



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**ANME - Aplicación web de
noticias, meteorología y
eventos**



Presentado por José Manuel Rodríguez Iglesias
en Universidad de Burgos — 8 de junio
de 2023

Tutores: Dr. Alfredo Bol Arreba, Dr. Bruno
Baruque Zanon y D. Héctor Cogollos Adrian



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



Dr. Alfredo Bol Arreba, profesor del departamento de Física.

Dr. Bruno Baruque Zanon, profesor del departamento de Ingeniería Informática, del área de Ciencia de la Computación e Inteligencia Artificial.

D. Héctor Cogollos Adrián, profesor del departamento de Ingeniería Informática, del área de Lenguajes y Sistemas Informáticos.

Expone:

Que el alumno D. José Manuel Rodríguez Iglesias, con DNI 49368325-K, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado título de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 8 de junio de 2023

Vº. Bº. del Tutor: Vº. Bº. del co-tutor: Vº. Bº. del co-tutor:

Dr. Alfredo Bol
Arreba

Dr. Bruno Baruque
Zanon

D. Héctor Cogollos
Adrián

Resumen

En la actualidad, no existen muchas aplicaciones que nos proporcionen información (noticias, meteorología, eventos, etc.) de una ciudad o un punto geográfico de interés para el usuario a tiempo real. En este proyecto se busca crear una aplicación web que nos permita consultar diferentes categorías de información para una ubicación agregadas de forma simple y conveniente en un solo portal web.

El objetivo principal es facilitar al usuario la consulta de información para una ubicación de España. Adicionalmente, el usuario podrá elegir la información que quiere ver en cada momento.

Descriptores

Aplicación web, Predicciones meteorológicas, Noticias, Eventos, Python, Servidor web

...

Abstract

At present, there aren't many applications that provide information about regarding a city (news, meteorology, events,etc) or geographic points of interest for the user in real time. The objective of this project is to create an application that allows us to consult different information just in one website. Furthermore, one other objective is bringing the consultation of information to the user for a Spanish position. In addition, the user will be able to choose the information they want at any time.

Keywords

Web app, weather predictions, news, events, Python, web server

Índice general

Índice general	iii
Índice de figuras	v
Índice de tablas	vi
Introducción	1
1.1. Estructura de la memoria	2
Objetivos del proyecto	3
2.1. Objetivos técnicos	3
2.2. Objetivos Personales	4
Conceptos teóricos	5
3.1. API	5
3.2. Servidor Web	6
3.3. Base de Datos	6
3.4. Servicio Web	7
3.5. Protocolo HTTP	8
3.6. API Endpoint	8
Técnicas y herramientas	11
4.1. Lenguajes	12
4.2. Herramientas para el desarrollo	15
4.3. Apis	18
4.4. Librerías	20
4.5. Metodologías	22

Aspectos relevantes del desarrollo del proyecto	25
5.1. Principio del proyecto	25
5.2. Metodologías	26
5.3. Aprendizaje Autónomo	27
5.4. Desarrollo de la App	28
5.5. Documentación	30
5.6. Integración Continua	30
5.7. Dificultades en el desarrollo del proyecto	31
5.8. Publicación de la App	32
Trabajos relacionados	33
6.1. Proyectos	33
Conclusiones y Líneas de trabajo futuras	39
7.1. Conclusiones	39
7.2. Líneas de trabajo futuras	40
Bibliografía	41

Índice de figuras

4.1.	Patrón Modelo-Vista-Presentador. [9]	24
5.1.	Logo de ANME	26
5.2.	Conexión a SQLite desde Python.	29
5.3.	Solicitud de información al servicio NewsData.io	30
5.4.	Aplicación ANME	32
6.1.	Previsión meteorológica de cuatro días	34
6.2.	Reporte Metar	34
6.3.	Previsión meteorológica	35
6.4.	App TicketMaster	36
6.5.	App Newsflow	36
6.6.	Microsoft Bing	37
6.7.	Google News	38

Índice de tablas

4.1. Herramientas y tecnologías usadas en el proyecto	11
4.2. Servicios (Apis) utilizados en el desarrollo del proyecto	12

Introducción

Lo primero de todo, quiero agradecer a mis tutores por la paciencia que han tenido conmigo. Desde el inicio del desarrollo del proyecto me han explicado de la mejor forma posible, aportando fuentes de información e ideas. Quiero dar las gracias a Bruno, Alfredo y Héctor, no sé si este proyecto hubiera podido salir hacia adelante sin vosotros.

En este proyecto se ha desarrollado una aplicación web que nos proporcione información (noticias, meteorología, eventos, calidad del aire, etc.) de una ciudad o un punto geográfico de interés para el usuario a tiempo real. Además, esta aplicación le permite al usuario planificar sus actividades al aire libre o hacer planes de viaje con anticipación.

Actualmente, existen aplicaciones web de noticias, meteorología, eventos, calidad del aire, etc. solo que estas aplicaciones permiten consultar una sola categoría de información. No hay muchas aplicaciones web que nos permitan consultar la información de noticias, meteorología, eventos, calidad del aire, etc. de una ciudad dentro de la misma aplicación.

ANME tiene el propósito de crear una aplicación web nos proporcione información de una ubicación de España a tiempo real usando diferentes APIs. Para esto se ha utilizado el tipo de integración de datos ETL (Extract, Transform, Load).

Aunque no sea la única aplicación web que proporcione información de noticias, eventos y meteorología, el proyecto me ha permitido aprender de integración de APIs y ha sido un ejercicio de aprendizaje para iniciarme en el mundo del desarrollo web.

1.1. Estructura de la memoria

- **Objetivos del proyecto:** En esta sección vamos a enumerar los objetivos que proponemos lograr en el desarrollo de este proyecto.
- **Conceptos teóricos:** En este apartado vamos a explicar los conceptos teóricos claves para poder entender la solución propuesta.
- **Técnicas y herramientas:** En este apartado vamos a enumerar y a explicar todas las herramientas utilizadas en la realización del proyecto ANME. Además, explicaremos como las hemos utilizado nosotros.
- **Aspectos relevantes del desarrollo:** En esta sección destacaremos los aspectos más relevantes del desarrollo llevado a cabo.
- **Trabajos relacionados:** En este apartado se pretende mostrar proyectos similares a ANME disponibles en el mercado.
- **Conclusiones y líneas futuras:** En este apartado dialogaremos las conclusiones que hemos obtenido al finalizar el proyecto y revisaremos si se han cumplido todos los objetivos previstos.

Objetivos del proyecto

La propuesta central del proyecto consiste en simplificar la tarea de búsqueda de información local para los usuarios, con el objetivo de facilitar el acceso a esta información de manera que puedan encontrar la información de manera más sencilla y rápida. En resumen, se busca proporcionar una experiencia más fácil y cómoda para el usuario al buscar información de eventos programados en las ciudades de España.

2.1. Objetivos técnicos

1. Desarrollar una aplicación web que ofrezca información (noticias, meteorología, eventos, etc.) relacionados con una ciudad o un punto geográfico de interés para el usuario.
2. La aplicación web debe ser capaz de extraer la información a tiempo real de las APIs. Se han seleccionado varias APIs que proporcionan información a nuestra app.
3. Ofrecer una aplicación segura que proteja la información de los usuarios.
4. Facilitar la interpretación de la información mediante una interfaz de usuario (UI) simple, moderna y accesible.
5. Recopilar todos los datos solicitados de las APIs de manera ordenada y poder acceder a ellos de una forma sencilla.
6. Aportar información extra al usuario que ayude a tomar decisiones relacionadas con el clima.

7. Publicar la aplicación web en un servidor bajo un dominio propio.
8. Desarrollar una interfaz que sea responsive design y fácil de utilizar para el usuario.

2.2. Objetivos Personales

1. Desarrollar un proyecto personal donde se apliquen los conocimientos aprendidos en la titulación del Grado de *Ingeniería Informática*.
2. Aumentar la capacidad creativa mediante el planteamiento y resolución de un problema real.
3. Aprendizaje autónomo en nuevos temas relacionados con la *Ingeniería Informática*.
4. Capacidad de exposición en público del proyecto personal desarrollado, como por ejemplo la argumentación de las elecciones tomadas, etc.
5. Ser capaz de comunicar correctamente en otro idioma un resumen del trabajo realizado.
6. Profundizar en el desarrollo de aplicaciones Web.

Conceptos teóricos

En esta sección se explicará los conceptos teóricos que son necesarios para poder entender bien el desarrollo del presente proyecto.

3.1. API

Una API [12] puede definirse como un grupo de reglas y protocolos que especifican cómo los programas pueden interactuar entre sí. En el contexto del desarrollo de software, una API permite que un programa utilice la funcionalidad de otro programa o servicio de una manera estandarizada y segura.

Una API puede ser pública o privada. Una API pública ayuda a los programadores a crear aplicaciones que interactúen con un servicio web, mientras que una API privada está orientada para que los desarrolladores puedan interactuar con los datos y servicios de una empresa.

Las APIs mayoritariamente se utilizan para la integración de servicios de terceros en una aplicación, la automatización de procesos de negocio, la extracción de datos y la realización de aplicaciones móviles.

En resumen, una API es una forma estandarizada de permitir que los programas interactúen entre sí y aprovechen la funcionalidad de otros servicios o aplicaciones. Además, es una herramienta muy necesaria para el desarrollo de aplicaciones modernas y la automatización de procesos de negocio.

3.2. Servidor Web

Un servidor web [4] es un programa que se ejecuta en un ordenador y que es capaz de recibir y responder a peticiones HTTP de otros ordenadores en la red, normalmente a través de Internet. Es el encargado de recibir las solicitudes, procesarlas y enviarles las respuestas a los clientes correspondientes.

Para que un servidor web pueda procesar las solicitudes, debe estar configurado correctamente con el software y los archivos necesarios. Los servidores web más comunes son Apache, Nginx e IIS, aunque existen muchos otros.

Cuando un cliente solicita una página web a través de un navegador web, la petición se envía al servidor web correspondiente. El servidor web busca los archivos necesarios para construir la página web, los procesa y envía la respuesta al cliente. En general, la respuesta del servidor web es un archivo HTML, aunque también puede ser otro tipo de archivo como una imagen, un archivo de audio o un archivo de video.

Además de servir páginas web estáticas, los servidores web también pueden ejecutar programas en el servidor y enviar al cliente los resultados generados. Esto es especialmente útil para aplicaciones web complejas que requieren procesamiento del lado del servidor. Estas aplicaciones principalmente son desarrolladas en PHP, Python, Ruby y Java, entre otros.

En resumen, un servidor web es aquel que permite a los clientes solicitar y recibir información a través de Internet. Es una parte esencial de la infraestructura de la World Wide Web y permite la entrega rápida y eficiente de contenido web a usuarios de todo el mundo.

3.3. Base de Datos

Una base de datos [10] consiste en un conjunto de datos que se almacena en un dispositivo de manera organizada y se puede acceder y manipular de manera eficiente.

La organización de una base de datos consiste en tablas que contienen columnas y filas. Cada registro o entrada en la base de datos se representa mediante una fila, mientras que cada campo relacionado con el registro se representa mediante una columna.

Dependiendo del tipo de información que deseemos almacenar y de la forma en que se necesita acceder a ella, utilizaremos un tipo de base de datos u otro.

Además, las bases de datos se pueden acceder y actualizar mediante consultas y comandos específicos, lo que facilita la recuperación y actualización de información en grandes cantidades. Los usuarios pueden realizar consultas para buscar y filtrar información específica, o pueden utilizar comandos para agregar, modificar o eliminar datos en la base de datos.

El manejo de la información en una base de datos se gestiona mediante sistemas de gestión de bases de datos (DBMS), los cuales son programas diseñados para crear, modificar y gestionar bases de datos.

3.4. Servicio Web

Un servicio web [5] es una aplicación software que se ejecuta en un servidor y que permite a los usuarios realizar ciertas tareas o acceder a cierta información a través de una red, como Internet. Este tipo de aplicaciones se basan en el uso de estándares abiertos como HTTP, XML y SOAP, y se pueden acceder a través de una variedad de protocolos, como HTTP, FTP y SMTP.

Un servicio web consta de tres componentes principales: el proveedor de servicios, que es el servidor que aloja el servicio web y que recibe las solicitudes de los usuarios; el consumidor de servicios, que es la aplicación o sistema que utiliza el servicio web para realizar tareas o acceder a información; y la descripción de servicios, que es una especificación formal del servicio web que describe sus funcionalidades y cómo acceder a ellas.

Los servicios web pueden ser de dos tipos: basados en REST y basados en SOAP. Los servicios basados en REST (Representational State Transfer) utilizan los verbos HTTP (GET, POST, PUT, DELETE, etc.) para manipular los recursos y se basan en la representación de estos recursos en un formato específico, como XML o JSON. Por otro lado, los servicios basados en SOAP (Simple Object Access Protocol) utilizan un formato de mensaje XML para transmitir información y se basan en la definición de servicios como WSDL.

Los servicios web tienen una gran diversidad de aplicaciones, desde la integración de aplicaciones y sistemas hasta la automatización de procesos de negocio y el desarrollo de aplicaciones móviles. Además, han sido clave en la transformación de la web hacia el Internet de las cosas, al permitir la

interconexión y comunicación entre una amplia variedad de dispositivos y sistemas.

3.5. Protocolo HTTP

El protocolo HTTP [11] es un grupo de reglas que establecen cómo se deben intercambiar la información entre el cliente y el servidor en la Web. Se considera que es un protocolo de capa de aplicación que se utiliza en el nivel superior de la pila de protocolos de Internet.

En el modelo cliente-servidor, el cliente (navegador web) envía una solicitud HTTP al servidor web, que a su vez envía una respuesta HTTP al cliente. La solicitud y la respuesta se envían en formato de texto plano, lo que significa que se pueden leer y escribir manualmente, lo que lo hace muy accesible para los desarrolladores.

HTTP utiliza métodos de solicitud, como GET, POST, PUT y DELETE, entre otros, para indicar la acción que el cliente desea realizar sobre un recurso determinado en el servidor. El método GET se utiliza para solicitar recursos, mientras que el método POST se utiliza para enviar datos a un servidor web.

HTTP también utiliza códigos de estado, como el código 200 (Todo Ok), 404 (No encontrado) y 500 Error interno en el Servidor, entre otros, para indicar el resultado de la solicitud realizada. Estos códigos de estado son importantes porque permiten a los desarrolladores identificar rápidamente los problemas que puedan aparecer durante el intercambio de información entre el cliente y el servidor.

Una de las principales ventajas de HTTP es que es un protocolo sin estado. Esto significa que cada solicitud se considera independiente de las anteriores, lo que hace que las respuestas del servidor sean más rápidas y eficientes. Sin embargo, esto también significa que HTTP no puede mantener información sobre las solicitudes anteriores del cliente, lo que a veces puede ser un problema.

3.6. API Endpoint

Un API endpoint [8] es un punto de acceso en una API que permite que las aplicaciones se comuniquen con un servicio web o con un servidor web de una manera específica y estructurada. Se trata de una URL (Uniform

Resource Locator) específica que un cliente utiliza para acceder a una API y acceder a recursos específicos.

Los endpoints de una API pueden ser creados y personalizados por los desarrolladores de software para permitir que las aplicaciones accedan a los datos y servicios que se encuentran en un servidor web o en un servicio web. Estos puntos finales pueden ser utilizados para solicitar datos, enviar datos, actualizar datos o eliminar datos.

Los endpoints de una API están diseñados para ser utilizados por desarrolladores de aplicaciones, y se comunican con los clientes utilizando un protocolo de comunicación específico, como HTTP. Los clientes envían solicitudes HTTP a los puntos finales de la API y reciben respuestas en formato de datos estructurados, como JSON o XML.

Además, los endpoints de la API pueden ser públicos o privados, lo que significa que pueden estar disponibles para todos los usuarios o solo para los usuarios autorizados. Además, un solo servicio o aplicación puede tener múltiples puntos finales de API, cada uno diseñado para un propósito específico.

Técnicas y herramientas

En esta sección explicaremos detalladamente qué técnicas y herramientas hemos utilizado en cada parte del proyecto. A continuación se adjuntan dos tablas, en la Tabla 4.1 se muestra que herramientas y tecnologías hemos utilizado en cada parte del proyecto ,y en la Tabla 4.2 se muestra que servicios (Apis) hemos utilizado.

Herramientas	Front-end	Back-end	Documentación	SCRUM
Python		Pág 12		
HTML	Pág 13			
CSS	Pág 13			
SQL		Pág 14		
Latex			Pág 14	
Jira				Pág 15
GitHub		Pág 15		
Git		Pág 16		
Docker		Pág 17		
BootStrap	Pág 17			
Flask	Pág 18			
JSON		Pág 14		
SQLite		Pág 18		

Tabla 4.1: Herramientas y tecnologías usadas en el proyecto

Apis	Front-end	Back-end
OpenWeatherMaps	Pág 19	
TuTiempo	Pág 19	
NewData.io	Pág 19	
TicketMaster	Pág 19	

Tabla 4.2: Servicios (Apis) utilizados en el desarrollo del proyecto

4.1. Lenguajes

En esta sección explicaremos los lenguajes de programación que hemos utilizado para desarrollar nuestro proyecto. A continuación se detallará cada uno de ellos:

Python

Python es un lenguaje de programación que se caracteriza por su facilidad de uso y legibilidad. Es utilizado en múltiples aplicaciones, desde el desarrollo web hasta la inteligencia artificial. Python es un lenguaje interpretado, por lo que no necesita ser compilado antes de ser ejecutado, lo que lo hace más rápido para desarrollar y probar. Además, Python tiene una gran biblioteca estándar que proporciona una amplia gama de módulos para realizar tareas comunes. En general, este lenguaje de programación es muy conocido ,y además permite beneficiarse a cualquier persona con interés en la programación.

Python es el lenguaje que hemos utilizado para el desarrollo del modelo, es decir, creación del backned. Se ha utilizado este lenguaje porque nos proporciona todas las librerías necesarias tanto para el desarrollo del modelo como el desarrollo del presentador.

Además, este lenguaje nos ha permitido crear scripts con muy poco código comparado con otros lenguajes de programación. En nuestro caso hemos utilizado la versión 3.8.6 .

- Puedes consultar más información sobre *Python* desde el siguiente enlace: <https://www.python.org/>

HTML

HTML es un lenguaje de programación simple que se utiliza para estructurar el contenido de una página web y definir cómo se debe presentar en un navegador web. Se puede utilizar en diferentes navegadores web y plataformas.

Este lenguaje utiliza una serie de etiquetas o "tags" que se utilizan para definir diferentes elementos en una página web, como títulos, párrafos, imágenes, enlaces y listas. Estas etiquetas son interpretadas por el navegador web y son usadas para presentar el contenido en una página web de manera organizada y estructurada.

A parte de permitir la estructura básica de una página web, HTML también permite la inclusión de otros elementos como formularios para la entrada de datos y la integración de elementos multimedia, como vídeos y audio.

HTML es el lenguaje que hemos utilizado para el desarrollo del front-end, es decir, para generar las templates de la aplicación. Se ha elegido este lenguaje porque es el más utilizado para crear la parte visual y de interacción en las aplicaciones web, conocida como front-end.

- Puedes consultar más información sobre *HTML* desde el siguiente enlace: <https://html.com/>

CSS

CSS es un lenguaje de diseño utilizado para definir el estilo visual de una página web. Se utiliza en conjunto con HTML para dar formato y estilo a los componentes de una web, como por ejemplo el color, la tipografía, el tamaño, la posición y la disposición de los elementos.^[1]

Este lenguaje nos permite desvincular entre el diseño y el contenido de una página web, lo que nos ayuda a que el código sea más fácil de entender y de mantener. Además, CSS permite crear estilos reutilizables para aplicarlos a múltiples elementos en una página web, lo que hace que el diseño sea más eficiente y coherente.

CSS es el lenguaje que hemos utilizado para dar estilo a las templates de nuestra aplicación web. También hemos usado dos frameworks de prototipados llamados bootstrap y Font Awesom.

- Puedes consultar más información sobre *CSS* desde el siguiente enlace:
<https://css.com/>

SQL

SQL es un lenguaje de programación que se compone de un conjunto de comandos y declaraciones que permiten interactuar con una base de datos[7].

Con SQL, es posible crear y definir tablas, insertar y modificar datos en ellas, buscar y recuperar información, y realizar operaciones complejas como unir varias tablas y filtrar datos.[2].

Este lenguaje es utilizado en múltiples aplicaciones, desde la gestión de datos de empresas y organizaciones hasta la creación de sitios web dinámicos. Es un lenguaje muy versátil y poderoso que permite a los programadores manipular grandes cantidades de datos de manera efectiva y eficiente.

- Puedes consultar más información sobre SQL desde el siguiente enlace:
<https://www.mysql.com/>

LaTeX

LaTeX es el sistema de creación de textos que hemos utilizado para generar nuestra memoria sobre el desarrollo del proyecto que estás leyendo, así como los anexos.

En lugar de formatear el texto manualmente, LaTeX utiliza comandos y estructuras predefinidas para crear y organizar el contenido del documento. Esto permite a los usuarios no preocuparse por el formato, sino estar centrado en el contenido del documento.

Una vez que se domina, LaTeX puede ser una herramienta muy eficaz para producir documentos de alta calidad con un aspecto profesional.

- Puedes consultar más información sobre LaTeX desde la siguiente URL: <https://www.latex-project.org/>

JSON

JSON no se considera como un lenguaje de programación, sino como un archivo formado por datos estructurados que se utiliza para enviar información entre diferentes sistemas

Los datos en JSON se representan en pares clave-valor, donde la clave es una cadena que identifica el valor correspondiente.

Además, JSON es compatible con muchos lenguajes de programación, lo que lo hace fácil de implementar en cualquier aplicación web o móvil. Es una herramienta muy necesaria en el desarrollo de aplicaciones web modernas, ya que permite el intercambio de información entre distintos dispositivos.

- Puedes consultar más información sobre JSON desde la siguiente URL:
<https://www.json.org/json-es.html>

4.2. Herramientas para el desarrollo

Jira

Jira es una herramienta que ayuda a gestionar las tareas y actividades del equipo de trabajo. Gracias a esta herramienta podemos seguir los principios ágiles, la cual nos permite realizar un seguimiento de nuestro trabajo.

Además, permite a los equipos colaborar y organizar su trabajo en sprints o iteraciones, así como también realizar un seguimiento del progreso y la resolución de problemas. También ofrece funciones avanzadas como paneles de control personalizables, informes y automatización de flujos de trabajo.

En este proyecto hemos utilizado esta herramienta para gestionar las tareas del desarrollo y poder realizar un seguimiento del progreso del mismo.

- Puedes consultar más información sobre *Jira* desde el siguiente enlace:
<https://www.atlassian.com/es/software/jira>

GitHub

GitHub es una plataforma de alojamiento y gestión de proyectos de software basada en la nube. Es un lugar donde los desarrolladores pueden almacenar y colaborar en proyectos de software, seguir el historial de cambios y contribuir al código de otros desarrolladores. La plataforma utiliza Git, un sistema de control de versiones distribuido, para gestionar el código fuente y permite a los desarrolladores trabajar en equipo en proyectos de software de forma remota. Además, GitHub proporciona herramientas para la revisión de código, la gestión de problemas, la automatización de flujos de trabajo y la integración con otras herramientas populares de desarrollo de software.

Esta herramienta ha sido utilizada en el desarrollo del proyecto para tener una copia de seguridad del código del proyecto y poder ver los cambios realizados de forma progresiva.

- Puedes consultar más información sobre *GitHub* desde el siguiente enlace: <https://github.com/>

Git

Git es una herramienta que se utiliza para tener el control de versiones y la gestión de proyectos de desarrollo de software.

Permite a los desarrolladores trabajar en el mismo código fuente de forma colaborativa y mantener un registro completo de los cambios en el código, lo que facilita la coordinación y el seguimiento del progreso del proyecto. Además, Git también ofrece herramientas para la gestión de ramas y fusiones de código, lo que permite a los desarrolladores trabajar en diferentes características o correcciones de errores de forma aislada y luego combinarlas sin conflictos.

En el presente proyecto se ha utilizado la herramienta git para tener un control de versiones del código y mantener un registro de los cambios realizados previamente.

- Puedes consultar más información sobre *Git* desde el siguiente enlace: <https://git-scm.com/>

StarUML

StarUML es una herramienta de modelado de software que se utiliza para diseñar y visualizar diagramas UML. Es especialmente útil para la creación de diagramas de casos de uso, de clases, de secuencia u otros tipos de diagramas UML que se utilizan para visualizar y especificar el diseño de un sistema de software.

Esta herramienta es utilizada por desarrolladores de software, arquitectos de software y otros profesionales en todo el mundo para facilitar el proceso de diseño y desarrollo de software. Además, esta herramienta es gratuita y de código abierto.

En el presente proyecto se ha sido utilizado esta herramienta para realizar el modelado de software, es decir, los diagramas de casos de uso, de secuencia y de arquitectura.

- Puedes consultar más información sobre *StartUML* desde el siguiente enlace: <https://sourceforge.net/projects/staruml/>

Docker

Docker es una plataforma de contenedores de software que permite a los desarrolladores crear, distribuir y ejecutar aplicaciones en entornos aislados y portátiles. Utiliza contenedores, que son entornos de software aislados que incluyen todas las dependencias necesarias para ejecutar una aplicación. Esto permite a los desarrolladores crear aplicaciones en un entorno local y luego distribuirlas de manera confiable en diferentes entornos de producción, independientemente del sistema operativo o la infraestructura subyacente.

Esta plataforma ofrece una solución eficiente para la implementación de aplicaciones en diferentes entornos de ejecución, lo que facilita la portabilidad de las aplicaciones y reduce los problemas de compatibilidad. Además, también proporciona herramientas para la gestión de contenedores, el escalado de aplicaciones y la automatización de flujos de trabajo.

Gracias a esta plataforma se ha conseguido hacer el despliegue de nuestra aplicación en un servicio de alojamiento (hosting) con el fin de ser utilizada por los usuarios finales.

- Puedes consultar más información sobre *Docker* desde el siguiente enlace: <https://www.docker.com/>

Bootstrap

Bootstrap es un *framework* CSS y Javascript diseñado para crear interfaces limpias y con un diseño responsive en el mundo del desarrollo web. Además, nos proporciona diferentes herramientas y funcionalidades que nos permite crear una web desde cero muy fácilmente.

Este framework ha sido utilizado en el desarrollo del presente proyecto para crear la interfaz de usuario de nuestra aplicación de manera sencilla y rápida.

Gracias a Bootstrap, podemos ahorrar tiempo en el diseño de páginas web responsivas.

- Puedes consultar más información sobre Bootstrap desde el siguiente enlace: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>

Flask

Flask es un framework de desarrollo web escrito en Python que se utiliza para desarrollar aplicaciones web de manera rápida y sencilla, ofreciendo las herramientas necesarias para crear un servidor web y manejar solicitudes HTTP. Además, este framework sigue el paradigma de arquitectura Modelo-Vista-Controlador y se enfoca en la simplicidad.

Este micro-framework es conocido por su flexibilidad y por permitir a los desarrolladores construir aplicaciones web altamente personalizadas y adaptadas a sus necesidades específicas. Ofrece soporte para la integración con bases de datos, autenticación de usuarios, gestión de sesiones y otros aspectos importantes de las aplicaciones web modernas.

- Puedes consultar más información sobre *Flask* desde el siguiente enlace: <https://flask.palletsprojects.com/en/2.2.x/>

SQLite

SQLite es una herramienta ligera y flexible para la gestión de bases de datos que permite a los desarrolladores almacenar y manipular datos de forma eficiente en aplicaciones locales. A diferencia de otros sistemas, no funciona como un servidor independiente, sino que se ejecuta como una biblioteca dentro de una aplicación. Es una base de datos de archivo único, fácil de transportar y distribuir.

Esta base de datos es bastante utilizada en aplicaciones móviles y de escritorio con volúmenes de datos pequeños. Además, tiene un tamaño reducido y consume pocos recursos.

Esta herramienta ha sido utilizada en el presente proyecto para almacenar y manipular la información de nuestra aplicación de manera eficiente.

- Puedes consultar más información sobre SQLite desde el siguiente enlace: <https://www.sqlite.com/index.html>

4.3. Apis

En esta sección explicaremos detalladamente las *APIs* que hemos utilizado para realizar nuestra aplicación web.

OpenWeatherMaps

OpenWeatherMaps es un servicio online que proporciona datos y pronósticos meteorológicos inspirados en OpenStreetMap. Además, este servicio proporciona varios de sus servicios de manera gratuita, y utiliza distintas fuentes de datos como por ejemplo estaciones meteorológicas, radares, etc. Proporciona una API que permite realizar hasta 96 llamadas por minuto de hasta 200.000 ciudades diferentes.

- Puedes consultar más información sobre OpenWeatherMaps desde la siguiente URL: <https://openweathermap.org/api>

TuTiempo

TuTiempo es un servicio online que proporciona el pronóstico del tiempo para 7 días y datos horarios para las próximas 24 horas. Además, este servicio proporciona la información de manera gratuita. Proporciona una API que permite realizar hasta 65 llamadas por minuto.

- Puedes consultar más información sobre TuTiempo desde la siguiente URL: <https://api.tutiempo.net/>

NewsData.io

NewsData.io es un servicio online que brinda acceso a artículos de noticias de todo el mundo. Este servicio recopila noticias de más de, 9495 fuentes de noticias que cubren alrededor de 124 países en 62 idiomas. Proporciona una API que permite realizar 200 llamadas al día y ofrece poder buscar artículos en función de palabras claves, ubicación, fechas, idioma.

- Puedes consultar más información sobre NewsData.io desde la siguiente URL: <https://newsdata.io/documentation>

TicketMaster

Ticketmaster es un servicio online que proporciona información detallada de más de 230.000 eventos, atracciones y lugares disponibles de diferentes países. Además, este servicio proporciona toda la información de manera gratuita, y utiliza distintas fuentes de datos. Proporciona una API que permite realizar cinco llamadas por minuto con un límite de 5000 llamadas por día.

- Puedes consultar más información sobre Ticketmaster desde la siguiente URL: <https://developer.ticketmaster.com/>

4.4. Librerías

Requests

La librería *Requests* de Python es un estándar que nos permite realizar solicitudes HTTP cuando se está desarrollando la parte del servidor de una página web. Gracias a esta librería se simplifica todo en una API simple. También es importante saber que el protocolo HTTP hace referencia al protocolo de solicitud y respuesta basado en la arquitectura cliente-servidor.

La arquitectura cliente-servidor se basa en conexiones TCP/IP mediante las cuales se intercambian mensajes de respuesta y solicitud.

- Puedes consultar más información sobre la librería Requests desde el siguiente enlace: <https://docs.python-requests.org/en/v2.0.0/>

Threading

Threading es una librería de Python que permite a los programadores crear y controlar múltiples hilos de ejecución en una aplicación. Los hilos de ejecución son procesos que se ejecutan de manera simultánea y permiten que una aplicación realice varias tareas al mismo tiempo.

Esta librería ofrece una serie de funciones que permiten crear, iniciar y controlar hilos de ejecución, lo que facilita la creación de aplicaciones multihilo. Además, permite el uso de bloqueos y semáforos, que ayudan a prevenir errores de concurrencia y a garantizar que los hilos no interfieran entre sí.

- Puedes consultar más información sobre la librería *Threading* desde el siguiente enlace: <https://docs.python.org/3/library/threading.html>

Geopy

Geopy es una librería de Python que permite a los desarrolladores a realizar tareas de geolocalización y cálculo de distancias geográficas.

Con Geopy, los desarrolladores pueden buscar la ubicación geográfica de una dirección, encontrar la distancia entre dos ubicaciones geográficas, o calcular rutas entre dos puntos en un mapa. Además, esta es compatible con diferentes servicios de geolocalización, como Google Maps, OpenStreetMap y Bing Maps.

- Puedes consultar más información sobre la librería *Geopy* desde el siguiente enlace: <https://pypi.org/project/geopy/>

Folium

Folium es una librería muy poderosa de Python que facilita la visualización de datos que han sido manipulados en Python en un mapa de folleto interactivo. Está inspirada en las librerías leaflet.js, OpenStreetMap, Mapbox y Stamen. También permite vincular datos a un mapa con visualizaciones Choropleth, así como pasar visualizaciones en vector/raster/HTML como marcadores en el mapa.

- Puedes consultar más información sobre la librería *Folium* desde el siguiente enlace: <https://pypi.org/project/folium/>

Datetime

Datetime es una librería estándar de Python inspirada en Robot Framework que permite la creación y conversión de valores de fecha y hora. Algunos ejemplos de esta librería pueden ser: obtener la fecha actual, convertir hora, crear hora, etc.

- Puedes consultar más información sobre la librería *Datetime* desde el siguiente enlace: <https://docs.python.org/3/library/datetime.html>

OS

OS es una librería de Python que proporciona funciones para interactuar con el sistema operativo. Esta librería permite usar de forma portátil la funcionalidad dependiente del sistema operativo. Además, esta también se utiliza para interactuar con el sistema de ficheros.

- Puedes consultar más información sobre la librería *OS* desde el siguiente enlace: <https://docs.python.org/es/3.10/library/os.html>

Functools

Functools es una librería de Python que proporciona funciones útiles para trabajar con funciones de orden superior, es decir, funciones que operan con otras funciones. Esta librería es particularmente útil para desarrolladores que trabajan con programación funcional en Python, y ofrece varias herramientas para mejorar el rendimiento y la eficiencia de las funciones de orden superior.

- Puedes consultar más información sobre la librería *Functools* desde el siguiente enlace: <https://docs.python.org/es/3/library/functools.html>

4.5. Metodologías

En esta sección explicaremos las metodologías que hemos utilizado para la gestión del trabajo y de la arquitectura en la que se basa nuestra aplicación. La metodología que hemos usado para la gestión del trabajo ha sido la metodología SCRUM y para el diseño de la arquitectura hemos utilizado el patrón de arquitectura MVP (Model-View-Presenter). A continuación se detallarán cada una de las metodologías:

Metodología SCRUM

SCRUM es una metodología ágil de gestión de proyectos que se utiliza para desarrollar productos de manera iterativa e incremental. Se enfoca en la colaboración, la retroalimentación constante y la adaptación al cambio.

Esta metodología implica la creación de un equipo multidisciplinario y autoorganizado que trabaja en sprints (iteraciones) cortos, que suelen durar entre 1 y 4 semanas.

Gracias a esta metodología podemos ser flexibles a cambios espontáneos, es decir, si los requisitos del proyecto cambian repentinamente por parte del cliente.

Fases

1. **Planificación (*Product Backlog*):** En esta fase establecemos las tareas prioritarias y obtenemos información detallada sobre el desarrollo de nuestro proyecto. Todas las tareas establecidas las añadimos a

nuestro Product Backlog. Una vez que tenemos listo nuestro Product Backlog, comenzamos el sprint.

2. **Ejecución (*Sprint:*)** Durante esta fase desarrollaremos las tareas establecidas en la fase de *Planificación* durante un periodo de dos semanas.
3. **Control (*Burn Down:*)** En esta fase el equipo desarrollo se reúne para demostrar el trabajo completado y recibir comentarios del Product Owner y otros interesados.

Model-View-Presenter (MVP)

MVP es una arquitectura de software que se utiliza para desarrollar aplicaciones. Este patrón consta de tres fases que son modelo, vista y presentador. A continuación se detallará cada una de ellas:

- **Model:** es el componente que representa la lógica de negocio de la aplicación y su estado interno. Es el encargado de realizar las operaciones en los datos y de validar la información ingresada por el usuario. Este componente es independiente de la interfaz de usuario.
- **View:** es el componente que se encarga de mostrar la información al usuario final. Su función es presentar la información del modelo de una forma clara y sencilla para que el usuario pueda interactuar con ella.
- **Presenter:** es el componente que comunica al modelo y a la vista. Es el encargado de interpretar las acciones del usuario en la vista y actualizar el modelo en consecuencia. También es responsable de actualizar la vista con los cambios en el modelo.

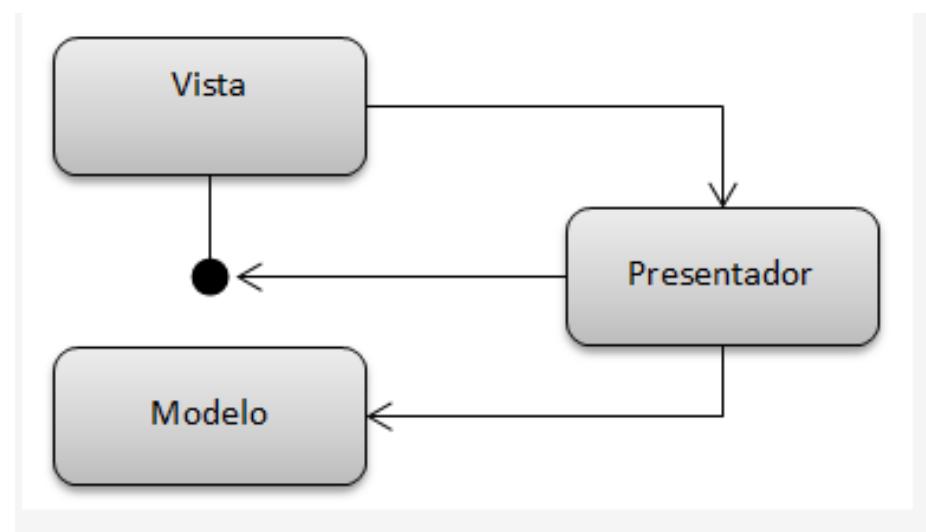


Figura 4.1: Patrón Modelo-Vista-Presentador. [9]

Aspectos relevantes del desarrollo del proyecto

En este apartado se detallarán los aspectos considerados más relevantes del presente proyecto. Desde los problemas que nos han surgido hasta los retos enfrentados en el proyecto.

5.1. Principio del proyecto

El tema del proyecto apareció de la inquietud de querer aprender desarrollo web y poder publicar algún día mi propia aplicación para que todos los usuarios puedan hacer uso de ella. Desde pequeño siempre he tenido la inquietud de querer desarrollar mis propias aplicaciones.

Por otra parte, los conocimientos adquiridos durante la carrera de Ingeniería Informática me han ayudado a pensar soluciones tecnológicas para cada uno de los problemas enfrentados en el desarrollo del proyecto.

Por ello pensamos en desarrollar una aplicación web, la cual ofreciera información de lugares de interés para el usuario.

Una vez que los tutores dieron por válida la idea del proyecto, comenzamos a trabajar desde el primer momento. De esta manera fue como apareció la idea de desarrollar la aplicación web ANME.



Figura 5.1: Logo de ANME

.4

5.2. Metodologías

Desde el primer día se tuvo en cuenta que el proyecto sería llevado a cabo de una manera que reflejase un alto nivel de profesionalismo. Para ello, se ha seguido la metodología SCRUM, que a continuación detallaremos.

La metodología SCRUM ha sido empleada para gestionar el proyecto, la cual nos ha permitido llevar un orden y un feedback de las tareas llevadas a cabo en cada sprint. Desde el inicio, el desarrollo se ha dividido en sprints de una duración 15 días aproximadamente.

Para cada ciclo de trabajo del proyecto (conocido como sprint), se establecieron una serie de objetivos y se asignó un valor a cada uno de ellos utilizando la técnica de estimación de *Story points*.

Para administrar el proyecto se ha empleado la herramienta Jira, que nos proporciona un tablero de tareas donde podemos mover las tareas de una etapa a otra. Una vez que una tarea está completa, se debe mover al estado Done en el *tablero*.

Al finalizar cada ciclo de trabajo del proyecto (sprint), se llevaron a cabo reuniones para revisar el trabajo realizado y planificar el siguiente sprint. Para planificar el sprint, se creó una lista de tareas a realizar.

El objetivo principal al concluir cada sprint era entregar una parte funcional del producto final.

5.3. Aprendizaje Autónomo

Para poder desarrollar el proyecto se han requerido varios conocimientos técnicos de los cuales no se habían tratado en el grado de Ingeniería Informática. Sobre todo, relacionados con el desarrollo web. A continuación se detallará cada uno de ellos y las fuentes utilizadas para su formación.

Funcionamiento de APIs

Para crear la aplicación web se propusieron varios servicios (APIs), de los cuales no sabíamos como funcionaban ni en qué formato devolvían la información. Para entender cómo funcionaba cada uno de los servicios tuvimos que investigar sobre la documentación de cada servicio, así de esta forma pudimos entender como solicitar la información que nosotros deseábamos a la API.

Para la formación del funcionamiento de las APIs se leyeron los siguientes recursos:

- Documentación de la API *OpenWeatherMaps*: <https://openweathermap.org/api>
- Documentación de la API *TuTiempo*: <https://api.tutiempo.net/>
- Documentación de la API *NewData.io*: <https://newsdata.io/documentation>
- Documentación de la API *TicketMaster*: <https://developer.ticketmaster.com/>

Flask

Al principio de todo se propuso la opción de utilizar Django [6] para crear la aplicación web, pero investigando encontramos el framework Flask [3] que era más sencillo de utilizar y de instalar que Django, por este motivo nos decantemos por elegir Flask.

Además, Flask es un *framework* de Python que facilita la creación de servidores web de manera sencilla y rápida, y además favorece la implementación del patrón Model-View-Presenter.

HTML, CSS y BOOTSTRAP

Para el desarrollo del *front-end* se decidió utilizar HTML y CSS, pero como no se había aprendido CSS durante el grado, se pensó en utilizar el *framework* Bootstrap para que el modelado de la aplicación fuera más fácil y conciso.

Al final también se implementaron clases en CSS por nuestra cuenta, gracias la página web de CSS, <https://css.com/>, que muestra muchos ejemplos de cómo hacer diferentes clases en CSS.

Coordenadas Geográficas

En principio se pensó utilizar JavaScript para conocer la ubicación a tiempo real del usuario para proporcionar información relevante respecto su ubicación.

Posteriormente, se decidió utilizar la librería *Geopy* de Python para saber la ubicación a tiempo real del usuario. Se tomó esta decisión porque al principio del proyecto se desconocía esta librería y vimos que tenía una precisión mucho mayor que utilizando JavaScript.

5.4. Desarrollo de la App

Durante la primera fase del desarrollo del presente proyecto, se investigó sobre trabajos relacionados con nuestro proyecto y además se decidió que entorno de desarrollo utilizaríamos para nuestro proyecto.

Una vez realizado el trabajo de investigación respecto de los trabajos relacionados con nuestro proyecto y decidido nuestro entorno de desarrollo, comenzemos a realizar las siguientes tareas:

- Diseño del prototipo de la interfaz de la aplicación.
- Extraer los datos necesarios de las *APIs*
- Crear la base de datos en SQLite.
- Conectar la base de datos a nuestra aplicación.
- Conectar Flask con HTML

En cuanto a la persistencia de datos, se optó por utilizar SQLite. Se trata de una base de datos relacional fácil de configurar y de usar.

```
def conection_DB():
    conn = sqlite3.connect('DB.db')
    return (conn,conn.cursor())
```

Figura 5.2: Conexión a SQLite desde Python.

Para diseñar la estructura de la aplicación, se optó por utilizar el patrón de arquitectura Modelo-Vista-Presentador (MVP), que define cómo organizar y estructurar los diferentes componentes del sistema, las relaciones entre ellos y sus responsabilidades.

Para la obtención de la información meteorológica se utilizaron las APIs proporcionadas por *OpenWeatherMaps* y *TuTiempo*, las cuales nos permitían realizar 96 y 65 llamadas por minuto de manera gratuita.

Para la obtención de noticias se ha utilizado la API *NewsData.io*, que nos consentía realizar 200 solicitudes diarias de forma gratuita.

Para la obtención de eventos se utilizó la API *TicketMaster*, que nos consentía realizar 5000 peticiones diarias de forma gratuita.

Una vez que ya teníamos implementada la parte del *back-end* y las templates del front-end ya estaban integradas en nuestra aplicación, se comenzó a realizar varias funcionalidades de nuestra aplicación como la del registro o el login de usuarios, y algunas de las principales como la visualización de noticias, eventos y meteorología.

Para realizar las funcionalidades anteriores fue necesario aprender como realizar peticiones al servidor desde la interfaz de usuario, y a obtener el fichero JSON con la información solicitada a la *API* para que la información fuera accesible.

```
datosObtenidos = requests.get("https://newsdata.io/api/1/news?apikey=API_KEY&q=news&language=es&country=es")
datosFormatonJSON = datosObtenidos.json()
```

Figura 5.3: Solicitud de información al servicio NewsData.io

Además, se implementó la opción de poder filtrar tanto las noticias como los eventos por ubicación o categorías.

Con el fin de garantizar la disponibilidad constante de nuestra aplicación web para los usuarios, se tomó la decisión de guardar los datos obtenidos a través de las APIs. De esta manera, los servicios ofrecidos por nuestra aplicación estarán siempre accesibles, sin interrumpir la experiencia del usuario.

Se puso un gran esfuerzo en garantizar la facilidad de uso y la accesibilidad en el diseño de nuestra aplicación, esto se puede apreciar en nuestra aplicación como texto de tamaño legible, iconos para facilitar la compresión de la información, temas de contrastes altos. Se siguieron varias directrices de diseño en cuanto a estilos y tipos de componentes.

5.5. Documentación

Desde el primer momento, se optó por escribir la documentación de la memoria y anexos en *LaTeX*. Se tomó esta decisión porque *LaTeX* nos ofrece un resultado profesional, no tenemos que tener en cuenta de cómo queda el texto, se adapta a otros formatos perfectamente y fuerza al autor a estructurar sus textos. De esta forma podemos visualizar la información directamente estructurada. La única desventaja que hemos detectado sobre *LaTeX*, es que hay que compilar cada vez que se hace una modificación.

5.6. Integración Continua

Al principio del desarrollo del presente proyecto no se tuvo en cuenta la implementación de integración continua sobre el mismo, pero a medida que se fue desarrollando el proyecto se optó por implementar la integración continua con el fin detectar de manera rápida defectos y así poder generar un código de más calidad.

También se ha utilizado la integración continua en el desarrollo de este proyecto para desplegar automáticamente nuestra aplicación web en un

entorno de producción con el fin de trabajar de manera más eficiente y reducir los errores en el código existente.

Gracias a la integración continua podemos asegurar que el software esté siempre en un estado funcional y de calidad.

5.7. Dificultades en el desarrollo del proyecto

A lo largo del desarrollo del presente proyecto nos hemos encontrado con varias dificultades, las cuales se detallarán a continuación:

- La primera dificultad con la que nos encontramos durante el desarrollo fue como implementar la geolocalización del usuario, debido a que la herramienta que utilizamos para obtener la ubicación actual del usuario no tenía una precisión exacta. Tras esto, decidimos extraer la geolocalización del usuario a través de las cookies del navegador con el objetivo de obtener la ubicación actual del usuario con la mayor precisión posible. La resolución de esta dificultad nos ha llevado bastante tiempo de investigación, ya que nunca habíamos trabajado con las cookies del navegador.
- La segunda dificultad con la que nos encontramos fue como realizar las llamadas a las APIs para solicitar la información necesaria para cada uno de los servicios de nuestra aplicación. Esto nos ha llevado bastante tiempo de investigación, debido a que nunca se había trabajado con estos servicios y a su complejidad.
- La tercera dificultad con la que nos encontramos fue con la disponibilidad de las APIs debido a que sus servicios no funcionaban todos los días. Entonces, se optó por almacenar toda la información de cada uno de los servicios en distintos ficheros JSON con el fin de tener disponible siempre los servicios de noticias, eventos y meteorología en nuestra aplicación web. Toda la información contenida en los ficheros JSON se actualiza diariamente si es posible. De esta forma se pudo resolver dicho inconveniente.
- Por otra parte, ha resultado un poco complejo el desarrollo del front-end debido a que nunca se había trabajado en el desarrollo de la interfaz de usuario. Para poder llevar a cabo el desarrollo del Front-End nos ha llevado bastante tiempo de investigación, ya que por cada elemento que se añadía a la interfaz de usuario había que darle estilo.

5.8. Publicación de la App

Una vez que la aplicación estuvo completamente desarrollada y lista para ser utilizada por los usuarios finales, se la publicó en un servicio de alojamiento (hosting).

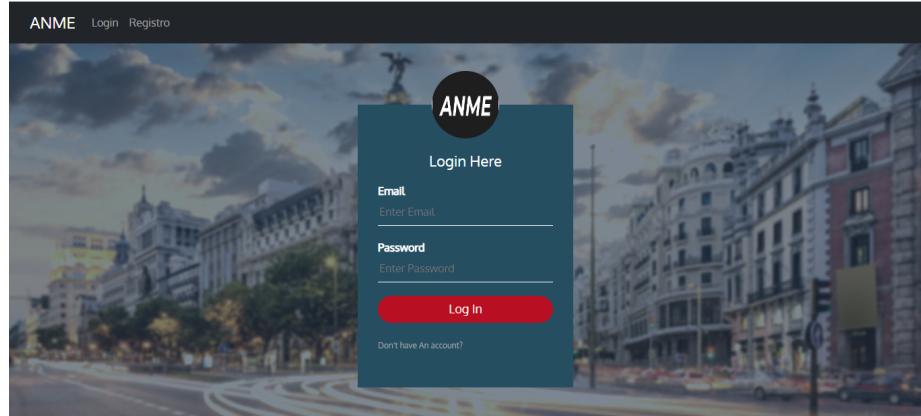


Figura 5.4: Aplicación ANME

Podemos acceder a la aplicación a través del siguiente enlace: <https://www.anme.city>

También podemos acceder al repositorio del código del presente proyecto a través del siguiente enlace: https://github.com/jmri1001/PROYECTO_TFG.git

Trabajos relacionados

En este apartado vamos a comentar trabajos similares al realizado en este proyecto.

6.1. Proyectos

Windyty

Windyty es una herramienta web en el campo de la meteorología. Se trata de una aplicación web basada en el proyecto Earth que aporta ciertas novedades que consideramos útiles para los profesionales de esta apasionada ciencia. Sus funcionalidades principales son proporcionar previsiones meteorológicas, reportes METAR y poder compartir sus mapas de meteorología.

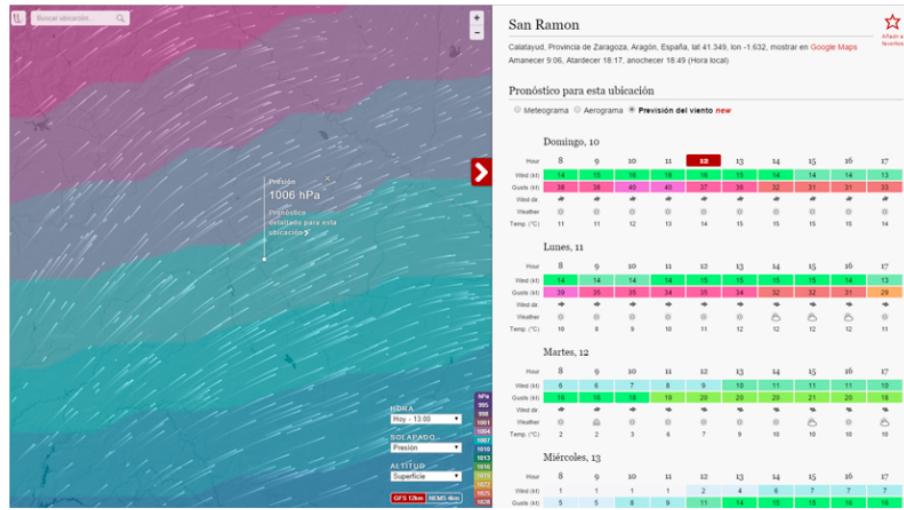


Figura 6.1: Previsión meteorológica de cuatro días

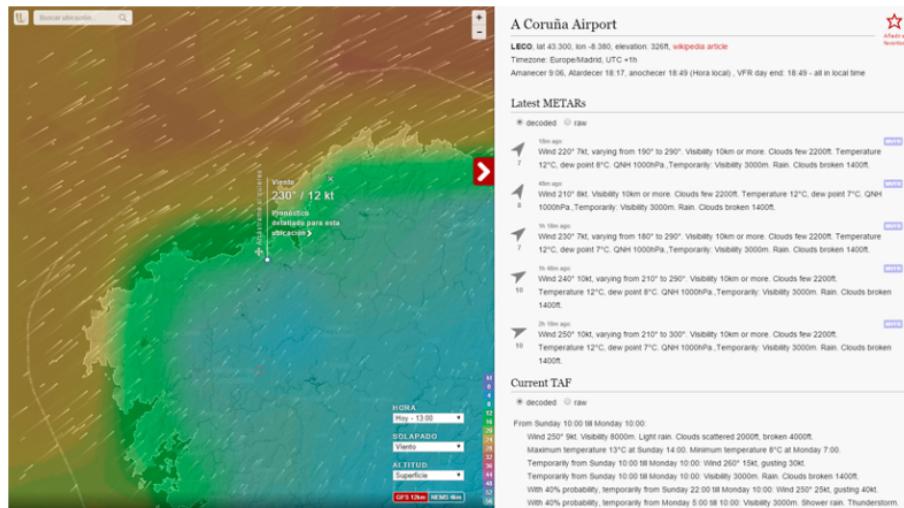


Figura 6.2: Reporte Metar

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://cazatormentas.net/windty-una-completa-aplicacion-web-para-la-meteorologia/>

Meteo Consult

Meteo Consult es una aplicación web que ofrece información detallada del clima de los próximos 15 días. Esta aplicación permite su uso a nivel

mundial. Dentro de la información que nos brinda esta app podemos destacar mapas de vientos, de oleajes y zonas costeras. Además, esta aplicación se encuentra disponible para Android e iOS



Figura 6.3: Previsión meteorológica

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://www.meteoconsult.es/clima-espana/previsiones-tiempo-espana-hoy>

TicketMaster

TicketMaster es una aplicación web dedicada a la gestión de venta de entradas para los eventos más importantes a nivel mundial. Permite consultar eventos de diferentes categorías como por ejemplo eventos de música, de teatro, de deportes, de shows, de magia, etc.

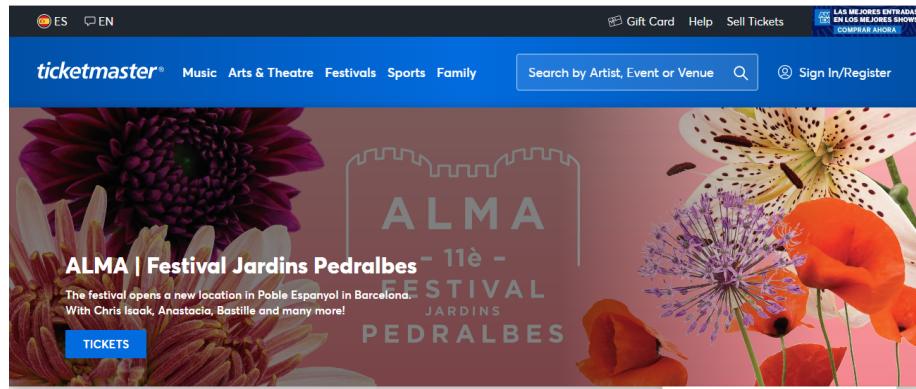


Figura 6.4: App TicketMaster

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://www.ticketmaster.es/>

Newsflow

Newsflow es una aplicación que reúne todas las noticias de sus sitios web favoritos en un solo lugar. Esta app descarga las noticias en nuestro dispositivo y las guarda de manera local, para que podamos leerlas en cualquier momento que deseemos.

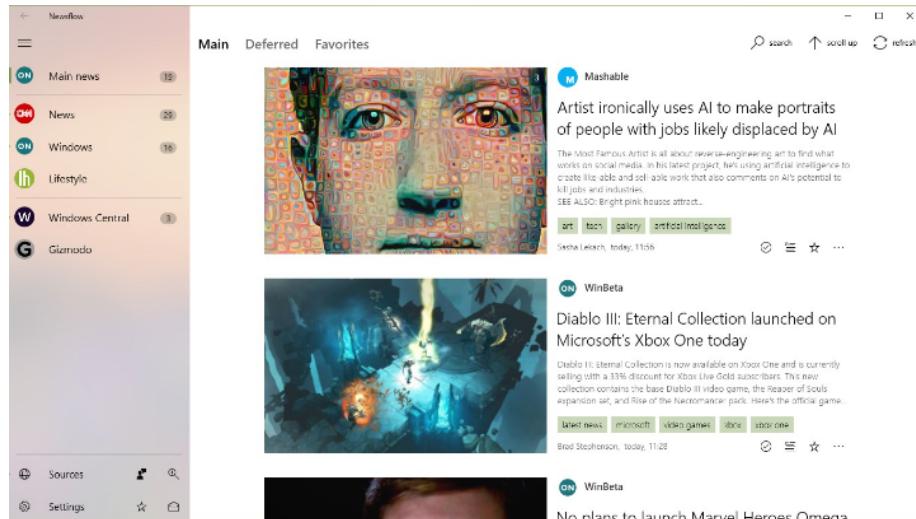


Figura 6.5: App Newsflow

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://apps.microsoft.com/store/detail/newsflow/9NBLGGH58S5R?hl=es-es&gl=es&rtc=1>

Microsoft Bing

Microsoft Bing es un navegador de búsqueda en línea que ofrece una variedad de servicios de búsqueda, incluyendo noticias, meteorología, imágenes, videos, etc. Bing ofrece una experiencia de búsqueda personalizada e intuitiva para los usuarios.

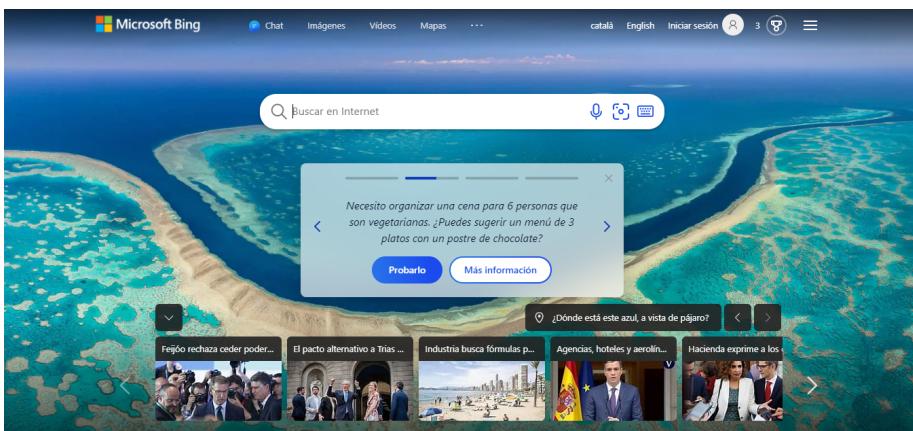


Figura 6.6: Microsoft Bing

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://www.bing.com/?setlang=es>

Google News

Google News es un servicio de noticias en línea proporcionado por Google que recopila y organiza noticias de diversas fuentes en todo el mundo. Proporciona a los usuarios acceso a una variedad de noticias, les permite personalizar su experiencia y les ayuda a estar informados sobre los eventos actuales y los temas de su interés.

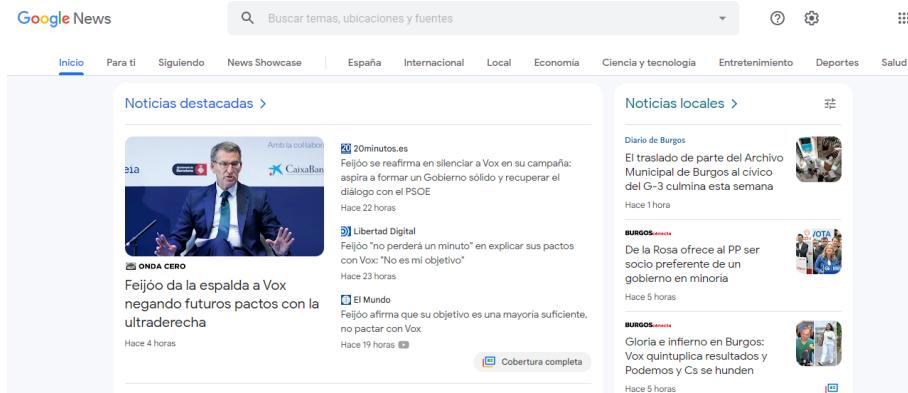


Figura 6.7: Google News

- Puedes consultar más información sobre este proyecto desde el siguiente enlace: <https://news.google.com/home?hl=es&gl=ES&ceid=ES:es>

Conclusiones y Líneas de trabajo futuras

7.1. Conclusiones

Después de llevar a cabo el desarrollo del proyecto, podemos obtener las siguientes conclusiones:

- Se ha logrado de manera exitosa el objetivo del proyecto. Ahora los usuarios disponen de una aplicación web que les permite estar informados tanto de noticias, como eventos y meteorología, sin necesidad de realizar búsquedas en el navegador.
- El proyecto ha englobado la mayoría de conocimientos adquiridos durante la etapa del grado. Además, durante el desarrollo del proyecto se ha adquirido una gran cantidad de conocimientos acerca de las aplicaciones web, los protocolos empleados y las solicitudes realizadas a través de los mismos.
- A lo largo del proyecto, se han utilizado muchas herramientas y tecnologías, las cuales en su mayoría han desempeñado un papel fundamental en la mejora de la calidad de nuestra aplicación.
- Gracias a la implementación de integración continua, se ha logrado detectar de manera rápida defectos en el código, lo que ha permitido generar un código de más calidad.
- Tras haber utilizado la metodología SCRUM nos ha permitido adquirir habilidades para estimar de manera precisa el tiempo dedicado a cada tarea del desarrollo.

7.2. Líneas de trabajo futuras

- Añadir la funcionalidad de iniciar sesión utilizando cuentas de GitHub, Google, entre otras.
- Implementar la funcionalidad de mostrar el tráfico de una ubicación de interés para el usuario.
- Se desea mejorar la seguridad de nuestra aplicación, para evitar robos de datos.
- Tener disponible nuestra aplicación web en el idioma Inglés.

Bibliografía

- [1] MDN contributors. *CSS developer guide*. Mozilla Developer Network, 2016.
- [2] Eric Godoc. *SQL: Los fundamentos del lenguaje*. Ediciones ENI, 2014.
- [3] Miguel Grinberg. *Flask web development: developing web applications with python*. .º'Reilly Media, Inc.", 2018.
- [4] Hostinger. Servidor web. <https://www.hostinger.es/tutoriales/que-es-un-servidor-web>, April 2020.
- [5] Carlos Andrés Morales Machuca. Estado del arte: Servicios web. *Universidad Nacional de Colombia, Tesis de Maestría*, 2010.
- [6] Openwebinars. Django. <https://openwebinars.net/blog/django-vs-flask/>, 2020.
- [7] Ryan Paul. *Consultas SQL*. Ars Technica, 2005.
- [8] AA Prayogi, M Niswar, M Rijal, et al. Design and implementation of rest api for academic information system. In *IOP Conference Series: Materials Science and Engineering*, page 012047. IOP Publishing, 2020.
- [9] Joaquín Medina Serrano. Modelo–vista–presentador. http://joaquin.medina.name/web2008/documentos/informatica/documentacion/logica/patrones/patronMVP/2012_06_09_PatronMVP.html, November 2013.
- [10] Wikipedia. Base de datos — wikipedia, la enciclopedia libre, 2016.

- [11] Wikipedia. Protocolo de transferencia de hipertexto — wikipedia, la enciclopedia libre, 2020. [Internet; descargado 10-abril-2020].
- [12] Wikipedia. Api — wikipedia, la enciclopedia libre, 2023. [Internet; descargado 25-mayo-2023].