UNIVERSIDAD NACIONAL DEL ALTIPLANO

"FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA"

Plataforma de Turismo Rural Comunitario

Learners:
TICONA CASA KATIA DAISHY

Teacher:
TORRES CRUZ FRED

Course:
SOFTWARE ENGINEERING

SEMESTER VII

PUNO, PERÚ

2024

1 Introduction

2 Arquitectura en Capas con Microservicios

- **Capa de Presentación**

  - Microservicio Web: Desarrollado con Spring Boot y frameworks web como Thymeleaf o AngularJS.

  - Microservicio Móvil: Implementado con Android Studio y Kotlin o Java.

  - Microservicio API Gateway: Utilizando Spring Cloud Gateway o Zuul para gestionar las solicitudes de API.

- **Capa de Lógica de Negocio**

  - Microservicio de Usuarios: Manejando el registro, autenticación, autorización y gestión de perfiles de usuarios con Spring Security.

  - Microservicio de Listados: Permitiendo a las comunidades locales crear, editar y administrar sus listados de experiencias turísticas con Spring Data JPA y Hibernate.

  . Microservicio de Reservas: Facilitando el proceso de reserva de experiencias turísticas por parte de los usuarios con Spring MVC y servicios de mensajería como RabbitMQ.

  - Microservicio de Comunicación y Colaboración: Proveyendo . herramientas de comunicación y colaboración entre usuarios, comunidades locales y administradores con Spring WebSockets o SockJS.

- **Capa de Acceso a Datos**

  - Microservicio de Base de Datos: Almacenando y administrando los datos de la plataforma con PostgreSQL y Spring Data JPA.

  - Microservicio de Búsqueda: Optimizando la búsqueda de experiencias turísticas relevantes para los usuarios con Elasticsearch o Solr.

  - Microservicio de Notificaciones: Enviando notificaciones por correo electrónico o SMS a usuarios y comunidades relevantes sobre eventos, reservas y otras acciones con Spring Boot Mail o Twilio.

- **Beneficios de esta Arquitectura**

  - Modularidad: Permite desarrollar, mantener y desplegar microservicios de forma independiente.

  - Escalabilidad: Facilita el escalado horizontal de microservicios para manejar un mayor volumen de usuarios y tráfico.

  - Tolerancia a fallos: Si un microservicio falla, los demás pueden seguir funcionando sin afectar la plataforma.

  - Heterogeneidad tecnológica: Permite utilizar diferentes tecnologías para cada microservicio según las necesidades.

  - Facilidad de mantenimiento: La modularidad y la separación de preocupaciones simplifican el mantenimiento del código

3 Requisitos Técnicos Mínimos para Plataforma de Turismo Rural Comunitario en Java

Servidor

- **Procesador** Mínimo 4 núcleos, idealmente 8 o más.
- **Memoria RAM** Mínimo 16 GB, idealmente 32 GB o más
- **Almacenamiento** Mínimo 100 GB de SSD, idealmente 500 GB o más.
- **Almacenamiento** Mínimo 100 GB de SSD, idealmente 500 GB o más.

Herramientas de Desarrollo

- **Java Development Kit (JDK)** Versión 17 o superior.
- **Maven** Herramienta de gestión de dependencias
- **Git** Sistema de control de versiones.

Bases de Datos

- **PostgreSQL** Base de datos relacional para almacenar datos de la plataforma.
- **Elasticsearch** Motor de búsqueda para optimizar la búsqueda de experiencias turísticas.

Consideraciones Adicionales

- **Escalabilidad** La infraestructura debe ser escalable para poder manejar un mayor volumen de usuarios y tráfico en el futuro.
- **Seguridad** Implementar medidas de seguridad robustas para proteger los datos de los usuarios y la plataforma.
- **Monitoreo** Monitorear el rendimiento y la salud de la plataforma para detectar y solucionar problemas rápidamente.

4 Servicios

Microservicio de Usuarios

- **Responsabilidades**

  - Registro de nuevos usuarios.
  - Autenticación de usuarios existentes.
  - Gestión de perfiles de usuario (información personal, configuración de notificaciones, etc.).
  - Autorización de usuarios para acceder a diferentes funcionalidades de la plataforma.
  - Gestión de roles de usuario (administradores, moderadores, usuarios comunes).

- **Tecnologías**

  - Spring Security para autenticación y autorización.
  - Spring Data JPA para almacenar y gestionar datos de usuarios en PostgreSQL.
  - RESTful API para exponer funcionalidades a otros microservicios y aplicaciones externas.

- **Monitoreo** Monitorear el rendimiento y la salud de la plataforma para detectar y solucionar problemas rápidamente.

Microservicio de Listados

- **Responsabilidades**

  - Permitir a las comunidades locales crear, editar y eliminar listados de experiencias turísticas.
  - Gestionar información de listados (títulos, descripciones, fotos, ubicaciones, precios, disponibilidad, etc.).
  - Categorizar listados por tipo de experiencia (alojamiento, actividades, gastronomía, etc.).

  Validar la calidad y veracidad de la información de los listados.

- **Tecnologías**

  - Spring MVC para crear formularios web y manejar solicitudes HTTP.
  - Spring Data JPA para almacenar y gestionar datos de listados en PostgreSQL.
  - RESTful API para exponer funcionalidades a otros microservicios y aplicaciones externas.

- **Monitoreo** Monitorear el rendimiento y la salud de la plataforma para detectar y solucionar problemas rápidamente.