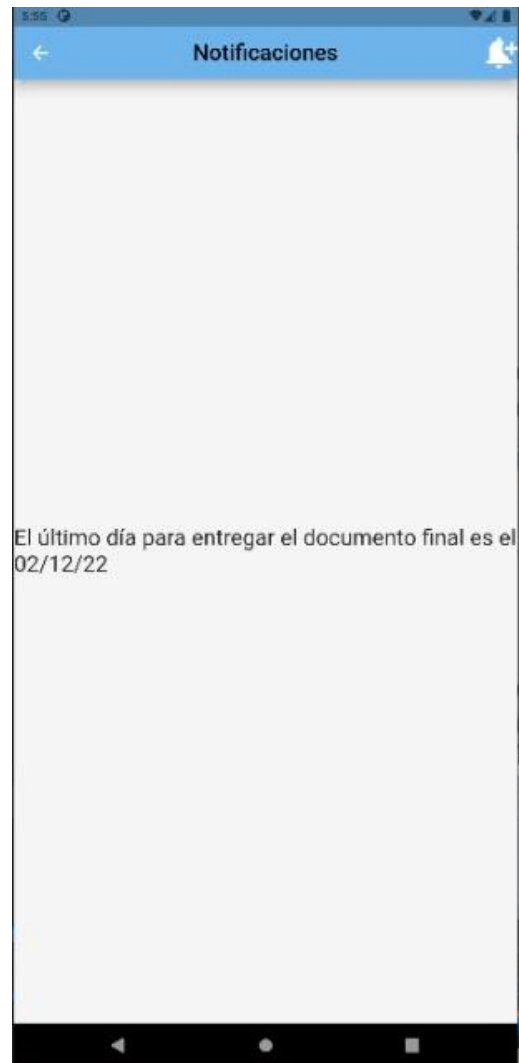


Ilustración 25. Pantalla de notificaciones

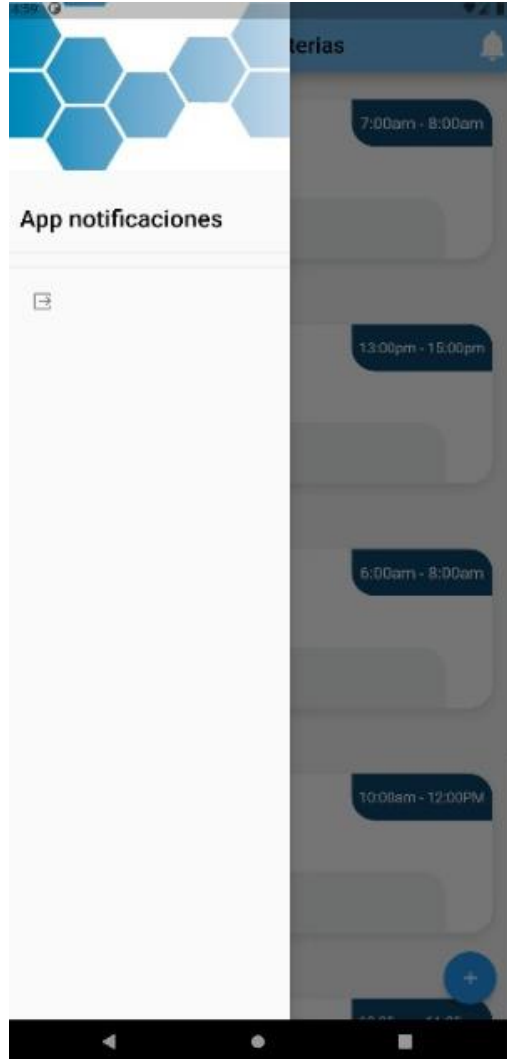


Ilustración 26. Menú para cerrar sesión

Y los resultados alcanzados de cada uno de los objetivos específicos se muestran cada una de las tareas específicas que se realizaron para poder cumplir con el objetivo general.

9.2 Resultados del primer objetivo específico

Como resultado de esta tarea se realizó una tabla la cual contiene los requisitos funcionales que nos ayudaron a conseguir una plataforma funcional ver Tabla 3.

Tabla 3. Requisitos funcionales de la aplicación

N°	Requisitos funcionales
1	Crear cuentas con correo y contraseña.
2	Iniciar sesión con correo y contraseña.
3	Dar de alta las materias.
4	Recibir notificaciones.
5	Inscribirse a la materia.

9.3 Resultados del segundo objetivo específico

Como resultado de esta tarea se utilizó algunas de las pantallas para mostrar el uso y funcionamiento de base de datos interna desde la plataforma de flutter crear una nueva cuenta de usuario y esta sea almacenada en la base de datos para su posterior inicio de sesión. En la primera pantalla Ilustración 27 se puede ver en el formulario los datos del usuario que se quiere crear, una vez que se le da clic al botón registrarse los datos se guardarán en la base de datos interna SQLite Ilustración 28.



Ilustración 27. Pantalla con el correo y contraseña a registrar

Identificador	Proveedores	Fecha de creación	↓	Fecha de acceso	UID de usuario ↑
prueba1@gmail.com	✉	25 nov 2022		25 nov 2022	pqjwG3OMhRgVmxMysiWRKFGiC...

Ilustración 28. Información almacenada en la BDD

9.4 Resultado del tercer objetivo específico

Y por último como resultado final de esta tarea se realizaron un par de pruebas para verificar la correcta funcionalidad de la aplicación de tal manera que estas nos ayudarán a comprobar el excelente desempeño para su posterior evaluación.

Tabla 4. Prueba de inicio de sesión

Prueba de validación inicio de sesión	
Tipo de prueba	Funcional
Caso #1	
Ingresos	Email y contraseña correcta
Resultado	Acceso a la aplicación
Caso #2	
Ingresos	Campos vacíos o Email y contraseñas incorrectas
Resultado	Mensaje: <ul style="list-style-type: none"> • El correo y contraseña son incorrectos. Intenta de nuevo.

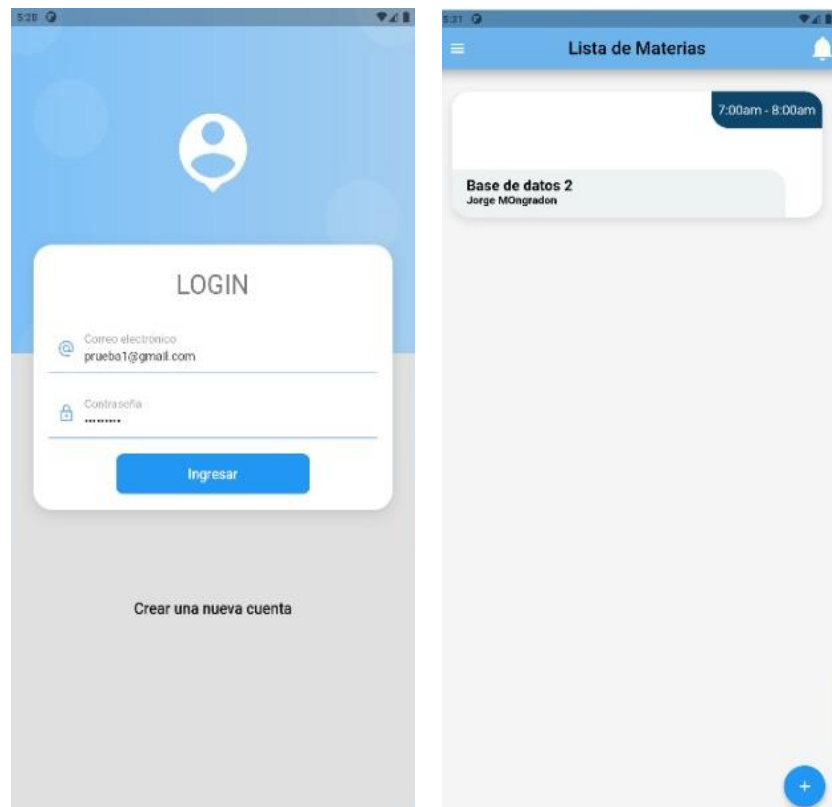


Ilustración 29. Prueba de validación caso#1

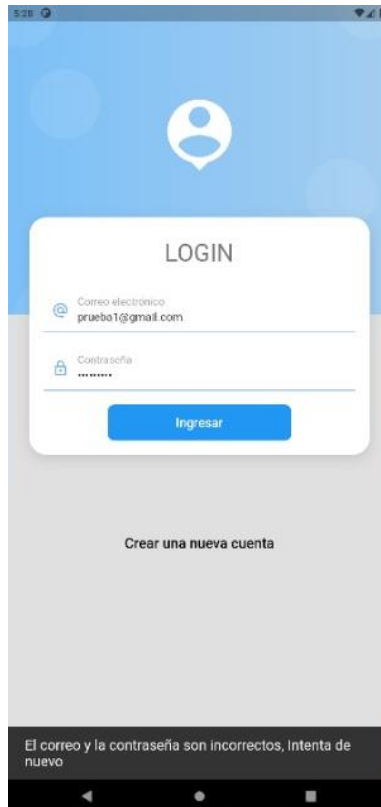
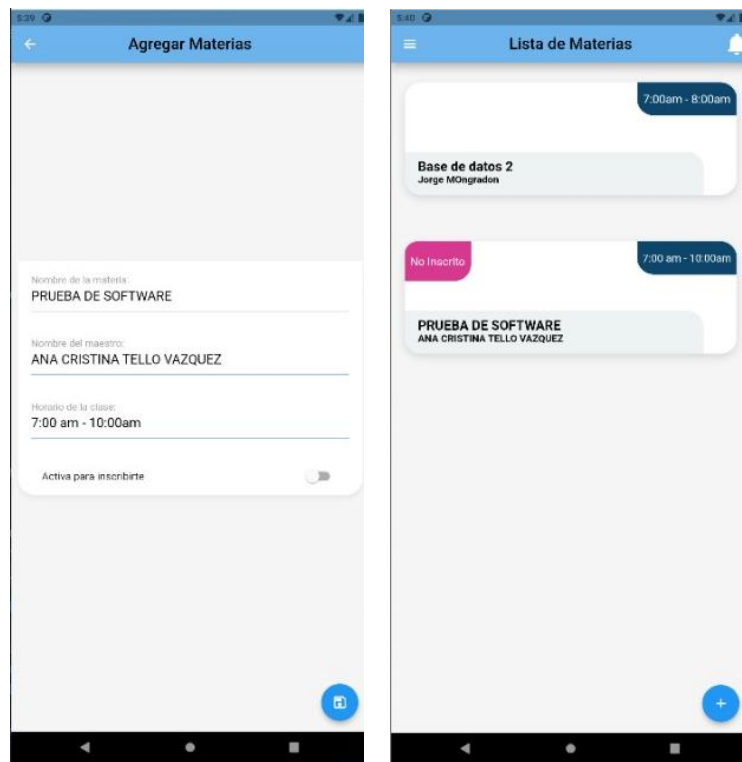


Ilustración 30. Prueba de validación caso #2

Tabla 5. Prueba de materias agregadas y BDD

Prueba de mostrar materias agregadas a la aplicación y a la base de datos	
Tipo de prueba	Funcional
Caso #1	
Ingresos	Llenar el formulario con la información correspondiente
Resultado	Se agrega a la pantalla principal de la aplicación y se agrega a la base de datos.



```

https://app-residencias-b70e7-default-rtdb.firebaseio.com/
├── Materias
│   ├── -NGqnKy5-f6g2Yo0sBwk
│   └── -NHjs9ZogtXJAdHxdvQR
│       ├── habilitado: false
│       ├── horario: "7:00 am - 10:00am"
│       ├── maestro: "ANA CRISTINA TELLO VAZQUEZ"
│       └── materia: "PRUEBA DE SOFTWARE"

```

Ilustración 31. Prueba de mostrar materias agregadas a la aplicación y a la base de datos

Tabla 6. Pruebas de notificaciones

Prueba de enviar y recibir notificaciones	
Tipo de prueba	Funcional
Caso #1	
Ingresos	Escribir el título y mensaje de la notificación que se quiere enviar
Resultado	La notificación llega a la barra de estado del dispositivo para después poder verla en la pantalla de notificaciones.

1 Notificación


Título de la notificación ⓘ

Maestra.. Nivia Iracemi García

Texto de la notificación

Materia.. Programación

Imagen de notificación (opcional) ⓘ

Ejemplo: <https://tuapp.com/imagen.png> 

Nombre de la notificación (opcional) ⓘ

Ingresar nombre opcional

Vista previa del dispositivo

En esta vista previa, se ofrece una idea general de cómo se mostrará tu mensaje en un dispositivo móvil. La apariencia real del mensaje varía en función del dispositivo. Para obtener resultados precisos, prueba con un dispositivo real.

Enviar mensaje de prueba

Estado inicial Vista expandida


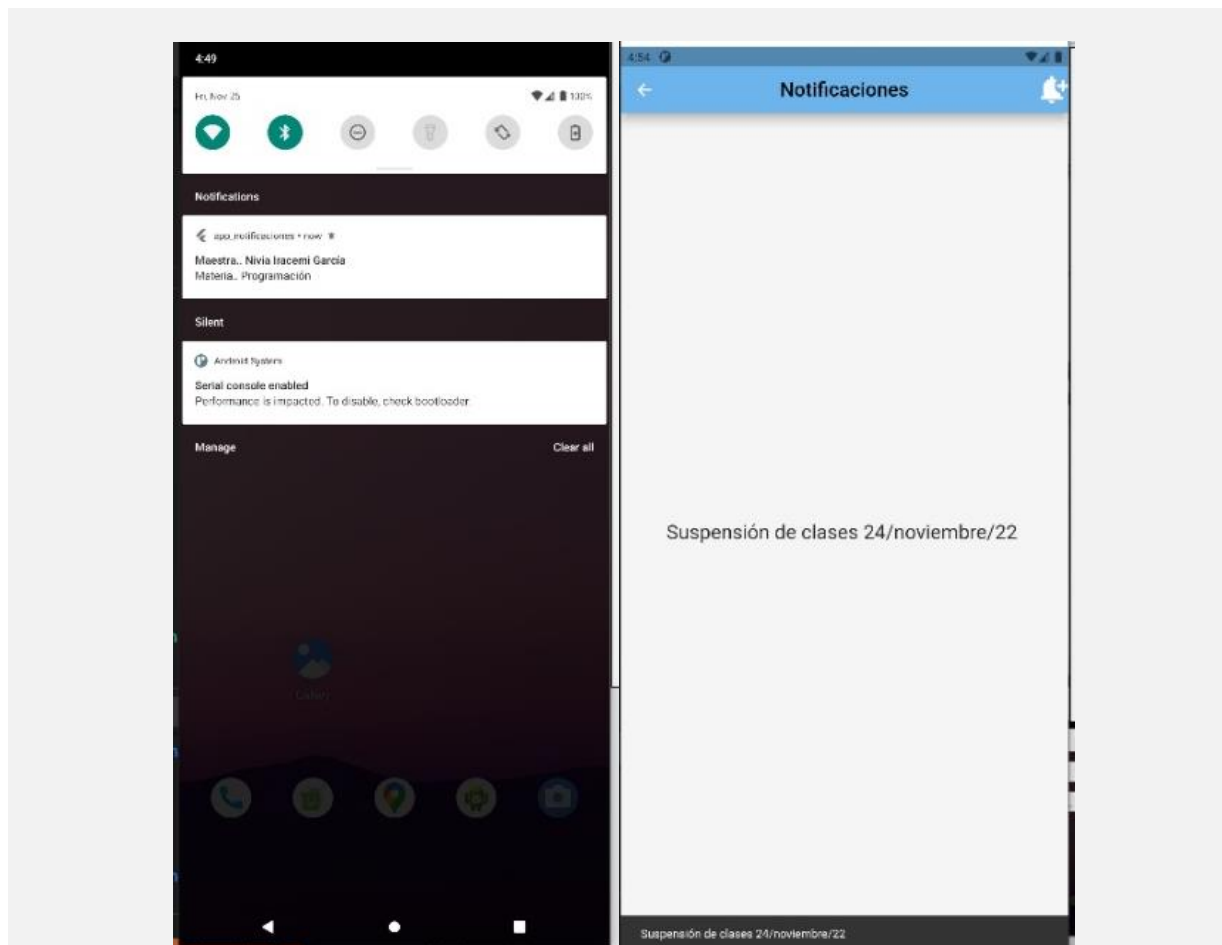



Ilustración 32. Prueba de enviar y recibir notificaciones

CAPÍTULO 5: CONCLUSIONES

10. Conclusiones del Proyecto

El proyecto realizado mediante la implementación de una aplicación móvil para la gestión y control de notificaciones academias en el aula es capaz de recibir notificaciones, los docentes pueden agregar sus materias con sus respectivos horarios para que después los alumnos puedan inscribirse a cada una de sus materias asignadas en su horario escolar. Para ello se ha utilizado la tecnología Dart uno de los lenguajes de programación que permite a los desarrolladores utilizar un lenguaje orientado a objetos y Flutter que permite la creación de aplicaciones nativas para dispositivos móviles. En el objetivo general tal y como se describe en la sección **5. Objetivos (General y Específicos)** es desarrollar una herramienta de software institucional que facilite la interacción entre alumno y maestro de manera formal dentro de las instituciones, tal que ayude a notificar sobre sucesos que se ameriten en sesiones de clase o en asesorías de cualquier tipo. Al obtener cada uno de los resultados se consideró que se ha cumplido con este objetivo ya que la aplicación es funcional, en los objetivos específicos mencionados también en esa sección son cada una de las actividades que se tuvieron que realizar para poder lograr con el objetivo general por lo tanto se ha realizado la aplicación para la gestión y control de las notificaciones académicas en el aula tal y como se esperaba debido a que cumple con cada uno de los requerimientos solicitados

CAPÍTULO 6: COMPETENCIAS DESARROLLADAS

11. Competencias desarrolladas y/o aplicadas.

1. Apliqué habilidades de análisis y diseño de aplicaciones móviles, para así poder implementarlas dentro del desarrollo del proyecto.
2. Utilicé tecnología de aplicaciones móviles para la programación del front-end y el back-end.
3. Desarrollé el diseño de la aplicación móvil con la plataforma de desarrollo Flutter.
4. Apliqué herramientas de Android Studio para la virtualización de la aplicación móvil.
5. Implementé las funciones principales con el lenguaje de programación Dart.
6. Obtuve conocimiento del lenguaje de programación Dart a través de un Curso llamado "Flutter: Tu guía completa de desarrollo para IOS y Android" en la plataforma de Udemy.
7. Apliqué habilidades de investigación con el fin de usarlos en nuestro documento y la práctica.

CAPÍTULO 7: FUENTES DE INFORMACIÓN

12. Fuentes de información

1. Cuello Javier, & Vittone José. (2013). Diseñando apps para móviles - Javier Cuello, José Vittone - Google Libros. Retrieved from.
2. https://books.google.es/books?hl=es&lr=&id=ATiqsjH1rvwC&oi=fnd&pg=PA7&dq=que+es+una+app&ots=a4a15_6q6t&sig=aqDgWIXuPetVPUuwkKvWq-YE2JY#v=onepage&q&f=false.
3. Pérez, S., Quispe, J., Mullicundo, F., & Lamas, D. (2021). Herramientas Y Tecnologías Para El Desarrollo Web Desde El Frontend Al Backend. *XXIII Workshop de Investigadores En Ciencias de La Computación*, 347–350. Retrieved from <http://sedici.unlp.edu.ar/handle/10915/120476>.
4. CUAED.U.(2021). LENGUAJES DE PROGRAMACIÓN.https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html.
5. ¿Qué es el lenguaje de programación Dart? | inLab FIB. (2021). Retrieved from <https://inlab.fib.upc.edu/es/blog/que-es-el-lenguaje-de-programacion-dart>.
6. Flutter. (2019). Persist data with SQLite - Flutter. Retrieved from <https://flutter.dev/docs/cookbook/persistence/sqlite>.
7. Felipe. (2021). Objetivos de las bases de datos y sus características. Retrieved from <https://www.hostingplus.pe/blog/objetivos-de-las-bases-de-datos-y-sus-caracteristicas/>.
8. Android Studio. (2022). Introducción a Android Studio | Desarrolladores de Android | Android Developers. Retrieved from <https://developer.android.com/studio/intro?hl=es-419>.
9. Hallinger, P., Heck, R. y Murphy, J. (2014). Teacher evaluation and school improvement: An analysis of the evidence. *Educational Assessment, Evaluation and Accountability*, 26(1), 5-28. <http://dx.doi.org/10.1007/s11092-013-9179-5>.
10. Tacca, D., Tacca, A. y Alva, M. (2019). Estrategias neurodidácticas, satisfacción y rendimiento académico en estudiantes universitarios. *Cuadernos de Investigación Educativa*, 10(2), 15-32. <https://dx.doi.org/10.18861/cied.2019.10.2.2905>.
11. Allcott, H., & Gentzkow, M. (2017, March 1). Social media and fake news in the 2016 election. *Journal of Economic Perspectives*. American Economic Association. <https://doi.org/10.1257/jep.31.2.211>.

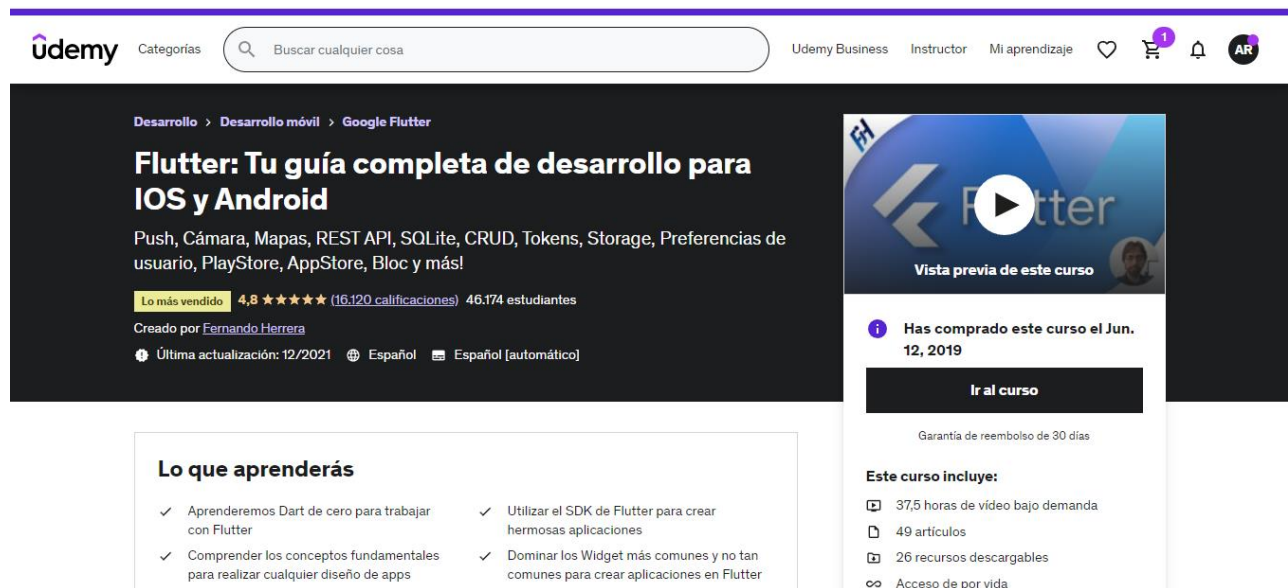
12. Cabrera Farfán, M. T., & Reyes Adan, I. A. (2017). Gestión educativa estratégica y gestión escolar del proceso de enseñanza-aprendizaje: una aproximación conceptual. *Análisis de Problemas Universitarios*, 28, 45–61.

CAPÍTULO 8: ANEXOS

13. Anexos

(carta de autorización por parte de la empresa u organización para la residencia profesional y otros si son necesarios).

13.1 Curso de Udemy



The screenshot shows the Udemy course page for "Flutter: Tu guía completa de desarrollo para IOS y Android". The page features a dark header with the Udemy logo, search bar, and navigation links. The main content area includes the course title, a description, a rating of 4.8 stars from 16,120 reviews, and 46,174 students. A "Lo más vendido" badge is present. The course is created by Fernando Herrera and was last updated in December 2021. A sidebar on the right shows a video preview, purchase information (purchased on June 12, 2019), and a list of course contents: 37.5 hours of on-demand video, 49 articles, 26 downloadable resources, and lifetime access. A "Garantía de reembolso de 30 días" is also mentioned.

Flutter: Tu guía completa de desarrollo para IOS y Android

Push, Cámara, Mapas, REST API, SQLite, CRUD, Tokens, Storage, Preferencias de usuario, PlayStore, AppStore, Bloc y más!

Lo más vendido 4,8 ★★★★★ (16.120 calificaciones) 46.174 estudiantes

Creado por [Fernando Herrera](#)

Última actualización: 12/2021 Español Español [automático]

Lo que aprenderás

- ✓ Aprenderemos Dart de cero para trabajar con Flutter
- ✓ Comprender los conceptos fundamentales para realizar cualquier diseño de apps
- ✓ Utilizar el SDK de Flutter para crear hermosas aplicaciones
- ✓ Dominar los Widget más comunes y no tan comunes para crear aplicaciones en Flutter

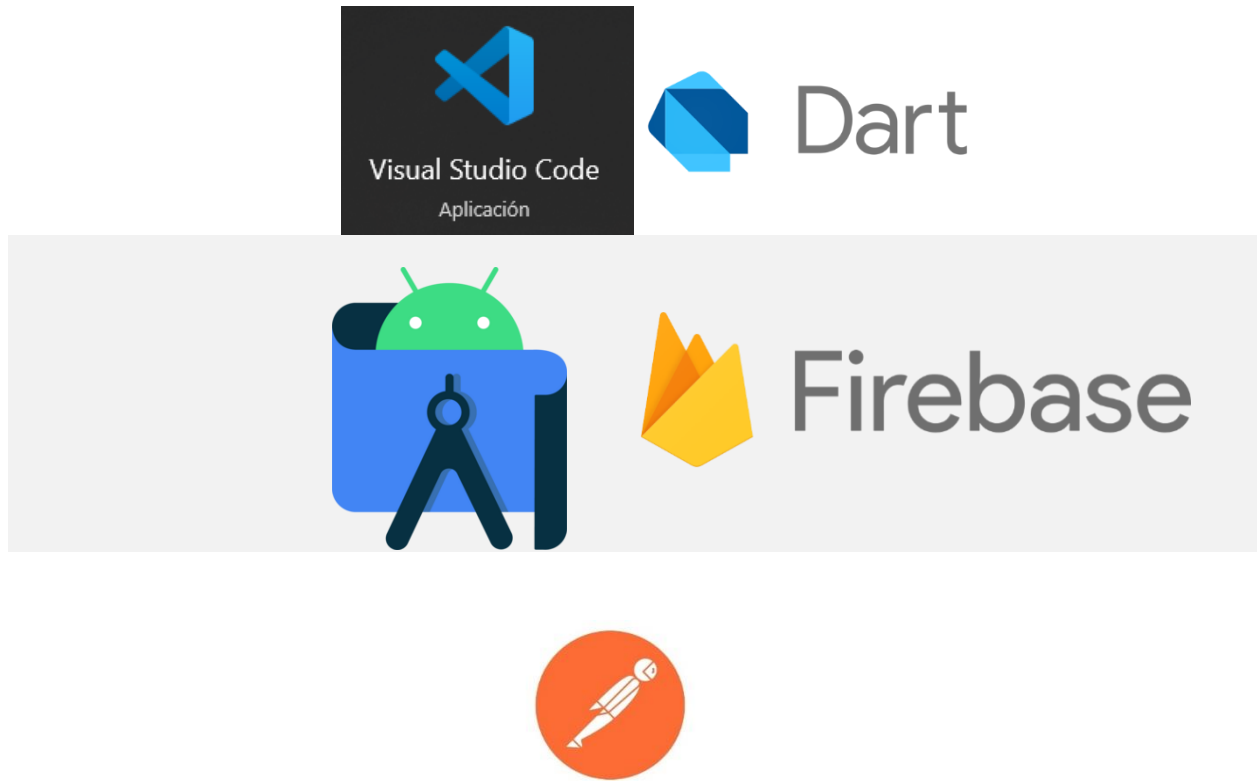
Este curso incluye:

- 37,5 horas de video bajo demanda
- 49 artículos
- 26 recursos descargables
- Acceso de por vida

Garantía de reembolso de 30 días

Ilustración 33. Curso de Flutter

13.2 Instalaciones de herramientas para el desarrollo



POSTMAN

Ilustración 34. Herramientas de desarrollo

14. Registros de Productos

(patentes, derechos de autor, compra-venta del proyecto, etc.).