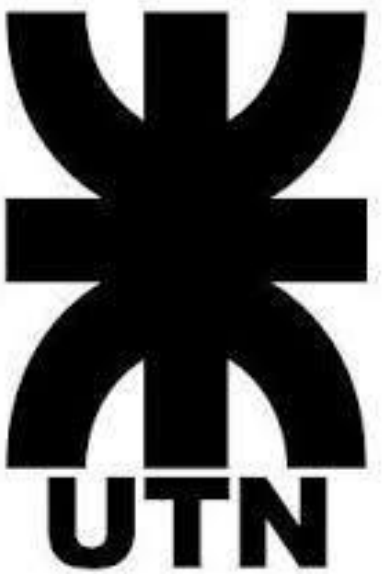
UNIVERSIDAD TECNOLÓGICA NACIONAL

FACULTAD REGIONAL ROSARIO

Tecnicatura Universitaria en Programación



Práctica Profesional Supervisada

Documentación del Proyecto:

“Task Minder”

2024-2TUP4

Integrantes:

Lorenzo Carignani Legajo: 50530

Ramiro Di Carlo Legajo: 51577

Franco García Legajo: 51661

Índice

Metodología de trabajo:

En nuestro proyecto de desarrollo de una aplicación de administración de tareas y agendas, utilizamos la metodología Scrum para garantizar un proceso de trabajo ágil y eficiente. Scrum es un marco de trabajo liviano que ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones adaptativas para problemas complejos.

Elementos Clave de Scrum:

Roles:

Product Owner: Responsable de maximizar el valor del producto y gestionar el Product Backlog.

Scrum Master: Facilita el proceso Scrum, eliminando obstáculos y asegurando que el equipo siga las prácticas de Scrum.

Desarrolladores: Encargados de convertir los elementos del Product Backlog en incrementos de valor durante los Sprint.

Eventos:

Sprint: Ciclo de trabajo de una duración fija (un mes o menos) en el que se crea un incremento de producto.

Sprint Planning: Reunión para planificar el trabajo del Sprint, definiendo el objetivo del Sprint y seleccionando los elementos del Product Backlog que se trabajarán.

Daily Scrum: Reunión diaria de 15 minutos para inspeccionar el progreso hacia el objetivo del Sprint y adaptar el plan según sea necesario.

Sprint Review: Revisión del trabajo realizado al final del Sprint con los interesados, para inspeccionar el incremento y adaptar el Product Backlog según lo aprendido.

Sprint Retrospective: Reunión para reflexionar sobre el Sprint y planificar mejoras en la calidad y efectividad del equipo.

Artefactos:

Product Backlog: Lista emergente y ordenada de todo lo necesario para mejorar el producto.

Sprint Backlog: Conjunto de elementos del Product Backlog seleccionados para el Sprint, más un plan para entregarlos.

Incremento: Un incremento es un peldaño concreto hacia el objetivo del producto, que debe ser utilizable y cumplir con la Definición de Terminado.

Implementación en Trello:

Utilizamos Trello para gestionar visualmente nuestro flujo de trabajo. En Trello, implementamos el método Kanban, que nos permite visualizar el progreso del trabajo y gestionar tareas de manera eficiente. Configuramos tableros con listas que representan las diferentes etapas del flujo de trabajo: " Backlog", " In Progress", "Stand By", “Canceled” y “Done”. Las tarjetas en cada lista representan las tareas individuales, que se mueven de una lista a otra a medida que se avanza en el proceso de desarrollo.

La integración de Trello con Scrum facilita la colaboración del equipo, mejora la transparencia y permite una inspección y adaptación continuas, siguiendo los principios de la metodología Scrum.

Introducción

Introducción al Sistema Desarrollado:

Nuestro sistema es una aplicación de administración de tareas y proyectos, diseñada para mejorar la organización y eficiencia en la gestión de proyectos, similar a herramientas como Trello o Jira.

Propósito:

El propósito del sistema es proporcionar una plataforma robusta y fácil de usar para la creación, asignación y gestión de tareas, proyectos y usuarios. Está pensado para facilitar el trabajo colaborativo y el seguimiento del progreso en un entorno profesional, permitiendo a los equipos gestionar sus proyectos de manera eficiente y efectiva.

Características Distintivas:

Gestión de Usuarios (ABML de Usuarios):

Alta, baja y modificación de usuarios: Permite a los administradores gestionar los usuarios del sistema con facilidad.

Listado de usuarios: Visualización detallada de los usuarios mediante una "User Card".

Roles y permisos: Implementación de roles como Sysadmin, Admin y Programador, cada uno con permisos específicos para garantizar la seguridad y la correcta distribución de tareas.

Autenticación: Funcionalidades de login y logout con validación de roles para mantener la seguridad del sistema.

Gestión de Proyectos (ABML de Proyectos):

Alta, baja y modificación de proyectos: Facilita la gestión completa de los proyectos, desde su creación hasta su finalización.

Listado y filtro de proyectos: Permite una organización eficiente y la fácil localización de proyectos mediante una "Project Card".

Gestión de Tareas (ABML de ToDos):

Alta, baja y modificación de tareas: Herramientas completas para la gestión de tareas.

Listado y filtro de tareas: Incluye visualización detallada y opciones de filtro mediante una "ToDo Card".

Notificaciones por correo: Envío de correos electrónicos para notificar sobre tareas vencidas, asegurando que los programadores se mantengan al día con sus responsabilidades.

Interfaz de Usuario:

Navbar y enrutamiento: Un sistema de navegación intuitivo que facilita el acceso a las funcionalidades principales.

Tema oscuro: Una opción de tema oscuro para mejorar la experiencia del usuario y reducir la fatiga visual.

Traducción: Opción de traducir la aplicación al inglés, lo cual extiende su accesibilidad a una audiencia más amplia.

Alcance del proyecto:

El proyecto tiene como objetivo desarrollar una aplicación web de administración de tareas y agendas, similar a Trello o Jira, para mejorar la organización y gestión de proyectos. Las funcionalidades incluirán la creación y asignación de tareas, gestión de proyectos, notificaciones y recordatorios, y un sistema de roles y permisos para administradores y usuarios. La aplicación se desarrollará utilizando React para el frontend, .NET para el backend y MySQL para la base de datos. Los entregables incluirán un prototipo funcional, una versión beta para pruebas y la versión final del producto, junto con la documentación técnica y manuales de usuario.

Cuestiones Fuera del Alcance:

Este proyecto no incluirá el desarrollo de una aplicación móvil ni integraciones avanzadas con herramientas externas como Slack o Google Calendar en esta fase. Además, no se proporcionará soporte técnico continuo ni servicios de mantenimiento después del lanzamiento inicial. La promoción y el marketing del producto tampoco están incluidos en el alcance del proyecto actual.

Descripción de los Usuarios del Sistema:

En nuestro sistema de administración de tareas y proyectos, hemos definido varios tipos de usuarios, cada uno con roles y responsabilidades específicas para asegurar una gestión eficaz y segura de las tareas y proyectos.

SuperAdmin (Administrador del Sistema):

Responsabilidades:

Gestionar todos los usuarios del sistema: alta, baja y modificación.

Acceso completo a la gestión de proyectos y tareas.

Configuración y mantenimiento del sistema, asegurando su correcto funcionamiento.

Permisos:

Acceso total a todas las funcionalidades del sistema, incluyendo ABML (Alta, Baja, Modificación y Listado) de usuarios, proyectos y tareas.

Visualización y modificación de cualquier usuario, proyecto o tarea.

Admin (Administrador de Proyectos):

Responsabilidades:

Gestionar proyectos: crear, modificar y eliminar proyectos.

Organizar y supervisar a los programadores dentro de los proyectos.

Permisos:

Acceso a la gestión de proyectos (ABML).

Puede listar todos los proyectos y modificar cualquier proyecto.

No puede gestionar usuarios, excepto en términos de asignación a proyectos.

Programmer (Programador):

Responsabilidades:

Gestionar sus propias tareas: crear, modificar y eliminar tareas.

Organizar sus tiempos y actividades para cumplir con los plazos.

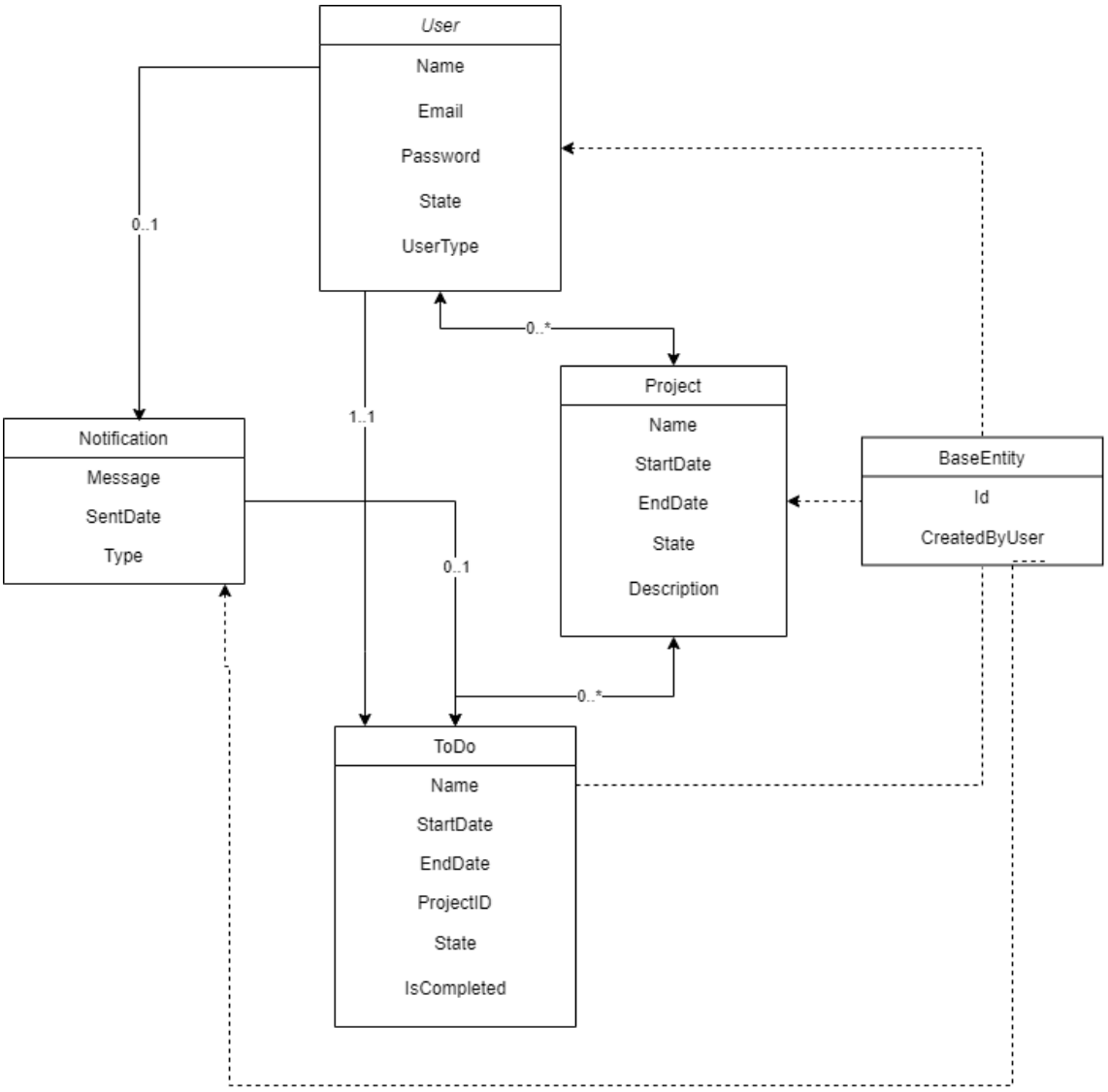
Permisos:

Acceso a la gestión de sus propias tareas (ABML).

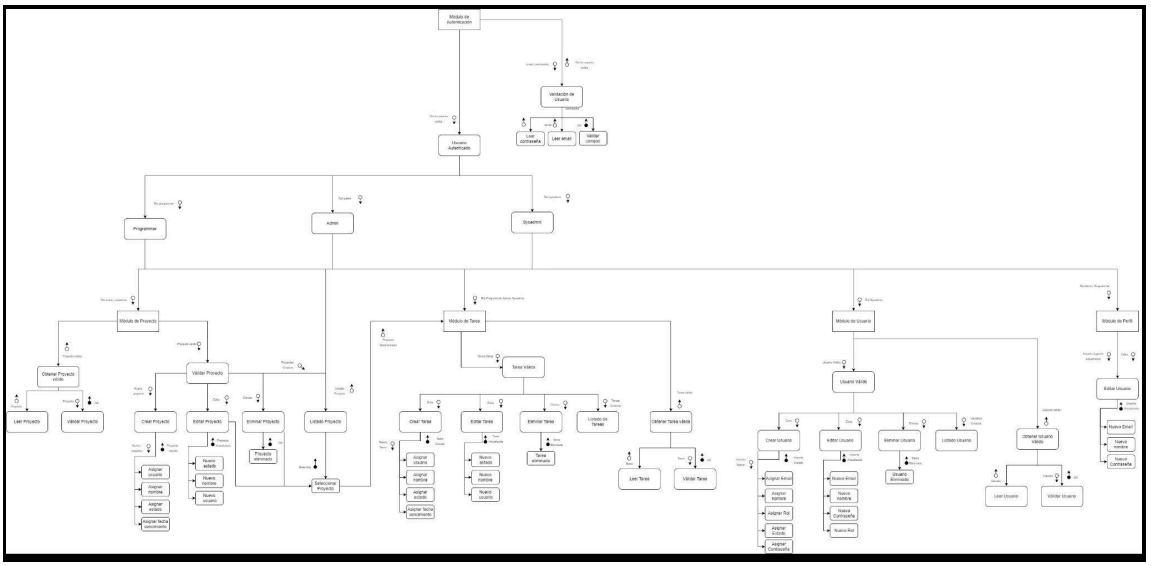
Puede listar todas las tareas, pero solo puede modificar las tareas asignadas a él mismo.

Recibe notificaciones por correo electrónico para las tareas por vencer.

Diagrama de dominio:



Módulos del sistema y su descripción



Módulo de Gestión de Usuarios:

Descripción: Permite la creación, modificación, y eliminación de usuarios en el sistema.

Funcionalidades:

- Alta, baja y modificación de usuarios.
- Asignación de roles y permisos.
- Autenticación y validación de usuarios.

Interacciones: Interactúa con el módulo de autenticación para validar los usuarios y con el módulo de gestión de proyectos para asignar usuarios a proyectos.

Roles y Permisos: Accesible por SuperAdmin y Admin, los cuales pueden gestionar todos los usuarios. Los Programadores solo pueden ver sus propios perfiles.

Módulo de Gestión de Proyectos:

Descripción: Facilita la creación y gestión de proyectos dentro del sistema.

Funcionalidades:

- Creación, modificación, y eliminación de proyectos.
- Asignación de usuarios a proyectos.
- Listado y filtrado de proyectos.

Interacciones: Interactúa con el módulo de gestión de tareas para vincular tareas a proyectos.

Roles y Permisos: Accesible por SuperAdmin y Admin, quienes pueden gestionar todos los proyectos. Los Programadores pueden ver y modificar los proyectos en los que están asignados.

Módulo de Gestión de Tareas:

Descripción: Permite la creación, seguimiento y finalización de tareas.

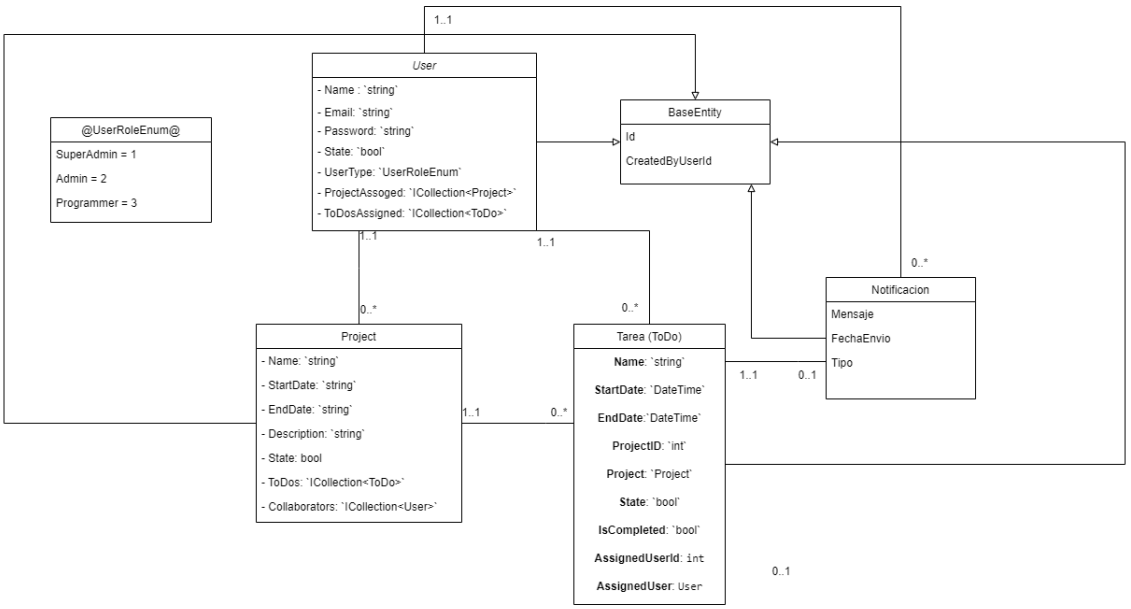
Funcionalidades:

- Creación, modificación, y eliminación de tareas.
- Asignación de tareas a usuarios.
- Notificaciones de vencimiento de tareas.

Interacciones: Interactúa con el módulo de gestión de proyectos para asignar tareas a proyectos y con el módulo de gestión de usuarios para asignar tareas a usuarios.

Roles y Permisos: Accesible por todos los usuarios, con diferentes niveles de permisos. Los SuperAdmin y Admin pueden gestionar todas las tareas, mientras que los Programadores pueden gestionar solo sus propias tareas.

DIAGRAMA DE CLASES



CURN: Gestión de Proyectos y Tareas en "Task Minder"

Nivel

- Estructura: Sin estructura
- Alcance: Negocio
- Caja: Negra
- Instanciación: Real
- Interacción: Semántica

Meta del Caso de Uso

El objetivo es que los proyectos, tareas y/o usuarios sean creados y asignados exitosamente.

Actores

- Primario: SuperAdmin
- Otros: Programador, Admin

Precondiciones (de negocio)

- El SuperAdmin debe estar autenticado en el sistema.

Disparador

- El SuperAdmin desea crear un nuevo usuario.
- El Admin desea Crear un Proyecto
- El Programer desea crear una ToDo

Flujo de Sucesos

Camino Básico para Crear un Usuario:

1. El SuperAdmin navega al módulo de usuarios en el sistema y selecciona la opción para crear un nuevo usuario.
2. El SuperAdmin ingresa los datos del usuario nombre, email, contraseña y rol, y guarda el usuario.
3. El sistema confirma que el usuario ha sido creado correctamente.

Camino Básico para Crear un Proyecto:

1. El Admin navega al módulo de gestión de proyectos en el sistema y selecciona la opción para crear un nuevo proyecto.
2. El Admin ingresa los detalles del proyecto: nombre, descripción, fecha de inicio, fecha de finalización y guarda el proyecto.
3. El sistema confirma que el proyecto ha sido creado correctamente.

Camino Básico para Crear una Tarea:

1. El Admin navega al módulo de gestión de tareas en el sistema y selecciona la opción para crear una nueva tarea.
2. El Admin ingresa los detalles de la tarea: título, descripción, fecha de vencimiento,selecciona al Programador al que desea asignar la tarea.Guarda la tarea.
3. El sistema confirma que la tarea ha sido creada y asignada correctamente.

Caminos Alternativos:

- 1.a. El Admin decide cancelar la creación del proyecto, tarea o usuario.
    - 1.a.1. El caso de uso termina sin crear el proyecto, tarea o usuario. Fin CU
- 1.b. El Admin no selecciona un Programador para una tarea.
    - 1.b.1. El sistema muestra un mensaje de error y solicita que se seleccione un Programador antes de continuar.
    - 1.b.2. El Admin selecciona un Programador y continúa con el paso 5.
- 1.c. El Admin ingresa detalles incompletos del proyecto, tarea o usuario.
    - 1.c.1. El sistema muestra un mensaje de error y solicita que se completen todos los campos obligatorios.
    - 1.c.2. El Admin completa los detalles y continúa con el paso 5.

Postcondiciones (de negocio)

- Éxito: El proyecto, tarea o usuario se creó y, en el caso de la tarea, se asignó al Programador correctamente.
- Fracaso: No se pudo crear el proyecto, tarea o usuario debido a datos incompletos o errores en el sistema.
- Éxito alternativo: El proyecto, tarea o usuario se creó después de corregir los datos incompletos y, en el caso de la tarea, seleccionar un Programador válido.

CURS: Gestión de Tareas en "Task Minder"

Nivel

- Estructura: Sin estructura
- Alcance: Sistema
- Caja: Negra
- Instanciación: Real
- Interacción: Semántica

Meta del Caso de Uso

El objetivo es que los proyectos, tareas y/o usuarios sean creados y asignados exitosamente.

Actores

- Primario: SuperAdmin,Programador, Admin
- Otros:

Precondiciones (de negocio)

- El SuperAdmin debe estar autenticado en el sistema.

Disparador

- El SuperAdmin desea crear un nuevo usuario.
- El Admin desea Crear un Proyecto
- El Programer desea crear una ToDo

Flujo de Sucesos

Camino Básico para Crear un Usuario:

1. El SuperAdmin ingresa los datos del usuario nombre, email, contraseña y rol, y guarda el usuario. El sistema registra

Camino Básico para Crear un Proyecto:

1. El Admin ingresa los detalles del proyecto: nombre, descripción, fecha de inicio, fecha de finalización y guarda el proyecto.EL sistema registra

Camino Básico para Crear una Tarea:

1. El Admin ingresa los detalles de la tarea: título, descripción, fecha de vencimiento,selecciona al Programador al que desea asignar la tarea.Guarda la tarea.El sistema registra

Caminos Alternativos:

- 1.a. El Admin decide cancelar la creación del proyecto, tarea o usuario.
    - 1.a.1. El sistema informa la cancelación. Fin del CU
- 1.b. El Admin no selecciona un Programador para una tarea.
    - 1.b.1. El sistema muestra un mensaje de error y solicita que se seleccione un Programador antes de continuar.
- 1.c. El Admin ingresa detalles incompletos del proyecto, tarea o usuario.
    - 1.c.1. El sistema muestra un mensaje de error y solicita que se completen todos los campos obligatorios.

Postcondiciones

- Éxito: El proyecto, tarea o usuario se registró en el sistema correctamente.
- Fracaso: El sistema no registró el proyecto, tarea o usuario debido a falta de datos o datos erróneos.

CURSR: Gestión de Tareas en "Task Minder"

Nivel

- **Estructura**: Reestructurado
- **Alcance**: Sistema
- **Caja**: Negra
- **Instanciación**: Real
- **Interacción**: Semántica

Meta del Caso de Uso

El objetivo es que los proyectos, tareas y/o usuarios sean creados y asignados exitosamente.

Actores:

- **Primario**: SuperAdmin,Programador, Admin
- **Otros**:

Precondiciones (de sistema):

El Usuario debe estar autenticado en el sistema.

Precondiciones (de negocio):

El SuperAdmin tiene la autorización para crear Usuarios.

El Admin tiene la autorización para crear y asignar proyectos.

Disparador:

El Usuario desea crear nuevos proyectos, tareas y/o usuarios.

Flujo de Sucesos:

Camino Básico para Crear un Usuario:

1. El SuperAdmin indica que quiere crear un usuario invocando el **CUU-1 Creacion de usuario**

Camino Básico para Crear un Proyecto:

1. El Admin indica que quiere crear un proyecto invocando el **CUU-2 Creación de Proyecto**

Camino Básico para Crear una Tarea:

1. El Programer indica que quiere crear una Tarea invocando el **CUU-3 Creación de Tarea**

Caminos Alternativos:

- <vacío>

Postcondiciones (de negocio)

- **Éxito**: El usuario, proyecto o tarea se creó correctamente.
- **Fracaso**: <vacío>
- **Éxito alternativo**: <vacío>

Postcondiciones (de sistema)

- **Éxito**: El sistema registra el usuario, proyecto o tarea.
- **Fracaso**: <vacío>
- **Éxito alternativo**: <vacío>

Casos de prueba

Caso de prueba Login:

Descripción:

Este caso de prueba verifica la funcionalidad del login de usuarios. Los usuarios deben ingresar con un correo electrónico y una contraseña registrados para acceder al sistema.

Escenarios de Prueba:

1. **Ingreso Exitoso:**
    - **Descripción:** Verificar que un usuario puede iniciar sesión correctamente con credenciales válidas.
    - **Entradas:**
        - Email: **`admin@gmail.com`**
        - Contraseña: **`123456`**
    - **Acciones:**
        - Ingresar el email **`admin@gmail.com`** en el campo de correo electrónico.
        - Ingresar la contraseña **`123456`** en el campo de contraseña.
        - Hacer clic en el botón de login.
    - **Resultados Esperados:**
        - El usuario es redirigido a la página principal o al dashboard.
2. **Email Inválido:**
    - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando se ingresa un email con formato incorrecto.
    - **Entradas:**
        - Email: **`usuario@gmail.com`**
        - Contraseña: **`123456`**
    - **Acciones:**
        - Ingresar el email **`usuario@gmail.com`** en el campo de correo electrónico.
        - Ingresar la contraseña **`123`** en el campo de contraseña.
        - Hacer clic en el botón de login.
    - **Resultados Esperados:**
        - El sistema muestra un mensaje de error indicando que el formato del correo electrónico es incorrecto.
        - El usuario no puede proceder al inicio de sesión.
3. **Contraseña Vacía:**
    - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando el campo de contraseña está vacío.

- **Entradas:**
  - Email: **`usuario@gmail.com`**
  - Contraseña: **<vacio>**
- **Acciones:**
  - Ingresar el email **`usuario@gmail.com`** en el campo de correo electrónico.
  - Dejar el campo de contraseña vacío.
  - Hacer clic en el botón de login.
- **Resultados Esperados:**
  - El sistema muestra un mensaje de error indicando que el campo de contraseña no puede estar vacío.
  - El usuario no puede proceder al inicio de sesión.

4. **Email No Registrado:**

- **Descripción:** Verificar que el sistema muestra un mensaje de error cuando se ingresa un email no registrado.
- **Entradas:**
  - Email: **`usuario@gmail.com`**
  - Contraseña: **123124**
- **Acciones:**
  - Ingresar el email **`usuario@gmail.com`** en el campo de correo electrónico.
  - Ingresar la contraseña **123456** en el campo de contraseña.
  - Hacer clic en el botón de login.
- **Resultados Esperados:**
  - El sistema muestra un mensaje de error indicando que el correo electrónico no está registrado.
  - El usuario no puede proceder al inicio de sesión.

5. **Contraseña Incorrecta:**

- **Descripción:** Verificar que el sistema muestra un mensaje de error cuando se ingresa una contraseña incorrecta.
- **Entradas:**
  - Email: **`admin@gmail.com`**
  - Contraseña: **123**
- **Acciones:**
  - Ingresar el email **`admin@gmail.com`** en el campo de correo electrónico.
  - Ingresar la contraseña **123** en el campo de contraseña.
  - Hacer clic en el botón de login.
- **Resultados Esperados:**
  - El sistema muestra un mensaje de error indicando que la contraseña es incorrecta.
  - El usuario no puede proceder al inicio de sesión.

Caso de Prueba para la Creación de Proyectos

Descripción:

Este caso de prueba verifica la funcionalidad de crear nuevos proyectos. Los usuarios deben poder ingresar los detalles necesarios para crear un proyecto, incluyendo el nombre, fecha de inicio, fecha de fin, creado por y descripción.

Precondiciones:

1. El sistema debe estar en el dashboard o en la página de creación de proyectos.
2. Los campos de nombre, fecha de inicio, fecha de fin, creado por y descripción deben estar presentes en la página.

Escenarios de Prueba:

1. **Creación Exitosa de Proyecto:**
    - **Descripción:** Verificar que un usuario puede crear un nuevo proyecto correctamente con todos los campos válidos.
    - **Entradas:**
        - Nombre: **`Proyecto Desarrollar App`**
        - Fecha de Inicio: **`2024-07-01`**
        - Fecha de Fin: **`2024-12-31`**
        - Creado por: **`admin@example.com`**
        - Descripción: **`Desarrollo de una aplicación web completa`**
    - **Acciones:**
        - Ingresar el nombre **`Proyecto Desarrollar App`** en el campo de nombre.
        - Ingresar la fecha de inicio **`2024-07-01`** en el campo de fecha de inicio.
        - Ingresar la fecha de fin **`2024-12-31`** en el campo de fecha de fin.
        - Ingresar **`admin@example.com`** en el campo de creado por.
        - Ingresar la descripción **`Desarrollo de una aplicación web completa`** en el campo de descripción.
        - Hacer clic en el botón de agregar proyecto.
    - **Resultados Esperados:**
        - El nuevo proyecto se muestra en la lista de proyectos.
        - El usuario recibe una confirmación de que el proyecto ha sido creado exitosamente.

2. **Nombre de Proyecto Vacío:**
    - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando el campo de nombre está vacío.

- **Entradas:**
  - Nombre: (vacío)
  - Fecha de Inicio: **`2024-07-01`**
  - Fecha de Fin: **`2024-12-31`**
  - Creado por: **`admin@example.com`**
  - Descripción: **`Desarrollo de una aplicación web completa`**
- **Acciones:**
  - Dejar el campo de nombre vacío.
  - Ingresar la fecha de inicio **`2024-07-01`** en el campo de fecha de inicio.
  - Ingresar la fecha de fin **`2024-12-31`** en el campo de fecha de fin.
  - Ingresar **`admin@example.com`** en el campo de creado por.
  - Ingresar la descripción **`Desarrollo de una aplicación web completa`** en el campo de descripción.
  - Hacer clic en el botón de agregar proyecto.
- **Resultados Esperados:**
  - El sistema muestra un mensaje de error indicando que el campo de nombre no puede estar vacío.
  - El proyecto no se agrega.

3. **Fecha de Fin Anterior a Fecha de Inicio:**

   - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando la fecha de fin es anterior a la fecha de inicio.
   - **Entradas:**
     - Nombre: **`Proyecto Desarrollar App`**
     - Fecha de Inicio: **`2024-12-31`**
     - Fecha de Fin: **`2024-07-01`**
     - Creado por: **`admin@example.com`**
     - Descripción: **`Desarrollo de una aplicación web completa`**
   - **Acciones:**
     - Ingresar el nombre **`Proyecto Desarrollar App`** en el campo de nombre.
     - Ingresar la fecha de inicio **`2024-12-31`** en el campo de fecha de inicio.
     - Ingresar la fecha de fin **`2024-07-01`** en el campo de fecha de fin.
     - Ingresar **`admin@example.com`** en el campo de creado por.
     - Ingresar la descripción **`Desarrollo de una aplicación web completa`** en el campo de descripción.
     - Hacer clic en el botón de agregar proyecto.
   - **Resultados Esperados:**

        - El sistema muestra un mensaje de error indicando que la fecha de fin no puede ser anterior a la fecha de inicio.
        - El proyecto no se agrega.

4. **Todos los Campos Vacíos:**

    - **Descripción:** Verificar que el sistema muestra mensajes de error adecuados cuando todos los campos están vacíos.
    - **Entradas:**
        - Nombre: (vacío)
        - Fecha de Inicio: (vacío)
        - Fecha de Fin: (vacío)
        - Creado por: (vacío)
        - Descripción: (vacío)
    - **Acciones:**
        - Dejar todos los campos vacíos.
        - Hacer clic en el botón de agregar proyecto.
    - **Resultados Esperados:**
        - El sistema muestra mensajes de error indicando que todos los campos son obligatorios.
        - El proyecto no se agrega.

Caso de Prueba para la Agregación de Tareas

Precondiciones:

1. El sistema debe estar en el dashboard o en la página de creación de tareas.
2. Los campos de nombre, fecha de inicio, fecha de fin, creado por y descripción deben estar presentes en la página.
3. Debe haber proyectos creados y elegir un proyecto al cual agregar una tarea

Descripción:

Este caso de prueba verifica la funcionalidad de agregar nuevas tareas. Los usuarios deben poder ingresar los detalles necesarios para crear una tarea, incluyendo el nombre, fecha de inicio, fecha de fin, creador, usuario asignado y estado de completado.

Escenarios de Prueba:

1. **Agregación Exitosa de Tarea:**

    - **Descripción:** Verificar que un usuario puede agregar una nueva tarea correctamente con todos los campos válidos.
    - **Entradas:**
        - Nombre: **`Desarrollar módulo de login`**
        - Fecha de Inicio: **`2024-07-01`**
        - Fecha de Fin: **`2024-07-15`**
        - Creado por: **`admin@example.com`**

        - Usuario Asignado: **`user@example.com`**
        - Completado: **`false`**
    - **Acciones:**
        - Ingresar el nombre **`Desarrollar módulo de login`** en el campo de nombre.
        - Ingresar la fecha de inicio **`2024-07-01`** en el campo de fecha de inicio.
        - Ingresar la fecha de fin **`2024-07-15`** en el campo de fecha de fin.
        - Ingresar **`admin@example.com`** en el campo de creado por.
        - Ingresar **`user@example.com`** en el campo de usuario asignado.
        - Seleccionar **`false`** en el campo de completado.
        - Hacer clic en el botón de agregar tarea.
    - **Resultados Esperados:**
        - La nueva tarea se muestra en la lista de tareas.
        - El usuario recibe una confirmación de que la tarea ha sido creada exitosamente.

2. **Fecha de Fin Anterior a Fecha de Inicio:**

    - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando la fecha de fin es anterior a la fecha de inicio.
    - **Entradas:**
        - Nombre: **`Desarrollar módulo de login`**
        - Fecha de Inicio: **`2024-07-15`**
        - Fecha de Fin: **`2024-07-01`**
        - Creado por: **`admin@example.com`**
        - Usuario Asignado: **`user@example.com`**
        - Completado: **`false`**
    - **Acciones:**
        - Ingresar el nombre **`Desarrollar módulo de login`** en el campo de nombre.
        - Ingresar la fecha de inicio **`2024-07-15`** en el campo de fecha de inicio.
        - Ingresar la fecha de fin **`2024-07-01`** en el campo de fecha de fin.
        - Ingresar **`admin@example.com`** en el campo de creado por.
        - Ingresar **`user@example.com`** en el campo de usuario asignado.
        - Seleccionar **`false`** en el campo de completado.
        - Hacer clic en el botón de agregar tarea.
    - **Resultados Esperados:**
        - El sistema muestra un mensaje de error indicando que la fecha de fin no puede ser anterior a la fecha de inicio.
        - La tarea no se agrega.

3. **Usuario Asignado No Registrado:**
    - **Descripción:** Verificar que el sistema muestra un mensaje de error cuando se ingresa un usuario asignado que no está registrado.
    - **Entradas:**
        - Nombre: **`Desarrollar módulo de login`**
        - Fecha de Inicio: **`2024-07-01`**
        - Fecha de Fin: **`2024-07-15`**
        - Creado por: **`admin@example.com`**
        - Usuario Asignado: **`nonexistent@example.com`**
        - Completado: **`false`**
    - **Acciones:**
        - Ingresar el nombre **`Desarrollar módulo de login`** en el campo de nombre.
        - Ingresar la fecha de inicio **`2024-07-01`** en el campo de fecha de inicio.
        - Ingresar la fecha de fin **`2024-07-15`** en el campo de fecha de fin.
        - Ingresar **`admin@example.com`** en el campo de creado por.
        - Ingresar **`nonexistent@example.com`** en el campo de usuario asignado.
        - Seleccionar **`false`** en el campo de completado.
        - Hacer clic en el botón de agregar tarea.
    - **Resultados Esperados:**
        - El sistema muestra un mensaje de error indicando que el usuario asignado no está registrado.
        - La tarea no se agrega.

4. **Todos los Campos Vacíos:**
    - **Descripción:** Verificar que el sistema muestra mensajes de error adecuados cuando todos los campos están vacíos.
    - **Entradas:**
        - Nombre: (vacío)
        - Fecha de Inicio: (vacío)
        - Fecha de Fin: (vacío)
        - Creado por: (vacío)
        - Usuario Asignado: (vacío)
        - Completado: (vacío)
    - **Acciones:**
        - Dejar todos los campos vacíos.
        - Hacer clic en el botón de agregar tarea.
    - **Resultados Esperados:**
        - El sistema muestra mensajes de error indicando que todos los campos son obligatorios.
        - La tarea no se agrega.

Descripción Detallada de la Arquitectura Utilizada en el Proyecto "Task Minder"

La arquitectura del sistema "Task Minder" está diseñada para ser escalable y eficiente, utilizando tecnologías modernas para proporcionar una gestión efectiva de tareas y proyectos. La arquitectura se divide en cuatro componentes principales:

1. Frontend

- **Tecnología**: React
- **Descripción**: La interfaz de usuario que permite la interacción directa con el sistema. Incluye funcionalidades como:
  - Autenticación y gestión de sesiones.
  - Creación, modificación y visualización de proyectos y tareas.
  - Interacción con notificaciones.
- **Flujo de Trabajo**:
  - Los usuarios ingresan al sistema enviando su email y contraseña al módulo de autenticación.
  - Reciben un token de autenticación que se utiliza para realizar solicitudes al backend.
  - El frontend puede recuperar el usuario logueado enviando el token en cualquier momento.

2. Backend

- **Tecnología**: .NET
- **Descripción**: El backend gestiona las solicitudes de datos realizadas por el frontend, interactuando con la base de datos y el módulo de autenticación para:
  - Validar tokens de autenticación.
  - Procesar y responder a las solicitudes CRUD (Crear, Leer, Actualizar, Eliminar) para usuarios, proyectos, y tareas.
  - Manejar la lógica del negocio y las reglas de la aplicación.

3. Base de Datos

- **Tecnología**: MySQL
- **Descripción**: La base de datos almacena toda la información relevante del sistema, incluyendo usuarios, roles, proyectos, tareas, y notificaciones.
- **Interacción**:
  - El backend realiza consultas a la base de datos para recuperar o actualizar la información según las solicitudes recibidas.
  - La base de datos envía los datos solicitados o un error en caso de fallo en la consulta.

4. **Módulo de Autenticación**
    - **Descripción**: Maneja la autenticación de usuarios y la generación de tokens. Utiliza una base de datos para verificar las credenciales de los usuarios.
    - **Interacción**:
        - Al iniciar sesión, el módulo de autenticación envía el email y la contraseña del usuario a la base de datos.
        - La base de datos devuelve los datos del usuario o un error.
        - El módulo de autenticación genera un token de autenticación que se envía al frontend.

Compatibilidad y versiones

Descripción del Proyecto

Este proyecto está desarrollado utilizando .NET 8.0 y tiene como objetivo implementar un backend robusto y escalable para manejar las operaciones de negocio. El proyecto incluye diversas funcionalidades que son soportadas por múltiples bibliotecas y paquetes.

Configuración del Proyecto

El proyecto utiliza el SDK de Microsoft para aplicaciones web, asegurando que se cumplan los estándares más recientes y eficientes en el desarrollo de aplicaciones web.

Propiedades Principales

- **Target Framework:** net8.0
- **Nullable:** enable
- **Implicit Usings:** enable

Dependencias del Backend

El proyecto está soportado por varias dependencias clave que proporcionan funcionalidades críticas:

- **ErrorOr** (v2.0.1): Manejo de errores.
- **Microsoft.AspNetCore.Authentication.JwtBearer** (v8.0.6): Autenticación basada en JWT.
- **Microsoft.AspNetCore.Cors** (v2.2.0): Soporte para CORS.
- **Microsoft.EntityFrameworkCore** (v8.0.6): ORM para manejar bases de datos.
- **Microsoft.EntityFrameworkCore.Design** (v8.0.6): Herramientas de diseño para EF Core.
- **Microsoft.EntityFrameworkCore.SqlServer** (v8.0.6): Proveedor de base de datos SQL Server para EF Core.

- **Microsoft.EntityFrameworkCore.Tools** (v8.0.6): Herramientas de desarrollo para EF Core.
- **Microsoft.IdentityModel.Tokens** (v7.6.2): Tokens de seguridad.
- **Quartz.Extensions.DependencyInjection** (v3.10.0) y **Quartz.Extensions.Hosting** (v3.10.0): Programación de trabajos en segundo plano.
- **Swashbuckle.AspNetCore** (v6.6.2): Documentación de API con Swagger.
- **System.IdentityModel.Tokens.Jwt** (v7.6.2): Manejo de tokens JWT.
- **Volo.Abp.BackgroundJobs.Abstractions** (v8.2.0): Abstracciones para trabajos en segundo plano.
- **xunit** (v2.8.1) y **xunit.assert** (v2.8.1): Framework de pruebas unitarias.
- **Moq** (v4.20.70): Framework de creación de mocks para pruebas unitarias.

Dependencias del Frontend

El frontend del proyecto está desarrollado en React y cuenta con las siguientes dependencias:

- **@fortawesome/fontawesome-svg-core** (v6.5.1)
- **@fortawesome/free-solid-svg-icons** (v6.5.1)
- **@fortawesome/react-fontawesome** (v0.2.0)
- **@testing-library/jest-dom** (v5.17.0)
- **@testing-library/react** (v13.4.0)
- **@testing-library/user-event** (v13.5.0)
- **axios** (v1.7.2)
- **bootstrap** (v5.3.2)
- **i** (v0.3.7)
- **jwt-decode** (v4.0.0)
- **react-bootstrap** (v2.9.1)
- **react-dom** (v18.2.0)
- **react-npm** (v2.6.1)
- **react-router-dom** (v6.20.1)
- **react-router** (v6.20.1)
- **react-scripts** (v5.0.1)
- **react-toastify** (v9.1.3)
- **react** (v18.2.0)
- **styled-components** (v6.1.1)
- **web-vitals** (v2.1.4)

Base de Datos

El proyecto utiliza Microsoft SQL Server 2019:

- **Versión de SQL Server:** Microsoft SQL Server 2019 (RTM-GDR) (KB5035434) - 15.0.2110.4 (X64)
- **Sistema Operativo:** Windows 10 Pro 10.0 (Build 19045)

Gestión de Errores

Para mantener el código limpio y manejable, se han eliminado componentes innecesarios:

- Se ha eliminado el directorio **ServiceErrors**.
- Se ha eliminado el archivo **Data\Models\ProjectPostDto.cs**.

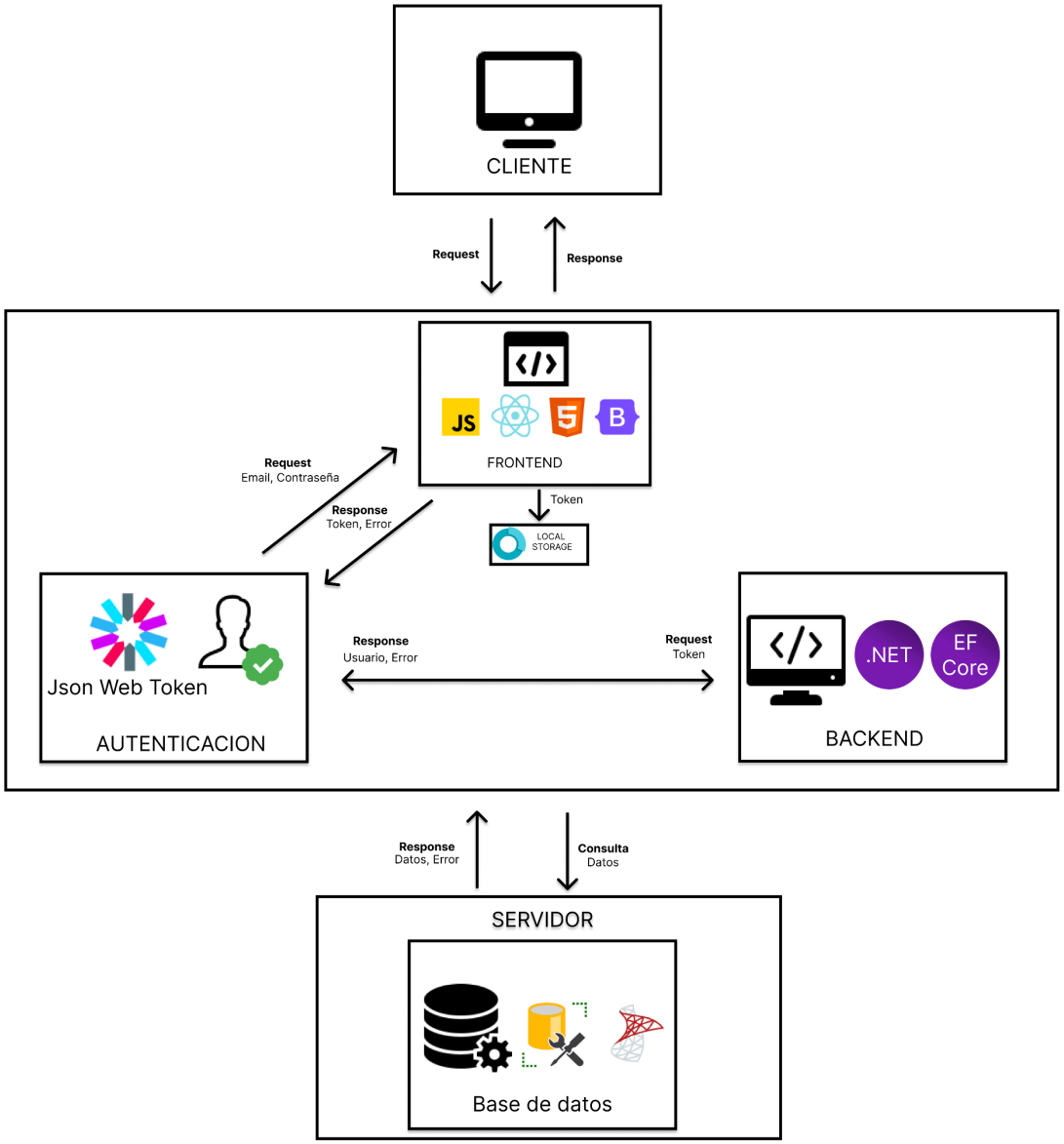
Consideraciones de Seguridad

El proyecto incluye autenticación JWT para asegurar las operaciones y garantizar que solo usuarios autorizados puedan acceder a los recursos.

Manejo de Base de Datos

Se utiliza Entity Framework Core como ORM principal, con soporte para SQL Server.

Diagramas de Arquitectura



1. Diagrama de Arquitectura General

Este diagrama ilustra la estructura general del sistema, mostrando cómo interactúan los diferentes componentes (Frontend, Backend, Base de Datos, y Módulo de Autenticación).

2. Diagrama de Componentes

Describe los módulos específicos del sistema y sus interacciones, como la gestión de usuarios, proyectos y tareas.

3. Diagrama de Secuencia

Ejemplifica el flujo de un proceso típico, como la creación de una nueva tarea, desde la solicitud del usuario en el frontend hasta la actualización en la base de datos.

4. Diagrama de Despliegue

Describe cómo se despliega el sistema en el entorno de producción, incluyendo servidores web y de aplicaciones, balanceadores de carga, y configuración de la base de datos.

Levantar el Proyecto

Describe los pasos necesarios para clonar y levantar el proyecto en tu entorno de desarrollo. Asegúrate de seguir estos pasos cuidadosamente para poder ejecutar el proyecto sin problemas.

Paso 1: Clonar el Repositorio

Para comenzar, clona el repositorio en tu máquina local utilizando el siguiente comando en tu terminal:

```
git clone https://github.com/FrancoExeqGarcia/TP-PPS.git
```

Paso 2: Configuración e instalación del Sql Server

Descargar el Sql Server Express 2019:

Copiar código

```
https://www.microsoft.com/es-ar/download/details.aspx?id=101064
```

Una vez instalado, levantas el servicio en el Administrador de configuración de SQL Server 2019 y abres el SqlServer Management Studio, conectas con tu configuración definida en la instalación del SqlServer Express.

Te diriges a properties de la base de datos, database settings y editas las direcciones data, log, backup:

- **Data**: [donde clonaste el proyecto]\TP-PPS\Base de Datos\MSSQL15.TASKMANAGER\MSSQL\DATA\
- **Log**: [donde clonaste el proyecto]\TP-PPS\Base de Datos\MSSQL15.TASKMANAGER\MSSQL\Logs\
- **Backup**: [donde instalaste SqlServer] generalmente C:\Program Files\Microsoft SQL Server\MSSQL15.LOCAL\MSSQL\Backup

Paso 3: Levantar el Frontend

Una vez que hayas clonado el repositorio, navega a la carpeta Frontend:

Dentro de la carpeta Frontend, instala las dependencias del proyecto con npm:

npm i

Una vez que se hayan instalado las dependencias, inicia el servidor de desarrollo con el siguiente comando:

npm start

Esto pondrá en marcha la parte frontend de la aplicación y podrás acceder a ella en tu navegador visitando http://localhost:3000.

Paso 4: Levantar el Backend

Dentro de la carpeta de Backend, inicia el servicio con el siguiente comando:

dotnet run

Esto pondrá en marcha la parte backend de la aplicación.

Ahora puedes probar la aplicación.

PD: Podrás acceder al swagger en tu navegador visitando https://localhost:7165/swagger.

Si tienes alguna pregunta o enfrentas problemas, no dudes en consultar con el equipo de desarrollo. ¡Feliz codificación!

Conclusión

La arquitectura de "Task Minder" está diseñada para ser robusta y flexible, permitiendo una gestión eficiente de proyectos y tareas mediante el uso de tecnologías modernas y una estructura clara y bien definida. La implementación de esta arquitectura garantiza que el sistema sea escalable y adaptable a las necesidades cambiantes de los usuarios y organizaciones.

Este sistema optimiza la gestión de tareas y proyectos, adecuándose a las necesidades específicas de los diferentes roles dentro de una organización, ofreciendo una solución completa y personalizable para la administración de proyectos y tareas. La aplicación está diseñada para ser altamente personalizable y escalable, permitiendo a la organización adaptarla a sus necesidades específicas, brindando una gestión de proyectos de manera eficiente y efectiva.