Primer Trabajo Práctico - PW2

Proyecto Tablero de mensajes

El proyecto trata de gestionar mensajes recibidos, los cuales deben estar persistidos en una base de datos, concretamente una sqlite. Estos deben ser visibles en una interfaz de usuario que muestre aquellos mensajes recibidos por un destinatario específico

Modelos del proyecto.

Este proyecto cuenta con un solo modelo llamado Mensajes, este contiene:

- Remitente
- Destinatario
- Cuerpo de mensaje
- Fecha y hora del mensaje

El cual esta definido en el archivo models.py dentro de la app mensajes del proyecto

Para que este modelo quede reflejado en la base de datos debemos primero generar una migración y luego realizar la misma

además django cuenta con algunos modelos que vienen por defecto en la aplicación las cuales también deben ser migradas

Este proceso se realiza con dos comandos sencillos
python manage.py makemigration
esto creará migraciones correspondientes a los modelos que hayamos definidos en modesl.py
luego para realizar dicha migración hacer
python manage.py migrate
esto creará las tablas correspondientes en la base de datos que refleje los modelos definidos, en nuestro caso una tabla llamada Mensajes debería ser creada

Proceso de implementación de la vista y cómo se vinculó a la URL.

Función `base`: Esta función renderiza una página base (`base.html`) que recibe como parámetro un formulario (ListarDestinatarios) y una url hacia el método que atenderá la petición del formulario.

Función `show_mensajes`: Esta función obtiene los mensajes para un destinatario específico, el mismo se obtiene desde el request (`request.GET.get('destinatarios')`) para realizar la consulta correspondiente. Luego los envía en el contexto junto con el destinatario seleccionado.

Clase `ListarDestinatarios`: Crea un formulario (`select`) que contiene una lista de destinatarios únicos obtenidos de la base de datos. Esto permite al usuario seleccionar un destinatario desde un menú desplegable.

Clase `MensajesListView`: Crea una tabla (SingleTableView) del modelo mensajes y utiliza tabla.html

Primer Trabajo Práctico - PW2

Para la vinculación de las URLs en un archivo llamado urls.py (este debe ser creado) dentro de la aplicación mensajes debemos importar las vistas del proyecto (views.py) y crear los patrones de las url

Para acceder al selector de destinatarios se utilizó una url vacía, con esto podemos acceder simplemente con la dirección raíz del proyecto (localhost:8000/) y el método que responde a esta llamada es base

Para acceder a los mensajes recibidos se definió una url llamada mensajes/recibidos y el método que responde es show_mensajes

Descripción de la plantilla HTML utilizada.

Lo primero es crear una carpeta “template” dentro del proyecto donde se guardan todos los htmls, y vincular el path dentro de settings.py del proyecto para que este conozca la ruta a todos los templates

Ambos html cuentan con lenguaje de etiquetado DLT

base.html

Es un html sencillo que renderiza el formulario de tipo select de django para seleccionar al destinatario. Además este cuenta con el token csrf para proteger al proyecto

show_mensajes.html

Este también es un html sencillo, aquí se hace uso de la librería django_tables2 para poder mostrar una tabla de manera más sencilla

Entorno virtual

Para este proyecto se creó un entorno virtual con el comando
virtualenv -p /usr/bin/python3.11 venv_TableroMensajes
Una vez creado podemos activarlo haciendo source sobre el archivo activate en el directorio del entorno

Dentro del archivo requirements.txt estan las dependencias necesarias para que el proyecto funcione

La principal es django la cual puede ser instalada con pip install django==4.2

Otra utilizada es django tables2 para generar tablas, se instala con pip install django-tables2

Detalle del script de datos de prueba

Una forma sencilla de poblar una base de datos es a través de un script.
Un script es un archivo o conjunto de instrucciones escritas en un lenguaje de programación que se ejecutan para realizar una tarea específica de manera automatizada

Primer Trabajo Práctico - PW2

Para poder hacer uso debemos crear un archivo .py e importar “from django.core.management.base import BaseCommand”
luego podes ir generando objetos de tipo Mensajes e insertarlos en la base de datos utilizado la función save()
El script que se utilizó en el proyecto se encuentra en el directorio de la aplicación dentro de management/commands

Detalles sobre la gestión del código fuente y el uso de GitHub.

Para poder gestionar el proyecto se utilizó Git y GitHub
Git es un sistema de control de versiones distribuido que se utiliza para gestionar y rastrear cambios en el código fuente a lo largo del tiempo
GitHub es una plataforma basada en la web que utiliza Git para el control de versiones y proporciona una interfaz de usuario para colaborar en proyectos de software.

Para poder hacer uso del mismo primero hay que instalar git en nuestro sistema operativo dependiendo del SO esta puede variar
Luego de instalarla debemos crear e iniciar sesión dentro de la página github
Una vez dentro podemos crear el repositorio donde subiremos nuestro proyecto hay que tener en cuenta las configuraciones necesarias para que este funcione correctamente
Una de estas es la creación de una clave ssh que utiliza para autenticar y asegurar la comunicación entre tu máquina local y los servidores de GitHub

Reflexión

Lo primero que me gustaría es destacar que el uso de un framework MVT agiliza mucho el trabajo de desarrollo tanto en configuración como en la velocidad en la que se genera código, facilitando el trabajo del desarrollador y permitiendo que este se centra rápidamente en los puntos importantes del proceso de crear software como puede ser el front, back y el negocio de una aplicación web
Esto acompañado de un sistema de control de versiones, permitiendo a los desarrolladores rastrear y revertir cambios con facilidad sumado a su capacidad para manejar ramas que facilita el desarrollo paralelo y la integración de nuevas funcionalidades sin afectar la estabilidad del código principal

Algunas dificultades que se presentaron durante el desarrollo se deben al SO.
Estos al ser tan variados de maquina en maquina pueden generar inconvenientes cuando se trata de preparar las herramientas que vamos a utilizar
En mi caso la instalación del entorno como las aplicaciones que esta fuera a utilizar tuve que hacerlas como sudo su, esto conllevaba a que en ocasiones debía activar dos veces el entorno virtual para hacer uso de pip

Bibliografía

Apuntes de Unidad 1 - Julio César Casco
- Patrones MVC y MVT

Primer Trabajo Práctico - PW2

- Pasos a seguir para migrar el Modelo de la aplicación tareas
- Clonar proyecto Django
- Introducción a GitHub
- Ejemplos realizados en clases
- Clases grabadas

Página oficial Django - https://docs.djangoproject.com/

Uso de la IA Chat GPT