Nama : Muhammad Ridho

Kelas : 4IA06

NPM : 51421055

Materi : Konsep Model – View – Controller (MVC)

Mata Pratikum : Rekayasa Perangkat Lunak 2

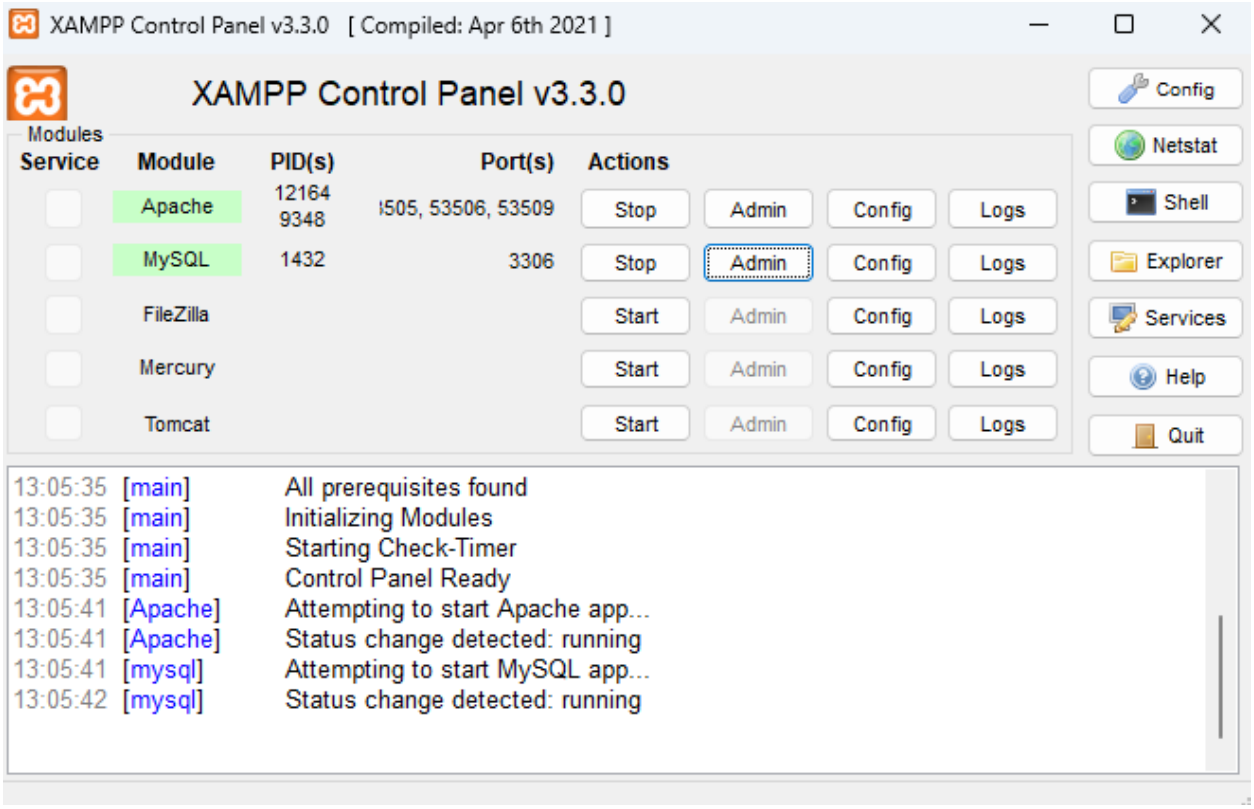
ACTIVITY 3

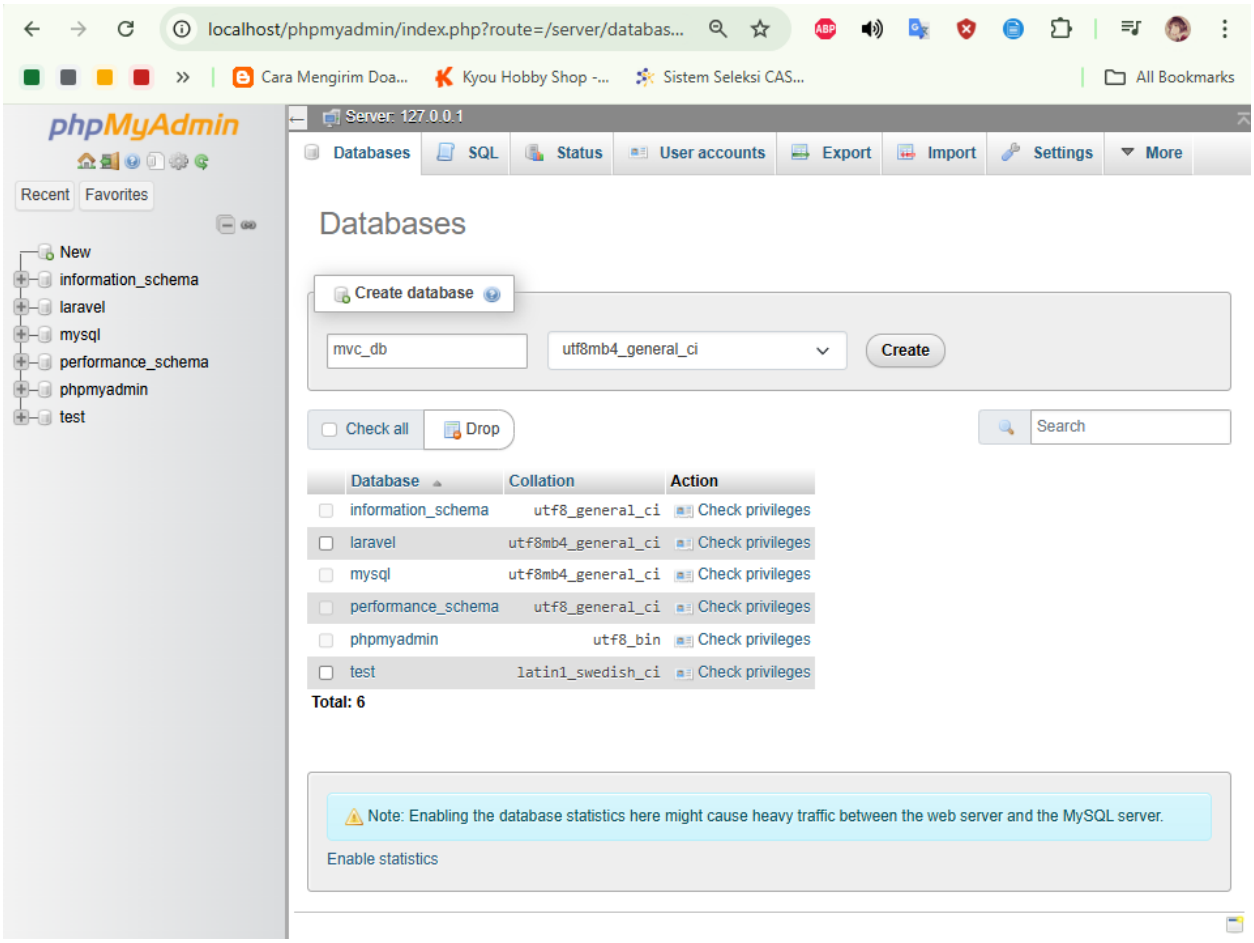
SOAL

1. Jelaskan bagaimana cara membuat database “mvc_db” menggunakan XAMPP?
2. Screenshot hasil codingan kalian lengkap dengan window Projects, Navigator, dan Outputnya.

JAWABAN

1. Pertama, buka XAMPP lalu nyalakan Apache dan MySQL.
   Kedua, masuk ke admin MySQL.
   Ketiga, buat database mvc_db.
   Keempat, run query untuk membuat table mahasiswa, lalu tekan go.
2.




localhost/phpmyadmin/index.php?route=/server/databas...
Cara Mengirim Doa...
Kyou Hobby Shop -...
Sistem Seleksi CAS...
All Bookmarks
phpMyAdmin
Recent
Favorites
New
information_schema
laravel
mysql
performance_schema
phpmyadmin
test
Server: 127.0.0.1
Databases
SQL
Status
User accounts
Export
Import
Settings
More
Databases
Create database
mvc_db
utf8mb4_general_ci
Create
Check all
Drop
Search
Database
Collation
Action
information_schema utf8_general_ci Check privileges
laravel utf8mb4_general_ci Check privileges
mysql utf8mb4_general_ci Check privileges
performance_schema utf8_general_ci Check privileges
phpmyadmin utf8_bin Check privileges
test latin1_swedish_ci Check privileges
Total: 6
Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.
Enable statistics

Run SQL query/queries on database mvc_db:

```sql
CREATE TABLE `mahasiswa`(
    `id` int(11) NOT NULL,
    `ipk` float NOT NULL,
    `nama` varchar(55) DEFAULT NULL,
    `npm` varchar(10) DEFAULT NULL,
    `semester` int(3) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

ALTER TABLE `mahasiswa`
    ADD PRIMARY KEY(`id`);

ALTER TABLE `mahasiswa`
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
COMMIT;
```

Clear | Format | Get auto-saved query

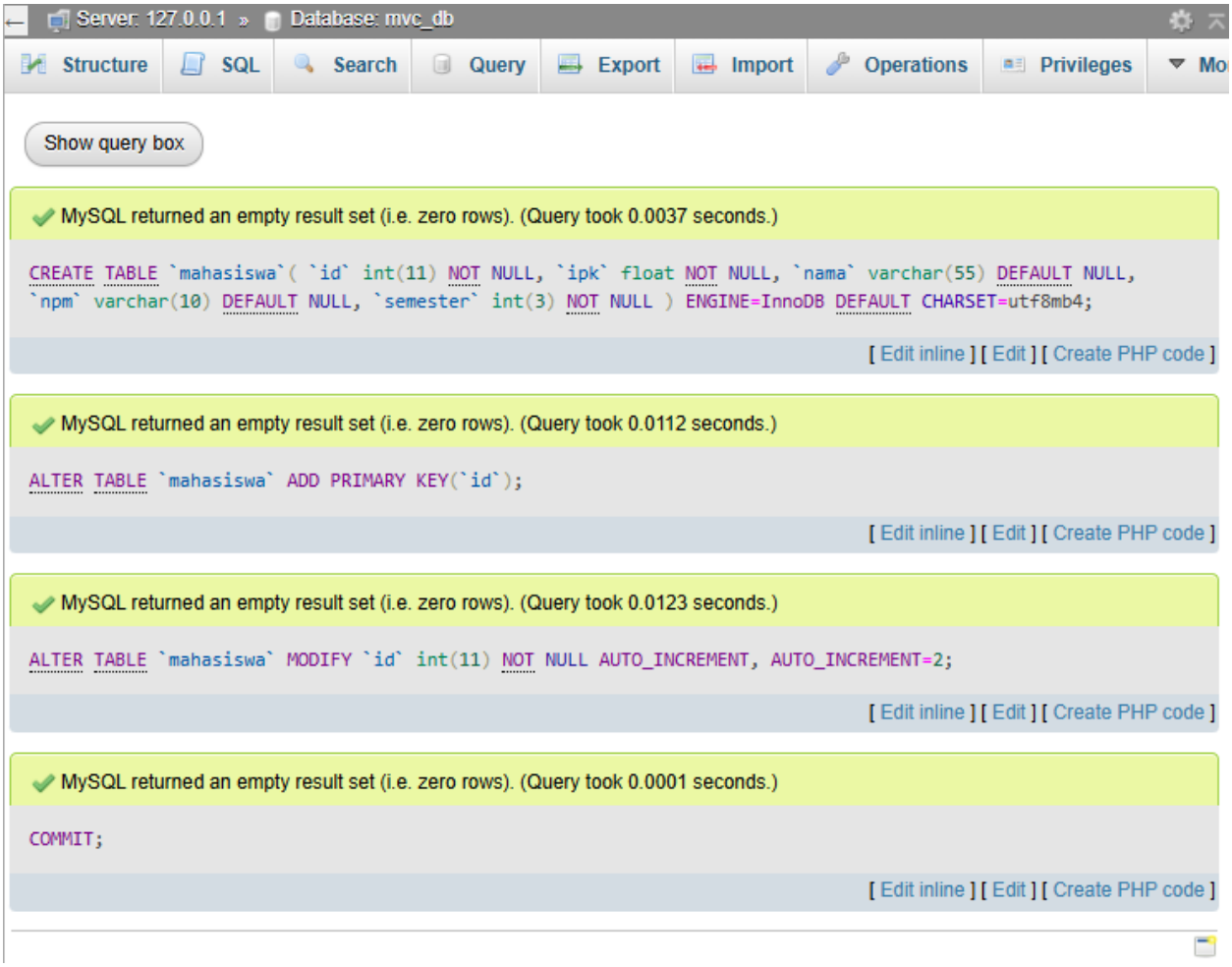
Bind parameters

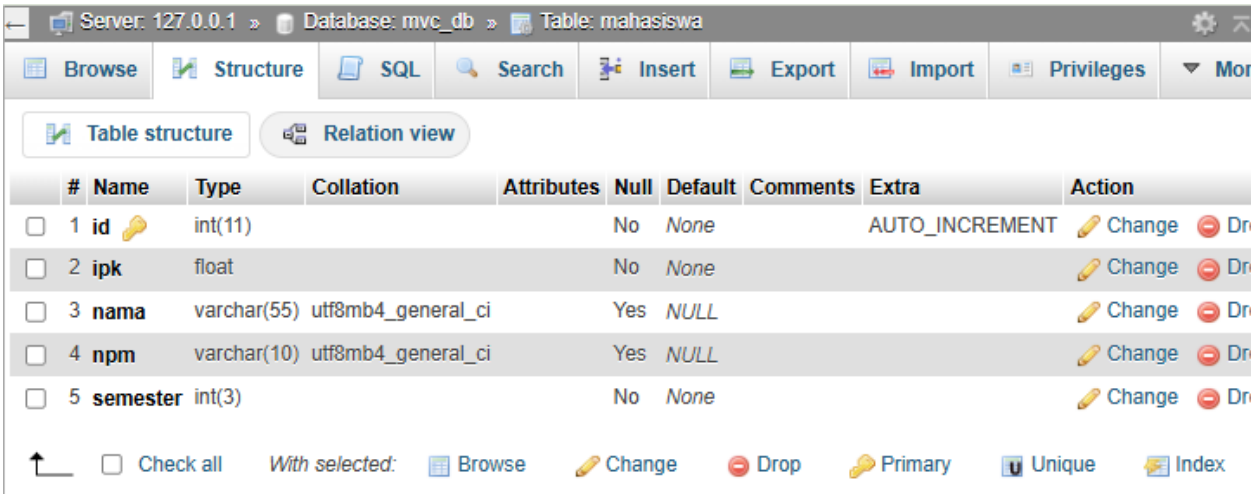
Delimiter ;

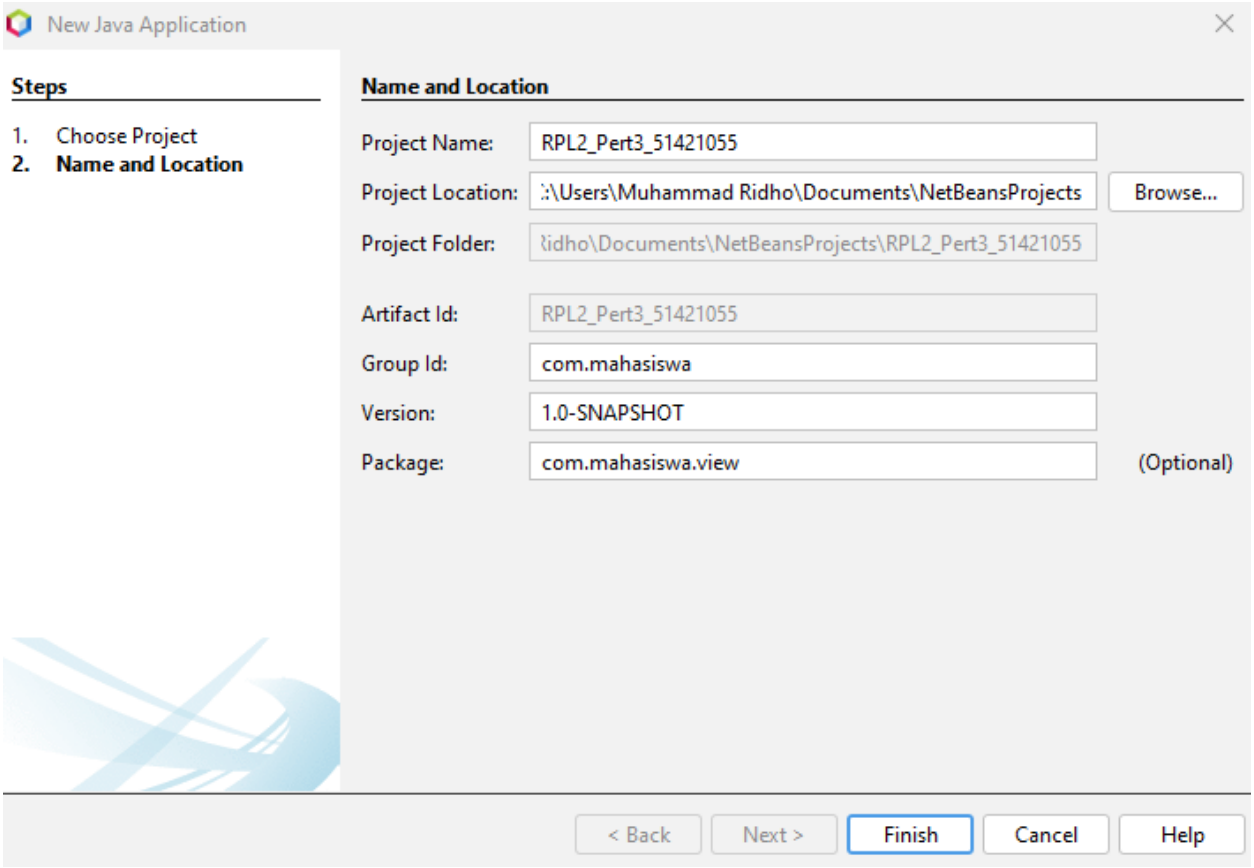
Show this query here again | Retain query box | Rollback when finished

Enable foreign key checks

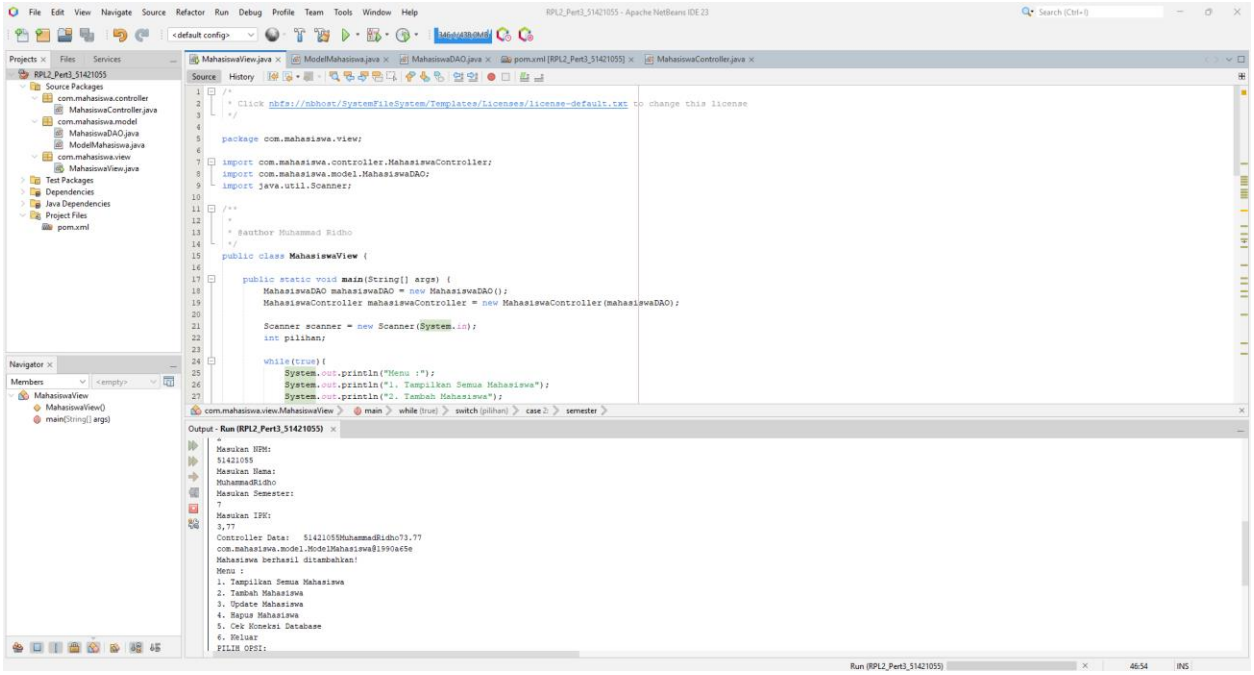
Go


Server: 127.0.0.1 » Database: mvc_db
Structure
SQL
Search
Query
Export
Import
Operations
Privileges
Show query box
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0037 seconds.)
CREATE TABLE `mahasiswa`(`id` int(11) NOT NULL, `ipk` float NOT NULL, `nama` varchar(55) DEFAULT NULL, `npm` varchar(10) DEFAULT NULL, `semester` int(3) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
[Edit inline] [Edit] [Create PHP code]
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0112 seconds.)
ALTER TABLE `mahasiswa` ADD PRIMARY KEY(`id`);
[Edit inline] [Edit] [Create PHP code]
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0123 seconds.)
ALTER TABLE `mahasiswa` MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
[Edit inline] [Edit] [Create PHP code]
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0001 seconds.)
COMMIT;
[Edit inline] [Edit] [Create PHP code]


Server: 127.0.0.1 » Database: mvc_db » Table: mahasiswa
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Table structure
Relation view
Name Type Collation Attributes Null Default Comments Extra Action
1 id int(11) No None AUTO_INCREMENT Change
2 ipk float No None Change
3 nama varchar(55) utf8mb4_general_ci Yes NULL Change
4 npm varchar(10) utf8mb4_general_ci Yes NULL Change
5 semester int(3) No None Change
Check all With selected: Browse Change Drop Primary Unique Index



MahasiswaView.java



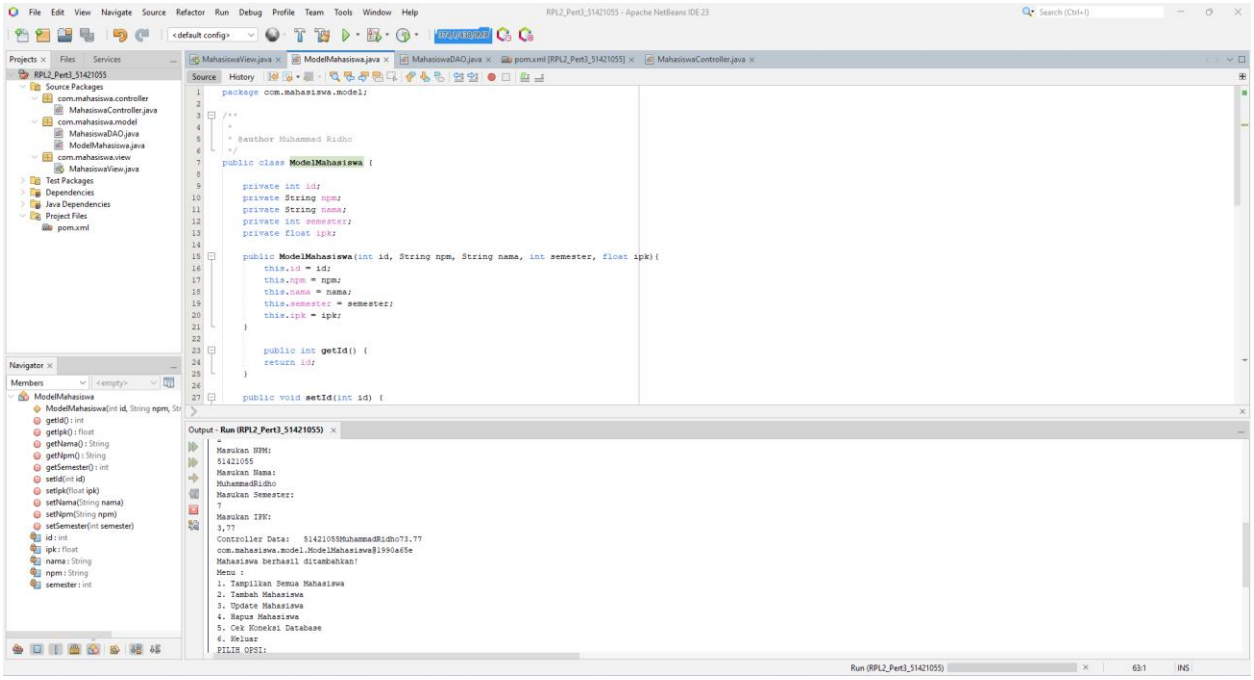
Source Code:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mahasiswa.view;

import com.mahasiswa.controller.MahasiswaController;
import com.mahasiswa.model.MahasiswaDAO;
import java.util.Scanner;

/**
 *
 * @author Muhammad Ridho
 */
public class MahasiswaView {

    public static void main(String[] args) {
        MahasiswaDAO mahasiswaDAO = new MahasiswaDAO();
        MahasiswaController mahasiswaController = new MahasiswaController(mahasiswaDAO);

        Scanner scanner = new Scanner(System.in);
        int pilihan;

        while(true){
            System.out.println("Menu :");
            System.out.println("1. Tampilkan Semua Mahasiswa");
            System.out.println("2. Tambah Mahasiswa");
            System.out.println("3. Update Mahasiswa");
```

```
System.out.println("4. Hapus Mahasiswa");
System.out.println("5. Cek Koneksi Database");
System.out.println("6. Keluar");
System.out.println("PILIH OPSI: ");
pilihan = scanner.nextInt();
scanner.nextLine();

switch(pilihan){
   case 1:
      mahasiswaController.displayAllMahasiswa();
      break;
   case 2:
      System.out.println("Masukan NPM: ");
      String npm = scanner.next();
      System.out.println("Masukan Nama: ");
      String nama = scanner.next();
      System.out.println("Masukan Semester: ");
      int semester = scanner.nextInt();
      System.out.println("Masukan IPK: ");
      float ipk = scanner.nextFloat();

      mahasiswaController.addMahasiswa(npm, nama, semester, ipk);
      break;
   case 3:
      System.out.println("Masukan ID Mahasiswa: ");
      int id = scanner.nextInt();
      scanner.nextLine();

      System.out.println("Masukan NPM: ");
```

```
                String npmBaru = scanner.next();
                System.out.println("Masukan Nama: ");
                String namaBaru = scanner.next();
                System.out.println("Masukan Semester");
                int semesterBaru = scanner.nextInt();
                System.out.println("Masukan IPK: ");
                float ipkBaru = scanner.nextFloat();

                mahasiswaController.updateMahasiswa(id, npmBaru, namaBaru, semesterBaru, ipkBaru);
                break;
            case 4:
                System.out.println("Masukan ID Mahasiswa yg Ingin Dihapus: ");
                int idHapus = scanner.nextInt();
                mahasiswaController.deleteMahasiswa(idHapus);
                break;
            case 5:
                mahasiswaController.checkDatabaseConnection();
                break;
            case 6:
                mahasiswaController.closeConnection();
                System.out.println("Program selesai");
                return;
            default:
                System.out.println("Input Tidak Valid");
        }
    }
  }
}
```

ModelMahasiswa.java



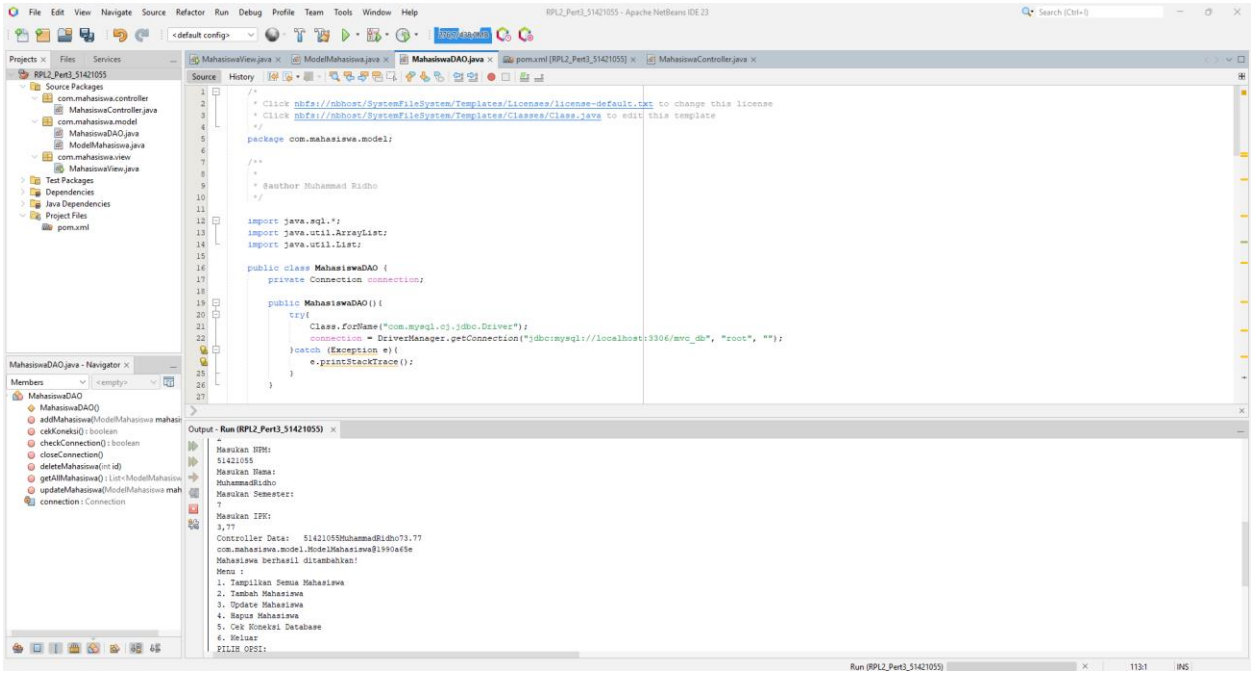
Source Code:

```
package com.mahasiswa.model;


/**
 *
 * @author Muhammad Ridho
 */
public class ModelMahasiswa {


    private int id;
    private String npm;
    private String nama;
    private int semester;
    private float ipk;


    public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk){
```

```
    this.id = id;
    this.npm = npm;
    this.nama = nama;
    this.semester = semester;
    this.ipk = ipk;
}

    public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getNpm() {
    return npm;
}

public void setNpm(String npm) {
    this.npm = npm;
}

public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
```

```
    }

    public int getSemester() {
        return semester;
    }

    public void setSemester(int semester) {
        this.semester = semester;
    }

    public float getIpk() {
        return ipk;
    }

    public void setIpk(float ipk) {
        this.ipk = ipk;
    }
}
```

MahasiswaDAO.java



Source Code:

```
/*
 * 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 com.mahasiswa.model;


/**
 *
 * @author Muhammad Ridho
 */


import java.sql.*;
import java.util.ArrayList;
import java.util.List;
```

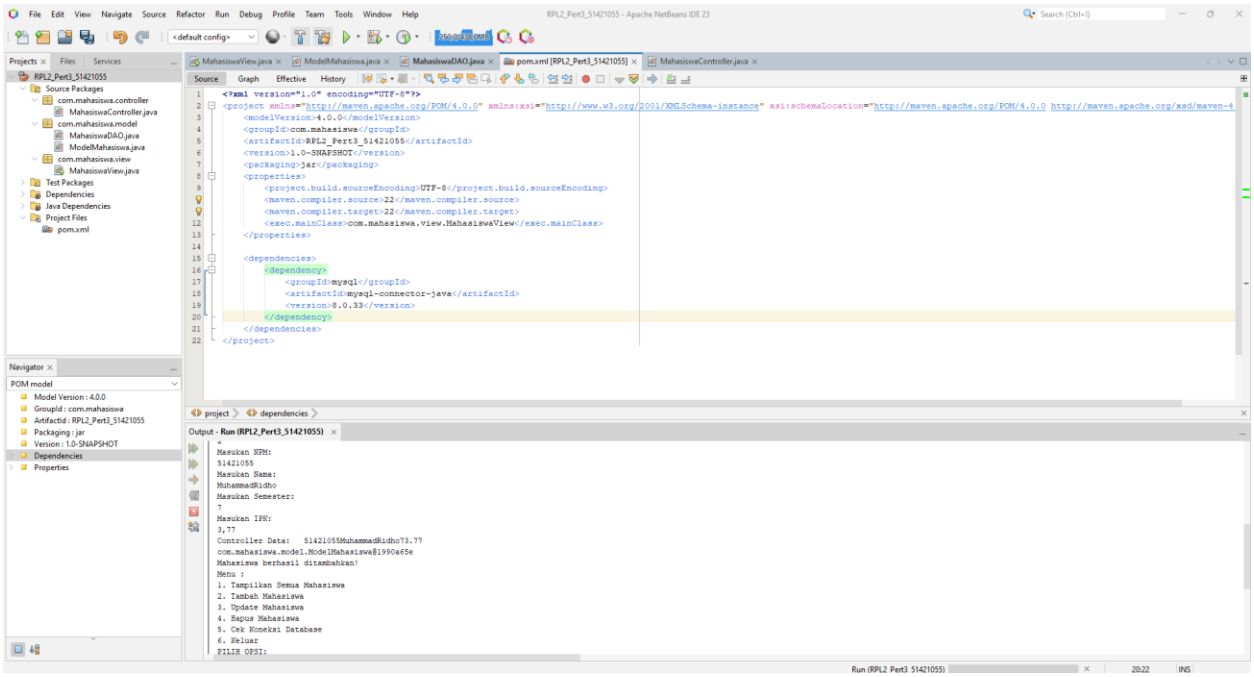
```
public class MahasiswaDAO {
    private Connection connection;

    public MahasiswaDAO(){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mvc_db", "root", "");
        }catch (Exception e){
            e.printStackTrace();
        }
    }

    public boolean cekKoneksi () {
        try{
            if(connection != null && connection.isClosed());
                return true;
        }catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }

    public void addMahasiswa(ModelMahasiswa mahasiswa){
        String sql = "INSERT INTO mahasiswa (npm, nama, semester, ipk) VALUES (?, ?, ?, ?)";
        try{
            PreparedStatement pstmt = connection.prepareStatement(sql);
            pstmt.setString(1, mahasiswa.getNpm());
            pstmt.setString(2, mahasiswa.getNama());
            pstmt.setInt(3, mahasiswa.getSemester());
```

```
        pstmt.setFloat(4, mahasiswa.getIpk());
        pstmt.executeUpdate();
    } catch(SQLException e){
        e.printStackTrace();
    }
}
public List<ModelMahasiswa> getAllMahasiswa(){
    List<ModelMahasiswa> mahasiswaList = new ArrayList<>();
    String sql = "SELECT * FROM mahasiswa";
    try{
        Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery(sql);
        while(rs.next()){
            mahasiswaList.add(new ModelMahasiswa(
                    rs.getInt("id"),
                    rs.getString("npm"),
                    rs.getString("nama"),
                    rs.getInt("semester"),
                    rs.getFloat("ipk")
            ));
        }
    } catch(SQLException e){
        e.printStackTrace();
    }
    return mahasiswaList;
}

public void updateMahasiswa(ModelMahasiswa mahasiswa){
    String sql = "UPDATE mahasiswa SET npm = ?, nama = ?, semester = ?, ipk = ? WHERE id = ?";
```

```
    try{
        PreparedStatement pstmt = connection.prepareStatement(sql);
        pstmt.setString(1, mahasiswa.getNpm());
        pstmt.setString(2, mahasiswa.getNama());
        pstmt.setInt(3, mahasiswa.getSemester());
        pstmt.setFloat(4, mahasiswa.getIpk());
        pstmt.setInt(5, mahasiswa.getId());
        pstmt.executeUpdate();
    } catch(SQLException e){
        e.printStackTrace();
    }
}

public void deleteMahasiswa(int id){
    String sql = "DELETE FROM mahasiswa WHERE id = ?";
    try{
        PreparedStatement pstmt = connection.prepareStatement(sql);
        pstmt.setInt(1, id);
        pstmt.executeUpdate();
    } catch(SQLException e){
        e.printStackTrace();
    }
}

// Method untuk menutup koneksi database
public void closeConnection() {
    try {
        if (connection != null) {
            connection.close();
```

```
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public boolean checkConnection() {
    throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody
}
}
```

Pom.xml

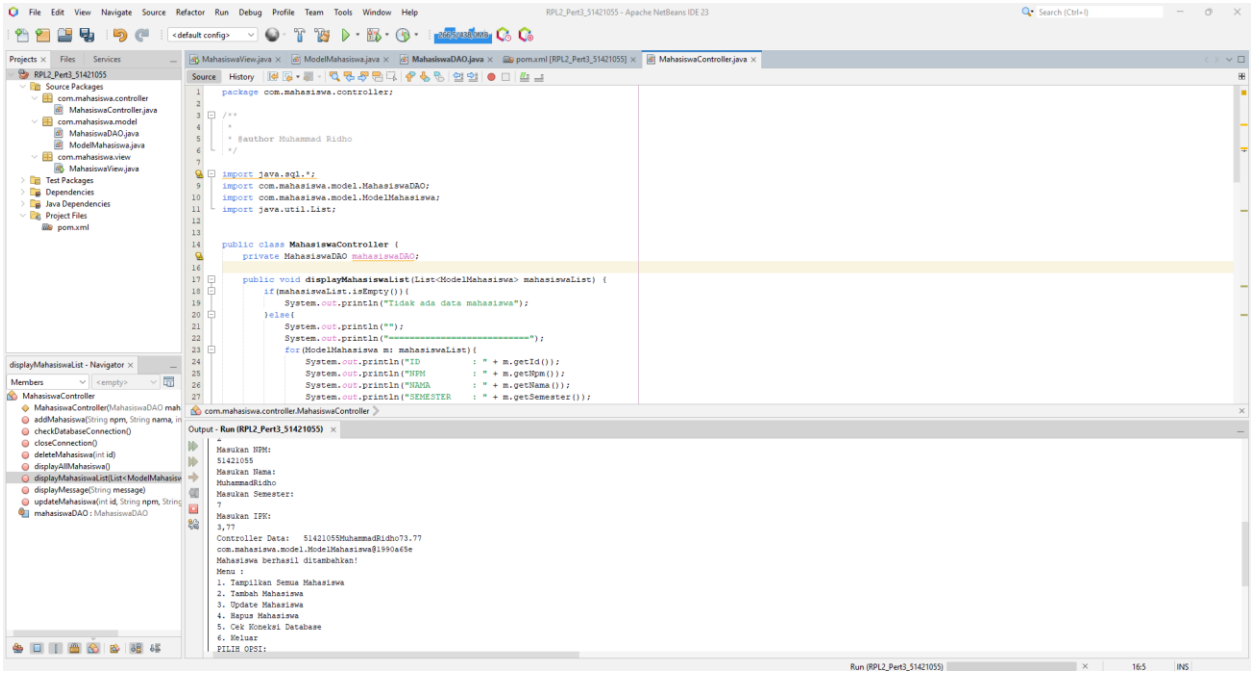


Source Code:

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mahasiswa</groupId>
    <artifactId>RPL2_Pert3_51421055</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>22</maven.compiler.source>
        <maven.compiler.target>22</maven.compiler.target>
        <exec.mainClass>com.mahasiswa.view.MahasiswaView</exec.mainClass>
    </properties>

    <dependencies>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.33</version>
        </dependency>
    </dependencies>
</project>
```

MahasiswaController.java



Source Code:

```
package com.mahasiswa.controller;


/**
 *
 * @author Muhammad Ridho
 */

import java.sql.*;
import com.mahasiswa.model.MahasiswaDAO;
import com.mahasiswa.model.ModelMahasiswa;
import java.util.List;


public class MahasiswaController {
```

```
private MahasiswaDAO mahasiswaDAO;

public void displayMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
   if(mahasiswaList.isEmpty()){
      System.out.println("Tidak ada data mahasiswa");
   }else{
      System.out.println("");
      System.out.println("==========================");
      for(ModelMahasiswa m: mahasiswaList){
         System.out.println("ID         : " + m.getId());
         System.out.println("NPM        : " + m.getNpm());
         System.out.println("NAMA       : " + m.getNama());
         System.out.println("SEMESTER   : " + m.getSemester());
         System.out.println("IPK        : " + m.getIpk());
         System.out.println("==========================");
      }
      displayMessage("Mahasiswa berhasi ditampilkan");
   }
}
public void displayMessage(String message){
   System.out.println(message);
}

public MahasiswaController(MahasiswaDAO mahasiswaDAO){
   this.mahasiswaDAO = mahasiswaDAO;
}

public void checkDatabaseConnection(){
   boolean isConnected = mahasiswaDAO.cekKoneksi();
```

```
    if (isConnected){
        displayMessage("Koneksi ke db berhasil");
    } else{
        displayMessage("Koneksi DB Gagal");
    }
}

public void displayAllMahasiswa(){
    List<ModelMahasiswa> mahasiswaList = mahasiswaDAO.getAllMahasiswa();
    displayMahasiswaList(mahasiswaList);
}

public void addMahasiswa(String npm, String nama, int semester, float ipk){
    ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
    System.out.println("Controller Data:   " + npm + nama + semester + ipk);
    System.out.println(mahasiswaBaru);
    mahasiswaDAO.addMahasiswa(mahasiswaBaru);
    displayMessage("Mahasiswa berhasil ditambahkan!");
}

public void updateMahasiswa(int id, String npm, String nama, int semester, float ipk){
    ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(id, npm, nama, semester, ipk);
    mahasiswaDAO.updateMahasiswa(mahasiswaBaru);
    displayMessage("Mahasiswa berhasil diperbarui!");
}

public void deleteMahasiswa(int id){
    mahasiswaDAO.deleteMahasiswa(id);
    displayMessage("Mahasiswa Berhasil Dihapus!");
```

```
    }

    public void closeConnection() {
        mahasiswaDAO.closeConnection();
    }
}
```

OUTPUT: