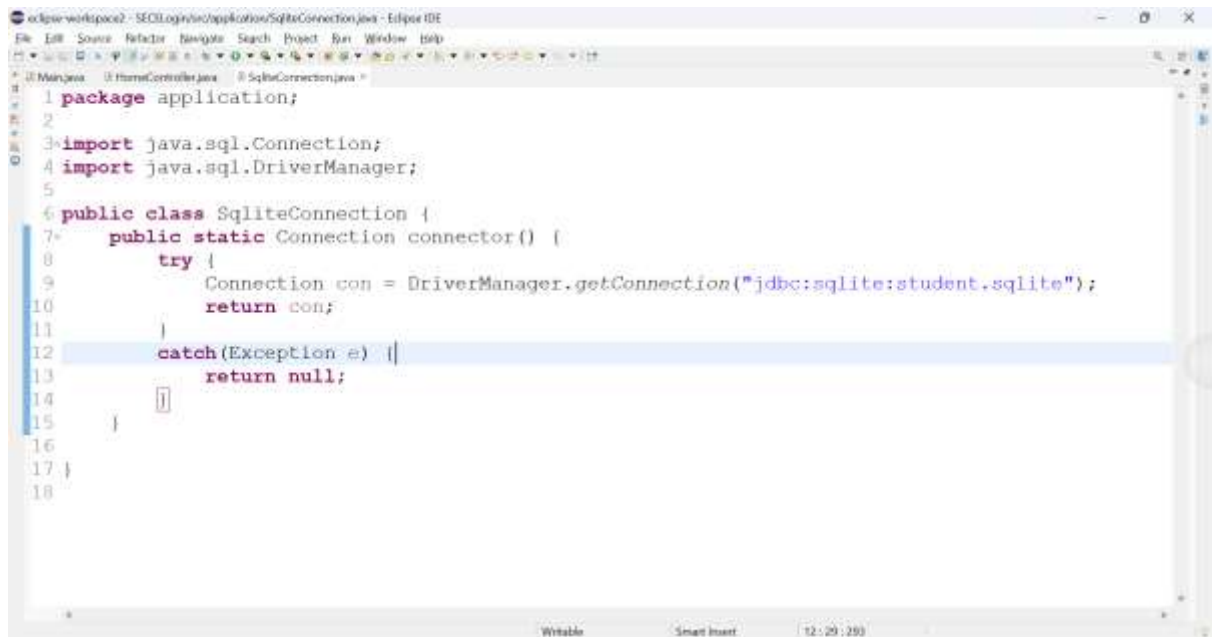


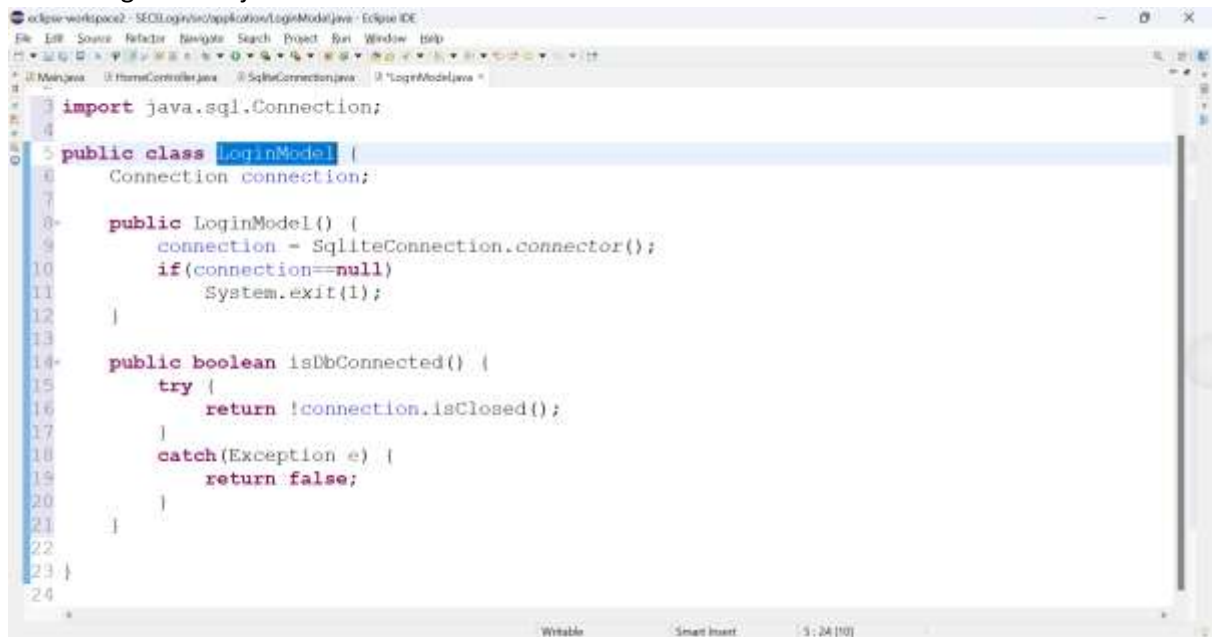
1. Create SQLiteConnection.java file



The screenshot shows the Eclipse IDE with the file `SQLiteConnection.java` open. The code defines a package `application` and imports `java.sql.Connection` and `java.sql.DriverManager`. It then defines a public class `SQLiteConnection` with a static method `connector()`. This method attempts to establish a database connection using `DriverManager.getConnection("jdbc:sqlite:student.sqlite");` and returns the connection. If an exception occurs, it catches the `Exception` and returns `null`.

```
1 package application;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5
6 public class SQLiteConnection {
7     public static Connection connector() {
8         try {
9             Connection con = DriverManager.getConnection("jdbc:sqlite:student.sqlite");
10            return con;
11        }
12        catch(Exception e) {}
13        return null;
14    }
15 }
16
17 }
18
```

2. Create LoginModel.java



The screenshot shows the Eclipse IDE with the file `LoginModel.java` open. The code imports `java.sql.Connection` and defines a public class `LoginModel`. It has a private attribute `Connection connection`. The `LoginModel()` constructor initializes the `connection` by calling `SQLiteConnection.connector()` and checks if it is `null`; if so, it calls `System.exit(1)`. The `isDbConnected()` method attempts to return `!connection.isClosed()` and catches any `Exception` to return `false`.

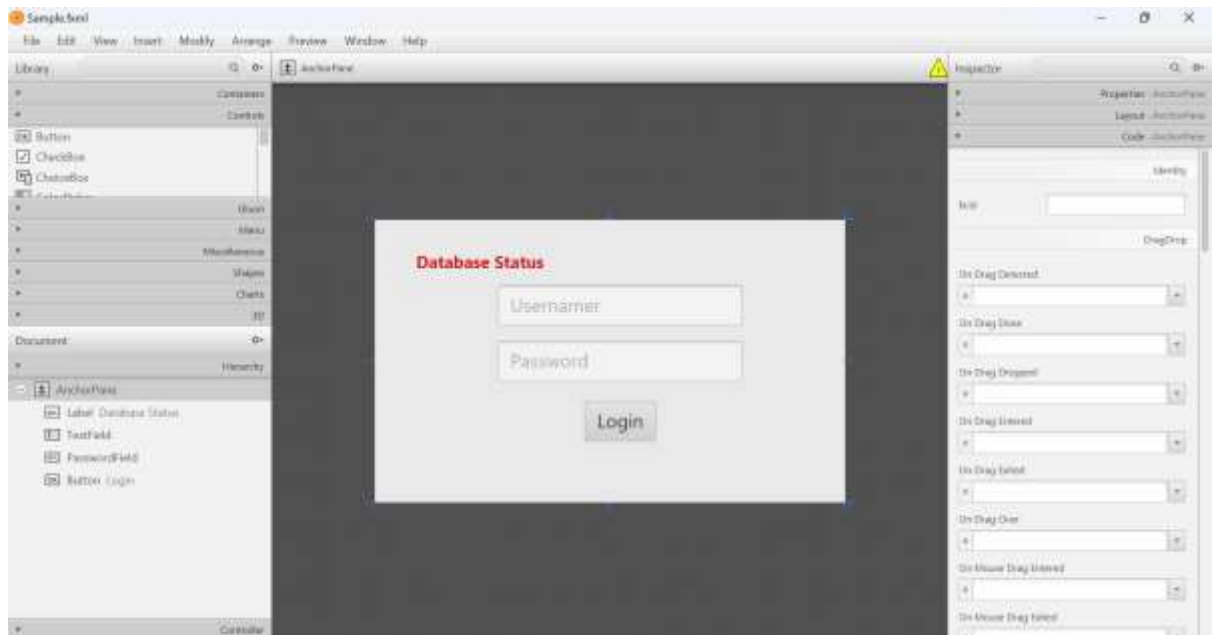
```
1 import java.sql.Connection;
2
3 public class LoginModel {
4     Connection connection;
5
6     public LoginModel() {
7         connection = SQLiteConnection.connector();
8         if(connection==null)
9             System.exit(1);
10    }
11
12    public boolean isDbConnected() {
13        try {
14            return !connection.isClosed();
15        }
16        catch(Exception e) {
17            return false;
18        }
19    }
20 }
21
22 }
23
24
```

3. Type below code in SampleContorller.java

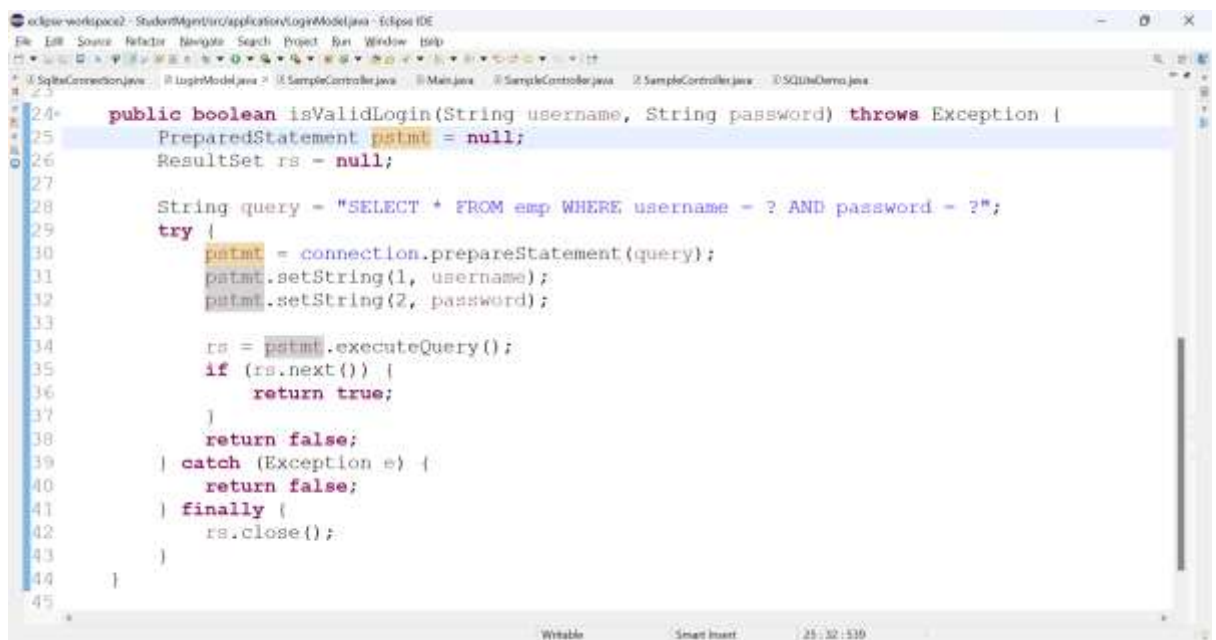


```
1 package application;
2
3 import java.net.URL;
4 import java.util.ResourceBundle;
5
6 import javafx.event.ActionEvent;
7 import javafx.fxml.FXML;
8 import javafx.fxml.Initializable;
9 import javafx.scene.control.Button;
10 import javafx.scene.control.Label;
11 import javafx.scene.control.PasswordField;
12 import javafx.scene.control.TextField;
13
14 public class SampleController implements Initializable {
15     LoginModel loginModel = new LoginModel();
16
17     @FXML
18     private Label myLabel;
19     @FXML
20     private Button mybtn;
21     @FXML
22     private TextField usernameTxt;
23
24     @Override
25     public void initialize(URL arg0, ResourceBundle arg1) {
26         if (loginModel.isDbConnected()) {
27             myLabel.setText("DB Connected...");
28         } else {
29             myLabel.setText("DB Connection Failed...");
30         }
31     }
32 }
33
```

4. Create the Scene as per below



5. Update code in LoginModel.java

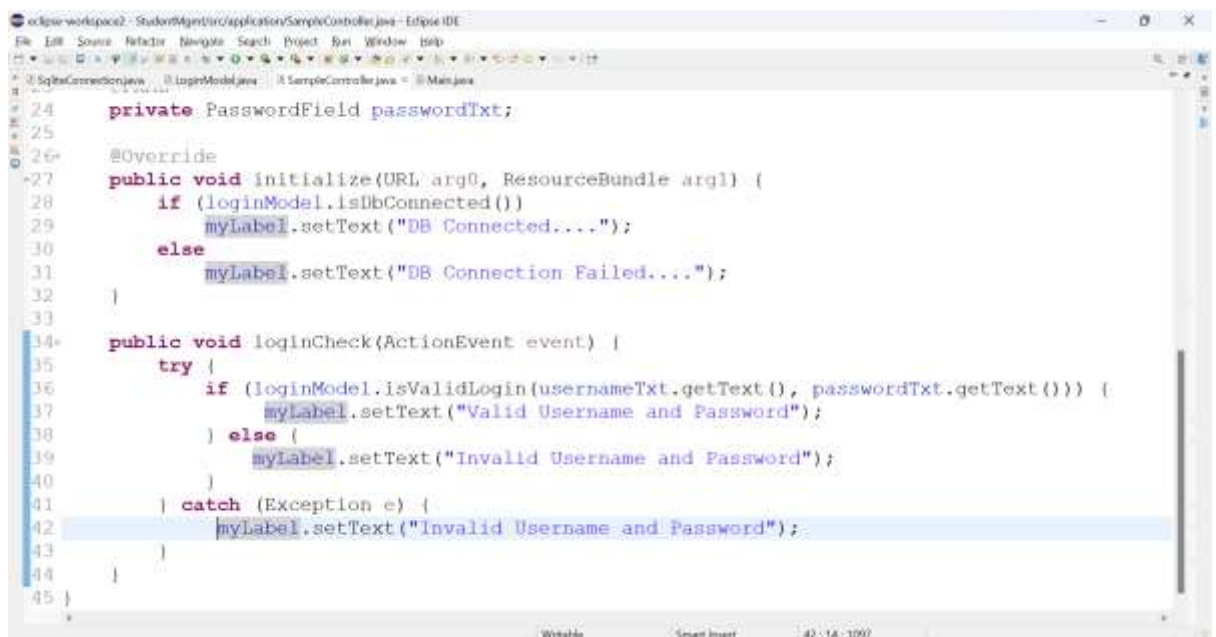


Step 6 : Updated in LoginModel.java



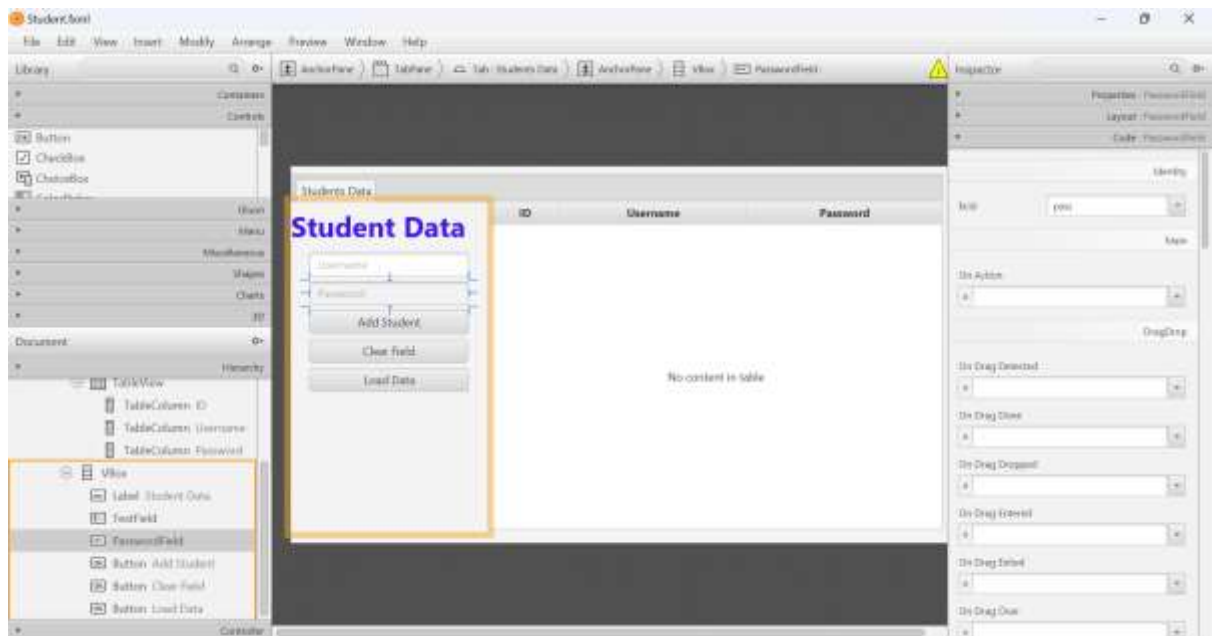
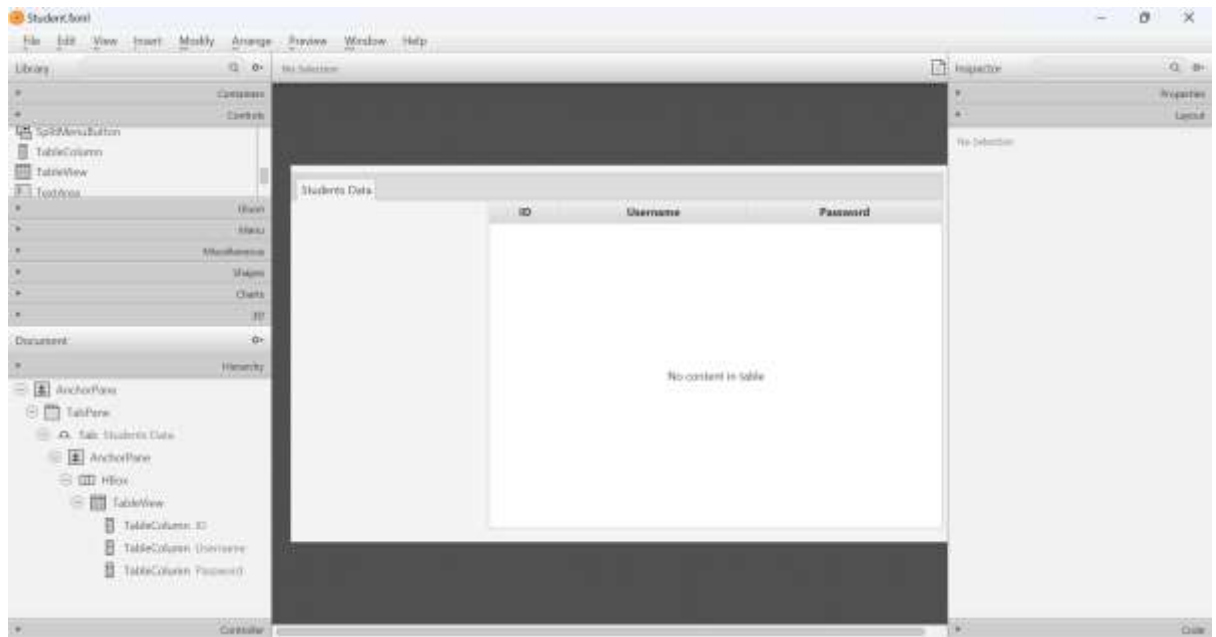
```
25 public boolean isValidLogin(String username, String password) throws Exception {
26     PreparedStatement pstmt = null;
27     ResultSet rs = null;
28
29     String query = "SELECT * FROM emp WHERE username = ? AND password = ?";
30     try {
31         pstmt = connection.prepareStatement(query);
32         pstmt.setString(1, username);
33         pstmt.setString(2, password);
34
35         rs = pstmt.executeQuery();
36         System.out.println("rs");
37         if (rs.next()) {
38             return true;
39         }
40         return false;
41     } catch (Exception e) {
42         return false;
43     } finally {
44         rs.close();
45     }
46 }
```

7. Type below code in SampleContorller.java



```
24 private PasswordField passwordTxt;
25
26 @Override
27 public void initialize(URL arg0, ResourceBundle arg1) {
28     if (loginModel.isDbConnected())
29         myLabel.setText("DB Connected....");
30     else
31         myLabel.setText("DB Connection Failed...");
32 }
33
34 public void loginCheck(ActionEvent event) {
35     try {
36         if (loginModel.isValidLogin(usernameTxt.getText(), passwordTxt.getText())) {
37             myLabel.setText("Valid Username and Password");
38         } else {
39             myLabel.setText("Invalid Username and Password");
40         }
41     } catch (Exception e) {
42         myLabel.setText("Invalid Username and Password");
43     }
44 }
45 }
```

8 . Design the Scene



8. Update SampleController.java

```

33     myLabel.setText("DB Connected...");
34     else
35         myLabel.setText("DB Connection Failed...");
36 }
37
38 public void loginCheck(ActionEvent event) {
39     try {
40         if (loginModel.isValidLogin(usernameTxt.getText(), passwordTxt.getText())) {
41             myLabel.setText("Valid Username and Password");
42             Stage primaryStage = new Stage();
43             Parent root = FXMLLoader.load(getClass().getResource("Student.fxml"));
44             Scene scene = new Scene(root);
45             scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());
46             primaryStage.setScene(scene);
47             primaryStage.show();
48         } else {
49             myLabel.setText("Invalid Username and Password");
50         }
51     } catch (Exception e) {
52         myLabel.setText("Invalid Username and Password");
53     }
54 }

```

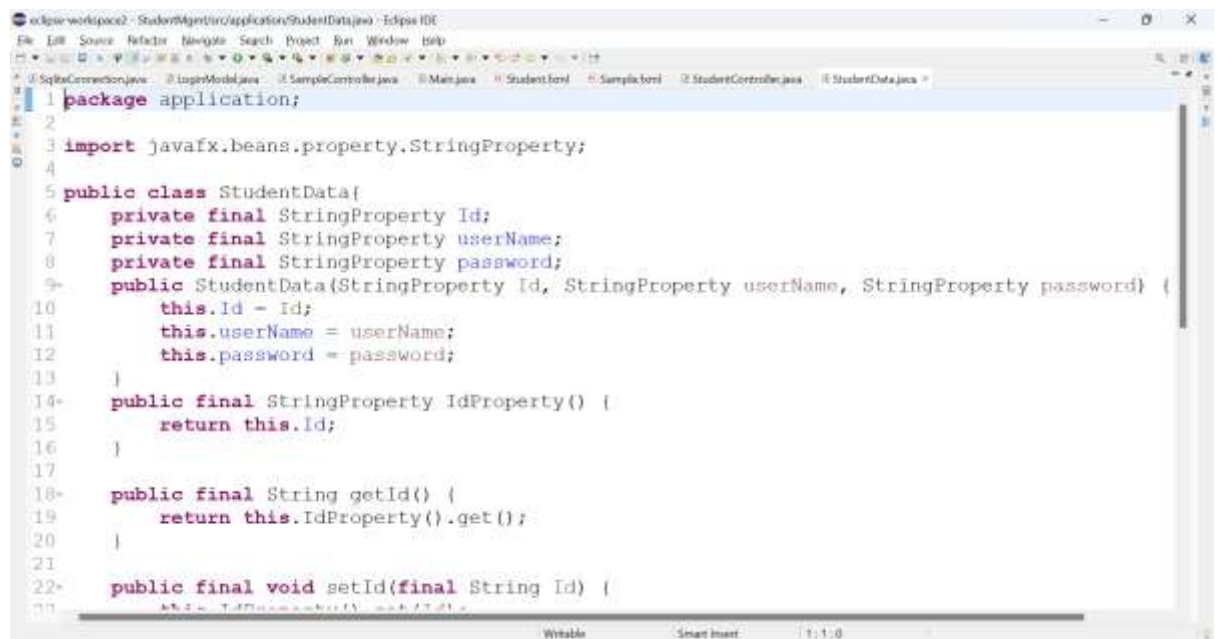
9. Auto Close Login Page – Update in LoginController.java

```

39 public void loginCheck(ActionEvent event) {
40     try {
41         if (loginModel.isValidLogin(usernameTxt.getText(), passwordTxt.getText())) {
42             myLabel.setText("Valid Username and Password");
43             Stage stage = (Stage) this.passwordTxt.getScene().getWindow();
44             stage.close();
45             openPage(event);
46         } else {
47             myLabel.setText("Invalid Username and Password");
48         }
49     } catch (Exception e) {
50         myLabel.setText("Invalid Username and Password");
51     }
52 }
53 public void openPage(ActionEvent event) throws IOException {
54     Stage primaryStage = new Stage();
55     Parent root = FXMLLoader.load(getClass().getResource("Student.fxml"));
56     Scene scene = new Scene(root);
57     scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());
58     primaryStage.setScene(scene);
59     primaryStage.show();
60 }

```

10. Create StudentData.java



```
1 package application;
2
3 import javafx.beans.property.StringProperty;
4
5 public class StudentData{
6     private final StringProperty Id;
7     private final StringProperty userName;
8     private final StringProperty password;
9     public StudentData(StringProperty Id, StringProperty userName, StringProperty password) {
10         this.Id = Id;
11         this.userName = userName;
12         this.password = password;
13     }
14     public final StringProperty IdProperty() {
15         return this.Id;
16     }
17
18     public final String getId() {
19         return this.IdProperty().get();
20     }
21
22     public final void setId(final String Id) {
23         this.IdProperty().set(Id);
24     }
25 }
```

Writable Smart Insert 1:1:0