**WEEK-5**

**Exercises on Microservices with Spring Boot 3.0**

**1. Build a User and Order Management System**

**UserClient.java**

package com.example.order\_service.client;  
  
import com.example.order\_service.dto.UserDTO;  
import org.springframework.cloud.openfeign.FeignClient;  
import org.springframework.web.bind.annotation.\*;  
  
@FeignClient(name = "user-service", url = "http://localhost:8081")  
public interface UserClient {  
 @GetMapping("/users/{id}")  
 UserDTO getUserById(@PathVariable("id") Long id);  
}

**OrderController.java**

package com.example.order\_service.controller;  
  
import com.example.order\_service.client.UserClient;  
import com.example.order\_service.dto.UserDTO;  
import com.example.order\_service.model.Order;  
import com.example.order\_service.repository.OrderRepository;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.\*;  
  
@RestController  
@RequestMapping("/orders")  
public class OrderController {  
  
 private final OrderRepository orderRepo;  
 private final UserClient userClient;  
  
 public OrderController(OrderRepository orderRepo, UserClient userClient) {  
 this.orderRepo = orderRepo;  
 this.userClient = userClient;  
 }  
  
 @PostMapping  
 public Order placeOrder(@RequestBody Order order) {  
 return orderRepo.save(order);  
 }  
  
 @GetMapping("/{id}")  
 public Map<String, Object> getOrderWithUser(@PathVariable Long id) {  
 Order order = orderRepo.findById(id).orElse(null);  
 if (order == null) return Map.*of*("error", "Order not found");  
  
 UserDTO user = userClient.getUserById(order.getUserId());  
  
 return Map.*of*(  
 "order", order,  
 "user", user  
 );  
 }  
  
 @GetMapping  
 public List<Order> getAllOrders() {  
 return orderRepo.findAll();  
 }  
}

**UserDTO.java**

package com.example.order\_service.dto;  
  
public class UserDTO {  
 private Long id;  
 private String name;  
 private String email;  
  
 public UserDTO() {  
 }  
  
 public UserDTO(Long id, String name, String email) {  
 this.id = id;  
 this.name = name;  
 this.email = email;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 @Override  
 public String toString() {  
 return "UserDTO{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", email='" + email + '\'' +  
 '}';  
 }  
}

**Order.java**

package com.example.order\_service.model;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Order {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private Long userId;  
 private String product;  
 private double price;  
  
 public Order() {  
 }  
  
 public Order(Long id, Long userId, String product, double price) {  
 this.id = id;  
 this.userId = userId;  
 this.product = product;  
 this.price = price;  
 }  
 public Long getId() {  
 return id;  
 }  
 public void setId(Long id) {  
 this.id = id;  
 }  
 public Long getUserId() {  
 return userId;  
 }  
 public void setUserId(Long userId) {  
 this.userId = userId;  
 }  
 public String getProduct() {  
 return product;  
 }  
 public void setProduct(String product) {  
 this.product = product;  
 }  
 public double getPrice() {  
 return price;  
 }  
 public void setPrice(double price) {  
 this.price = price;  
 }  
}

**OrderRepository.java**

package com.example.order\_service.repository;  
  
import com.example.order\_service.model.Order;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface OrderRepository extends JpaRepository<Order, Long> {  
}

**OrderServiceApplication.java**

package com.example.order\_service;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.openfeign.EnableFeignClients;  
  
@SpringBootApplication  
@EnableFeignClients  
public class OrderServiceApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(OrderServiceApplication.class, args);  
 }  
}

**pom.xml**

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/>  
 </parent>  
  
 <groupId>com.example</groupId>  
 <artifactId>order-service</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>order-service</name>  
 <description>Order microservice</description>  
  
 <properties>  
 <java.version>17</java.version>  
 <spring-cloud.version>2022.0.4</spring-cloud.version>  
 </properties>  
  
 <dependencies>  
 <!-- Web and REST -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.32</version> <!-- Or latest -->  
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>com.mysql</groupId>  
 <artifactId>mysql-connector-j</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-openfeign</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
  
 <!-- Testing -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <dependencyManagement>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-dependencies</artifactId>  
 <version>${spring-cloud.version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 </dependencies>  
 </dependencyManagement>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**UserController.java**

package com.example.user\_service.controller;  
  
import com.example.user\_service.model.User;  
import com.example.user\_service.repository.UserRepository;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/users")  
public class UserController {  
  
 private final UserRepository repo;  
  
 public UserController(UserRepository repo) {  
 this.repo = repo;  
 }  
  
 @PostMapping  
 public User saveUser(@RequestBody User user) {  
 return repo.save(user);  
 }  
  
 @GetMapping("/{id}")  
 public User getUser(@PathVariable Long id) {  
 return repo.findById(id).orElse(null);  
 }  
  
 @GetMapping  
 public List<User> getAllUsers() {  
 return repo.findAll();  
 }  
}

**User.java**

package com.example.user\_service.model;  
  
import jakarta.persistence.\*;  
import lombok.Data;  
import lombok.NoArgsConstructor;  
import lombok.AllArgsConstructor;  
  
@Entity  
@Data  
@NoArgsConstructor  
@AllArgsConstructor  
public class User {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
 private String email;  
}

**UserRepository.java**

package com.example.user\_service.repository;  
  
import com.example.user\_service.model.User;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface UserRepository extends JpaRepository<User, Long> {  
}

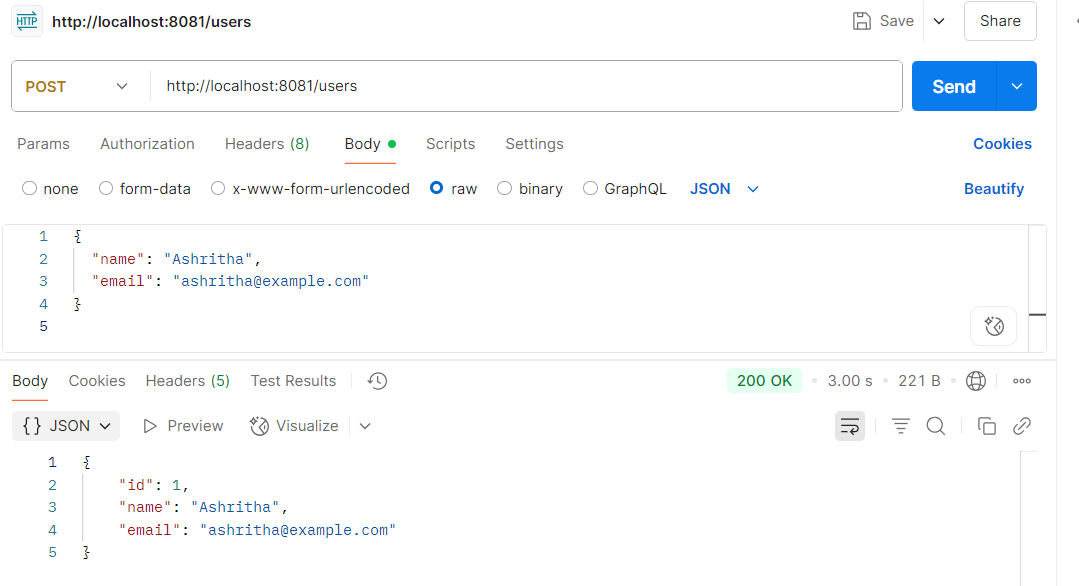
**UserServiceApplication.java**

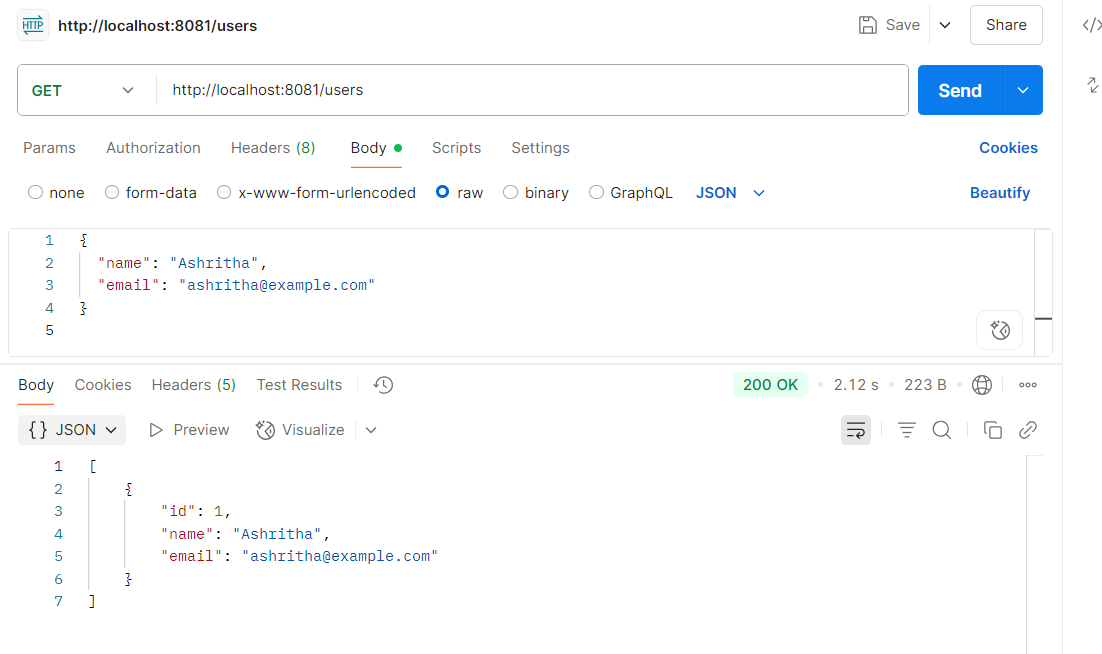
package com.example.user\_service;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class UserServiceApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(UserServiceApplication.class, args);  
 }  
  
}

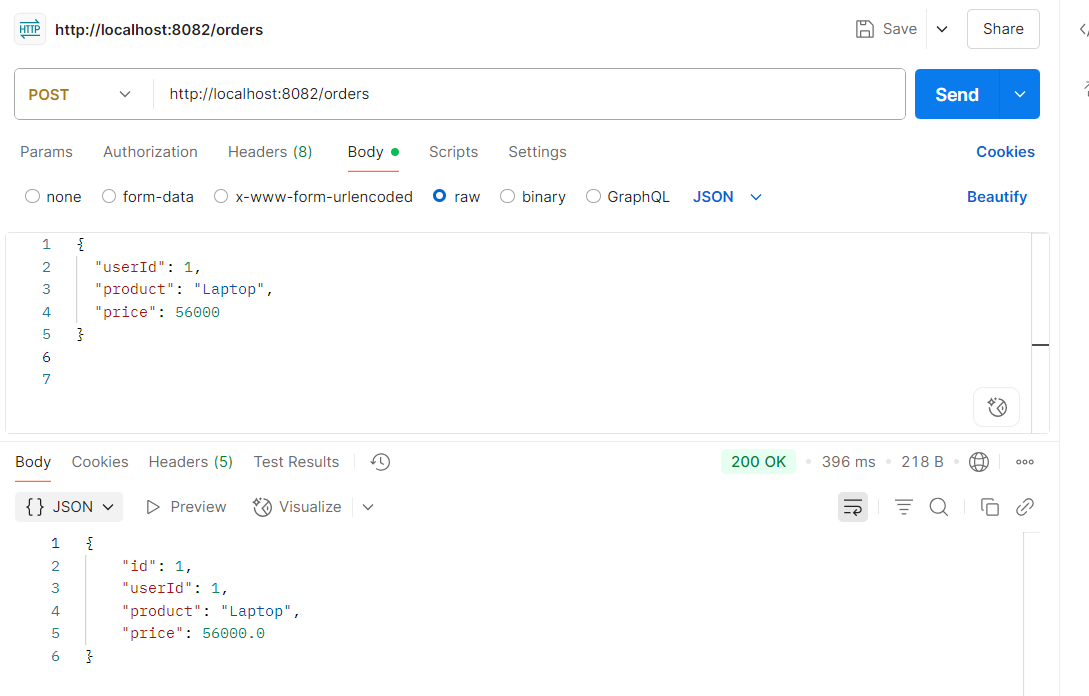
**pom.xml**

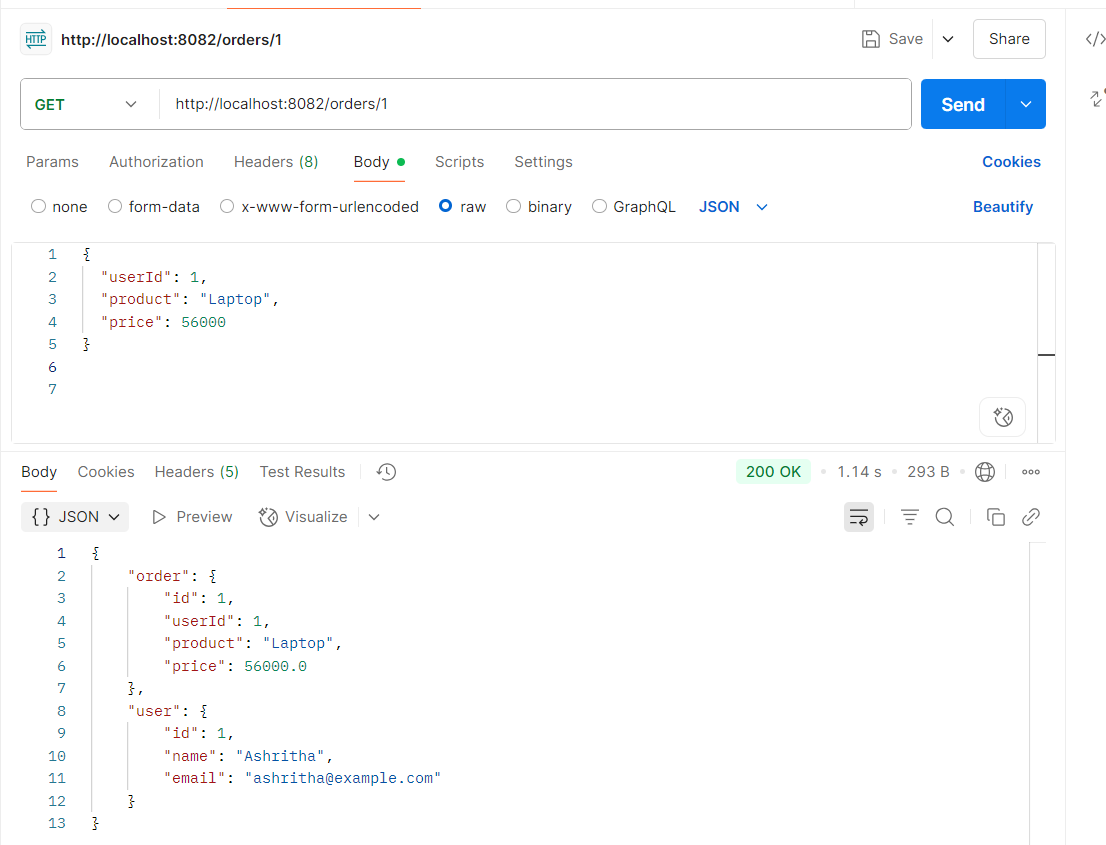
<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.5.3</version>  
 <relativePath/>   
 </parent>  
 <groupId>com.example</groupId>  
 <artifactId>user-service</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>user-service</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>17</java.version>  
 <spring-cloud.version>2025.0.0</spring-cloud.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.32</version>   
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-openfeign</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>com.mysql</groupId>  
 <artifactId>mysql-connector-j</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
 <dependencyManagement>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-dependencies</artifactId>  
 <version>${spring-cloud.version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 </dependencies>  
 </dependencyManagement>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**Output:**

****







**2. Inventory Management System with Service Discovery**

**EurekaServerApplication.java**

package com.example.eureka\_server;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;  
@SpringBootApplication  
@EnableEurekaServer  
public class EurekaServerApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(EurekaServerApplication.class, args);  
 }  
}

**application.properties**

spring.application.name=eureka-server

**application.yml**

server:  
 port: 8761  
  
eureka:  
 client:  
 register-with-eureka: false  
 fetch-registry: false

**ConfigServerApplication.java**

package com.example.config\_server;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.config.server.EnableConfigServer;  
  
@SpringBootApplication  
@EnableConfigServer   
public class ConfigServerApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(ConfigServerApplication.class, args);  
 }  
}

**application.properties**

spring.application.name=config-server

**application.yml**

server:  
 port: 8888  
  
spring:  
 cloud:  
 config:  
 server:  
 git:  
 uri: file:///${user.home}/config-repo

**inventory-service.yml**

spring:  
 datasource:  
 url: jdbc:mysql://localhost:3306/inventorydb  
 username: root  
 password: Ashritha@2  
 jpa:  
 hibernate:  
 ddl-auto: update  
 show-sql: true

**product-service.yml**

spring:  
 datasource:  
 url: jdbc:mysql://localhost:3306/productdb  
 username: root  
 password: Ashritha@2  
 jpa:  
 hibernate:  
 ddl-auto: update  
 show-sql: true

**InventoryController.java**

package com.example.inventory\_service.controller;  
  
import com.example.inventory\_service.model.Inventory;  
import com.example.inventory\_service.repository.InventoryRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/inventory")  
public class InventoryController {  
 @Autowired  
 private InventoryRepository repository;  
  
 @PostMapping  
 public Inventory create(@RequestBody Inventory inventory) {  
 return repository.save(inventory);  
 }  
  
 @GetMapping("/{productId}")  
 public Inventory getByProductId(@PathVariable Long productId) {  
 return repository.findByProductId(productId);  
 }  
}

**Inventory.java**

package com.example.inventoryservice.model;  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.GeneratedValue;  
import jakarta.persistence.GenerationType;  
import jakarta.persistence.Id;  
import jakarta.persistence.Table;  
  
@Entity  
@Table(name = "inventory")  
public class Inventory {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
  
 private Long productId;  
 private int availableStock;  
  
 public Inventory() {}  
  
 public Inventory(Long id, Long productId, int availableStock) {  
 this.id = id;  
 this.productId = productId;  
 this.availableStock = availableStock;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public Long getProductId() {  
 return productId;  
 }  
  
 public void setProductId(Long productId) {  
 this.productId = productId;  
 }  
  
 public int getAvailableStock() {  
 return availableStock;  
 }  
  
 public void setAvailableStock(int availableStock) {  
 this.availableStock = availableStock;  
 }  
}

**InventoryRepository.java**

package com.example.inventory\_service.repository;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
import com.example.inventory\_service.model.Inventory;  
  
public interface InventoryRepository extends JpaRepository<Inventory, Long> {  
 Inventory findByProductId(Long productId);  
}

**InventoryServiceApplication.java**

package com.example.inventory\_service;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class InventoryServiceApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(InventoryServiceApplication.class, args);  
 }  
  
}

**ProductController.java**

package com.example.product\_service.controller;  
  
import com.example.product\_service.model.Product;  
import com.example.product\_service.repository.ProductRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/products")  
public class ProductController {  
  
 @Autowired  
 private ProductRepository repository;  
  
 @PostMapping  
 public Product create(@RequestBody Product product) {  
 return repository.save(product);  
 }  
  
 @GetMapping  
 public List<Product> getAll() {  
 return repository.findAll();  
 }  
}

**Product.java**

package com.example.product\_service.model;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Product {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
 private String name;  
 private int quantity;  
  
 // Constructors  
 public Product() {  
 }  
  
 public Product(Long id, String name, int quantity) {  
 this.id = id;  
 this.name = name;  
 this.quantity = quantity;  
 }  
  
 // Getters and Setters  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getQuantity() {  
 return quantity;  
 }  
  
 public void setQuantity(int quantity) {  
 this.quantity = quantity;  
 }  
}

**ProductRepository.java**

package com.example.product\_service.repository;  
  
import com.example.product\_service.model.Product;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface ProductRepository extends JpaRepository<Product, Long> {  
}

**ProductServiceApplication.java**

package com.example.product\_service;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class ProductServiceApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(ProductServiceApplication.class, args);  
 }  
  
}