Objetivos

- Desarrollar un sistema de gestión de tareas fácil de usar y seguro que permita a los usuarios crear tareas, asignarlas a otros usuarios y realizar un seguimiento del progreso de las tareas.

Para el sistema de gestión de tareas, se puede utilizar una arquitectura cliente-servidor basada en la web. El servidor alojará la base de datos y el código de la aplicación, mientras que el cliente será una aplicación web a la que los usuarios pueden acceder a través de un navegador web.

componentes:

- Interfaz de usuario: la parte visible del sistema donde los usuarios podrán interactuar con él para crear, asignar y realizar un seguimiento de las tareas.

- Base de datos: para almacenar las tareas creadas por los usuarios y su información correspondiente.

- Servidor de aplicación: para procesar las solicitudes de los usuarios y realizar la lógica de negocio detrás del sistema.

- API: para permitir que otras aplicaciones se integren con el sistema de gestión de tareas.

Diseñar la interacción entre componentes:

- La interfaz de usuario enviará solicitudes HTTP a la API RESTful del servidor de aplicación para realizar operaciones sobre las tareas.

- El servidor de aplicación procesará las solicitudes y realizará operaciones CRUD sobre la base de datos para crear, leer, actualizar o eliminar tareas.

Para implementar esta arquitectura, se pueden utilizar las siguientes herramientas y tecnologías:

Lenguaje de programación: Python

Framework web: Flask

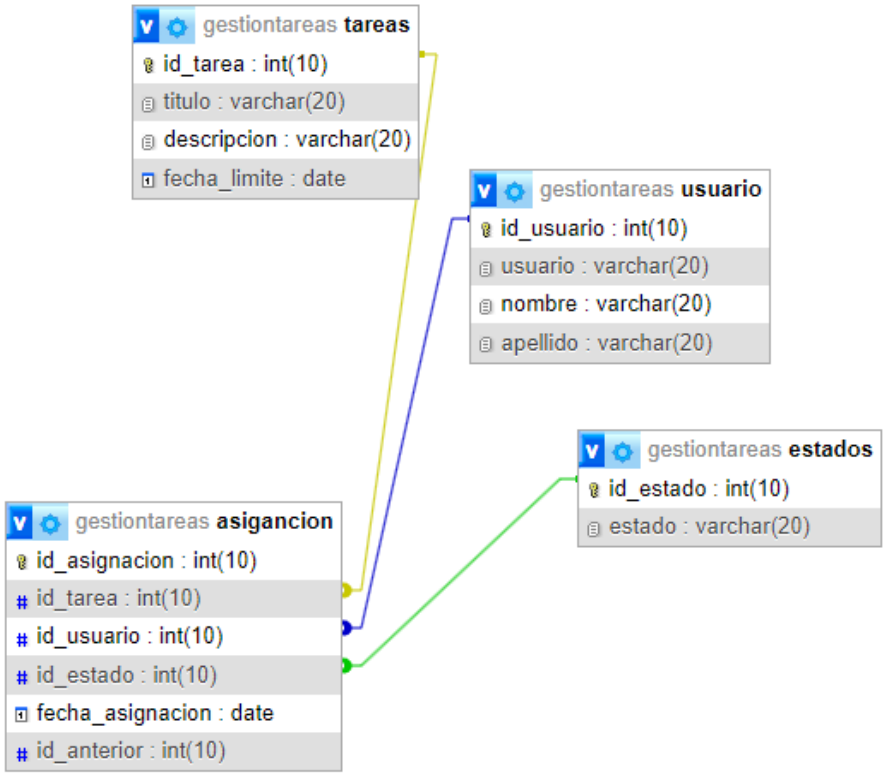
Base de datos: MySQL

Servidor web: Para alojar la aplicación del servidor, se puede utilizar un servidor web como Apache o Nginx.

Herramientas de control de versiones: Git.

API RESTful: Para permitir que la aplicación del cliente se comunique con el servidor.

Gestión de base de datos: Para permitir que los usuarios creen, editen y eliminen tareas en la base de datos.



La base de datos del sistema de gestión de tareas es una base de datos relacional alojada en un servidor MySQL. La base de datos tiene las siguientes tablas:

Usuario: contiene información sobre los usuarios del sistema como nombre usuario, nombre, apellido, dirección de correo electrónico

Tarea: contiene información sobre las tareas creadas por los usuarios, como el título, la descripción, el estado y la fecha de vencimiento.

Asignación: contiene información sobre las asignaciones de tareas a los usuarios, como la tarea, el usuario asignado, el estado, la fecha de asignación y el id de la misma tabla a quien le pertenecía anteriormente en esta tabla quedara la trazabilidad de las tareas en caso de ser reasignada no se reescribirá la fila sino que se creara una nueva

Estado: contiene el id del estado y su estado ['aprobado', 'pendiente', 'incumplimiento', 'en progreso']

Aprobado: la tarea fue realizada con éxito

Pendiente: la tarea aun no ha sido realizada

Incumplimiento: la tarea ha pasado el limite de tiempo establecido

En progreso: la tarea se encuentra en progreso

Arquitectura general del sistema: El sistema gestión de tareas se basa en una arquitectura cliente-servidor. El cliente es un navegador web que permite a los usuarios crear y asignar las tareas de la empresa . . El servidor alojará la base de datos y el código de la aplicación y se accede a el por medio de un api restful.

Componentes principales y su funcionamiento: El sistema plantilla consta de varios componentes principales, como el cliente de escritorio, el servidor web, la base de datos y las plantillas predefinidas. El cliente de escritorio es responsable de proporcionar una interfaz de usuario para la creación y edición de documentos y se comunica con el servidor web para acceder a las plantillas y almacenar los documentos. El servidor web es responsable de proporcionar acceso a las plantillas y procesar los documentos enviados por el cliente. La base de datos se utiliza para almacenar información sobre las plantillas y los documentos creados por los usuarios. Las plantillas predefinidas se utilizan como punto de partida para la creación de nuevos documentos.

Tecnologías utilizadas: El cliente web se desarrolla utilizando el lenguaje de programación y la biblioteca de interfaz de usuario Swing. El servidor web se desarrolla utilizando el lenguaje de programación Python y el framework Flask. La base de datos se implementa utilizando PostgreSQL. Las plantillas predefinidas se crean utilizando HTML, CSS y JavaScript.

Requisitos del sistema: El sistema plantilla debe ser capaz de gestionar múltiples usuarios y documentos simultáneamente. Los usuarios deben poder acceder a las plantillas desde cualquier lugar y en cualquier momento. Los documentos deben ser almacenados de forma segura y ser accesibles solo por los usuarios autorizados. El sistema debe ser escalable y capaz de manejar un alto volumen de usuarios y documentos.

En resumen, para documentar el diseño de la implementación del sistema plantilla, es necesario considerar la arquitectura general del sistema, los componentes principales y su

funcionamiento, las tecnologías utilizadas y los requisitos del sistema. Al hacerlo, se puede garantizar que el sistema se desarrolla de manera eficiente y cumple con los requisitos del usuario.

Informe breve acerca de el enfoque

Para el desarrollo del sistema de gestión de tareas se tomó un enfoque basado en los requisitos del proyecto y en la utilización de herramientas modernas y eficientes para el desarrollo web. En primer lugar, se definieron los requisitos específicos del proyecto, como la creación de tareas, la asignación de tareas a otros usuarios y el seguimiento del progreso de las tareas.

A continuación, se decidió utilizar un enfoque cliente-servidor para el sistema, donde el backend se implementaría como un API RESTful utilizando el framework Flask de Python y el frontend como una aplicación web en PHP. Esta arquitectura permitiría separar la lógica de la aplicación en capas distintas, lo que facilitaría el mantenimiento y la escalabilidad del sistema.

Se implementó el backend en Flask, utilizando una base de datos MySQL para almacenar las tareas y la información de los usuarios. Se crearon los endpoints necesarios para la creación de tareas, la asignación de tareas y el seguimiento del progreso de las tareas. Además, se implementaron las medidas de seguridad necesarias para proteger los datos de los usuarios, como la autenticación de los usuarios y la validación de los datos de entrada.

Por otro lado, se implementó el frontend utilizando PHP y HTML/CSS/JavaScript. Se creó una interfaz de usuario fácil de usar y atractiva para los usuarios, que permitía crear y asignar tareas, así como hacer un seguimiento del progreso de las mismas.