Documentación de arquitectura

1. Uso de herramientas y tecnologías

Lenguajes de programación: La base del desarrollo de software (Python).

Frameworks: Proporcionan una estructura y conjunto de librerías para facilitar el desarrollo de aplicaciones (Django).

Bases de datos: Herramientas para almacenar y administrar datos. (MongoDB).

Control de versiones: Herramientas para gestionar el código fuente y su evolución a lo largo del tiempo (GitHub).

Pruebas de software: Herramientas para realizar pruebas automatizadas o manuales en el software para garantizar su calidad (Selenium).

Se utilizaron dichas herramientas gracias a la experiencia del equipo desarrollador en las mismas, lo que permite un progreso rápido en el desarrollo de funcionalidades del sistema, además de los beneficios individuales de cada una, como la sintaxis legible y ordenada de Python. Django aporta en la cantidad de componentes que permiten el desarrollo web y MongoDB es enfocado a bases de datos no relacionales, la cual es necesaria para el desarrollo el proyecto Python al contar con librerías enfocadas a la integración de mongo permite un correcto uso de dicha herramienta.

1. Uso de patrones de diseño

Patrón Decorator: este patrón permite agregar comportamiento a un objeto en tiempo de ejecución, envolviéndolo en un objeto de decorador que agrega la funcionalidad adicional.

Patrón Template Method: este patrón define el esqueleto de un algoritmo en una superclase, permitiendo que las subclases redefinan ciertas etapas del algoritmo sin cambiar su estructura general.

MVT (Model-View-Template):

El patrón MVT divide una aplicación web en tres componentes principales:

1. Model: Este componente es responsable de manejar la lógica del negocio y los datos de la aplicación. El modelo interactúa con la base de datos y maneja la recuperación, la creación, la actualización y la eliminación de los datos.
2. View: Este componente se encarga de presentar los datos al usuario en forma de una interfaz gráfica de usuario. La vista no realiza ninguna manipulación de datos, simplemente muestra los datos que se han recuperado del modelo.
3. Template: Este componente define la estructura de la página y cómo se muestran los datos recuperados por la vista. Los templates se utilizan para generar HTML dinámicamente en función de los datos proporcionados por la vista.

En general, el patrón MVT ayuda a separar la lógica del negocio, la presentación y la estructura de la página en componentes separados, lo que facilita la comprensión del código y su mantenimiento.

Existen varias razones por las cuales es beneficioso utilizar el patrón MVT en un proyecto web:

* Separación de responsabilidades: Al separar la lógica del negocio, la presentación y la estructura de la página en componentes separados, el patrón MVT permite una mejor organización del código y facilita su mantenimiento.
* Reutilización de código: Al dividir la aplicación en componentes separados y bien definidos, se puede reutilizar fácilmente el código en otros lugares de la aplicación o en otras aplicaciones.
* Mayor escalabilidad: Al separar la lógica del negocio de la presentación, se puede escalar cada componente por separado, lo que permite una mayor escalabilidad de la aplicación en su conjunto.
* Facilidad de prueba: Al separar la lógica del negocio de la presentación, se pueden realizar pruebas de manera más efectiva y con mayor precisión, lo que ayuda a reducir la cantidad de errores en la aplicación.
* Facilidad de colaboración: Al separar la lógica del negocio, la presentación y la estructura de la página en componentes separados, se puede trabajar en equipo de manera más efectiva, ya que cada miembro del equipo puede centrarse en una parte específica del código sin interferir con los demás.

1. Decisiones sobre la arquitectura

Modelo:

Base de datos de empresas: tabla que almacena información sobre las empresas de software en el departamento del Valle del Cauca, como el nombre, la dirección, la descripción, etc. Esta tabla se relaciona con la tabla de trabajadores mediante una clave foránea.

Base de datos de trabajadores: tabla que almacena información sobre los trabajadores de cada empresa, como el nombre, el cargo, el correo electrónico, el número de teléfono, etc. Esta tabla se relaciona con la tabla de empresas mediante una clave foránea.

Capa de persistencia: esta capa se encarga de gestionar la conexión y las operaciones de lectura y escritura en la base de datos. Se utiliza un ORM (Object-Relational Mapping) para abstraer la base de datos y simplificar las operaciones CRUD.

Lógica de negocio:

Capa de servicios: esta capa se encarga de manejar la lógica de negocio de la aplicación, como validar los datos ingresados por el usuario, procesar las operaciones CRUD en las tablas de la base de datos y gestionar la comunicación entre los diferentes componentes. Aquí se implementan las funciones que se encargan de cargar los datos de la base de datos y procesarlos para su posterior uso.

Vista:

Página de inicio: muestra un formulario de inicio de sesión para los usuarios registrados y un enlace para registrarse para los nuevos usuarios. Esta vista se encarga únicamente de presentar la interfaz al usuario y enviar los datos a la capa de servicios para su validación y procesamiento.

Mapa: muestra un mapa del departamento del Valle del Cauca con marcadores que indican la ubicación de las empresas de software. Al hacer clic en un marcador, se muestra una ventana emergente con información resumida sobre la empresa correspondiente. Esta vista recibe los datos de la capa de servicios y se encarga únicamente de presentarlos al usuario.

Perfil de empresa: muestra información detallada sobre la empresa seleccionada, incluyendo su ubicación en un mapa, la descripción de la empresa, los servicios que ofrece, y una lista de trabajadores con su información de contacto. Esta vista también recibe los datos de la capa de servicios y se encarga únicamente de presentarlos al usuario.

Perfil de usuario: muestra la información del usuario registrado y permite al usuario editar su información personal. Esta vista se encarga únicamente de presentar la interfaz al usuario y enviar los datos a la capa de servicios para su validación y procesamiento.

Template:

Manejador de inicio de sesión: se encarga de procesar las solicitudes del usuario relacionadas con el inicio de sesión, como validar las credenciales del usuario y redirigir al usuario a la página de inicio después del inicio de sesión. Este controlador recibe los datos de la vista y los envía a la capa de servicios para su validación y procesamiento.

Manejador de registro de usuarios: se encarga de procesar las solicitudes del usuario relacionadas con el registro de nuevos usuarios, como validar los datos ingresados y agregar un nuevo usuario a la base de datos de usuarios. Este controlador recibe los datos de la vista y los envía a la capa de servicios para su validación y procesamiento.

Manejador de mapa: se encarga de procesar las solicitudes







