

# **LAPORAN PRAKTEK**

## **SPRINGBOOT SERVICE**

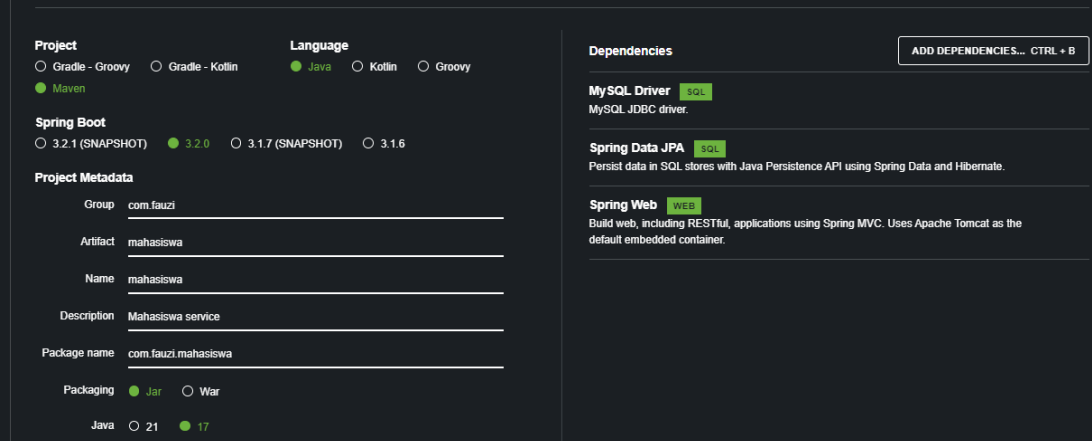


Nama : Fauzi Isyrin Apridal  
NIM : 2111083007  
Kelas : TRPL 3C  
Mata Kuliah : Sistem Terdistribusi

**TEKNOLOGI REKAYASA PERANGKAT LUNAK**  
**TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI PADANG**  
**TAHUN AKADEMIK 2023/2024**

Java Spring Boot adalah alat yang merupakan bagian dari Spring Framework, untuk membantu *java developer* membuat aplikasi segera berjalan dengan menyederhanakan proses pembuatan dan mengaktifkan konvensi atas konfigurasi, yang berarti IT Developer tidak perlu menghabiskan waktu untuk mengkonfigurasi setiap aspek aplikasi. Sejalan dengan ini, *java developer* dapat berfokus pada pengembangan fitur.

Dalam membuat service mahasiswa pertama-tama perlu juga untuk membuat project di start.spring.io menggunakan JDK 21 atau 17 serta spring boot 3.2.0, project yang digunakan adalah maven karena dibutuhkan beberapa dependensi berupa Driver JDBC untuk menghubungkan java dengan MySQL serta Spring JPA (java persistene API) untuk sejumlah fungsi untuk merepresentasikan operasi database seperti findById() dan lain sebagainya, terakhir dibutuhkan Spring Web untuk memetakan MVC (Model-View-Controller) dari aplikasi yang akan dibuat termasuk di dalamnya methode HTTP.



The screenshot displays the Spring Initializr web application interface. It is divided into two main sections: 'Project Metadata' on the left and 'Dependencies' on the right.

**Project Metadata:**

- Project:** ☐ Gradle - Groovy, ☐ Gradle - Kotlin, ☒ Maven
- Language:** ☒ Java, ☐ Kotlin, ☐ Groovy
- Spring Boot:** ☐ 3.2.1 (SNAPSHOT), ☒ 3.2.0, ☐ 3.1.7 (SNAPSHOT), ☐ 3.1.6
- Project Metadata:**
  - Group:
  - Artifact:
  - Name:
  - Description:
  - Package name:
  - Packaging: ☒ Jar, ☐ War
  - Java: ☐ 21, ☒ 17

**Dependencies:**

- MySQL Driver** (SQL): MySQL JDBC driver.
- Spring Data JPA** (SQL): Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
- Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

A button labeled 'ADD DEPENDENCIES... CTRL + B' is located at the top right of the Dependencies section.

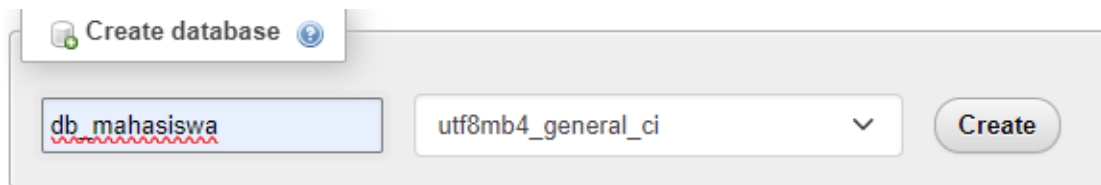
Selanjutnya spring akan menggenerate project yang kita butuhkan berupa rar lalu dapat dibuka di netbeans.

Setelah membuka project di netbeans buka application.properties di other sources default package sesuaikan dengan database yang akan dibuat serta port yang akan digunakan.

```
server.port=9001
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/db_mahasiswa
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.show-sql: true
```

Setelah itu klik kanan pada project lalu build with dependencies.

Gunakan web server xampp untuk membuat database sesuai dengan yang nama di application.properties



Selanjutnya buat package baru dengan nama mahasiswa.entity serta java class Mahasiswa.java untuk membuat table di dalam database dengan kolom id, nama, dan email serta constructor, setter dan getternya. Gunakan @Entity untuk menandakan bahwa java class ini adalah entity.

```

@Entity
@Table
public class Mahasiswa {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
    private String nama;
    private String email;

    public Mahasiswa() {
    }

    public Mahasiswa(long id, String nama, String email) {
        this.id = id;
        this.nama = nama;
        this.email = email;
    }
}

```

Selanjutnya buat package baru dengan nama mahasiswa.repository dengan java interface MahasiswaRepository yang mewarisi class JpaRepository berfungsi untuk CRUD database serta validasinya. Gunakan penanda @Repository.

```

import com.fauzi.mahasiswa.entity.Mahasiswa;
import java.util.Optional;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

/**
 *
 * @author Fauzi
 */
@Repository
public interface MahasiswaRepository extends JpaRepository<Mahasiswa, Long>{

    public Optional<Mahasiswa> findMahasiswaByEmail(String email);

}

```

Setelah itu buat package baru lagi dengan nama mahasiswa.service serta java class MahasiswaService.java untuk menerapkan logika dalam menangani dan memvalidasi data yang diinsert, didelete, diupdate, atau diread dari dan ke database menggunakan fungsi dari class repository yang telah dibuat sebelumnya. @Service menandakan bahwa class ini adalah service.

```
@Service
public class MahasiswaService {
    @Autowired
    private MahasiswaRepository mahasiswaRepository;

    public List<Mahasiswa> getAll() {
        return mahasiswaRepository.findAll();
    }

    public void insert(Mahasiswa mahasiswa) {
        Optional<Mahasiswa> mahasiswaOptional = mahasiswaOptional
```

Terakhir membuat package controller dan java class controller dengan penanda `@RestController` untuk memanggil fungsi CRUD di service sesuai dengan metode HTTP yang digunakan `@PostMapping`, `@PutMapping`, `@DeleteMapping` atau `@GetMapping` serta menggunakan `@RequestMapping` untuk menentukan url yang akan digunakan untuk memanggil seluruh fungsi tersebut.

```

@RestController
@RequestMapping("api/v1/mahasiswa")
public class MahasiswaController {

    @Autowired
    private MahasiswaService mahasiswaService;

    @GetMapping
    public List<Mahasiswa> getAll() {
        return mahasiswaService.getAll();
    }

    @GetMapping(path = "{id}")
    public Mahasiswa getMahasiswa(@PathVariable("id") Long id) {
        return mahasiswaService.getMahasiswa(id);
    }

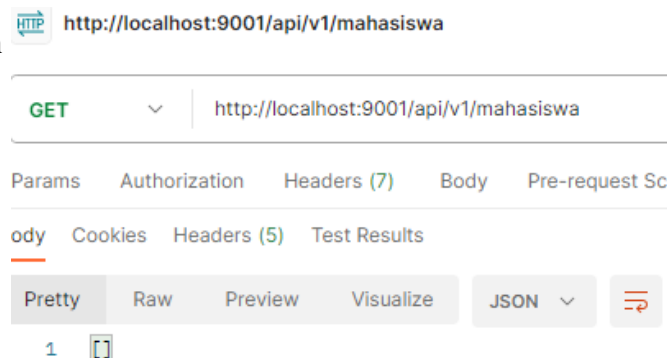
    @PostMapping
    public void insert(@RequestBody Mahasiswa mahasiswa) {
        mahasiswaService.insert(mahasiswa);
    }

    @DeleteMapping(path = "{id}")
    public void delete(@PathVariable("id") Long id) {
        mahasiswaService.delete(id);
    }

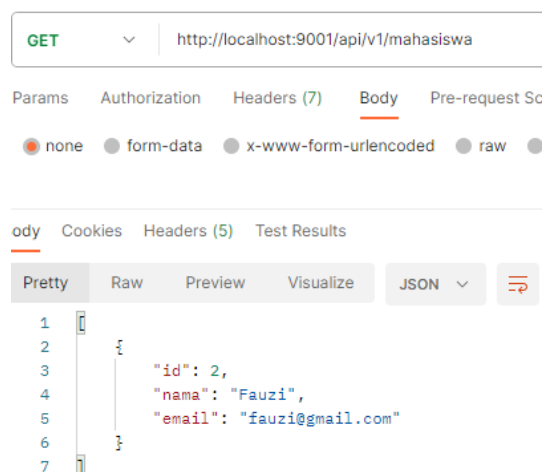
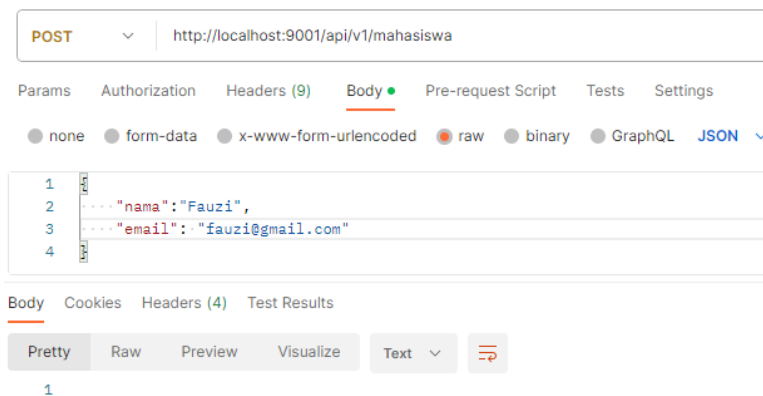
    @PutMapping(path = "{id}")
    public void update(@PathVariable("id") Long id,

```

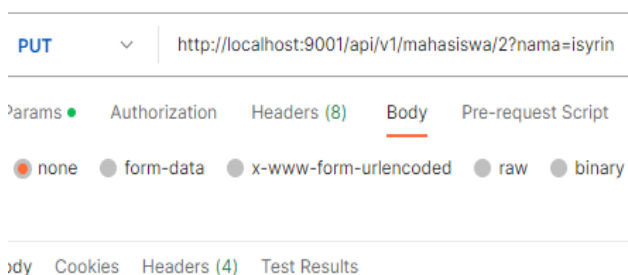
Setelah semuanya selesai jalankan MahasiswaApplication di package utama yang akan menggenerate table mahasiswa di database db\_mahasiswa dan jalankan juga postman untuk testing API yang telah dibuat dengan url <http://localhost:9001/api/v1/mahasiswa> dengan metode GET aplikasi akan mengambil data yang ada di table mahasiswa dikarenakan masih belum ada data akan dikembalikan



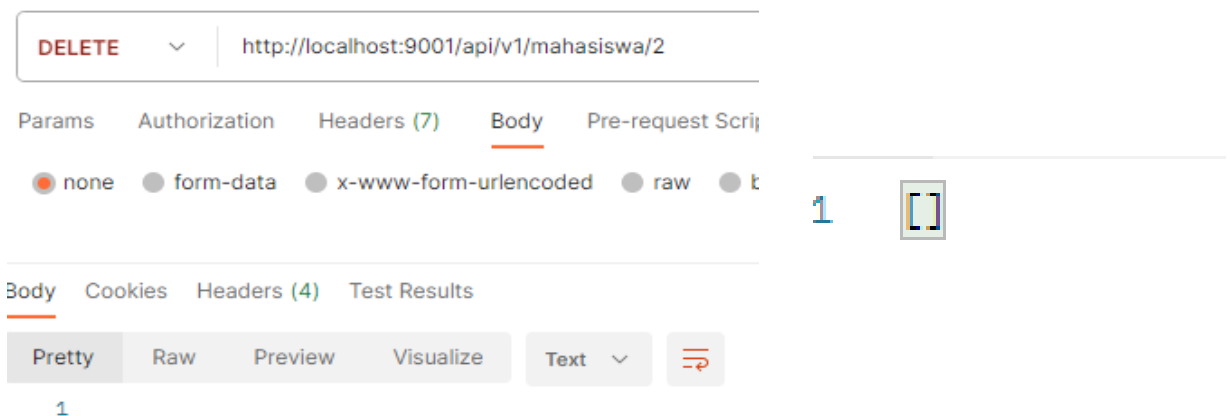
Menggunakan metode POST untuk melakukan insert data di postman dengan url yang sama masuk ke tab body dan radio raw masukkan data dalam bentuk JSON untuk nama dan email. Ketika dikirim akan dikembalikan kosong tandanya insert berhasil Kembali ke metode GET untuk mengambil data.



Untuk mengupdate data gunakan metode PUT dan url yang sama ditambahkan dengan `/id?nama kolom=nilaibaru` contohnya <http://localhost:9001/api/v1/mahasiswa/2?nama=isyryn> akan dikembalikan nilai kosong dan jika Kembali ke GET data akan berubah



Untuk menghapus data gunakan url yang sama ditambah id yang akan dihapus contohnya <http://localhost:9001/api/v1/mahasiswa/2> setelah dikirim akan muncul nilai kosong dan jika Kembali ke GET akan mengembalikan array kosong.



Selanjutnya dapat ditambahkan untuk Matakuliah dan Nilai dengan cara yang sama.