```java
package simplilearn;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import simplilearn.RailwayCrossing;
@WebServlet(name = "AddCrossingServlet", urlPatterns = {
"/addCrossing" })
public class AddCrossingServlet extends HttpServlet/
private static final long serialVersionUID = 1L;

protected void doPost(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
// Retrieve the form data from the request
String name = request.getParameter("name");
String address = request.getParameter("address");
String landmark = request.getParameter("landmark");
String trainSchedules = request.getParameter("trainSchedules");
String personInCharge = request.getParameter("personInCharge");
String status = request.getParameter("status");
// Create a new RailwayCrossing object with the form data
RailwayCrossing crossing = new RailwayCrossing();
crossing.setName(name);
crossing.setAddress(address);
crossing.setLandmark(landmark);
crossing.setTrainSchedule(trainSchedules);
crossing.setPersonInCharge(personInCharge);
crossing.setStatus(status);
// Save the new crossing to the database
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
crossingDAO.addCrossing(crossing);
// Redirect to the admin homepage after adding the crossing
response.sendRedirect("adminHome.jsp");
}
```

```
package simplilearn;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

public class AddToFavoritesServlet extends HttpServlet {
    /**
 *
 */
private static final long serialVersionUID = 1L;

protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        String crossingId = request.getParameter("crossingId");
        if (crossingId != null && !crossingId.isEmpty()) {
            int railwayCrossingId = Integer.parseInt(crossingId);
            // Perform the database operation to add the crossing to
favorites
            RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
            crossingDAO.addToFavorites(railwayCrossingId); //
Implement this method in RailwayCrossingDAO

        }
        // Redirect back to the userHome.jsp page
        response.sendRedirect("userHome.jsp");
    }
}
```

```
package simplilearn;

import java.io.IOException;

import java.util.List;


import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/adminHomePage")
public class AdminHomePageServlet extends HttpServlet {
   /**
 *
 */
private static final long serialVersionUID = 1L;
private RailwayCrossingDAO crossingDAO;

   public void init() {
      crossingDAO = new RailwayCrossingDAO();
   }
   protected void doGet(HttpServletRequest request, HttpServletResponse
                        response)
                              throws ServletException, IOException {
                        // Fetch all railway crossing details from the database
                        List<RailwayCrossing> crossings = crossingDAO.getAllCrossings();

                        // Pass the crossing details to the JSP page
                        request.setAttribute("crossings", crossings);
                        request.getRequestDispatcher("adminHome.jsp").forward(request,
                        response);
                     }
                     }

package simplilearn;
```

```
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/adminLogin")
public class AdminLoginServlet extends HttpServlet {
   private static final long serialVersionUID = 1L;

   private static final String ADMIN_ID = "soumya.rout@gmail.com";
   private static final String ADMIN_PASSWORD = "soumyarout";

   protected void doPost(HttpServletRequest request, HttpServletResponse
response)
         throws ServletException, IOException {
      String email = request.getParameter("email");
      String password = request.getParameter("password");

      if (email != null && email.equals(ADMIN_ID) && password != null &&
password.equals(ADMIN_PASSWORD)) {
         response.sendRedirect("adminHome.jsp");
      } else {
         request.getRequestDispatcher("adminLogin.jsp").forward(request,
response);
      }
   }
```

```
}

package simplilearn;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class DatabaseConnection {
   private static final String URL =
"jdbc:mysql://localhost:3306/railway_crossing_status";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "123456789";

   public static Connection getConnection() {
      Connection connection = null;
      try {
         Class.forName("com.mysql.jdbc.Driver");
         connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
      } catch (ClassNotFoundException | SQLException e) {
         e.printStackTrace();
      }
      return connection;
   }
}
package simplilearn;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/deleteCrossing")
public class DeleteCrossingServlet extends HttpServlet {
   protected void doPost(HttpServletRequest request, HttpServletResponse
response)
          throws ServletException, IOException {
       // Retrieve the crossing ID from the request
       int crossingId = Integer.parseInt(request.getParameter("id"));

       // Perform the delete operation (implement your logic)
       RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
       crossingDAO.deleteCrossing(crossingId);
       // Redirect to the admin homepage after the deletion
       response.sendRedirect("adminHome.jsp");
   }
}

package simplilearn;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

```
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
   protected void doPost(HttpServletRequest request, HttpServletResponse
response)
         throws ServletException, IOException {
      // Retrieve the form data
      String email = request.getParameter("email");
      String password = request.getParameter("password");

      // Check if the provided credentials are valid
      UserDAO userDAO = new UserDAO();
      User user = userDAO.getUserByEmail(email);

      if (user != null && user.getPassword().equals(password)) {
         // Authentication successful
         // You can store user details in session for further use
         HttpSession session = request.getSession();
         session.setAttribute("user", user);

         // Redirect to a success page or perform any other necessary actions

         response.sendRedirect("userHome.jsp");
      } else {
         // Authentication failed
         // Redirect to an error page or display an error message
         response.sendRedirect("userRegister.jsp?error=1");
      }
   }
```

```
}



package simplilearn;



public class RailwayCrossing {
private int id;
private String name;
private String address;
private String landmark;
private String trainSchedule;
private String personInCharge;
private String status;
public RailwayCrossing() {
super();
// TODO Auto-generated constructor stub
}
public RailwayCrossing(int id, String name, String address, String
landmark, String trainSchedule,
String personInCharge, String status) {
super();
this.id = id;
this.name = name;
this.address = address;
this.landmark = landmark;
this.trainSchedule = trainSchedule;
this.personInCharge = personInCharge;
this.status = status;
}
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
```

```
this.name = name;
}
public String getAddress() {
return address;
}
public void setAddress(String address) {
this.address = address;
}
public String getLandmark() {
return landmark;
}
public void setLandmark(String landmark) {
this.landmark = landmark;
}
public String getTrainSchedule() {
return trainSchedule;
}
public void setTrainSchedule(String trainSchedule) {
this.trainSchedule = trainSchedule;
}
public String getPersonInCharge() {
return personInCharge;
}
public void setPersonInCharge(String personInCharge) {
this.personInCharge = personInCharge;
}
public String getStatus() {
return status;
}
public void setStatus(String status) {
this.status = status;
}
public void setId(String string) {
// TODO Auto-generated method stub
}
}


package simplilearn;

import java.sql.Connection;
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import simplilearn.RailwayCrossing;
public class RailwayCrossingDAO {
private Connection connection;
public RailwayCrossingDAO() {
// Initialize the database connection
connection = DatabaseConnection.getConnection();
}
// Fetch all values from the database
public List<RailwayCrossing> getAllCrossings() {
List<RailwayCrossing> crossings = new ArrayList<>();
try {
String query = "SELECT * FROM railway_crossing";
PreparedStatement statement = connection.prepareStatement(query);
ResultSet resultSet = statement.executeQuery();
while (resultSet.next()) {
RailwayCrossing crossing = new RailwayCrossing();
crossing.setId(resultSet.getInt("id"));
crossing.setName(resultSet.getString("name"));
crossing.setAddress(resultSet.getString("address"));
crossing.setLandmark(resultSet.getString("landmark"));
crossing.setTrainSchedule(resultSet.getString("train_schedule"));
crossing.setPersonInCharge(resultSet.getString("person_in_char"));
crossing.setStatus(resultSet.getString("status"));
crossings.add(crossing);
}
} catch (SQLException e) {
e.printStackTrace();
}
return crossings;
}
// Search values by id and display it from the database
public RailwayCrossing getCrossingById(int crossingId) {
RailwayCrossing crossing = null;
try {
 String query = "SELECT * FROM railway_crossing WHERE id = ?";
PreparedStatement statement = connection.prepareStatement(query);
statement.setInt(1, crossingId);
```

```
ResultSet resultSet = statement.executeQuery();
if (resultSet.next()) {
crossing = new RailwayCrossing();
crossing.setId(resultSet.getInt("id"));
crossing.setName(resultSet.getString("name"));
crossing.setAddress(resultSet.getString("address"));
crossing.setLandmark(resultSet.getString("landmark"));
crossing.setTrainSchedule(resultSet.getString("train_schedule"));

crossing.setPersonInCharge(resultSet.getString("person_in_charge"));
crossing.setStatus(resultSet.getString("status"));
}
} catch (SQLException e) {
e.printStackTrace();
}
return crossing;
}
// update the values in the database
public void updateCrossing(RailwayCrossing crossing) {
try {
String query = "UPDATE railway_crossing SET name=?, address=?,
landmark=?, train_schedule=?, person_in_charge=?, status=?, WHERE
id=?";
PreparedStatement statement = connection.prepareStatement(query);
statement.setString(1, crossing.getName());
statement.setString(2, crossing.getAddress());
statement.setString(3, crossing.getLandmark());
statement.setString(4, crossing.getTrainSchedule());
statement.setString(5, crossing.getPersonInCharge());
statement.setString(6, crossing.getStatus());
statement.setInt(7, crossing.getId());
statement.executeUpdate();
} catch (SQLException e) {
e.printStackTrace();
}
}
// Delete Railway Crossing
public void deleteCrossing(int crossingId) {
try {
connection.setAutoCommit(false); // Start transaction
// Delete associated favorite crossings
```

```
String deleteFavoriteCrossingsQuery = "DELETE FROM favorite_crossing
WHERE railway_crossing_id = ?";

 try (PreparedStatement deleteFavoriteCrossingsStatement =
 connection
 .prepareStatement(deleteFavoriteCrossingsQuery)) {
 deleteFavoriteCrossingsStatement.setInt(1, crossingId);
 deleteFavoriteCrossingsStatement.executeUpdate();
 }

 // Delete the railway crossing
 String deleteRailwayCrossingQuery = "DELETE FROM railway_crossing
WHERE id = ?";

 try (PreparedStatement deleteRailwayCrossingStatement = connection
 .prepareStatement(deleteRailwayCrossingQuery)) {
 deleteRailwayCrossingStatement.setInt(1, crossingId);
 deleteRailwayCrossingStatement.executeUpdate();
 }

 connection.commit(); // Commit the transaction
 connection.setAutoCommit(true); // Reset auto-commit to true
 } catch (SQLException e) {
 try {
 connection.rollback(); // Rollback the transaction if an error
occurs;

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


 // Add values to the database
 public void addCrossing(RailwayCrossing crossing) {
 try {
 String query = "INSERT INTO railway_crossing (name, address,
landmark, train_schedule, person_in_charge, status) VALUES (?, ?, ?,
?, ?, \r\n"
           + "         ?)";
```

```
PreparedStatement statement = connection.prepareStatement(query);
statement.setString(1, crossing.getName());
statement.setString(2, crossing.getAddress());
statement.setString(3, crossing.getLandmark());
statement.setString(4, crossing.getTrainSchedule());
statement.setString(5, crossing.getPersonInCharge());
statement.setString(6, crossing.getStatus());
statement.executeUpdate();
} catch (SQLException e) {
e.printStackTrace();
}
}

public List<RailwayCrossing> getFavoriteCrossings() {
List<RailwayCrossing> favoriteCrossings = new ArrayList<>();

try (Connection connection = DatabaseConnection.getConnection()) {
String query = "SELECT rc.* FROM railway_crossing rc "
+ "JOIN favorite_crossing fc ON rc.id = fc.railway_crossing_id";

PreparedStatement statement = connection.prepareStatement(query);
ResultSet resultSet = statement.executeQuery();

while (resultSet.next()) {
RailwayCrossing crossing = new RailwayCrossing();
crossing.setId(resultSet.getInt("id"));
crossing.setName(resultSet.getString("name"));
crossing.setAddress(resultSet.getString("address"));
crossing.setLandmark(resultSet.getString("landmark"));
crossing.setTrainSchedule(resultSet.getString("train_schedule"
));
crossing.setPersonInCharge(resultSet.getString("person_in_charge"));

crossing.setStatus(resultSet.getString("status"));

favoriteCrossings.add(crossing);
}
} catch (SQLException e) {
e.printStackTrace();
}

return favoriteCrossings;
}
```

```
 public void addToFavorites(int crossingId) {
 try (Connection connection = DatabaseConnection.getConnection()) {
 String sql = "INSERT INTO favorite_crossing (railway_crossing_id)
VALUES (?)";

 PreparedStatement statement = connection.prepareStatement(sql);
 statement.setInt(1, crossingId);
 statement.executeUpdate();
 } catch (SQLException e) {
 e.printStackTrace();
 // Handle the exception as needed
 }
 }

 public void removeFromFavorites(int crossingId) {
      try (Connection connection = DatabaseConnection.getConnection())
{
 String sql = "DELETE FROM favorite_crossing WHERE railway_crossing_id
= ?";

 PreparedStatement statement = connection.prepareStatement(sql);
 statement.setInt(1, crossingId);
 statement.executeUpdate();
 } catch (SQLException e) {
 e.printStackTrace();
 // Handle the exception as needed
 }
 }

 }
```

```
package simplilearn;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
* Servlet implementation class RemoveFromFavoritesServlet
```

```
*/
public class RemoveFromFavoritesServlet extends HttpServlet {
   protected void doPost(HttpServletRequest request, HttpServletResponse
response)
         throws ServletException, IOException {
      // TODO Auto-generated method stub
      String crossingId = request.getParameter("crossingId");
      if (crossingId != null && !crossingId.isEmpty()) {
         int railwayCrossingId = Integer.parseInt(crossingId);
         // Perform the database operation to remove the crossing fromfavorites

         RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
         crossingDAO.removeFromFavorites(railwayCrossingId); // Implementthis method in
RailwayCrossingDAO

      }
      // Redirect back to the userHome.jsp page
      response.sendRedirect("userHome.jsp");
   }

}
```

```
package simplilearn;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import simplilearn.RailwayCrossing;
@WebServlet(name = "SearchCrossingServlet", urlPatterns = {
"/searchCrossing"
})
public class SearchCrossingServlet extends HttpServlet {
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
// Retrieve the search ID from the request
int searchId = Integer.parseInt(request.getParameter("searchId"));
// Implement your logic to search for the crossing by ID and retrieve
its
```

```
// details from the database
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
RailwayCrossing crossing = crossingDAO.getCrossingById(searchId);
// Set the search result in the request attribute
request.setAttribute("crossing", crossing);
// Redirect to the admin homepage with the search results
request.getRequestDispatcher("adminHome.jsp").forward(request,
response);
}
}

package simplilearn;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import simplilearn.RailwayCrossing;
@WebServlet("/updateCrossing")
public class UpdateCrossingServlet extends HttpServlet {
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
try {
// Retrieve the crossing ID from the hidden field
String idParam = request.getParameter("crossingId");
if (idParam != null && !idParam.isEmpty()) {
 int crossingId = Integer.parseInt(idParam);

// Retrieve the updated details for the crossing
String name = request.getParameter("name");
String address = request.getParameter("address");
String landmark = request.getParameter("landmark");
String trainSchedules =
request.getParameter("trainSchedules");
String personInCharge =
request.getParameter("personInCharge");
String status = request.getParameter("status");
// Create a new RailwayCrossing object with the updated details
```

```
RailwayCrossing updatedCrossing = new RailwayCrossing();
updatedCrossing.setId(crossingId);
updatedCrossing.setName(name);
updatedCrossing.setAddress(address);
updatedCrossing.setLandmark(landmark);
updatedCrossing.setTrainSchedule(trainSchedules);
updatedCrossing.setPersonInCharge(personInCharge);
updatedCrossing.setStatus(status);
// Perform the update operation
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
crossingDAO.updateCrossing(updatedCrossing);
// Redirect to the admin homepage after the update
response.sendRedirect("adminHome.jsp");
} else {
// Handle case where idParam is empty or null
throw new ServletException("Crossing ID is missing.");
}
} catch (NumberFormatException e) {
// Handle case where idParam is not a valid integer
throw new ServletException("Invalid Crossing ID format.", e);
} catch (Exception e) {
// Handle other exceptions
throw new ServletException("An error occurred during the crossing
update.", e);
}
}
}
```

```
package simplilearn;

public class User {
private String name;
private String email;
private String password;
public User() {
super();
}
public User(String name, String email, String password) {
this.name = name;
this.email = email;
this.password = password;
```

```
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getEmail() {
return email;
}
public void setEmail(String email) {
this.email = email;
}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
}
```

```
package simplilearn;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
public class UserDAO {
   private Connection connection;

   public UserDAO() {
      // Initialize the database connection
      connection = DatabaseConnection.getConnection();
   }

   public void registerUser(User user) {
      try {
         String query = "INSERT INTO user_signup (name, email, password) VALUES (?, ?, ?)";

         PreparedStatement statement = connection.prepareStatement(query);
         statement.setString(1, user.getName());
         statement.setString(2, user.getEmail());
         statement.setString(3, user.getPassword());
         statement.executeUpdate();
```

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

    public boolean loginUser(String email, String password) {
        try {
            String query = "SELECT * FROM user_signup WHERE email=? AND password=?";

            PreparedStatement statement = connection.prepareStatement(query);
            statement.setString(1, email);
            statement.setString(2, password);
            ResultSet resultSet = statement.executeQuery();
            return resultSet.next();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }

    public User getUserByEmail(String email) {
        try {
            String query = "SELECT * FROM user_signup WHERE email=?";
            PreparedStatement statement = connection.prepareStatement(query);
            statement.setString(1, email);
            ResultSet resultSet = statement.executeQuery();

            if (resultSet.next()) {
                User user = new User();
                user.setName(resultSet.getString("name"));
                user.setEmail(resultSet.getString("email"));
                user.setPassword(resultSet.getString("password"));
                return user;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }

}
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Add Railway Crossing</title>
</head>
<body>
<div class="panel">
<h2>Railway Crossings</h2>
<h4>Add Railway Crossing Information</h4>
<form action="addCrossing" method="post">
<label for="name">Enter Name</label> <input type="text"
id="name"
name="name" required><br> <label
for="address">Address</label>
<input type="text" id="address" name="address"
required><br>
<label for="landmark">Landmark</label> <input type="text"
id="landmark" name="landmark" required><br> <label
for="trainSchedules">Train Schedules</label> <input
type="text"
id="trainSchedules" name="trainSchedules"
required><br>
<label for="personInCharge">Person in Charge</label>
<input
type="text" id="personInCharge"
name="personInCharge" required><br>
<label for="status">Crossing Status</label> <select
id="status"
name="status">
<option value="Open">Open</option>
<option value="Closed">Closed</option>
</select><br>
<button type="submit">Submit</button>
</form>
```

```
</div>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ page import="simplilearn.RailwayCrossing"%>
<%@ page import="simplilearn.RailwayCrossingDAO"%>
<%@ page import="java.util.List"%>
<%
// Create an instance of RailwayCrossingDAO
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
// Retrieve all crossings from the DAO
List<RailwayCrossing> crossings = crossingDAO.getAllCrossings();
// Get the ID from the search request parameter
String searchId = request.getParameter("searchId");
// Check if the searchId parameter is provided
if (searchId != null && !searchId.isEmpty()) {
// Parse the searchId as an int
int crossingId = Integer.parseInt(searchId);
// Retrieve the crossing by ID from the DAO
RailwayCrossing searchedCrossing =
crossingDAO.getCrossingById(crossingId);

// Set the crossing as an attribute to be displayed in the table
request.setAttribute("crossing", searchedCrossing);
}
%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1>Admin Home Page</h1>
<h3>
Railway Crossing[<%=crossings.size()%>]
```

```
</h3>
<!-- Buttons -->
<div class="button-container">
<form action="HomePage.jsp" method="post">
<button type="submit">Home</button>
</form>
<form action="addRailCrossingForm.jsp" method="post">
<button type="submit">Add Railway Crossing</button>
</form>
<form action="searchCrossing" method="post">
<input type="text" name="searchId" placeholder="Enter
ID">
<button type="submit">Search Crossing</button>
</form>
<form action="adminHome.jsp" method="post">
<button type="submit">Display All Railway
Crossings</button>
</form>
<!-- Logout Button -->
<form action="adminRegister.jsp" method="post">
<button type="submit">Logout</button>
</form>
</div>
<!-- Table -->
<table>
<thead>
<tr>
<th>Sr.No</th>
<th>Name</th>
<th>Address</th>
<th>Landmark</th>
<th>Train Schedule</th>
<th>Person In-Charge</th>
<th>Status</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<%
if (request.getAttribute("crossing") != null) { // Check

RailwayCrossing crossing = (RailwayCrossing)
```

```
request.getAttribute("crossing");
%>
<tr>
<td><%=crossing.getId()%></td>
<td><%=crossing.getName()%></td>
<td><%=crossing.getAddress()%></td>
<td><%=crossing.getLandmark()%></td>
<td><%=crossing.getTrainSchedule()%></td>
<td><%=crossing.getPersonInCharge()%></td>
<td><%=crossing.getStatus()%></td>
<td></td>
</tr>
<%
} else { // Display all crossings
for (RailwayCrossing anotherCrossing : crossings) {
%>
<tr>
<td><%=anotherCrossing.getId()%></td>
<td><%=anotherCrossing.getName()%></td>
<td><%=anotherCrossing.getAddress()%></td>
<td><%=anotherCrossing.getLandmark()%></td>
<td><%=anotherCrossing.getTrainSchedule()%></td>
<td><%=anotherCrossing.getPersonInCharge()%></td>
<td><%=anotherCrossing.getStatus()%></td>
<td>
<form action="updateRailCrossingForm.jsp"
method="post">
<input type="hidden" name="id"
value="<%=anotherCrossing.getId()%>">
<button type="submit" class="update
button">Update</button>
</form>
<form action="deleteCrossing" method="post">
<input type="hidden" name="id"
value="<%=anotherCrossing.getId()%>">
<button type="submit" class="delete
button">Delete</button>
</form>
</td>
</tr>
<%
}
```

```
}
%>
</tbody>
</table>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>>Admin Login</title>
</head>
<body>
<div class="panel">
<h2>Railway Crossing</h2>
<h4>Admin Login</h4>
<form action="adminLogin" method="post">
<label for="email">Email:</label> <input type="text"
id="email"
name="email" required><br> <label
for="password">Password:</label>
<input type="password" id="password" name="password"
required><br>
<button type="submit">Login</button>
</form>
</div>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
```

```
<title>>Railway Crossing Status</title>
</head>
<body>
<nav>
<div class="container-fluid">
<a class="navbar-brand" href="#">Railway Crossing
Status</a>
<button class="navbar-toggler" type="button"
data-bs-toggle="collapse" data-bs
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria
expanded="false"
aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse"
id="navbarSupportedContent">
<ul class="navbar-nav custom-ml-auto">
<!-- Move the "Home", "Quiz", "Login/Signup"
links to the right using custom CSS -->
<li class="nav-item"><a class="nav-link
active"
aria-current="page"
href="#">Home</a></li>
<li class="nav-item"><a class="nav-link"
href="adminLogin.jsp">Admin-Login/Signup</a></li>
<li class="nav-item"><a class="nav-link"
href="userLogin.jsp">User-Login/Signup</a>
</li>
</ul>
</div>
</div>
</nav>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.b
undle.min.js"
integrity="sha384
geWF76RCwLtnZ8qwWowPQNguL3RmwHVBC9FhGdlKrxdiJJigb/j/68SIy3Te4Bkz"
crossorigin="anonymous"></script>

</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Search Crossing</title>
</head>
<body>
<h1>Search Crossing</h1>
<form action="searchCrossing" method="post">
<label for="searchId">Search ID:</label> <input type="number"
id="searchId" name="searchId" required> <input
type="submit"
value="Search">
</form>
<c:if test="${not empty crossing}">
<h2>Search Result</h2>
<table>
<tr>
<th>ID</th>
<th>Name</th>
<th>Address</th>
<th>Landmark</th>
<th>Train Schedule</th>
<th>Person in Charge</th>
<th>Status</th>
</tr>
<tr>
<td>${crossing.id}</td>
<td>${crossing.name}</td>
<td>${crossing.address}</td>
<td>${crossing.landmark}</td>
<td>${crossing.trainSchedule}</td>
<td>${crossing.personInCharge}</td>
<td>${crossing.status}</td>
</tr>
</table>
</c:if>
```

```
<a href="adminHome.jsp">Back to Admin Homepage</a>
</body>
</html>



<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Update Railway Crossing</title>
</head>
<body>
<div class="panel">
<h2>Railway Crossings</h2>
<h4>Update Railway Crossing Information</h4>
<form action="updateCrossing" method="post">
<label for="crossingId">Crossing ID</label> <input
type="text"
id="crossingId" name="crossingId"
value="${crossing.crossingId}"
required><br> <label for="name">Enter Name</label>
<input
type="text" id="name" name="name"
value="${crossing.name}" required><br>
<label for="address">Address</label> <input type="text"
id="address"
name="address" value="${crossing.address}"
required><br>
<label for="landmark">Landmark</label> <input type="text"
id="landmark" name="landmark"
value="${crossing.landmark}" required><br>
<label for="trainSchedules">Train Schedules</label>
<input
type="text" id="trainSchedules"
name="trainSchedules"
value="${crossing.trainSchedule}" required><br>
<label
for="personInCharge">Person in Charge</label>
<input type="text"
```

```
id="personInCharge" name="personInCharge"
value="${crossing.personInCharge}" required><br>
<label >
<for="status">Crossing Status</label> <select
id="status"
name="status">
<option value="Open" ${crossing.status == 'Open' ?
'selected' : ''}>Open</option>
<option value="Closed"
${crossing.status == 'Closed' ? 'selected' :
''}>Closed</option>
</select><br>
<button type="submit">Submit</button>
</form>
</div>

</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page import="simplilearn.RailwayCrossing"%>
<%@ page import="simplilearn.RailwayCrossingDAO"%>
<%@ page import="java.util.List"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1>User Home Page</h1>

<!-- Display Search Crossings button -->
<button>Home</button>

<!-- Display All button -->
<button>All Crossings</button>


<!-- Display Favorite Crossings button -->
```

```
<button>Favorite Crossings</button>

<!-- Display Search Crossings button -->
<button>Search Crossings</button>

<!-- Logout button -->
<button>Logout</button>

<%
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
List<RailwayCrossing> allCrossings = crossingDAO.getAllCrossings();
for (RailwayCrossing crossing : allCrossings) { String status =
crossing.getStatus();
String statusClass = status.equalsIgnoreCase("Open") ? "statusopen" :
"status-closed";
%>
<div>
<h3><%=crossing.getName()%></h3>
<p>
Status: <strong><span class="status
<%=statusClass%>"><%=crossing.getStatus()%></span></strong>
</p>
<p>
Person in Charge: <strong><%=crossing.getPersonInCharge()%></strong>
</p>
<p>
Train Schedules: <strong><%=crossing.getTrainSchedule()%></strong>
</p>
<p>
Landmark: <strong><%=crossing.getLandmark()%></strong>
</p>
<p>
Address: <strong><%=crossing.getAddress()%></strong>
</p>
<form action="addToFavorites" method="post" style="display: inline;">
<input type="hidden" name="crossingId" value="<%=crossing.getId()%>">
<button type="submit"
style="border-radius: 10px; font-weight: bold; margin-top: 10px;
margin-left: 0px;">ADD
TO FAVORITES</button>
</form>
</div>
```

```
<%
}
%>
</body>
</html>



<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page import="simplilearn.RailwayCrossing"%>
<%@ page import="simplilearn.RailwayCrossingDAO"%>
<%@ page import="java.util.List"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>>Railway Crossings - Favorite Crossings</title>
</head>
<body>
<h2>User Home Page</h2>
<!-- Display All button -->
<button onclick="location.href='userHome.jsp'">All Crossings</button>
<!-- Favorite Crossings Container -->
<div class="favorites">
<%
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
List<RailwayCrossing> favoriteCrossings =
crossingDAO.getFavoriteCrossings();
for (RailwayCrossing crossing : favoriteCrossings) {
String status = crossing.getStatus();
String statusClass = status.equalsIgnoreCase("Open") ? "status-open" :
"status-closed";
%>
<div class="container">
<h3><%=crossing.getName()%></h3>
<p>
Status: <strong><span class="status
<%=statusClass%>"><%=status%></span></strong>
</p>
<p>
Person in Charge:
```

```
<strong><%=crossing.getPersonInCharge()%></strong>
</p>
<p>
Train Schedules:
<strong><%=crossing.getTrainSchedule()%></strong>
</p>
<p>
Landmark:
<strong><%=crossing.getLandmark()%></strong>
</p>
<p>
Address:
<strong><%=crossing.getAddress()%></strong>
</p>
<form action="removeFromFavorites" method="post"
style="display: inline;">
<input type="hidden" name="crossingId"
value="<%=crossing.getId()%>">
<button type="submit"
style="border-radius: 10px; font-weight:
bold; margin-top: 10px;">REMOVE
FROM FAVORITES</button>
</form>
</div>
<%
}
%>
</div>
</body>
</html>



<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>User Login</title>
</head>
<body>
```

```
<div class="panel">
<h2>Railway Crossing</h2>
<h4>User Login</h4>
<form action="login" method="post">
<label for="email">Email:</label> <input type="text"
id="email"
name="email" required><br> <label
for="password">Password:</label>
<input type="password" id="password" name="password"
required><br>
<button type="submit">Login</button>
</form>
<div class="signup-link">
<p>
Don't have an account? <a
href="userRegister.jsp">Create New
Account</a>
</p>
</div>
</div>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>User Registration</title>
</head>
<body>
<div class="panel">
<h2>Railway Crossing</h2>
<h4>User Register</h4>
<form action="register" method="post">
<label for="name">Name:</label> <input type="text"
id="name"
name="name" required><br> <label
for="email">Email:</label>
```

```
<input type="text" id="email" name="email" required><br>
<label for="password">Password:</label> <input
type="password"
id="password" name="password" required><br>
<button type="submit">Register</button>
</form>
<div class="signup-link">
<p>
Already have an account? <a
href="userLogin.jsp">Sign in</a>
</p>
</div>
</div>
</body>
</html>


<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page import="simplilearn.RailwayCrossing"%>
<%@ page import="simplilearn.RailwayCrossingDAO"%>
<%@ page import="java.util.List"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>>Railway Crossings - Search Crossings</title>
</head>
<body>
<h2>Search Crossings</h2>
<!-- Display All button -->
<button>All
Crossings</button>
<!-- Display Favorite Crossings button -->
<button onclick="location.href='userHomeFavoriteCrossing.jsp'"
class="favouriteCrossing-button"
style="border-radius: 10px; font-weight: bold;">Favorite
Crossings</button>
<!-- Search input -->
<div class="search-container">
<input type="text" id="searchInput" placeholder="Search by
name"
onkeyup="searchCrossings()">
```

```
<!-- Logout button -->
<button onclick="location.href='userLogin.jsp'" class="logout
button"
>Logout</button>
</div>
<!-- Search Results Container -->
<div class="search-results">
<%
RailwayCrossingDAO crossingDAO = new RailwayCrossingDAO();
List<RailwayCrossing> allCrossings =
crossingDAO.getAllCrossings();
for (RailwayCrossing crossing : allCrossings) {
String status = crossing.getStatus();
String statusClass = status.equalsIgnoreCase("Open") ?
"status-open" : "status-closed";
%>
<div class="container crossing">
<h3><%=crossing.getName()%></h3>
<p>
Status: <strong><span class="status
<%=statusClass%>"><%=crossing.getStatus()%></span></strong>
</p>
<p>
Person in Charge:
<strong><%=crossing.getPersonInCharge()%></strong>
</p>
<p>
Train Schedules:
<strong><%=crossing.getTrainSchedule()%></strong>
</p>
<p>
Landmark:
<strong><%=crossing.getLandmark()%></strong>
</p>
<p>
Address:
<strong><%=crossing.getAddress()%></strong>
</p>
<form action="addToFavorites" method="post"
style="display: inline;">
<input type="hidden" name="crossingId"
value="<%=crossing.getId()%>">
```

```
<button type="submit"
style="border-radius: 10px; font-weight:
bold; margin-top: 10px; margin-left: 0px;">ADD
TO FAVORITES</button>
</form>
</div>
<%
}
%>
</div>
</body>
</html>

CREATE TABLE user_signup (
      id INT PRIMARY KEY AUTO_INCREMENT,
      name VARCHAR(255) NOT NULL,
      email VARCHAR(255) NOT NULL,
      password VARCHAR(255) NOT NULL
     );

CREATE TABLE favorite_crossing (
       id INT PRIMARY KEY AUTO_INCREMENT,
       railway_crossing_id INT,
       FOREIGN KEY (railway_crossing_id) REFERENCES
railway_crossing(id)
     );
CREATE TABLE railway_crossing (
  id INT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(255),
  address VARCHAR(255),
  landmark VARCHAR(255),
  train_schedule VARCHAR(255),
  person_in_charge VARCHAR(255),
  status VARCHAR(50)
);
```