Initialization:

- Start the application.
- Display the welcome screen.
- Prompt the user to choose between the "Public" and "Government" sections.

Public Section:

If the user selects "Public," perform the following:

a. Prompt the user to either create an account or log in.

b. If the user chooses to create an account, collect their name, email, and password. Store this information in the database.

c. If the user chooses to log in, verify their credentials against the database.

d. Once logged in, display the user's home page.

User Home Page:

a. Display options: "All," "Favorite Crossings," and "Logout."

b. If "All" is selected, fetch and display details of all railway crossings, including their names, statuses, and other information.

c. If "Favorite Crossings" is selected, display a list of user-favorited railway crossings.

d. Implement a search function to allow users to search for railway crossings by name.

e. Allow users to mark railway crossings as favorites and display a list of their favorite crossings.

Railway Crossing Status:

a. Display the status of railway crossings.

b. Include information such as the crossing's name, status (open or closed), the person in charge, train schedules, landmark, and address.

c. Provide an option to add a crossing to favorites.

Government Section:

If the user selects "Government," prompt them to log in with a pre-created admin account.

Admin Home Page:

a. Display options: "Search Crossing," "Add Railway Crossing," and a list of existing crossings.

b. Implement search functionality to find specific railway crossings.

c. Provide the option to add new railway crossings to the database.

Manage Crossings:

a. Display a list of existing railway crossings with options to delete or update their status.

b. Allow administrators to update the status of a crossing (open or closed).

c. Handle exceptions for invalid input.

d. Implement a navigation bar for easy access to various options.

Update Crossing Details:

Allow administrators to update details of a railway crossing, including its name, address, landmark, train schedule, person in charge, and status.

Provide a submit option to save the changes.

Delete a Crossing:

Allow administrators to delete a railway crossing.

Return a suitable message upon successful deletion.

User Authentication:

Implement user authentication for public and admin users.

Ensure that only authorized users can access specific sections and perform actions.

Database Interaction:

Use JDBC or Hibernate for database access to perform CRUD operations on railway crossings.

Maintain a "railway_crossings" table in the MySQL database to store crossing information.

Exception Handling:

Implement appropriate exception handling to handle errors and invalid input.