

**COURSE NAME: CAB302**

# **Electronic Billboard Display And Management System**

---

## **Group: 62**

**Fernando Barbosa Silva (n10338471)**

**Sasitheran Vijayasingam (n10308091)**

**Naif Mohammed A Alsherbini (n10391274)**

**Atif Aaiman (n10287469)**

---

---

## **OVERVIEW:**

Our project was to create an electronic billboard display and management system, separated into three software applications, which we successfully implemented:

### **BILLBOARD CONTROL PANEL STATE OF COMPLETENESS-**

<b>Features Completed</b>	<b>Yes</b>	<b>No</b>
<b>Login Validation</b>	✓	
<b>Create Billboards</b>	✓	
<b>Edit Billboards</b>	✓	
<b>Delete Billboards</b>	✓	
<b>Import/Export XML</b>	✓	
<b>Preview Billboard</b>	✓	
<b>List Billboards</b>	✓	
<b>Import Pictures from URL</b>	✓	
<b>Import Pictures from User's computer</b>	✓	
<b>Edit User</b>	✓	
<b>Create User</b>	✓	
<b>Delete User</b>	✓	
<b>Define User Permissions</b>	✓	
<b>List Users</b>	✓	
<b>Create Schedule</b>	✓	

---

<b>Delete Schedule</b>	✓	
<b>View Weekly Schedule</b>	✓	
<b>View All Schedules</b>	✓	
<b>Schedules can be repeated</b>	✓	
<b>Get setup from props file</b>	✓	
<b>Unit Testing</b>	✓	

**Billboard Control Panel** - The control panel is a GUI application, which lets the user interact with the application and do a number of different tasks such as: Creating new billboards, editing/removing existing billboards , scheduling billboards, keeping track of users and manipulating their respective permissions. Once executed, the control panel displays a login screen which prompts the user to enter a username and password. This username and password is cross-checked with the Billboard Server database and depending on the input, the user gets the corresponding screen. The control panel allows users with administrator privileges to create new users, delete users, edit their permissions. Upon creating a new billboard, the user can also preview and check how it would appear on the Billboard Viewer. Users are presented with a calendar view and an option to repeat the chosen billboard when they are in the scheduling sub-menu of the billboard control panel. It is important to note that the control panel does communicate with the Billboard Viewer directly.

---

## BILLBOARD VIEWER STATE OF COMPLETENESS-

Features Completed	Yes	No
Connects to the Server every 15 seconds	✓	
Error message if server not connected	✓	
Get setup from props file	✓	
Show message if there is no schedule available	✓	
Close application if 'Esc' key is pressed or mouse is clicked	✓	
Always fullscreen	✓	
Unit Testing	✓	

**Billboard Viewer** - The billboard viewer is a GUI application as well, and it is used to display the billboard, fullscreened, chosen by a user at the Billboard Control Panel. The accuracy in this application is on how it displays the correct billboards according to the scheduling provided in the control panel. Once started, this application refreshes every 15 seconds and connects to the Billboard Server to retrieve information on the billboard to be currently displayed. The displayed screen does not require any user interaction and simply closes when the 'Esc' key is pressed or the user clicks any of the two buttons on the mouse.

---

## BILLBOARD SERVER STATEMENT OF COMPLETENESS-

Features Completed	Yes	No
Setup from props file	✓	
Check if database exists	✓	
Create tables in database if tables do not exist	✓	
Provide error message if server cannot find/connect the database	✓	
Server waits for connection from Control Panel and Viewer	✓	
Server determines which billboard is to be shown	✓	
Connections will be short-lived, stateless connection	✓	
Server creates token and sends to user	✓	
Server store password in salted	✓	
Server validate user permissions	✓	
Server administrates invalid or expired tokens	✓	

---

Unit Testing	✓	
--------------	---	--

**Billboard Server** - This is a simple command-line program which runs until closed. The server is responsible for holding information of users, schedules and billboards using a database via MariaDB. The server listens to the port connections for the other two applications and retrieves information about the database from a properties file called db.props. The server checks if the database is present when the application is executed, and returns an error if not found. This application is also responsible for creating the required tables when provided with a fresh database. In layman's terms, the Billboard Server acts as a bridge between the Control Panel and the Viewer.

---

## CONTRIBUTIONS:

**Important Note:** The GitHub repository provided with the submission was created a few days prior to the submission and therefore mimics the commit history and messages of the original repository. This was done due to a fatal error caused by a team member, rendering the original repository useless and dysfunctional. Apologies for the inconvenience.

**GitHub:** [https://github.com/atifaaiman/CAB302\\_Main\\_Project\\_062](https://github.com/atifaaiman/CAB302_Main_Project_062)

Name	Student Number	Server Database	Server Connection	Control Panel	Viewer	Report	Test
<b>Atif Aaiman</b>	n10287469			✓	✓	✓	✓
<b>Naif Mohammed A Alsherbini</b>	n10391274	✓	✓	✓	✓		
<b>Sasitheran Vijayasingam</b>	n10308091			✓	✓	✓	✓
<b>Fernando Barbosa Silva</b>	n10338471	✓	✓	✓	✓		✓

---

# **CLASS EXPLANATION:**

## **Common Classes:**

Some common classes are used across all three applications, arranged under the “common” packages of each application. They are:

**Billboard:** This class encapsulates billboards to be stored into the database and transferred through the network between server and clients.

**Message:** This is not a class, but rather an interface which defines a behaviour of transferable objects through the network. Contains only constants and methods to get required data.

**MessageBuilder:** Builds objects to be transferred over the network.

**TestMessageBuilder:** Tests the message builder class using JUnit 5 for the unit-test development

**Permission:** Contains only constants to define available permissions for users.

**Schedule:** Encapsulates schedule objects to be transferred over the network and to be stored in the database.

**User:** Encapsulates user objects to be transferred over the network and to be stored into the database.

**Hash:** This class is exclusively found under the common package of Billboard Server. Its function is to hash the password.



---

## **Billboard Control Panel:**

**LoginPanel:** Draws all the components for the login panel. This class is void of listeners as that part is mainly done in the Controller class.

**BillboardsPanel:** Draws all the components for the Billboard panel where users with permissions are able to view, delete, edit or add Billboards.

**SchedulesPanel:** Draws all the components for the schedule billboard screen, such as adding buttons, layouts etc. This class also uses the DatePicker class to draw the calendar in the add schedules sub-menu.

**UsersPanel:** Draws all the components inside the display users panel. Contains a table listing the username, administrator privileges, and permissions. Updates the user table when it is modified.

**GUI:** The main GUI class which encapsulates the user interface for the control panel. This class calls the other panel classes to generate the corresponding interface.

**Client:** Connects to the server and returns to the server listener, displaying an error if the connection was not established.

**Controller:** Accepts all user actions such as clicks on a button or select images. The controller is also responsible for updating the GUI when a command is input.

**InputCommandHandler:** Encapsulates handler for all input objects from the server. Uses the Observer interface to update it, once new input is received from the Message (common) interface. For the case of InputCommandHandler, the Observer is the Controller class.

**OutputCommandHandler:** Used to communicate with the server, mainly sending requests such as login, logout, add user, add schedule, delete user, delete billboards, etc.

---

**DatePicker:** Used to create the calendar in Schedule billboards sub-menu.

**GUIPreview:** Used to generate the preview in the create billboards sub-menu. This class is used to parse xml and updates the billboard using an input document. Scales message, picture and information attributes according to the rules of their presence.

**Main:** The main method which executes and runs the Billboard Control Panel. Checks if the server is connected, then proceeds and initiates the GUI and Controller.

**Observable:** An interface used to update the observer object once it is changed. Contains only one method. In this project, the observer is the Controller class, while the observable is the InputCommandHandler class.

**ReadPropsFile:** Reads the network props file provided and sets the host and port variables.

**ServerListener:** Listens to responses from the server. Uses two handlers to handle incoming messages (InputCommandHandler) and outgoing messages (OutputCommandHandler).

---

## **Billboard Server:**

**ClientListener:** Listens to the client requests. Executes a command every time the client inputs a 'message' from the Message Interface.

**DB:** Encapsulates the MariaDB connection with the database. Initiates the connection and creates tables if the database does not contain them. In respect to all functions carried out in the GUI, this class updates it in the database, such as the creation or deletion of a billboard, getting the permission for a user, getting and adding the salt, etc.

**DBCheckSchema:** Verifies if the database exists after reading the properties file called 'db.props'.

**DBPropsFileRead:** Reads the db.props file and puts the values in variables that can be used by the application, such as the url, schema, username and password.

**Main:** The main method which runs the server.

**Server:** Encapsulates server to respond on all requests from clients and starts the server on port.

**TestClientListener:** Tests the client class using JUnit 5 for the unit-test development.

**TestServer:** Tests the server class using JUnit 5 for the unit-test development.

---

## **Billboard Viewer:**

**GUI:** Draws the interface for the viewer. Scales message, picture and information attributes according to the rules of their presence. Always appears full screened, updates the billboard every 15 seconds.

**Main:** The main method which runs the viewer. Checks if a billboard is scheduled at current time and then displays if yes.

**ReadPropsFile:** Reads the network props file provided and sets the host and port variables.

**ServerListener:** Listens to the server to update with the database inorder to display the correct billboard.

---

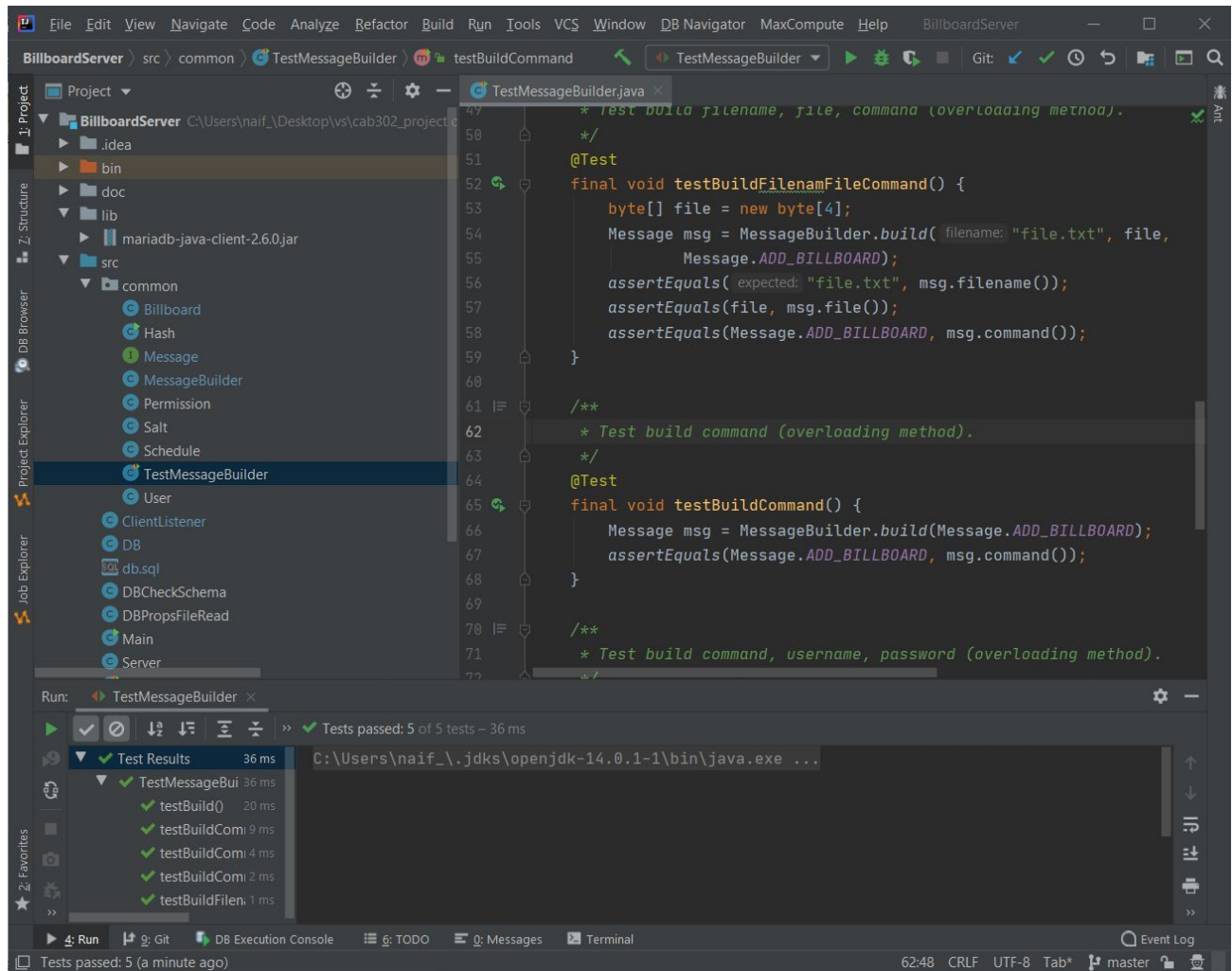
# **TEST-DRIVEN DEVELOPMENT:**

The main aim of creating unit testing is to help produce a high-quality codebase as much as possible. The test classes are named using the convention of Test where the tests are named the same as the classes they are testing. All unit testing was done by using JUnit 5 Jupiter and no other external libraries were used. Moreover, unit tests are all capable of being run without requiring any external resources such as the server or the database. It should support the program with the following tests:

- Test Message Builder
- Test Client Listener
- Test Server

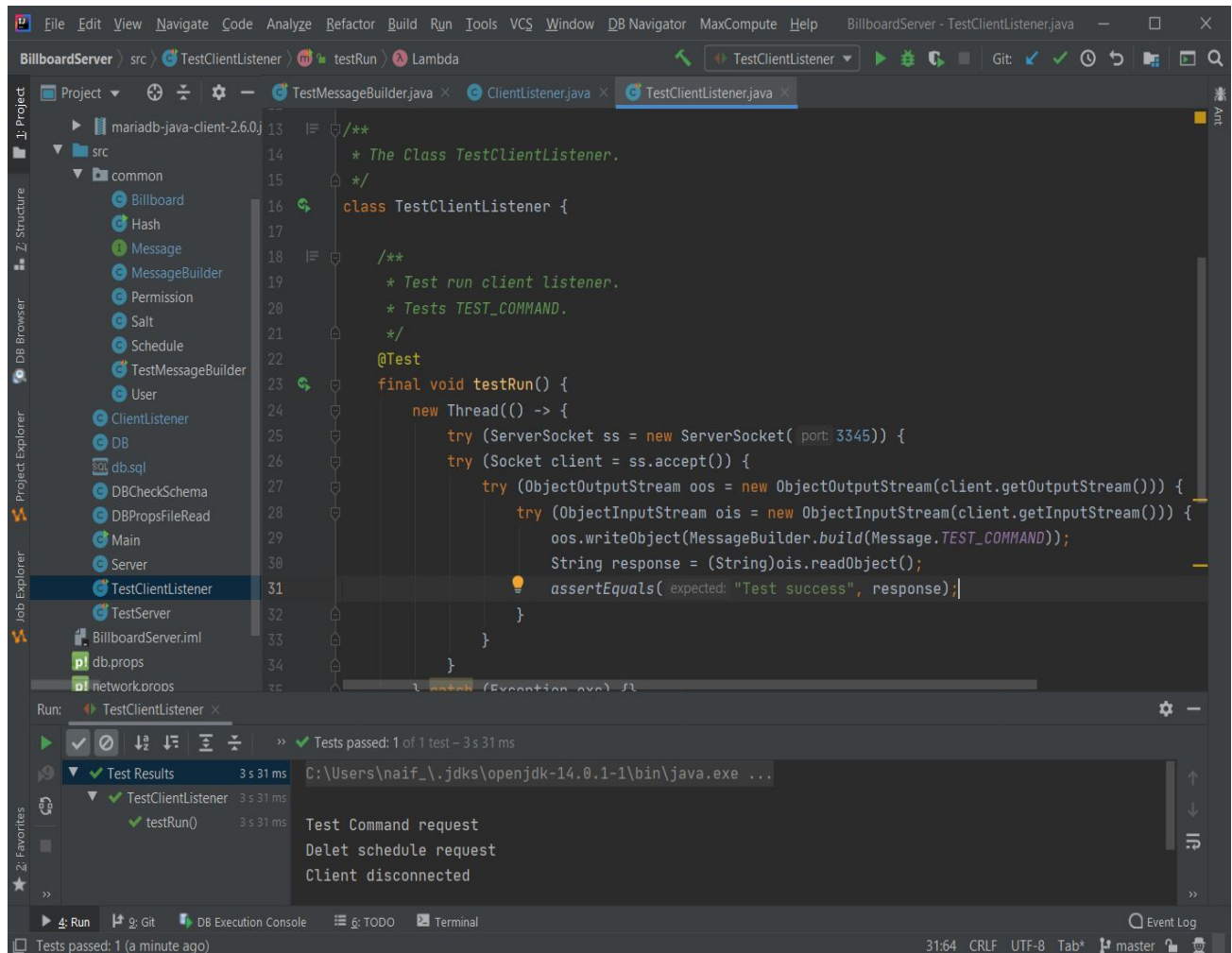
The next 3 pages will explain briefly each test, and which method they are testing with screenshots to show each test result.

## 1- Test Message Builder



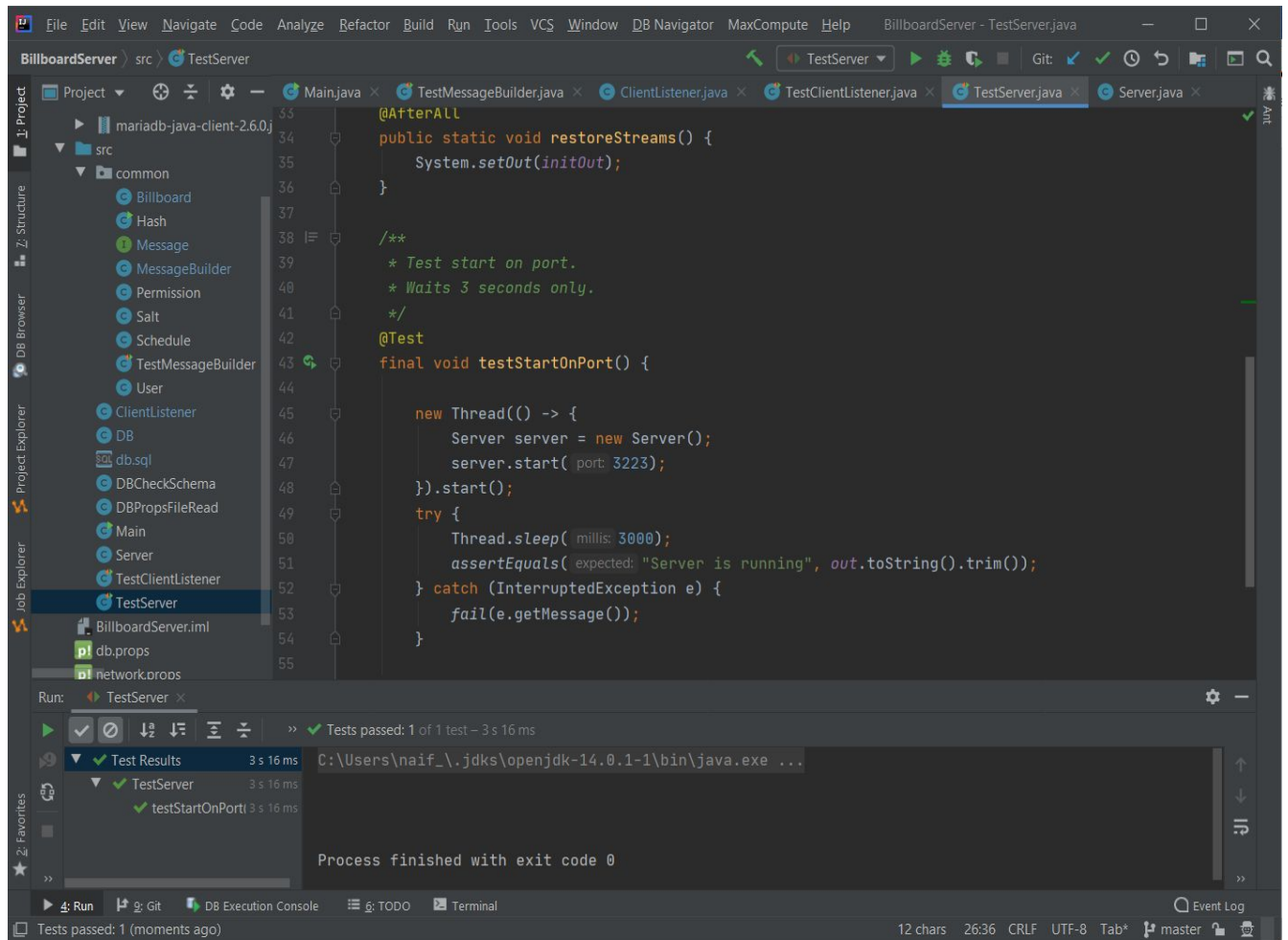
Test Message Builder class is testing the communications between server and clients that are performed through the Message Builder in the common package for the three programs Billboard Control Panel, Billboard Server, and Billboard Viewer. It tests in particular build commands that include build command (overloading method), Test build filename, file, command (overloading method), Test build command, token, and Test build full method.

## 2- Test Client Listener



Test Client Listener class will test Client Listener in the Billboard Server program where this client will listen to the two other clients, in this case, the viewer and the control panel. The test is testing one important method which is the run () for the client listener that performed Test command listener in a separate thread with a fake port to see if the response is successful.

### 3- Test Server



Test Server will test the server to set the up streams then restore streams. Also, to start on a fake port, and waits 3 seconds only to simulate the update time that is used in the server.



---

# **SETUP INSTRUCTIONS:**

1. Install and run MariaDB.
2. Choose the database you want to use with MariaDB.
3. Open IntelliJ and clone the repository or use the files from the project zip folder.
4. Now, open three separate IntelliJ windows, one containing only the Billboard Server files, one with Billboard Viewer and one with the Billboard Control Panel.
5. Now, go to the db.props file in Billboard Server program and beside 'schema', enter the name of your database.
6. If your database contains a username and password, write them beside each corresponding field. If not, just leave the username as it is and remove the password.
7. For the url, insert your local ip instead of the given ip.
8. Now, right-click on the root folder (Billboard Server) and add a new props file. Name it network, so that it reads 'network.props'.
9. Add the following lines to this file:

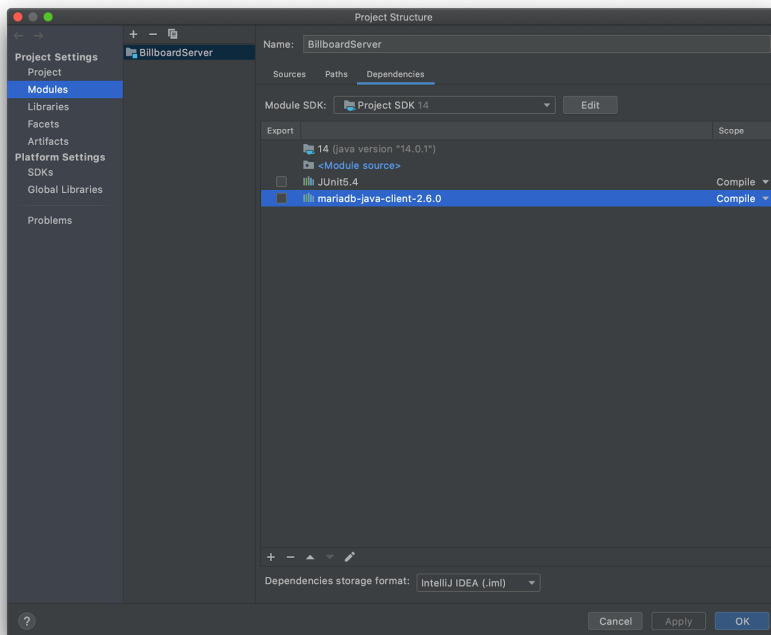
host=localhost //pointing for the server

port=3333 //desired port

10. Add this network.props file for the other two applications.
11. Run the Billboard Server via the Main class.
12. Go to the Billboard Control Panel, and run the program via the Main class.
13. Use default username: admin and password: admin
14. Be sure to create a schema and put the schema name in the db.probs file.

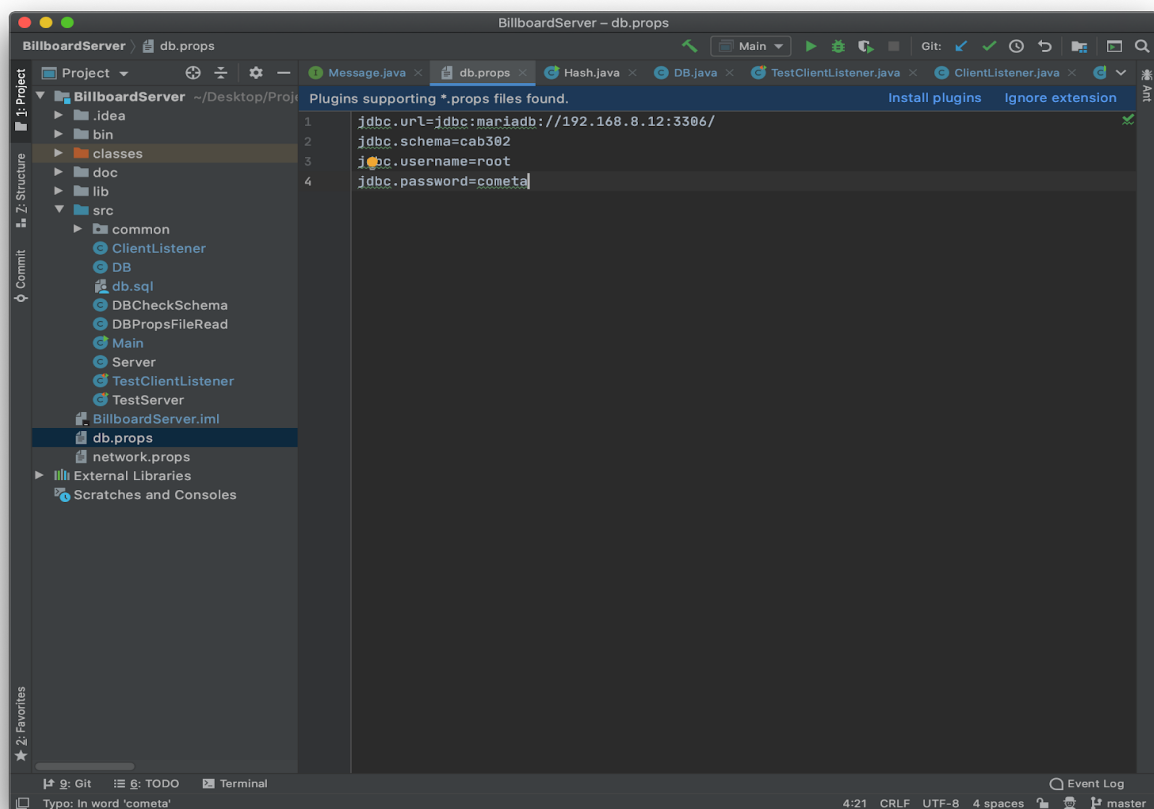
---

## Adding libraries:



Adding all the libraries that are necessary to make the program run.

**db.probs file:**



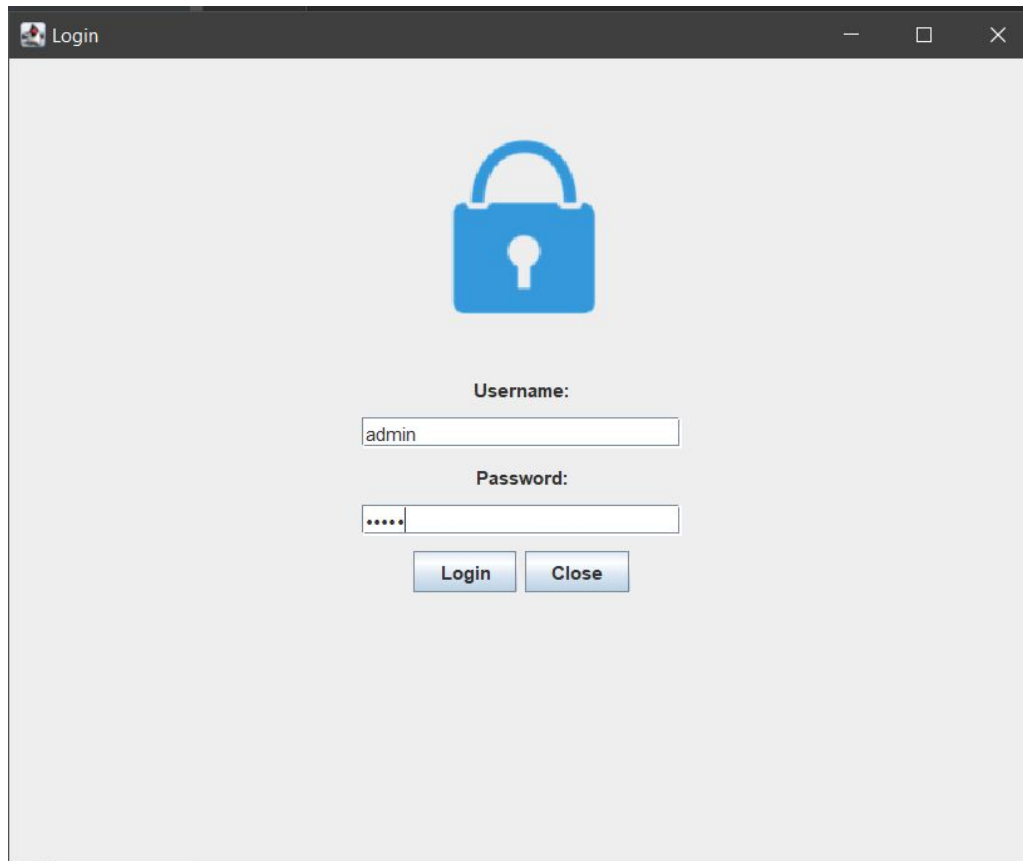
**Make sure to add the necessary information for the database setup such as the example below:**

1. `jdbc.url=jdbc:mariadb://localhost:3306/`
2. `jdbc.schema=cab302`
3. `jdbc.username=root`
4. `jdbc.password=`

---

## **WALKTHROUGH:**

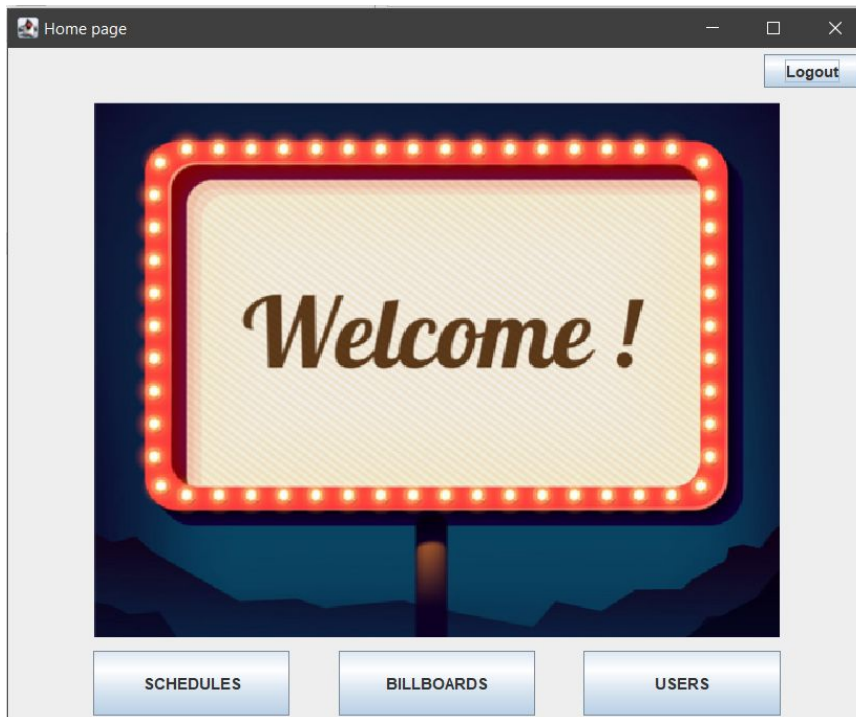
**First step login page:**



To start using the program, when new users open the Billboard control panel for the first time, they have to login as an admin user with the (username: admin and password: admin). As an admin user, you have full permission.

---

## Second step Home page:

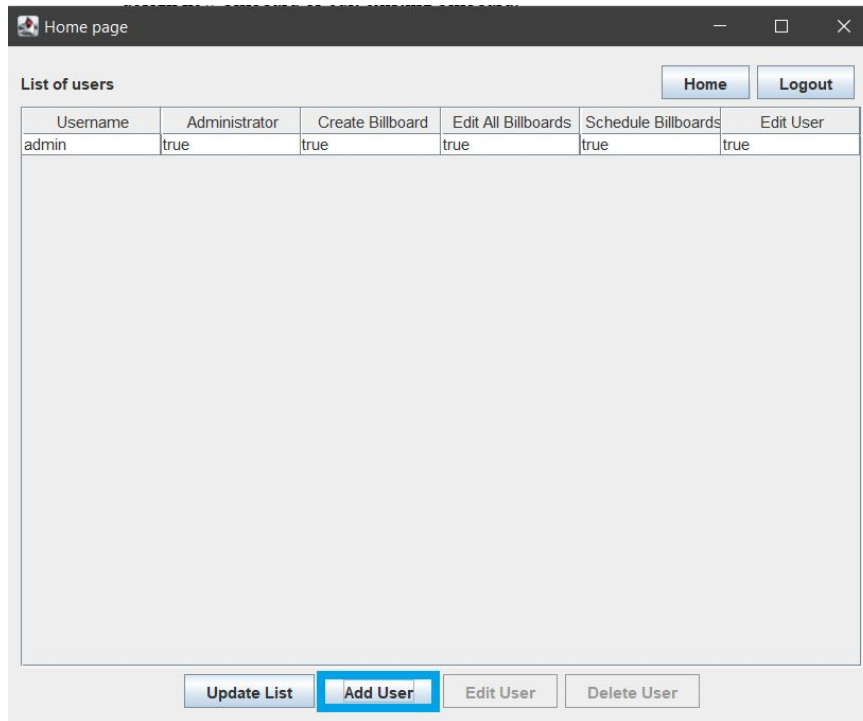


On the home page, the user is greeted with a welcome billboard and they have three main options.

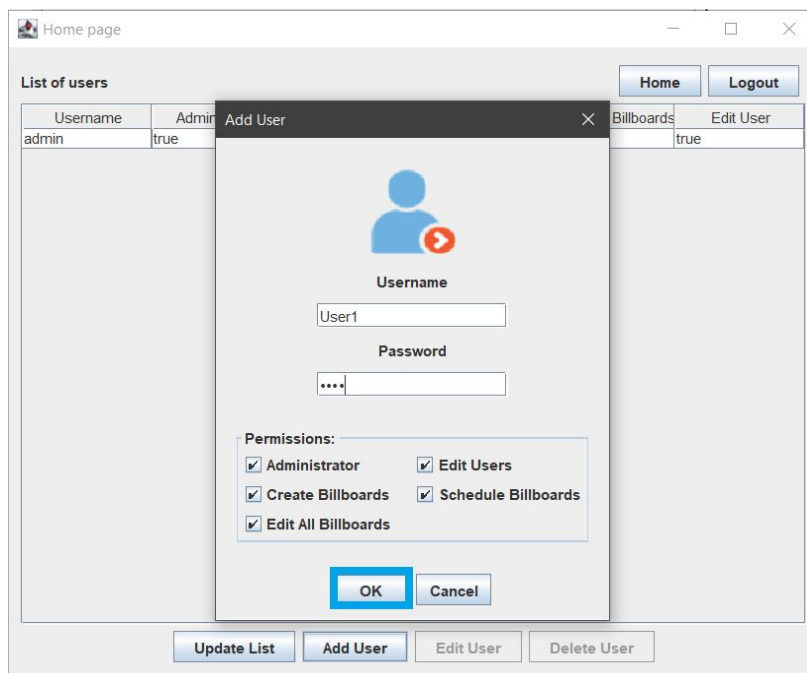
- 1- Users: which allow admin and other users with edit user permission to create a user or edit any existing users.
- 2- Billboards: will allow users with permission to create and edit billboards, they can design new billboards or edit existing billboards.
- 3- Schedules: users with permission schedule billboards will be able to have a look at the weekly schedule for the billboard, also they can schedule a new time for the billboard they want.

The walkthrough will describe each one of the options starting with creating a user then creating a billboard and last scheduling the billboard in detail in the next section.

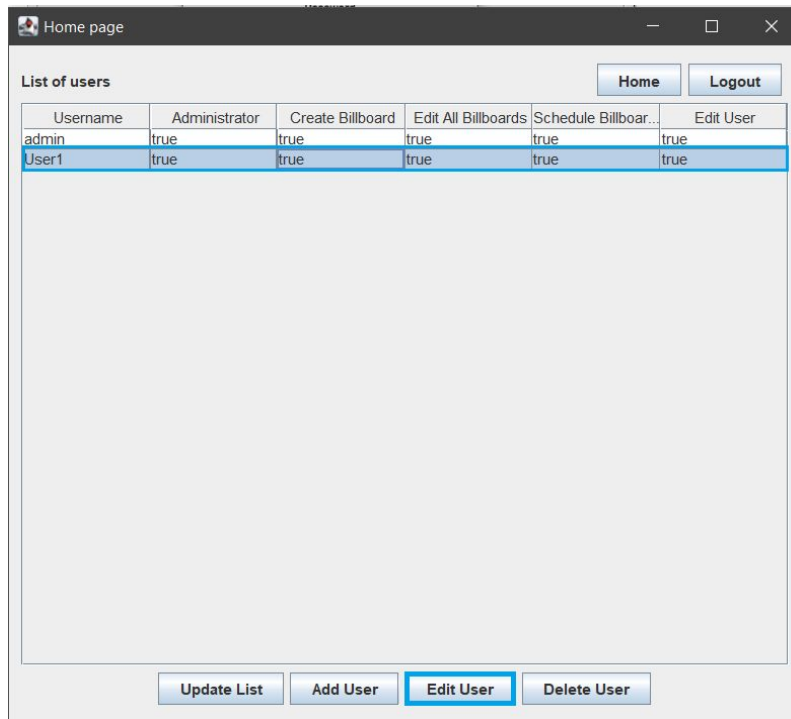
### Third step User page.



First, to create a new user click the button “Add User”, then a small screen will appear. You can see it in the picture below.

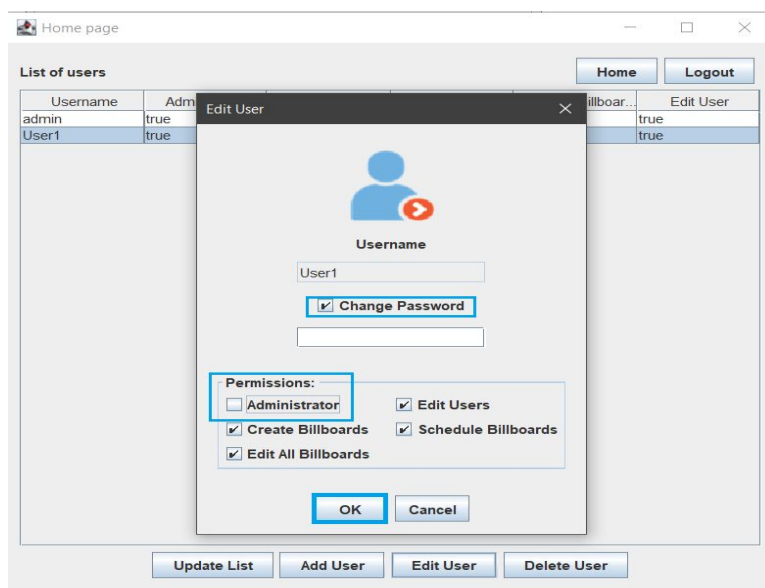


## Editing existing users:



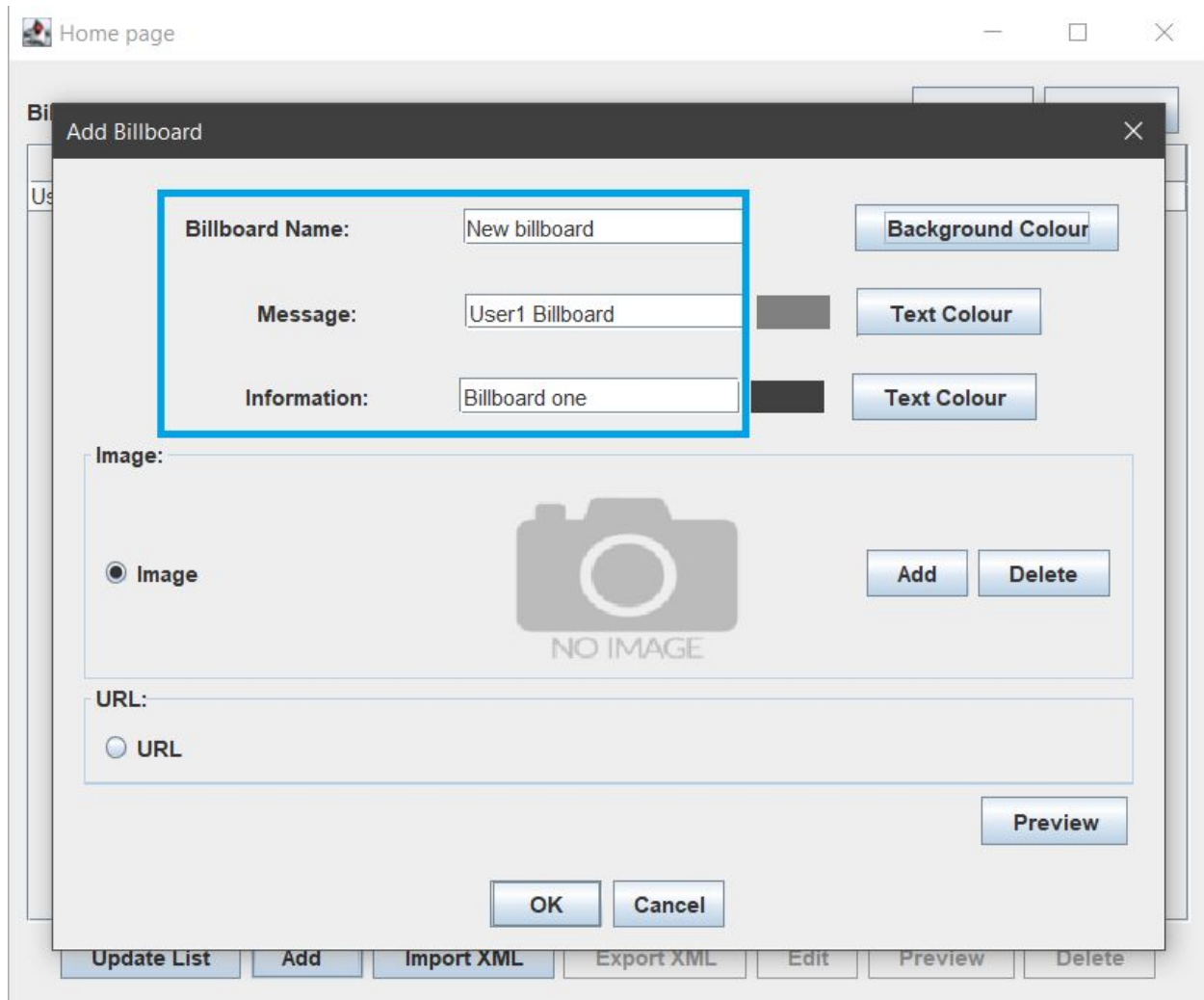
To edit any user, it is required to click on the row that contains the user that needs to be edited, then choose the “Edit User” option at the bottom of the window.

(Edit User window will appear)



fourth step Billboard page.

### Adding Billboard:



The screenshot shows a software application window titled 'Home page' with a standard Windows-style title bar (minimize, maximize, close buttons). A modal dialog box titled 'Add Billboard' is open in the center. The dialog has a dark grey header bar with a close button (X) on the right. The main content area is light grey and contains several input fields and buttons. A blue rectangular box highlights the top three text input fields: 'Billboard Name:' (containing 'New billboard'), 'Message:' (containing 'User1 Billboard'), and 'Information:' (containing 'Billboard one'). To the right of these fields are three color selection buttons: 'Background Colour' (with a grey swatch), 'Text Colour' (with a dark grey swatch), and another 'Text Colour' button (with a black swatch). Below the highlighted fields is an 'Image:' section with a radio button labeled 'Image' (which is selected) and a large placeholder area showing a camera icon and the text 'NO IMAGE'. To the right of this area are 'Add' and 'Delete' buttons. Below the 'Image' section is a 'URL:' section with a radio button labeled 'URL' (which is not selected). At the bottom right of the dialog is a 'Preview' button. At the bottom center are 'OK' and 'Cancel' buttons. The background of the application window shows a toolbar with buttons: 'Update List', 'Add', 'Import XML', 'Export XML', 'Edit', 'Preview', and 'Delete'.

There are three main text fields in the add billboard window that will take all the necessary text information from the user to create the new billboard.

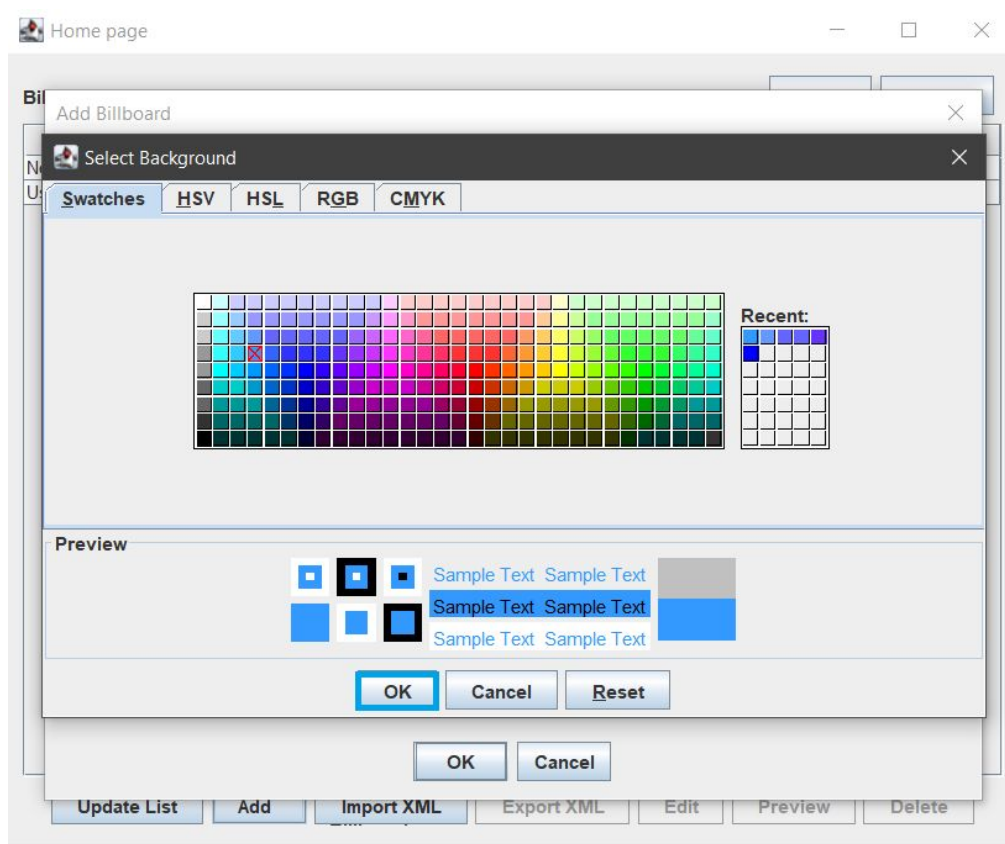


(Colouring the text or the background).

The screenshot shows a 'Home page' window with a sidebar containing 'Bi', 'Ne', and 'Us'. The main area displays the 'Add Billboard' dialog box. The dialog has a title bar with a close button. It contains the following fields and controls:

- Billboard Name:** A text input field containing 'New Billboard'.
- Message:** A text input field containing 'User1'.
- Information:** A text input field containing 'test'.
- Image:** A section with a radio button labeled 'Image' (which is selected). Below it is a large placeholder area with a camera icon and the text 'NO IMAGE'. To the right of this area are 'Add' and 'Delete' buttons.
- URL:** A section with a radio button labeled 'URL' (which is not selected).
- Color Selection:** On the right side of the dialog, there are three color selection controls, each with a small color swatch and a button:
  - The top control has a white swatch and a button labeled 'Background Colour'.
  - The middle control has a grey swatch and a button labeled 'Text Colour'.
  - The bottom control has a black swatch and a button labeled 'Text Colour'.
- Buttons:** At the bottom of the dialog are 'OK' and 'Cancel' buttons. At the bottom of the main window are 'Update List', 'Add', 'Import XML', 'Export XML', 'Edit', 'Preview', and 'Delete' buttons.

(Select color window)



After selecting one of the main JButton colors, a window for selecting the color will provide the user with a variety color model such as Swatches, RBG, And HSV.

Adding picture:

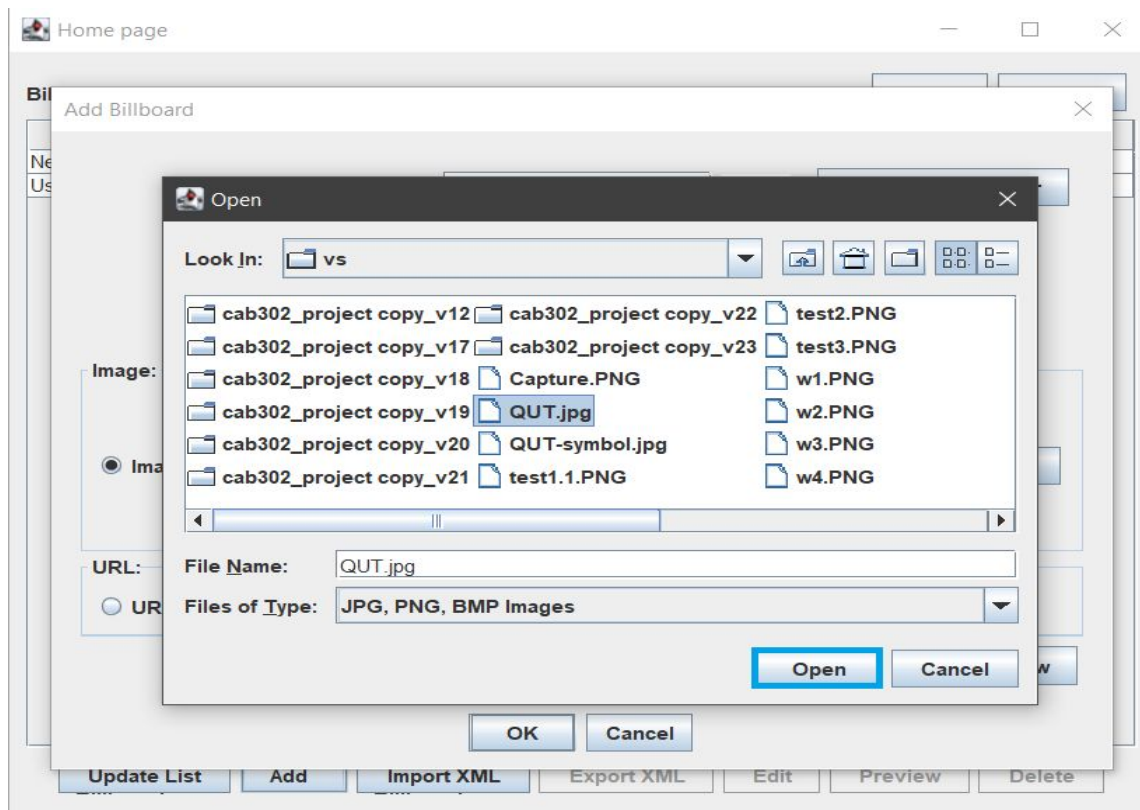
The screenshot shows a web application interface with a 'Home page' window in the background. Overlaid on this is a dialog box titled 'Add Billboard'. The dialog contains the following elements:

- Billboard Name:** A text input field containing 'New billboard'.
- Message:** A text input field containing 'New.User'.
- Information:** A text input field containing 'test'.
- Background Colour:** A color selection button.
- Text Colour:** Two buttons for selecting text color, one for the message and one for the information.
- Image:** A section highlighted with a blue box. It contains a camera icon, the text 'NO IMAGE', and 'Add' and 'Delete' buttons. A radio button labeled 'Image' is selected.
- URL:** A section highlighted with a blue box. It contains a radio button labeled 'URL'.
- Preview:** A button located at the bottom right of the dialog.
- OK/Cancel:** Buttons at the bottom center of the dialog.

The background window shows a toolbar with buttons: 'Update List', 'Add', 'Import XML', 'Export XML', 'Edit', 'Preview', and 'Delete'.

For adding pictures there are two options, first adding pictures, from the user device by clicking the add button in the middle of the add billboard window.

(window for choosing the picture from the user device)



This window will allow the user to navigate through their machine to select the picture they wish, with any type of the following JPG, PNG, BMP.

Second option is adding image by (URL)

The screenshot shows a web application interface. A 'Home page' window is open in the background. In the foreground, an 'Add Billboard' dialog box is displayed. The dialog contains the following elements:

- Billboard Name:** A text input field containing 'New billboard'.
- Message:** A text input field containing 'New.User'.
- Information:** A text input field containing 'test'.
- Background Colour:** A color picker button.
- Text Colour:** Two color picker buttons.
- Image:** A section with a radio button labeled 'Image' and two buttons labeled 'Change' and 'Delete'.
- URL:** A section with a radio button labeled 'URL' and a text input field. This section is highlighted with a blue border.
- Preview:** A button located at the bottom right of the dialog.
- OK/Cancel:** Two buttons at the bottom center of the dialog.

The background window shows a toolbar with buttons: 'Update List', 'Add', 'Import XML', 'Export XML', 'Edit', 'Preview', and 'Delete'.

Import XML:

The approach of Importing XML is the same approach of adding an image form the user device.

## Fifth step Schedule page.

(add schedule)

The screenshot shows the 'Add Schedule' dialog box with a 'Select Date' calendar. The calendar is for June 2020, with columns for Sun, Mon, Tue, Wed, Thur, Fri, and Sat. The 'Add Schedule' dialog box has the following fields:

- Billboard Name: New billboard
- Once: Start date (01/06/2020), Finish date
- Time start: 20 hours, 29 minutes
- Time finish: 20 hours, 29 minutes
- Recur: No Recur, Every Day, Every Hour (selected), Every Minutes
- Recur every: 02 minutes
- Duration: 05 minutes
- OK, Cancel buttons

A blue arrow points from the 'Start date' field to the 'Select Date' calendar.

After choosing the add option on the schedule main page a small window will pop up, for the user to specify all the information for scheduling the new billboard.

The screenshot shows the 'Home page' with a 'Calendar of billboards scheduled' and a 'Schedule list' table. The calendar is for June 2020, with columns for Thursday, Friday, Saturday, Sunday, Monday, Tuesday, and Wednesday. The 'Schedule list' table has the following data:

Schedule id	Billboard name	Start date-time	Finish date-time	Creator	Create date
1	User example	04/06/2020 21:15:0	04/06/2020 23:54:	admin	2020-06-04 20:56:

A blue arrow points from the 'Week schedules' button to the 'Schedule list' table.

Users can see all the information about the billboards that have been scheduled in the main schedule page in the table. Also, they can view all the billboards that are scheduled for the current week by clicking the "Week Schedules" button.

---

**Last step running the Billboard viewer:**



When running the billboard viewer at the time the billboard is scheduled, the user will be able to view the billboard in full-screen size. Users can close the viewer screen by either the echapp button or any mouse click.

**(billboard if nothing is scheduled)**

---

# No billboard scheduled

Have a good day.

---

(billboard if the server is not running)

---

# Server not found

Please contact the system administrator.