Estándares

Se exige usar todos los estándar para una claridad y un único estilo al código, incluyendo el uso del inglés en todo el apartado de código.

Estándares a utilizar:
PEP 8
Documentación de Django
Conventional Commits.

Implementaciones adicionales

Investigar el uso de:
JSON
addEventListener Para el manejo de evento desde el frontend
url.py en django
views.py en django
`TemplateView django`

Implementación técnica

1. Se debe crear una nueva app, para el caso de uso “Ubicar centro (Y1)”.
   Elegir un nombre adecuado para la app respetando el estándar de django, a diferencia de otras app, está no tendrá definido un modelo asociado, ya que hay que evaluar los datos que requiere la API de Google maps y evitar gastar los token y request gratis que podría ofrecer la API. Para un nuevo feature se definirá la app completa este es un esquema para definir la funcionalidad lógica, que define la Key API de google maps la configuraciones para correr la visualización del mapa.
2. flujo de orden: definir un vista-template-url.
   La vista: indica la ejecución en la página web
   El template: indica la presentación asociada (frontend)
   La url: indica la conexión entre vista a hacia otra vista asociada a esa app

3. Se debe definir 3 localidades ejemplo:
   Bella Vista
   Ziruma
   Circulación 2

Las localidades pueden ser aleatorias.
Se recomienda a extraer la dirección exacta desde google maps:

“Av 4 Bella Vista, con Calle 63 C. 64, Maracaibo 4002, Zulia”
Formato de dirección

Estas direcciones deben estar simplificadas (el nombre del sector ejemplo Ziruma) y solo se mostrará una página visual con un evento: botón o choice dependiendo el diseño. Se recomienda usar un json con clave “nombre dirección”: valor “dirección completa”.

4. Cuando se presione un botón esta debe disparar y redirigir a una página asociada a ella.
   mostrar otra página con el nombre de la localidad en formato json con todos los datos incluido en el botón.

5. La redirección puede ser con con js. Se considera tener en cuenta para el siguiente feature se debe realizar el trabajo completo de la app.

Prueba de testeo

Prueba #1

al iniciar el servidor
y escribir la url de la página asociado a la página de la app ejemplo:
http:localhost:8000/nombredelproyecto/nombredelaapp/nombredelaurlasociado

Resultado:
debe mostrar la página con el html con el boton de redireccion

Prueba #2
se debe probar las 3 url para evaluar la conexion con el template:
http:localhost:8000/nombredelproyecto/nombredelaapp/url1
Resultado:
Carga la página correctamente de las otras url.

Prueba #3
se debe mostrar como redirecciona la pagina segun la eleccion elegida
Aquí se debe mostrar el dato JSON con toda la información de la dirección, cuando se dirige a otra página con el método que use se debe anexar el JSON definido en la página principal para ser mostrado en la otra pagina hay varias manera de realizar se da libertad de elegir cualquiera siempre que sea desde el frontend. Por el backend solo puede ser en la prueba opcional

Resultado: Persistencia y que los datos del json sea los mismo que el definido en la página principal

prueba opcional#4:
Conectar el frontend con el backend y depurar como las url reacciona con la elección desde el frontend para redirigir la página asociada. Usando fetch en el frontend y usando response json desde el backend enviando el estado del servidor errores.

Referencia:

diagrama de objeto y clases UML
https://app.diagrams.net/#G1YAHawV_4b5U76BS-9gmL8sMlVYVHPauZ#%7B%22pageId%22%3A%22U-iC_3jA6ggDQsA9Twkj%22%7D
Casos de usos
https://docs.google.com/document/d/1zZqVG-jTQpbP9ZQByzLtTL7_y9ByX7rc5bEmPENYOZQ/edit?tab=t.0

PEP 8
https://peps.python.org/pep-0008/
Documentación oficial de Django
https://docs.djangoproject.com/es/6.0/

convención de commit
https://www.conventionalcommits.org/en/v1.0.0/