WoTaaCa

Workspace Tasks and Categories

Proyecto de Fin de Ciclo

Desarrollo de Aplicaciones Web

2024

Realizado por: **Sergio Fernández Fernández**

Indice General

- Introducción
- Palabras clave
- Indice General
- Objetivos
- Justificación
- Descripción del proyecto
- Modulos formativos aplicados
- Herramientas/Lenguajes utilizados
- Fases del proyecto
- Conclusiones y mejoras
- Anexos
- Bibliografía

Introducción

WoTaaCa es un software diseñado para ayudar a los usuarios a organizar sus tareas de manera flexible y sin imponer una metodología específica. En lugar de seguir un enfoque rígido, WoTaaCa permite a cada usuario estructurar sus actividades según sus propias preferencias.

En un workspace, los usuarios pueden crear tareas, asignar relaciones entre ellas y establecer categorías para una mejor organización. Además, es posible invitar a otros usuarios al workspace, facilitando la colaboración en equipo.

Palabras clave

- Workspace: Un workspace es un espacio aislado en el que los usuarios pueden organizar un conjunto específico de tareas y categorías.
- Tarea: Una tarea es una actividad que el usuario debe realizar, puede ser una tarea simple o una tarea compuesta por subtareas.
- Categoría: Una categoría permite a los usuarios organizar sus tareas y subtareas en grupos, por ejemplo, trabajo, estudio, personal, etc.
  No esta opinionada por lo que tambien puede representar prioridades, o estados de las tareas.
- Prioridad: La prioridad de una tarea indica la importancia de la tarea, puede ser baja, media, alta o critica
- Angular: Angular es un framework de desarrollo de aplicaciones web desarrollado por Google.
- NodeJS: NodeJS es un entorno de ejecución de JavaScript que permite a los desarrolladores crear aplicaciones de servidor.
- Express: Express es un framework de desarrollo de aplicaciones web para NodJS.
- Typescript: Typescript es un lenguaje de programación que extiende JavaScript con características de programación orientada a objetos.
- Docker: Docker es una plataforma de software que permite a los desarrolladores crear, probar y desplegar aplicaciones de forma rápida y eficiente.
- mysql: Mysql es un sistema de gestión de bases de datos relacional.
- TailwindCSS: TailwindCSS es un framework de diseño de sitios web que permite a los desarrolladores crear sitios web con un diseño moderno y atractivo.

Objetivos

Como objetivos he buscado:

- He buscado desarrollar una aplicación que me permita organizar mis tareas, primero de forma no opinionada es decir:
  - No se base en una metodologia en concreto, sino que poder organizar las tareas en función del deseo del usuario.
- Permitir agregar tags que permitan organizar las tareas estos han de ser personalizables en función de las necesidades del usuario, algunos ejemplos:
  - [Bug, Feature, Improvement, etc]
  - [Pendiente, En progreso, Completado, etc]
- Permitir la colaboración dentro de un workspace:
  - Invitar a otros usuarios.
  - Asignación de tareas.
  - Comentarios en las tareas. (No implementado)
- Organizar las tareas ha de ser sencillo y rápido.
  - Todas las tareas pueden ser modificadas en cualquier momento.
  - Ocultar campos que no se usen. (No implementado)
  - Permitir el filtrado de tareas en función de los tags.
  - Permitir añadido de subtareas a las tareas.

Justificación

A lo largo del tiempo he usado diferentes aplicaciones para organizar mis tareas, y he encontrado que la mayoría de ellas son demasiado complicadas o no se adaptan a mis necesidades.
Por eso, he decidido desarrollar mi propia aplicación que me permita organizar mis tareas de forma eficiente y efectiva.

Modulos formativos aplicados

Desarrollo de aplicaciones web en entorno cliente

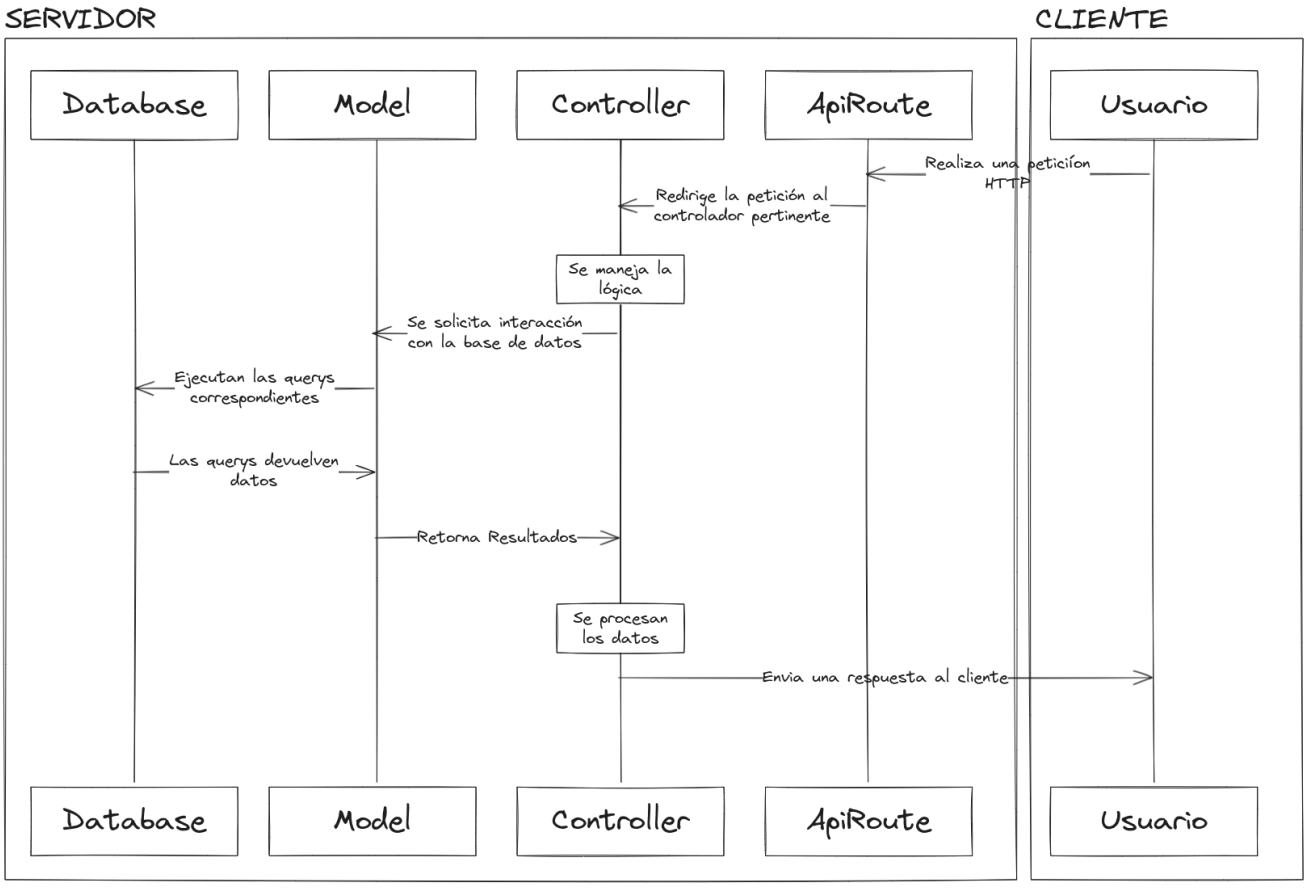
Del módulo de desarrollo de aplicaciones web en entorno cliente, se han aprovechado los siguientes conceptos:

- Uso del framework Angular para el desarrollo.

Desarrollo de aplicaciones web en entorno servidor

Del módulo de desarrollo de aplicaciones web en entorno servidor, saco los siguientes conceptos aplicados:

- Uso de la arquitectura api rest.
- Implementación de controladores y modelos, para la gestión separada de la lógica de negocio y la gestión de la base de datos.



Diseño de interfaces web

Durante el módulo de diseño de interfaces web, se han aprendido los conceptos de diseño de interfaces de usuario, usabilidad y accesibilidad, que se han aplicado en el diseño de la aplicación.

Herramientas/Lenguajes utilizados

Herramientas de desarrollo

- **NeoVim**
  - He Decidido usar NeoVim, por su personalización que me ha permitido adaptarlo a mi gusto, otro de los motivos es que es de codigo abierto una mentalidad que comparto, ademas de ser muy rapido y ligero.
- **Docker**
  - He utilizado docker para el despliegue de una base de datos mysql.
- **Git**
  - Git ha sido usado para manejar las versiones del proyecto, permitiendo navegar entre una rama de testeo, desarrollo y producción, lo que ha facilitado el desarrollo del proyecto.
- **Github**
  - Github me ha permitido alojar los ficheros del proyecto, y asi poder compartirlo entre diferentes equipos.

FrontEnd

- **Angular**
  - La elección de angular radica en:
    - Su estudio en desarrollo de aplicaciones web en entorno cliente.
    - La facilidad que tiene a la hora escalar proyectos.
  - Sin duda angular ha sido una parte clave durante el desarrollo del proyecto, el uso de servicios ha permitido mantener un scope global de los datos, con lo que distintas partes de la aplicación pueden acceder a los datos de forma mas sencilla.
- **TailwindCSS**
  - La elección de tailwindcss para la aplicación de estilos tiene que ver con:
    - Su facilidad de uso, con la que el desarrollo puede ocurrir de forma mas rápida sin tener que entreterse entre ficheros css o ser demasiodo repetitivo.
    - La compatibilidad con angular, que permite su uso sin problema.

BackEnd

- **NodeJS**
  - NodeJS ha sido la elección para el desarrollo del backend por:
    - El rapido acercamiento que permite el framework, gracias a su uso de javascript no ha sido necesario aprender un nuevo lenguaje.
- **Typescript**

    - Typescript ha ofrecido una ayuda inestimable a la hora de desarrollar gracias al tipado, detectar errores durante el desarrollo.

BBDD

- **Mysql**
    - MySql es una base de datos rapida y fiable, ha permitido almacenar y acceder a los datos de forma eficiente.

Fases del proyecto

Estudio de mercado

En esta fase se ha realizado un estudio de mercado para identificar las necesidades de los usuarios y las características de las aplicaciones de gestión de tareas existentes. Se ha analizado la competencia y se han identificado las fortalezas y debilidades de las aplicaciones existentes.

Debilidades de las aplicaciones existentes

- Complejidad: Muchas aplicaciones de gestión de tareas son demasiado complejas y ofrecen más funciones de las necesarias, lo que puede resultar abrumador para los usuarios.
- Rigidez: Algunas aplicaciones están basadas en metodologías específicas, lo que limita la flexibilidad y la adaptabilidad a las necesidades de los usuarios.
- Colaboración: La colaboración entre usuarios puede ser limitada o poco intuitiva en algunas aplicaciones, lo que dificulta la comunicación y la coordinación entre los miembros del equipo.

Fortalezas de las aplicaciones existentes

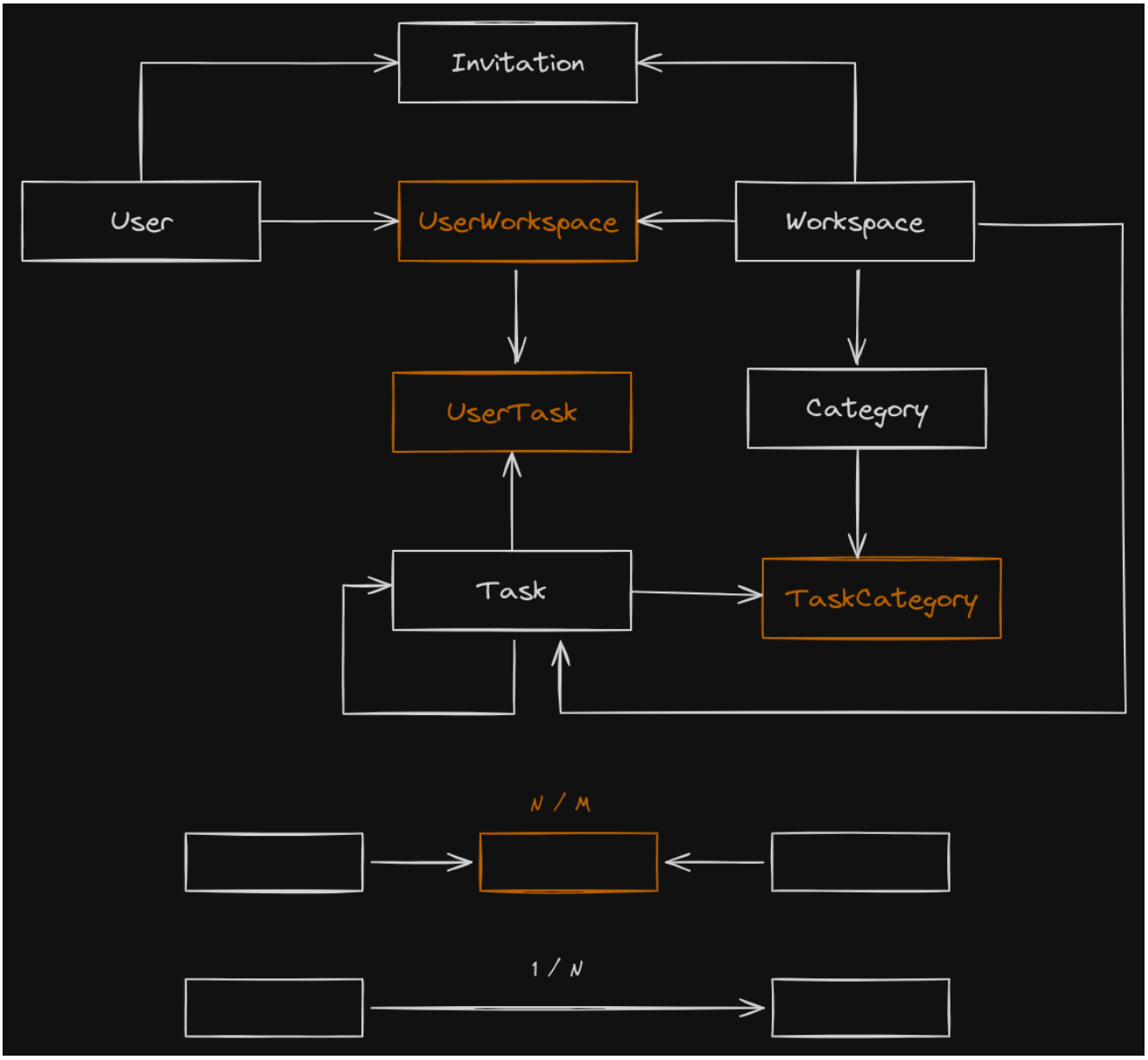
- Funcionalidades avanzadas: Algunas aplicaciones ofrecen funciones avanzadas como seguimiento del tiempo, integración con calendarios, etc.
- Diseño atractivo: Muchas aplicaciones tienen un diseño moderno y atractivo que mejora la experiencia del usuario.
- Integración con otras herramientas: Algunas aplicaciones se integran con otras herramientas como Slack, Google Drive, etc., lo que facilita la gestión de tareas y la colaboración.

Extrapolar las necesidades de los usuarios

Se ha considerado que los usuarios buscan una aplicación de gestión de tareas que sea sencilla, flexible y eficiente.
Los usuarios desean una aplicación que les permita organizar sus tareas de forma personalizada, sin imponer una metodología específica. Además, los usuarios valoran la colaboración y la comunicación entre miembros del equipo, por lo que es importante que la aplicación facilite la colaboración y la asignación de tareas.

Modelo de datos

Diagrama Base de Datos



La base de datos esta compuesta por siete tablas,
5 tablas principales y 3 tablas de relación.

Entidades

User

La entidad User esta destinada a almacenar los datos puros de un usuario.

Field	Type	null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
username	varchar(255)	NO		NULL	
email	varchar(255)	NO		NULL	

Field	Type	null	Key	Default	Extra
password	varchar(255)	NO		NULL	
createdAt	datetime	NO		NULL	
settings	longtext	YES		NULL	
deleted	tinyint	NO		0	

Workspace

La entidad Workspace almacena los datos de los workspace creados por los usuarios.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
name	varchar(50)	NO		NULL	
description	varchar(255)	YES		NULL	
createdAt	datetime	NO		NULL	
deleted	tinyint(1)	NO		0	

UserWorkspace

La entidad UserWorkspace almacena la relación entre un usuario y un workspace, determinando la propiedad y permisos del usuario sobre el workspace.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
idUser	int(11)	NO	MUL	NULL	
idWorkspace	int(11)	NO	MUL	NULL	
role	enum('ADMIN', 'MEMBER', 'GUEST')	NO		GUEST	
deleted	tinyint(1)	NO		NULL	

Invitation

La entidad Invitation consiste en los registros de invitaciones enviadas a un usuario para unirse a un workspace.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
email	varchar(100)	NO		NULL	
idWorkspace	int(11)	NO	MUL	NULL	
createdAt	datetime	NO		NULL	

Field	Type	Null	Key	Default	Extra
deleted	tinyint(1)	NO		0	

Task

La entidad Task almacena los datos de las tareas creadas por los usuarios dentro de un workspaace.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
name	varchar(50)	NO		NULL	
description	varchar(255)	YES		NULL	
createdAt	datetime	NO		NULL	
idWorkspace	int(11)	NO	MUL	NULL	
visibility	enum('PUBLIC', 'PRIVATE')	NO		PRIVATE	
deadline	datetime	YES		NULL	
completed	tinyint(1)	NO		0	
priority	enum('NONE', 'LOW', 'MEDIUM', 'HIGH', 'CRITICAL')	NO		NONE	
dependentIdTask	int(11)	YES	MUL	NULL	
deleted	tinyint(1)	NO		0	

Category

La entidad Category almacena los datos de las categorías creadas por los usuarios dentro de un workspace, que permiten dar matices a las tareas.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
name	varchar(50)	NO		NULL	
description	varchar(255)	YES		NULL	
idWorkspace	int(11)	NO	MUL	NULL	
color	varchar(7)	YES		NULL	
completed	tinyint(1)	NO		0	
deleted	tinyint(1)	NO		0	

TaskCategory

La entidad TaskCategory almacena la relación entre una tarea y una categoría.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
idTask	int(11)	NO	MUL	NULL	
idCategory	int(11)	NO	MUL	NULL	

UserTask

La entidad UserTask almacena la relación entre un usuario y una tarea, determinando la propiedad y permisos del usuario sobre la tarea.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
idUser	int(11)	NO	MUL	NULL	
idTask	int(11)	NO	MUL	NULL	
role	enum('ADMIN','MEMBER','GUEST')	NO		GUEST	

Diseño de la aplicación

Estilos visual

Para el estilo visual se ha buscado un diseño tipo TUI o Terminal User Interface, el cual se caracteriza por su estética simple y funcional. En este diseño, se destacan el uso de colores planos y tipografías monoespaciadas, creando un aspecto retro y nostálgico que rememora las antiguas interfaces de terminal.

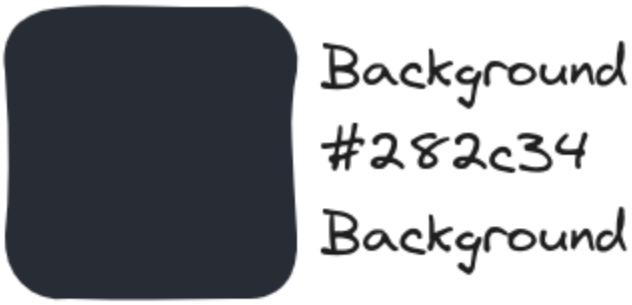
El uso de fuentes monoespaciadas no solo contribuye a la apariencia vintage del diseño, sino que también ofrece ventajas prácticas, como la alineación perfecta de caracteres, lo cual es esencial en entornos de programación y desarrollo. Estas fuentes monoespaciadas se han aprovechado especialmente en los títulos de las secciones, dando un toque distintivo y coherente a la interfaz.

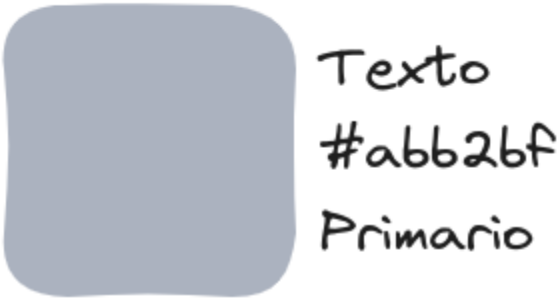
Además, la elección de colores planos simplifica el diseño, eliminando distracciones visuales y permitiendo que el contenido sea el protagonista. Esta simplicidad cromática es fundamental para mantener la claridad y legibilidad, aspectos cruciales en una interfaz de usuario terminal.

En resumen, el diseño TUI combina funcionalidad y estilo, haciendo uso de elementos visuales que no solo mejoran la estética, sino que también optimizan la experiencia del usuario. Este enfoque minimalista pero efectivo convierte a las interfaces de usuario de terminal en una opción atractiva para desarrolladores y usuarios que valoran la eficiencia y la simplicidad.

Paleta de colores

La paleta de colores se ha visto influenciada por el tema One Dark, un esquema de colores creado para el editor de codigo atom, que se caracteriza por una combinación de tonos oscuros y colores vibrantes.





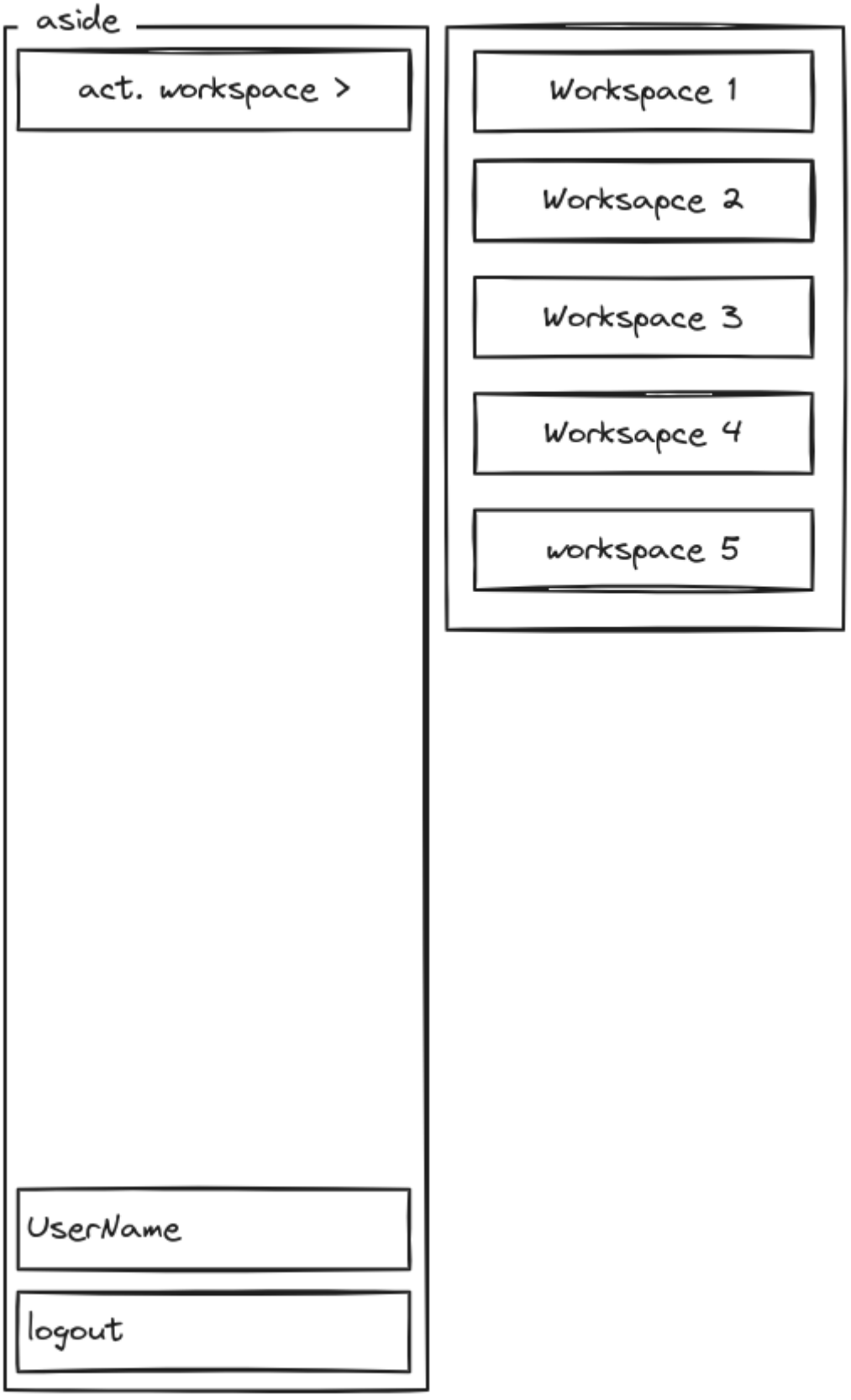
Tipografía

Para la tipografía he decidido usar la fuente Inter por su legibilidad y su aspecto moderno. La fuente Inter es conocida por su claridad y su diseño optimizado para la lectura en pantallas, lo que la convierte en una elección ideal para la interfaz de usuario. Su aspecto limpio y contemporáneo contribuye a una experiencia de usuario agradable y eficiente.

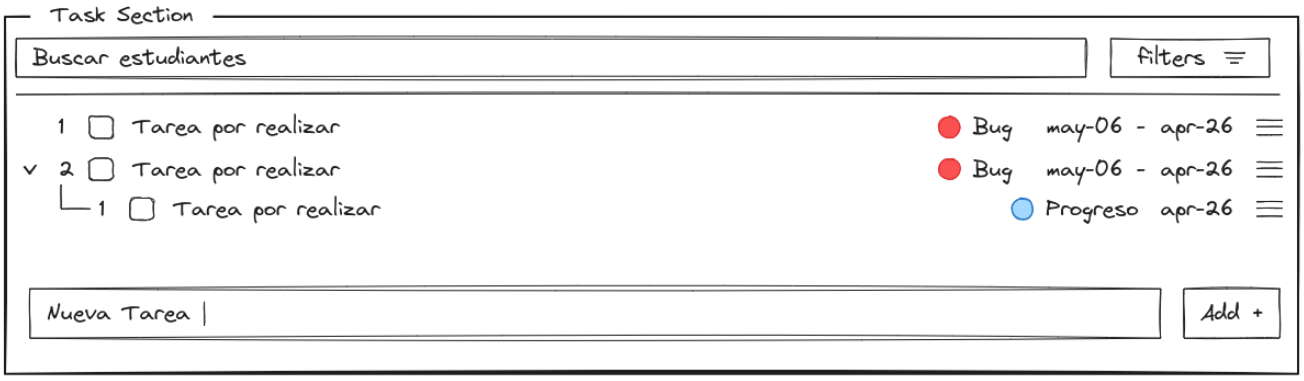
Por otro lado, los títulos aprovechan la fuente Fira Code por su aspecto monoespaciado. Fira Code no solo ofrece la alineación perfecta de caracteres, esencial en entornos de programación y desarrollo, sino que también incluye ligaduras de programación que mejoran la legibilidad del código. Estas caracerísticas hacen que Fira Code sea una opción excelente para los títulos, proporcionando un contraste visual interesante y manteniendo la coherencia estilística con el diseño TUI.

Wireframes

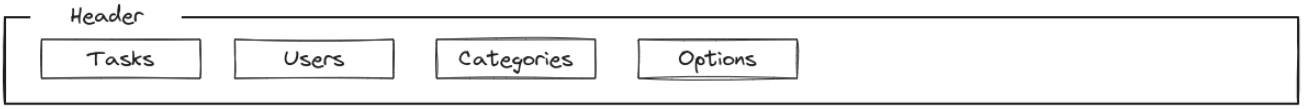
Aside



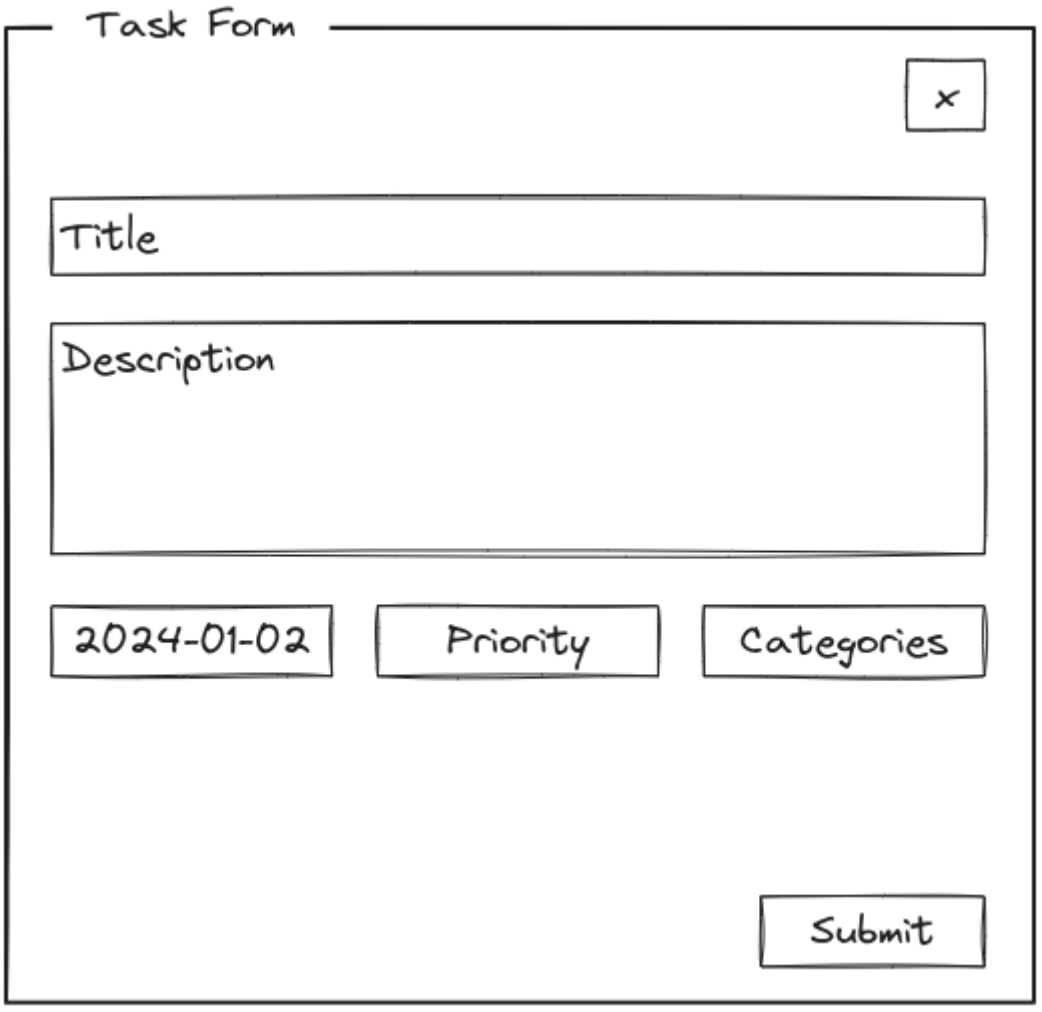
Task Section



Header



Modal Forms

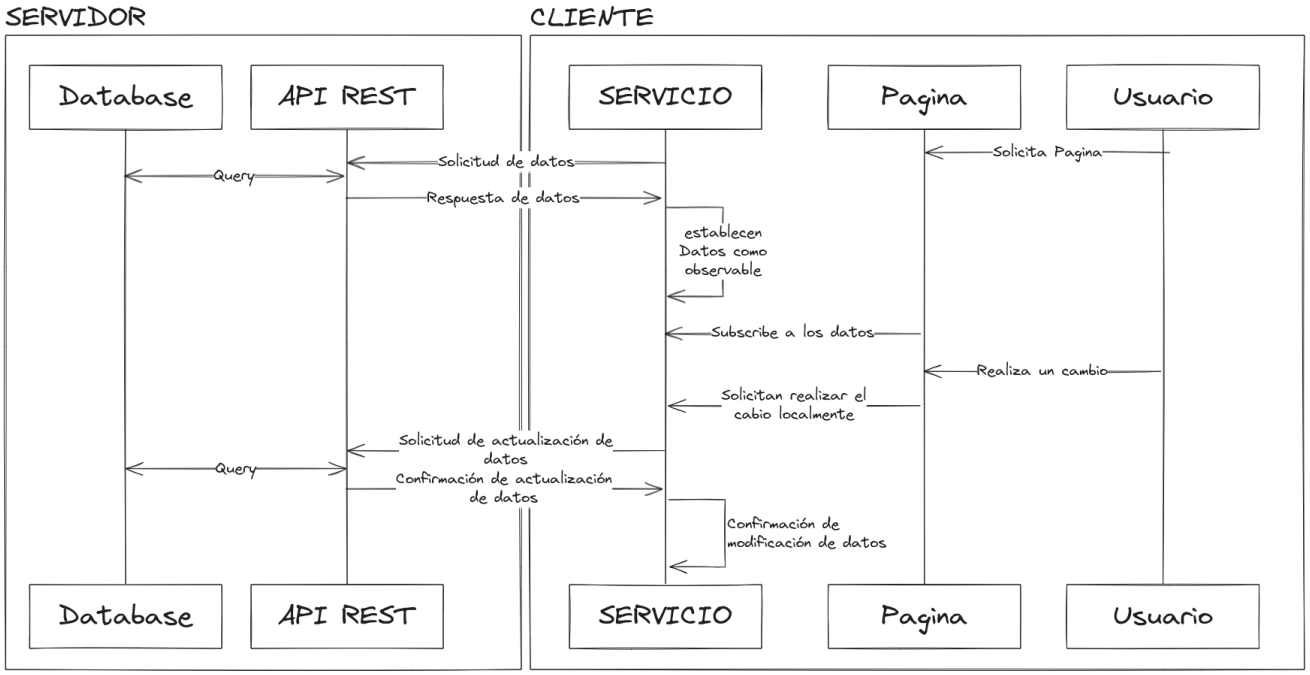


Iconos

Para los iconos se ha usado la libreria Remix Icon, que ofrece una amplia variedad de iconos modernos y estilizados que se adaptan perfectamente al diseño TUI. Estos iconos no solo añaden un toque visual atractivo, sino que también mejoran la usabilidad y la accesibilidad de la interfaz, facilitando la identificación de las funciones y elementos de la aplicación.

Diagramas de flujo

Flujo de Solicitud y modificación de datos



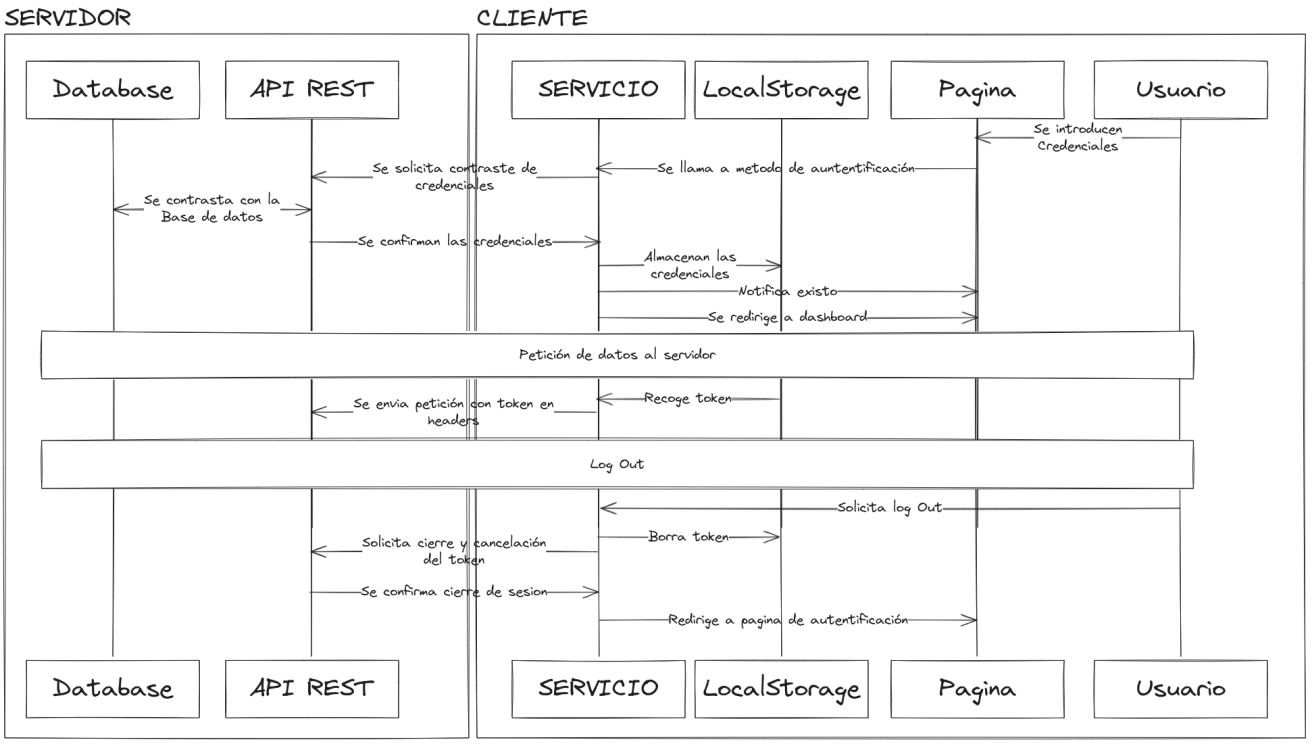
A la hora de manejar los datos de la aplicación, se ha seguido una arquitectura que permite primero la modificación de los datos en el cliente, para posteriormente pedir una actualización de los datos en el servidor.
El servidor confirmara o denegara la actualización de estos datos, y en función de la respuesta se actualizaran los datos en el cliente.

Esto nos permite una mayor responsividad en la aplicación, ya que los datos se actualizan de forma inmediata y no es necesario pasar por ninguna recarga de la página.

Este flujo de datos se utiliza para todas las operaciones de la apliación desde la creación de una tarea, hasta la eliminación de un workspace.

Los datos se almacenan en la base de datos, el servidor devuelve los datos al cliente que lo almacena en un servicio en forma de observable, al cual se suscriben los distintos componentes de la aplicación. Permitiendoles acceder a los datos actualizados en tiempo real.

Flujo de autenticación



El flujo de autenticación se basa en el uso de tokens JWT, que permiten la autenticación de los usuarios en la aplicación. Cuando un usuario se registra o inicia sesión, se genera un token JWT que se almacena en el cliente y se envía en las cabeceras de las peticiones al servidor. El servidor verifica la validez del token y permite el acceso a las rutas protegidas si el token es válido.

Este metodo aunque seguro, no es infalible, ya que un atacante podria robar el token y hacerse pasar por el usuario implementar un sistema de refresco de tokens, permitiria mejorar la seguridad de la aplicación.

Conclusiones y mejoras

En conclusión, el proyecto PALET es un planificador de tareas que permite a los usuarios organizar sus actividades de forma no opinionada y eficiente.

Futuras mejoras:

- Sistema de asignación multiple de usuarios asinados a una tarea.
- Reformar la pestaña de invitaciones para convertirla en una pestaña de notificaciones, completa con indicaciones varias.
- Implementar un sistema de que notifique al usuario cuando una tarea está próxima a vencer.
- Notificar al usuario cuando una tarea le ha sido asignada por otro usuario.
- Implementar pagina de estadisticas de workspace, con graficos indicando el progreso de las tareas, que usuarios han completado mas tareas, etc.
- Sistema de comentarios en las tareas, esto permitiria a los usuarios comunicarse de forma mas eficiente.
- Mejorar el sistema de refresco de tokens, para mejorar la seguridad de la aplicación.
- Mejorar el actual sistema de refresco de datos, para que los datos se actualicen de forma mas eficiente.

Anexos

Instalación de dependencias

1. Acceder a la carpeta mysql y ejecutar el comando `docker compose up`
2. Acceder a la carpeta backend y ejecutar el comando `npm install`
3. Acceder a la carpeta frontend y ejecutar el comando `npm install`

Ejecución del proyecto

Hay que ejecutar tres modulos

1. Acceder a la carpeta backend y ejecutar el comando `npm run dev` o `node --run dev`
2. Acceder a la carpeta frontend y ejecutar el comando `ng serve`

Si se ha finalizado la ejecución del docker de mysql. 3. Acceder a la carpeta mysql y ejecutar el comando `docker compose start`

Acceso a la aplicación

Acceder a la dirección `http://localhost:4200` para acceder a la aplicación.

Acceso a la base de datos

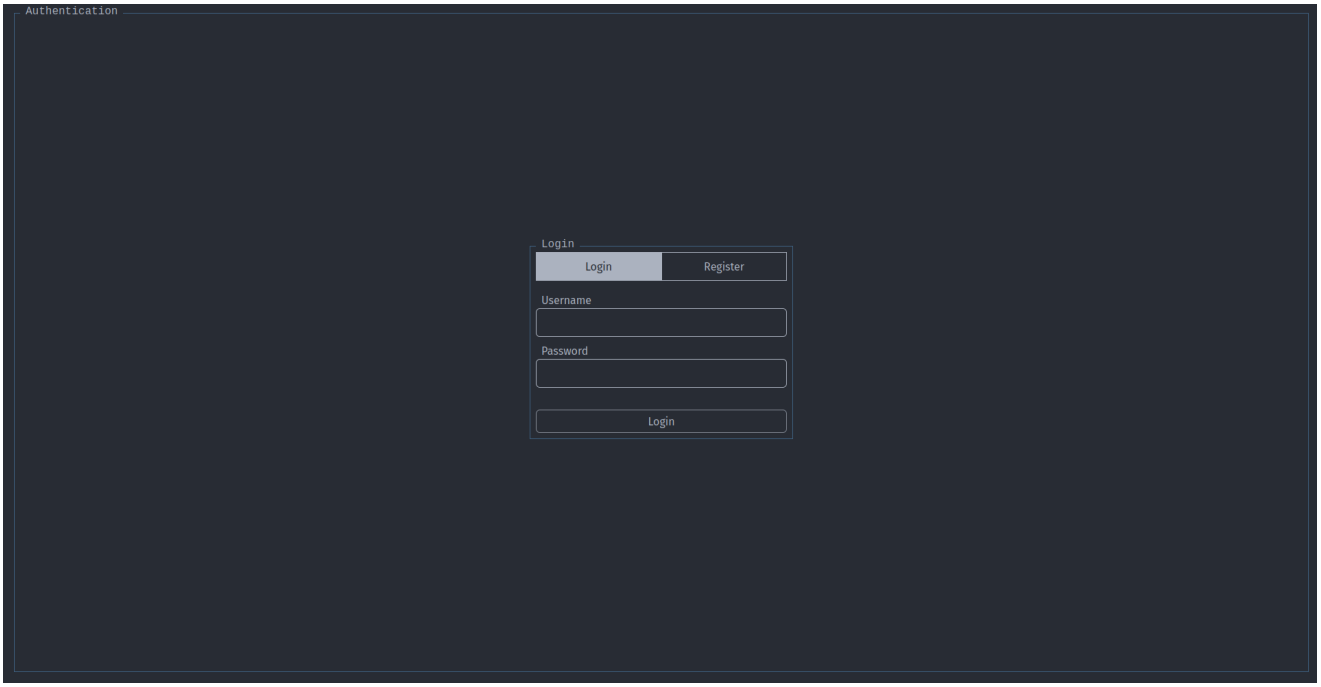
Para acceder a la base de datos, se puede usar cualquier cliente de mysql, como por ejemplo `MySQL Workbench` o `DBeaver`.

Tambien se puede acceder a la base de datos en `http://localhost:8080` con las siguientes credenciales:

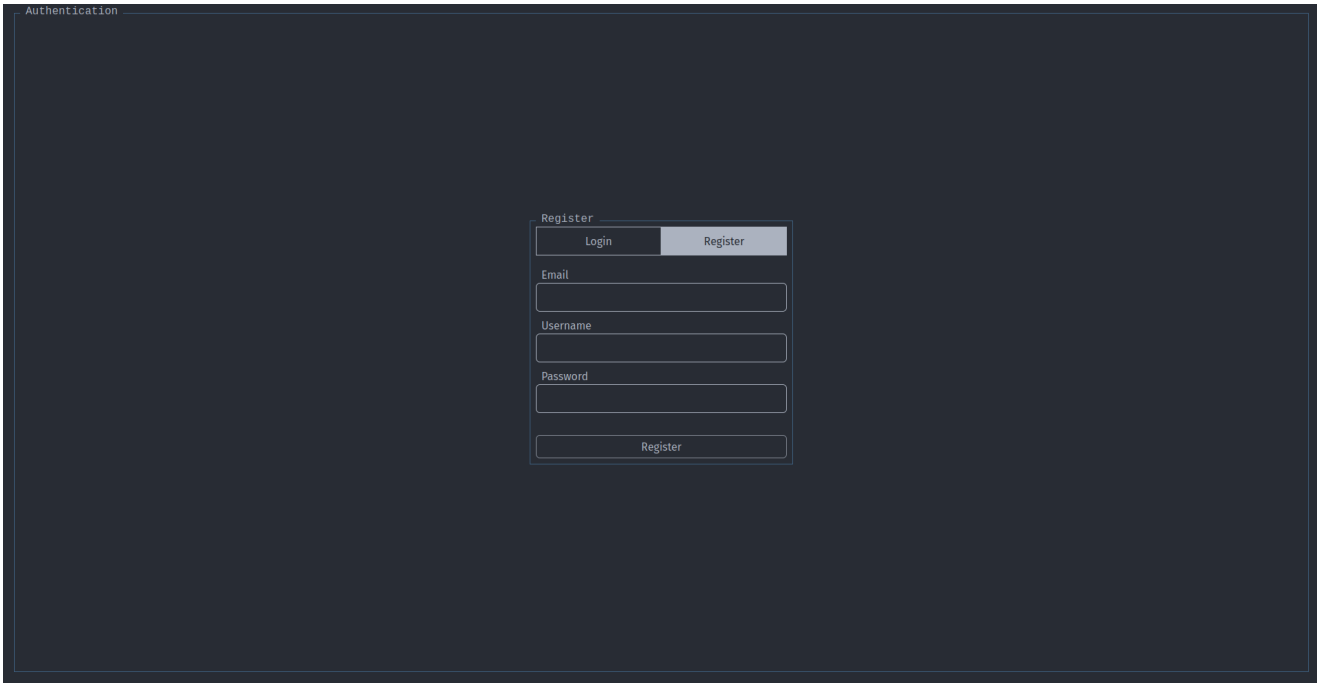
- Usuario: root
- Contraseña: rootpassdev

Imagenes de la aplicación

Login



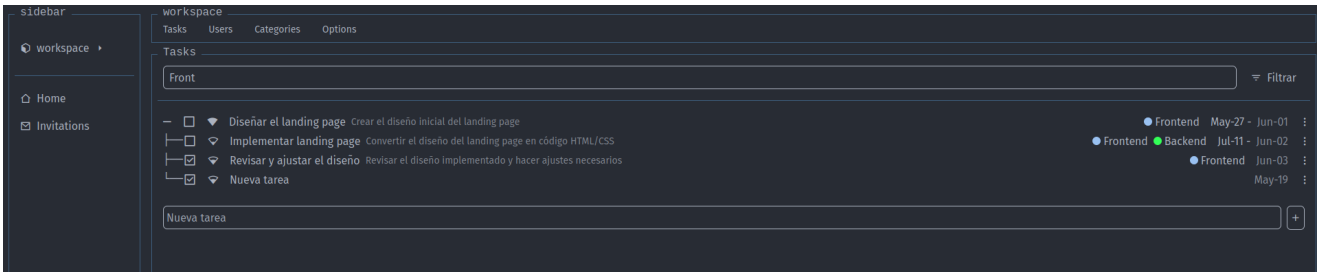
Register



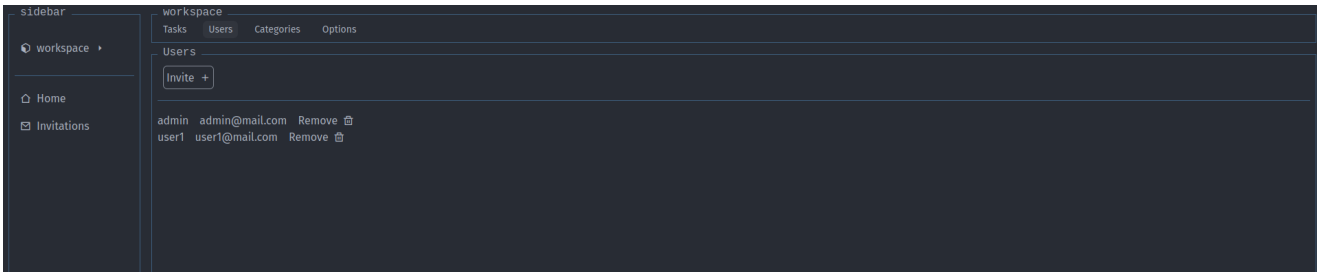
Workspaces

“./assets/img/workspace.png” could not be found.

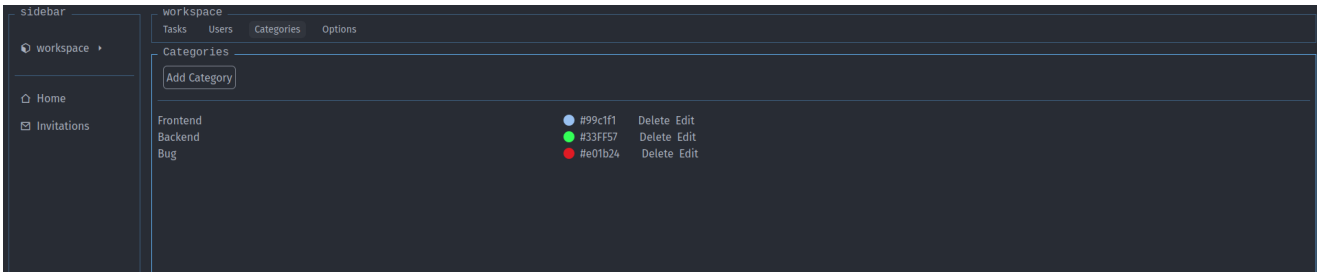
Tasks



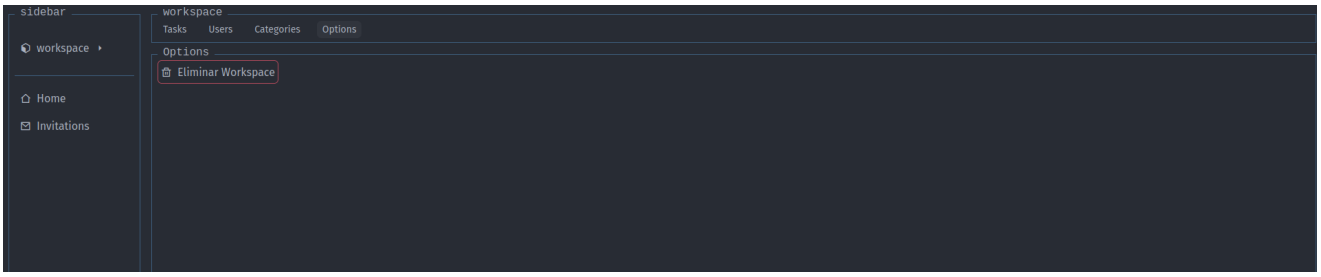
Users



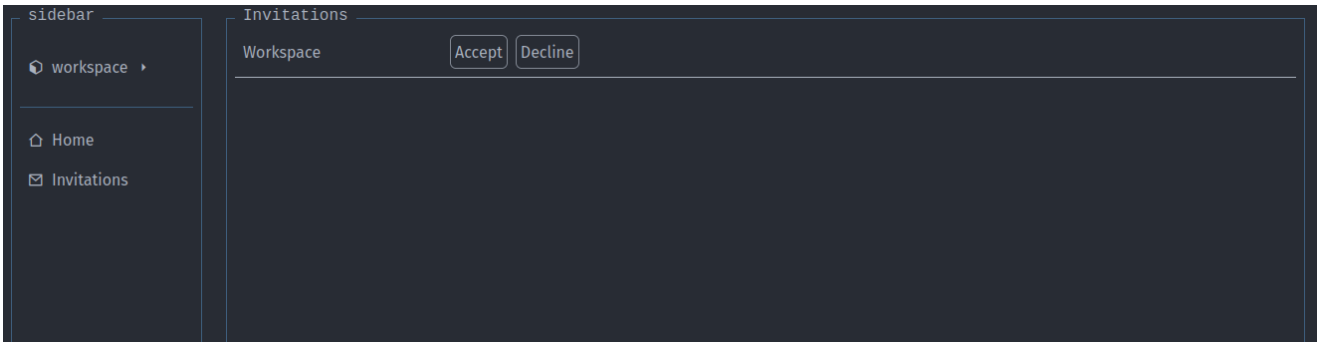
Categories



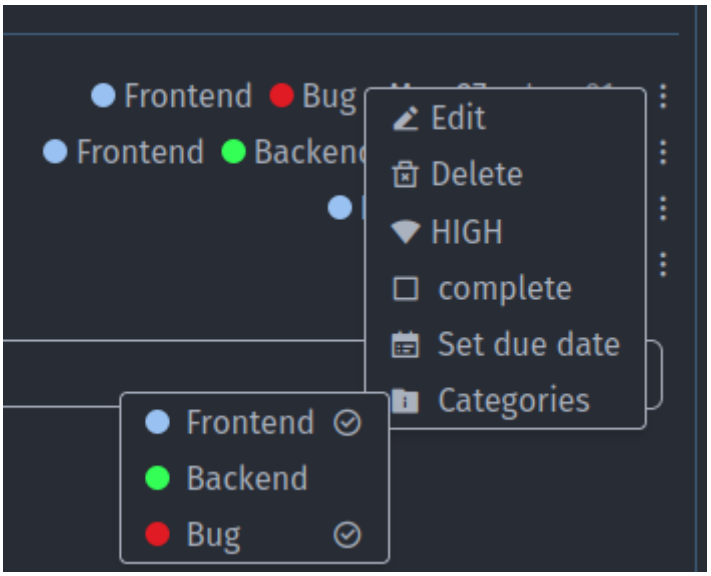
Options



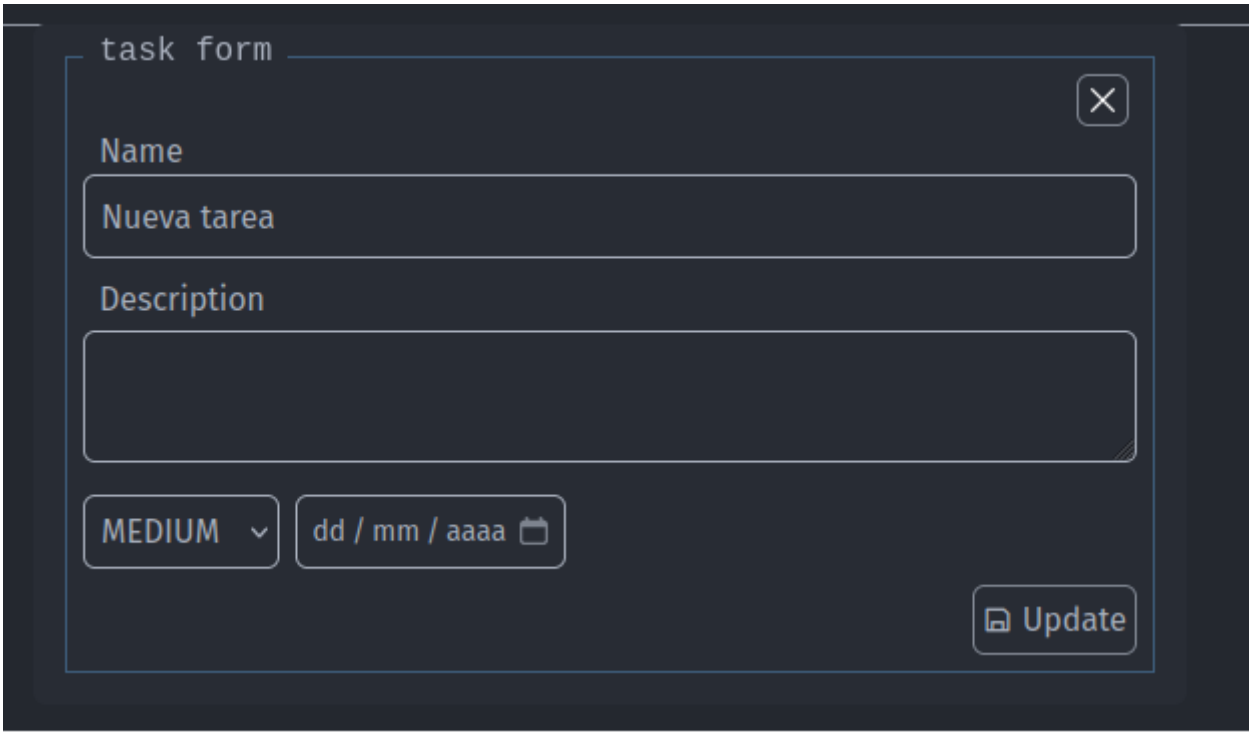
Invitations



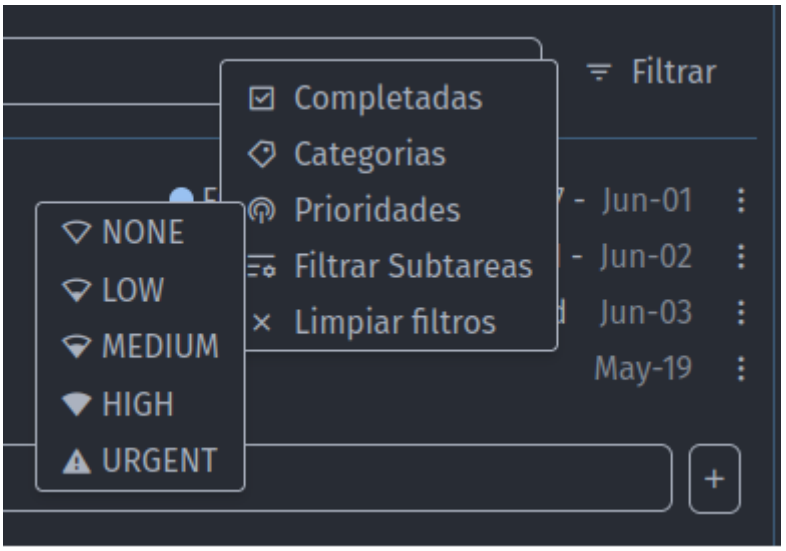
Submenus



Task Form



Filters



Bibliografía

- Angular
- TailwindCSS
- NodeJS
- Express
- Typescript