**LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : RPL 2

Kelas : 4IA06

Praktikum ke- : 5

Tanggal : 12 November 2024

Materi : Framework Spring,Pembuatan Project Spring, dan Hibernate

NPM : 50421535

Nama : Fizri Rosdiansyah

Ketua Asisten :

Paraf Asisten :

Nama Asisten : Suryo Aji Widago

Jumlah Lembar :12 Lembar

LABORATORIUM TEKNIK INFORMATIKA UNIVERSITAS GUNADARMA

2024

Rekayasa Perangkat Lunak 2

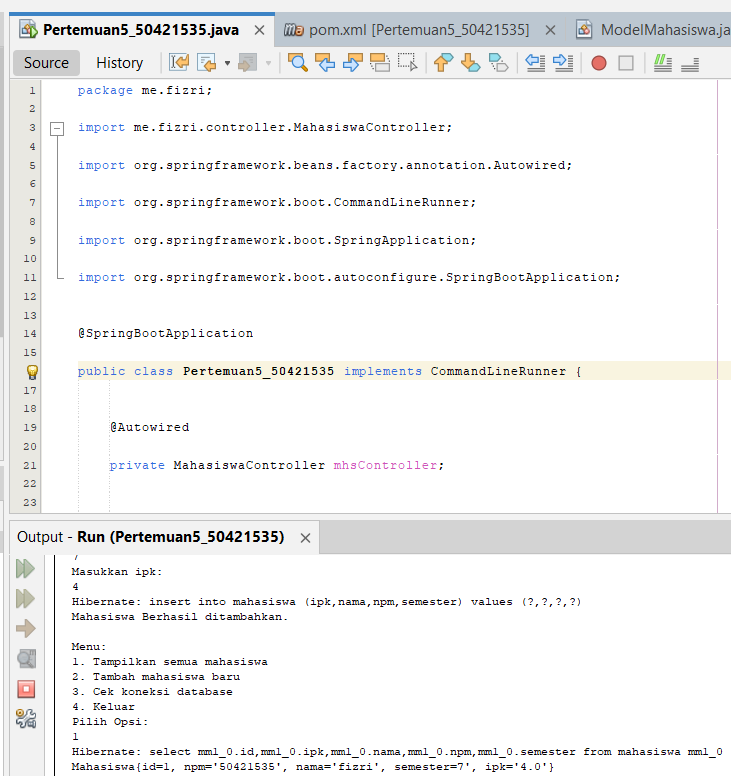
Soal

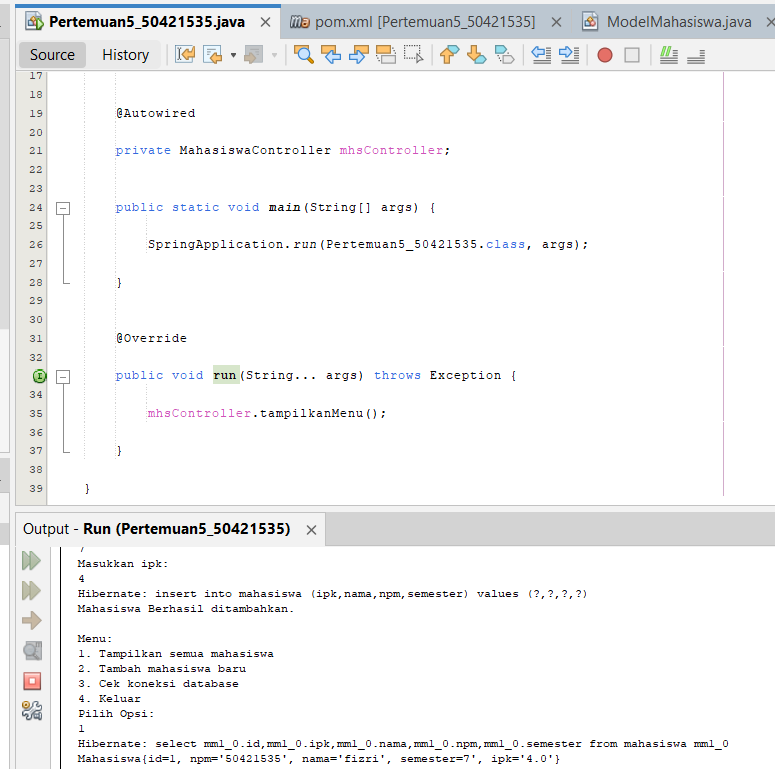
1. Jelaskan apa itu Spring Boot dan bagaimana perbedaanya dengan kode pada pertemuan sebelumnya. Apa keuntungan utama yang ditawarkan Spring Boot bagi pengembang aplikasi?  
2. Jelaskan kode dan langkah-langkah program yang telah dibuat!

Jawab

1.

* **Spring Boot** adalah framework berbasis Java yang dirancang untuk mempermudah pengembangan aplikasi dengan Spring.
* Perbedaan codingannya:  
  dimana Codingan Sebelumnya Harus  
  **Mengatur file konfigurasi XML** atau kode konfigurasi Java untuk mengelola dependensi, bean, dan pengaturan database.
* **Mengonfigurasi ORM dan Hibernate secara manual** dengan menyetel koneksi database, entitas, dan session factory secara eksplisit.
* **Mengelola server aplikasi secara eksternal**, di mana Anda harus menyebarkan aplikasi ke dalam server aplikasi (misalnya Tomcat) untuk menjalankannya.
* **Keuntungan Utama Spring Boot bagi Pengembang Aplikasi**
* **Waktu Pengembangan yang Lebih Cepat**: Dengan konfigurasi otomatis dan starter dependencies, Spring Boot sangat mengurangi waktu dan upaya konfigurasi.
* **Aplikasi Standalone**: Dengan server bawaan, aplikasi bisa dijalankan langsung tanpa server aplikasi eksternal, yang memudahkan pengujian dan penyebaran.
* **Production-ready Features**: Spring Boot menyediakan alat pemantauan dan manajemen untuk aplikasi produksi, seperti Spring Boot Actuator, yang memberi wawasan mendalam tentang status aplikasi.
* **Dukungan Cloud dan Microservices**: Spring Boot memiliki integrasi yang baik untuk pengembangan aplikasi berbasis microservices dan penggunaan di lingkungan cloud.

2.A.Pertemuan5\_50421535  


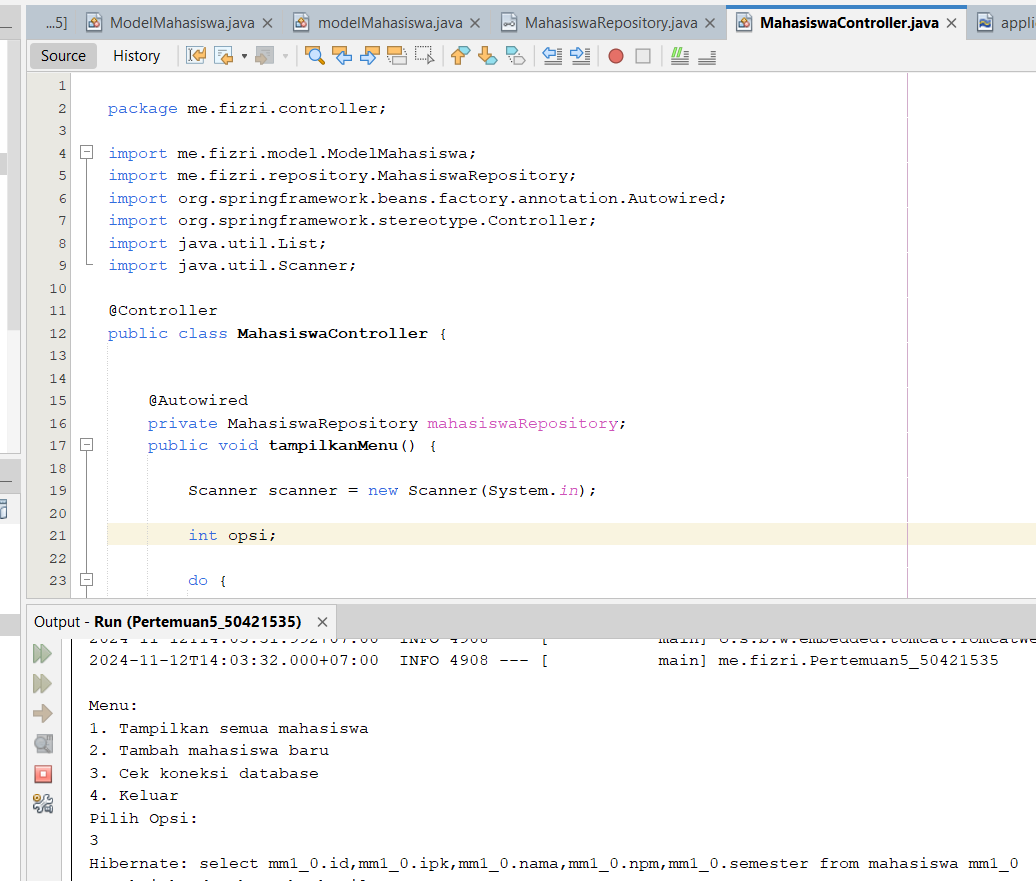


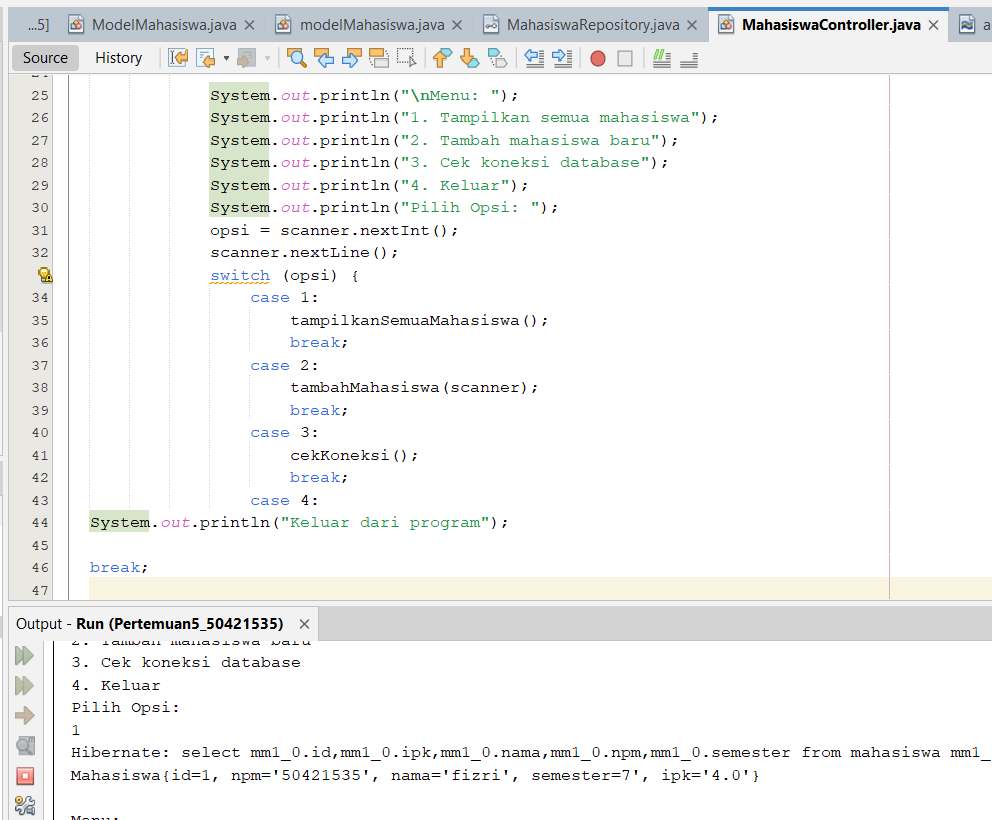
1. **Inisialisasi Aplikasi Spring Boot**: Kelas Pertemuan5\_50421535 adalah titik masuk aplikasi Spring Boot, dan menjalankan aplikasi dengan konfigurasi otomatis.
2. **Dependency Injection**: Menggunakan anotasi @Autowired, Spring secara otomatis menyediakan (meng-*inject*) objek MahasiswaController ke dalam kelas ini, sehingga dapat digunakan tanpa perlu membuatnya secara manual.
3. **Eksekusi Logika Saat Startup**: Kelas ini mengimplementasikan CommandLineRunner, yang memungkinkan metode run() dijalankan segera setelah aplikasi dimulai. Di dalam metode ini, mhsController.tampilkanMenu() dipanggil, yang kemungkinan besar memicu tampilan menu untuk pengguna.

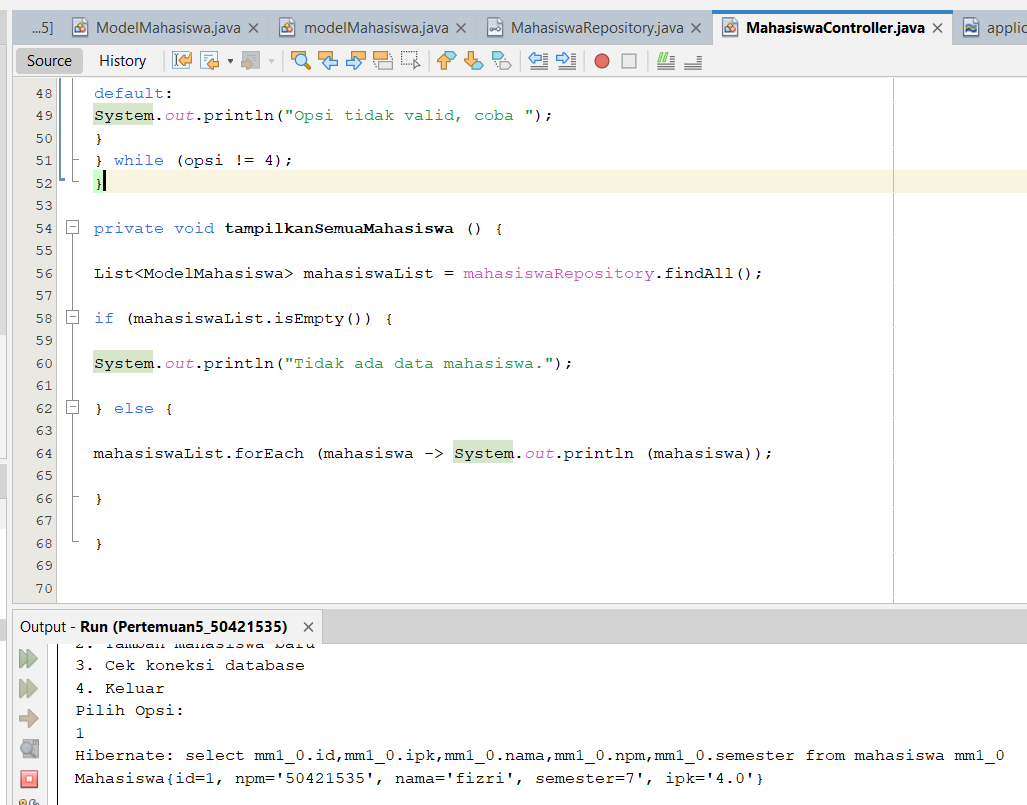
**Intinya:**

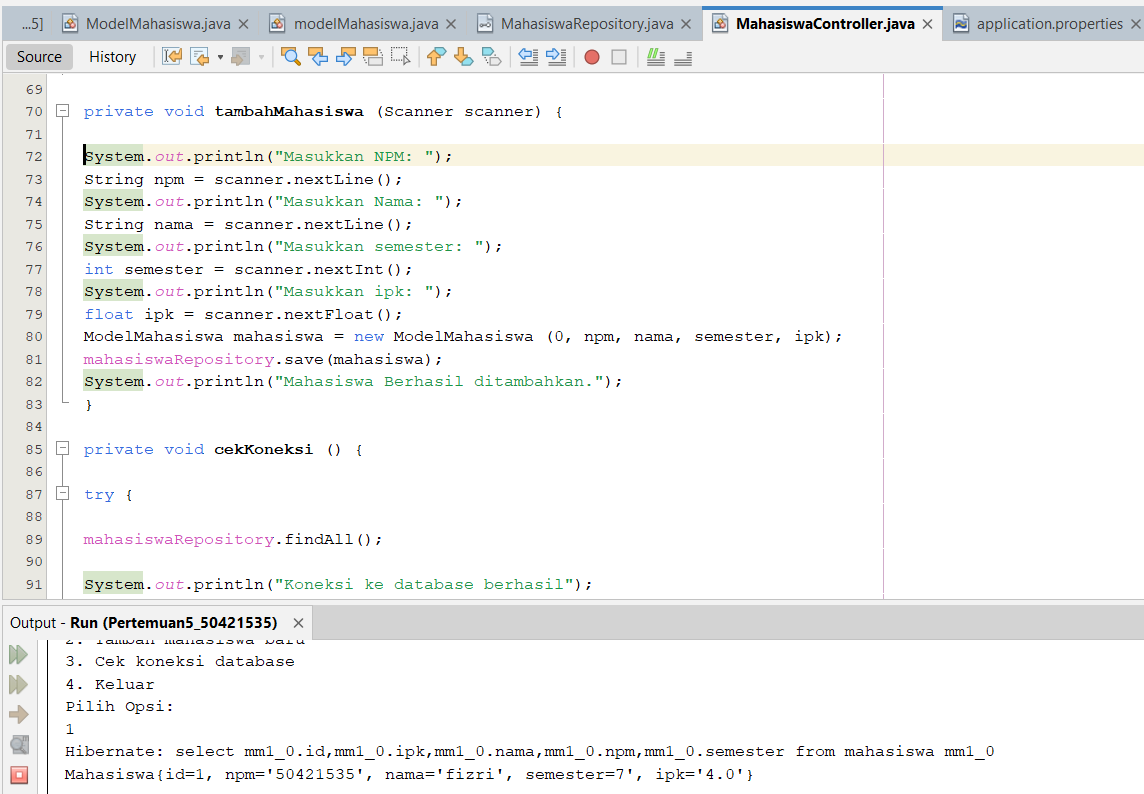
Kode ini adalah aplikasi Spring Boot sederhana yang, pada saat dijalankan, memulai aplikasi, menginisialisasi MahasiswaController, dan menampilkan menu dengan memanggil tampilkanMenu().

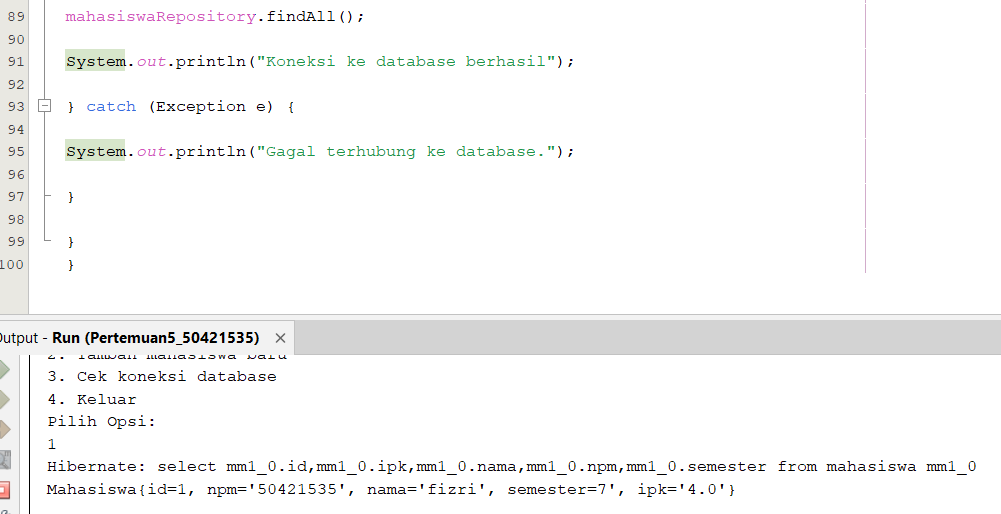
B.MahasiswaController



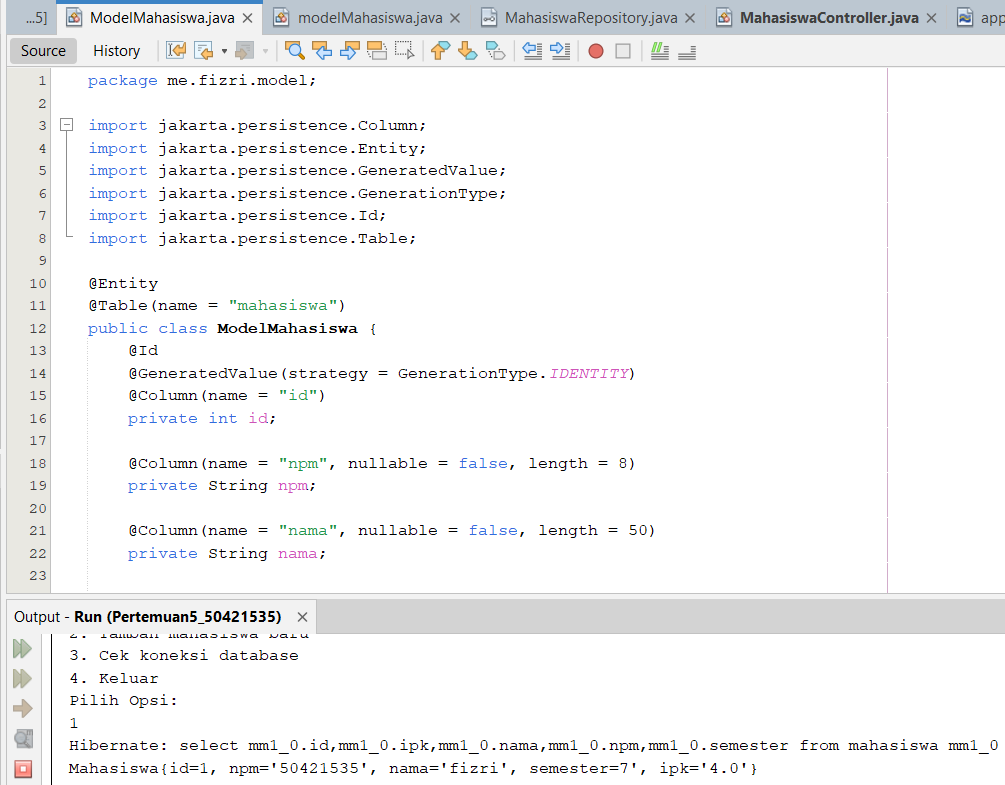


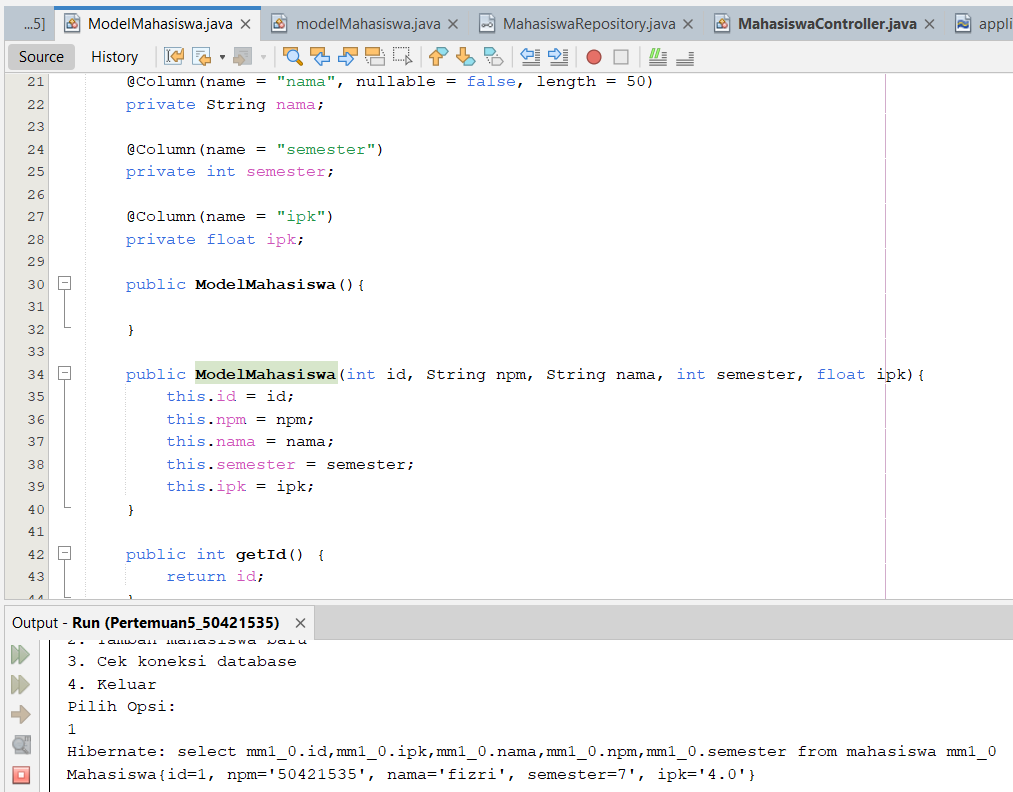


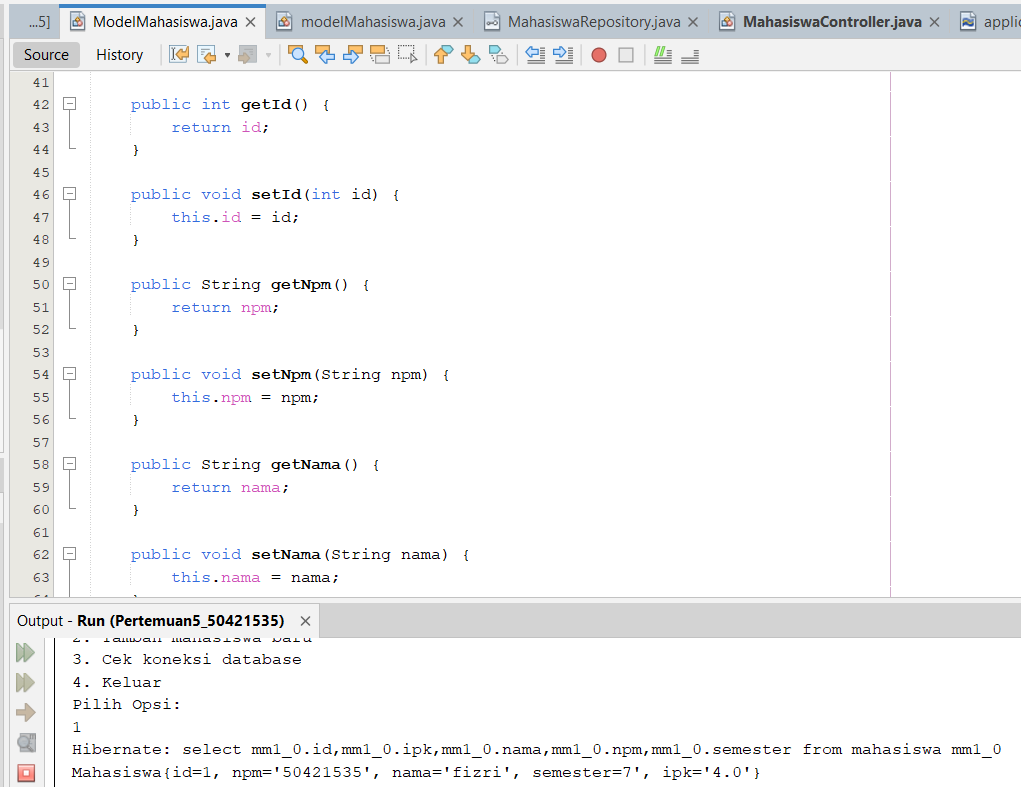


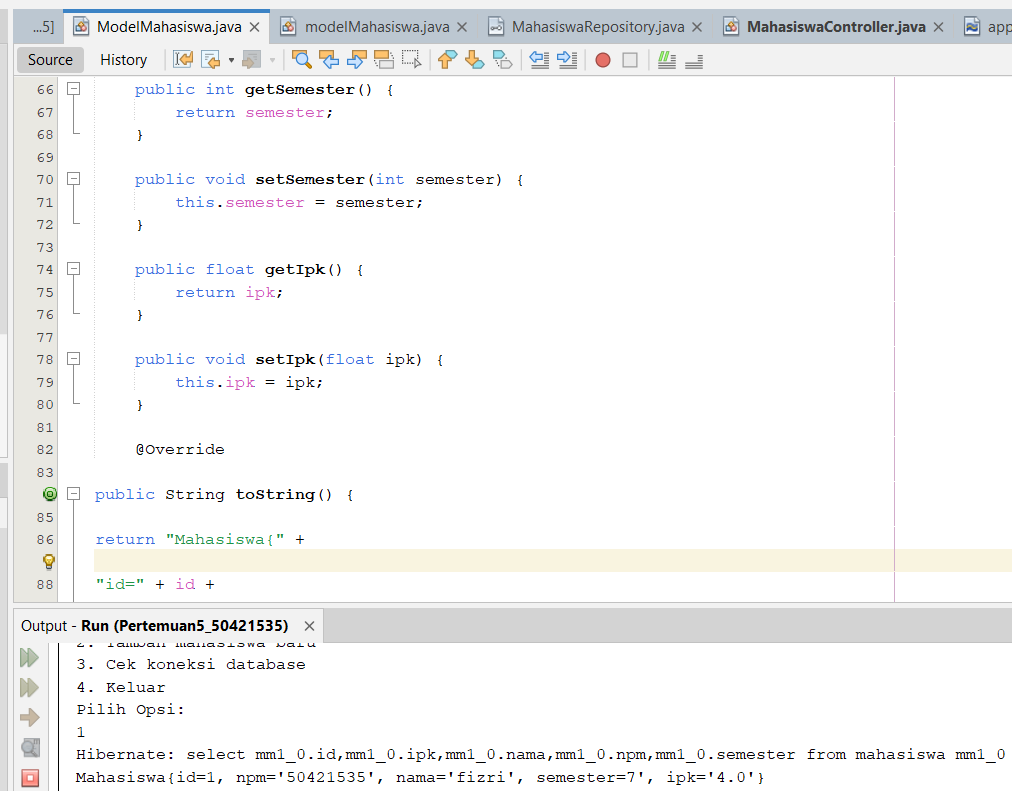


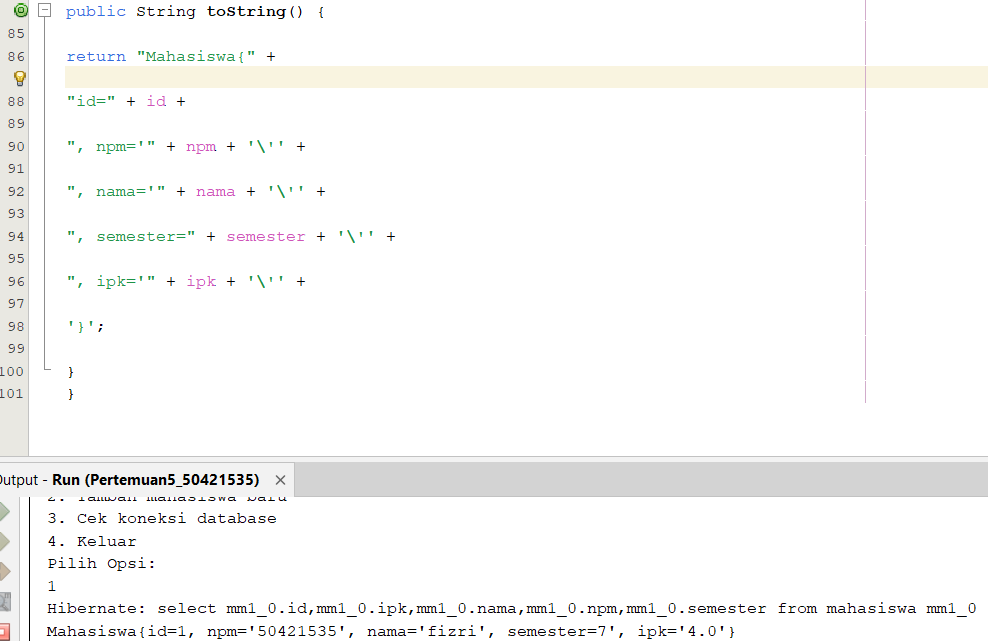
* **Anotasi @Controller**:
* Kelas ini adalah bagian dari lapisan *controller* dalam Spring dan diberi anotasi @Controller, sehingga Spring mengenalinya sebagai komponen *controller*.
* **Dependency Injection dengan @Autowired**:
* MahasiswaRepository di-*inject* ke dalam kelas ini menggunakan anotasi @Autowired, memungkinkan kelas ini untuk berinteraksi dengan database melalui MahasiswaRepository tanpa perlu membuat objeknya secara manual.
* **Metode tampilkanMenu**:
* Metode ini menampilkan menu utama bagi pengguna untuk melakukan berbagai tindakan terkait data mahasiswa. Menggunakan *loop* do-while agar menu tetap ditampilkan sampai pengguna memilih opsi "Keluar".
* **Pilihan dalam Menu**:
* **Opsi 1 - tampilkanSemuaMahasiswa**: Menampilkan semua data mahasiswa yang ada dalam database. Jika data kosong, akan mencetak pesan "Tidak ada data mahasiswa."
* **Opsi 2 - tambahMahasiswa**: Menambahkan data mahasiswa baru ke database. Mengambil input dari pengguna untuk detail mahasiswa seperti NPM, nama, semester, dan IPK, lalu menyimpannya melalui mahasiswaRepository.
* **Opsi 3 - cekKoneksi**: Memeriksa koneksi ke database dengan mencoba mengakses data. Jika koneksi berhasil, mencetak pesan keberhasilan; jika gagal, mencetak pesan kegagalan.
* **Opsi 4 - Keluar**: Menghentikan program.
* **Metode tampilkanSemuaMahasiswa**:
* Mengambil data semua mahasiswa dari database menggunakan mahasiswaRepository.findAll(), lalu menampilkannya satu per satu.
* **Metode tambahMahasiswa**:
* Mengambil input dari pengguna untuk membuat objek ModelMahasiswa baru, lalu menyimpan objek tersebut ke database.
* **Metode cekKoneksi**:
* Memeriksa koneksi ke database dengan mencoba menjalankan findAll() dari mahasiswaRepository. Menampilkan pesan berhasil atau gagal sesuai dengan hasil koneksi.

C.ModelMahasiswa  








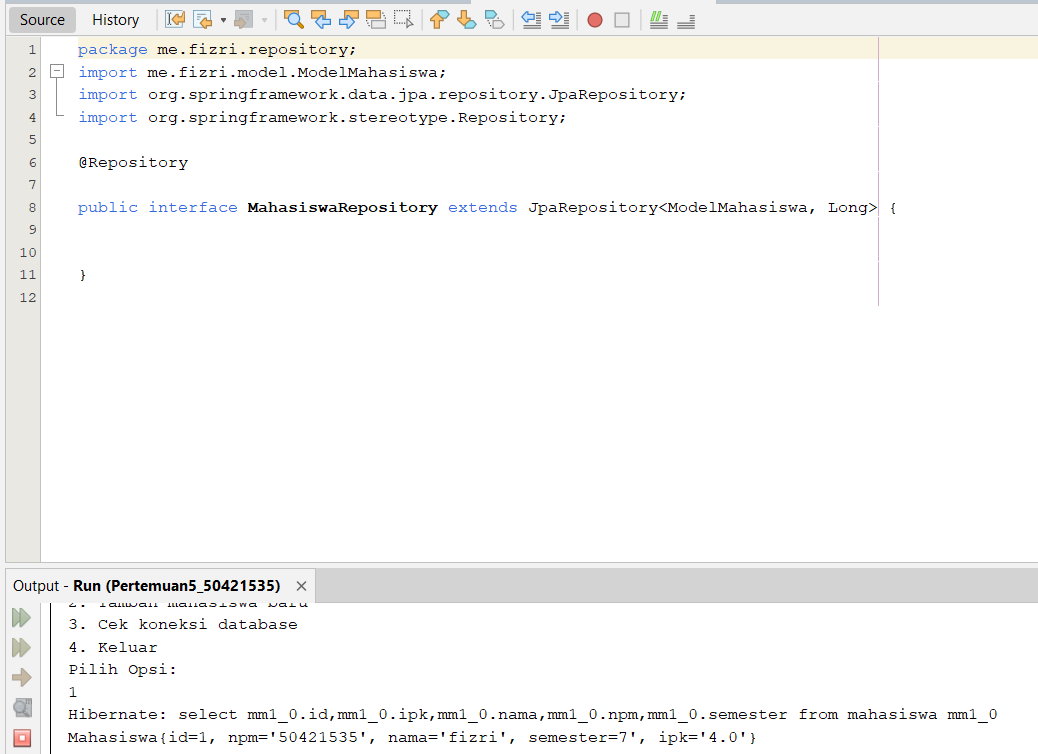


1. **Anotasi @Entity dan @Table**:
   * Anotasi @Entity menunjukkan bahwa kelas ini adalah entitas yang akan dipetakan ke dalam tabel database.
   * @Table(name = "mahasiswa") menunjukkan bahwa entitas ini dipetakan ke tabel bernama mahasiswa di database.
2. **Atribut (Fields) dan Anotasi Kolom**:
   * Kelas ini memiliki beberapa atribut yang mewakili kolom dalam tabel mahasiswa:
     + **id**: Kolom ID sebagai *primary key*, dengan anotasi @Id dan *auto increment* (@GeneratedValue(strategy = GenerationType.IDENTITY)).
     + **npm**: Kolom NPM, sebuah String yang dibatasi hingga 8 karakter dan tidak boleh bernilai null.
     + **nama**: Kolom Nama, sebuah String yang dibatasi hingga 50 karakter dan tidak boleh bernilai null.
     + **semester**: Kolom Semester, disimpan sebagai tipe int.
     + **ipk**: Kolom IPK, disimpan sebagai tipe float.
   * Setiap kolom diberi anotasi @Column untuk mengatur nama kolom dan beberapa atribut lainnya, seperti nullable dan length.
3. **Constructor**:
   * **Constructor tanpa parameter**: Diperlukan oleh Hibernate untuk membuat instance entitas secara otomatis.
   * **Constructor dengan parameter**: Mempermudah pembuatan objek ModelMahasiswa dengan semua atribut yang diperlukan.
4. **Getter dan Setter**:
   * Setiap atribut memiliki metode get dan set untuk memungkinkan akses dan modifikasi nilai dari luar kelas.
5. **Metode toString**:
   * Metode toString memberikan representasi String dari objek ModelMahasiswa, mencantumkan semua atribut. Ini berguna untuk debugging atau ketika ingin menampilkan data mahasiswa dengan format yang mudah dibaca.

**Intinya:**

Kelas ModelMahasiswa adalah model entitas yang memetakan data mahasiswa ke tabel mahasiswa dalam database. Kelas ini memiliki atribut yang sesuai dengan kolom tabel, serta metode untuk mengakses dan memodifikasi data tersebut.

D.MahasiswaRepository.java

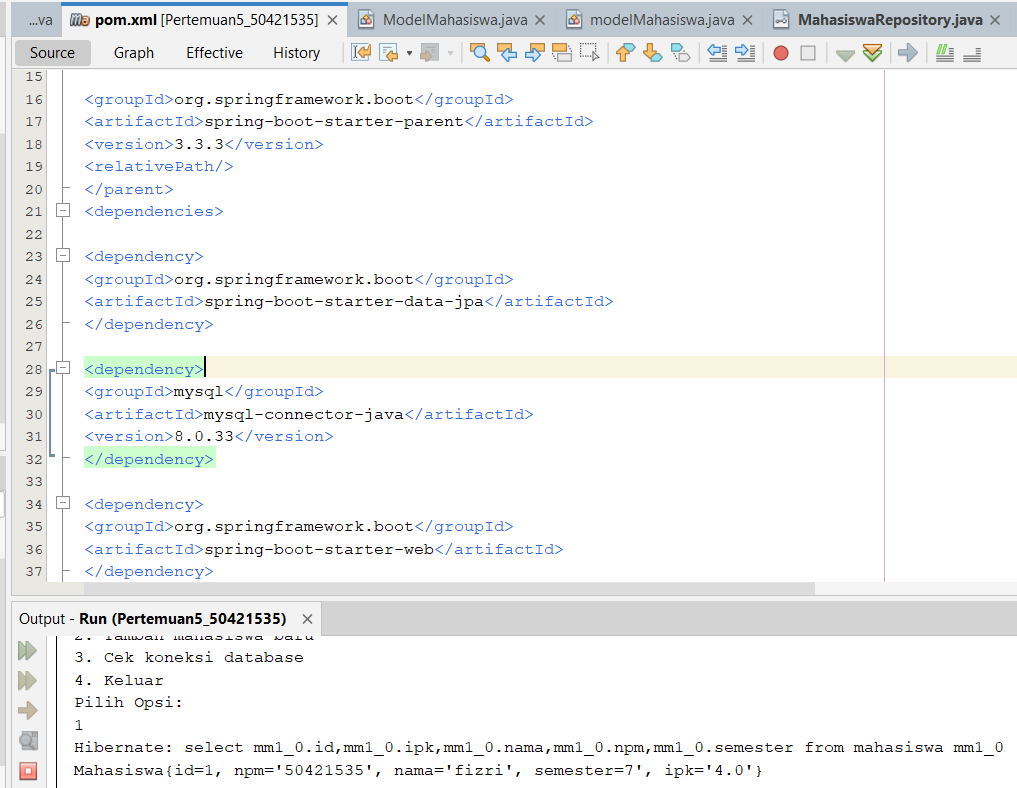


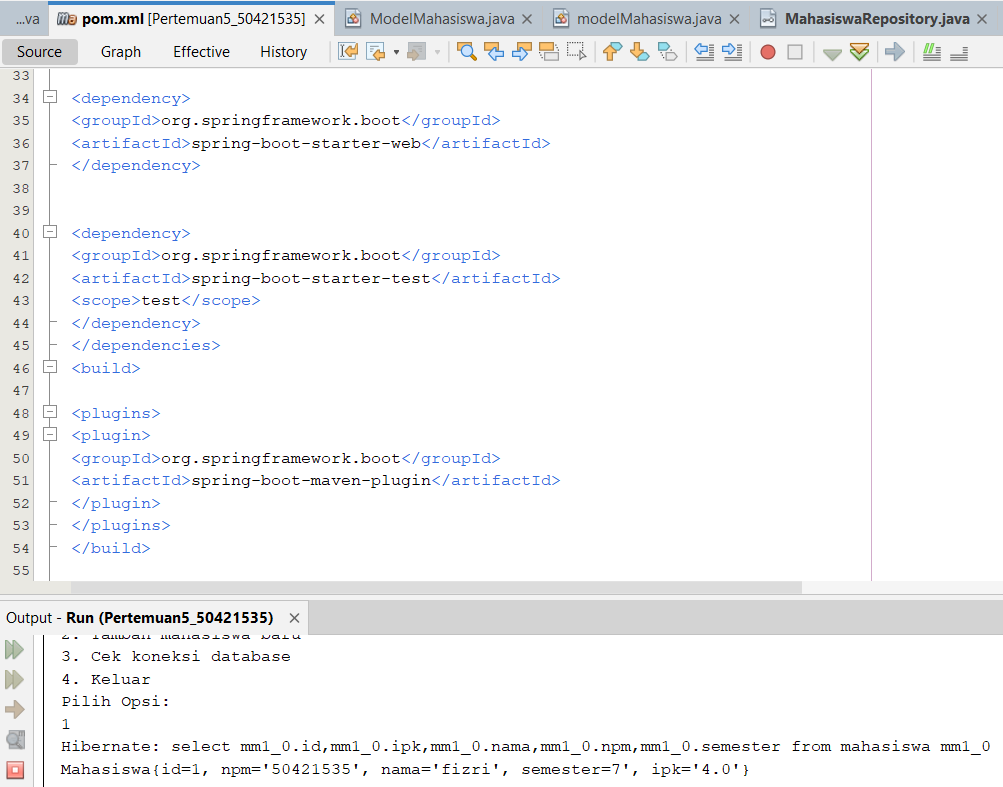
1. **Anotasi @Repository**:
   * Menandai interface ini sebagai komponen Spring untuk interaksi dengan database. Dengan anotasi ini, Spring akan mengenali MahasiswaRepository sebagai komponen Repository yang menangani operasi CRUD pada entitas ModelMahasiswa.
2. **Pewarisan dari JpaRepository**:
   * Interface MahasiswaRepository mewarisi dari JpaRepository, yang merupakan interface bawaan Spring Data JPA.
   * JpaRepository<ModelMahasiswa, Long> berarti MahasiswaRepository dikhususkan untuk menangani entitas ModelMahasiswa dengan *primary key* bertipe Long.
3. **Operasi CRUD Otomatis**:
   * Dengan mewarisi JpaRepository, interface ini secara otomatis mendapatkan berbagai metode standar untuk operasi basis data, seperti:
     + findAll() - Mengambil semua data entitas.
     + save() - Menyimpan atau memperbarui entitas ke database.
     + findById() - Mengambil entitas berdasarkan ID.
     + deleteById() - Menghapus entitas berdasarkan ID.
   * Hal ini berarti Anda tidak perlu menulis kode tambahan untuk operasi CRUD dasar.

**Intinya:**

MahasiswaRepository adalah interface yang memfasilitasi interaksi dengan database untuk entitas ModelMahasiswa. Dengan menggunakan JpaRepository, Spring secara otomatis menyediakan berbagai metode CRUD standar, sehingga menghemat waktu dalam pembuatan query manual.

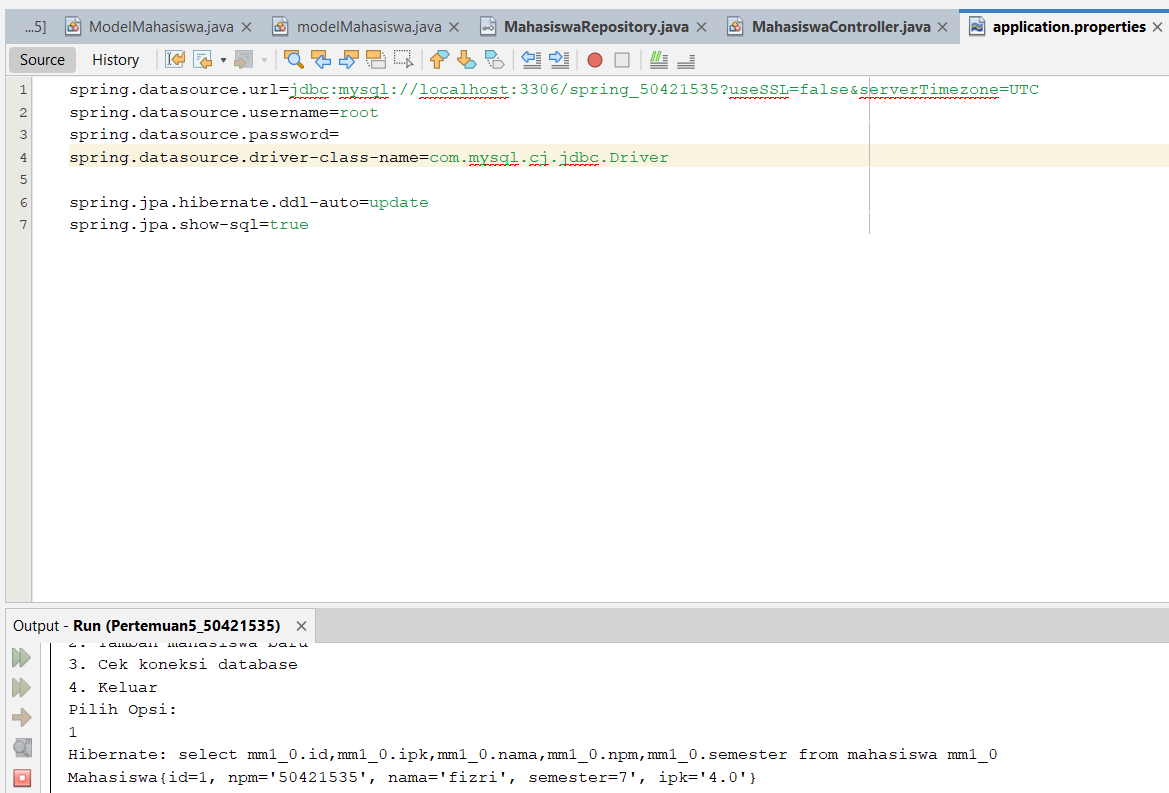
E.pom.xml





File pom.xml ini mengonfigurasi proyek Maven untuk mengimpor dependensi yang diperlukan untuk mengembangkan aplikasi Spring Boot dengan dukungan JPA, konektor MySQL, dan pengembangan aplikasi web. Selain itu, juga menyiapkan plugin Maven yang memungkinkan aplikasi untuk dijalankan dengan mudah dan diuji.

F.application.properties



Konfigurasi ini mengatur aplikasi Spring Boot untuk terhubung ke database MySQL dengan nama spring\_50421535 di localhost, menggunakan pengguna root (dengan atau tanpa kata sandi), dan menggunakan Hibernate untuk mengelola skema database. Query SQL yang dieksekusi oleh Hibernate akan ditampilkan di log aplikas