Nama: Pradana Argo Pangestu

Nim : 2311104079

MODUL 10

- 1. Membuat Database dengan nama Penjualan
 - a. Buat tabel dengan nama Product dengan struktur berikut



b. Isi Tabel Product



2. Membuat Konfigurasi DB pada Spring Boot

```
application.properties ×

1 spring.application.name=rest-api
2 spring.datasource.url=jdbc:mysql://localhost:3306/penjualan
3 spring.datasource.username=root
4 spring.datasource.password=
5 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
6 spring.jpa.hibernate.ddl-auto=update
7 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
```

Gambar berikut merupakan pengaturan dari aplikasi Spring Boot dengan nama "restapi", dengan koneksi ke database MySQL dengan nama "penjualan" yang berjalan di localhost pada port 3306. Aplikasi menggunakan username "root" tanpa password untuk mengakses database, serta memanfaatkan driver JDBC MySQL (com.mysql.cj.jdbc.Driver).Hibernate dikonfigurasi untuk secara otomatis memperbarui skema database sesuai dengan entitas JPA melalui opsi ddl-auto=update. Selain itu, digunakan dialek Hibernate MySQL8Dialect agar query yang dihasilkan sesuai dengan sintaks MySQL versi 8.

3. Membuat Model Entity

```
arsitektur - springbootrestapi/rest-api/src/main/java/com/example/restapi/model/Product.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
■ Productjava × RestApiAppl... Springbootr...
                                                                                                package com.example.restapi.model;
               example
        M HELP.md
                                                        3 import jakarta.persistence.*;
        mvnw
                                                       5 @Entity
                                                                class Product {
        lmx.mog
                                                       6 public
                                                              @Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
      ∨ ⊜ main
         java

∨ Decom

             example

→ Controller

                                                            public Product(String name, double price) {
                                                                                                                                                 <sup>①</sup> Con
                     ProductController.java
                                                      16
17
18
19
20°
21
22
23
24°
25
26
                                                                 this.price = price;

∨ De model

                     Product.java
                                                                                                                                                   or c
                 > 🍅 repository
                                                                                                                                                 E Outl
                 > 🍃 service
                  RestApiApplication.java
                                                                                                                                                 ~ @ F
         resources
                                                             public void setId(Long id) {
                                                                this.id = id;
       > 🗁 test
    target

∨ 

⇔ classes

                                                             public String getName() {
                                                            }
        > META-INE
           application.properties
                                                            public void setName(String name) {

∨ 
  b test-classes

                                                                 this.name = name;
                                                           }
         > 😂 com
      HELP.md
                                                            public double getPrice() {
      mvnw
      pom.xml
                                                      Problems @ Javadoc Declaration Console × Console
                                                       terminated> rest-api [Maven Build] C\Users\pradana\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_23.0.2.v
```

Gambar berikut nama class Product yang merepresentasikan entitas produk dalam aplikasi. Class ini akan disimpan sebagai tabel di database karena diberi anotasi @Entity. Setiap produk memiliki tiga atribut utama: id sebagai identitas unik (yang akan dibuat otomatis oleh database), name sebagai nama produk, dan price sebagai harga produk. Class ini juga menyediakan konstruktor default, konstruktor dengan parameter, serta getter dan setter untuk mengakses dan mengubah nilai dari masingmasing atribut.

4. Membuat Repository

```
FIIE EGIT SOURCE RETACTOR INAVIGATE SEARCH PROJECT KUN WINGOW HEID
🖹 😩 🕼 🖁 🗖 🗎 🚇 Product.java 🔛 springbootr... 🖼 springbootr...
                                                                              mvnw mvnw.cmd ProductRepos... × "?
□ Package Explorer ×
           > @ evamnle
                                             1 package com.example.restapi.repository;
      HELP.md
                                             3⊕ import com.example.restapi.model.Product;
      mvnw
       mvnw.cmd
       8 public interface ProductRepository extends JpaRepository<Product, Long> {
     🗸 🗁 main
       🗸 🗁 java

∨ Decom

          example
            restapi
              ProductController.iava
              🗸 🗁 model
                 Product.java
              repository
                 ProductRepository.java
              > 🗁 service
                RestApiApplication.java
```

Gambar berikut merupakan interface ProductRepository yang berfungsi sebagai penghubung antara aplikasi dan database untuk entitas Product. Dengan mewarisi JpaRepository, interface ini secara otomatis memiliki fitur bawaan seperti menyimpan, mencari, menghapus, dan memperbarui data produk tanpa perlu menulis kode SQL secara manual. Anotasi @Repository menunjukkan bahwa ini adalah komponen yang berperan dalam pengelolaan data di Spring.

5. Membuat Service Layar

```
E 😂 🔝 🖁 🗖 🖟 Product.java 🗎 springbootr... 🗎 springbootr...
                                                                                                    ackage Explorer ×
                                                       1 package com.example.restapi.service;
                      ♠ ProductService.class
                   ♠ RestApiApplication.class
                                                       3®import com.example.restapi.model.Product;□
        resources
            application.properties
                                                      10 @Service
      > 🗁 test
                                                      11 public class ProductService {
                                                              @Autowired
   target
                                                             private ProductRepository repository;
      > 🗁 com
                                                             public List<Product> getAllProducts() {
        > META-INF
                                                      16
17
18
                                                                return repository.findAll();
          application properties
      public Optional<Product> getProductById(Long id) {
                                                      19⊜
        ∨ 🗁 com
                                                                  return repository.findById(id);
           > 🗁 example
                                                      21
22

    HELP.md

                                                              public Product saveProduct(Product product) {
                                                      23<sup>©</sup>
24
25
26
27<sup>©</sup>
      mvnw
                                                                  return repository.save(product);
      mvnw.cmd
      lmx.mog
 public Product updateProduct(Long id, Product productDetails) {
                                                      28
29
30
31
                                                                  return repository.findById(id).map(product -> {
    product.setName(productDetails.getName());

→ P main

      🗸 🗁 java
                                                                       product.setPrice(productDetails.getPrice());
                                                                       return repository.save(product);
          🗸 🗁 example
                                                      32
33
34
                                                                  }).orElse(null);
             🗸 🗁 restapi
                                                             }
               35⊜
36
                                                              public void deleteProduct(Long id) {
                   ProductController.iava
                                                                  repository.deleteById(id);

∨ Description > model

                   Product.java
               ProductRepository.java
               ProductService.java
```

Gambar berikut merupakan class ProductService yang berfungsi sebagai lapisan layanan (service layer) dalam aplikasi untuk mengatur logika bisnis terkait data Product. Class ini menggunakan ProductRepository untuk berinteraksi dengan database. Dengan bantuan anotasi @Service, Spring mengenali class ini sebagai komponen service. Di dalamnya terdapat beberapa method utama, seperti mengambil semua produk (getAllProducts), mengambil produk berdasarkan ID (getProductById), menyimpan produk baru (saveProduct), memperbarui data produk (updateProduct), dan menghapus produk berdasarkan ID (deleteProduct). Semua operasi ini menggunakan method bawaan dari JpaRepository, sehingga memudahkan pengelolaan data tanpa perlu menulis query secara manual.

6. Membuat Controller

```
₽ Package Explorer ×

    ⊕ ProductService.class
    ↓ 1 package com.example.restapi.controller;
    ⊕ RestApiApplication.class
    ↓
    ↓
    ☐ 2 package com.example.restapi.controller;
    ↓
    ☐ 2 package com.example.restapi.controller;
    ☐ 3 package com.example.restapi.controller;
    ☐ 4 package com.example.restapi.controller;
    ☐ 5 package com.example.restapi.controller;
    ☐ 6 package com.example.restapi.controlle
                                                                                                                                                               3 import com.example.restapi.model.Product;

✓ ➢ resources

                                        application.properties
                                                                                                                                                          11 @RestController
12 @RequestMapping("/products")
13 public class ProductController {
                        > @ test
                  @Autowired
                        15
                                                                                                                                                                                 private ProductService service;
                                                                                                                                                                             @GetMapping
public List<Product> getAll() {
    return service.getAllProducts();
}
                              > 🍅 META-INF
                                                                                                                                                            18
19
20
                                   application.properties

∨ Ø com

                                                                                                                                                            21
22⊜
                                                                                                                                                                                 @GetMapping("/{id}")
public ResponseEntity<Product> getById(@PathVariable Long id) {
    Optional<Product> product = service.getProductById(id);
    return product.map(ResponseEntity::ok).orElseGet(() ->
                                      > 🗁 example

    HELP.md

                                                                                                                                                            23
24
25
26
27
28
                        mvnw.cmd
                                                                                                                                                                                                      ResponseEntity.notFound().build());

    pom.xml

✓ 
➢ src

                                                                                                                                                                                @PostMapping
public Product create(@RequestBody Product product) {
    return service.saveProduct(product);
                  🗸 🗁 main
                                                                                                                                                            29<sup>®</sup>
30
31
32
33
34<sup>®</sup>
35
                        🗸 🗁 java
                              ∨ ⊜ com
                                           🗸 🗁 restapi
                                                                                                                                                                                 @PutMapping("/{id}")
                                                                                                                                                                                 public ResponseEntity<Product> update(@PathVariable Long id, @RequestBody Produ
                                                Product updatedProduct = service.updateProduct(id, productDetails);
return updatedProduct != null ? ResponseEntity.ok(updatedProduct) :
                                                            ProductController.java

∨ Description → model

                                                                                                                                                                                                      ResponseEntity.notFound().build();
                                                                                                                                                            38
                                                           Product.java
                                                 @DeleteMapping("/{id}")
public ResponseEntity<Void> delete(@PathVariable Long id) {
    service deleteProduct(id):
                                                             ProductRepository.java
                                                                                                                                                            41⊖
                                                  ProductService.iava
```

gambar berikut merupakan class ProductController yang berfungsi sebagai pengatur jalur (controller) dalam aplikasi REST API untuk menangani permintaan HTTP terkait

data produk. Dengan anotasi @RestController dan @RequestMapping("/products"), class ini menangani semua permintaan ke endpoint /products. Melalui metode-metode seperti getAll, getById, create, update, dan delete, controller ini memfasilitasi operasi CRUD (Create, Read, Update, Delete) pada data produk. Setiap permintaan akan diteruskan ke ProductService untuk diproses, lalu mengembalikan respons yang sesuai kepada client, baik itu berupa data produk, pesan sukses, atau status "tidak ditemukan".

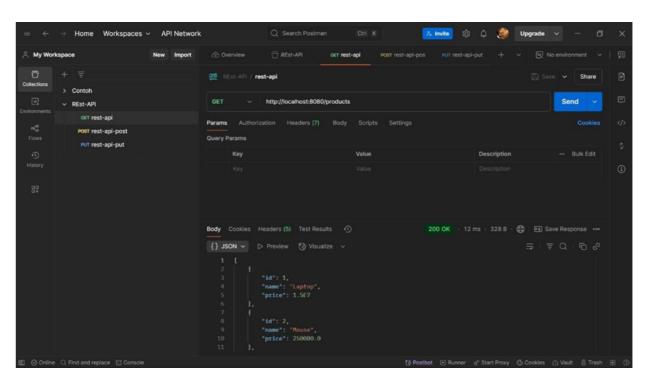
7. Menjalankan Spring Boot

```
■ Package Explorer ×
                                                                                                                                                                                                                     🖹 😂 🐷 🖟 🗖 🗖 Product.java 🖟 RestApiAppl... × 🖟 ProductRepos... 🐪 🖟 ProductServ... 🖟 ProductCont...
                                                                             example 🗠
                                                                                                                                                                                                                                                                                         1 package com.example.restapi;
                                            HELP.md
                                                                                                                                                                                                                                                                                                      3*import org.springframework.boot.SpringApplication;
                                              mvnw
                                              mvnw cmd
                                                                                                                                                                                                                                                                                                      6 @SpringBootApplication
                                                                                                                                                                                                                                                                                                              public class RestApiApplication {
   public static void main(String[] args) {
        SpringApplication.run(RestApiApplication.class, args);
}
                                              lmx,mog 📓
                                   main
                                              🗸 🗁 iava
                                                          ∨ ⊜ com
                                                                     restapi

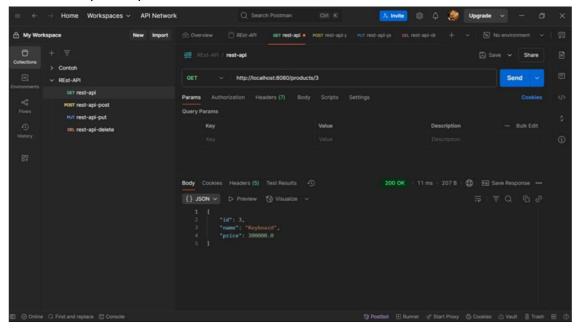
∨ Description → Descripti
                                                                                                                   Product.java
                                                                                               repository
                                                                                                                  ProductRepository.iava
                                                                                                        service
                                                                                                                   ProductService.iava
                                                                                                       RestApiApplication.java
```

Gambar berikut merupakan class utama RestApiApplication yang menjadi titik awal (entry point) untuk menjalankan aplikasi Spring Boot. Dengan anotasi @SpringBootApplication, Spring akan secara otomatis mengkonfigurasi dan memulai seluruh komponen aplikasi, termasuk pemindaian komponen (component scan), konfigurasi otomatis, dan inisialisasi konteks aplikasi. Method main memanggil SpringApplication.run, yang akan menjalankan aplikasi dan mengaktifkan server web agar siap menerima permintaan (request) dari client.

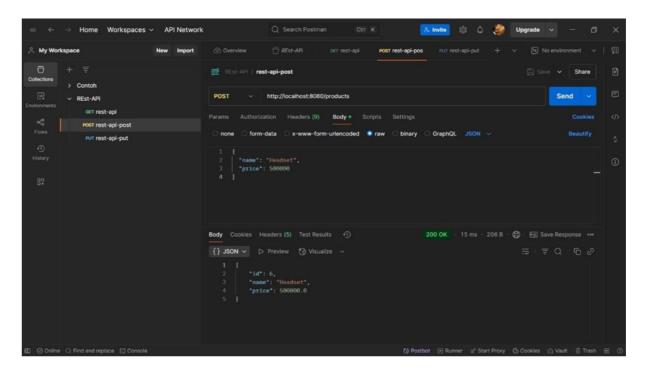
8. GET – Mendapatkan semua product



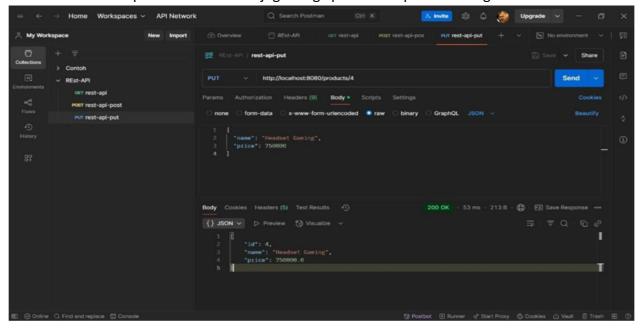
9. GET- Mendapatkan product berdasarkan Id

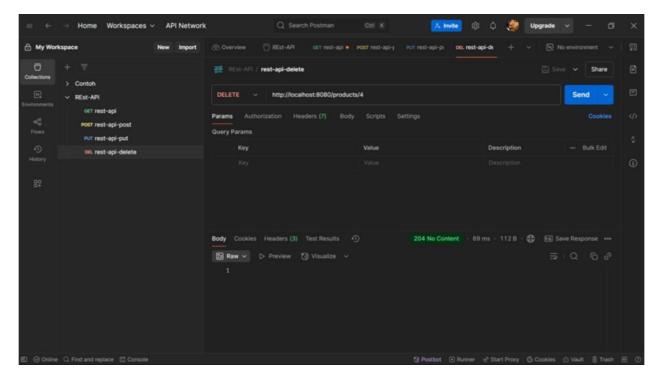


10. POST – menambahkan product baru yang dinamai Headset

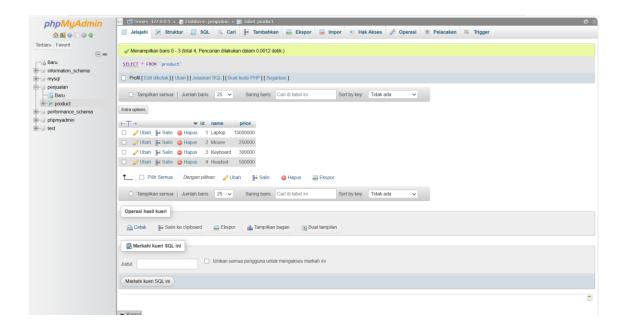


11. PUT – memperbarui nama dan juga harga pada tabel product dengan id 4





12. DEL – Menghapus isi product yang idnya 4



Kesimpulannya adalah untuk mengetahui bagaimana cara mengimplementasikan rest-api menggunakan spring pada eclipse, dan juga mengetahui cara penggunaan mysql dengan eclipse. juga implementasi apa saja yang digunakan untuk menyambungkan dan membuat api.