Strathmore University

Faculty of Information Technology

# BACHELOR OF BUSINESS INFORMATION TECHNOLOGY

## BBT 2202: ADVANCED OBJECT-ORIENTED PROGRAMMING

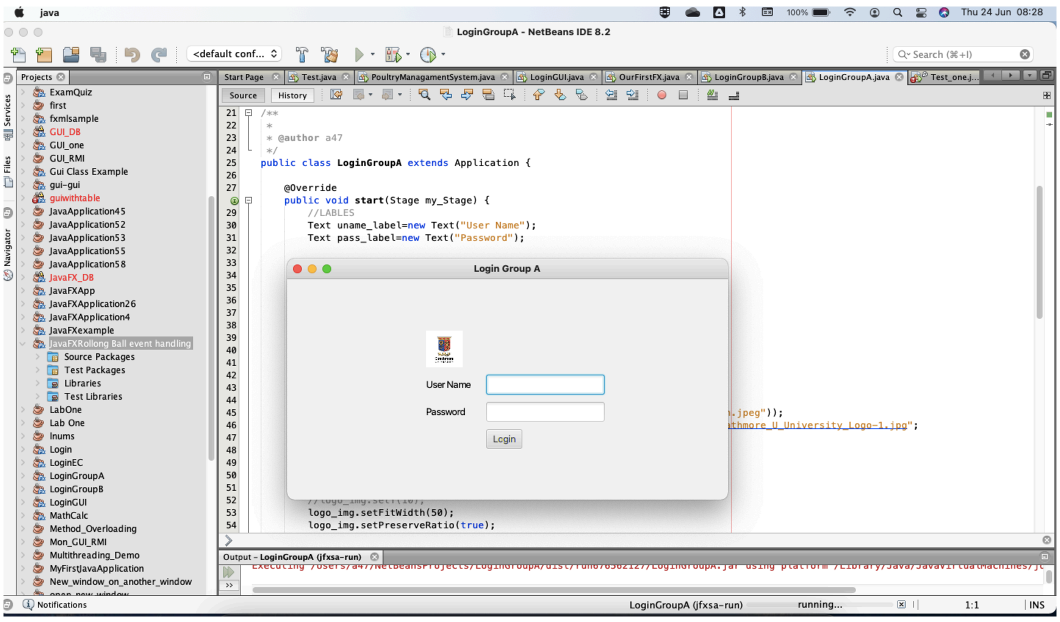
**May -Aug 2021 Session**

**Lab Two**

**Date: 24th Jun 2021**

**NOTE: The work should be done in pairs (Groups of TWO).**

**HOW TO SUBMIT: Take the screenshots of your GUI with Netbeans code on the background (Look at the example below), paste them on a single word document. You can submit it as a word document or a pdf.**



```
/**
 *
 * @author a47
 */
public class LoginGroupA extends Application {

    @Override
    public void start(Stage my_Stage) {
        //LABLES
        Text uname_label=new Text("User Name");
        Text pass_label=new Text("Password");
```

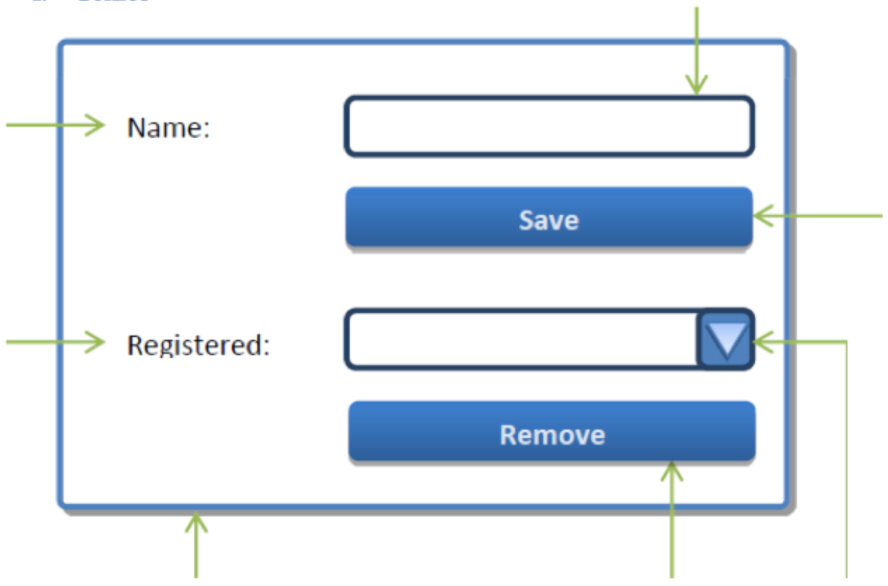
```
        logo_img.setFitWidth(50);
        logo_img.setPreserveRatio(true);
```

## Task: Developing GUIs in JavaFX similar to the provided

Develop the GUIs for the Video Library System as below using JavaFX, note that the look and feel will be different and much better from the provided GUI diagrams. Your task is to design them as close and organized as the original, the better the design the more the marks. *<u>The first one has been done for you,</u>* follow it through then proceed to finish ALL the GUIs.

## Interface Designs

1. **Genres**

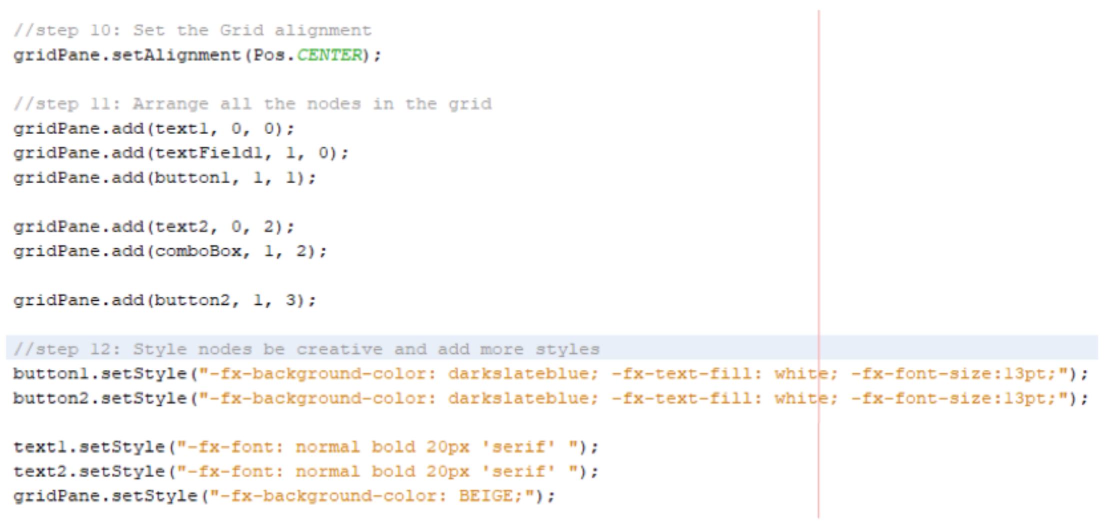


### Step A: Preparing the Scene Graph

As per your application, you need to prepare a scene graph with required nodes. Since the root node is the first node, you need to create a root node.

Inside the start() method follow the steps below:

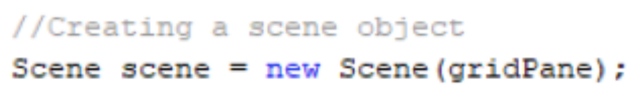
```
//step 1: creat label Name
Text text1 = new Text("Name:");

//step 2: creat label Rgistered
Text text2 = new Text("Registered:");

//step 3: Creat Text Filed for name
TextField textField1 = new TextField();

//step 4: creat Combo Box for Registerd
ComboBox comboBox = new ComboBox();


//step 5: Creat Buttons
Button button1 = new Button("Save");
Button button2 = new Button("Remove");

//step 6: Creating a Grid Pane and Import relevant Classes
GridPane gridPane = new GridPane();

//step 7: Set up size for the pane
gridPane.setMinSize(600, 400);

//step 8: Set padding
gridPane.setPadding(new Insets(10, 10, 10, 10));

//step 9: Set the vertical and horizontal gaps between the columns
gridPane.setVgap(10);
gridPane.setHgap(10);
```



```java
//step 10: Set the Grid alignment
gridPane.setAlignment(Pos.CENTER);

//step 11: Arrange all the nodes in the grid
gridPane.add(text1, 0, 0);
gridPane.add(textField1, 1, 0);
gridPane.add(button1, 1, 1);

gridPane.add(text2, 0, 2);
gridPane.add(comboBox, 1, 2);

gridPane.add(button2, 1, 3);

//step 12: Style nodes be creative and add more styles
button1.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white; -fx-font-size:13pt;");
button2.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white; -fx-font-size:13pt;");

text1.setStyle("-fx-font: normal bold 20px 'serif' ");
text2.setStyle("-fx-font: normal bold 20px 'serif' ");
gridPane.setStyle("-fx-background-color: BEIGE;");
```

**Step B: Preparing the scene**

A JavaFX scene is represented by the Scene class of the package javafx.scene. You can create a Scene by instantiating this class as shown in the following cod block. While instantiating, it is mandatory to pass the root object to the constructor of the scene class.



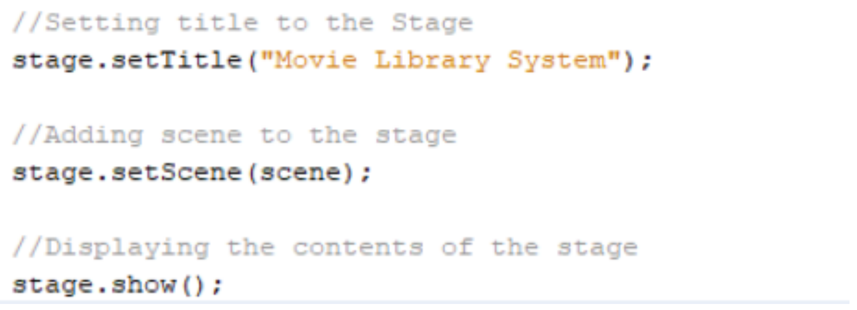
```java
//Creating a scene object
Scene scene = new Scene(gridPane);
```

**Step C: Preparing the stage**

This is the container of any JavaFX application and it provides a window for the application. It is represented by the Stage class of the package javafx.stage. An object of this class is passed as a parameter of the start() method of the Application class.

Using this object, you can perform various operations on the stage. Primarily you can perform the following –

- Set the title for the stage using the method setTitle().
- Attach the scene object to the stage using the setScene() method.
- Display the contents of the scene using the show() method as shown below.



```java
//Setting title to the Stage
stage.setTitle("Movie Library System");

//Adding scene to the stage
stage.setScene(scene);

//Displaying the contents of the stage
stage.show();
```

**Step D: Calling method launch() on the main method**

Compile and run the App by calling method launch as below

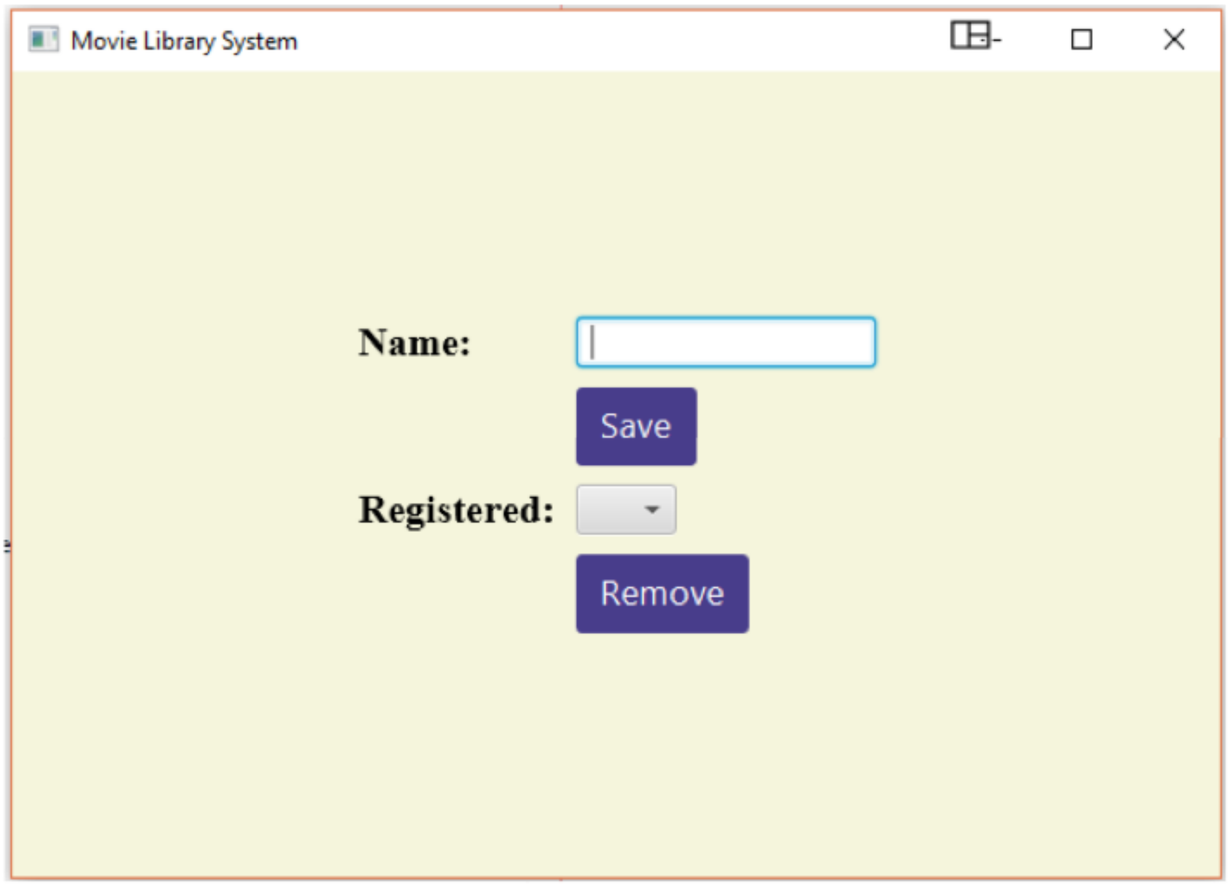
```
public static void main(String[] args) {
    launch(args);
}
```

Run the App. In case the error below displays then go back to the official open JavaFX source and fix the error as directed in the below URL based on your specific IDE

```
Error: JavaFX runtime components are missing, and are required to run this application
```

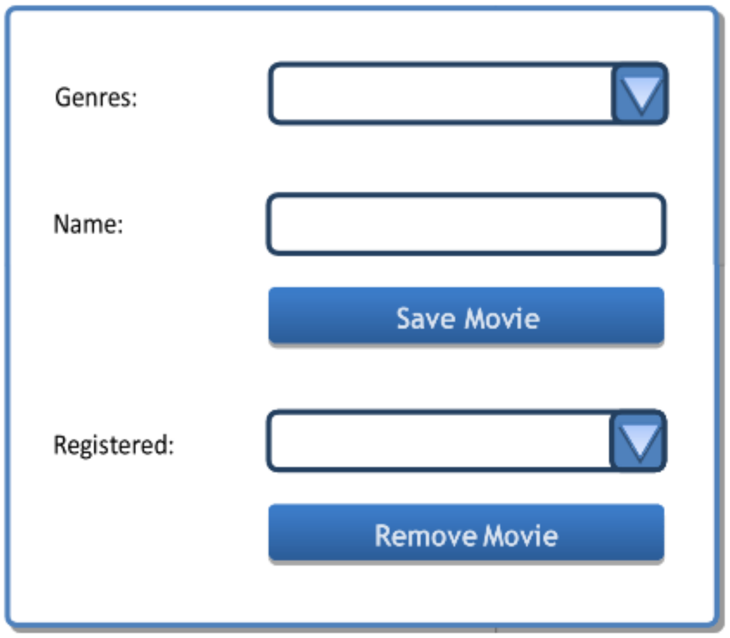
https://openjfx.io/openjfx-docs/

**Output**



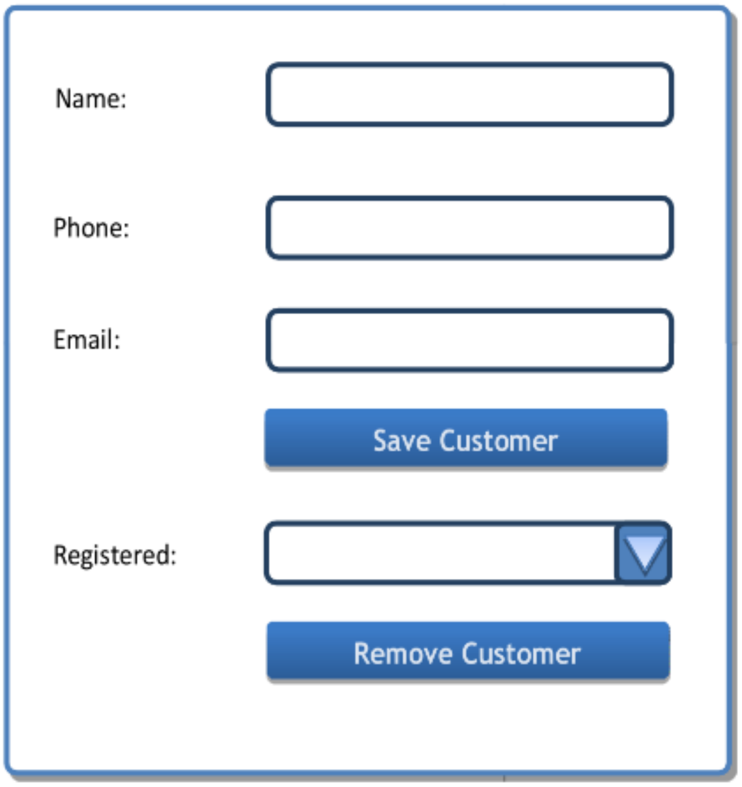
**2. Movies**

Now proceed to create a JavaFX GUI similar to the below in the same project If a movie is added, it is linked to the genre selected on the drop-down list labelled “Genre”.  “Registered” shows a list of all movies in the selected genre only.



**3. Customers**

“Registered” shows a list of all customers already registered into the system

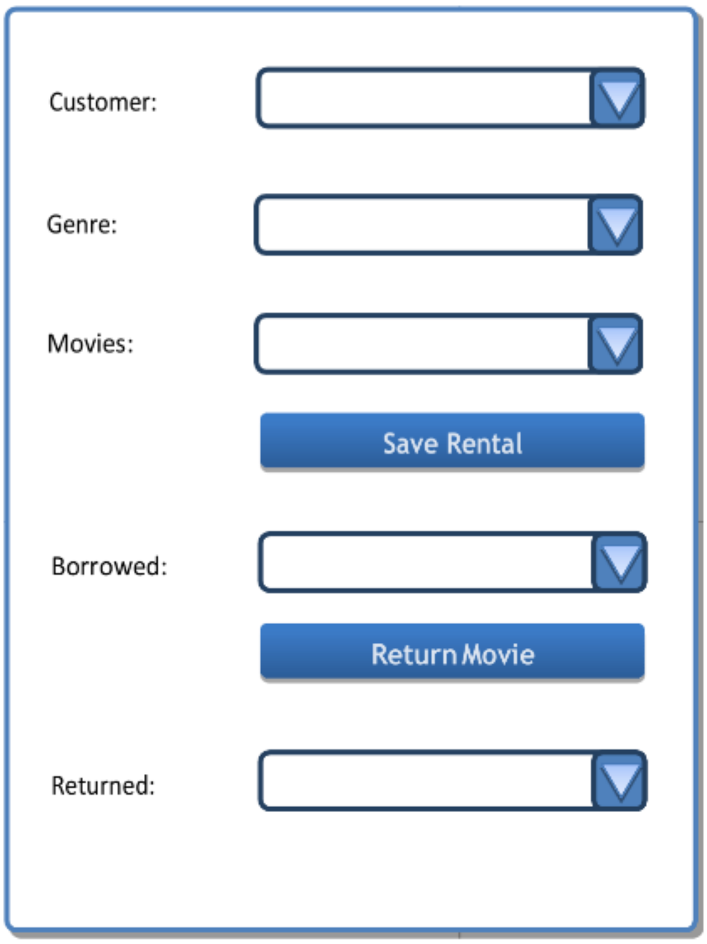


**4. Rentals**

This interface assigns a movie to the client selected in the drop-down menu labelled "Customer". On selecting a "Genre" the drop-down list labelled "Movies" shows those movies registered to the selected genre. Clicking on "Save rental" assigns the movie to the selected customer.

"Borrowed" shows a list of all movies borrowed by the selected customer. Clicking on "Return Movie" sets the selected movie for the selected customer as returned.

"Returned" shows the list of movies that had been borrowed but have been returned by the selected user.


Customer:
Genre:
Movies:
Save Rental
Borrowed:
Return Movie
Returned: