

INSTITUTO DE FORMACIÓN TÉCNICA SUPERIOR N°11

DOCUMENTACIÓN DE “StudyHub” UNA APLICACIÓN PARA ESTUDIANTES

Materia: Desarrollo de Aplicaciones para dispositivos

Profesor: Damián Gómez

Integrantes: Mauricio Soto, Estefania Romero, Luciano de la Peña, Leonel Godoy

Detalle: Documentación técnica de la aplicación desarrollada para dispositivos móviles.

Año: 2025

Versión: 28.10.25



NOTAS PRELIMINARES.....	2
1. DEFINICIÓN GENERAL DEL PROYECTO DE SOFTWARE.....	2
a. IDEA GENERAL.....	2
b. OBJETIVOS.....	3
c. USUARIOS.....	3
2. ESPECIFICACIÓN DE REQUERIMIENTOS Y LÍMITES DEL PROYECTO.....	3
a. REQUERIMIENTOS GENERALES.....	3
b. REQUERIMIENTOS FUNCIONALES.....	3
c. ALCANCE Y LIMITACIONES.....	4
3. ESPECIFICACIONES DE PROCEDIMIENTOS.....	4
a. PROCEDIMIENTO DE INSTALACIÓN Y PRUEBA.....	4
b. PROCEDIMIENTO DE DESARROLLO.....	4
Epica.....	4
Historias de usuario.....	5
CAPTURAS DE PANTALLA.....	6
Herramientas informáticas usadas.....	8
Herramientas el desarrollo.....	8
Herramientas para comunicación y organización.....	8
Fecha de desarrollo.....	9
4. ARQUITECTURA DEL SISTEMA.....	9
1. Frontend: Desarrollo de la Aplicación Móvil.....	9
2. Backend: Lógica del Servidor y API.....	10
3. Flujo de Interacción.....	11

NOTAS PRELIMINARES

1.DEFINICIÓN GENERAL DEL PROYECTO DE SOFTWARE

a. IDEA GENERAL

StudyHub es una aplicación móvil que permite a estudiantes subir su material de estudio a la nube y tenerlo organizado por materias.

b. OBJETIVOS

El objetivo es desarrollar un producto de software orientado a estudiantes que permita subir y organizar su material de estudio en forma de listas por cada materia. Los usuarios podrán registrarse con su email o bien iniciar sesión con Google o GitHub. Una vez logueados, los usuarios podrán crear materias que serán las listas para ir cargando sus recursos de estudio.

c. USUARIOS

Los destinatarios de la app serán estudiantes que necesiten tener su material de estudio organizado y disponible en la nube.

2.ESPECIFICACIÓN DE REQUERIMIENTOS Y LÍMITES DEL PROYECTO

a. REQUERIMIENTOS GENERALES

Este proyecto busca cumplir con los siguientes puntos centrales:

- Desarrollar una aplicación móvil utilizando como herramientas principales Ionic Framework y Typescript.
- Utilizar al menos dos apis: una para el Login -API de Firebase-, y otra para almacenamiento en la nube -Filestack.
- Utilizar una funcionalidad nativa del dispositivo -en nuestro caso será el almacenamiento o *localstorage*.
- Tener un ícono.

b. REQUERIMIENTOS FUNCIONALES

- Al iniciar la app el usuario deberá poder registrarse con email o bien iniciar sesión con su cuenta de Google.
- Al entrar por primera vez al Home, estará vacío, sin materias.
- Habrá un botón para crear materias.
- Al crear una materia, ésta aparecerá en pantalla en forma de una tarjeta (ionic-card).

- En esa tarjeta el usuario podrá agregar sus archivos de estudio, los cuales se subirán a la nube.
- Al entrar a una materia se mostrarán los archivos disponibles.
- Habrá una pestaña de Comunidad donde se verá material subido por otros usuarios.
- El código de la aplicación estará alojado en GitHub.
- El código podrá descargarse y desplegarse en un equipo con Windows 10/11.

c. ALCANCE Y LIMITACIONES

La API de almacenamiento consumida es de capa gratuita, por lo cual la cantidad de consultas y el espacio de almacenamiento son limitados; no obstante la aplicación desarrollada deberá ser capaz de subir y consultar una variedad de archivos de prueba: documentos de texto, pdfs, imágenes. Esta aplicación no está pensada para subir videos.

3.ESPECIFICACIONES DE PROCEDIMIENTOS

a. PROCEDIMIENTO DE INSTALACIÓN Y PRUEBA

1. Para bajar el proyecto abrir la línea de comandos y ejecutar:

```
git clone https://github.com/estefaniansr/StudyHub.git
```

Esto bajará la carpeta del proyecto.

2. instalar bibliotecas necesarias:

```
npm install -f
```

3. Correr proyecto de Ionic:

```
ionic serve
```

b. PROCEDIMIENTO DE DESARROLLO

Epica

Descripción de la épica: implementar un sistema que permita subir y tener organizado el material de estudio.

Rol: usuario (estudiante)

Objetivo: subir y organizar su material de estudio

Resultado: subir y poder consultar el material de estudio almacenado en la nube.

Historias de usuario

A continuación se listan las Historias de Usuario que guiaron el desarrollo.

HU#1: Como usuario quiero poder subir a la nube mi material de estudio (archivos de texto, pdfs, imágenes, etc) para poder acceder a él desde cualquier dispositivo.

HU#2: Como usuario quiero poder descargar a mi dispositivo los archivos que haya subido a la nube, para poder tenerlos disponibles sin conexión.

HU#3: Como usuario quiero poder restablecer mi contraseña en caso de perderla.

HU#4: Como usuario quiero poder acceder con mi cuenta de Google, para no tener que registrarme.

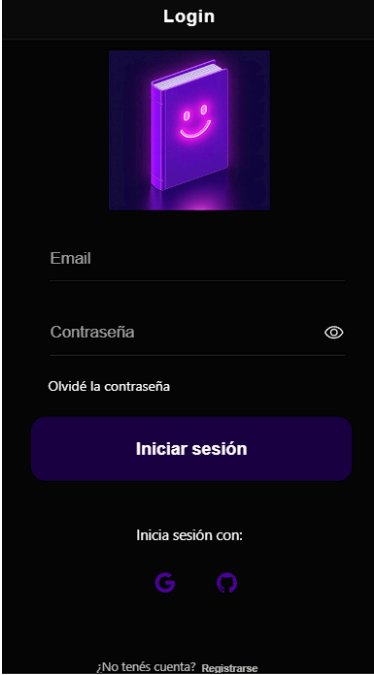
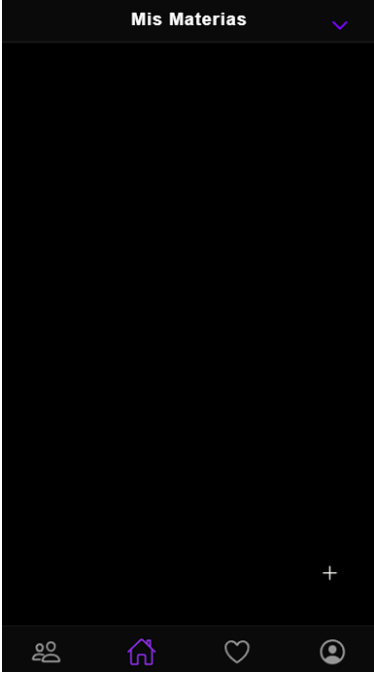
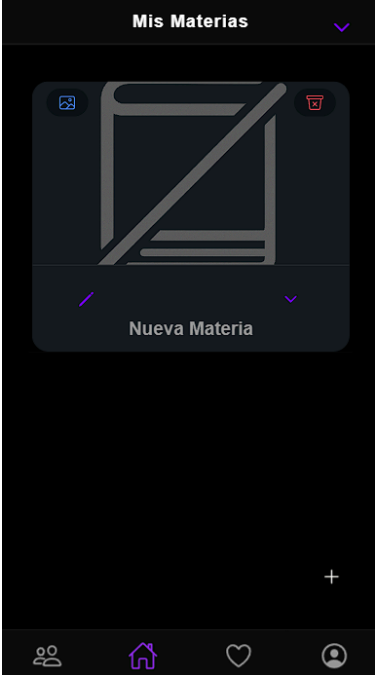
HU#5: Como usuario quiero poder agregar y quitar materias de una lista de favoritos, para acceder rápidamente a ellas.

HU#6: Como usuario quiero poder acceder al material compartido por la comunidad.

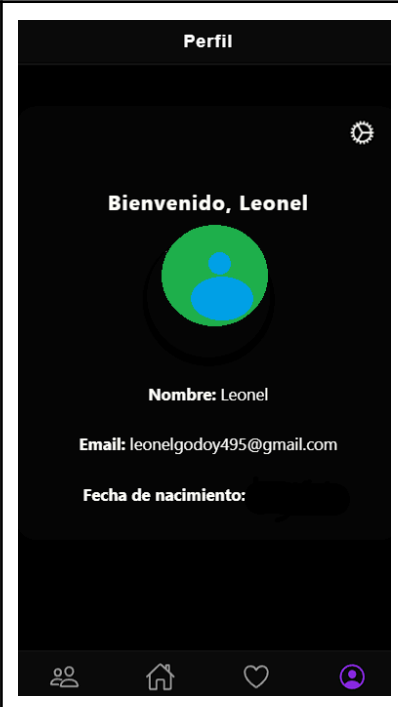
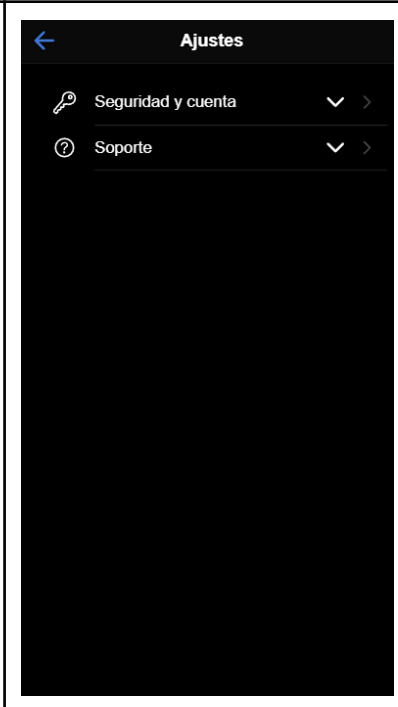
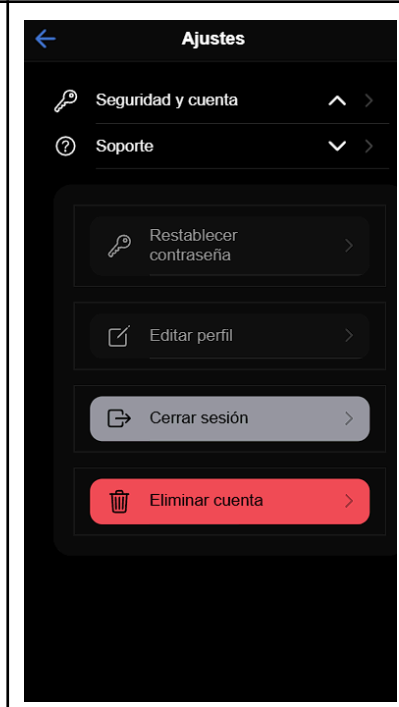
HU#7: Como usuario quiero poder descargar los archivos compartidos por la Comunidad.

CAPTURAS DE PANTALLA

Nota: En el Manual de Usuario se describe el paso a paso para realizar cada tarea. En este apartado sólo se busca ilustrar las diferentes pantallas y funciones que componen a la aplicación.

		
<p>Pantalla de Login. Está la opción de loguearse con Google/GitHub.</p>	<p>La barra inferior permite navegar: Comunidad, Mis Materias, Favoritos, Perfil.</p>	<p>Para crear una nueva materia, dentro del tab Mis Materias, tocamos el ícono (+)</p>

		
<p>Editamos el nombre tocando en el ícono del lápiz.</p>	<p>Es posible crear más materias si hace falta.</p>	<p>A cada materia se puede agregar una imagen de portada, para encontrarla más fácilmente.</p>

		
<p>Pantalla de Perfil. Tocando la tuerca superior derecha se</p>	<p>Pantalla de Ajustes. También es posible comunicarse con el</p>	<p>Dentro de Seguridad y cuenta, se encuentran funciones como</p>

accede a los Ajustes.	equipo técnico de la app tocando en Soporte.	Restablecer contraseña, Cerrar sesión y Eliminar cuenta.
-----------------------	--	--

		
<p>En la tab Comunidad tendremos disponible el material compartido por otros usuarios de la comunidad de StudyHub.</p>	<p>Si queremos compartir una materia con la comunidad hacemos clic en el candado inferior.</p>	<p>Volvemos a la pestaña comunidad, actualizamos y se verá el material.</p>

Herramientas informáticas usadas

Herramientas el desarrollo

Git y GitHub para el versionado de código.

Google drive y su suite ofimática para trabajar documentación.

Visual Studio Code como editor de código.

Herramientas para comunicación y organización.

Whatsapp para mensajería y Discord para videollamadas.

Trello y Notion para el seguimiento de tareas.

Fecha de desarrollo

La aplicación fue desarrollada para el segundo cuatrimestre del año 2025, entre los meses de agosto y noviembre.

4.ARQUITECTURA DEL SISTEMA

La arquitectura del sistema de la aplicación móvil se fundamenta en un modelo de desarrollo **híbrido** y una estructura de *stack* tecnológico **moderno y robusto**, diseñado para garantizar la eficiencia, el rendimiento y la facilidad de mantenimiento.

El sistema se compone esencialmente de dos capas principales: el **Frontend**, que gestiona la interfaz de usuario en el dispositivo móvil, y el **Backend**, que maneja la lógica de negocio, la persistencia de datos y la comunicación con otros servicios.

1. Frontend: Desarrollo de la Aplicación Móvil

El desarrollo del lado del cliente se centra en la utilización del *framework* **ionic**, potenciado por **Angular**, lo que permite crear una única base de código para ser desplegada en múltiples plataformas (iOS y Android) a partir de tecnologías web estándar (HTML, CSS, JavaScript).

Componente	Descripción	Rol en la Arquitectura
Ionic Framework	Un <i>framework</i> de código abierto que facilita el desarrollo de aplicaciones móviles híbridas. Utiliza tecnologías web para empaquetar la aplicación en un contenedor nativo (como Cordova o Capacitor), permitiendo el acceso a las funcionalidades del dispositivo.	Permite el despliegue multiplataforma y proporciona componentes UI listos para usar, logrando una experiencia de usuario nativa.
Angular	Es un <i>framework</i> de desarrollo de aplicaciones web basado en TypeScript. Es el esqueleto lógico del <i>frontend</i> .	Gestiona la lógica de la interfaz de usuario , el estado de la aplicación , la navegación entre vistas, y la comunicación asíncrona

		con el <i>backend</i> a través de servicios HTTP.
TypeScript / JavaScript	Lenguajes de programación.	Implementación de la lógica del cliente y la interacción con los componentes de Ionic/Angular.

2. Backend: Lógica del Servidor y API

La capa de servidor es la responsable de manejar las peticiones del *frontend*, procesar la lógica de negocio, interactuar con la base de datos y exponer los *endpoints* necesarios para la aplicación móvil.

Componente	Descripción	Rol en la Arquitectura
Node.js	Un entorno de ejecución de JavaScript del lado del servidor, asíncrono y basado en eventos. Es la tecnología base del <i>backend</i> .	Proporciona la plataforma escalable para construir la API RESTful que sirve como puente de comunicación entre la aplicación móvil y la base de datos o sistemas del Instituto.
Express	<i>Framework</i> minimalista y flexible para Node.js (típicamente usado con Node.js para APIs).	Facilita la creación de rutas , el manejo de peticiones HTTP (GET, POST, PUT, DELETE) y la gestión de <i>middleware</i> (autenticación, validación).
Base de Datos	El material de estudio se almacena en una base de datos de firebase.	Almacenamiento y recuperación eficiente de los datos de las materias. Node.js interactúa con ella a través de <i>drivers</i> o ORMs.

3. Flujo de Interacción

La interacción en la arquitectura sigue un patrón cliente-servidor estándar:

1. **Petición (Frontend → Backend):** El usuario interactúa con la aplicación (ej: solicita ver sus materias). La lógica de Angular en el dispositivo formula una **petición HTTP**.
2. **Procesamiento (Backend):** La petición es recibida por el servidor **Node.js/Express**.
 - Se verifica la autenticación y autorización.
 - Se ejecuta la lógica de negocio.
 - Se consulta o modifica la información en la **Base de Datos**.
3. **Respuesta (Backend → Frontend):** Node.js envía una **respuesta** a la aplicación móvil, generalmente en formato **JSON**, conteniendo los datos solicitados (ej: la lista de calificaciones).
4. **Renderizado (Frontend):** Angular recibe los datos JSON, actualiza el estado de la aplicación y usa los componentes de Ionic para **mostrar la información** en la interfaz de usuario.