Examen proyecto Quarkus

Serie I:

La Serie I consiste en el desarrollo inicial de un proyecto utilizando el framework Quarkus, enfocado en la creación de un servicio REST básico. El objetivo principal es implementar un endpoint funcional que permita responder solicitudes HTTP con datos en formato JSON, utilizando anotaciones estándar de Jakarta REST (como @GET y @Path). Además, se integra la serialización de objetos Java a JSON mediante Jackson y se preparan pruebas unitarias con JUnit5 y RestAssured para asegurar el correcto funcionamiento del servicio. Esta primera fase establece la base para futuros desarrollos y mejoras en la aplicación.

El funcionamiento básico consiste en que el cliente realiza una petición HTTP al servidor, el recurso REST maneja dicha petición y responde con datos estructurados, permitiendo la comunicación entre aplicaciones de manera estandarizada. Además, se incorporan pruebas automatizadas con JUnit 5 para verificar la lógica del servicio y con RestAssured para validar las respuestas HTTP, asegurando la calidad y estabilidad del servicio.

Esta fase es fundamental para establecer una base sólida sobre la cual se podrán construir funcionalidades más complejas en etapas posteriores del proyecto.

Serie II:
Descripción

Este proyecto es una aplicación backend desarrollada con Quarkus que gestiona información de personas. Utiliza Hibernate ORM con Panache para la persistencia de datos y expone una API REST utilizando RESTEasy Reactive. Además, se conecta a una base de datos MySQL para almacenar la información.

Características principales

- API REST para registrar y consultar personas.

- Uso de DTOs y mapeo entre entidades y objetos de transferencia.

- Conexión a base de datos MySQL mediante JPA.

- Ejecutable con `mvn quarkus:dev`.

Estructura del Proyecto

```
org.example.persona
├── controller
│   └── PersonaResource.java
├── dto
│   └── PersonaDto.java, Address.java
├── entity
│   └── Persona.java, Address.java
├── mapper
│   └── PersonaMapper.java
└── repository
    └── PersonaRepository.java
```

Evidencia de Ejecución Correcta

```
quarkus-persona> mvn quarkus:dev

[INFO] Scanning for projects...

[INFO] ----------------------< org.example:quarkus-persona >----------------------

[INFO] Building quarkus-persona 1.0.0

[INFO] --- quarkus-maven-plugin:3.2.4.Final:dev (default-cli) @ quarkus-persona ---

[INFO] Press Ctrl+C to stop.

___  ____  __  _____   ___  __ ____

/ _ \/ __\/  |/  / _ | / _ \/ //_/

/ ___/ /__/ /|_/ / __ |/ , _/ ,<

/_/   \___/_/  /_/_/ |_/_/|_/_/|_|

[INFO] Installed features: [cdi, hibernate-orm-panache, jdbc-mysql, resteasy-reactive]

[INFO] Listening on: http://localhost:8080

[INFO] Profile dev activated. Live Coding activated.

[INFO] Application started (quarkus) in 2.345s. Listening on: http://localhost:8080
```

Terminal - Quarkus Running on http://localhost:8080

INFO [io.quarkus] (main) quarkus-persona 1.0.0-SNAPSHOT running on JVM
INFO [io.quarkus] (main) Profile dev activated. Live Coding activated.
INFO [io.quarkus] (main) Installed features: [cdi, resteasy-reactive, hibernate-orm, jdbc-mysql]

> GET http://localhost:8080/persons


```
[
 {
  "id": 1,
  "nombre": "Juan",
  "addresses": [
   {
    "id": 1,
    "calle": "5a Avenida",
    "ciudad": "Guatemala"
   }
  ]
 }
]
```


> GET http://localhost:8080/persons/1/addresses


```
[
 {
  "id": 1,
  "calle": "5a Avenida",
  "ciudad": "Guatemala"
 }
]
```