**MANUAL DE USUARIO TASK\_MANAGER**

1. **Tecnologías usadas para el Backend:**



**FastAPI:** FastAPI es un marco web moderno, rápido (de alto rendimiento) para crear API con Python 3.7+ basado en sugerencias de tipo estándar de Python.

****

**JWT:** JWT (JSON Web Token) es un estándar qué está dentro del documento RFC 7519.

En el mismo se define un mecanismo para poder propagar entre dos partes, y de forma segura, la identidad de un determinado usuario, además con una serie de claims o privilegios.

Estos privilegios están codificados en objetos de tipo JSON, que se incrustan dentro del payload o cuerpo de un mensaje que va firmado digitalmente.

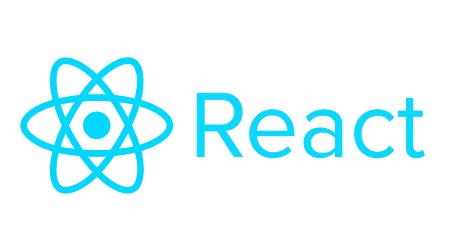


**MongoDB:** es un sistema de base de datos NoSQL, orientado a documentos y de código abierto.

En lugar de guardar los datos en tablas, tal y como se hace en las bases de datos relacionales, MongoDB guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.

MongoDB es una base de datos adecuada para su uso en producción y con múltiples funcionalidades. Esta base de datos se utiliza mucho en la industria.

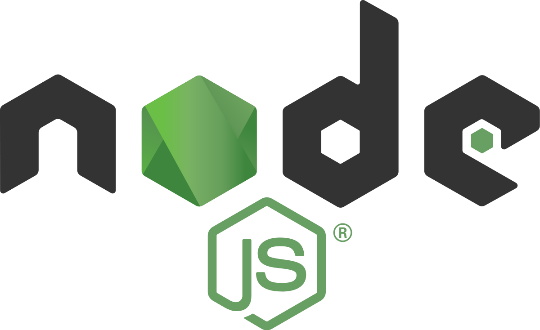
1. **Tecnologías usadas para el Frontend:**



**React:** Es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre. En el proyecto hay más de mil desarrolladores libres.



**Axios:** Es un cliente HTTP basado en promesasnode.js para el navegador. Es isomorfo (= puede ejecutarse en el navegador y nodejs con la misma base de código). En el lado del servidor usa el httpmódulo nativo node.js, mientras que en el cliente (navegador) usa XMLHttpRequests.

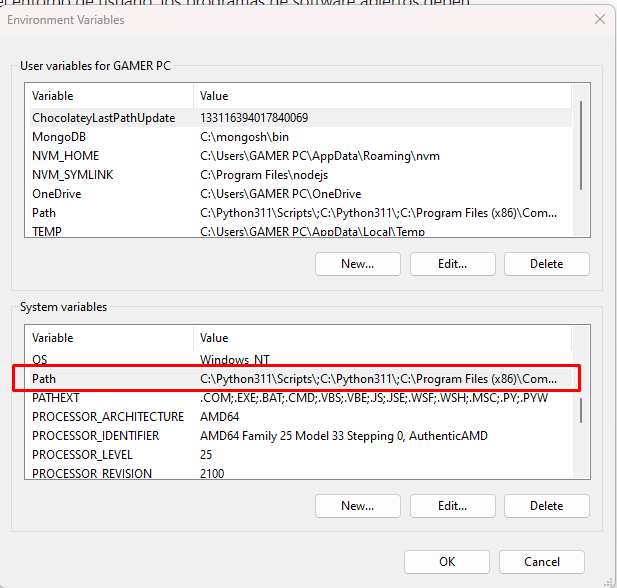


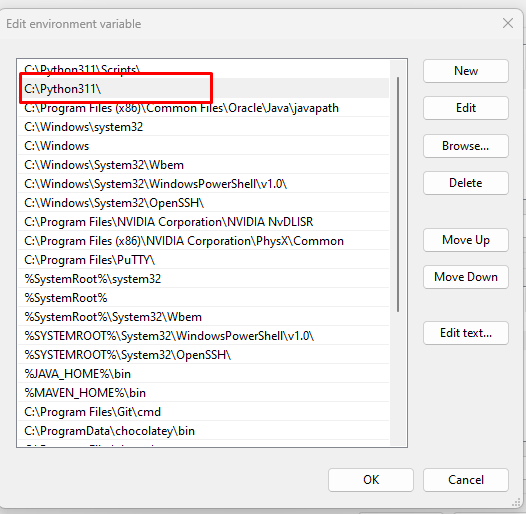
**Node:** Es un entorno de tiempo de ejecución de JavaScript de back-end, se ejecuta en el motor de JavaScript V8 y ejecuta el código de JavaScript fuera de un navegador web.

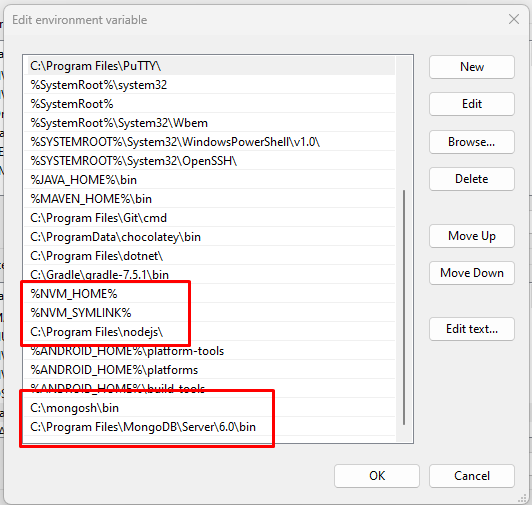
1. **Configuración e instalación de las herramientas:**

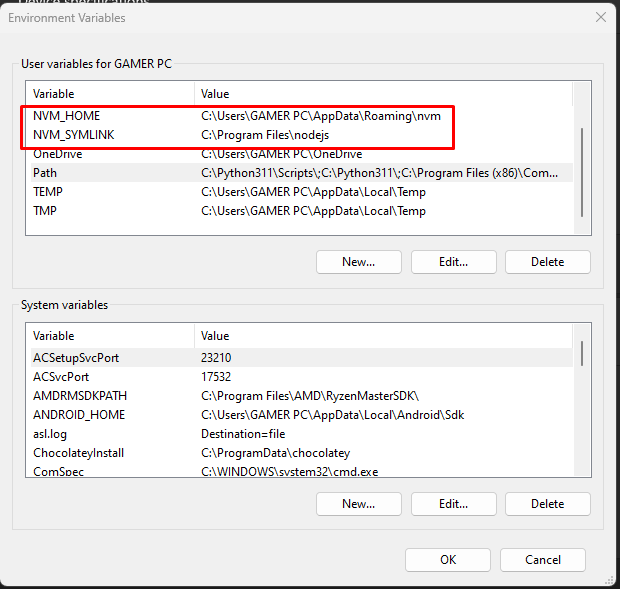
Para el proyecto de administrador de tareas se requirió configurar lo siguiente:

* **Variables de entorno:** Las variables de entorno son cadenas que contienen información acerca del entorno para el sistema y el usuario que ha iniciado sesión en ese momento.





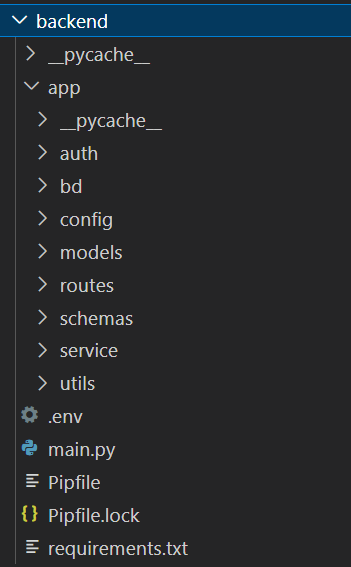




1. **Estructura General del proyecto:**

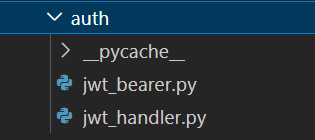
El proyecto está divido en dos carpetas generales:

* **Backend:**

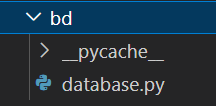


**App/:** Contiene los paquetes:

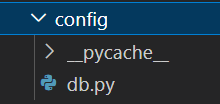
**Auth:** Contiene las clases para la autenticación del usuario con jwt.



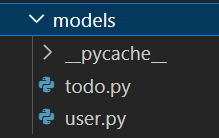
**bd:** Contiene el archivo que genera la conexión con la base de datos y trae los métodos para hacer las consultas.



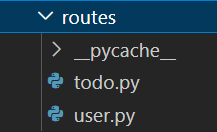
**Config:** Trae una clase que configura la conexión de la base de datos en una variable llamada “con”.



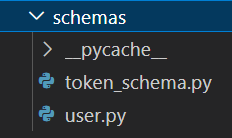
**Models:** Estas clases traen la estructura general de las clases de la base de datos.



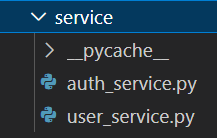
**Routes:** En este paquete se crean las clases que generan las rutas y los diversos métodos de petición que son consumidos desde el frontend.



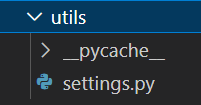
**Schemas:** En este paquete se encuentran las clases que contienen los esquemas de la clase Token y la clase User.



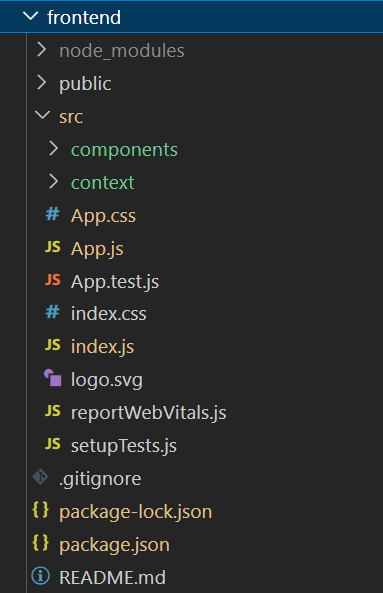
**Service:** Este paquete contiene los service de la clase autenticación y del usuario.



**Util:** En este paquete aparece la clase Settings que configura en variables los campos de la base de datos y del token.

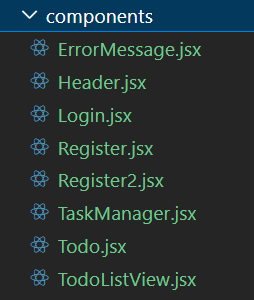


* **Frontend:**

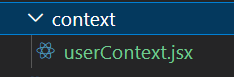
****

**Src/:** Contiene las carpetas de:

**Components/:** Contiene los componentes más importantes de la aplicación.



**Context/:** Contiene el userContext que genera el Token.



Y sus archivos principales son:

**App.js:** Contiene los métodos axios que consumen los Endpoints de la API y además centraliza los componentes que se utilizan en la aplicación.

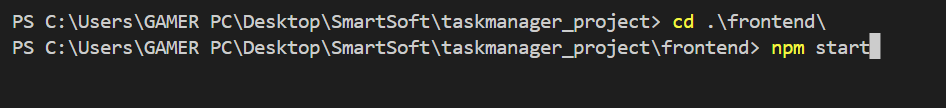
**Index.js:** Consume el App.js

1. **Ejecución del proyecto:**

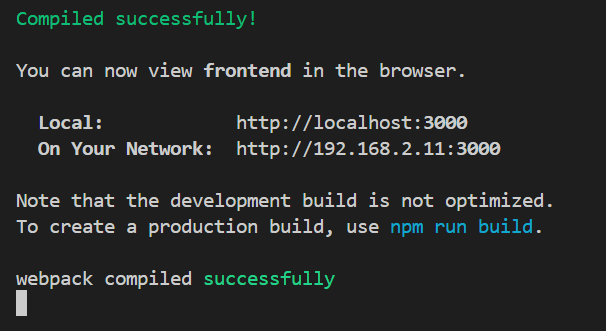
Para la ejecución del proyecto se requiere ejecutar el Fronted y el Backend.

Para ejecutar el Frontend se siguen los siguientes pasos:

1. Dirigirse a la carpeta del Frontend y correr el siguiente comando: “npm start”:

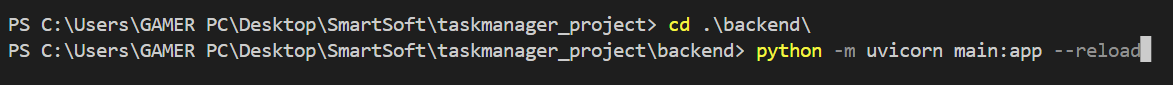


1. Posteriormente se recibirá el siguiente mensaje:



Para ejecutar el backend se siguen los siguientes pasos:

1. Dirigirse a la carpeta del Backend y correr el siguiente comando: “Python -m uvicorn main:app --reload”:



1. Recibirá el siguiente mensaje:

