

PROYECTO FIN DE CICLO

Título:

Alumno:

Nº Alumno:

**CICLO FORMATIVO DE GRADO SUPERIOR EN DAM / DAW**

Curso 2023-2024

**DECLARACIÓN JURADA DE AUTORÍA**

D. / Dña. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, con N.I.F. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, D. / Dña. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, con N.I.F. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, D. / Dña. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, con N.I.F. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, como autor/es de este documento académico, titulado:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

y presentado como Proyecto Fin de Ciclo para la obtención del Ciclo Formativo de Grado Superior en\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,

DECLARO QUE

Soy/somos el/los único/s autor/es del trabajo, con la excepción de referencias a contenidos o ideas de otros autores, en cuyo caso han sido explícitamente citados.

El trabajo remitido es un documento original, que no ha sido publicado, ya sea total o parcialmente, ni presentado para obtención de un título académico en ninguna institución académica u organización.

No he trasgredido ninguna norma universitaria con respecto al plagio ni a las leyes establecidas que protegen la propiedad intelectual.

Soy consciente de que el hecho de no respetar estos extremos es una falta grave de integridad académica y podrá ser objeto de sanciones.

En \_\_\_\_\_\_\_\_\_\_\_\_\_, a \_\_\_\_ de \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ de 20\_\_\_\_

(firma) (firma) (firma)

Fdo.: Fdo.: Fdo.:

**RESUMEN**

(Resumen en español)

**ABSTRACT**

(Resumen en inglés)

Contenido

[1. INTRODUCCIÓN 5](#_Toc157786724)

[2. REQUISITOS FUNCIONALES 6](#_Toc157786725)

[3. DISEÑO DE LA SOLUCIÓN 8](#_Toc157786726)

[3.1. Mapa de navegación 8](#_Toc157786727)

[3.2. Modelo de base de datos 15](#_Toc157786728)

[3.3. Diseño de la arquitectura 16](#_Toc157786729)

[3.4. Lenguajes utilizados 16](#_Toc157786730)

[4. DISEÑO TECNICO: 17](#_Toc157786731)

[4.1. Libro de Estilo: 17](#_Toc157786732)

[4.2. Definición de componentes: 17](#_Toc157786733)

[4.2.1. Web Scraper(Pyhton): 17](#_Toc157786734)

[4.3. Diseño de Tablas de Modelo de Datos: 20](#_Toc157786735)

**ÍNDICE DE ILUSTRACIONES**

[Ilustración 1: Mapa Global de navegación. Fuente 9](#_Toc153992300)

[Ilustración 2: Página de Inicio. Fuente: 10](#_Toc153992301)

[Ilustración 3: Inicio de Sesión. Fuente 11](#_Toc153992302)

[Ilustración 4: Página Principal, elegir género. Fuente 12](#_Toc153992303)

[Ilustración 5: Distribución de pantalla principal a su género. Fuente 13](#_Toc153992304)

[Ilustración 6: visualización de titulares, género político. Fuente 14](#_Toc153992305)

[Ilustración 7: Modelo de Datos. Fuente 15](#_Toc153992306)

**ÍNDICE DE TABLAS**

*Tabla 1: Título de la tabla. Fuente:* **Error! Bookmark not defined.**

*De momento no hay tablas, voy a mantener la estructura*

# INTRODUCCIÓN

En la actualidad, las redes sociales han adquirido un papel primordial como fuente de información, especialmente para los jóvenes. Sin embargo, la saturación de información ha llevado a una disminución en la calidad de los contenidos, priorizando la cantidad sobre el valor informativo. Esto ha llevado al éxito de formatos de contenido corto como *Twitter, Snapchat o TikTok*. La falta de métodos de control para validar la información producida en estas redes ha generado campañas de desinformación que han polarizado la sociedad a nivel global.

En base a lo anterior, mi proyecto de final de grado se basará en la creación de una aplicación de noticias informativas con formato de contenido corto, donde los usuarios puedan ver los titulares de noticias en tiempo real, dentro de un marco de 24 horas.

El objetivo principal de esta aplicación es ofrecer a los usuarios un acceso rápido y sencillo a las noticias de actualidad. Para lograr esto, recopilaremos los titulares de la agencia EFE, una empresa de información fiable que provee un gran volumen informativo dentro del marco de las 24 horas.

Para simplificar el proceso de obtención de los titulares, dividiremos las noticias en cuatro géneros principales: Política, Economía y Finanzas, Deporte y Cultura. En cuanto al modelo de negocio, en caso de que este proyecto se lleve a cabo más allá de la demostración final, utilizaríamos un enfoque conocido como *Freemium*, donde la aplicación sería gratuita para descargar y utilizar, de manera que los usuarios no tengan que pagar por nada. Los ingresos se generarían únicamente mediante la venta de datos recopilados.

# REQUISITOS FUNCIONALES

Los requisitos funcionales de la aplicación los dividiremos en básicos y opcionales. La razón por la que dividiremos estos aparatados es porque, la aplicación debe integrar los conocimientos básicos que requiere el curso de DAM debiéndose demostrar mediante la creación de la aplicación de *Newsreel*, mientras que los requisitos opcionales será la ampliación real de crear una app con requisitos para el mundo real. Dicho esto, a continuación, enumeraremos los requisitos funcionales:

*Básicos*

1. **Registro de Usuarios:**
   * Los usuarios pueden crear cuentas personalizadas.
   * Iniciar sesión con credenciales de usuario.
2. **Navegación y Visualización de Noticias:**
   * Ver titulares de noticias en tiempo real.
   * Filtrar noticias por género (Política, Economía, Deporte, Disturbios y Cultura).
   * Visualizar en formato Tinder con el método *Scroll*, y vista de titulares en formato Twitter, donde el contenido de noticias se visualizará con un límite de caracteres al solo poder verse el titular.
3. **Actualizaciones en Tiempo Real:**
   * Las noticias se actualizan automáticamente en la aplicación.
4. **Recopilación de Datos:**
   * Recopilación de datos de uso de la aplicación de forma anónima y segura.
   * Informar a los usuarios sobre la recopilación de datos y obtener su consentimiento.
5. **Estadísticas y Análisis:**
   * Herramientas internas para analizar el uso de la aplicación y la popularidad de las noticias.
   * Informes sobre tendencias de usuarios y preferencias de noticias.
6. **Venta de Datos:**
   * Recopilación de datos de uso de la aplicación de forma anónima y segura.
   * Informar a los usuarios sobre la recopilación de datos y obtener su consentimiento.
   * Explicar claramente en la política de privacidad que los datos recopilados se utilizarán con fines comerciales, como la venta de datos.

*Opcional:*

1. **Gestión de Favoritos:**
   * Los usuarios pueden marcar noticias como favoritas para acceder fácilmente a ellas más tarde.
   * Opción para eliminar noticias de la lista de favoritos.
2. **Seguridad:**
   * Protección de la información del usuario mediante medidas de seguridad adecuadas.
   * Encriptación de datos para garantizar la privacidad de la información del usuario.

*Muy opcional:*

1. **Compatibilidad Multiplataforma:**
   * Disponibilidad de la aplicación en múltiples plataformas (iOS, Android, web).
2. **Soporte y Retroalimentación:**
   * Formulario de contacto o soporte dentro de la aplicación.
   * Opción para que los usuarios proporcionen comentarios o informen sobre problemas.

# DISEÑO DE LA SOLUCIÓN

Basado en los requisitos funcionales, en esta sección se especifican el diseño que se plantea del proyecto para asegurar que dará respuesta a las funcionalidades planteadas.

## Mapa de navegación

En esta sección se especifican las pantallas y las interacciones que tendrá el Proyecto para asegurar que dará respuesta a las funcionalidades planteadas. Para mostrar y explicar el proceso he creado un diseño de figma para especificar cual es el recorrido de la aplicación, las pantallas y la distribución de la aplicación.

A continuación, mostraré el mapa general de navegación global:

A computer screen shot of a diagram

Description automatically generated

Ilustración 1: Mapa Global de navegación. Fuente

Como podemos observar tenemos 5 ventanas de acceso de las cuales la penúltima ventana abrirá cinco ventanas distintas dependiendo del género de noticia que escojamos, concretaremos más abajo.

Esta primera pantalla, será la presentación de bienvenida de la aplicación con un diseño para darle calidad estética UX/Ui.

Screens screenshot of a screenshot of a screenshot

Description automatically generated

Ilustración 2: Página de Inicio. Fuente:

En una vez la primera pantalla haya realizado la animación ira a la siguiente página que será el inicio de sesión, registro, donde o podrás iniciar sesión con tu usuario y contraseña de forma manual o accediendo con una cuenta de Google.

Screens screenshot of a login screen

Description automatically generated

Ilustración 3: Inicio de Sesión. Fuente

Una vez hayas iniciado sesión de forma manual o con Google, una vez clicado el *Sign in* o el *Sign in with Googl*e, iremos a la página principal, donde la aplicación te hará escoger el género de las noticias que escogerás. Depende en que genero cliques, te llevara a su respectiva pantalla. Como tenemos 5 géneros distintos, te llevara a 5 pantallas distintas donde podremos visualizar los titulares de las noticias según el género escogido.

A screenshot of a computer

Description automatically generated

Ilustración 4: Página Principal, elegir género. Fuente

Como hemos mencionado anteriormente depende de donde cliques iras a una página acorde con el género escogido, donde se te mostraran los titulares de los respectivos géneros requeridos. Primero mostraremos la distribución de cómo se muestran las pantallas. Y después explicaremos el funcionamiento de una de las ventanas, al tener todas las mismas características.

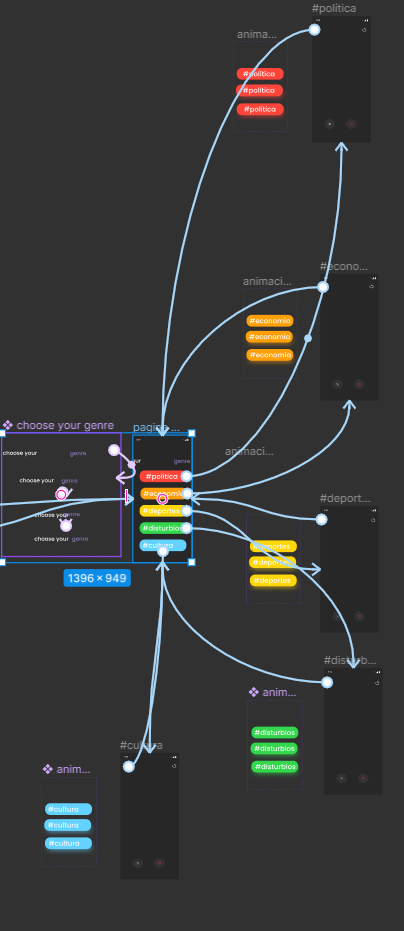


Ilustración 5: Distribución de pantalla principal a su género. Fuente

En la última pantalla, visualizaremos los titulares, pese a que no se ven titulares, vemos la pantalla y formato en el que lo visualizaremos. Empezando por la parte superior de la ventana, tenemos el icono de una flecha horizontal, que representa la vuelta a la pantalla principal en caso de que queramos ver titulares de noticias de otros géneros. El icono de la izquierda, en el que se ve una flecha redondeada, servirá para poder actualizar los titulares, donde el clic a este botón nos permitirá hacer una petición de refresh para que nos devuelva a tiempo real las noticias que hayan sido actualizadas des de la página web mediante el Script de Pyhton automatizado.

A screenshot of a phone

Description automatically generated

Ilustración 6: visualización de titulares, género político. Fuente

## Modelo de base de datos

En este apartado incluiremos el modelo Entidad-Relación de la base de datos de la aplicación. Primero mostraremos el modelo con sus relaciones. Y mas abajo detallaremos las tablas con sus atributos y sus respectivas funciones.

A screenshot of a computer screen

Description automatically generated

Ilustración 7: Modelo de Datos. Fuente

1. Usuario:
   1. idUsuario (PK): Identificador del usuario.
   2. correoElectronico: Correo electrónico del usuario (nombre de usuario).
   3. contrasena: Contraseña del usuario.
   4. fechaRegistro: Fecha de registro del usuario.
2. Genero:
   1. idGenero (PK): Identificador del género.
   2. nombreGenero: Nombre del género (Política, Economía, Cultura, Disturbios).
3. Noticia:
   1. idNoticia (PK): Identificador de la noticia.
   2. titular: Título de la noticia.
   3. fechaPublicacion: Fecha de publicación de la noticia.
   4. horaPublicacion: Hora de publicación de la noticia.
   5. **idGenero (FK): Clave foránea que referencia a la tabla Genero.**
4. EntradaSalidaControl:
   1. idRegistro (PK): Identificador único del registro.
   2. **idNoticia (FK): Foreign Key que referencia a la tabla Noticia.**
   3. **idUsuario (FK): Foreign Key que referencia a la tabla Usuario.**
   4. tipoRegistro: registra una entrada o salida.
   5. fechaRegistro: Fecha y hora del registro.

## Diseño de la arquitectura

La arquitectura muestra todos los componentes a utilizar en la solución, así como su relación entre ellos.

De forma general para una web se muestran los componentes principales (sin ser excluyentes):

* + 1. *Back-end*:
       1. Scripts de Python para *scraping* y limpieza de datos ETL.
       2. Servicios web utilizando *Flask* para la comunicación con la aplicación móvil.
       3. Conexión a la base de datos *MySQL* a través de *SQLAlchemy*.
    2. *Front-end*:
       1. Interfaz de usuario diseñada en *Figma*.
       2. Desarrollo de la aplicación móvil en Kotlin con Android Studio.
    3. Servidores:
       1. Servidor de aplicaciones: *Flask* para el *backend*.
    4. Gestor de Base de Datos:
       1. *MySQL* para almacenar datos de usuarios, noticias y preferencias.

Incluir una infografía de todos los componentes ayuda mucho a su entendimiento.

## Lenguajes utilizados

* + 1. *Python*:
       1. *BeautifulSoup*: para realizar scraping de la agencia EFE.
       2. *Pandas*: para limpiar la información de los titulares en formato de tabla.
       3. *SQLAlchemy*: para realizar peticiones a la base de datos y procesar la información de los *dataframes* creados.
       4. *Flask*: para la creación y gestión del api que sirva para gestionar la comunicación entre la base de datos y interfaz de la aplicación.
    2. *PowerShell*: para automatizar los scripts de Python.
    3. *MySQL*: para crear bases de datos, recopilar información a través del *scraping*, gestionar usuarios y almacenar información para su posterior análisis.
    4. *Figma*: para diseñar la interfaz de usuario.
    5. *Kotlin*: para desarrollar la aplicación.

# DISEÑO TECNICO:

## Libro de Estilo:

En este apartado aún no tengo muy claro el estilo que seguiré ya que por el momento estoy probando. Por el momento, lo sí que puedo decir es que la aplicación utilizara variables cortas y de fácil entendimiento. Para tener un entendimiento general de lectura en el código tanto los comentarios como las variables y componentes del código en cualquier parte del proyecto serán escritas en castellano. Por lo demás, necesito establecer un orden de escritura para mantener una integridad en el desarrollo del proyecto.

## Definición de componentes:

En esta parte del proyecto nos dedicaremos a explicar lo que tenemos hecho hasta el momento, en esta segunda entrega se nos especificó crear el proceso de ABM o alta/baja/ modificación del usuario en la aplicación. Para desarrollar esto es necesario seguir un seguido de pasos que nos permitirán realizar esto, incluyendo la segunda parte que es el diseño del modelo de datos.

## Web Scraper(Pyhton):

Como mencione el apartado anterior, he desarrollado un script de Python que scrapea los titulares de efe servicios.

A screenshot of a computer program

Description automatically generated

Ilustración 8 Web Scrape IMG1. Fuente

Arriba podemos ver las bibliotecas utilizadas, las urls y la lista donde almacenaremos los artículos. Para sacar todos los títulos de cada web creamos un for que recorra cada url, estableciendo conexión con ella. Si estalece la conexión, hacemos otro for anidado en el que extraeremos la información que queremos que es la fecha y el titular, y por cada url que recorra, le asignara un numero u otro empezando por política y acabando con cultura.

A screenshot of a computer

Description automatically generated

Ilustración 9: Web Scrape IMG2. fuente

A screenshot of a computer

Description automatically generated

Ilustración 10: Web Scrape IMG3.Fuente

A screenshot of a computer

Description automatically generated

Ilustración 11: Web Scrape IMG5. Fuente

En estas dos siguientes líneas de código haremos lo que se llama ETL, en el que simplemente limpiaremos los datos a nuestro antojo, para que queden mas legibles. Primero, crearemos una nueva columna donde extraeremos de la columna fecha o date, la hora y a los valores nulos le asignaremos 00:00. De la columna date extraeremos los saltos de line y otros símbolos y nos quedaremos solo con la fecha que pasaremos a formato día/mes/año.

A screenshot of a computer

Description automatically generated

Ilustración 12: Web Scrape IMG5. Fuente

A screenshot of a computer

Description automatically generated

Ilustración 13:Web Scrape IMG6. Fuente

A computer screen shot of a computer code

Description automatically generated

Ilustración 14:Web Scrape IMG6. Fuente

Una vez limpiado los datos, estableceremos conexión con la base de datos, utilizando el MySQL conector, estableciendo la contraseña el usuario y la base de datos que utilizaremos.

Una vez conectados buscamos a fecha y ora actuales y verificaremos mediante una Query de SQL en que verifique que inserte en la tabla aquellas noticias en la tabla que contengan noticias en la misma fecha o mayores, para que no se vayan repitiendo las noticias cuando automatizados el script. No queremos tuplas repetidas. Finalmente insertaremos cambiaremos las columnas del dataframe por las de SQL e insertaremos los datos dentro de la Base de datos que ya tenemos creada

A screenshot of a computer

Description automatically generated

Ilustración 15:Web Scrape(SQL demostracion) IMG7. Fuente

Como podemos ver arriba queda detallado el formato de como queda en la base de datos este script. Como por ahora estoy aun viendo como insertare los datos de forma automatizada, esto solo son dos o 3 lanzamientos de scripts con aproximadamente 120 registros para después poder utilizarla para ver si la api se conecta a la base de datos.

## 4.2.2 API:

Aquí explicaremos como hemos creado la api, que lenguaje hemos utilizado y el testeo ABM con postman, ya que pese a funcionar, no he podido establecer conexión con la interfaz en Android Studio.

A screenshot of a computer

Description automatically generated

Ilustración 16: API IMG1 . Fuente

Como podemos ver esta es la estructura que sigue nuestra api, como es muy grande solo explicaremos la mas importante que es la de Usuario, donde haremos el proceso de alta baja y modificación que pide esta segunda entrega del proyecto. Esta api la hemos hecho en Java con el framework de Springboot con el entorno del Intelij.

A screen shot of a computer

Description automatically generated

Ilustración 17: API IMG2. Fuente

Para establecer conexión con nuestra base de datos iremos a los application.propperties y daremos la url de la base de datos, la contraseña, el usuario, el permiso de creación o actualización de tablas, por lo que realmente la api ha creado las tablas en SQL, y finalmente el puerto donde se conectara, en este caso el 8082.

A screenshot of a computer program

Description automatically generated

Ilustración 18: API IMG3. Fuente

A screen shot of a computer program

Description automatically generated

Ilustración 19: API IMG4. Fuente

Crearemos la tabla y los constructores, estableciendo el valor que tendrá y las características de cada columna, en el objeto Usuario y usuarioDTO.

A screen shot of a computer

Description automatically generated

Ilustración 20:API IMG5. Fuente

Crearemos el Usuario repository para obtener los métodos prehechos de JPA repositorio para el objeto Usuario.

A screen shot of a computer program

Description automatically generated

Ilustración 21:API IMG6. Fuente

A screenshot of a computer program

Description automatically generated

Ilustración 22:API IMG7. Fuente

En esta parte vemos que hemos creado 2 servicios donde esableceremos la lógica de de negocio que en este caso es hacer un crud de la tabla usuarios pero que como hemos visto al principio este hecho con todas las tablas de la base de datos. En la fábrica permitiremos la conversión entre Usuarios y UsuariosDTO. En el UsuService, utilizaremos los métodos preestablecidos del interfaz de UsuarioRepository y su extensión, para hacer operaciones de crud, que serían el alta con save o put, y baja con deletebyID o delete.

A screen shot of a computer program

Description automatically generated

Ilustración 23:API IMG8. Fuente

Aquí establecemos os endpoints para las operaciones crud declaradas en el USuService, que serán serializadas en forma JSON, de forma automática con el @restConttroller. El Mapping, será la ruta de acceso al api a la que le daremos formato donde posteriormente accederemos mediante POSTMAN.

A screenshot of a video

Description automatically generated

Ilustración 24: API IMG9. Fuente

A screenshot of a computer

Description automatically generated

Ilustración 25: API IMG10. Fuente

A screenshot of a computer

Description automatically generated

Ilustración 26: API IMG11. Fuente

Como podemos ver hemos hecho modificaciones en la base de datos insertando y viendo los datos con get y put, me falta el delete que me da error ya que debo cambiar el id\_usuario por idusuario sin \_ este símbolo ya que me da problemas. Como no tengo tiempo esto queda por hacer. Pero se ve que la api ha comunicado con la base de datos y ha creado la base de datos des de la api.

## 4.2.3 Aplicación Interfaz:

Lamentablemente en esta parte hasta ahora no hemos podido establecer la comunicación HTTP con la interfaz que hemos explicado anteriormente. Esto será el siguiente paso. Aunque con el postman hemos demostrado que funciona la API.

Lo que si podremos ver es el diseño de como ha quedado la interfaz de usuario del login y el registro responsive que podrá ser visto cuando adjunte el código. También colgare un pequeño video donde se puede mostrar que cambia de pantalla y si tiene un orden.

A screenshot of a phone

Description automatically generated

Ilustración 27: Kotlin IMG1. Fuente

A screenshot of a phone

Description automatically generated

Ilustración 28: Kotlin IMG2. Fuente

## Diseño de Tablas de Modelo de Datos:

A screenshot of a computer

Description automatically generated

Ilustración 29: BD IMG1. Fuente

No entraremos mucho en detalle en la explicación de la base de datos, ya que en la primera entrega lo detallamos mejor, lo que si me gustaría comentar las dos vías que tengo para la tabla de entrada\_salida\_control.

* La primera idea es crear registros en el control a través de triggers o if else statements en sql basándonos en las entradas o salidas de los cierres de sesión de la aplicación, así como de acceso mediante el login, validado. Lo mismo cuando se introduzcan nuevos usuarios
* La segunda idea es un poco mas compleja y es conseguir crear a través de Android studio extraer la información de la aplicación donde el concepto de registrar información se va mucho de mis conocimientos.