CAB302 Assessment Report

Electronic Billboard Display and Management System

Contributors (Group 53):
Mitchell Crompton - n9948015 (Github: @Mitch2099)
Tom Hunter - n10235736 (Github: @tomhuntx)
Hussein Al Alwan - n9470964 (Github: Hussein-95)
Jordon Khiosudta - n9481346 (Github:@jordon-khiosudta)

Date of Submission: **5/06/2020**

Table of Contents

Statement of Completions and Contributions **3**

Design Description **4**

Billboard Viewer 4

DisplayBillboard: 4

Display: 4

BillboardContents: 4

ViewerTest: 5

Billboard Control Panel 6

High-Level Design Description 6

The GUI 7

Billboard Server **8**

RunServer 8

The Billboard Control Panel Manual 9

Create Billboards 11

List Billboards 12

Schedule Billboards 14

Edit Users 16

Billboard Viewer Walkthrough 19

Test-Driven Development **19**

Test-Driven Development and the Control Panel 19

Source Control and Documentation **20**

Statement of Completions and Contributions

Task	Contributor(s)	Description
Billboard Viewer		
Viewer handler	Mitchell Crompton	Display class
Billboard Contents	Mitchell Crompton	GUI class
Event handling	Hussein Al Alwan	Specified exit events
Timing	Hussein Al Alwan	Added timed billboard check
Label design	Mitchell Crompton	Designed appropriate labels
XML parsing	Mitchell Crompton	Parsing xml as readable attributes
Label placement	Mitchell Crompton	Placed labels accordingly
Control Panel		
Control Panel Login	Tom Hunter	Login window and authentication
Control Panel Hub	Tom Hunter	Hub window for control panel (including logout and change passwords)
Create Billboards	Tom Hunter	Tool for creating and editing billboards
List Billboards	Tom Hunter	Tool for listing current billboards
Schedule Billboards	Tom Hunter	Tool for scheduling billboards
Edit Users	Tom Hunter	Tool for creating and editing users
Control Panel Users	Tom Hunter	User functionality (username, passwords, permissions)
Control Panel Tests	Tom Hunter	A series of JUnit tests that confirm the core functionality of the control panel.
Billboard Server		
Database	Jordon Khiosudta	A MariaDB database used to store a collection of tables that hold the xml Data

Design Description

Billboard Viewer

The Viewer package is a program which will create a GUI which cycles through billboards based upon files it collects from the server program. At the current stage of development the program is unable to connect to the server, so in place of that it performs its main function using the given example billboards as String data types. It contains three classes, DisplayBillboard, Display, and BillboardContents.

DisplayBillboard

This is the main class of the Viewer program. It is responsible for initialising the Display class. The main method calls the method createAndShowGUI method which instantiates the display variable as an object of the Display class, and then performs a for loop which performs the Display methods runDisplay and delayDisplay, using the DELAY constant as input.

Display

This class is meant to connect to the server and create a GUI based on a string of XML. The timesServerConnection method is designed to perform the connectToServer method every 15 seconds. The connectToServer method is designed to create a connection to the Server program, and would then extract the contents of a new billboard though this functionality was not implemented. The data connectivity portion of this class is incomplete, so it is simulated using the newXML method. The newXML method returns a random string of XML code from the example billboards given with this assignment. The runDisplay method sets the billboardContents variable to a new object of the BillboardContents class then runs the showGUI method on that variable and is called first by the DisplayBillboard createAndShowGUI method. The second method called by the createAndShowGUI method is the delayDisplay method which takes an integer delay variable as input then tries to sleep the current thread for that delay in milliseconds, before performing the resetGUI method on the billboardContents variable, catching an interrupted exception event.

BillboardContents

This class is an extension of the JFrame class which creates a graphical user interface. The interface is a fullscreen window with only JLabel attributes, which exits upon clicking or pressing the escape key. These JLabel attributes are determined by the input parameter, a string variable describing an XML file, and can be any combination of a message label, picture label, and information label. The class instantiates by initializing a set of private variables, the displayPanel JPanel, all the JLabels it contains, and other variables for use in other methods, and the frame itself.
The showGUI method is the method called on a new BillboardContents object within the Display class runDisplay method, and it calls the exitEvents, updateBillboard, and

labelPlacement private methods before adding the displayPanel JPanel to the frame and showing the GUI. The resetGUI method only closes the GUI so as to be called independently of showing the GUI, after the delay of showing each billboard. The exitEvents method adds listeners to the frame to exit the billboard upon either a click or press of the escape key.
The updateBillboard method tries to parse the string of XML code to get every individual attribute as a Node object then perform the private updateAttributes method for each, and also sets the displayPanel background colour if specified. The updateAttributes method is called for each individual attribute of the billboard, whether it be a message, picture, or information, and updates global variables and objects accordingly. The labelPlacement method places whichever labels exist in their correct placements, including label colours, font sizes, and image sizes. The fontScale method returns an appropriate font size for whichever JLabel is input.

ViewerTest

This class implements JUnit unit testing to test both the Display and the BillboardContent classes functionality. For the Display class it tests the runDisplay and delayDisplay methods, and for the BillboardContent class it tests creating billboards with message, information, and picture attributes, individually and altogether.

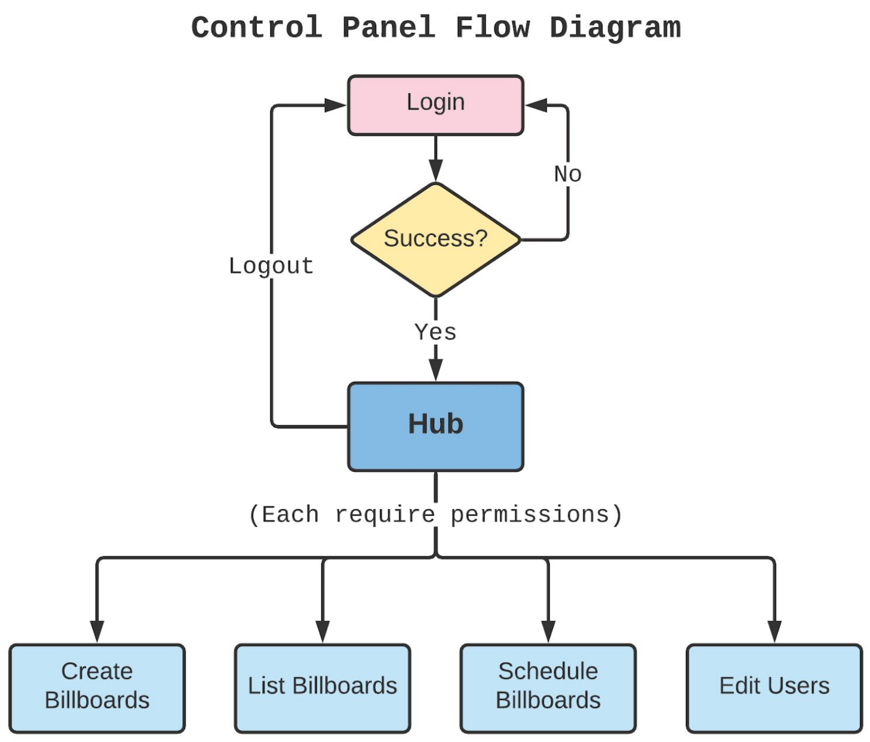
Billboard Control Panel

High-Level Design Description

The **Billboard Control Panel** serves a variety of purposes; most notably the ability to create, list, and schedule billboards, and to edit and manage users. The control panel is a GUI application that is made up of a variety of classes within the "(...)ControlPanel" package. This package and its components were created by Tom Hunter.

The ControlPanelManager class is the starting point of the program that creates the UI, connects to the database, and then runs the login GUI via the ControlPanelUI class. The ControlPanelUI class covers the vast majority of the GUI functionality (over 1000 lines of code), that extends JFrame and displays and processes the currently running UI. Another major class is User which handles the creation and management of any user accounts of the application. There is also the BillboardListTable which handles the JTable for the List Billboards section of the control panel and its intractable buttons. Similarly, the OpenXML class is used to open .xml files for the Create Billboards section. The package also contains the ControlPanelTest class that tests the login and user functionality of the application via a series of JUnit test cases.

Designing the control panel primarily involved designing the layout of each UI element. These elements were sketched on paper to include each core component and how their functionality would be achieved (such as a scrollable table using a JTable and a JScrollPane). The overall flow of the control panel was also designed beforehand, represented by the following flow diagram:



This diagram represents the key design decision to make the control panel two separate windows: the **login window** and the **control panel window**. Upon successfully logging in via a known username and password, the application disposes of the login window and creates the control panel window, bringing the user to the control panel hub.

The GUI

The control panel hub is a swing-based GUI that uses the ControlPanelUI class to switch the window between a variety of frames. Its UI is created and managed in the ControlPanelHub() method. Users can also change their passwords and logout back to the login window. Screenshots of the Control Panel's components and more in-depth descriptions of their outputs can be found in the Manual. In this section, the functionality of these section's methods will be explained.

The control panel window is only disposed of if the user quits or logs out, where the main panel of the window is changed based on an accompanying method. Buttons such as the "returnHub" buttons in each window outside of the hub initiate these transitions by removing the current container frame from the content pane (the window), to then run a method to create and set a new main container for the GUI, followed by revalidating the window. This design decision allows users to smoothly transition between the components of the control panel. These components are contained within four core sections: Create Billboards, List Billboards, Schedule Billboards, and Edit Users. Each section sets the container to a new tool that provides specific functionality that is handled by the section's method and accompanying buttons (such as the CreateBillboards() method for the Create Billboards section). These sections also each require a unique permission to access, constructed and handled by the User class.

The Create Billboards section provides an interactive tool used to create billboards, with fully-functional importing .xml abilities, changing the background colour, font size, and other adjustments. The Schedule Billboards section presents a weekly calendar with date, time, duration, and repetition type inputs. This allows users to schedule a billboard with this information. The List Billboards section shows a table of every known billboard with intractable buttons to be used to preview, edit, or delete billboards. These buttons are handled in the separate class BillboardListTable as they are quite complex. Finally, the Edit Users section allows users to create new users with a given username and password. This section does not currently have functionality to edit current users as it lacks server connectivity.

These sections and their accompanying methods simply create the GUIs that the users can interact with, where the information processing is generally handled by each window's JButtons. These buttons are created globally below the method that they are used in, to then be formatted and displayed by these methods. This button processing includes GUI transitions such as the "returnHub" button to return to the hub or the "scheduleConfirm" button used to confirm billboard scheduling that processes the given information involved in billboard scheduling. These buttons also require much exception handling to prevent unprocessable input, such as a user attempting to schedule a billboard in the past.

Billboard Server

RunServer

The RunServer Class is used to handle communication between the Billboard Viewer and the Billboard Control Panel to the Database. Upon starting, the server will connect to the database and check if the necessary tables are present. If the tables are not present, they will be generated and ready to store data in. The Server was then meant to create a Socket in which it would wait for communication from either the Control Panel or the Viewer and would confirm communication and send over the requested information, however this was not completely implemented.

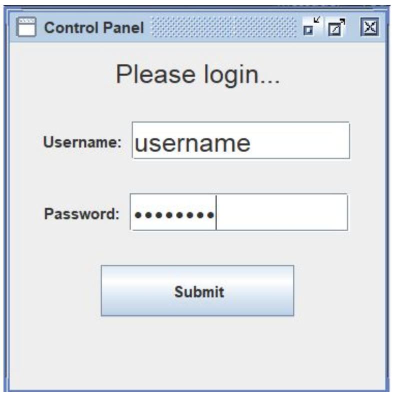
xml2String((File xmlFile) is a method that was designed to Object of type File and convert it into a String. This was to be implemented by the control panel after successfully creating a billboard in order to be sent across the network socket to the Server, such that the method addBillboard(PreparedStatement insertStatement, String xmlString, String FileName) could correctly store the byte data in the database.

retrieveBillboard(Statement queryStatement, String fileName) was capable of returning the desired billboard labeled fileName from the database as an Object and was intended to be able to send the byte data of the billboard over to the viewer.

The Billboard Control Panel Manual

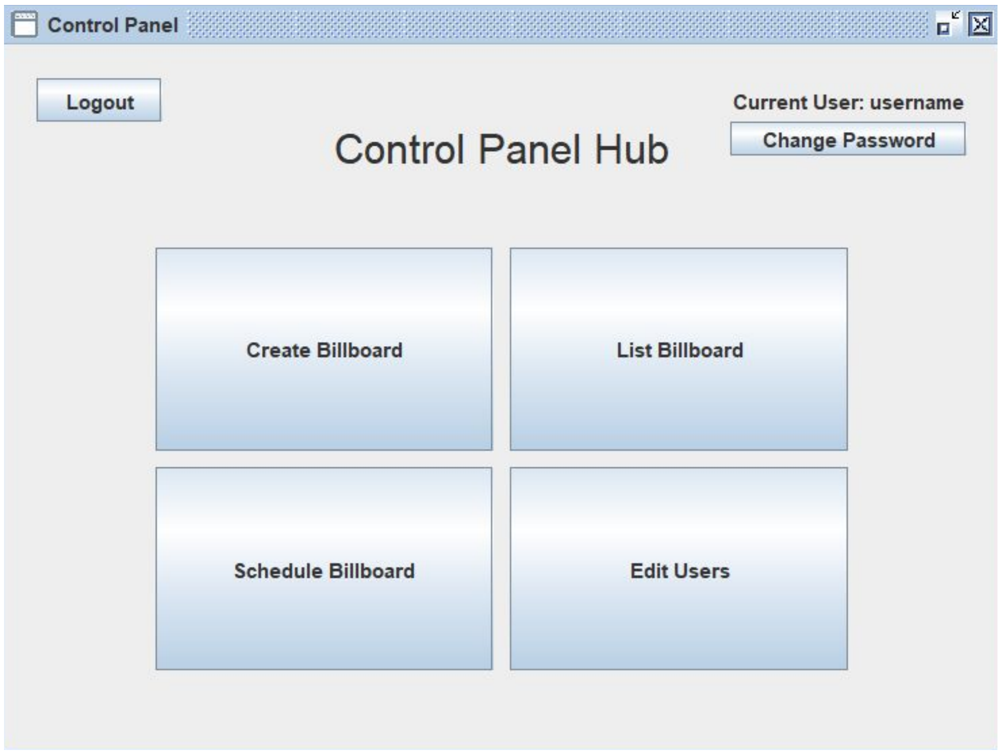
The Billboard Control panel is a GUI application responsible for managing billboards and users. The Control panel currently lacks the connectivity functionality to the other two core sections (Billboard Server and Viewer), yet its own core components are functional. The control panel was created, in its entirety, by Tom Hunter.

The Control Panel begins with a login popup, asking the user to enter their username and password. If these details match that of a user in the database, they successfully login and are sent to the Control Panel Hub.



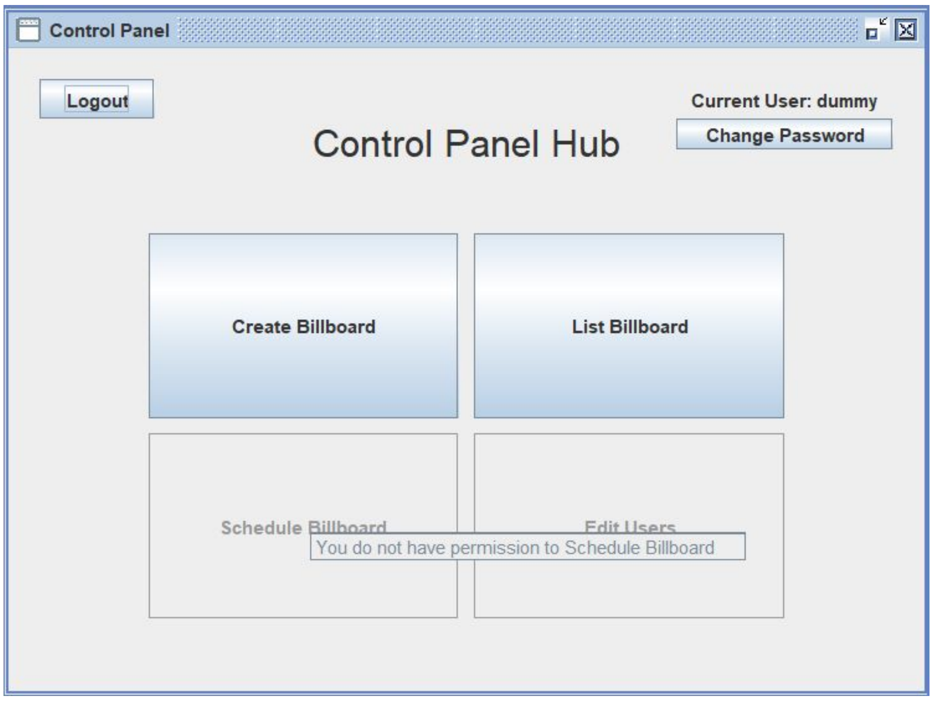
It is from this hub that the user can access the four main features of the control panel: Creating Billboards, Listing Billboards, Scheduling Billboards, and Editing Users. Each of these sections have a unique permission that the user’s account must hold for them to be accessed. In this hub, any user can also logout or change the password of their account.

Admin user “**username**” and password “**password**” with full access:



-All users are stored in a local list, where new users can be added, removed and changed locally, and tested by logging into these new accounts. New users can be created in the “Edit Users” section that will be covered later.

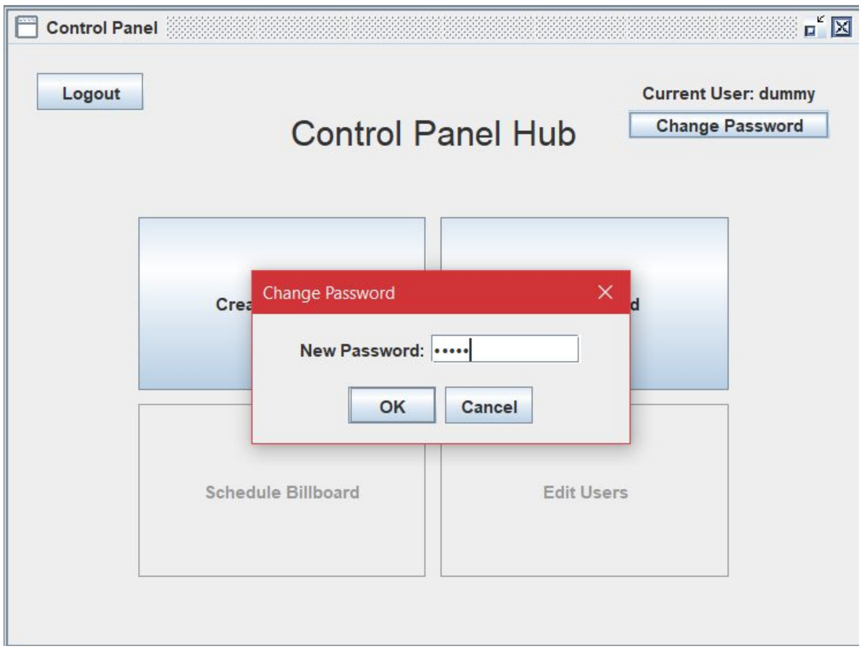
Control Panel Hub appearance for user “**dummy**” with password “**12345**” that only has permissions to create and list billboards:



You do not have permission to Schedule Billboard

-Hovering over the un-permitted buttons has the above tooltip.
-Logging out sends users back to the login window.

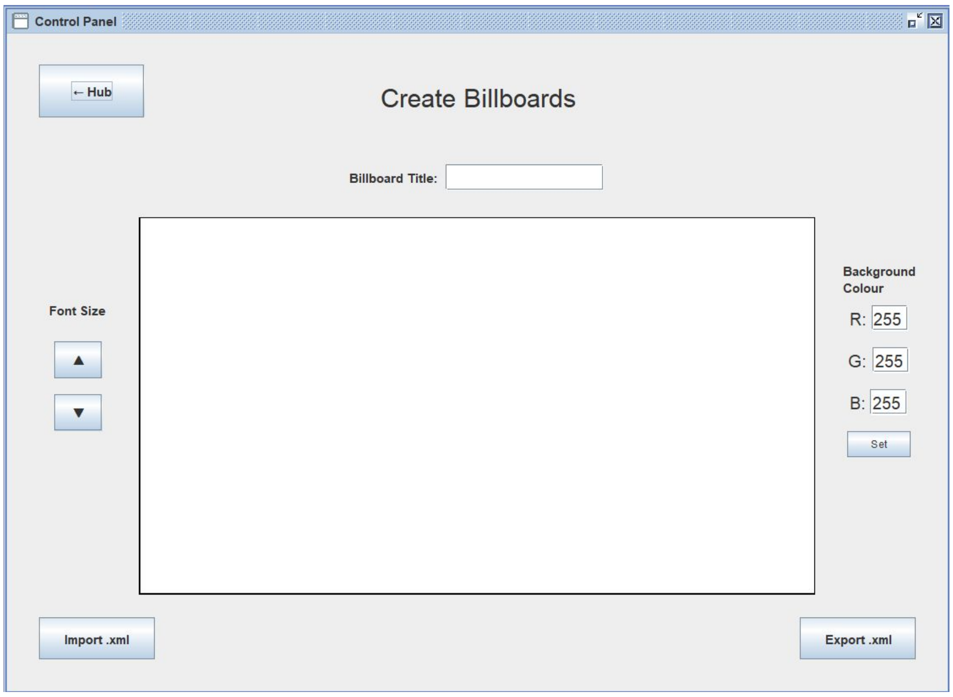
Pressing the “Change Password” button creates the following popup:



-Users that change their password can logout and then log back in only with their new password.
-Pressing the main hub buttons such as “Create Billboard” will bring the user to the corresponding UI section.
-All sections have a “Hub” button in the top-left that return the user to the hub.

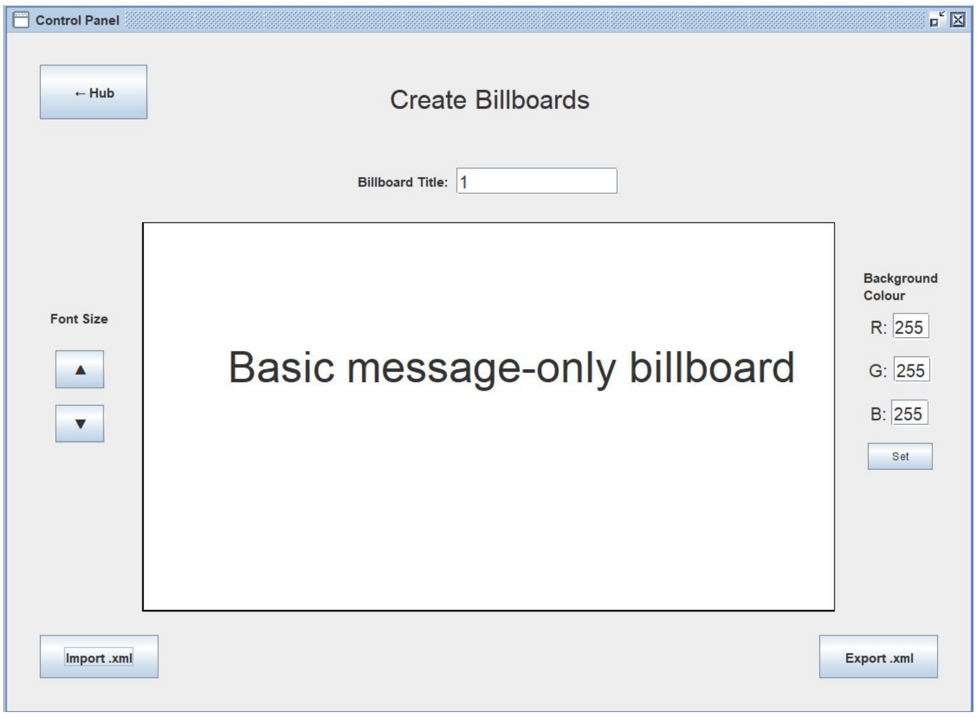
Create Billboards

The Create Billboards section of the control panel requires the “Create Billboards” permission and presents the following billboard creation tool:

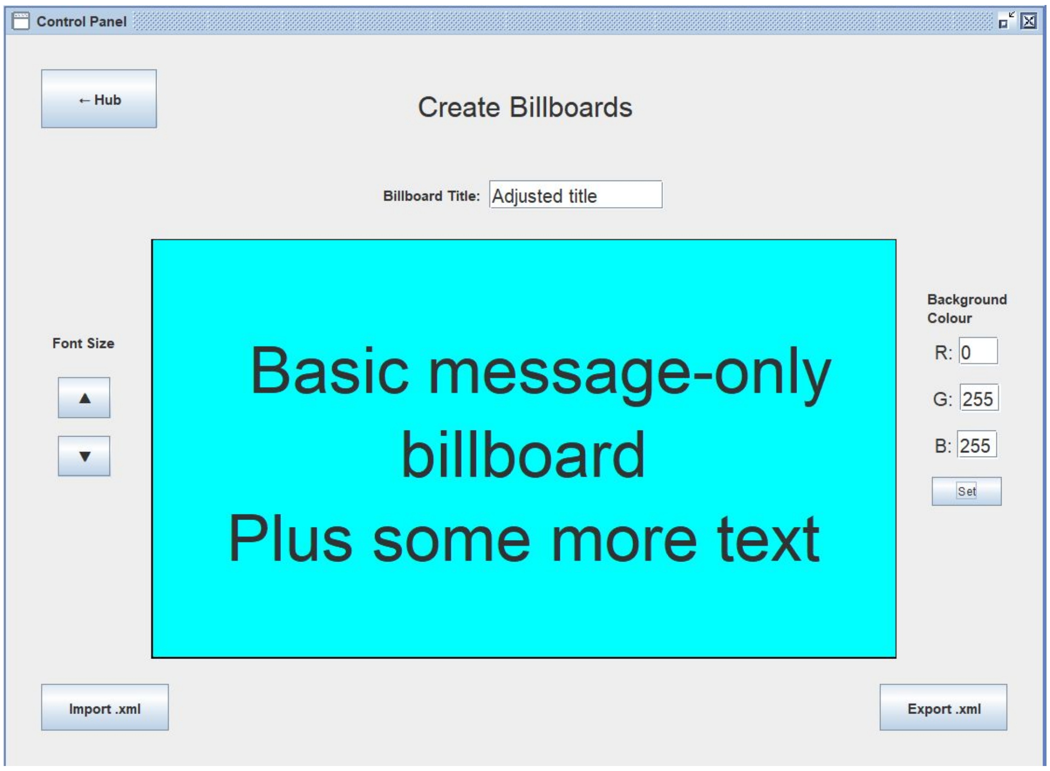


-Users can change the title of the billboard, write text on it, change the font size, and change the background colour.

Users can also import .xml files into the editor, with the “Import .xml” button. This brings up a file search window that only accepts .xml files. Pressing this button and selecting the example 1.xml file produces the following output:



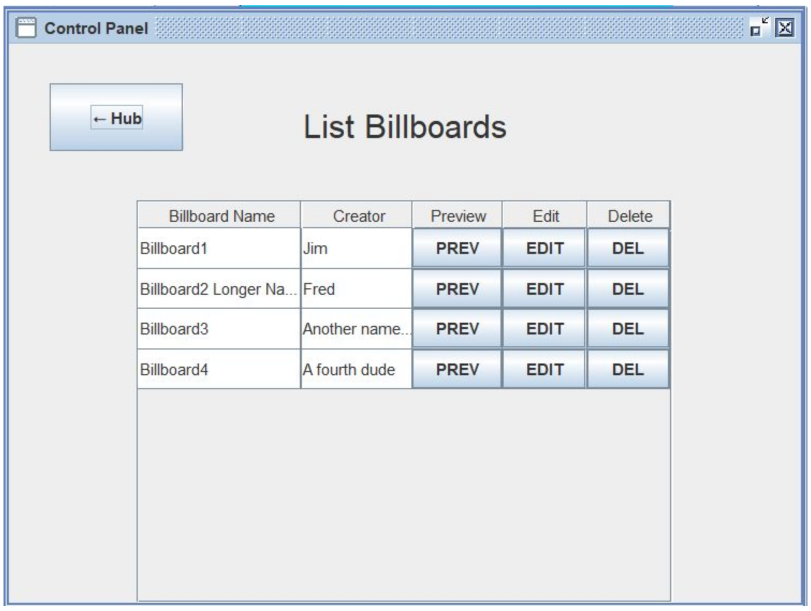
Adjusting this 1.xml billboard by changing the title, increasing the font size, and setting the background colour to 0, 255, 255 produces the following output:



-This background colour is stored in a hexadecimal format (#00ffff in the above case) yet there is currently no full functionality to export this file as an .xml.

List Billboards

The List Billboards section of the control panel requires the “List Billboards” permission and presents the following table:

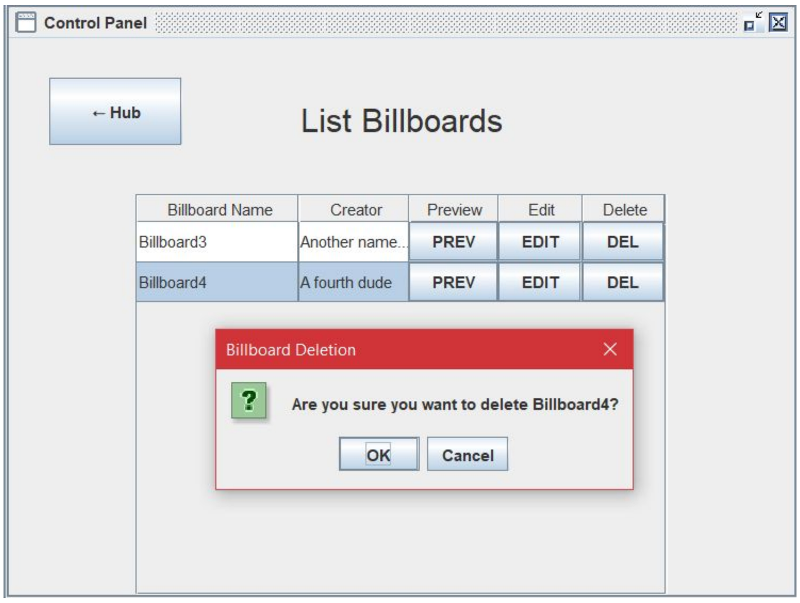


Billboard Name	Creator	Preview	Edit	Delete
Billboard1	Jim	PREV	EDIT	DEL
Billboard2 Longer Na...	Fred	PREV	EDIT	DEL
Billboard3	Another name...	PREV	EDIT	DEL
Billboard4	A fourth dude	PREV	EDIT	DEL

-This table lacks billboard server connections, but contains four local dummy billboards, each with a unique name, creator name, and buttons.

The PREV, EDIT, and DEL buttons allow users to preview, edit, or delete each billboard. As this lacks connectivity to the server, the preview and edit buttons currently simply have a popup to prove functionality, and the delete buttons delete the selected row. This deletion currently only occurs locally, as the billboards do not remain deleted if the page is refreshed, but these features show that the functionality simply lacks server interaction.

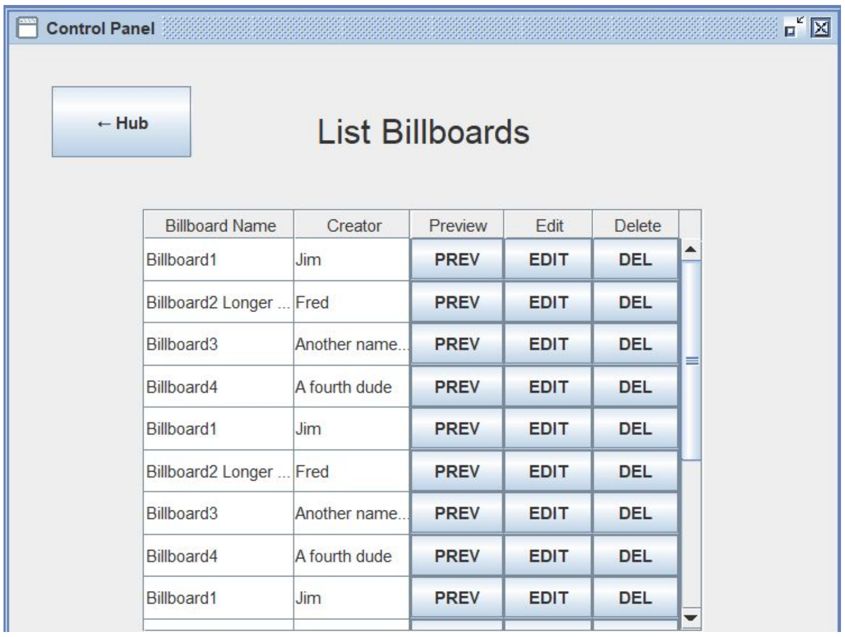
Deleting the first two rows from the abovand then selecting the DEL button to delete the “Billboard4” row produces the following output:



Billboard Name	Creator	Preview	Edit	Delete
Billboard3	Another name...	PREV	EDIT	DEL
Billboard4	A fourth dude	PREV	EDIT	DEL

Are you sure you want to delete Billboard4?

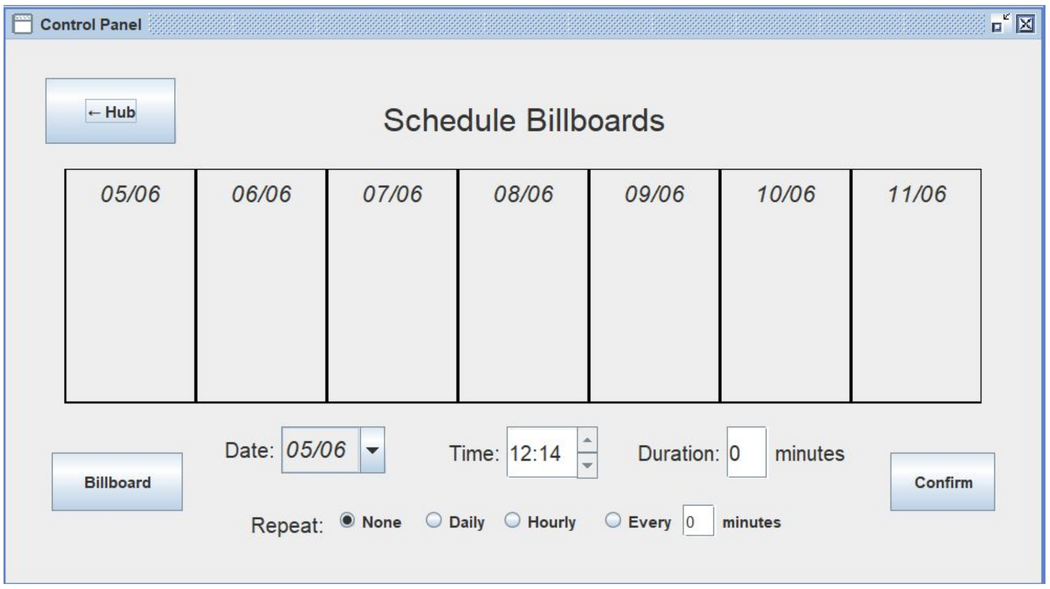
If this table becomes filled with enough billboards, a scrollbar is revealed allowing users to access all the information in the list. Duplicating the above four billboards a few times allows us to produce the following result to preview this:



Billboard Name	Creator	Preview	Edit	Delete
Billboard1	Jim	PREV	EDIT	DEL
Billboard2 Longer ...	Fred	PREV	EDIT	DEL
Billboard3	Another name...	PREV	EDIT	DEL
Billboard4	A fourth dude	PREV	EDIT	DEL
Billboard1	Jim	PREV	EDIT	DEL
Billboard2 Longer ...	Fred	PREV	EDIT	DEL
Billboard3	Another name...	PREV	EDIT	DEL
Billboard4	A fourth dude	PREV	EDIT	DEL
Billboard1	Jim	PREV	EDIT	DEL

Schedule Billboards

The List Billboards section of the control panel requires the “List Billboards” permission and presents the following scheduling tool:



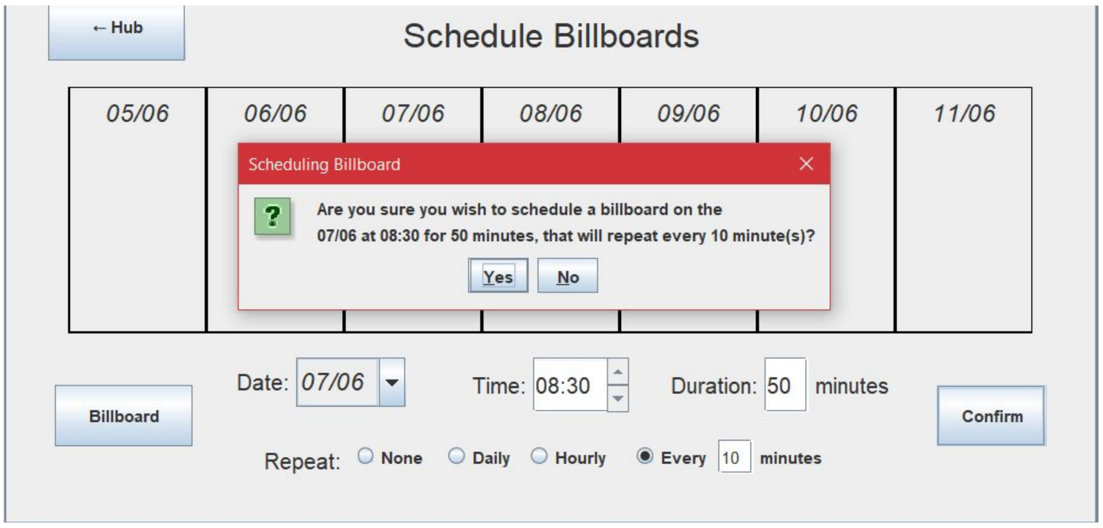
Date: 05/06 Time: 12:14 Duration: 0 minutes

Repeat: [✓ None] Daily Hourly Every 0 minutes

This tool shows all the current billboards that are scheduled at any time and allows users to schedule a billboard with a specific date, time of day, duration, and repeat type. The tool sets the starting values and the weekly calendar to the user’s current date and time.

The date option is a combo box with options for selecting any day of the current week (all are shown in the weekly calendar), the time input is a 24 hour clock with hours and minutes processed as such, the duration is a textbox that only accepts numbers, and the repeat line is a series of radio buttons, where only one can be selected and processed at a time.

For example, scheduling a billboard for the 07/06th at 8:30am for a duration of 50 minutes, that will repeat every 10 minutes produces the following output:



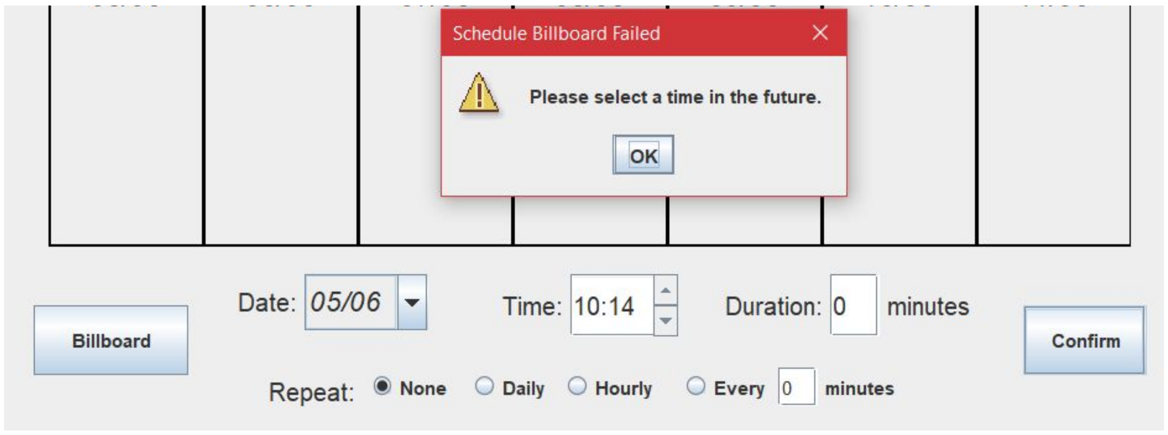
Are you sure you wish to schedule a billboard on the 07/06 at 08:30 for 50 minutes, that will repeat every 10 minute(s)?

Date: 07/06 Time: 08:30 Duration: 50 minutes

Repeat: None Daily Hourly [✓ Every] 10 minutes

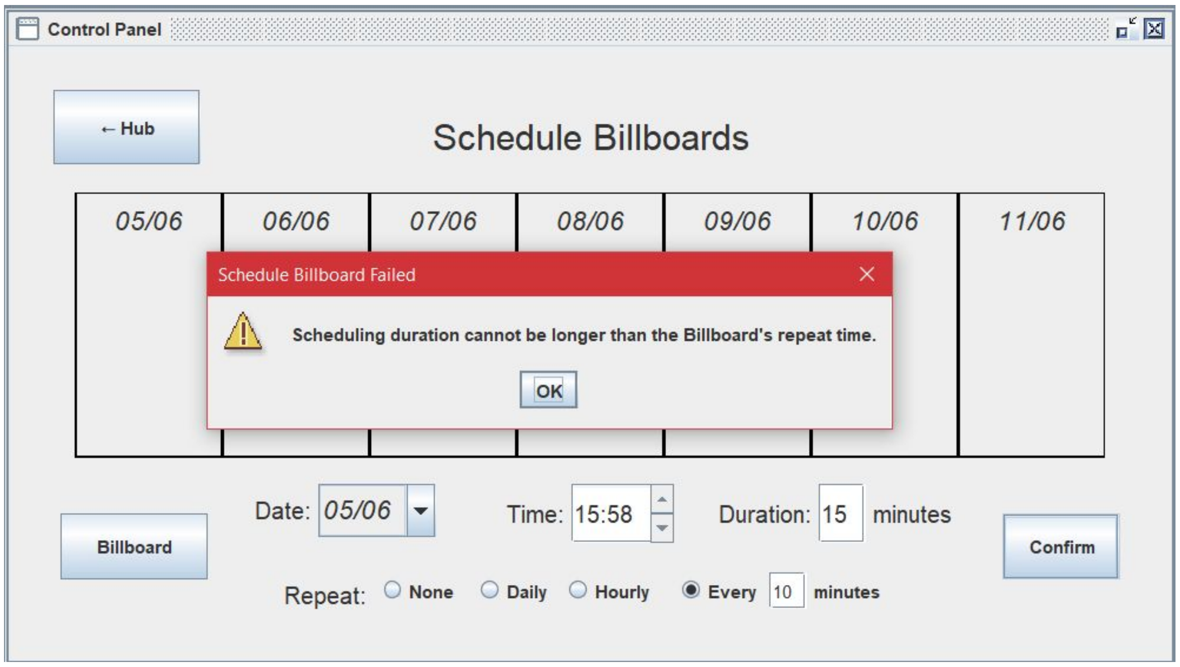
-No billboard is truly synced and scheduled to the server, as the server lacks core functionality and thus I have made it produce this popup to show that all the required information is there, it is just unprocessable without a fully-functional server.

There is also plentiful exception handling with the scheduler, where no text boxes accept negative, non-numerical, or otherwise unprocessable input. For example, the duration box must be above zero, and the selected scheduling time must be in the present or the future. Each of these exception handlers have a unique popup, such as the following being a result of trying to schedule a billboard in the past:


Schedule Billboard Failed

Please select a time in the future.

Another key exception that is handled is preventing users from attempting to schedule a billboard to repeat at a time that is less than its duration - as this case would cause multiple billboards to schedule at once or otherwise cause other errors. Attempting this produces the warning prompt:


Schedule Billboard Failed

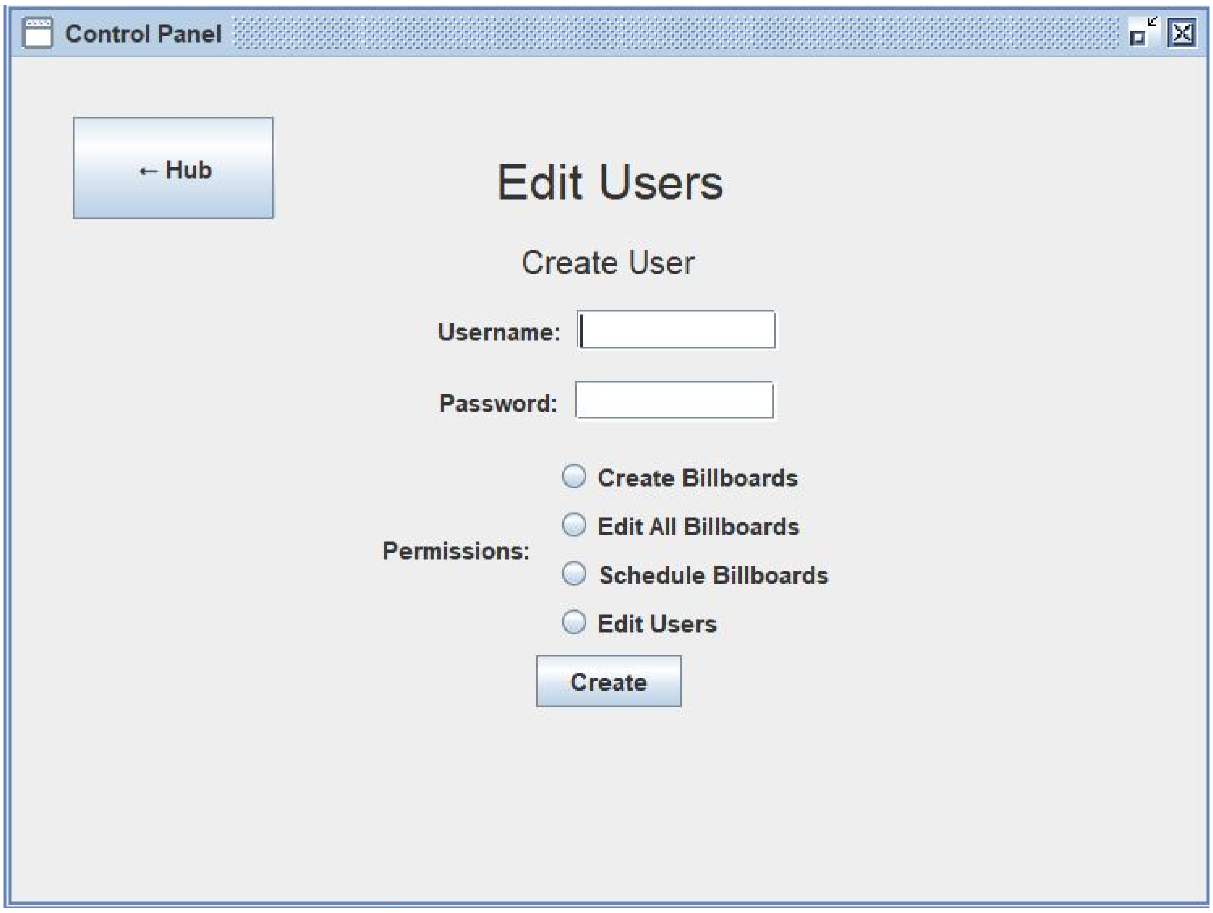
Scheduling duration cannot be longer than the Billboard's repeat time.

Edit Users

The Edit Users section of the control panel requires the “Edit Users” permission which permits the user to create new users, and edit the details of existing users. This permission would generally be reserved for administrators only.

This section currently does not have functionality to edit existing users, yet I imagine this would involve a JTable similar to that of the Billboard List that lists all existing users, and allows for adjustments. This section does, however, currently allow for the creation of new users that are added to a local list of users.

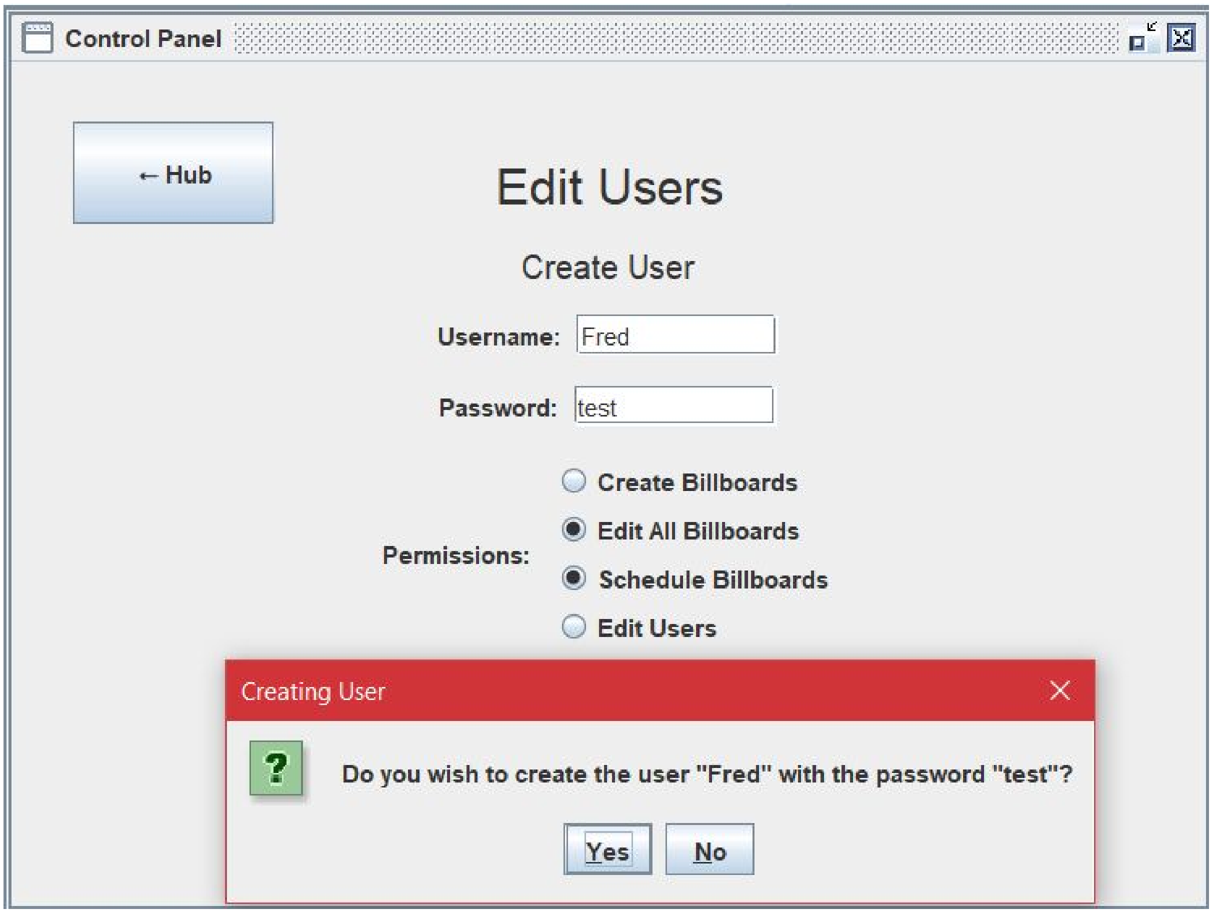
The Edit Users panel currently looks as follows:



Administrators can create new users by setting the user’s username, password, and permissions by selecting one or more permission radio buttons.

After creating a new user, the current user can logout and then login to the new user’s account using the set username and password, which shows that full user functionality is there. This functionality simply lacks connectivity to the server.

As a test, let's create the user "Fred" with the password "test" and the permissions "Edit All Billboards" and "Schedule Billboards." We do this by entering the following into the edit users section:


Control Panel

← Hub

Edit Users

Create User

Username: Fred

Password: test

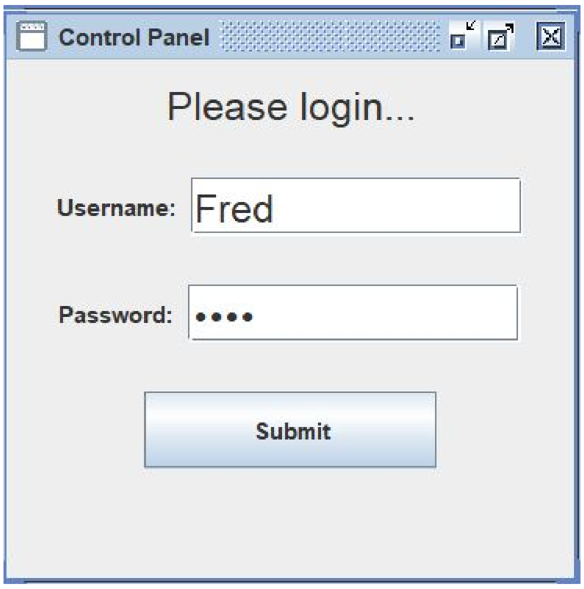
Permissions:
- ○ Create Billboards
- ● Edit All Billboards
- ● Schedule Billboards
- ○ Edit Users

Creating User

Do you wish to create the user "Fred" with the password "test"?

Yes No

Then, we can test if this worked by returning to the hub and logging out of the admin user. We can then login to Fred's new account by entering his details into the login panel:


Control Panel

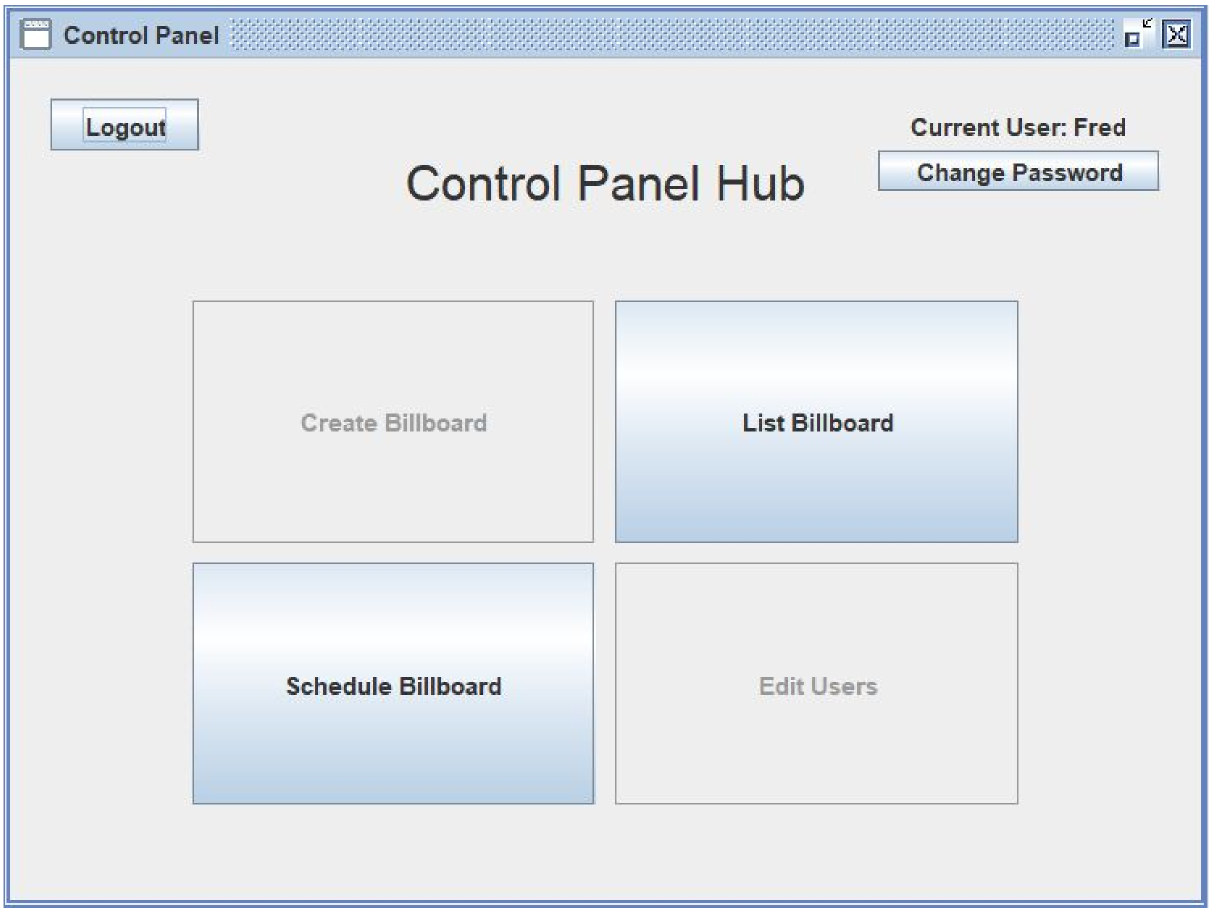
Please login...

Username: Fred

Password: ••••

Submit

Pressing submit then leads us to the control panel's hub, which looks like this for Fred:



Creating users also has much exception handling, using similar popups as the other sections, where new users and their passwords cannot be blank, the username must be less than 15 characters, and the password must be at least 4 characters in length.

Billboard Viewer Walkthrough

The viewer application is responsible for displaying the billboard in a full screen GUI.

To test the viewer simply run the main function in the DisplayBillboard class

Exit functionality
It has a timer to connect to the server.
Sending requests and receiving billboards has not been implemented.

Currently it is not retrieving billboards from the server but displays a random billboard already provided to it before running.

For more details please review the design description section.

Test-Driven Development

Test-Driven Development and the Control Panel

Test Driven Development, in short, is development that begins with tests of expected results that fail, for code to be implemented that follows these tests in order to make these tests succeed. As the Control Panel was a GUI-based application, using developing it using TDD was challenging but quite helpful. Small-scale and essential components of the control panel, such as the login panel, benefited from TDD, where more intricate components such as each section and their many small parts were best tested manually.

I (Tom Hunter) used JUnit to create these tests in the **ControlPanelTest** class. This test class covers test cases for the classes ControlPanelManager and ControlPanelUI. The tests within this class start by ensuring a new GUI can be opened, then the following attempts login, or attempt to login, with a variety of inputs. Permission tests are also covered, which involve logging in to the admin and dummy accounts and then checking if the current user's permissions match that of the account.

There are currently two users to the control panel: "username" and "dummy" with the passwords "password" and "12345" respectively. This information and a more in-depth walkthrough is covered in the manual section of this document.

Tests such as this ensure only known users are able to login to the control panel, and users have their appropriately assigned permissions. Ensuring these components work successfully is paramount for a secure and reliable system.

Source Control and Documentation

As a quick final note, our group has used GitHub to manage the contributions and source code of our project. Our GitHub usernames can be found in the title page of this document and our key changes have been well-documented. We have also kept up regular communications on discord detailing our additions and discussing our workflow. This document was also created together via Google Docs.

We have also each included detailed comments to all components of our code to describe its functionality and outputs at any point. We have used JavaDoc-style comments for each of our classes and major methods as well as regular comments for variables and in-method descriptions.