



Asignatura

DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA DPS941

Docente

Ing. Alexander Sigüenza

PRIMER PROYECTO EN REACT 20%

Integrantes	Carnet
Bolaños Mancía, Manuel Antonio	BM192191
Figueroa Aguilar, José Manuel	FA200209
Hernandez Soriano, Rene Alexander	HS191498
Muños Reyes, Christopher Alberto	MR202832
Montoya Reyes, Cristian Alberto	MR211303
Senora Reyes, Jonathan Rafael	SR232918

Enlace al proyecto Git Hub:

<https://github.com/manuelbmaa/management-system/tree/main>

Enlace al proyecto alojado en Vercel:

<https://management-system-manuelbmaa-manuels-projects-f39b4c36.vercel.app/login>

Introducción

En el presente manual, se desarrolla una guía paso a paso a través del proceso de creación de una aplicación web utilizando MERN, se mostrará como configurar el entorno de desarrollo, como iniciar un proyecto con React.js y a conectarlo con MongoDB.

El MERN stack, compuesto por MongoDB, Express.js, React.js y Node.js, se ha convertido en una elección popular entre los desarrolladores web modernos. Esta combinación de tecnologías ofrece una solución completa y eficiente para construir aplicaciones web robustas y escalables.

Este manual es una guía a través del proceso de ejecutar una aplicación desarrollada con el stack MERN (MongoDB, Express.js, React.js, Node.js) directamente desde Visual Studio Code.

Ejecución de la aplicación web

Requisitos previos

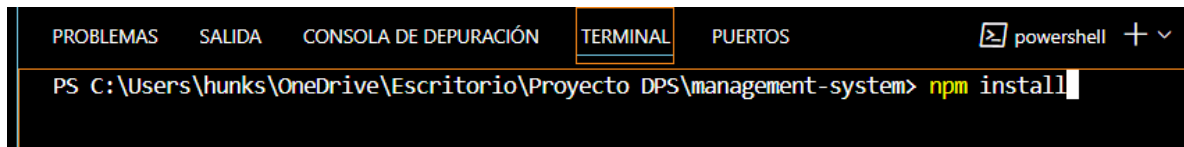
- Visual Studio Code instalado y configurado.
- Node.js y npm (o yarn) instalados globalmente.
- Una aplicación MERN ya desarrollada en tu entorno local.

Pasos

1. Abrir el proyecto en Visual Studio Code:
 - Abre Visual Studio Code y selecciona "Abrir carpeta" (o "Open Folder").
 - Navega hasta la carpeta raíz de tu proyecto MERN y selecciónala.
2. Instalar las dependencias:
 - Abre una terminal integrada en Visual Studio Code (puedes hacerlo presionando Ctrl+` o yendo al menú "Terminal" -> "Nueva terminal").



- Ejecuta el siguiente comando para instalar las dependencias de tu proyecto:

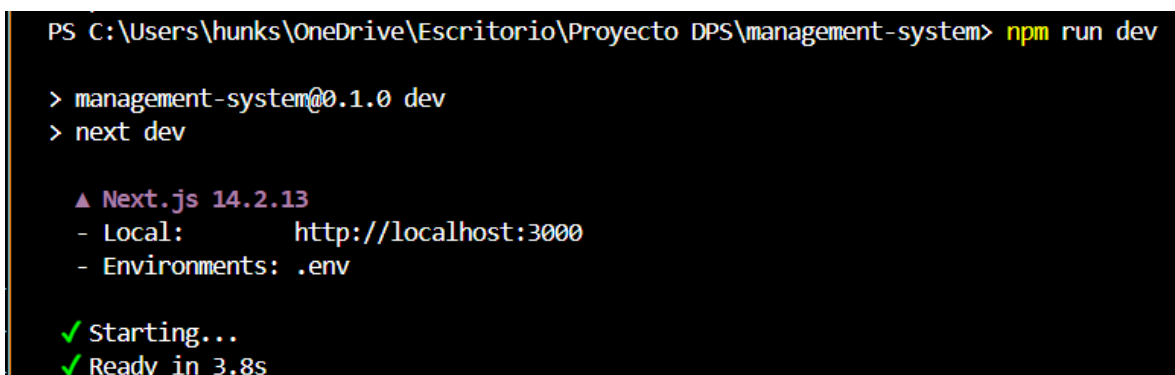


```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS  powershell + v
PS C:\Users\hunks\OneDrive\Escritorio\Proyecto DPS\management-system> npm install
```

3. Iniciar el servidor de desarrollo:

- **Iniciar el servidor de desarrollo:** Navega hasta la carpeta raíz de tu proyecto MERN en la terminal.

Ejecuta el siguiente comando:



```
PS C:\Users\hunks\OneDrive\Escritorio\Proyecto DPS\management-system> npm run dev

> management-system@0.1.0 dev
> next dev

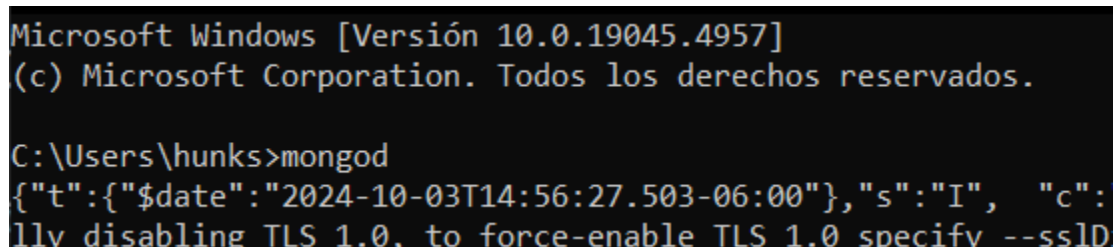
▲ Next.js 14.2.13
- Local:      http://localhost:3000
- Environments: .env

✓ Starting...
✓ Ready in 3.8s
```

4. Acceder a la aplicación:

- Una vez que ambos servidores estén en ejecución, abre un navegador y ve a la dirección indicada en la terminal (generalmente <http://localhost:3000>).

Antes de iniciar la aplicación debes asegurarte de estar conectado al servidor MongoDB, para poder levantar la aplicación web, en el caso de la imagen con el comando: “mongod” ponemos en función la bd.



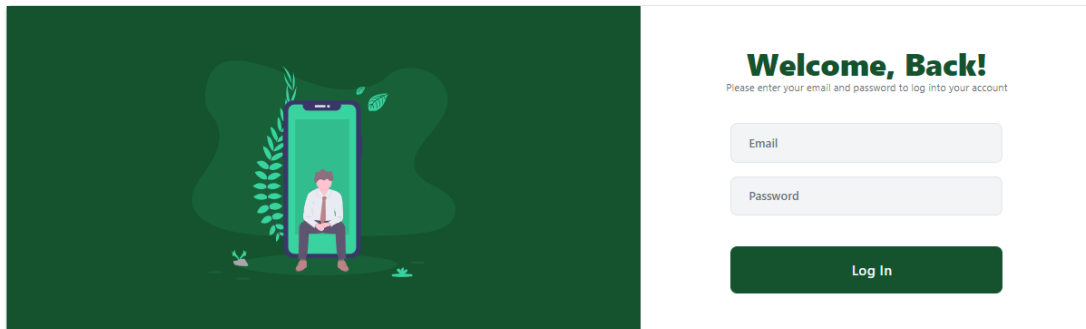
```
Microsoft Windows [Versión 10.0.19045.4957]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\hunks>mongod
{"t":{"$date":"2024-10-03T14:56:27.503-06:00"},"s":"I", "c":
lly disabling TLS 1.0, to force-enable TLS 1.0 specify --sslD
```

También es importante que antes de iniciar tu aplicación tengas las configuraciones correctas con la base de datos correspondiente, identificándola con su nombre, puerto de conexión y contraseña, en el caso del ejercicio esta es la configuración que se desarrolló:

```
⚙️ .env
1 MONGODB_CONTAINER_NAME=mongodps1
2 MONGODB_DATABASE_NAME=dps1
3 MONGODB_URI= mongodb://localhost/management-system
4 NEXTAUTH_URL= http://localhost:3000
5 NEXTAUTH_SECRET=0000
```

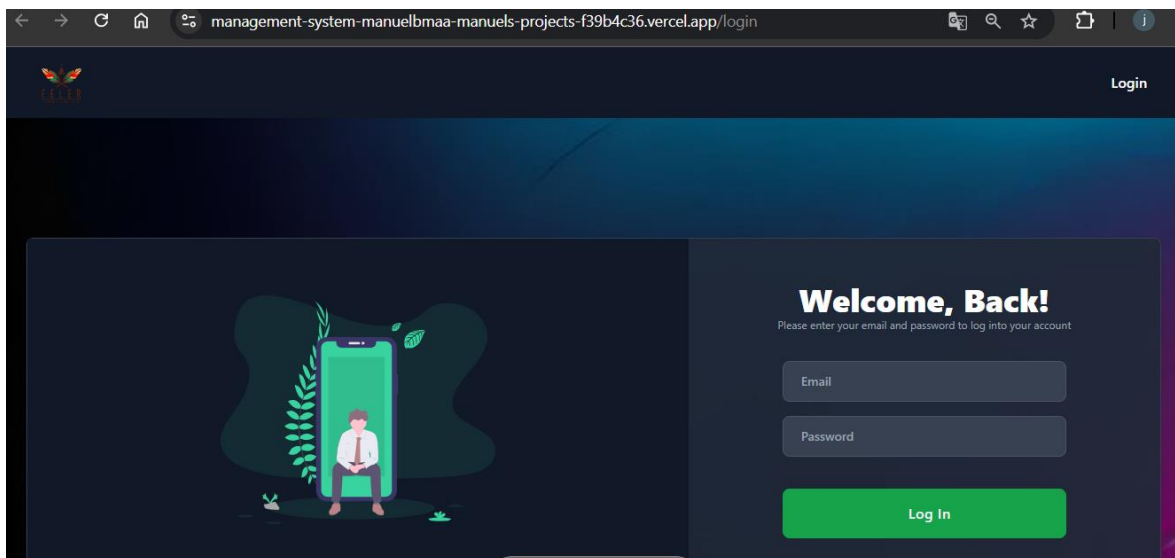
Al ingresar al puerto del servidor en el navegador podremos visualizar el inicio de la aplicación web y empezar a probar funcionalidades, también en caso de realizar cambios en el código se estarán realizando en tiempo real en el servidor, lo que aumenta la eficacia ya que no es necesario estarlo levantando continuamente.



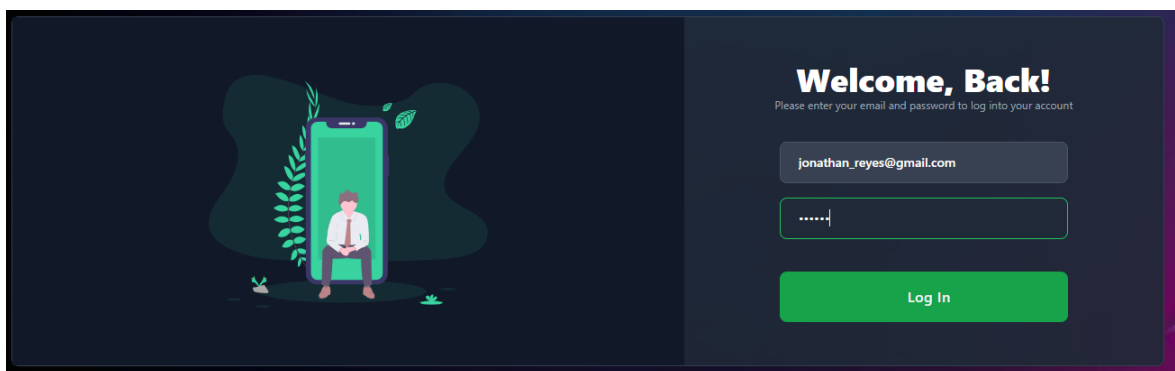
Funcionamiento de la aplicación web alojada en Vercel

CREDENCIALES DE SESION			
Nº	Team	Correo Electrónico	Contraseña
1	Member	jonathan_reyes@gmail.com	123456
2	Member	jose_manuel@gmail.com	123456
3	Manager	rene_hernandez@gmail.com	123456
4	Manager	cristian_montonya@gmail.com	123456
5	Admin	christopher_reyes@gmail.com	123456
6	Admin	manuel_marcia@gmail.com	123456

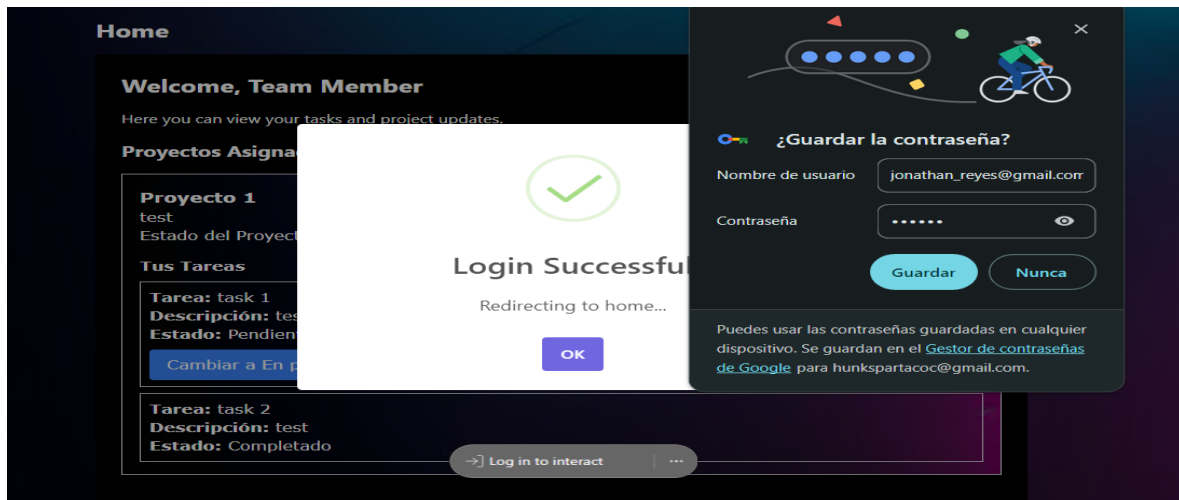
Pestaña de inicio



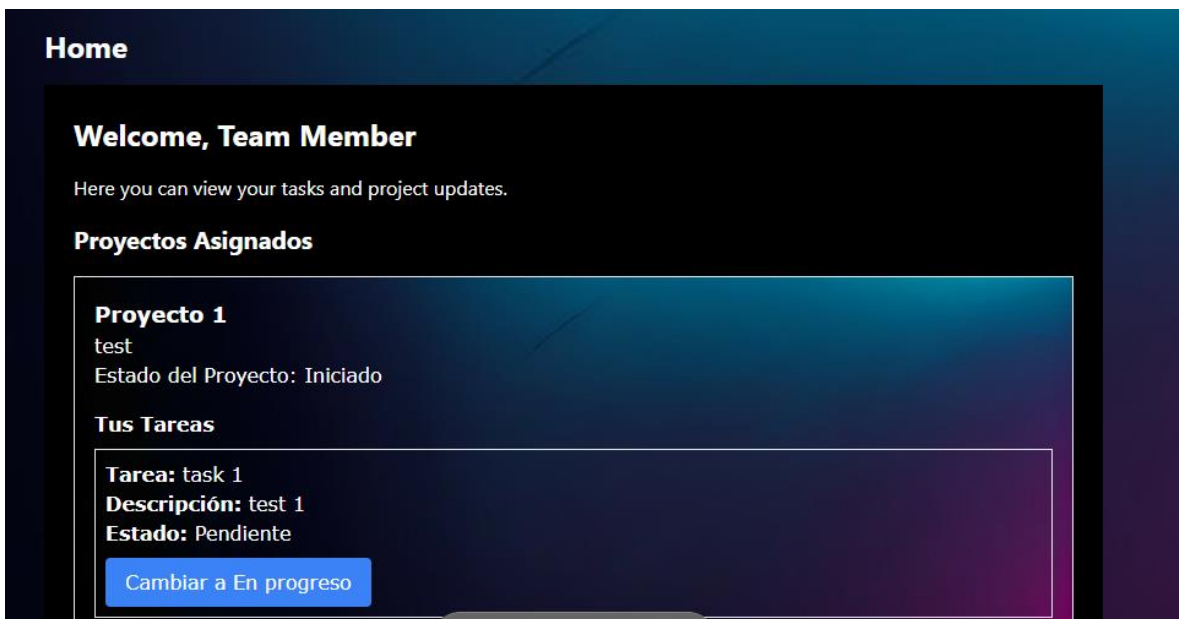
Recibe el formato asignado en correo electrónico y la contraseña en modo seguro.



Verificación de contraseña en la base de datos y validación correspondiente.



Pantalla TeamMember únicamente puede revisar tareas asignadas y cambiar el estatus de la misma.



Pantalla del gestor de proyectos, sus funcionalidades son asignar proyectos y delegar tareas a los miembros del team almacenados en la base de datos, así como eliminación, modificación y actualización de tareas.

Home

Home Dashboard Log out

Gestión de Proyectos

Nombre del proyecto

Descripción del proyecto

Crear Proyecto

Proyectos Actuales

Proyecto 1
test
Estado: Iniciado

Eliminar Proyecto Editar Proyecto

Asignar Tarea

Nombre de la tarea

Descripción de la tarea

Seleccione un miembro para la tarea

Asignar Tarea

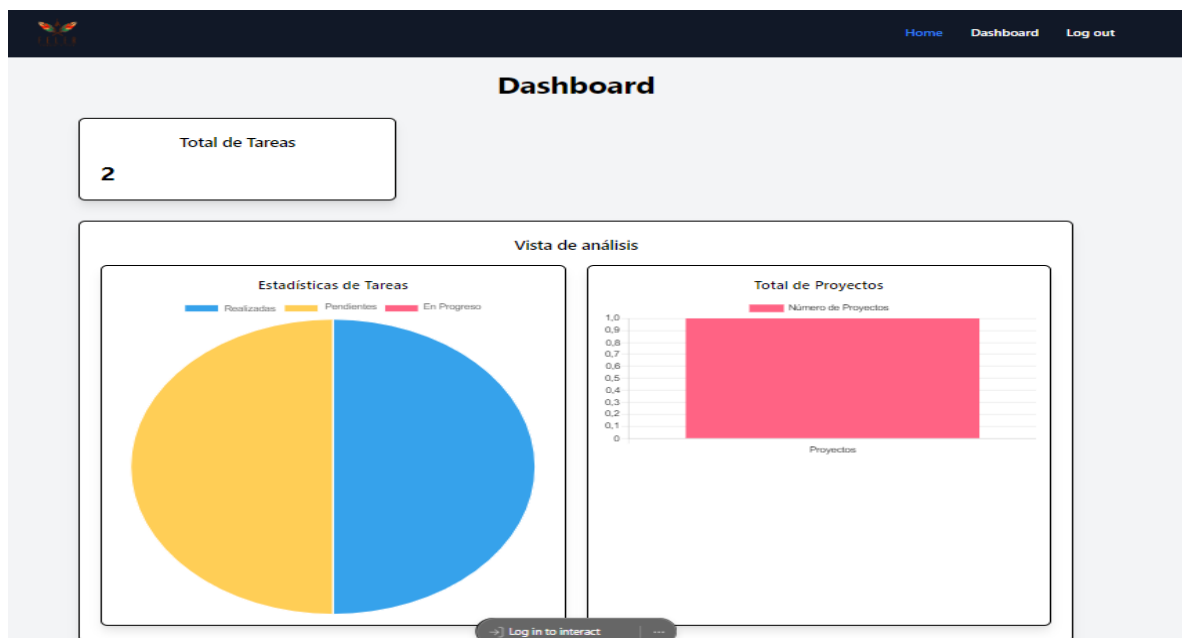
Tareas

Tareas task 1
Descripción: test 1
Asignado a: Jonathan Rafael Señora Reyes
Estado: Pendiente

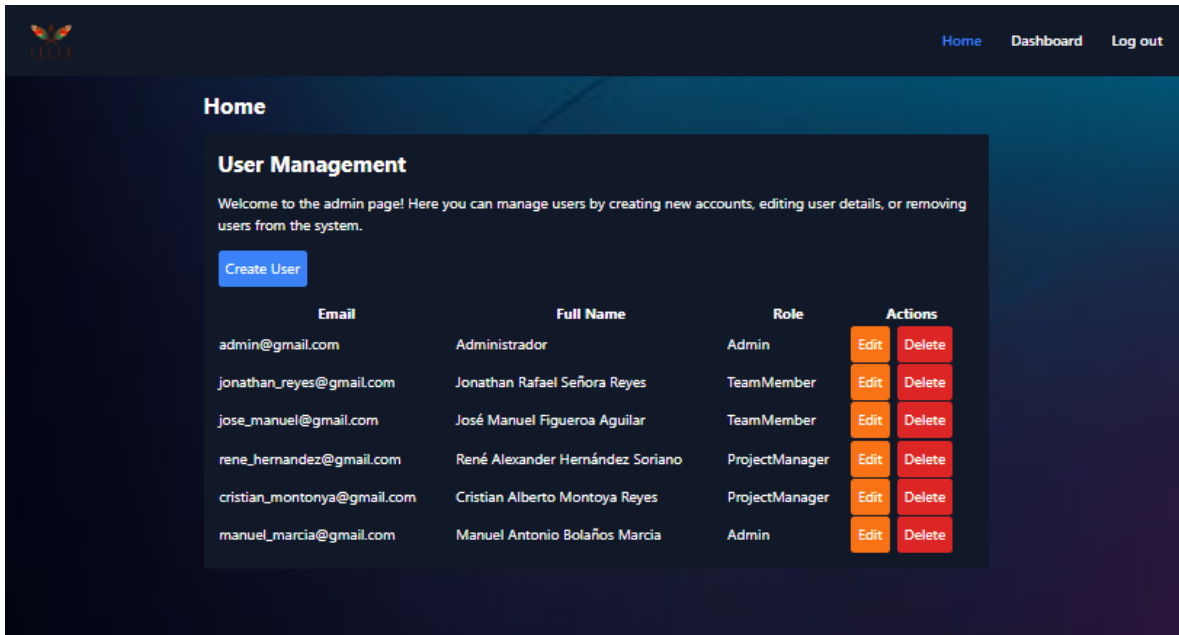
Editar Tarea Eliminar Tarea

Log in to interact

Dentro de este rol incluye una pestaña Dashboard para visualizar estatus de tareas asignadas y sus totales, para una correcta toma de decisiones.



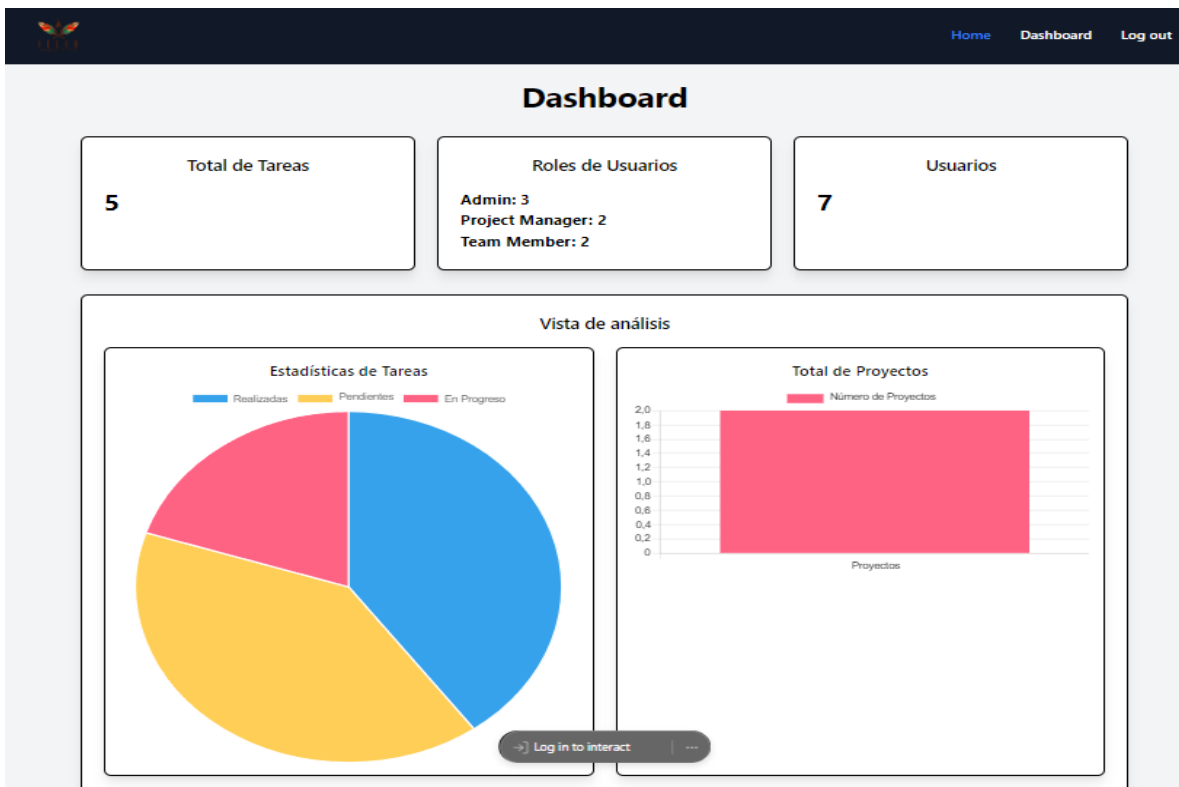
Se muestra la pantalla principal del administrador, este rol puede administrar todos los usuarios existentes y realizar acciones CRUD a los mismos.



The screenshot shows the 'Home' page of an application with a 'User Management' section. The page has a dark blue header with a logo on the left and navigation links 'Home', 'Dashboard', and 'Log out' on the right. The 'User Management' section includes a welcome message, a 'Create User' button, and a table of users.

Email	Full Name	Role	Actions
admin@gmail.com	Administrador	Admin	Edit Delete
jonathan_reyes@gmail.com	Jonathan Rafael Señora Reyes	TeamMember	Edit Delete
jose_manuel@gmail.com	José Manuel Figueroa Aguilar	TeamMember	Edit Delete
rene_hernandez@gmail.com	René Alexander Hernández Soriano	ProjectManager	Edit Delete
cristian_montonya@gmail.com	Cristian Alberto Montoya Reyes	ProjectManager	Edit Delete
manuel_marcia@gmail.com	Manuel Antonio Bolaños Marcia	Admin	Edit Delete

Igualmente posee su dashboard correspondiente a la información de usuarios (Esta información solo la puede ver este usuario) y las tareas asignadas a todos los proyectos.



The screenshot shows the 'Dashboard' page of the application. It features a light blue header with the same navigation links as the previous page. The dashboard includes three summary cards at the top: 'Total de Tareas' (5), 'Roles de Usuarios' (Admin: 3, Project Manager: 2, Team Member: 2), and 'Usuarios' (7). Below these is a 'Vista de análisis' section containing two charts: a pie chart for 'Estadísticas de Tareas' and a bar chart for 'Total de Proyectos'.

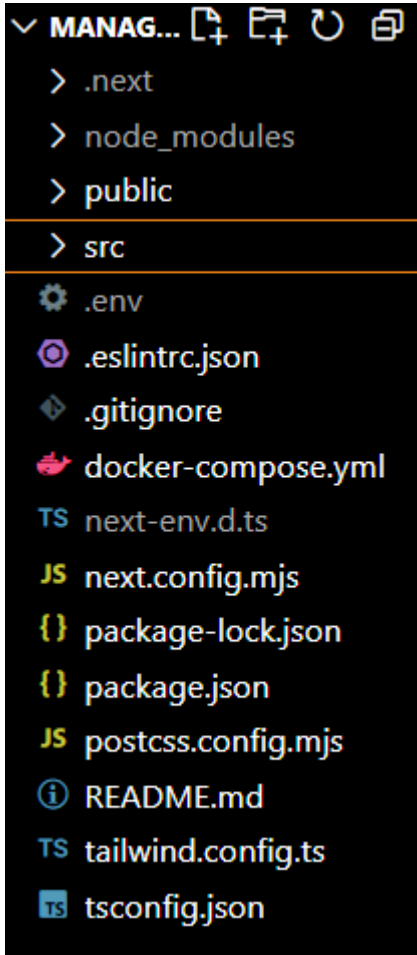
Estadísticas de Tareas

Categoría	Valor
Realizadas	3
Pendientes	2
En Progreso	0

Total de Proyectos

Proyecto	Número de Proyectos
Proyectos	2

Resumen del código



El proyecto es una aplicación desarrollada en **Next.js** que implementa autenticación con **NextAuth.js** y utiliza una base de datos MongoDB para gestionar los usuarios y tareas. A continuación, se explica cómo funcionan las principales funcionalidades del proyecto:

.next/: Carpeta generada automáticamente por Next.js (framework para React), que contiene los archivos compilados y optimizados para la aplicación web.

node_modules/: Directorio que contiene todas las dependencias y bibliotecas de Node.js que se instalan a través de npm o yarn.

public/: Carpeta que almacena recursos estáticos como imágenes, fuentes, etc.

src/: Generalmente, la carpeta principal de la aplicación donde se encuentra el código fuente. Aquí están los archivos React y las configuraciones específicas de la aplicación.

.env: Archivo que contiene configuraciones de la base de datos.

.eslintrc.json: Archivo de configuración para ESLint, que es una herramienta de linting usada para identificar y corregir problemas en el código JavaScript.

.gitignore: Archivo que especifica qué archivos o carpetas deben ser ignorados por Git, como `node_modules/` y `.env`, entre otros.

docker-compose.yml: Archivo de configuración para Docker Compose, usado para definir y administrar entornos con múltiples contenedores de Docker, como bases de datos o aplicaciones web.

next-env.d.ts: Archivo generado automáticamente en proyectos de Next.js con TypeScript, para proporcionar tipos a las configuraciones de Next.js.

next.config.mjs: Archivo de configuración de Next.js donde se pueden establecer rutas, configuraciones de compilación, optimizaciones, etc.

package-lock.json: Archivo generado automáticamente por npm para bloquear las versiones de las dependencias instaladas, asegurando que todos los colaboradores del proyecto usen las mismas versiones.

package.json: Archivo que contiene información sobre el proyecto, como dependencias, scripts de inicio, versiones y más.

postcss.config.mjs: Archivo de configuración para PostCSS, una herramienta para transformar estilos CSS usando plugins como TailwindCSS.

README.md: Archivo de texto generalmente usado para proporcionar documentación básica del proyecto, incluyendo instrucciones de instalación y uso.

tailwind.config.ts: Archivo de configuración para TailwindCSS, que define temas, colores, fuentes, etc., usados en la aplicación.

tsconfig.json: Archivo de configuración para TypeScript que define cómo debe ser compilado el código TypeScript del proyecto.

1. Autenticación con NextAuth.js

- El archivo `src/app/api/auth/[...nextauth]/route.ts` implementa la autenticación utilizando un proveedor de credenciales, funciona así:
 - **Conexion a la base de datos:** Se conecta a la base de datos MongoDB mediante la función `connectDB()`.
 - **Validación de credenciales:** Se usa el proveedor de credenciales (`CredentialsProvider`), que toma un correo electrónico y contraseña ingresados por el usuario, luego busca al usuario en la base de datos y compara la contraseña con la almacenada usando `bcrypt`.
 - **Manejo de sesiones:** Utiliza sesiones basadas en JWT (token JSON Web). Cuando el usuario inicia sesión, se crea un token que contiene la información del usuario autenticado, que luego se adjunta a la sesión.
 - **Callbacks:** Los callbacks permiten personalizar lo que se incluye en el token y la sesión. En este caso, se incluye la información del usuario en el token para las futuras solicitudes.

2. Registro de usuarios

- En `src/app/api/auth/signup/route.ts` se maneja el registro de usuarios. Los pasos principales son:
 - **Validación de datos:** Valida los campos recibidos desde la solicitud (correo electrónico, contraseña, nombre completo, rol).
 - **Hash de la contraseña:** Utiliza `bcrypt` para cifrar la contraseña antes de guardarla en la base de datos.
 - **Creación de usuarios:** Guarda la información del nuevo usuario en la base de datos MongoDB.

3. Diseño y Layout

- En `layout.tsx`, se define la estructura principal del sitio web, con un fondo de imagen y un componente `Navbar`. Este archivo se asegura de que todas las páginas tengan una apariencia coherente y gestiona los componentes compartidos.

5. Despliegue y Documentación

- El proyecto está configurado para ser desplegado en Vercel. La guía de inicio rápido en el archivo `README.md` menciona cómo ejecutar la aplicación localmente y cómo desplegarla en Vercel.