LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: RPL 2
Kelas	: 4IA01
Praktikum ke-	: 5
Tanggal	: 13 November 2025
Materi	: Spring
NPM	: 50422657
Nama	: Harry Mardika
Ketua Asisten	: Satya Bara Justitia
Paraf Asisten	: -
Nama Asisten	: 1.
	2.
	3.
Jumlah Lembar	: 7 Lembar

LABORATORIUM INFORMATIKA

UNIVERSITAS GUNADARMA

2025

Soal

1. Jelaskan apa itu Spring Boot dan bagaimana perbedaannya dengan kode pada pertemuan sebelumnya. Apa keuntungan yang ditawarkan Spring Boot bagi pengembangan aplikasi?

2. Jelaskan kode dan langkah - langkah program yang telah dibuat!

Penjelasan

---

1. Spring Boot adalah sebuah proyek dalam ekosistem Spring yang dirancang untuk menyederhanakan proses penyiapan dan pengembangan aplikasi Java yang standalone (mandiri) dan siap produksi, dengan fokus utama pada layanan web dan arsitektur microservices. Perbedaannya dengan kode pada pertemuan sebelumnya sangat fundamental. Kode pada pertemuan sebelumnya adalah aplikasi desktop (Java Swing) yang memerlukan konfigurasi manual dan eksplisit untuk setiap komponen. Pada program sebelumnya diharuskan membuat kelas HibernateUtil secara manual untuk membangun SessionFactory (kemungkinan dari file hibernate.cfg.xml), dan di dalam MahasiswaControllerImpl, diharuskan mengelola siklus hidup Session dan Transaction (beginTransaction(), commit(), rollback()) secara manual untuk setiap operasi database (CRUD). Sebaliknya, Spring Boot menganut prinsip "Convention over Configuration" (konvensi lebih diutamakan daripada konfigurasi). Dengan starter seperti spring-boot-starter-data-jpa, Spring Boot akan mengkonfigurasi DataSource dan EntityManager (pengganti SessionFactory di JPA) secara otomatis hanya dengan menambahkan properti sederhana (seperti URL database, username, dan password) di file application.properties. Lebih lanjut, alih-alih menulis implementasi MahasiswaControllerImpl secara manual, programmer cukup mendefinisikan sebuah interface (misalnya MahasiswaRepository) yang mewarisi JpaRepository. Spring Boot kemudian akan secara ajaib menyediakan implementasi untuk semua metode CRUD dasar (save, findById, findAll, delete) secara otomatis.

Keuntungan utama yang ditawarkan Spring Boot adalah pengembangan yang sangat cepat (rapid development) berkat starters (paket dependensi siap pakai) dan autoconfiguration. Hal ini secara drastis mengurangi boilerplate code (kode berulang) yang harus seorang programmer tulis; kelas seperti HibernateUtil dan seluruh logika manajemen transaksi manual di controller menjadi tidak diperlukan lagi, digantikan oleh konfigurasi otomatis dan anotasi sederhana (seperti @Transactional). Selain itu, Spring Boot menyertakan embedded server (seperti Tomcat atau Jetty) di dalamnya, sehingga aplikasi dapat dijalankan sebagai file .jar mandiri (menggunakan perintah java -jar) tanpa perlu menginstal atau mengkonfigurasi server web eksternal, yang sangat menyederhanakan proses deployment dan pengujian.

2.

- Overview

Program ini adalah aplikasi konsol (console application) sederhana yang dibuat menggunakan Spring Boot. Tujuannya adalah untuk mengelola data mahasiswa (melakukan operasi CRUD - Create, Read, Update, Delete, meskipun yang diimplementasikan di sini baru Create dan Read). Aplikasi ini menggunakan Spring Data JPA untuk berinteraksi dengan database MySQL. Interaksi dengan pengguna dilakukan melalui menu teks di konsol.

- Arsitektur & Komponen Utama

Aplikasi ini dibagi menjadi beberapa lapisan (layers) yang saling berhubungan:

1) Model (ModelMahasiswa.java): Mendefinisikan struktur data (objek) mahasiswa. Ini adalah "cetakan" untuk data yang akan disimpan ke database.
2) Repository (MahasiswaRepository.java): Bertugas sebagai jembatan antara aplikasi dan database. Ini adalah lapisan Data Access Object (DAO) yang disediakan secara otomatis oleh Spring Data JPA.
3) Controller (MahasiswaController.java): Mengandung logika bisnis aplikasi. Ini yang mengurus "apa yang harus dilakukan" ketika pengguna memilih menu, seperti mengambil data atau menyimpan data baru.
4) Entry Point (Pertemuan5SpringBootApplication.java): Kelas utama yang menjalankan aplikasi Spring Boot dan memicu tampilan menu konsol.
5) Konfigurasi (pom.xml & application.properties):
    - pom.xml: Mendefinisikan dependensi/pustaka (library) apa saja yang dibutuhkan proyek (seperti Spring Data JPA, Driver MySQL).
    - application.properties: Mengatur koneksi ke database dan konfigurasi JPA/Hibernate.

- Penjelasan Kode per File

Berikut adalah rincian fungsi dari setiap file yang telah dibuat pada activity:

1) pom.xml (File Konfigurasi Proyek Maven)

   File ini memberi tahu Maven (alat bantu build) dependensi apa yang dibutuhkan proyek.

    - <parent>...spring-boot-starter-parent...</parent>: Menginduk ke konfigurasi default Spring Boot, ini menyederhanakan banyak hal.
    - spring-boot-starter-data-jpa: Dependensi inti untuk menggunakan JPA (Java Persistence API) dengan Spring. Ini sudah termasuk Hibernate di dalamnya.
    - mysql-connector-j: Driver JDBC yang spesifik agar Java bisa "berbicara" dengan database MySQL.
    - spring-boot-starter-web: Meskipun ini aplikasi konsol, dependensi ini sering ditambahkan untuk fitur Spring Boot terkait web atau jika akan dikembangkan menjadi aplikasi web.
    - spring-boot-maven-plugin: Plugin untuk mem-paket aplikasi menjadi satu file .jar yang bisa dieksekusi.
2) application.properties (File Konfigurasi Aplikasi)

   File ini berisi pengaturan untuk aplikasi yang telah dibuah.

    - spring.datasource.url=...: URL koneksi ke database. Di sini, ia terhubung ke localhost:3306 (port default MySQL) dengan nama database pert4_rpl2.
    - spring.datasource.username=root: Username untuk login ke database.
    - spring.datasource.password=: Password untuk login (dalam kasus ini, kosong).
    - spring.jpa.hibernate.ddl-auto=update: Penting. Perintah ini menyuruh Hibernate (implementasi JPA) untuk secara otomatis memperbarui skema database (misal: membuat tabel, menambah kolom) agar sesuai dengan ModelMahasiswa.java setiap kali aplikasi dimulai.
    - spring.jpa.show-sql=true: Menampilkan query SQL yang dieksekusi oleh JPA di konsol. Ini sangat membantu untuk debugging.

3) ModelMahasiswa.java (Model / Entity)

File ini adalah representasi Java dari tabel mahasiswa di database.

- @Entity: Menandakan bahwa kelas ini adalah entitas JPA (akan dipetakan ke tabel).
- @Table(name = "mahasiswa"): Secara eksplisit memberi tahu JPA bahwa kelas ini dipetakan ke tabel bernama "mahasiswa".
- @Id: Menandakan bahwa id adalah Primary Key (kunci unik) tabel.
- @GeneratedValue(strategy = GenerationType.IDENTITY): Mengatur agar nilai id dibuat secara otomatis oleh database (auto-increment).
- @Column(...): Memetakan variabel Java (npm, nama, dll.) ke kolom di tabel. Atribut seperti nullable = false dan length = 8 adalah batasan (constraint) untuk kolom tersebut.
- public ModelMahasiswa() {}: Konstruktor kosong. Ini wajib ada untuk JPA.
- public ModelMahasiswa(int id, ...): Konstruktor dengan parameter untuk memudahkan pembuatan objek baru di aplikasi.
- Getter/Setter & toString(): Metode standar Java untuk mengakses data dan mencetak informasi objek.

4) MahasiswaRepository.java (Repository / Data Access)

Ini adalah interface (bukan kelas) yang menjadi "sihir" dari Spring Data JPA.

- @Repository: Menandakan bahwa ini adalah komponen Spring yang bertugas untuk akses data.
- extends JpaRepository<ModelMahasiswa, Long>: Dengan memperluas (extend) JpaRepository, interface ini secara otomatis "mewarisi" banyak metode siap pakai seperti save(), findAll(), findById(), delete(), dll. Spring Boot akan membuat implementasi (kode) untuk interface ini saat runtime.

5) MahasiswaController.java (Controller / Logika Aplikasi)

Ini adalah otak dari aplikasi konsol.

- @Controller: Menandakan bahwa ini adalah komponen Spring (sebuah "Bean").
- @Autowired private MahasiswaRepository mahasiswaRepository;: Ini adalah Dependency Injection. Spring secara otomatis akan mencari "Bean" MahasiswaRepository yang sudah dibuat dan "menyuntikkannya" ke dalam variabel ini sehingga tidak perlu membuat objeknya secara manual (new MahasiswaRepository()).
- public void tampilkanMenu(): Metode utama yang berisi game loop aplikasi. Ia menggunakan Scanner untuk membaca input, do-while agar menu terus tampil (sampai opsi != 4), dan switch untuk memilih aksi.
- private void tampilkanSemuaMahasiswa(): Memanggil mahasiswaRepository.findAll() untuk mengambil semua data dari tabel mahasiswa dan mencetaknya.
- private void tambahMahasiswa(Scanner scanner): Meminta input pengguna, membuat objek ModelMahasiswa baru, lalu menyimpannya ke database menggunakan mahasiswaRepository.save(mahasiswa).

- private void cekKoneksi(): Metode sederhana untuk menguji koneksi. Ia mencoba mengambil data; jika berhasil, koneksi aman; jika gagal (melempar Exception), koneksi gagal.

6) Pertemuan5SpringBootApplication.java (Entry Point)

Ini adalah kelas yang dijalankan untuk memulai seluruh aplikasi.

- @SpringBootApplication: Anotasi "ajaib" yang menggabungkan tiga hal: @Configuration, @EnableAutoConfiguration, dan @ComponentScan. Intinya, ini memberi tahu Spring Boot untuk "mulai bekerja, cari semua komponen, dan konfigurasikan semuanya secara otomatis".
- implements CommandLineRunner: Interface ini memberi tahu Spring Boot: "Setelah semua konfigurasi selesai dan aplikasi siap, tolong jalankan metode run()".
- public static void main(String[] args): Metode main standar Java. SpringApplication.run(...) adalah perintah untuk memulai Spring Boot.
- @Autowired private MahasiswaController mhsController;: Sama seperti di controller, Spring menyuntikkan MahasiswaController yang sudah dibuat ke sini.
- @Override public void run(String... args): Metode ini akan dieksekusi setelah aplikasi Spring siap. Di sinilah pememanggilan mhsController.tampilkanMenu() untuk pertama kalinya, yang akan memulai menu konsol.

- Langkah-Langkah Eksekusi Program (Alur Kerja)

Berikut adalah urutan kejadian saat menjalankan program:

1) Program menjalankan kelas Pertemuan5SpringBootApplication (misalnya, klik "Run" di IDE).
2) Metode main dieksekusi, memanggil SpringApplication.run().
3) Spring Boot memulai:
   - Memindai (scan) semua kelas yang memiliki anotasi seperti @Controller, @Repository, @Entity.
   - Membaca application.properties untuk data koneksi database.
   - Membuat instance (Bean) dari MahasiswaRepository (implementasi otomatis).
   - Membuat instance (Bean) dari MahasiswaController dan menyuntikkan MahasiswaRepository ke dalamnya.
   - Membaca ModelMahasiswa dan (karena ddl-auto=update) memeriksa tabel mahasiswa di database MySQL. Jika tabelnya tidak ada, tabel itu akan dibuat.
   - Membuat instance (Bean) dari Pertemuan5SpringBootApplication dan menyuntikkan MahasiswaController ke dalamnya.
4) Setelah semua siap, Spring Boot melihat bahwa kelas utama mengimplementasikan CommandLineRunner.
5) Spring Boot memanggil metode run() pada Pertemuan5SpringBootApplication.
6) Metode run() memanggil mhsController.tampilkanMenu().
7) Menu konsol muncul di layar CLI ("Menu: 1. Tampilkan...").
8) Aplikasi sekarang berada dalam loop do-while di tampilkanMenu(), menunggu input.

9) Jika memilih '1' (Tampilkan):
   - switch akan memanggil tampilkanSemuaMahasiswa().
   - Metode itu memanggil mahasiswaRepository.findAll().
   - JPA mengirimkan query SELECT * FROM mahasiswa ke database.
   - Hasilnya dikonversi menjadi List<ModelMahasiswa> dan dicetak ke konsol.
10) Jika memilih '2' (Tambah):
   - switch akan memanggil tambahMahasiswa().
   - Lalu akan diminta memasukkan NPM, Nama, dsb.
   - Objek ModelMahasiswa baru dibuat.
   - Metode mahasiswaRepository.save(mahasiswa) dipanggil.
   - JPA mengirimkan query INSERT INTO mahasiswa (...) VALUES (...) ke database.
11) Jika memilih '4' (Keluar):
   - Kondisi do-while (opsi != 4) menjadi false.
   - Metode tampilkanMenu() selesai.
   - Metode run() selesai.
   - Program dan aplikasi Spring Boot berhenti.