

LANGKAH-LANGKAH MEMBUAT SPRING BOOT + MYSQL BACKEND MENGGUNAKAN VS CODE

1. Install setiap tools yang dibutuhkan:

Pastikan tools di bawah ini sudah terinstall:

a. Java Development Kit (JDK)

- Install **JDK 17 or 21**
- Untuk verifikasi bisa melakukan perintah ini dalam cmd/terminal: `java -version`

b. Visual Studio Code

- Install ekstensi VS Code:
 - **Extension Pack for Java**
 - **Spring Boot Dashboard**
 - **Spring Initializr**
 - **Lombok Annotations Support** (opsional tapi direkomendasikan)

c. MySQL Server + MySQL Workbench

- Buat database baru: `CREATE DATABASE dbmarket;`

2. Membuat Spring Boot Project (Menggunakan Spring Initializr in VS Code)

1. Buka VS Code
2. Tekan **Ctrl + Shift + P**
3. Pilih **"Spring Initializr: Create a Maven Project"**
4. Pilih:
 - **Java 17** atau **21**
 - **Spring Boot version** (3.x recommended)
5. Isi:
 - Group: `com.rmn`
 - Artifact: `mybackend`
6. Add dependencies:
 - **Spring Web**
 - **Spring Data JPA**
 - **MySQL Driver**
 - (Optional) **Lombok**
7. Generate the project → choose a folder → Finish.

3. Konfigurasi MySQL Connection dalam application.properties

Dalam `src/main/resources/application.properties`:

```
spring.application.name=mybackend
# --- Konfigurasi Server ---
server.port=8080

# --- Konfigurasi MySQL ---
spring.datasource.url=jdbc:mysql://localhost:3306/dbmarket?useSSL=false&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=
```

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

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

4. Membuat Model (Entity)

Model (Entity): Digunakan untuk merepresentasikan data.

Contoh: HaloResponse.java

```
package com.rmn.model;

public class HaloResponse {
    private String pesan;
    private String status;

    public HaloResponse(String pesan, String status) {
        this.pesan = pesan;
        this.status = status;
    }

    public String getPesan() {
        return pesan;
    }

    public String getStatus() {
        return status;
    }

    public void setPesan(String pesan) {
        this.pesan = pesan;
    }

    public void setStatus(String status) {
        this.status = status;
    }
}
```

5. Membuat Controller REST API

Ini adalah komponen yang bertugas menerima permintaan HTTP dari klien dan mengirimkan respons kembali. *Controller* digunakan untuk mengekspos *endpoint* REST.

Buat *package* baru (misalnya com.rmn.controller) dan buat kelas berikut:

Contoh: HaloController.java

```
package com.rmn.controller;

import com.rmn.model.HaloResponse;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("/api/v1")
public class HaloController {
```

```

    @GetMapping("/halo")
    public HaloResponse sapaDunia() {
        HaloResponse response = new HaloResponse("Halo dari Microservice
Spring Boot!", "OK");
        return response;
    }

    @GetMapping("/test")
    public String testKoneksi() {
        return "Koneksi berhasil!";
    }
}

```

6. Membuat SecurityConfig

Ini adalah konfigurasi yang kita buat untuk menonaktifkan form *login* Spring Security, memungkinkan akses ke API Anda. Buat *package* baru (misalnya `com.rmn.config`) dan buat kelas berikut:

```

package com.rmn.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.web.SecurityFilterChain;

@Configuration
@EnableWebSecurity
public class SecurityConfig {

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws
Exception {
        http
            .csrf(csrf -> csrf.disable())

            .authorizeHttpRequests(authorize -> authorize
                .anyRequest().permitAll()
            );

        return http.build();
    }
}

```

7. Memindahkan Main Class

Pindahkan main class (Misal: `MybackendApplication`) ke dalam package `com.rmn` sehingga kodenya seperti berikut:

```

package com.rmn;

```

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class MybackendApplication {

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

}
```

7. Menjalankan Aplikasi

Pada VS Code:

- Gunakan terminal:
./mvnw spring-boot:run

Check API:

- GET: <http://localhost:8080/api/v1/halo>
- GET: <http://localhost:8080/api/v1/test>

8. Uji Menggunakan Postman

Install **Postman**