UNIVERSIDAD DE LA SIERRA JUÁREZ

CARRERA:
LICENCIATURA EN INFORMÁTICA

ASIGNATURA:
SISTEMAS DISTRIBUIDOS

NOMBRE DEL TEMA:
MICROSERVICIOS

Profesor: M.T.C.A Francisco Javier Méndez Vázquez

Alumno: Bernardino Hernández Rojas
Semestre: Octavo

19 de junio de 2023

Universidad de la Sierra Juárez
Licenciatura en Informática

Índice

Índice de figuras

Resumen

El trabajo realizado, utilidad del trabajo realizado, cómo se hizo , qué resultados se obtuvieron, y la importancia del trabajo realizado. (de 250 palabras máximo) en esta sección no se citan autores.

1. Introducción.

El siguiente reporte aborda la implementación de **microservicios** utilizando el lenguaje de programación ”Java” empleando el framework llamado **Spring Boot**, nos enfocamos en el principio de ”fragmentación”, el cual nos dice que una aplicación monolítica se puede descomponer en piezas de software más pequeñas independientes que se encargan de una tarea especifica dentro de una aplicación más grande.

La importancia de conocer el funcionamiento y la implementación de microservicios se ve refleja en las nuevas tecnologías que tienen como tendencia la migración a la nube...

2. Objetivo general

Desarrollar microservicios capaces de alimentarse mutuamente y con esto lograr el escalamiento hacia una aplicación más grande sin comprometer su funcionalidad, brindando flexibilidad, escalabilidad, y tolerancia a fallos.

2.1. Objetivos específicos

- Desarrollar microservicios capaces de implementar nuevas funcionalidades a un sistema mayor.
- Comprender la funcionalidad de un microservicio.
- Aprender a desarrollar microservicios con Spring Boot.
- Utilizar Postman para probar la funcionalidad de los microservicios desarrollados.
- Aprender a dar funcionalidad a un microservicio desde requerimientos generales.
- Analizar los requerimientos generales y proporcionar una solución lógica y genérica sobre el microservicio a desarrollar.

3. Desarrollo

3.1. Tecnologías y Software empleados

- IntelliJ IDEA 2023.1.2 (Ultimate Editon)
- Java correto-17 versión 17.0.17)
- Git versión 2.34.1

- MySQL Workbench 8 versión 8.0.32
- Mysql Ver 8.0.33-0ubuntu0.22.04.2
- Spring Boot 3.0.1
- Ubuntu 22.04 LTS
- Postman v10.15.1

3.2. Procedimiento

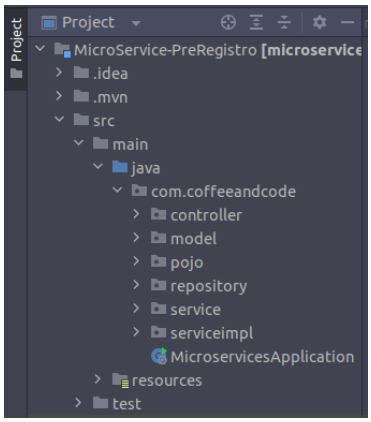
Desde un requerimiento general se debe realizar un análisis a profundidad sobre las necesidades, dicho análisis debe ser genérico; es decir debe valer para cualquier aplicación que requiera de esa funcionalidad.

Una vez teniendo el análisis se revisa la documentación de Spring Boot sobre la implementación de los microservicios, con esto podemos transportar el análisis realizado en papel a código de programación, en este caso utilizamos Java como lenguaje de programación principal apoyándonos de un framework que facilita la implementación de funcionalidades que son demandantes de tiempo al estar desarrollando como lo es la conexión a base de datos, facilidad en integrar dependencias, facilidad de visualizar el desarrollo en el navegador integrando un contenedor de aplicaciones web para ser desplegado.

3.2.1. Patrón MVC

Utilizamos el patrón de desarrollo MVC para logar la estructura de carpetas ordenadas sin mezclar la lógica de negocio con la funcionalidad a implementar. El patrón de desarrollo MVC se compone de esta estructura:

- Modelos: Representan los datos y la lógica de negocio de la aplicación, con ellas gestionamos las entidades de las bases de datos, validaciones a nivel de tipo de datos y limitantes de aceptación. En estos componentes es donde se establece la cardinalidad de las entidades a utilizar.
- Vista: Es la interfaz de usuario de la aplicación, se encarga de la representación visual de los datos y la interacción con el usuario, en este caso omitimos esta parte en microservicios ya que estos solo devuelven colecciones de tipo Json o xml según sean las necesidades de la aplicación a escalar.
- Controlador: Actuá como intermediario entre el modelo y la vista, además su parte principal es controlar el flujo de la aplicación, coordinando las diferentes acciones y actualizaciones internas del sistema.



Se decide utilizar este patrón de desarrollo, debido a que permite la modularidad y flexibilidad en el desarrollo de nuestro microservicio, cada componente tiene una responsabilidad especifica y se puede modificar o remplazar de manera independiente, la ejemplificación del patrón MVC la podemos observar en la siguiente imagen.

3.2.2. Servicios, interfaces de repositorios, implementación de interfaces.

Una de las facilidades que permite Spring Boot es la inyección de dependencias, con estas se hace referencia a los servicios e interfaces de repositorios, en otros componentes, esto similar al funcionamiento de los controladores.
Se utilizan para dividir las responsabilidades y facilitar la prueba y mantenimiento del código, se podría decir que también son pieza importante donde se permite la modularidad, escalabilidad y flexibilidad del sistema.

- Servicios: Aquí se implementa la lógica de negocio de una aplicación, representan la capa intermedia entre los controladores y los repositorios, con ellas se busca proporcionar funcionalidad especifica y operaciones más complejas.

- Interfaces de repositorios: Definen métodos para interactuar con la capa de almacenamiento de datos(bases de datos, o servicios web). Permiten la abstracción de la lógica de acceso a datos y proporcionan una forma estandarizada para las operaciones CRUD(Crear, Leer, Actualizar, Eliminar) sobre los objetos persistentes.

- Implementación de interfaces: Contienen la creación de clases que implementan las interfaces definidas, dichas clases contienen la lógica concreta para llevar a cabo las operaciones definidas en las interfaces, es decir aquí es en donde se desarrolla el método completo.

3.2.3. Desarrollo de microservicios

Iniciamos con el proceso de desarrollo de microservicios, primero definimos que tipo de datos aceptará y entregará nuestro microservicio, para este caso especifico se decide que será en Json.

Utilizando MVC definimos las entidades de nuestra base de datos en el apartado **model**, en este se establece la cardinalidad que hemos definido en nuestro diagrama entidad relación que diseñamos durante el análisis de requerimientos, es aquí donde definimos los atributos que contendrá la tabla, ademas del nombre de la misma todo con notaciones que pertenecen a dependencias utilizadas para **maven** apoyados del framework **Spring Boot.**
Describe tus diagramas de clase utilizado en caso de haber todos elementos empleados.

3.3. Resultados

Aquí describir los resultados obtenidos evidencia del programa funcionando y describir las funcionalidades puedes incluir imágenes.

4. Conclusión

Explique por qué su diseño sirve (o nó) y qué hace falta por hacer. Responda las expectativas generadas en la Introducción. Analice sus resultados críticamente y diga si son creíbles y responda qué tan seguras son sus propuestas.