



Ricardo Fernández Guzmán

Curso 2023/24

2º DAW

Índice

Índice	1
Identificación de las necesidades del proyecto	2
Breve análisis/comparativa con las alternativas del mercado	3
Trello	3
Asana	3
Comparativa con Blue Bull-Task Manager	4
Justificación del proyecto	4
Uso de stack tecnológico. Justificación del mismo	6
Frontend: Angular	6
Backend: Spring	6
Estilo: CSS y Librería de Bootstrap	6
Base de Datos: MySQL	6
Autenticación: JSON Web Tokens (JWT)	6
Herramientas de Desarrollo: Git y GitHub	6
Herramientas para el Despliegue: AWS	6
Esquema E-R y descripción de las entidades y campos de la base de datos	7
Prototipo de la Aplicación Web	8
Definición API REST publicación servicios	13
AuthController	14
UsuarioController	14
AnotacionController	16
ProyectoController	17
TareaController	18
Manual de Despliegue	19
En AWS	19
En local	23
Postmortem y conclusiones del proyecto.	25
Anexo	29

Introducción

En la actualidad, la gestión eficiente de tareas y proyectos es esencial tanto para individuos como para equipos que buscan maximizar su productividad y organización. Sin embargo, muchas herramientas disponibles en el mercado no se ajustan completamente a las necesidades específicas de todos los usuarios. Motivado por esta carencia, nace Blue Bull - Task Manager, una aplicación web diseñada para proporcionar una solución integral a la gestión de tareas y proyectos.

Identificación de las necesidades del proyecto

Requisitos Funcionales: En este apartado iré poniendo los requisitos funcionales que quiero implementar en el proyecto

- **RF01** - Sistema de autenticación y login: Permitir loguearse a los usuarios y redirección según el rol
- **RF02** - Registro de Usuarios: Permitir un registro para los usuarios
- **RF03** - Gestión de Perfiles de Usuario: Los usuarios podrán editar y actualizar sus datos.
- **RF04** - Gestión de Usuarios por parte del Administrador: Operaciones CRUD sobre los perfiles de usuario.
- **RF05** - Creación de Anotaciones: Los usuarios podrán crear anotaciones, similar a Post-it
- **RF06** - Creación de Tareas: Los usuarios podrán crear tareas para un proyecto
- **RF07** - Creación de Proyectos: Los usuarios podrán crear proyectos desde cero, incluyendo tareas iniciales y los usuarios que pertenezcan a este proyecto
- **RF08** - Gestión de Anotaciones, Tareas y Proyectos por parte del Administrador: Operaciones CRUD llevadas a cabo por el usuario Administrador sobre todas estas entidades
- **RF09** - Eliminación de las Tareas y Proyectos por parte del usuario
- **RF10** - Gestión del proyecto y de la organización de las tareas de estos: Los usuarios podrán reorganizar de manera intuitiva el estado de las tareas del proyecto

Breve análisis/comparativa con las alternativas del mercado

En el mercado actual existen diversas herramientas diseñadas para la gestión de tareas y proyectos, cada una con sus puntos fuertes y débiles. A continuación, se presenta una comparativa entre Blue Bull - Task Manager y algunas de las aplicaciones más populares, como Trello o Asana

Trello

Puntos Fuertes:

- **Interfaz Intuitiva:** Basada en tableros y tarjetas, lo que facilita la visualización y organización de tareas.
- **Flexibilidad:** Permite una amplia personalización mediante Power-Ups (integraciones) y etiquetas.
- **Colaboración:** Fácil de usar en equipos pequeños y medianos, con funcionalidades de comentarios y menciones.

Puntos Débiles:

- **Funcionalidades Limitadas en la Versión Gratuita:** Algunas características avanzadas requieren suscripción.
- **Escalabilidad:** Puede resultar menos eficiente en la gestión de proyectos muy grandes o complejos.
- **Búsqueda y Filtrado:** Aunque útil, podría mejorarse para manejar grandes volúmenes de datos.

Asana

Puntos Fuertes:

- **Gestión de Proyectos:** Ofrece vistas variadas (lista, tablero, calendario) y funcionalidades avanzadas de seguimiento de progreso.
- **Integraciones:** Se integra bien con otras herramientas y servicios.
- **Automatizaciones:** Permite automatizar flujos de trabajo para mejorar la eficiencia.

Puntos Débiles:

- **Curva de Aprendizaje:** Puede ser complejo para nuevos usuarios.
- **Costo:** Las características más útiles están en los planes de pago.
- **Sobrecarga de Funcionalidades:** Puede ser excesivo para proyectos pequeños y usuarios individuales.

Comparativa con Blue Bull-Task Manager

Puntos Fuertes de AR-Task:

- **Interfaz Intuitiva:** Diseñada para ser fácil de usar tanto para individuos como para equipos.
- **Fácil de utilizar:** No requiere una alta curva de aprendizaje para desempeñar todas sus funcionalidades
- **Gestión Integral:** Combina funcionalidades de gestión de anotaciones y proyectos en una sola plataforma

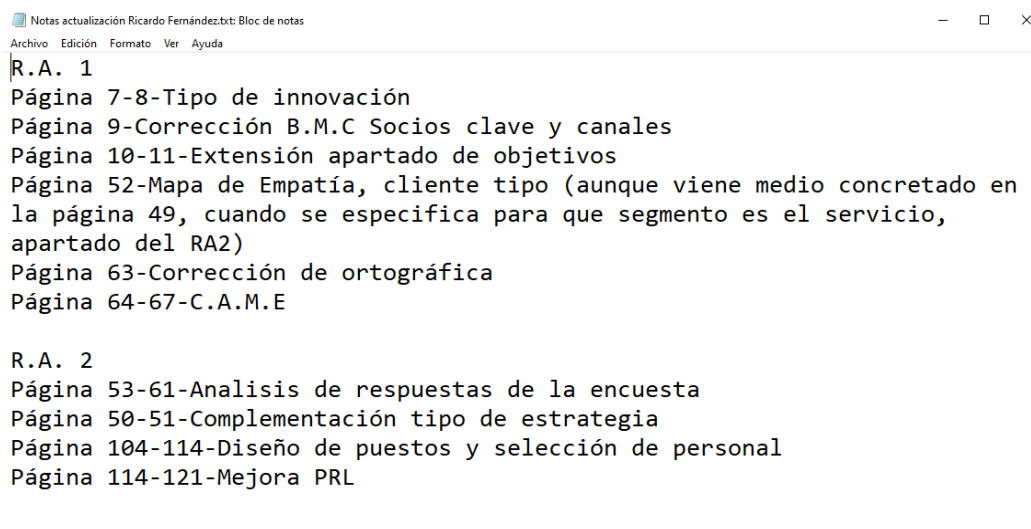
Puntos Débiles de AR-Task:

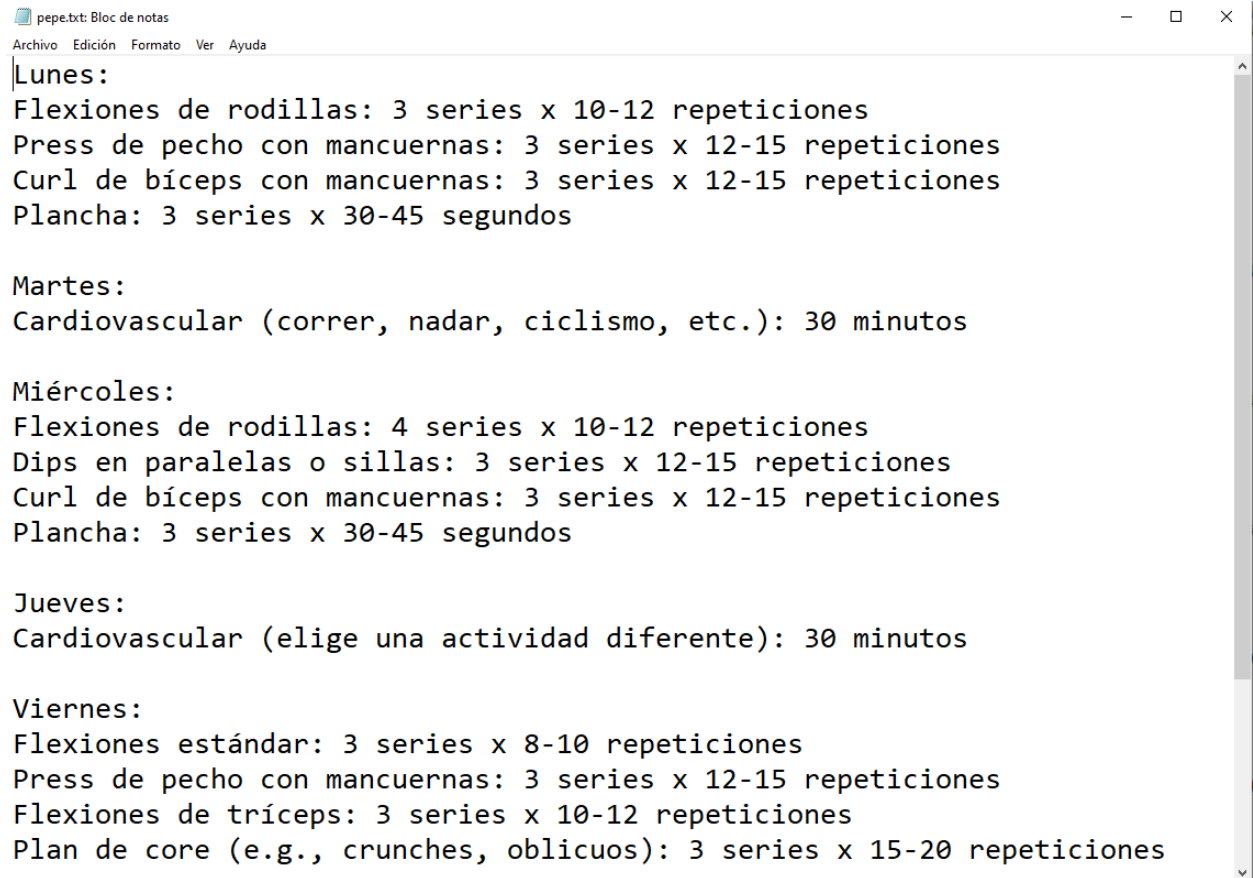
- **Desarrollo en Proceso:** Algunas funcionalidades aún están en desarrollo y no se ha determinado completamente
- **Recursos y Comunidad:** Al ser una aplicación emergente, no cuenta con la misma base de usuarios y recursos que herramientas consolidadas como Trello o Asana.

Justificación del proyecto

Como estudiante interesado por la informática y el desarrollo de software, me encontré enfrentando un desafío recurrente en mi vida académica y sobre todo a la hora de enfrentar trabajos o llevar simplemente un listado de todo lo que tenía que hacer para x día o simplemente para llevar organizado todo.

Puede que sea el único que no utilice herramientas de este tipo ya, ya sea con google o porque simplemente no me siento cómodo usando ninguna de estas, termino utilizando el bloc de notas como listado de todas las cosas que voy haciendo o me quedan por hacer.





Dejando a mi paso, un reguero de archivos de texto desperdigados por mi ordenador. Motivado por esta situación es que empecé a maquinar en mi cabeza la idea de Blue Bull-Task Manager . Una aplicación web que integre todas estas características.



Uso de stack tecnológico. Justificación del mismo

Frontend: Angular

Angular se elige para el frontend debido a su capacidad de crear aplicaciones escalables y modulares, con componentes reutilizables que facilitan el desarrollo y mantenimiento.

Backend: Spring

Spring se utiliza en el backend por su modularidad y flexibilidad, permitiendo construir aplicaciones adaptadas a necesidades específicas. Su integración con Spring Security proporciona autenticación y autorización seguras, mientras que Spring Boot facilita la creación de aplicaciones Java de alto rendimiento con mínima configuración.

Estilo: CSS y Librería de Bootstrap

CSS, junto con la librería Bootstrap, se emplea para el estilo por su control total sobre el diseño visual y capacidad de crear interfaces responsivas que se adaptan a diferentes tamaños de pantalla. Bootstrap acelera el desarrollo al ofrecer un sistema de grid y componentes predefinidos, asegurando consistencia y eficiencia en el diseño.

Base de Datos: MySQL

MySQL es seleccionado como la base de datos por su popularidad, rendimiento y confiabilidad, capaz de manejar grandes volúmenes de datos y operaciones complejas.

Autenticación: JSON Web Tokens (JWT)

JWT se usa para la autenticación debido a su seguridad y estandarización, permitiendo una gestión de sesiones sin estado que facilita la escalabilidad. Los tokens son portables y pueden ser utilizados en cualquier cliente, mejorando la eficiencia al no necesitar almacenamiento de sesiones en el servidor.

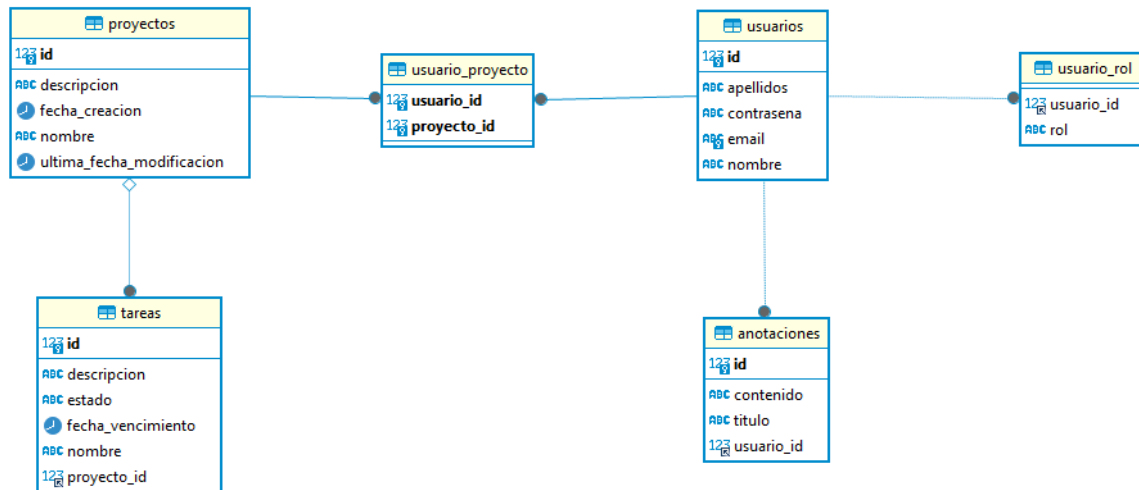
Herramientas de Desarrollo: Git y GitHub

Git y GitHub se eligen para el control de versiones y la colaboración, proporcionando herramientas eficientes para la gestión de cambios y revisión de código. GitHub facilita la colaboración entre desarrolladores, integrándose bien con otras herramientas de desarrollo y despliegue, mejorando la productividad y calidad del código.

Herramientas para el Despliegue: AWS

AWS (Amazon Web Services) se utiliza para el despliegue debido a su robusta infraestructura en la nube y a su amplia gama de servicios que facilitan la gestión de aplicaciones a gran escala. AWS permite un despliegue eficiente y seguro de la aplicación, asegurando consistencia y confiabilidad en diferentes entornos.

Esquema E-R y descripción de las entidades y campos de la base de datos



Usuarios: Se trata de los usuarios que conforman la aplicación, todos los registrados en ella

id: Identificador único del usuario (clave primaria).

apellidos: Apellidos del usuario.

contrasena: Contraseña del usuario.

email: Correo electrónico del usuario.

nombre: Nombre del usuario.

Usuario_rol: Tabla en la que se designa el rol que cada usuario tiene

usuario_id: Identificador del usuario (clave foránea).

rol: Rol asignado al usuario. Puede variar entre dos estados; USUARIO y ADMINISTRADOR

Proyectos: Recoge todos los proyectos de la aplicación

id: Identificador único del proyecto (clave primaria).

descripcion: Descripción del proyecto.

fecha_creacion: Fecha de creación del proyecto.

nombre: Nombre del proyecto.

ultima_fecha_modificacion: Última fecha en que se modificó el proyecto.

Tareas: En este campo se encuentran todas las tareas generadas para todos los proyectos

id: Identificador único de la tarea (clave primaria).

descripcion: Descripción de la tarea.

estado: Estado actual de la tarea. Esta puede tener 3 estados, PENDIENTE, EN_PROCESO y COMPLETADA

fecha_vencimiento: Fecha de vencimiento de la tarea.

nombre: Nombre de la tarea.

proyecto_id: Identificador del proyecto al que pertenece la tarea (clave foránea).

Anotaciones: En este campo se encuentran todas las anotaciones creadas por los usuarios

id: Identificador único de la anotación (clave primaria).

contenido: Contenido de la anotación.

titulo: Título de la anotación.

usuario_id: Identificador del usuario que creó la anotación (clave foránea).

Usuario_proyecto: Tabla creada por la relaciono N:M entre la tabla proyectos y usuarios

usuario_id: Identificador del usuario (clave foránea).

proyecto_id: Identificador del proyecto (clave foránea).

Las relaciones entre las entidades son las siguientes:

- Un proyecto puede tener múltiples tareas, sin embargo una tarea solo puede estar en un proyecto
- Un usuario puede estar relacionado con múltiples proyectos a través de la tabla usuario_proyecto y viceversa, en un proyecto pueden intervenir múltiples usuarios.
- Un usuario puede tener múltiples roles a través de la tabla usuario_rol.
- Un usuario puede crear múltiples anotaciones, pero una misma anotación sólo puede estar asociada a un usuario.

Prototipo de la Aplicación Web

Para ver con mayor atención el prototipo de la Aplicación Web se puede mediante el siguiente enlace a Figma, adicionalmente se encuentra un pdf con las pantallas del prototipo en el GitHub de este proyecto, en “doc/prototipoFigma”

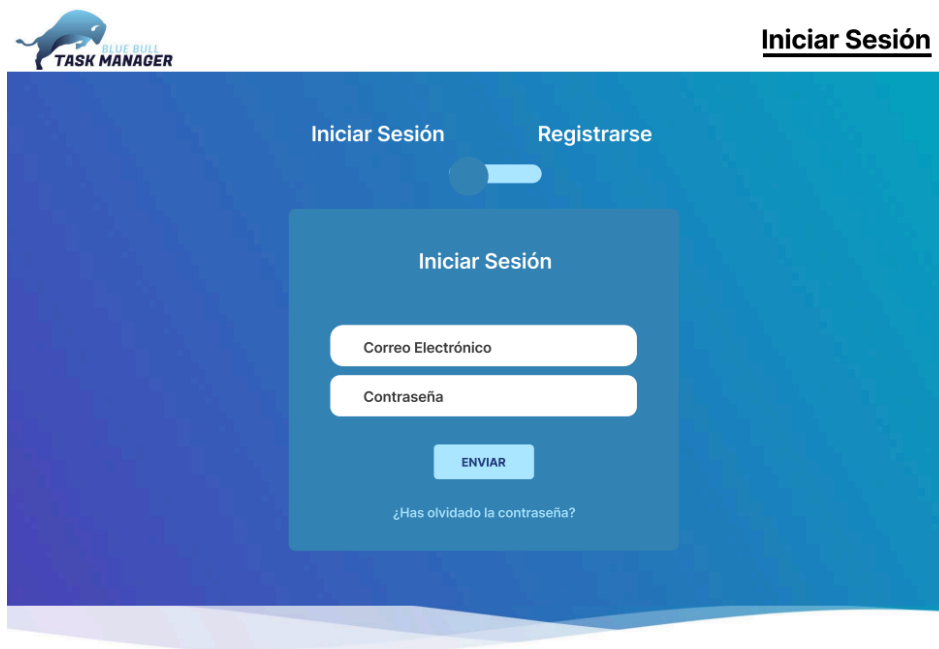
(<https://github.com/RicardoFgX/Proyecto-Final/tree/main/docs/prototipoFigma>). Enlace al figma:

<https://www.figma.com/design/0pT3RIh9RoYocCwsvtkTLE/Untitled?node-id=0-1&t=yFs9hJkM2Qt74EBs-1>

La aplicación cuenta con una primera pantalla, que es el **home**, en donde se habilita la posibilidad de iniciar Sesión para poder ver el resto de contenidos de la misma



Ventana de Inicio de Sesión



Ventana de registro de usuario

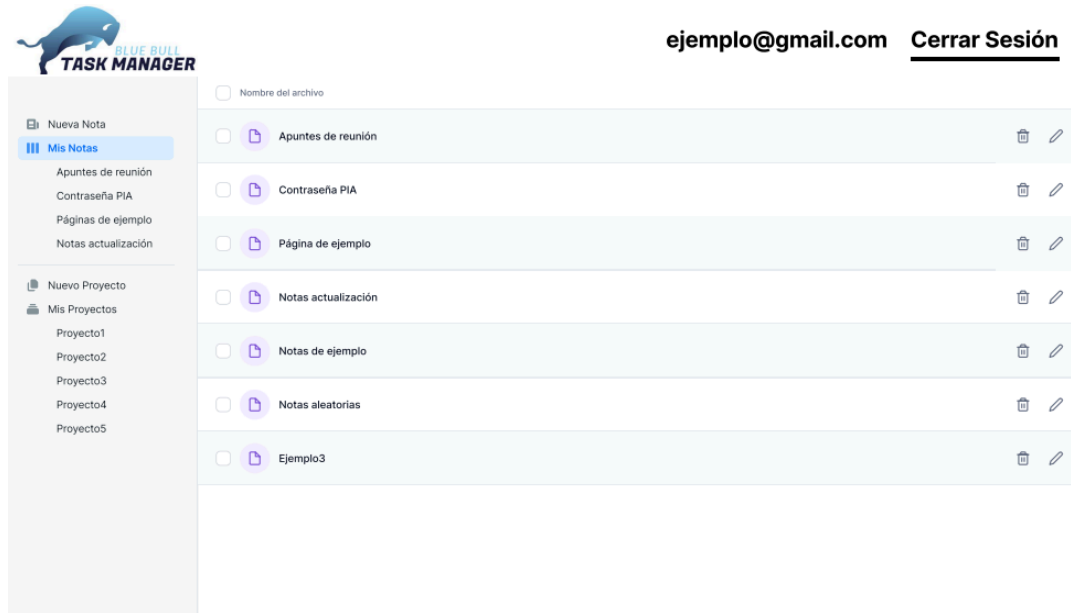
The screenshot shows the 'Iniciar Sesión' (Login) page of the 'BLUE BULL TASK MANAGER' application. At the top right, there is a link to 'Iniciar Sesión'. Below it, a toggle switch is set to 'Registrarse' (Register). A central modal box titled 'Registrarse' contains three input fields: 'Nombre' (Name), 'Correo Electrónico' (Email), and 'Contraseña' (Password). Below these fields is a blue button labeled 'ENVIAR' (Send).

El **panel de control del Administrador** posee una barra lateral donde poder navegar entre los usuarios, notas (Ver Anexo) y proyectos (Ver Anexo) para gestionarlos. Para las operaciones sensibles incluye un modal para advertir de la acción que está a punto de realizar (Ver Anexo).

The screenshot shows the 'Admin View' of the 'BLUE BULL TASK MANAGER' application. The top right corner displays the email 'ejemplo@gmail.com' and a 'Cerrar Sesión' (Logout) link. The left sidebar contains navigation links: 'Admin View' (Gestiona usuarios y proyectos), 'Usuarios' (selected), 'Apuntes', and 'Proyectos'. Below these are four links for 'Apartados aún por determinar'. The main content area displays a table of users.

<input type="checkbox"/>	Nombre	Rol	Correo Electrónico	Contraseña	
<input type="checkbox"/>	Admin ejemplo	Usuario Admin	admin@gmail.com	admin	
<input type="checkbox"/>	Prueba admin2	Usuario Admin	admin2@gmail.com	admin2	
<input type="checkbox"/>	Pedro Jiménez Salas	Usuario Estándar	prueba@gmail.com	prueba	
<input type="checkbox"/>	Ricardo Fernández Guzmán	Usuario Estándar	Lucas@gmail.com	ricardo123	
<input type="checkbox"/>	Pablo Sierra Montiel	Usuario Estándar	pablo1@gmail.com	pablo1304	


La **vista principal del usuario “normal”**, similar a la del administrador en cuanto estética, además de que solo te permite ver tus proyectos y notas, cuenta con una desplegable en la barra lateral para ver los proyectos y notas



Ventana de Creación/Modificación de Notas

The screenshot shows the "Ventana de Creación/Modificación de Notas" (Note Creation/Modification Window). The interface is similar to the main view, with the "Mis Notas" section expanded in the sidebar. The main content area has a form for creating or editing a note. It includes a "Nombre de la nota" (Note Name) field with the placeholder text "Nombre". Below this is a "Contenido de la nota" (Note Content) field with the placeholder text "Introduzca un texto". The "Cerrar Sesión" link is still visible at the top right.

Ventana de un proyecto: Cuenta con un menú que permite arrastrar tareas en los 3 posibles estados



Nueva Nota

Mis Notas

Apuntes de reunión

Contraseña PIA

Páginas de ejemplo

Notas actualización

Nuevo Proyecto

Mis Proyectos

Proyecto XYZ

Proyecto2

Proyecto3

Proyecto4

Proyecto5

ejemplo@gmail.com

Cerrar Sesión

Proyecto XYZ

PENDIENTE

Entregable III

Entregable IV

Entregable V

Entregable VI

+

EN PROGRESO

• Entregable II

a. Proceso de autenticación / Dashboard inicial de administrador (en caso de existir) y de usuario

b. Diseño del prototipado de la interfaz empleando alguna herramienta como Figma.

+

COMPLETADA

• Entregable I

a. Diseño cerrado de la base de datos (se deberán ir mandando las versiones a través del repo)

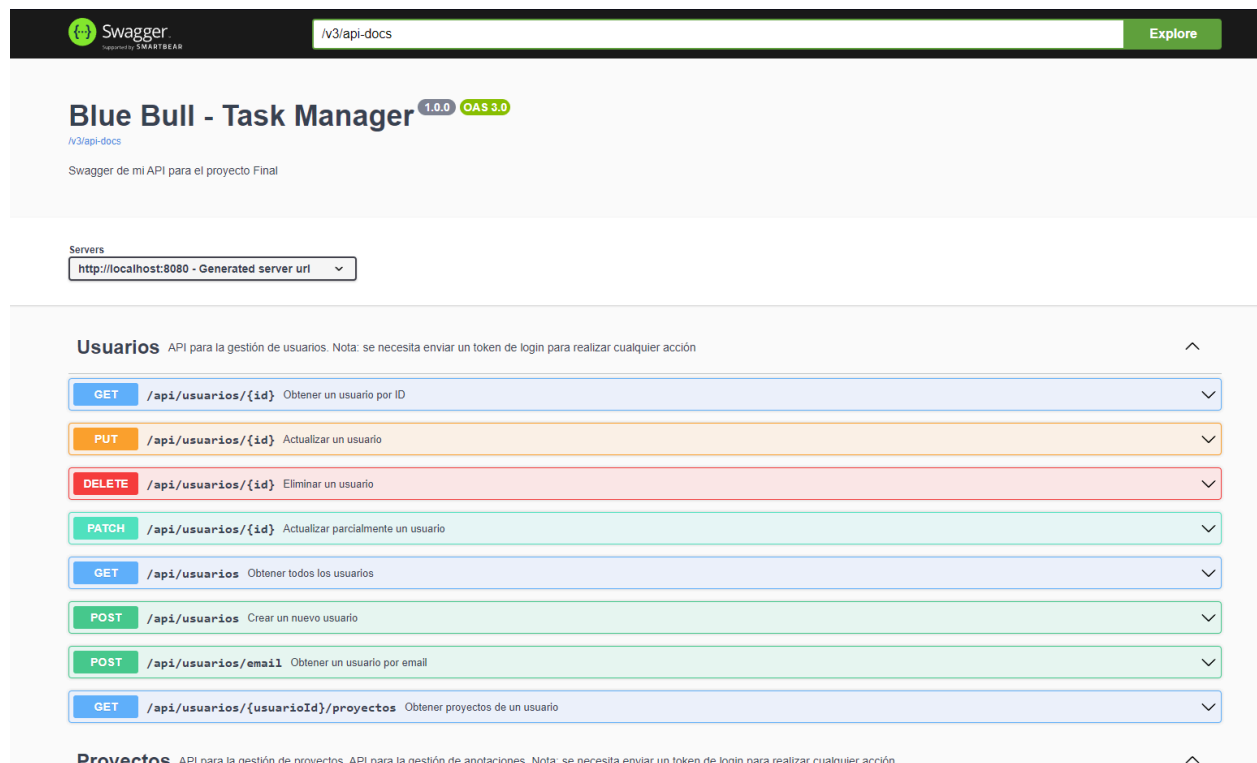
b. Descripción del proyecto, su alcance y el stack tecnológico elegido detallado para que podamos ir probando los avances. Todos estos puntos deben ocupar 2 páginas como mínimo.

+

Definición API REST publicación servicios

Se ha utilizado tanto Swagger-UI como Postman para llevar a cabo esto. Para lo primero, este se genera al desplegar el backend con la siguiente URL:

<http://localhost:8080/swagger-ui/index.html> o si está con la version desplegada en AWS como <http://ec2-44-204-237-104.compute-1.amazonaws.com:8080/swagger-ui/index.html> y se ve de la siguiente forma:



Para ver desde el Postman todos los métodos, el fichero se encuentra en la carpeta “doc/postman” en el GitHub de este proyecto.

(<https://github.com/RicardoFgX/Proyecto-Final/tree/main/docs/postman>)

A continuación se exponen de igual forma las tablas con los Endpoints de la API

AuthController

Endpoint	Tipo	Parámetros	Descripción
/api/auth/signup	POST	RegistroRequest (objeto JSON)	Registra un nuevo usuario en el sistema. Devuelve un token JWT.
/api/auth/signin	POST	LoginRequest (objeto JSON)	Inicia sesión para un usuario existente. Devuelve un token JWT.

Detalles de los parámetros

❖ RegistroRequest

- ❖ **nombre(opcional):** El nombre del usuario (string).
- ❖ **email:** La dirección de correo electrónico del usuario (string).
- ❖ **contrasena:** La contraseña del usuario (string).

❖ LoginRequest

- ❖ **email:** La dirección de correo electrónico del usuario (string).
- ❖ **contrasena:** La contraseña del usuario (string).

UsuarioController

Endpoint	Tipo	Parámetros	Descripción
/api/usuarios	GET	Ninguno	Obtiene la lista de todos los usuarios.
/api/usuarios/{id}	GET	id (Long): ID del usuario	Obtiene los detalles de un usuario específico por su ID.
/api/usuarios/email	POST	EmailRequest (objeto JSON)	Obtiene los detalles de un usuario específico por su email.

/api/usuarios	POST	Usuario (objeto JSON)	Crea un nuevo usuario con la información proporcionada.
/api/usuarios/{id}	PUT	id (Long): ID del usuario, Usuario (objeto JSON)	Actualiza la información de un usuario específico.
/api/usuarios/{id}	PATCH	id (Long): ID del usuario, Usuario (objeto JSON)	Actualiza parcialmente la información de un usuario específico.
/api/usuarios/{id}	DELETE	id (Long): ID del usuario	Elimina un usuario específico por su ID.
/api/usuarios/{usuarioid}/proyectos	GET	usuarioid (Long): ID del usuario	Obtiene la lista de todos los proyectos de un usuario específico por su ID.

Detalles de los parámetros

❖ EmailRequest

- ❖ **email:** La dirección de correo electrónico del usuario (string).

❖ Usuario

- ❖ **id:** El identificador único del usuario (Long).
- ❖ **nombre(opcional):** El nombre del usuario (string).
- ❖ **apellidos(opcional):** Los apellidos del usuario (string).
- ❖ **email:** La dirección de correo electrónico del usuario (string).
- ❖ **contrasena:** La contraseña del usuario (string).
- ❖ **rol(opcional, por defecto es "USUARIO"):** El rol del usuario (Set<Rol>), admite solo "USUARIO" y "ADMINISTRADOR"
- ❖ **proyectos(opcional):** Los proyectos asociados al usuario (Set<Proyecto>).
- ❖ **anotaciones(opcional):** Las anotaciones del usuario (Set<Anotacion>).

AnotacionController

Endpoint	Tipo	Parámetros	Descripción
/api/ anotaciones	GET	Ninguno	Obtiene la lista de todas las anotaciones.
/api/ anotaciones/{id}	GET	id (Long): ID de la anotación	Obtiene los detalles de una anotación específica por su ID.
/api/ anotaciones/usuario/{usuarioid}	GET	usuarioid (Long): ID del usuario	Obtiene la lista de todas las anotaciones de un usuario específico por su ID.
/api/ anotaciones	POST	Anotacion (objeto JSON)	Crea una nueva anotación con la información proporcionada.
/api/ anotaciones/{id}	PUT	id (Long): ID de la anotación, Anotacion (objeto JSON)	Actualiza la información de una anotación específica.
/api/ anotaciones/{id}	DELETE	id (Long): ID de la anotación	Elimina una anotación específica por su ID.

Detalles de los parámetros

❖ Anotación

- ❖ **id**: El identificador único de la anotación (Long).
- ❖ **titulo(opcional)**: El título de la anotación (string).
- ❖ **contenido(opcional)**: El contenido de la anotación (string).
- ❖ **usuario(Ver parámetros de UsuarioController para más información)**: El usuario asociado a la anotación (Usuario).

ProyectoController

Endpoint	Tipo	Parámetros	Descripción
/api/proyectos	GET	Ninguno	Obtiene la lista de todos los proyectos.
/api/proyectos/{id}	GET	id (Long): ID del proyecto	Obtiene los detalles de un proyecto específico por su ID.
/api/proyectos	POST	Proyecto (objeto JSON)	Crea un nuevo proyecto con la información proporcionada.
/api/proyectos/{id}	PUT	id (Long): ID del proyecto, Proyecto (objeto JSON)	Actualiza la información de un proyecto específico.
/api/proyectos/{id}	PATCH	id (Long): ID del proyecto, Proyecto (objeto JSON)	Actualiza parcialmente la información de un proyecto específico.
/api/proyectos/{id}	DELETE	id (Long): ID del proyecto	Elimina un proyecto específico por su ID.

Detalles de los parámetros

❖ Proyecto

- ❖ **id**: El identificador único del proyecto (Long).
- ❖ **nombre(opcional)**: El nombre del proyecto (string).
- ❖ **descripcion(opcional)**: La descripción del proyecto (string).
- ❖ **fechaCreacion(opcional)**: La fecha de creación del proyecto (LocalDate).
- ❖ **ultimaFechaModificacion(opcional)**: La última fecha de modificación del proyecto (LocalDate).
- ❖ **usuarios(Ver parámetros de UsuarioController para más información)**: Los usuarios asociados al proyecto (Set<Usuario>).
- ❖ **tareas(Ver parámetros de TareaController para más información)**: Las tareas asociadas al proyecto (Set<Tarea>).

TareaController

Endpoint	Tipo	Parámetros	Descripción
/api/tareas	GET	Ninguno	Obtiene la lista de todas las tareas.
/api/tareas/{id}	GET	id (Long): ID de la tarea	Obtiene los detalles de una tarea específica por su ID.
/api/tareas	POST	Tarea (objeto JSON)	Crea una nueva tarea con la información proporcionada.
/api/tareas/{id}	PUT	id (Long): ID de la tarea, Tarea (objeto JSON)	Actualiza la información de una tarea específica.
/api/tareas/{id}	DELETE	id (Long): ID de la tarea	Elimina una tarea específica por su ID.
/api/tareas/proyecto/{proyectoid}	GET	proyectoid (Long): ID del proyecto	Obtiene la lista de todas las tareas de un proyecto específico por su ID.

Detalles de los parámetros

❖ Proyecto

- ❖ **id:** El identificador único de la tarea (Long).
- ❖ **nombre(opcional):** El nombre de la tarea (string).
- ❖ **descripcion(opcional):** La descripción de la tarea (string).
- ❖ **fechaVencimiento:** La fecha de vencimiento de la tarea (LocalDate).
- ❖ **estado:** El estado de la tarea (EstadoTarea). Puede variar entre "PENDIENTE", "COMPLETADO" y "EN_PROCESO"
- ❖ **proyecto:** El proyecto asociado a la tarea (Proyecto).

Manual de Despliegue

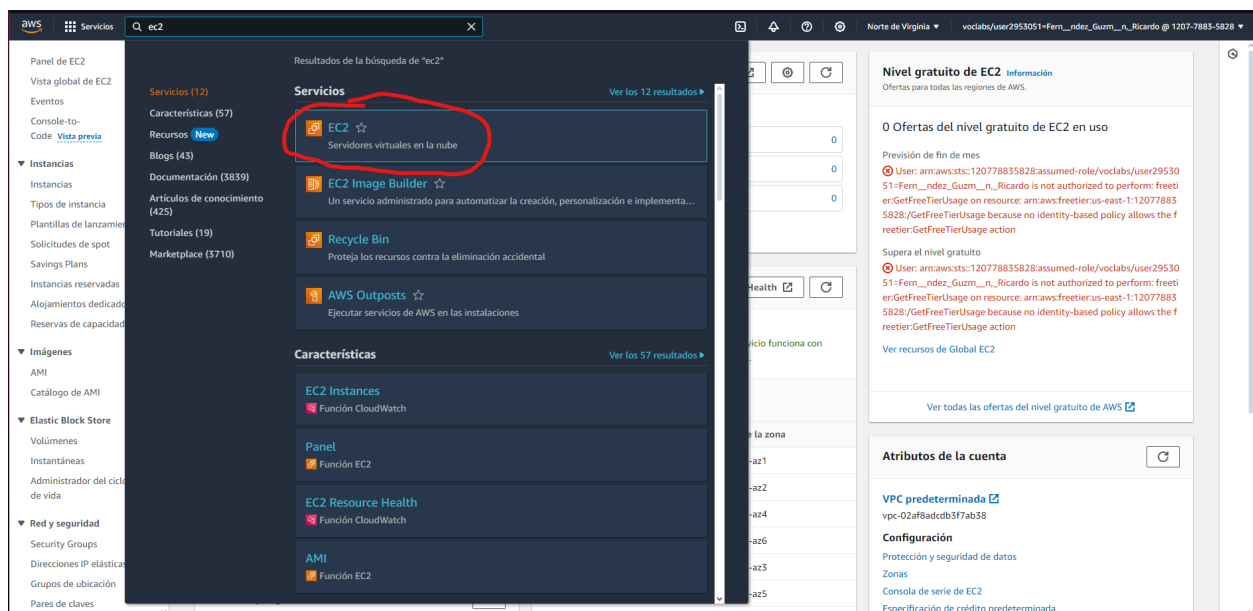
Este apartado vamos a documentar cómo llevar a cabo el despliegue de la aplicación, en principio está preparado para hacerse en AWS pero por si se quisiera realizar de manera local también adjuntan los pasos para llevarlo a cabo de esta forma

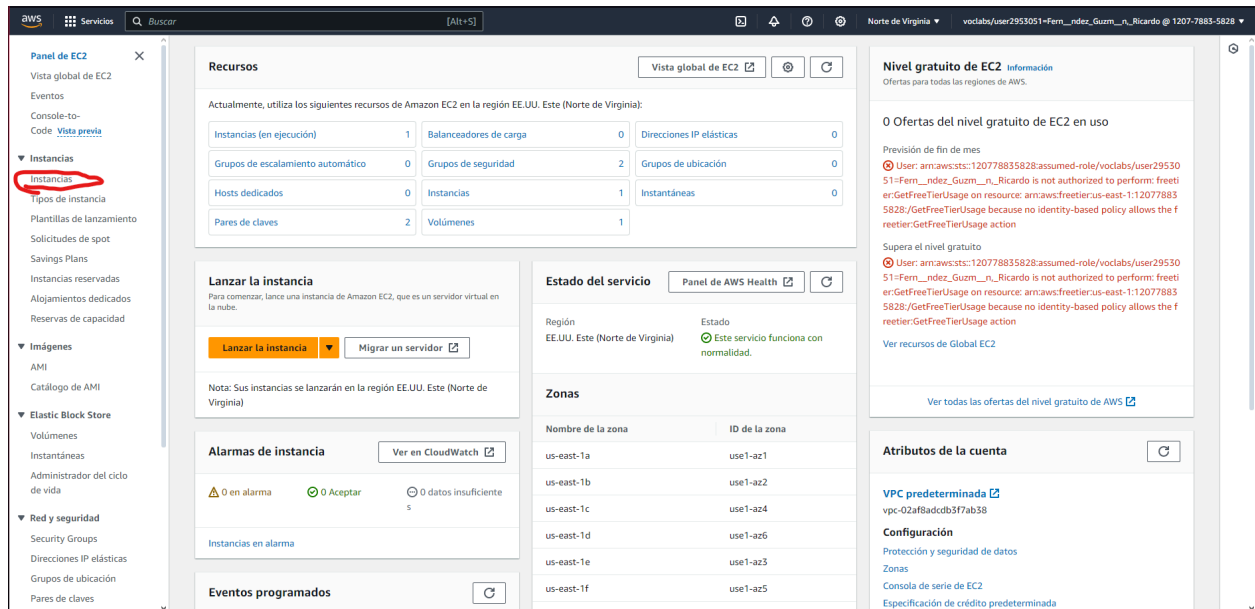
En AWS

Antes de empezar, tenemos que ejecutar el laboratorio en el que se tiene nuestro material (base de datos, front y back).

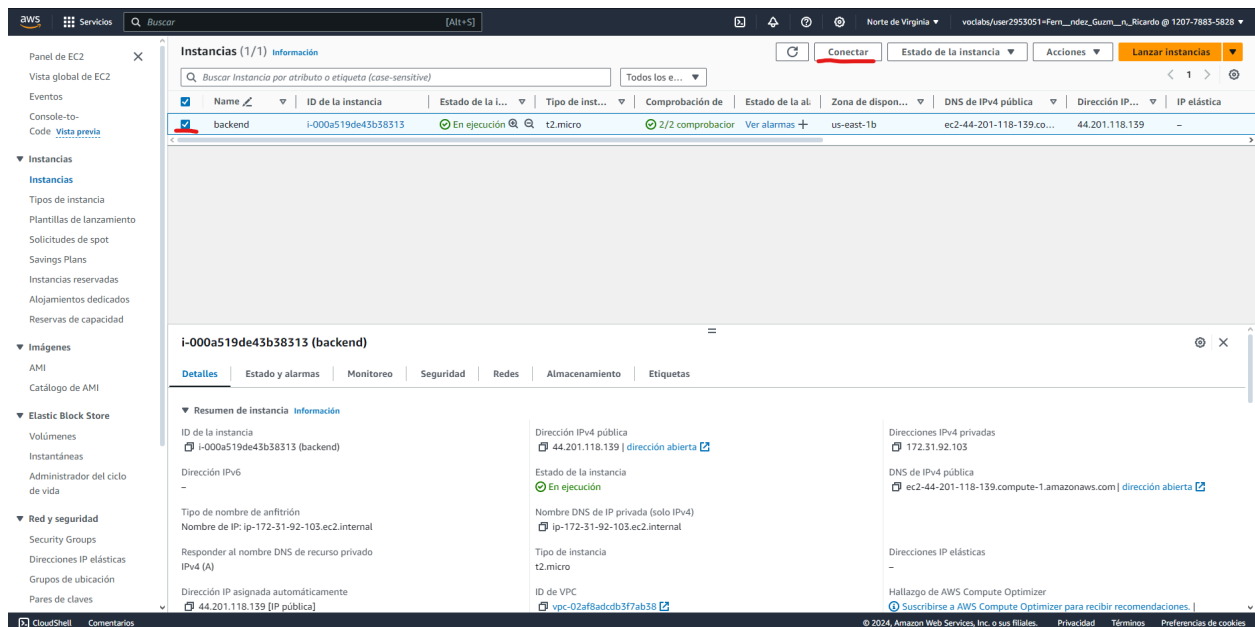
Para llevar el despliegue en AWS es bastante más fácil que en local ya que prácticamente ya está todo subido en la nube y solo se tendrían que realizar unos simples pasos, concretamente desplegar el backend desde la máquina virtual (EC2) de AWS. Los pasos son los siguientes:

Desde AWS nos tenemos que dirigir a EC2 y buscar el apartado instancias





Buscamos la instancia llamada backend, la marcamos y damos en el botón conectar de la parte superior.



Aquí se describen unos pasos a seguir que explicaré más tarde. Lo importante es quedarse con el comando del final, `ssh -i "backend.pem" ec2-user@ec2-44-201-118-139.compute-1.amazonaws.com` (puede variar la dirección de amazon la vez que lo intentéis).

EC2 > Instancias > i-000a519de43b38313 > Conectarse a la instancia

Conectarse a la instancia Información

Conéctese a la instancia i-000a519de43b38313 (backend) mediante cualquiera de estas opciones

Conexión de la instancia EC2

Administrador de sesiones

Cliente SSH

Consola de serie de EC2

ID de la instancia
i-000a519de43b38313 (backend)

1. Abra un cliente SSH.
2. Localice el archivo de clave privada. La clave utilizada para lanzar esta instancia es backend.pem
3. Ejecute este comando, si es necesario, para garantizar que la clave no se pueda ver públicamente.
chmod 400 "backend.pem"
4. Conéctese a la instancia mediante su DNS público:
ec2-44-201-118-139.compute-1.amazonaws.com

Ejemplo:
ssh -i "backend.pem" ec2-user@ec2-44-201-118-139.compute-1.amazonaws.com

Nota: En la mayoría de los casos, el nombre de usuario adivinado es correcto. Sin embargo, lea las instrucciones de uso de la AMI para comprobar si el propietario de la AMI ha cambiado el nombre de usuario predeterminado de la AMI.

Cancelar

Para poder acceder a la máquina virtual AWS se necesita descargar el par de claves que permite el acceso a esta máquina virtual, la cual nos concede que podamos conectarnos de manera remota. Se trata de un archivo de extensión .pem. Se encontrará en el repositorio de este proyecto dentro de la carpeta “par_claves_AWS”

(<https://github.com/RicardoFgX/Proyecto-Final/tree/main/docs/prototipoFigma>)

```
PS C:\Users\Ric\Desktop\clave\clave> ls

Directorio: C:\Users\Ric\Desktop\clave\clave

Mode                LastWriteTime         Length Name
----                -
-ar---            11/06/2024   16:25           1674 backend.pem

PS C:\Users\Ric\Desktop\clave\clave>
```

© 2006 The Authors

```
PS C:\Users\Ric\Desktop\clave\clave> $env:USERNAME
Ric
```

```
ssh -i "backend.pem" ec2-user@ec2-44-201-118-139.compute-1.amazonaws.com
```

```
PS C:\Users\Ric\Desktop\clave> ssh -i "C:\Users\Ric\Desktop\clave\backend.pem" ec2-user@ec2-44-201-118-139.compute-1.amazonaws.com
```

```
      _#_
     /###\   Amazon Linux 2023
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| https://aws.amazon.com/linux/amazon-linux-2023

      _#_
     /###\   Amazon Linux 2023
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| https://aws.amazon.com/linux/amazon-linux-2023

      _#_
     /###\
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| V~' '->

      _#_
     /###\
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| V~' '->

      _#_
     /###\
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| V~' '->

      _#_
     /###\
    /#####\
   /#####\
  /#####\
 /#####\
/_#####\_#|
          |__| V~' '->
```

```
Last login: Tue Jun 11 17:07:14 2024 from 146.158.164.106
[ec2-user@ip-172-31-92-103 ~]$
```

```
[ec2-user@ip-172-31-92-103 ~]$ ls
proyectoFinal-0.0.1-SNAPSHOT.jar
```

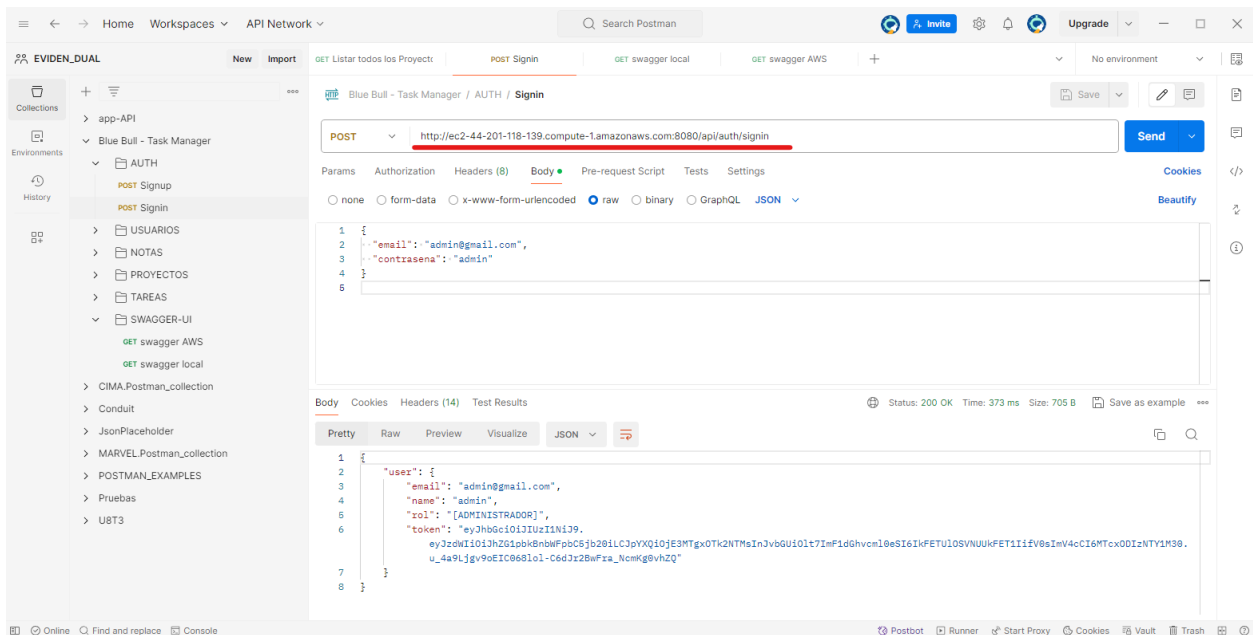
Para ejecutarlo utilizamos el comando: **java -jar proyectoFinal-0.0.1-SNAPSHOT.jar**

```
[ec2-user@ip-172-31-92-103 ~]$ java -jar proyectoFinal-0.0.1-SNAPSHOT.jar

BlueBull

2024-06-12T13:09:07.862Z INFO --- [backend] [ main] c.daw2.proyectoFinal.BackendApplication : Starting
BackendApplication v0.0.1-SNAPSHOT using Java 17.0.11 with PID 2779 (/home/ec2-user/proyectoFinal-0.0.1-SNAPSHOT.jar sta
rtd by ec2-user in /home/ec2-user)
2024-06-12T13:09:07.879Z INFO --- [backend] [ main] c.daw2.proyectoFinal.BackendApplication : The follo
wing 1 profile is active: "demo"
```

Podemos comprobar que funciona por ejemplo con Postman realizando alguna llamada a alguno de los endpoints de la API, por ejemplo para el login:



Una vez desplegado el back, podremos trabajar abrir la aplicación que está contenida en un bucket en AWS y cuya dirección es estática, por lo que el enlace es el siguiente:

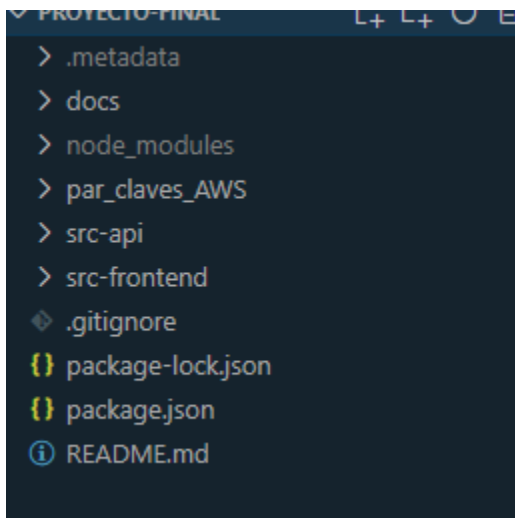
<http://mi-app-angular.s3-website-us-east-1.amazonaws.com>

En local

Requisitos previos:

- Tener java instalado en la máquina
- Tener descargado MySQL y saberse las credenciales del local

Lo primero es clonar el repositorio de GitHub del proyecto (<https://github.com/RicardoFgX/Proyecto-Final>), dejando un arbol de carpeta similar a este (obviando la carpeta metadata y node_modules)



Vamos a desplegar primero el back, para ello lo primero es configurar nuestra base de datos como la de nuestro local, para ello tendremos que editar las propiedades “ProyectoFinal\Proyecto-Final\src-api\src\main\resources\application.properties”

```
application.properties
src-api > src > main > resources > application.properties
1  spring.application.name=backend
2  spring.profiles.active=demo
3  spring.main.allow-bean-definition-overriding=true
4  spring.main.lazy-initialization=true
5
6  jwt.secret=8a17da062537be45a8335bd121669b18259589638bb1be45c49b6266e4b7f538
7
8  server.port=8080
9  server.error.whitelabel.enabled=false
10
11  spring.datasource.url=jdbc:mysql://backend-db.c5wsbcrrxx455.us-east-1.rds.amazonaws.com:3306/backend?createDatabaseIfNotExist=true
12  spring.datasource.username=admin
13  spring.datasource.password=12341234
14
15  spring.jpa.hibernate.ddl-auto=update
16  spring.jpa.show-sql=true
17  spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
18  spring.jpa.database-platform=org.hibernate.dialect.MySQL5InnoDBDialect
19
```

En la imagen anterior está configurada según la configuración de mi maquina. Se tiene que modificar las propiedades **spring.datasource.url**, **spring.datasource.username** y **spring.datasource.password**. Para el primero la url seguiría un patrón como este:

jdbc:mysql://localhost:3306/backend?createDatabaseIfNotExist=true

Con la configuración que venía inicialmente nos permite conectarnos a la base de datos desplegada en AWS.



Postmortem y conclusiones del proyecto.

La ejecución del proyecto "Blue Bull - Task Manager" ha sido un proceso de aprendizaje y desarrollo continuo. Desde la identificación de las necesidades hasta la implementación y despliegue. En cada uno de los pasos se han ido encontrando dificultades que se han ido solventando, aprendiendo de los errores por el camino.

El proyecto comenzó con una fase de análisis y diseño, donde se definieron los requisitos funcionales y una primera puesta de que es lo que se quería hacer con la aplicación

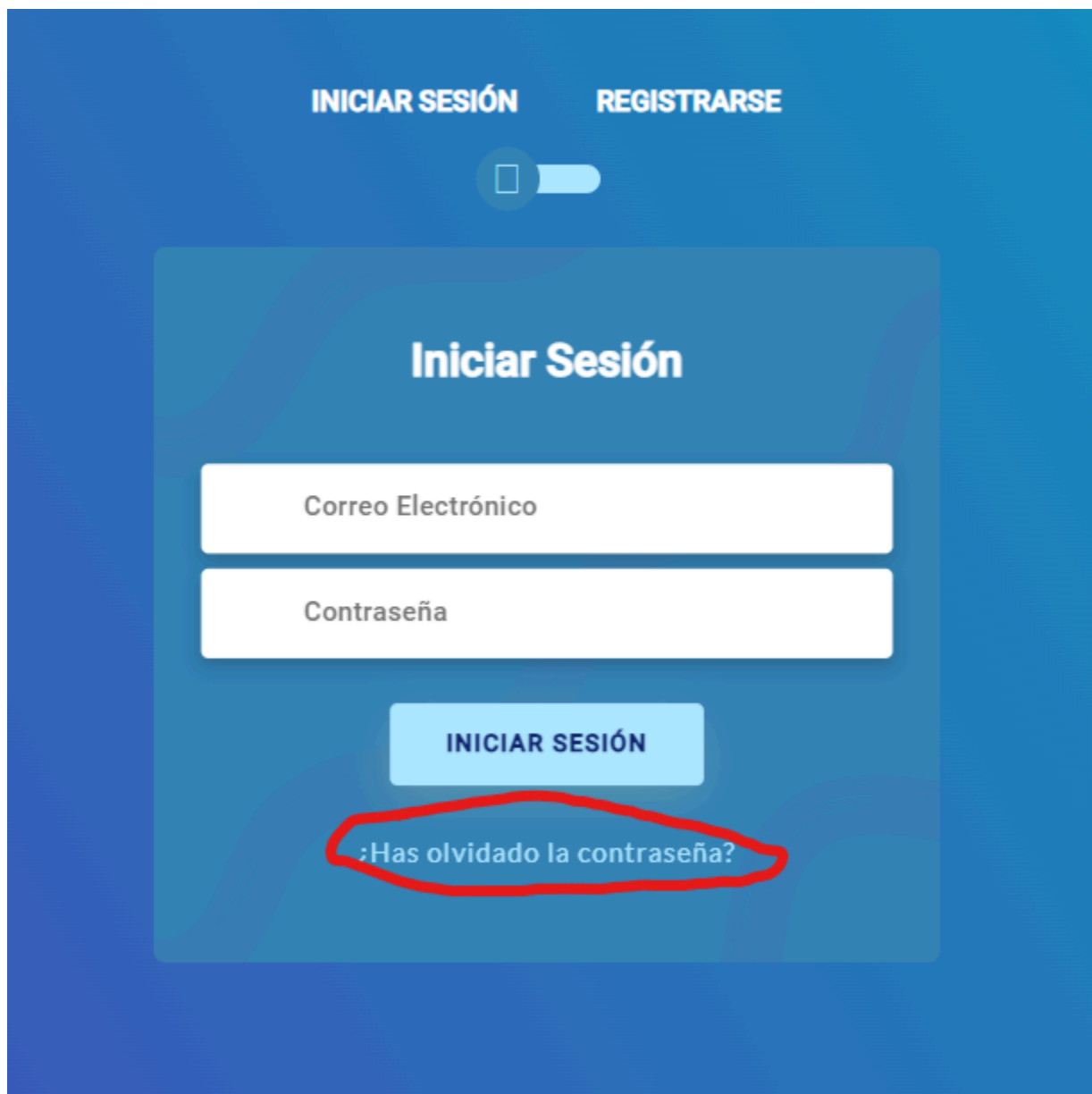
La fase de desarrollo se dividió en frontend y backend, utilizando Angular y Spring respectivamente, por otro lado, se utilizó MySQL para manejar la base de datos de manera eficiente. La autenticación se implementó con JSON Web Tokens (JWT) para asegurar la seguridad de las sesiones de usuario.

El uso de herramientas como Git y GitHub facilitó el control de versiones y la colaboración, mientras que AWS se empleó para asegurar un despliegue consistente en diferentes entornos. El prototipo de la aplicación se creó en Figma, proporcionando una guía visual para el desarrollo.


Posibles Mejoras

Aunque el proyecto ha cumplido con la mayoría de los objetivos, hay áreas que podrían mejorarse, como por ejemplo:

- **Funcionalidades Adicionales:** Incluir más funcionalidades como integraciones con otras herramientas populares, como por ejemplo notificaciones en tiempo real o un sistema para recuperar la contraseña en caso de perderla en la pantalla de login (idea descartada por falta de tiempo)



The image shows a login interface with a blue background. At the top, there are two links: "INICIAR SESIÓN" and "REGISTRARSE". Below them is a toggle switch. The main content area is a light blue box with the title "Iniciar Sesión". It contains two input fields: "Correo Electrónico" and "Contraseña". Below the fields is a blue button labeled "INICIAR SESIÓN". At the bottom of the box, the text ":Has olvidado la contraseña?" is circled in red.

- 
- **Optimización de la Interfaz de Usuario:** Crear una interfaz más intuitiva que requiera de una pequeña curva de aprendizaje para un usuario nuevo.
 - **Automatización de Pruebas:** Implementar pruebas automatizadas para asegurar la calidad del código y facilitar el mantenimiento futuro en diferentes versiones del proyecto. La función de estas sería una vez desplegada una nueva versión desplegarlas para comprobar que los nuevos cambios o implementaciones no han afectado a la funcionalidad del aplicativo. Para ello se podrían utilizar herramientas de automatización como cypress, selenium u Open RPA.

Viabilidad de Puesta en Marcha Real


La viabilidad de poner en marcha un simple proyecto como "Blue Bull - Task Manager" en un entorno real requeriría considerar varios aspectos:

- **Infraestructura:** Asegurar una infraestructura robusta y escalable en la nube, utilizando servicios como AWS para manejar el tráfico y el almacenamiento de datos de manera eficiente, de forma que sea posible el tráfico de grandes cantidades de usuarios simultáneamente
- **Seguridad:** Implementar medidas de seguridad avanzadas para proteger los datos de los usuarios y asegurar la integridad de la aplicación. Para esto he encontrado que podrían servirnos algunos servicios con los que ya cuenta AWS como IAM para la gestión de identidades y accesos, AWS Shield para protección DDoS y AWS WAF para protección contra amenazas web.
- **Marketing y Adopción:** Desarrollar una estrategia de marketing para atraer usuarios y promover la adopción de la aplicación. Esto puede incluir campañas en redes sociales, blogs, webinars y colaboraciones con influencers del sector.

- **Soporte y Mantenimiento:** Es importante una vez desplegada la aplicación atender a la demanda de los usuarios cuando estos encuentren problemas o necesiten ayuda sobre el aplicativo, para ello sería importante establecer un equipo de soporte para manejar consultas y problemas de los usuarios, así como un plan de mantenimiento regular para actualizar y mejorar la aplicación.
- **Monetización:** Por último, pero no por ello menos importante, el tema dinero. Para que un proyecto sea viable es necesario que otorgue un beneficio económico, por ello, para que el proyecto viera futuro a la larga, sería importante definir un modelo de negocio, como suscripciones premium o servicios adicionales, para asegurar la sostenibilidad económica del proyecto.

En conclusión, el proyecto "Blue Bull - Task Manager" ha presentado un desafío en términos de desarrollo e implementación. Sin embargo, con algunas mejoras y una planificación adecuada, tiene el potencial de convertirse en una herramienta valiosa en el mercado de gestión de tareas y proyectos. La experiencia adquirida durante este proceso será un bien invaluable para futuros proyectos y desarrollos.

Anexo



Admin View
Gestiona usuarios y proyectos

- Usuarios
- Apuntes**
- Proyectos






















^/ Apartados aún por determinar

✖ Apartados aún por determinar

</> Apartados aún por determinar

☐ Nombre del archivo








☐ Autor
















<input type="checkbox"/>  Apuntes de reunión	pedro@gmail.com	 
<input type="checkbox"/>  Contraseña PIA	pedro@gmail.com	 
<input type="checkbox"/>  Página de ejemplo	pedro@gmail.com	 
<input type="checkbox"/>  Notas actualización	pedro@gmail.com	 
<input type="checkbox"/>  Notas de ejemplo	pedro@gmail.com	 
<input type="checkbox"/>  Notas aleatorias	pedro@gmail.com	 
<input type="checkbox"/>  Ejemplo3	pedro@gmail.com	 

ejemplo@gmail.com


Cerrar Sesión



-  **Admin View**
Gestiona usuarios y proyectos
-  Usuarios
-  Apuntes
-  **Proyectos**
-  Apartados aún por determinar
-  Apartados aún por determinar
-  Apartados aún por determinar

<input type="checkbox"/>	Nombre del proyecto	Autor		
<input type="checkbox"/>	 Proyecto Gorgonzola	pedro@gmail.com		
<input type="checkbox"/>	 Proyecto XYZ	pedro@gmail.com		
<input type="checkbox"/>	 Hotel 101	pedro@gmail.com		
<input type="checkbox"/>	 Nerfeo Riven	pedro@gmail.com		
<input type="checkbox"/>	 Proyecto Ejemplo	pedro@gmail.com		

Iniciar Sesión



ejemplo@gmail.com

Cerrar Sesión

Admin View
Gestiona usuarios y proyectos

Usuarios
















Apuntes

Proyectos

⌵ Apartados aún por determinar

⌵ Apartados aún por determinar

⌵ Apartados aún por determinar

<input type="checkbox"/>	Nombre	Rol ¹	Correo Electrónico	Contraseña	
<input type="checkbox"/>	 Admin ejemplo	Usuario Admin	admin@gmail.com	admin	 
<input type="checkbox"/>	 Prueba admin2	Usuario Admin	admin2@gmail.com	admin2	 
<input type="checkbox"/>	 Pedro Jimenez Salas	Usuario Estándar	prueba@gmail.com	prueba	 
<input type="checkbox"/>	 Ricardo Fernández Cuatrecasas	Usuario Estándar	ricardo@gmail.com	ricardo123	 
<input type="checkbox"/>	 Pablo Sierra		pablo1304		 

¿Estas seguro de que quieres borrar el usuario?

CANCELAR

BORRAR