Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2024

PROYECTO NO. 3 - DESAROLLO WEB

201730511 – Edy Rolando Rojas González

Resumen

El navegador de hoy en día es una herramienta poderosa e indispensable para el día a día de muchas personas no solo en el ámbito cotidiano sino que también laboral, educativo, profesional e incluso en la investigación.

Es sorprendente que a medida que la tecnología avanza se crean nuevos roles con el fin de cubrir áreas muy especializadas, y en el mundo de la web y los navegadores surgieron dos roles muy importantes. Por un lado el frontend se orienta a todo lo que tiene que ver con lo que se muestra al usuario y el backend la lógica de negocio lo cual el usuario no verá, pero la unión de ambos mundos crea aplicaciones web y páginas más interactivas e intuitivas.

Algo muy curioso de ambos campos es que las tecnologías no necesariamente deben ser la misma, puede haber un frontend construido en React o Django y el backend puede estar desarrollado en Java Spring Boot o Express.js y ambas partes podrán comunicarse entre sí sin problemas.

Palabras clave

Frontend, Backend, Frameworks y librerías, API 's y Métodos HTTP.

Abstract

Today's browser is a powerful and indispensable tool for the daily lives of many people, not only in everyday life but also at work, in education, professionally and even in research.

It is surprising that as technology advances, new roles are created in order to cover very specialized areas, and in the world of the web and browsers two very important roles emerged. On the one hand, the frontend is oriented towards everything that has to do with what is shown to the user and the backend the business logic which the user will not see, but the union of both worlds creates more interactive and intuitive web applications and pages. .

Something very curious about both fields is that the technologies do not necessarily have to be the same, there can be a frontend built in React or Django and the backend can be developed in Java Spring Boot or Express.js and both parties will be able to communicate with each other without problems.

Keywords

Frontend, Backend, Frameworks and libraries, API's and HTTP Methods.

Introducción

Desarrollar backend y frontend permite al estudiante entender como dos plataformas construidas en tecnologías distintas se comunican entre sí, siendo esto de gran valor de cara al campo laboral.

Python para el backend con Flask ha sido una grata experiencia, sintaxis sencilla, amplia documentación y configuración simple, aspectos que la dotan de mucho valor a la hora de levantar un backend básico.

Por otro lado, Django es una herramienta robusta y altamente completa con un grupo de herramientas que sabiendo cómo utilizar se presenta como una solución sólida para el frontend, su modelo de MVT es muy intuitivo y fácil de comprender y adoptar, agreguemos que heredar templates y reutilizar componentes es apenas complejo, lo hacen un framework cómodo y gratificante con el cual trabajar. Quizá su configuración inicial tome algo de tiempo al inicio, sin embargo, su documentación oficial es una maravilla para empezar.

Desarrollo del tema

Backend - Flask

El diagrama de Entidad Relación proporcionó una idea clara y concisa de la estructura que debía de adoptar el backend, específicamente en el apartado de los modelos y una vista general de los controladores y servicios a desarrollar.

El backend se compone de 4 módulos generales adoptando una arquitectura de controladores, servicios y entidades entendiendo que se tiene que administrar archivos XML como Bases de Datos.

A continuación se describen con más detalle los módulos principales.

- Controllers

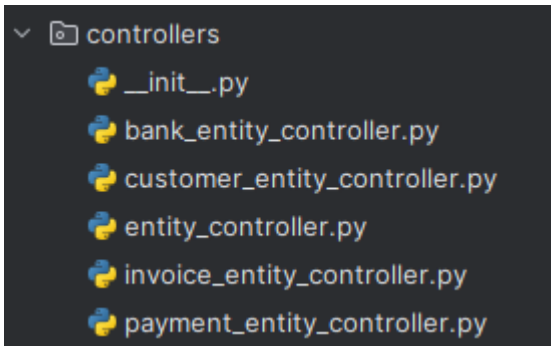


Figura 1. Captura de las clases que componen el módulo Controllers.

Fuente: elaboración propia.

Este módulo se encarga de administrar la creación de las entidades, validando que la información recibida sea correcta para crear la entidad y posteriormente ser agregada a la base de datos.

Los controladores heredan de una plantilla base llamada Entity Controller que define la estructura básica de las operaciones que deben de realizar.

- Entities

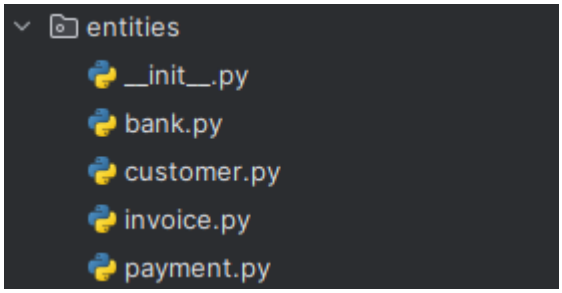


Figura 2. Captura de las clases que componen el módulo Entities.

Fuente: elaboración propia.

Se encuentran las clases que dan forma a las entidades abstraídas que son necesarias para la correcta administración de los datos que se almacenarán en las Bases de Datos.

- Services

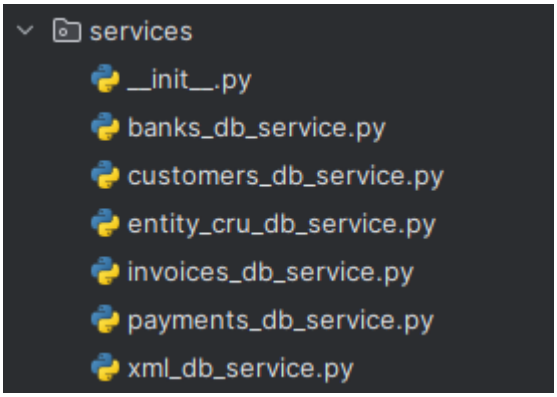


Figura 3. Captura de las clases que componen el módulo de Services.

Fuente: elaboración propia.

Las clases que se encuentran en este módulo son las encargadas de comunicarse con la Bases de Datos estos servicios heredan de una plantilla base que se llama XML DB Service que define su estructura básica, también existe otra clase abstracta que define el modelo base de las funciones de Create and Update, clase la cual implementa Customers DB Service y Banks DB Service, ya que solo estan tendran ese comportamiento.

Este módulo representó un reto considerable debido a que este cargaría con el mayor peso de lógica debido a que se comunica directamente con las Bases de Datos, también es importante resaltar que al definirse un buen modelo ER se tenía claro como las tablas se relacionan entre sí orientando de manera general cuáles serían las consultas que se realizan y por lo tanto que funciones y métodos debían de ser implementados en los servicios.

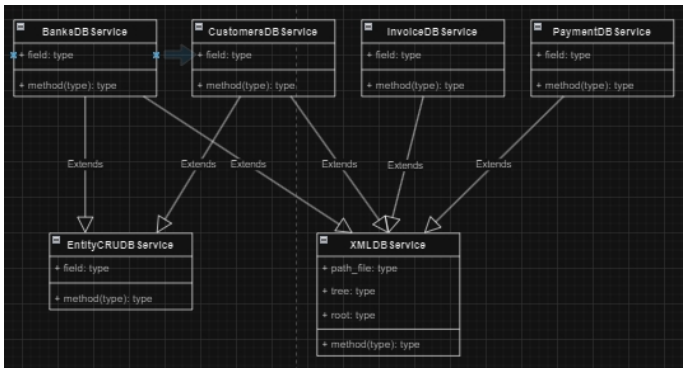


Figura 4. Captura de las clases que componen el módulo de Services.

Fuente: elaboración propia utilizando draw.io.

- Utils

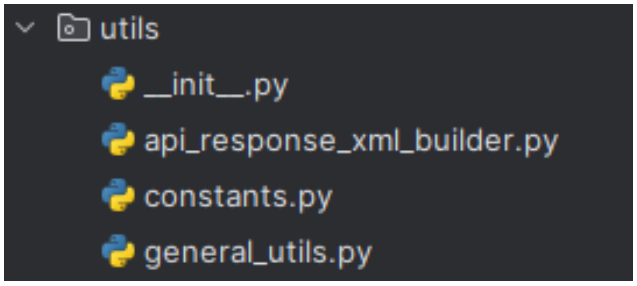


Figura 5. Captura de las clases que componen el módulo de Utils.

Fuente: elaboración propia.

Módulo que almacena clases utilitarias como por ejemplo API Response XML Builder que se encarga de crear las respuesta en formato XML que posteriormente de utilizaran para retornar respuestas en los endpoints o rutas.

App - Flask

Se definieron 6 endpoints o routes para el Backend, se utilizó un formato el cual sea descriptivo y consistente para el nombramiento de los endpoints, tomando algunas sugerencias proporcionadas por el auxiliar del curso:

1. api/v1/config

2. api/v1/transaction
3. api/v1/reset_database
4. api/v1/customers/<customer_nit>
5. api/v1/customers/all
6. api/v1/payment/<date>

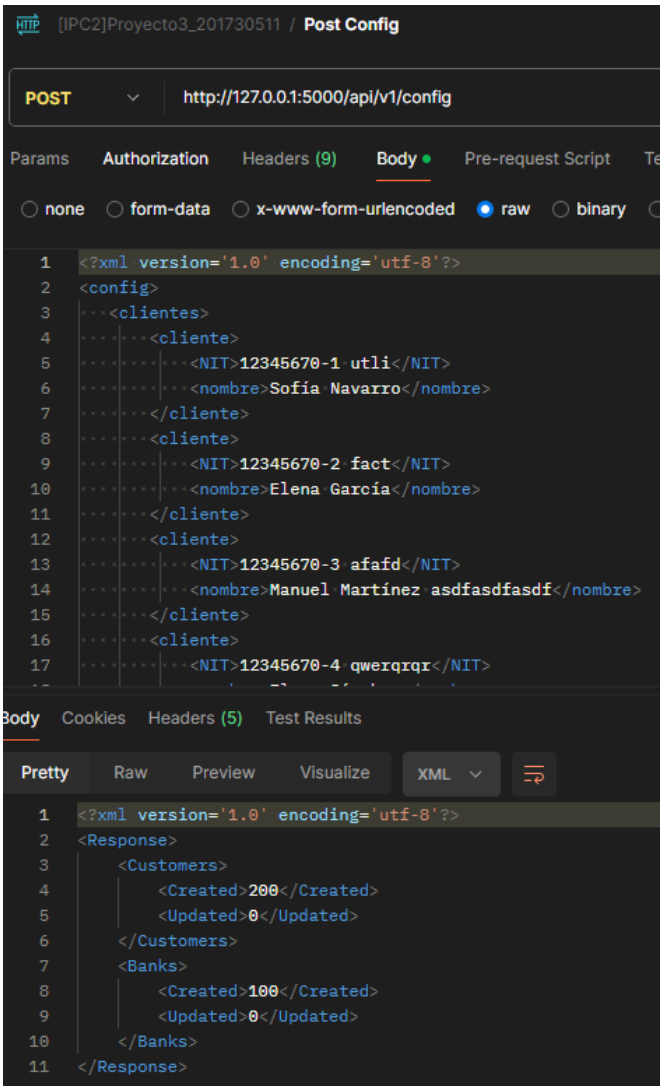


Figura 6. Captura del uso de uno de los endpoints por medio de Postman y la respuesta del Backend a la solicitud.

Fuente: elaboración propia.

Frontend - Django

La configuración inicial de Django puede llegar a ser un poco complicada si no se maneja el concepto de entornos virtuales y su posterior activación.

Los conocimientos solidos de HTML, CSS y JS son fundamentales para utilizar el framework, la comprension basica de las templates y componentes son obligatorias para entrar el mundo de Django ya que el uso apropiado de las mismas facilitan el desarrollo de una interfaz amigable.

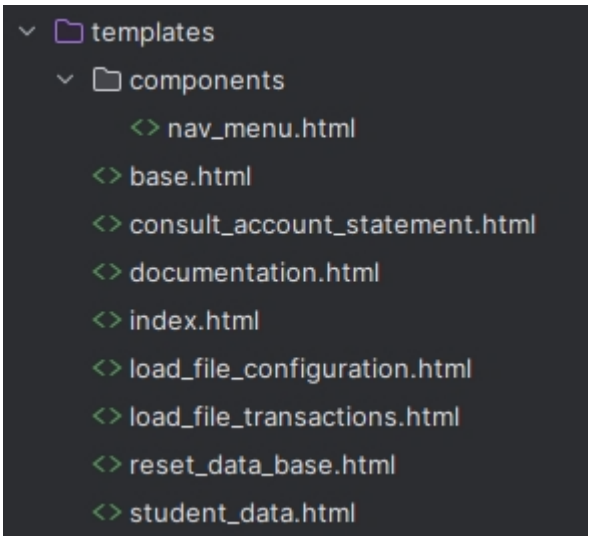


Figura 7. Captura de las templates desarrolladas para el Frontend.

Fuente: elaboración propia.

Templates

Se utilizó una plantilla base y aprovechando la herencia de templates se definió la estructura básica de la Web Application.

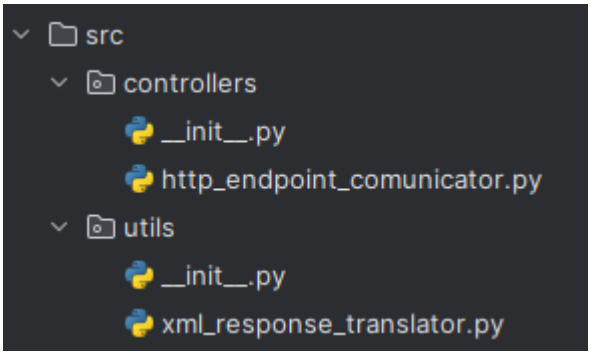
Components

Se implementó un menú de navegación como componente en la template base debido a que este debía de estar presente en todas la páginas que componen la Web Application

CSS

Aprovechando la posibilidad de importar css en otros archivos utilizando @import ‘file_css’ se dividió en varias hojas de estilos que importan a un CSS general para segmentar los estilos en pequeños módulos dedicados para cada página y componente.

Se desarrollaron 3 módulos principales



Conclusiones

Extensión: de cuatro a siete páginas como máximo