ACTIVITY PERTEMUAN 1

NAMA : FADLI ARDIANSYAH HARAHAP

NPM : 50421437

KELAS : 4IA14

MATERI :

MATA PRAKTIKUM :

Dependency Injection (DI) adalah sebuah pola desain dalam pemrograman yang bertujuan untuk membuat kode lebih modular dan mudah di-maintain dengan memisahkan logika komponen dari ketergantungan yang dibutuhkan. DI bekerja dengan menyuntikkan (inject) objek atau komponen lain yang dibutuhkan oleh suatu objek, sehingga objek tersebut tidak perlu langsung membuat atau menginisialisasi dependensinya sendiri.

Dalam **Spring Framework** dan khususnya **Spring Boot**, Dependency Injection menjadi sangat penting karena beberapa alasan:

1. **Meningkatkan Keterbacaan dan Pengelolaan Kode**: Dengan DI, kita dapat memisahkan setiap komponen atau kelas sesuai dengan tanggung jawabnya masing-masing. Ini membuat kode lebih mudah dibaca dan di-maintain, karena setiap bagian kode fokus pada tugas spesifiknya.
2. **Mempermudah Pengujian**: DI memungkinkan pembuatan instance palsu (mock) dari dependensi untuk tujuan testing. Ini membuat pengujian lebih mudah dan cepat tanpa perlu mengubah banyak kode atau menginisialisasi seluruh aplikasi.
3. **Loose Coupling (Keterkaitan yang Rendah)**: Dengan DI, komponen tidak langsung tergantung pada implementasi kelas lain. Alih-alih, mereka hanya tergantung pada interface atau abstraksi. Hal ini membuat kode lebih fleksibel dan mudah dimodifikasi atau diubah tanpa memengaruhi bagian lain dari sistem.
4. **Pengelolaan Komponen Otomatis**: Dalam Spring Boot, dengan menggunakan anotasi seperti @Autowired, Spring secara otomatis mengelola penyuntikan dependensi sehingga developer tidak perlu membuat instance dari setiap komponen secara manual.

```java
import me.akai.controller.MahasiswaController;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
/**
 *
 * @author fadli
 */
@SpringBootApplication
public class Pertemuan5_50421097 implements CommandLineRunner {

    @Autowired
    private MahasiswaController mhsController;

    public static void main(String[] args) {
        SpringApplication.run(Pertemuan5_50421097.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        mhsController.tampilkanMenu();
    }
}
```

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Project/Maven2/JavaApp/src/main/java/${packagePath}/${mainClassName}.java to edit this template
 */

package me.akai;

import me.akai.controller.MahasiswaController;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
/**
 *
 * @author fadli
 */
@SpringBootApplication
public class Pertemuan5_50421437 implements CommandLineRunner {

    @Autowired
    private MahasiswaController mhsController;

    public static void main(String[] args) {
        SpringApplication.run(Pertemuan5_50421437.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        mhsController.tampilkanMenu();
    }
}
```

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Project/Maven2/JavaApp/src/main/java/${packagePath}/${mainClassName}.java to edit this template
 */

package me.akai;

import me.akai.controller.MahasiswaController;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
/**
 *
 * @author fadli
 */
@SpringBootApplication
public class Pertemuan5_50421437 implements CommandLineRunner {

    @Autowired
    private MahasiswaController mhsController;

    public static void main(String[] args) {
        SpringApplication.run(Pertemuan5_50421437.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        mhsController.tampilkanMenu();
    }
}
```

```
# Konfigurasi MySQL Hibernate
spring.datasource.url=jdbc:mysql://localhost:3306/spring_50421437?useSSL=false&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Hibernate settings
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package me.akai.repository;

import me.akai.model.ModelMahasiswa;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
/**
 *
 * @author Al Muzammil Imam
 */
@Repository
public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Long> {

}
```

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package me.akai.controller;

import me.akai.model.ModelMahasiswa;
import me.akai.repository.MahasiswaRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;

import java.util.List;
import java.util.Scanner;

@Controller
public class MahasiswaController {

    @Autowired
    private MahasiswaRepository mahasiswaRepository;

    public void tampilkanMenu() {
        Scanner scanner = new Scanner(System.in);
        int opsi;

        do {
            System.out.println("\nMenu: ");
            System.out.println("1. Tampilkan semua mahasiswa");
            System.out.println("2. Tambah mahasiswa baru");
            System.out.println("3. Cek koneksi database");
            System.out.println("4. Keluar");
            System.out.println("Pilih Opsi: ");
            opsi = scanner.nextInt();
            scanner.nextLine();

            switch (opsi) {
                case 1:
                    tampilkanSemuaMahasiswa();
                    break;
                case 2:
                    tambahMahasiswa(scanner);
                    break;
                case 3:
                    cekKoneksi();
                    break;
                case 4:
                    System.out.println("Keluar dari program");
                    break;
                default:
```