softpatterns development

MANUAL TÉCNICO

SISTEMA CENTRALIZADO PAR-KUD COLOMBIA

Julio, 10 de 2023

Bogotá, Colombia

1. Introducción:

Durante este documento, se abordarán los temas referentes al concepto de solución planteado por el equipo de desarrollo para el proyecto ***Park-UD Colombia*** haciendo énfasis en las tecnologías, posibles bases de datos, tipos de arquitecturas, patrones de diseño y algunas herramientas adicionales que se planean utilizar durante la etapa de desarrollo del proyecto.

2. Tecnologías:

En esta sección se evaluaron algunas opciones de tecnologías tanto de Front como de Back-end, en las siguientes subsecciones mostraran las opciones evaluadas y la opción escogida:

2.1. Front-end: En esta sección se evaluaron las posibles tecnologías a utilizar en el desarrollo del Front-end del proyecto, algunas de las opciones propuestas por los integrantes del grupo durante la reunión fueron las siguientes:

- Angular
- React
- Vue
- Next
- JavaScript
- TypeScript
- Flutter

Sin embargo, después de la deliberación por parte de todos los integrantes del grupo de desarrollo se definió utilizar ***React*** como principal herramienta **Front-end**, evidentemente haciendo uso de otras herramientas necesarias para el desarrollo como ***HTML***, ***CSS*** y además ***JavaScript*** teniendo en cuenta que **React** es una librería basada en JavaScript basada en componentes permitiendo así la creación de interfaces de usuario interactivas de manera sencilla.

2.2. Back-end: En esta sección se evaluaron las posibles tecnologías a utilizar en el desarrollo del Back-end del proyecto, algunas de las opciones propuestas por los integrantes del grupo durante la reunión fueron las siguientes:

- NodeJS
- PHP
- Java
- .Net

Sin embargo, después de la deliberación por parte de todos los integrantes del grupo de desarrollo se definió utilizar ***NodeJS*** como principal herramienta **Back-end** por su versatilidad al momento de la creación de servicios web basados en **API REST**, es importante tener en cuenta que NodeJS es un lenguaje de programación basado en JavaScript. Otra tecnología que se utilizará es ***Express.js***, este es uno de los frameworks de back-end más utilizados para NodeJS ya que proporciona un sinfín de herramientas y características robustas que permiten desarrollar aplicaciones de back-end escalables. Al observar con detenimiento se puede encontrar que la mayoría de las tecnologías utilizadas se enfocan o están basadas en el uso de JavaScript, esto se hizo debido a que el equipo de desarrollo ***SoftPatterns Development*** cuenta con mayor

conocimiento en este lenguaje de programación, lo que disminuye de forma significativa la curva de aprendizaje de un lenguaje de programación.

3. Base de datos:

En tanto a este tema desde el inicio como equipo de trabajo definimos utilizar una ***base de datos relacional*** debido a que contamos con conocimiento sobre estas, haciendo más fácil el planteamiento de modelos como el Diagrama Entidad-Relación y Modelo Relacional que permitan entender principalmente el manejo de datos, su uso y sus relaciones. Ya teniendo esto como punto de partida, el siguiente tema a tocar fue el **Sistema Gestor de Base de Datos (SGBD)** a utilizar, en este tema se plantearon algunas opciones:

- PostgreSQL
- MySQL
- MariaDB
- Microsoft SQL Server

Pero nuevamente, teniendo en cuenta los conocimientos del equipo de trabajo, se optó por utilizar **MySQL** desde una herramienta de Software libre como **XAMPP**. Como equipo de desarrollo contamos con los conocimientos necesarios para que en conjunto trabajemos sobre este SGBD.

4. Arquitecturas y patrones de diseño:

En este apartado se abordaron los temas consignados en dicho numeral. Como equipo de trabajo, ***SoftPatterns Development*** optó por utilizar principalmente dos patrones arquitecturales.

El primero fue el **Modelo Vista-Controlador (MVC)**, ya que desde el inicio tuvimos claro que este modelo se ajusta correctamente al proyecto y a las tecnologías que optamos por utilizar. Por un lado, el modelo será proporcionado por la base de datos ya que desde allí se manejarán los datos y la lógica del negocio, la vista será proporcionada desde el Front-end utilizando React y el controlador desde NodeJS para actualizar los datos del modelo y la lógica de negocio.

El segundo patrón arquitectural fue la **Arquitectura Cliente-Servidor** ya que el cliente realizará solicitudes a un servidor para obtener datos o realizar acciones. Desde ese punto de vista, el cliente en este caso será la aplicación web del proyecto que solicitará algunas acciones al servidor utilizando diferentes métodos HTTP, como el GET, POST, DELETE o PUT y recibirá las respuestas que el servidor le proporcionará en formato JSON o XML, en este caso para el proyecto las respuestas proporcionadas por el servidor serán dadas en formato ***JSON***.

Por otro lado, dos de los patrones de diseño que tenemos claros que se utilizarán inicialmente son:

- **Singleton:** En el momento en que se hace la conexión de la base de datos, esa conexión se hace una única vez. De esa forma se garantiza que solo haya una conexión activa a la base de datos en todo momento.
- **Data Access Object (DAO):** Este patrón se utilizará para abstraer y encapsular el acceso a la base de datos separando la lógica de la aplicación de los detalles de implementación de la base de datos facilitando el mantenimiento y la evolución del código. En el contexto de una aplicación Node.js que utiliza una base de datos MySQL,

el patrón DAO se puede utilizar para encapsular el acceso a la base de datos y proporcionar una interfaz común para realizar operaciones CRUD (Create, Read, Update, Delete) en la base de datos.

5. Herramientas adicionales de diseño:

En este apartado se enfocó inicialmente utilizar la herramienta **Enterprise Architect**, esta herramienta se definió utilizar debido a su robustez en un sinfín de ámbitos de desarrollo. Principalmente se planea utilizar esta herramienta para la creación del modelo relacional ya que esta herramienta permite utilizar ingeniería inversa para la creación del script de la base de datos cuando ya el modelo relacional esté completo.

Atentamente, representantes de trabajo de **SOFTPATTERNS DEVELOPMENT**.

Nombre	Rol	E-mail	Código institucional
Juan Sebastián Gonzalez Forero	Líder de proyecto	jusgonzalezf@udistrital.edu.co	20181020029
Marcela del Pilar Porras Quevedo	Líder de planeación	mdel_porrasq@udistrital.edu.co	20191020131
Ibzan Jared Peralta	Líder de desarrollo	ijperaltar@udistrital.edu.co	20191020133