El proyecto "Paris Estilos" está organizado siguiendo una arquitectura desacoplada, también conocida como arquitectura cliente-servidor o API-driven. Esto se nota claramente al observar cómo están separadas las carpetas del backend y del frontend. Esta separación no solo es física, sino también lógica, ya que cada parte cumple una función diferente y se comunica mediante solicitudes HTTP.

Por un lado, el **backend** está construido con el framework Django, que es muy usado en Python para el desarrollo de aplicaciones web. Al revisar los archivos presentes, se pueden ver los clásicos **settings.py, urls.py, wsgi.py y asgi.py,** lo que confirma que es un proyecto Django estándar. Además, dentro de la estructura del backend hay una aplicación personalizada llamada **core,** y lo más interesante es que el archivo **urls.py** está configurado para exponer rutas bajo el prefijo **api/.** Esto es una señal clara de que el backend funciona como una API, probablemente usando **Django REST Framework (DRF),** ya que también se incluye un archivo serializers.py, el cual es característico de este tipo de desarrollo.

Por otro lado, el **frontend** está completamente separado en otra carpeta, llamada paris-estilos-frontend. En esta parte se encuentran archivos HTML que representan diferentes secciones del sitio, como **login.html, agendar-cita.html, productos.html**, entre otros. También se observa una carpeta llamada components, lo cual sugiere que se está usando una estructura modular, incluso si no se está utilizando un framework como React o Vue. Lo más probable es que este frontend se comunique con la API del backend usando JavaScript, posiblemente con fetch o alguna librería como axios, para mostrar los datos al usuario.

Esta arquitectura desacoplada trae muchas ventajas. Primero, permite que el backend y el frontend se desarrollen y mantengan de forma independiente. Esto significa que se pueden hacer cambios o actualizaciones en uno sin afectar directamente al otro. También es ideal si en el futuro se quiere desarrollar una aplicación móvil, ya que se puede reutilizar la misma API. Otra ventaja importante es que mejora la escalabilidad, porque se puede distribuir la carga entre servidores diferentes si es necesario.

En resumen, el proyecto está bien organizado y sigue un enfoque moderno en el desarrollo de aplicaciones web. Como estudiante de ingeniería en sistemas, este tipo de arquitectura es muy recomendable porque refleja buenas prácticas y está alineada con lo que se espera en entornos profesionales. Nos permite trabajar de forma más limpia, ordenada y escalable, y facilita la colaboración entre equipos de frontend y backend.