

Subject Code: ITC521

Subject Name: Programming in Java 2

Assessment Item: 3

Assignment Number: 2

Task: 1

Programming Title: University Grading System

Lecturer Name: Dr. Recep Ulusoy

Student First Name: Bikash

Student Surname : Katwal

Student ID: 11605995

Assignment Submission Date: 09/30/ 2016

**Introduction**

University Grading System is GUI application used for maintaining the student marks. It contains the information textfield to enter the student marks which calculates the result and grade and display in the table.

**System Configuration:**

1. JavaFX GUI-Front end design
2. MySql- Backend database

**System Overview:**

The design includes **RestrictiveTextField** and **CustomTableView** which is a **customize textfield** and **customized tableview** used for entering numeric value and displaying the data. It also include label to display information message, TableView to list the record and buttons to add and update student marks and clear the text field.

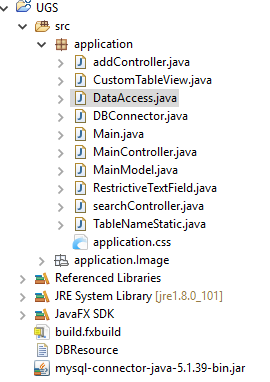
The following class is used in the application:

#**FrontEnd Design**

1. **Main.java**: Which is the main class used to built GUI using JavaFX which creates the table name based on the user input.
2. **MainModel.java**: It is a class containing constructor and accessor method.
3. **MainController.java**: It is the main GUI design for entering student record. It also contains function and method used for calculating the result and grade.
4. **RestrictiveTextField.java**: It is a customize textfield that limit the character and accept only numeric value.
5. **CustomTableView**.**java:** It is a tableview class which design tableview which is implemented in both **AddNewStudent** and **Search** form.
6. **TableNameStatic**.**java:** Display table name in the main form.
7. **addController**.**java:** It is the controller class that calculates the students marks and display the result in the tableview.
8. **searchController**.**java:** It is the controller class used to search the record from the database and display to the user.
9. **application.css**: It is a .**css** file used for styling the application.
10. **Image**: It is used for the icon and design of the application.

#**BackEnd Design**

1. **DBResource:** This provides the user and path information for application.
2. **DBConnector.java:** Connector class to connect database and application.
3. **DataAccess.java:** It contains function to create table, add, search and update data.
4. **Mysql-connector-java-5.1.39-bin.jar:** jar file to connect mysql server and the system.

****

**Valid Test and Explanation**

**System Graphics User Interface**

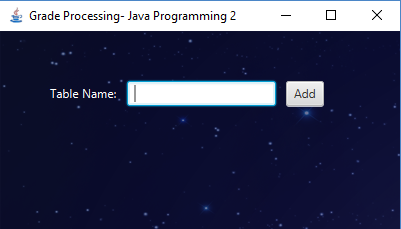
The design below are the GUI of the application:

The Icon and title of the application is shown in the figure below:

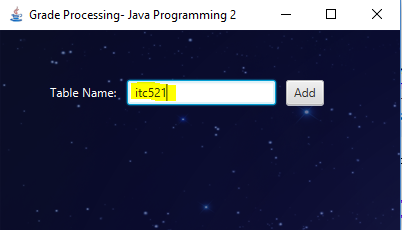
Icon.PNG

1. **Create Table**

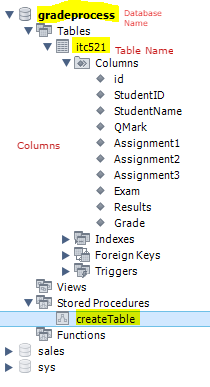
Enter table Name:



This figure is to enter the table name. when user click the **Add** button **or Press Enter,** enteredtable name is stored in mysql database.



As you see the table name **itc521** is entered which is then created in mysql server as shown in figure below:



The Store Procedure used to create table is in the figure below:

**CREATE** DEFINER**=**`root`@`localhost` **PROCEDURE** `createTable`**(IN** tableName **varchar(**50**))**

**BEGIN**

**SET** @tableName**=**tableName**;**

**SET** @createquery**=CONCAT(**'

CREATE TABLE IF NOT EXISTS ' **,** @tableName**,** ' (

id INT(11) UNSIGNED NOT NULL AUTO\_INCREMENT,

StudentID INT NOT NULL,

StudentName VARCHAR(50) NOT NULL,

QMark DOUBLE NOT NULL,

Assignment1 DOUBLE NOT NULL,

Assignment2 DOUBLE NOT NULL,

Assignment3 DOUBLE NOT NULL,

Exam DOUBLE NOT NULL,

Results DOUBLE NOT NULL,

Grade VARCHAR(50) NOT NULL,

PRIMARY KEY (id)

)

'**);**

**PREPARE** stmt **FROM** @createquery**;**

**EXECUTE** stmt**;**

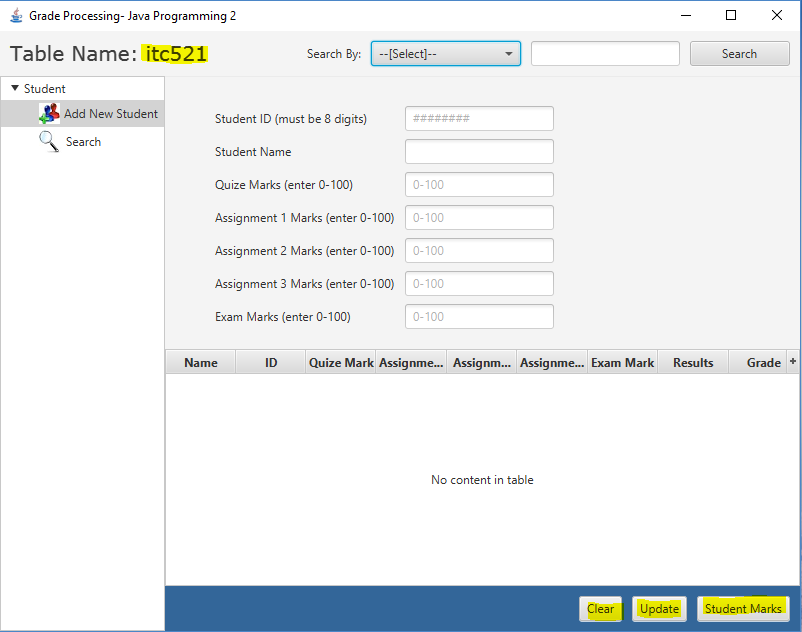
**DEALLOCATE** **PREPARE** stmt**;**

**SELECT** **TRUE;**

**END**

This store procedure creates the table as user input.The table with the same name is created only once. If the table name already exist it does not create table with a same name. The User can create any table under the database **gradeprocess.**

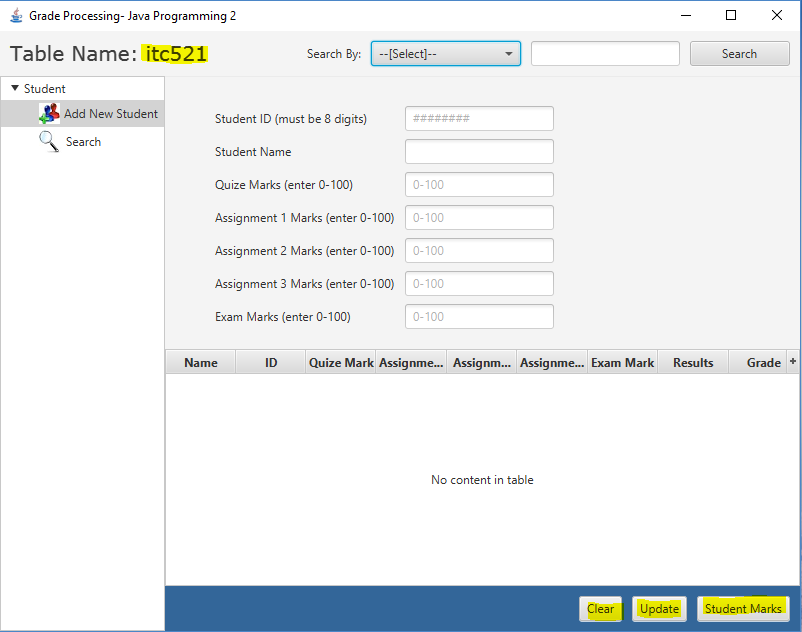
As shown in figure below , Table Name is displayed in the application which keeps on blinking during the execution of the system.



There are 3 buttons **Student Marks, Update, Clear:**

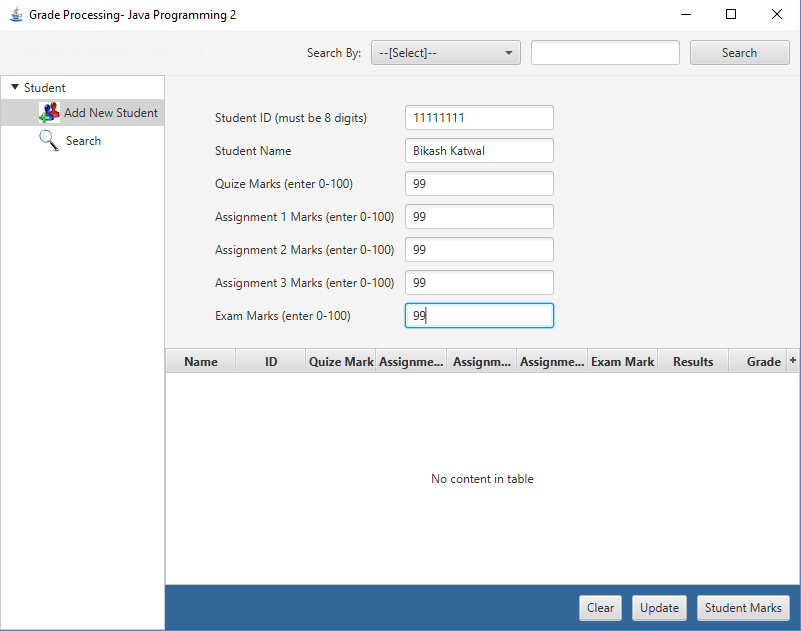
1. **Clear:** This button clears the textfield.
2. **Student Marks:** This button is used to add the record entered in the textfield. It calculates the **results** and **grade** and display in the table.
3. **Update:** This button update the record of the student.

**System Overall Design**

****

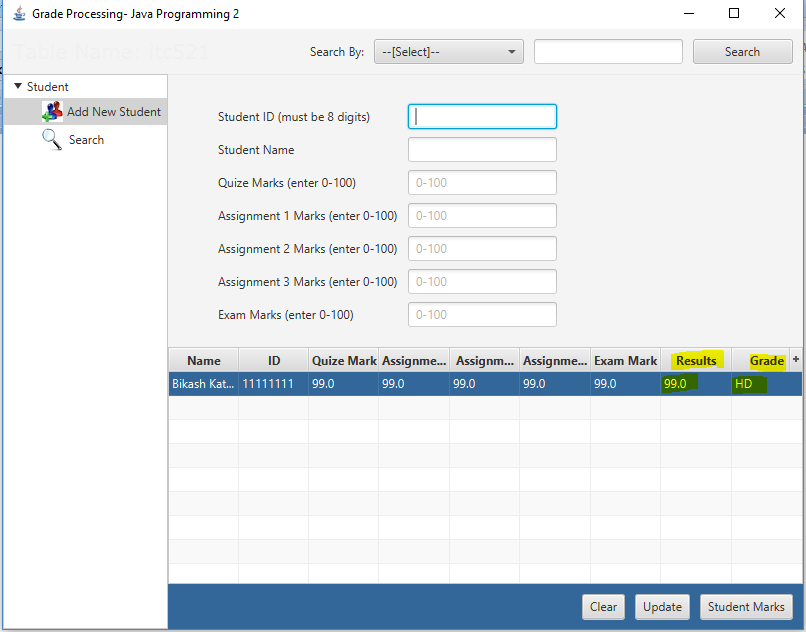
1. **Add New Student**

User enter student id, student Name, Quize Marks, Assignment 1, Assignment 2, Assignment 3, Exam Marks.

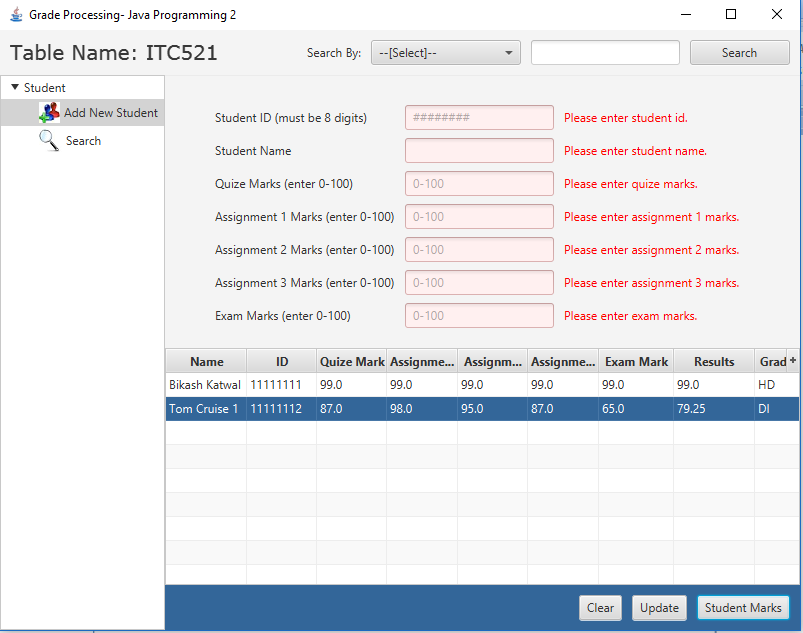


When you enter the valid data the design of application looks as the figure below. The system also clears the text and focus the cursor on the ID text field to make the work easier for the user.

When user click **Student Marks,** then the entered information is listed in the tableview as shown in figure below. It also calculates the result and grade and displays in the table.

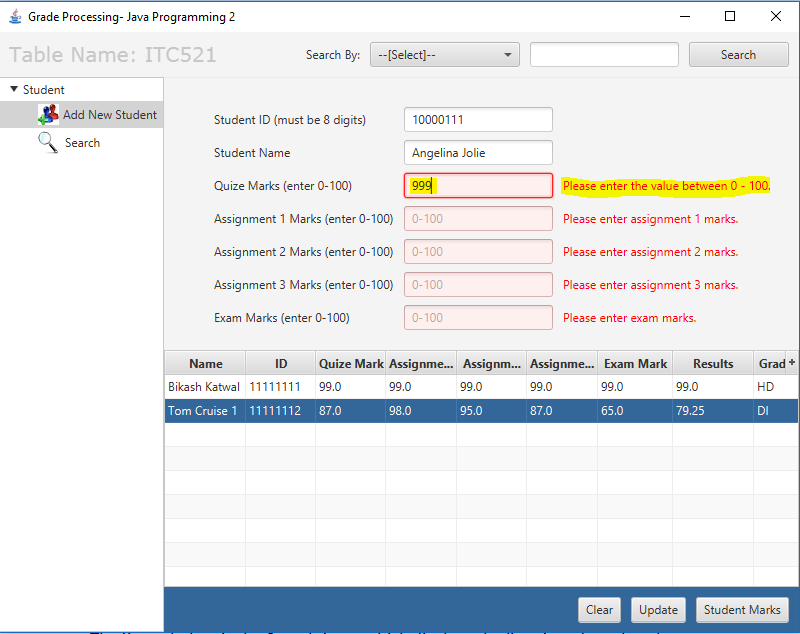


The entered row is automatically selected when the data is added. The data are added using **Student Marks** button or by pressing the **Enter** key when the cursor is on Exam Marks textfield. So, for the valid data entry all the textfield should be field otherwise it will **alert message** with **beep sound** as shown in the figure below:



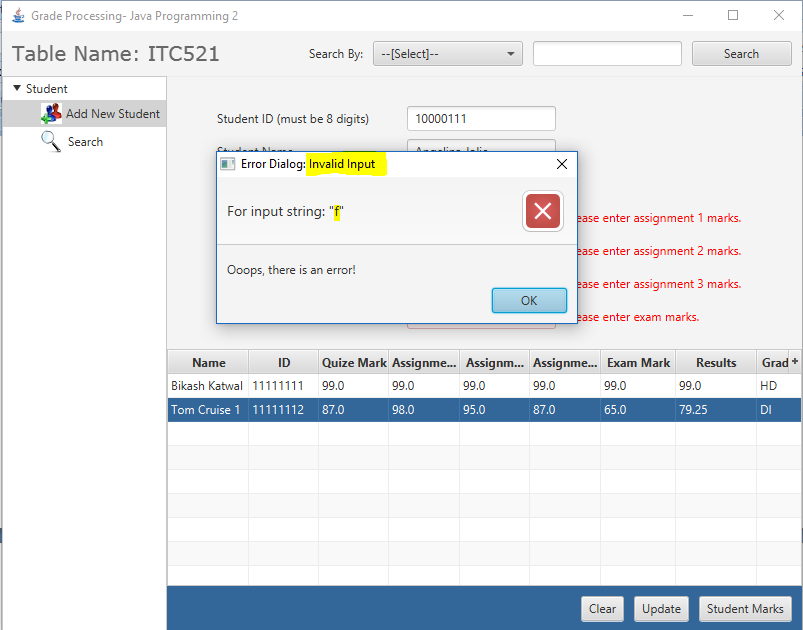
The figure above shows the information message in red text color and also the textfield color is changed to the red when user try to enter the empty record using Student Marks button.

And also, When user tries to enter the marks above 100, the following information is displayed to the user and alert with a beep sound which is shown below:



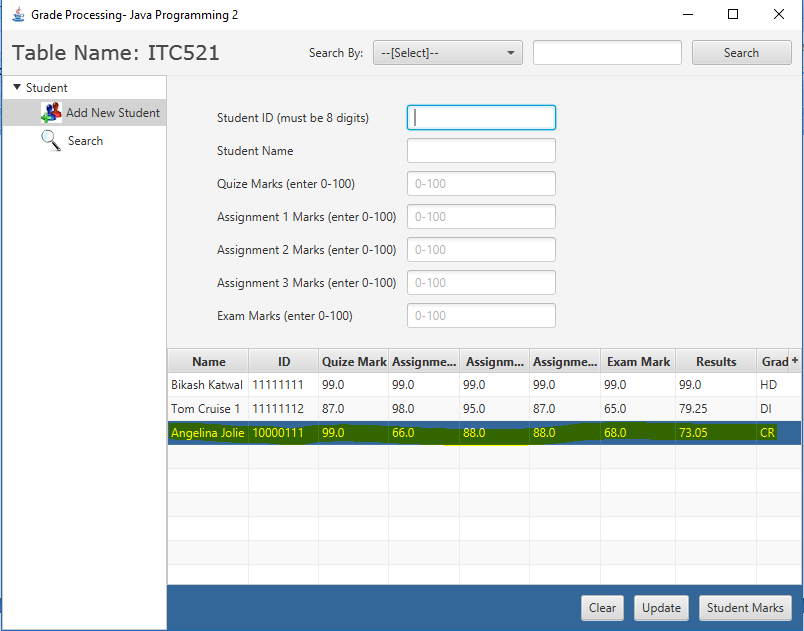
The information message “**Please enter the value between 0-100**” is displayed to the user when user tries to enter the value more then 100.

The system also provides the alert dialog box when the value is unacceptable. Which mean when you try to enter alphabet in the marks text field. The figure below shows the example:



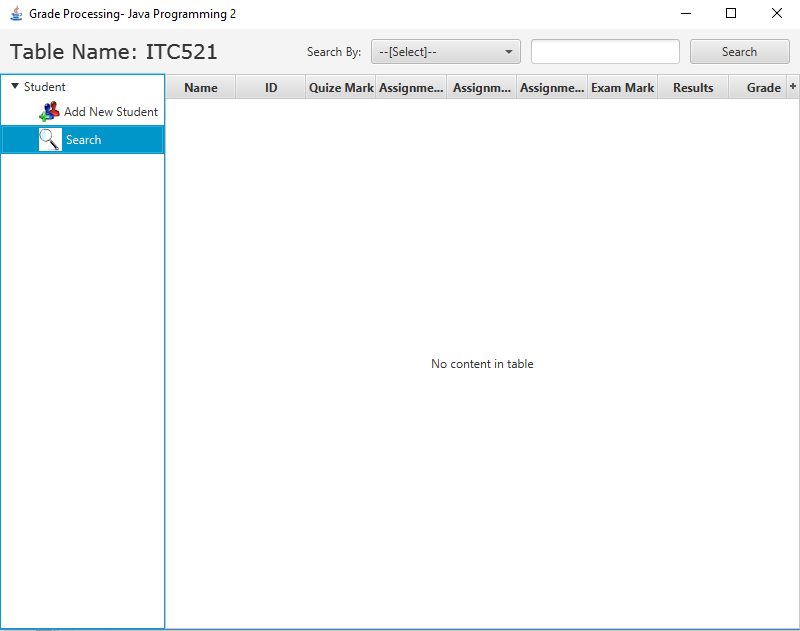
The above dialog box is displayed when uses tries to enter invalid input which shows a error dialog box for the user.

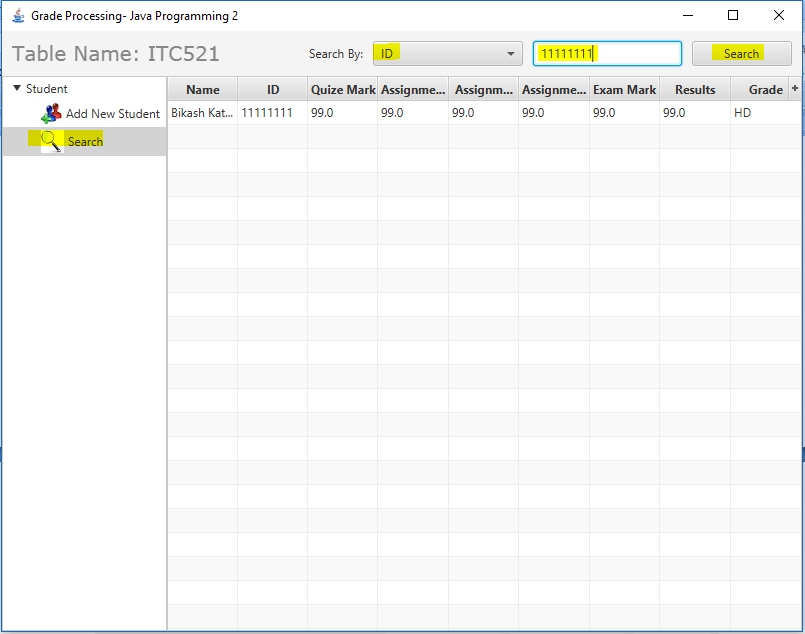
Finally, error free and valid data is entered in the database and displayed in the tableview as shown in figure below:



1. **Search Student**

The figure below is the Search form which displays the list of student that the user wants to search.

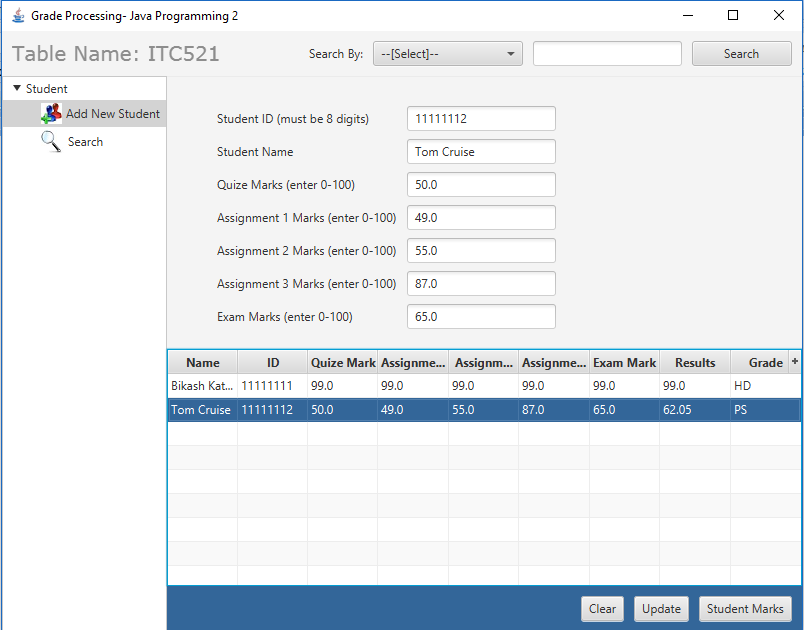
****

****

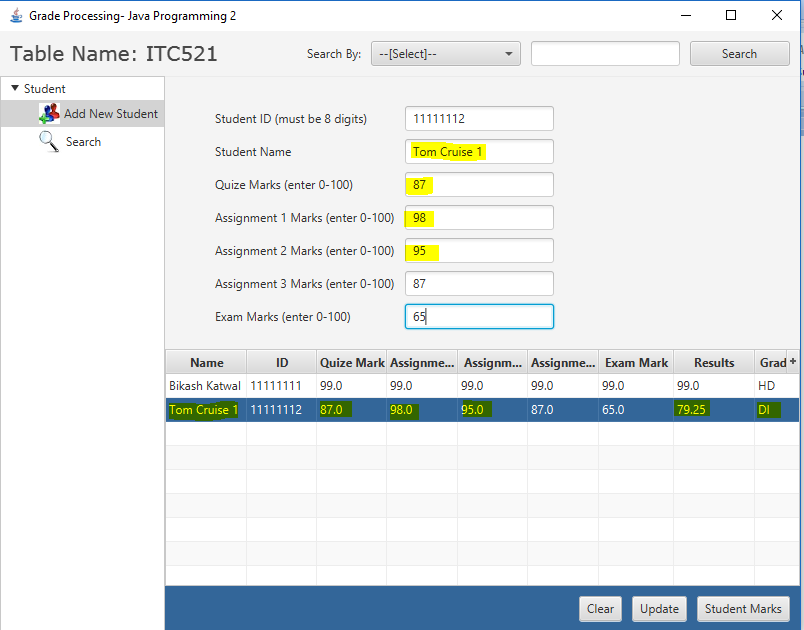
The user has to select the **ID** fromthe dropdown tosearch student record by **ID.** The ID textfield accept only 8 digit. When the user enter the student id and press enter or click search button the information of the specified student id is listed in the search form.

1. **Update Student Record**

When the user double click on the selected row the information will be pushed on the textfield where user can edit the appropriate data and update by clicking **Update** button.



Change made in the above figure’s data is shown in figure below:



The above figure shows the change is made on the textfield and when update button is clicked the data is updated and listed in the tableview as highlighted. You can also notice the change of Results and Grade as updated.

**Calculation method:**

When user click the Student Marks button it calculates the Result and Grade.

The results of a student is calculated using following formuls:

Results = Results = (Quiz \* 0.05)+(Assignment1 \* 0.15) +(Assignment2 \* 0.2) + (Assignment3 \* 0.10) + (Exam \* 0.5)

Whereas for the grading process the following range is used to calculate grade:

**Calculate Grade:**

HD: Results>=85

DI: 75<=Results<85

CR: 65<=Results<75

PS: 50<=Results<65

FL: Results<50

**All the Code is listed below:**

1. **Main.java**

**import** javafx**.**animation**.**FadeTransition**;**

**import** javafx**.**application**.\*;**

**import** javafx**.**geometry**.**Insets**;**

**import** javafx**.**stage**.**Stage**;**

**import** javafx**.**util**.**Duration**;**

**import** javafx**.**scene**.**Scene**;**

**import** javafx**.**scene**.**control**.**Button**;**

**import** javafx**.**scene**.**control**.**Label**;**

**import** javafx**.**scene**.**control**.**TextField**;**

**import** javafx**.**scene**.**image**.\*;**

**import** javafx**.**scene**.**input**.**KeyCode**;**

**import** javafx**.**scene**.**layout**.\*;**

**import** javafx**.**scene**.**paint**.**Color**;**

public class Main **extends** Application **{**

private Label lbl1**;**

private Label lbl2**;**

private TextField txtTableName**;**

private Button btnadd**;**

private Stage primaryStage**;**

private Scene scene**;**

private MainController mainController**;**

private addController controllerAdd**;**

BorderPane rootLayout **=** **new** BorderPane**();**

AnchorPane anchorPane **=** **new** AnchorPane**();**

@Override

public void start**(**Stage primaryStage**)** **{**

**try** **{**

**this.**primaryStage **=** primaryStage**;**

**this.**scene **=** **new** Scene**(**anchorPane**,** 400**,** 200**);**

**this.**primaryStage**.**setTitle**(**"Grade Processing- Java Programming 2"**);**

**this.**primaryStage**.**getIcons**().**add**(new** Image**(**getClass**().**getResourceAsStream**(**"java.jpg"**)));**

systemConfigurationGUI**(**anchorPane**);**

primaryStage**.**setScene**(**scene**);**

primaryStage**.**show**();**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

// System Configuration Design a form to enter table name and save it in database

public void systemConfigurationGUI**(**AnchorPane anchorPane1**)** **{**

ImageView img **=** **new** ImageView**(**"infinite.jpg"**);**

FadeTransition fadeTransition **=** **new** FadeTransition**(**Duration**.**seconds**(**3**),** img**);**

fadeTransition**.**setFromValue**(**0**);**

fadeTransition**.**setToValue**(**1**);**

fadeTransition**.**play**();**

GridPane grid **=** **new** GridPane**();**

grid**.**setHgap**(**10**);**

lbl1 **=** **new** Label**(**"Table Name:"**);**

lbl1**.**setTextFill**(**Color**.**WHITE**);**

lbl2 **=** **new** Label**(**"Saved Sucessfully!!"**);**

lbl2**.**setVisible**(false);**

txtTableName **=** **new** TextField**();**

btnadd **=** **new** Button**(**"Add"**);**

grid**.**add**(**lbl1**,** 0**,** 0**);**

grid**.**add**(**txtTableName**,** 1**,** 0**);**

grid**.**add**(**btnadd**,** 2**,** 0**);**

grid**.**add**(**lbl2**,** 1**,** 1**);**

grid**.**setPadding**(new** Insets**(**50**,** 0**,** 0**,** 50**));**

FadeTransition fadeTransition2 **=** **new** FadeTransition**(**Duration**.**seconds**(**3**),** grid**);**

fadeTransition2**.**setFromValue**(**0**);**

fadeTransition2**.**setToValue**(**1**);**

fadeTransition2**.**play**();**

anchorPane1**.**getChildren**().**addAll**(**img**,** grid**);**

// Creates table

btnadd**.**setOnAction**(**e **->** **{**

createTableName**();**

**});**

// Pressed enter

txtTableName**.**setOnKeyPressed**(**keypressed **->** **{**

**if** **(**keypressed**.**getCode**()** **==** KeyCode**.**ENTER**)** **{**

createTableName**();**

**}**

**});**

**}**

//Display Table Name on the Top Left of the design

public void createTableName**()** **{**

**try** **{**

**if** **(!**txtTableName**.**getText**().**isEmpty**())** **{**

mainController **=** **new** MainController**();**

String tableName **=** txtTableName**.**getText**();**

**if** **(**mainController**.**createDatabaseTable**(**tableName**))** **{**

primaryStage**.**close**();**

TableNameStatic**.**setTableName**(**tableName**);**

mainController**.**setMainApp**(**rootLayout**,** primaryStage**,** scene**);**

**}**

**}**

**}** **catch** **(**Exception e1**)** **{**

controllerAdd**.**alertMessage**(**e1**.**getMessage**());**

**}**

**}**

public static void main**(**String**[]** args**)** **{**

launch**(**args**);**

**}**

**}**

**2.MainModel.Java**

**import** javafx**.**beans**.**property**.**SimpleDoubleProperty**;**

**import** javafx**.**beans**.**property**.**SimpleIntegerProperty**;**

**import** javafx**.**beans**.**property**.**SimpleStringProperty**;**

public class MainModel **{**

private int idStudent**;**

private SimpleIntegerProperty ID**;**

private SimpleStringProperty Name**;**

private SimpleDoubleProperty QMarks**;**

private SimpleDoubleProperty A1**;**

private SimpleDoubleProperty A2**;**

private SimpleDoubleProperty A3**;**

private SimpleDoubleProperty ExamMarks**;**

private SimpleDoubleProperty Result**;**

private SimpleStringProperty Grade**;**

public MainModel**()** **{**

**}**

//Constructor

public MainModel**(**Integer id**,** String name**,** double qMarks**,** double a1**,** double a2**,** double a3**,** double examMarks**,**

double result**,** String grade**)** **{**

**super();**

**this.**ID **=** **new** SimpleIntegerProperty**(**id**);**

**this.**Name **=** **new** SimpleStringProperty**(**name**);**

**this.**QMarks **=** **new** SimpleDoubleProperty**(**qMarks**);**

**this.**A1 **=** **new** SimpleDoubleProperty**(**a1**);**

**this.**A2 **=** **new** SimpleDoubleProperty**(**a2**);**

**this.**A3 **=** **new** SimpleDoubleProperty**(**a3**);**

**this.**ExamMarks **=** **new** SimpleDoubleProperty**(**examMarks**);**

**this.**Result **=** **new** SimpleDoubleProperty**(**result**);**

**this.**Grade **=** **new** SimpleStringProperty**(**grade**);**

**}**

//Constructor which is used to update the student record

public MainModel**(**int idStudent**,** Integer id**,** String name**,** double qMarks**,** double a1**,** double a2**,** double a3**,**

double examMarks**,** double result**,** String grade**)** **{**

**super();**

**this.**idStudent **=** idStudent**;**

**this.**ID **=** **new** SimpleIntegerProperty**(**id**);**

**this.**Name **=** **new** SimpleStringProperty**(**name**);**

**this.**QMarks **=** **new** SimpleDoubleProperty**(**qMarks**);**

**this.**A1 **=** **new** SimpleDoubleProperty**(**a1**);**

**this.**A2 **=** **new** SimpleDoubleProperty**(**a2**);**

**this.**A3 **=** **new** SimpleDoubleProperty**(**a3**);**

**this.**ExamMarks **=** **new** SimpleDoubleProperty**(**examMarks**);**

**this.**Result **=** **new** SimpleDoubleProperty**(**result**);**

**this.**Grade **=** **new** SimpleStringProperty**(**grade**);**

**}**

public Integer getIdStudent**()** **{**

**return** idStudent**;**

**}**

public void setIdStudent**(**int idStudent**)** **{**

**this.**idStudent **=** idStudent**;**

**}**

public Integer getID**()** **{**

**return** ID**.**get**();**

**}**

public String getName**()** **{**

**return** Name**.**get**();**

**}**

public double getQMarks**()** **{**

**return** QMarks**.**get**();**

**}**

public double getA1**()** **{**

**return** A1**.**get**();**

**}**

public double getA2**()** **{**

**return** A2**.**get**();**

**}**

public double getA3**()** **{**

**return** A3**.**get**();**

**}**

public double getExamMarks**()** **{**

**return** ExamMarks**.**get**();**

**}**

public Double getResult**()** **{**

**return** Result**.**get**();**

**}**

public String getGrade**()** **{**

**return** Grade**.**get**();**

**}**

public void setID**(**int iD**)** **{**

ID **=** **new** SimpleIntegerProperty**(**iD**);**

**}**

**}**

**3.MainController.java**

**import** javafx**.**animation**.**FadeTransition**;**

**import** javafx**.**beans**.**value**.**ChangeListener**;**

**import** javafx**.**beans**.**value**.**ObservableValue**;**

**import** javafx**.**geometry**.**Pos**;**

**import** javafx**.**scene**.**Scene**;**

**import** javafx**.**scene**.**control**.\*;**

**import** javafx**.**scene**.**control**.**Alert**.**AlertType**;**

**import** javafx**.**scene**.**image**.**Image**;**

**import** javafx**.**scene**.**image**.**ImageView**;**

**import** javafx**.**scene**.**input**.**KeyCode**;**

**import** javafx**.**scene**.**layout**.\*;**

**import** javafx**.**scene**.**text**.**Font**;**

**import** javafx**.**stage**.**Stage**;**

**import** javafx**.**util**.**Duration**;**

public class MainController **{**

DataAccess dataAccess **=** **new** DataAccess**();**

// Main Form

private TreeView**<**String**>** treeView**;**

private RestrictiveTextField txtSearch**;**

private Button btnSearch**;**

private ComboBox**<**String**>** cmbSearchBy**;**

boolean running **=** **true;**

public static String tableName**;**

private addController controllerAdd**;**

searchController controllerSearch**;**

// Reference to Main.java

private BorderPane root**;**

public void setMainApp**(**BorderPane rootLayout**,** Stage primaryStage**,** Scene scene**)** **throws** Exception **{**

**this.**root **=** rootLayout**;**

scene **=** **new** Scene**(**root**,** 800**,** 600**);**

root**.**prefWidthProperty**().**bind**(**scene**.**widthProperty**());**

root**.**prefHeightProperty**().**bind**(**scene**.**heightProperty**());**

primaryStage**.**setTitle**(**"Grade Processing- Java Programming 2"**);**

primaryStage**.**getIcons**().**add**(new** Image**(**getClass**().**getResourceAsStream**(**"java.jpg"**)));**

scene**.**getStylesheets**().**add**(**getClass**().**getResource**(**"application.css"**).**toExternalForm**());**

primaryStage**.**setScene**(**scene**);**

primaryStage**.**setOnCloseRequest**(**e **->** **{**

running **=** **false;**

**});**

Init**(**root**);**

primaryStage**.**show**();**

**}**

// Creating the design of Main

@SuppressWarnings**(**"unchecked"**)**

public void Init**(**BorderPane root**)** **throws** Exception **{**

HBox hBox **=** **new** HBox**();**

Region r1 **=** **new** Region**();**

HBox**.**setHgrow**(**r1**,** Priority**.**ALWAYS**);**

HBox hbox1 **=** **new** HBox**();**

hbox1**.**setAlignment**(**Pos**.**CENTER\_LEFT**);**

Label lbltableName **=** **new** Label**(**"Table Name: " **+** TableNameStatic**.**getTableName**());**

lbltableName**.**setFont**(**Font**.**font**(**"Verdana"**,** 20**));**

hbox1**.**getChildren**().**add**(**lbltableName**);**

FadeTransition fadeIn **=** **new** FadeTransition**(**Duration**.**INDEFINITE**,** lbltableName**);**

fadeIn**.**setFromValue**(**1**);**

fadeIn**.**setToValue**(**0**);**

fadeIn**.**setAutoReverse**(true);**

FadeTransition fadeOut **=** **new** FadeTransition**(**Duration**.**seconds**(**1**),** lbltableName**);**

fadeOut**.**setFromValue**(**0**);**

fadeOut**.**setToValue**(**1**);**

fadeOut**.**setAutoReverse**(true);**

// Implementation of Thread to fadeIn/Fadeout table Name

**new** Thread**()** **{**

public void run**()** **{**

**while** **(**running**)** **{**

fadeIn**.**play**();**

fadeOut**.**play**();**

**}**

**}**

**}.**start**();**

HBox topHbox **=** **new** HBox**();**

topHbox**.**setAlignment**(**Pos**.**BASELINE\_RIGHT**);**

topHbox**.**setSpacing**(**10**);**

hBox**.**getStyleClass**().**add**(**"TopHBox"**);**

Label lbl **=** **new** Label**(**"Search By:"**);**

// Design for combo box

cmbSearchBy **=** **new** ComboBox**<**String**>();**

cmbSearchBy**.**setPrefWidth**(**150**);**

cmbSearchBy**.**setPromptText**(**"--[Select]--"**);**

cmbSearchBy**.**getItems**().**add**(**"ID"**);**

txtSearch **=** **new** RestrictiveTextField**();**

// Design for button

btnSearch **=** **new** Button**(**"Search"**);**

btnSearch**.**setPrefWidth**(**100**);**

topHbox**.**getChildren**().**addAll**(**lbl**,** cmbSearchBy**,** txtSearch**,** btnSearch**);**

hBox**.**getChildren**().**addAll**(**hbox1**,** r1**,** topHbox**);**

root**.**setTop**(**hBox**);**

// tree view design

treeView **=** **new** TreeView**<**String**>();**

treeView**.**setPrefWidth**(**165**);**

Image addicon **=** **new** Image**(**getClass**().**getResourceAsStream**(**"add.jpg"**));**

Image searchicon **=** **new** Image**(**getClass**().**getResourceAsStream**(**"search.jpg"**));**

TreeItem**<**String**>** treeItem **=** **new** TreeItem**<>(**"Student"**);**

treeItem**.**setExpanded**(true);**

TreeItem**<**String**>** Add **=** **new** TreeItem**<>(**"Add New Student"**,** **new** ImageView**(**addicon**));**

TreeItem**<**String**>** Search **=** **new** TreeItem**<>(**"Search"**,** **new** ImageView**(**searchicon**));**

treeItem**.**getChildren**().**addAll**(**Add**,** Search**);**

treeView**.**setRoot**(**treeItem**);**

root**.**setLeft**(**treeView**);**

treeView**.**getSelectionModel**().**clearAndSelect**(**1**);**

eventHandling**();**

LoadGUI**(**root**);**

**}**

// Event handle for Main form

private void eventHandling**()** **{**

treeView**.**getSelectionModel**().**selectedItemProperty**().**addListener**(new** ChangeListener**<**Object**>()** **{**

public void changed**(**ObservableValue**<?>** observable**,** Object oldValue**,** Object newValue**)** **{**

@SuppressWarnings**(**"unchecked"**)**

TreeItem**<**String**>** selectedItem **=** **(**TreeItem**<**String**>)** newValue**;**

**if** **(**selectedItem**.**getValue**()** **==** "Add New Student"**)** **{**

controllerAdd **=** **new** addController**();**

**try** **{**

controllerAdd**.**addNewStudentGUI**(**root**);**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

**}** **else** **{**

**try** **{**

controllerSearch **=** **new** searchController**();**

controllerSearch**.**searchStudentGUI**(**root**,** cmbSearchBy**,** txtSearch**);**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

**}**

**});**

// Search the student record

btnSearch**.**setOnAction**(**e **->** **{**

**try** **{**

searchStudentDetails**();**

**}** **catch** **(**Exception e1**)** **{**

e1**.**printStackTrace**();**

**}**

**});**

// input search value in the textfield

txtSearch**.**setOnKeyPressed**(**keyPressed **->** **{**

**if** **(**keyPressed**.**getCode**()** **==** KeyCode**.**ENTER**)** **{**

**try** **{**

searchStudentDetails**();**

**}** **catch** **(**Exception e1**)** **{**

e1**.**printStackTrace**();**

**}**

treeView**.**getSelectionModel**().**clearAndSelect**(**2**);**

**}**

**});**

//Select the item to search(eg:ID)

cmbSearchBy**.**setOnAction**(**select **->** **{**

**if** **(**cmbSearchBy**.**getSelectionModel**().**getSelectedItem**().**equals**(**"ID"**))** **{**

txtSearch**.**setPromptText**(**"########"**);**

txtSearch**.**setMaxLength**(**8**);**// limit character

txtSearch**.**setRestrict**(**"[0-9]"**);**

**}** **else** **{**

txtSearch **=** **new** RestrictiveTextField**();**

**}**

**});**

**}**

//Search Student Details when user select ID from the drop down list

private void searchStudentDetails**()** **throws** Exception **{**

controllerSearch **=** **new** searchController**();**

**if** **(**cmbSearchBy**.**getSelectionModel**().**getSelectedIndex**()** **>** **-**1**)** **{**

controllerSearch**.**searchStudentGUI**(**root**,** cmbSearchBy**,** txtSearch**);**

treeView**.**getSelectionModel**().**clearAndSelect**(**2**);**

**}** **else** **{**

Alert alert **=** **new** Alert**(**AlertType**.**INFORMATION**,** "Please select the Search By value"**,** ButtonType**.**OK**);**

alert**.**setTitle**(**"Information:"**);**

alert**.**showAndWait**();**

**}**

**}**

//Load Add Controller GUI in Main GUI form

private void LoadGUI**(**BorderPane root**)** **throws** Exception **{**

**if** **(**treeView**.**getSelectionModel**().**getSelectedIndex**()** **==** 1**)** **{**

controllerAdd **=** **new** addController**();**

**try** **{**

controllerAdd**.**addNewStudentGUI**(**root**);**

**}** **catch** **(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

**}** **else** **{**

controllerSearch **=** **new** searchController**();**

controllerSearch**.**searchStudentGUI**(**root**,** cmbSearchBy**,** txtSearch**);**

**}**

**}**

//Create table in database

public boolean createDatabaseTable**(**String tableName**)** **throws** Exception **{**

**return** dataAccess**.**createTableName**(**tableName**);**

**}**

**}**

**4.** **RestrictiveTextField.java**

**import** javafx**.**beans**.**property**.**IntegerProperty**;**

**import** javafx**.**beans**.**property**.**SimpleIntegerProperty**;**

**import** javafx**.**beans**.**property**.**SimpleStringProperty**;**

**import** javafx**.**beans**.**property**.**StringProperty**;**

**import** javafx**.**beans**.**value**.**ChangeListener**;**

**import** javafx**.**beans**.**value**.**ObservableValue**;**

**import** javafx**.**scene**.**control**.**TextField**;**

public class RestrictiveTextField **extends** TextField **{**

private IntegerProperty maxLength **=** **new** SimpleIntegerProperty**(this,** "maxLength"**,** **-**1**);**

private StringProperty restrict **=** **new** SimpleStringProperty**(this,** "restrict"**);**

//Restrict the non numeric and the count the digit.

public RestrictiveTextField**(){**

textProperty**().**addListener**(new** ChangeListener**<**String**>()** **{**

private boolean ignore**;**

@Override

public void changed**(**ObservableValue**<?** **extends** String**>** observable**,** String oldValue**,** String newValue**)** **{**

**if(**ignore **||** newValue**==null)**

**return;**

**if(**maxLength**.**get**()>-**1**&&**newValue**.**length**()>**maxLength**.**get**()){**

ignore**=true;**

setText**(**newValue**.**substring**(**0**,** maxLength**.**get**()));**

ignore**=false;**

**}**

**if** **(**restrict**.**get**()** **!=** **null** **&&** **!**restrict**.**get**().**equals**(**""**)** **&&** **!**newValue**.**matches**(**restrict**.**get**()** **+** "\*"**))** **{**

ignore **=** **true;**

setText**(**oldValue**);**

ignore **=** **false;**

**}**

**}**

**});**

**}**

/\*\*

\* The max length property.

\*

\* @return The max length property.

\*/

public IntegerProperty maxLengthProperty**()** **{**

**return** maxLength**;**

**}**

/\*\*

\* Gets the max length of the text field.

\*

\* @return The max length.

\*/

public int getMaxLength**()** **{**

**return** maxLength**.**get**();**

**}**

/\*\*

\* Sets the max length of the text field.

\*

\* @param maxLength The max length.

\*/

public void setMaxLength**(**int maxLength**)** **{**

**this.**maxLength**.**set**(**maxLength**);**

**}**

/\*\*

\* The restrict property.

\*

\* @return The restrict property.

\*/

public StringProperty restrictProperty**()** **{**

**return** restrict**;**

**}**

/\*\*

\* Gets a regular expression character class which restricts the user input.

\*

\* @return The regular expression.

\* @see #getRestrict()

\*/

public String getRestrict**()** **{**

**return** restrict**.**get**();**

**}**

/\*\*

\* Sets a regular expression character class which restricts the user input.

\* E.g. [0-9] only allows numeric values.

\*

\* @param restrict The regular expression.

\*/

public void setRestrict**(**String restrict**)** **{**

**this.**restrict**.**set**(**restrict**);**

**}**

**}**

**5.CustomTableView.java**

**import** java**.**util**.**ArrayList**;**

**import** javafx**.**scene**.**control**.**TableColumn**;**

**import** javafx**.**scene**.**control**.**TableView**;**

**import** javafx**.**scene**.**control**.**cell**.**PropertyValueFactory**;**

public class CustomTableView **extends** TableView**<**MainModel**>** **{**

// Used to design Tableview with the columns

public ArrayList**<**TableColumn**<**MainModel**,** **?>>** getColumn**(**TableView**<**MainModel**>** table**)** **{**

table**.**setStyle**(**"-fx-selection-bar: #336699; -fx-selection-bar-non-focused: #336699;"**);** // set

table**.**setColumnResizePolicy**(**CONSTRAINED\_RESIZE\_POLICY**);**

table**.**setTableMenuButtonVisible**(true);**

int i**;**

ArrayList**<**TableColumn**<**MainModel**,** **?>>** columns **=** **new** ArrayList**<**TableColumn**<**MainModel**,** **?>>();**

String**[]** columnNames **=** **{** "Name"**,** "ID"**,** "Quize Mark"**,** "Assignment 1"**,** "Assignment 2"**,**"Assignment 3"**,**"Exam Mark"**,** "Results"**,** "Grade" **};**// Header

String**[]** variableNames **=** **{** "Name"**,** "ID"**,** "QMarks"**,** "A1"**,** "A2"**,**"A3"**,**"ExamMarks"**,** "Result"**,** "Grade" **};**// Variables

i **=** 0**;**

TableColumn**<**MainModel**,** String**>** nameCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Integer**>** idCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** qmarksCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** a1Col **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** a2Col **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** a3Col **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** exammarksCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** Double**>** resultCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

TableColumn**<**MainModel**,** String**>** gradeCol **=** **new** TableColumn**<>(**columnNames**[**i**++]);**

i **=** 0**;**

nameCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** String**>(**variableNames**[**i**++]));**

idCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Integer**>(**variableNames**[**i**++]));**

qmarksCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

a1Col**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

a2Col**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

a3Col**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

exammarksCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

resultCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** Double**>(**variableNames**[**i**++]));**

gradeCol**.**setCellValueFactory**(new** PropertyValueFactory**<**MainModel**,** String**>(**variableNames**[**i**++]));**

columns**.**add**(**nameCol**);**

columns**.**add**(**idCol**);**

columns**.**add**(**qmarksCol**);**

columns**.**add**(**a1Col**);**

columns**.**add**(**a2Col**);**

columns**.**add**(**a3Col**);**

columns**.**add**(**exammarksCol**);**

columns**.**add**(**resultCol**);**

columns**.**add**(**gradeCol**);**

**return** columns**;**

**}**

**}**

**6.** **TableNameStatic.java**

public final class TableNameStatic **{**

private static String tableName**;**

//get and set tableName

public static String getTableName**()** **{**

**return** tableName**;**

**}**

public static void setTableName**(**String tableName**)** **{**

TableNameStatic**.**tableName **=** tableName**;**

**}**

**}**

**7.** **addController.java**

**import** java**.**awt**.**Toolkit**;**

**import** java**.**text**.**DecimalFormat**;**

**import** javafx**.**collections**.**FXCollections**;**

**import** javafx**.**collections**.**ObservableList**;**

**import** javafx**.**geometry**.**Insets**;**

**import** javafx**.**geometry**.**Pos**;**

**import** javafx**.**scene**.**control**.\*;**

**import** javafx**.**scene**.**control**.**Alert**.**AlertType**;**

**import** javafx**.**scene**.**input**.**KeyCode**;**

**import** javafx**.**scene**.**layout**.\*;**

public class addController **{**

DataAccess dataAccess **=** **new** DataAccess**();**

MainModel modelMain **=** **new** MainModel**();**

private int selectedIndex**;**

// controller for Add new form

private TableView**<**MainModel**>** tableMain**;**

private CustomTableView customTableView **=** **new** CustomTableView**();**

private static ObservableList**<**MainModel**>** mainModel **=** FXCollections**.**observableArrayList**();**

// Customize text field

private RestrictiveTextField txtId**;**

private RestrictiveTextField txtStudentName**;**

private RestrictiveTextField txtqMark**;**

private RestrictiveTextField txtA1**;**

private RestrictiveTextField txtA2**;**

private RestrictiveTextField txtA3**;**

private RestrictiveTextField txtExamMarks**;**

private Button btnStudentMarks**;**

private Button btnUpdateStudent**;**

private Button btnClear**;**

// label to display message

private Label lblIDmsg**;**

private Label lblNamemsg**;**

private Label lblQuizemsg**;**

private Label lblA1msg**;**

private Label lblA2msg**;**

private Label lblA3msg**;**

private Label lblExammsg**;**

private String actionToPerform **=** "ADD"**;**

public void addNewStudentGUI**(**BorderPane root**)** **throws** Exception **{**

tableMain **=** **new** TableView**<**MainModel**>();**

VBox vBox **=** **new** VBox**();**

GridPane gridPane **=** **new** GridPane**();**

gridPane**.**setMaxSize**(**Double**.**MAX\_VALUE**,** Double**.**MAX\_VALUE**);**

gridPane**.**setHgap**(**10**);**

gridPane**.**setVgap**(**8**);**

gridPane**.**setPadding**(new** Insets**(**30**,** 0**,** 20**,** 50**));**

// Grid Pane for top

gridPane**.**add**(new** Label**(**"Student ID (must be 8 digits)"**),** 0**,** 0**);**

txtId **=** **new** RestrictiveTextField**();**

txtId**.**setPromptText**(**"########"**);**

txtId**.**setMaxLength**(**8**);**// limit character

txtId**.**setRestrict**(**"[0-9]"**);**// only accepts number

lblIDmsg **=** **new** Label**();**

gridPane**.**add**(**txtId**,** 1**,** 0**);**

gridPane**.**add**(**lblIDmsg**,** 2**,** 0**);**

gridPane**.**add**(new** Label**(**"Student Name"**),** 0**,** 1**);**

txtStudentName **=** **new** RestrictiveTextField**();**

lblNamemsg **=** **new** Label**();**

gridPane**.**add**(**txtStudentName**,** 1**,** 1**);**

gridPane**.**add**(**lblNamemsg**,** 2**,** 1**);**

gridPane**.**add**(new** Label**(**"Quize Marks (enter 0-100)"**),** 0**,** 2**);**

txtqMark **=** **new** RestrictiveTextField**();**

txtqMark**.**setPromptText**(**"0-100"**);**

txtqMark**.**setRestrict**(**"[0-9.]"**);**

lblQuizemsg **=** **new** Label**();**

gridPane**.**add**(**txtqMark**,** 1**,** 2**);**

gridPane**.**add**(**lblQuizemsg**,** 2**,** 2**);**

gridPane**.**add**(new** Label**(**"Assignment 1 Marks (enter 0-100)"**),** 0**,** 3**);**

txtA1 **=** **new** RestrictiveTextField**();**

txtA1**.**setPromptText**(**"0-100"**);**

txtA1**.**setRestrict**(**"[0-9.]"**);**

lblA1msg **=** **new** Label**();**

gridPane**.**add**(**txtA1**,** 1**,** 3**);**

gridPane**.**add**(**lblA1msg**,** 2**,** 3**);**

gridPane**.**add**(new** Label**(**"Assignment 2 Marks (enter 0-100)"**),** 0**,** 4**);**

txtA2 **=** **new** RestrictiveTextField**();**

txtA2**.**setPromptText**(**"0-100"**);**

txtA2**.**setRestrict**(**"[0-9.]"**);**

lblA2msg **=** **new** Label**();**

gridPane**.**add**(**txtA2**,** 1**,** 4**);**

gridPane**.**add**(**lblA2msg**,** 2**,** 4**);**

gridPane**.**add**(new** Label**(**"Assignment 3 Marks (enter 0-100)"**),** 0**,** 5**);**

txtA3 **=** **new** RestrictiveTextField**();**

txtA3**.**setPromptText**(**"0-100"**);**

txtA3**.**setRestrict**(**"[0-9.]"**);**

lblA3msg **=** **new** Label**();**

gridPane**.**add**(**txtA3**,** 1**,** 5**);**

gridPane**.**add**(**lblA3msg**,** 2**,** 5**);**

gridPane**.**add**(new** Label**(**"Exam Marks (enter 0-100)"**),** 0**,** 6**);**

txtExamMarks **=** **new** RestrictiveTextField**();**

txtExamMarks**.**setPromptText**(**"0-100"**);**

txtExamMarks**.**setRestrict**(**"[0-9.]"**);**

lblExammsg **=** **new** Label**();**

gridPane**.**add**(**txtExamMarks**,** 1**,** 6**);**

gridPane**.**add**(**lblExammsg**,** 2**,** 6**);**

// tableMain;

tableMain**.**getItems**().**clear**();**

tableMain**.**getColumns**().**addAll**(**customTableView**.**getColumn**(**tableMain**));**

tableMain**.**getItems**().**addAll**(**dataAccess**.**getAllStudents**());**

// HBox in the bottom

HBox hBox **=** **new** HBox**();**

hBox**.**prefWidth**(**Double**.**MAX\_VALUE**);**

btnStudentMarks **=** **new** Button**(**"Student Marks"**);**

btnUpdateStudent **=** **new** Button**(**"Update"**);**

btnClear **=** **new** Button**(**"Clear"**);**

hBox**.**getChildren**().**addAll**(**btnClear**,** btnUpdateStudent**,** btnStudentMarks**);**

hBox**.**setAlignment**(**Pos**.**BOTTOM\_RIGHT**);**

hBox**.**setSpacing**(**10**);**

hBox**.**setPadding**(new** Insets**(**10**,** 10**,** 10**,** 0**));**

hBox**.**getStyleClass**().**add**(**"hBoxBackGroundColor"**);**

vBox**.**getChildren**().**addAll**(**gridPane**,** tableMain**,** hBox**);**

root**.**setCenter**(**vBox**);**

eventHandling**();**

**}**

// Action performed by the controls.

private void eventHandling**()** **{**

setTextFieldErrorProperty**();**// Display the error message.

btnStudentMarks**.**setOnAction**(**e **->** **{**

**if** **(**actionToPerform **==** "ADD"**)** **{**

addStudentDetails**();**

**}**

**});**

btnUpdateStudent**.**setOnAction**(**e2 **->** **{**

**try** **{**

updateStudentRecord**();**

**}** **catch** **(**Exception e1**)** **{**

alertMessage**(**e1**.**getMessage**());**

**}**

**});**

// Add record in table view when enter key is pressed in txtExamMarks

txtExamMarks**.**setOnKeyPressed**(**keyPressed **->** **{**

**if** **(**keyPressed**.**getCode**()** **==** KeyCode**.**ENTER**)** **{**

**if** **(**actionToPerform **==** "ADD"**)** **{**

addStudentDetails**();**

**}** **else** **{**

**try** **{**

updateStudentRecord**();**

**}** **catch** **(**Exception e1**)** **{**

alertMessage**(**e1**.**getMessage**());**

**}**

**}**

**}**

**});**

// Pass selected value to text field

tableMain**.**setOnMouseClicked**(**mouseClicked **->** **{**

**if** **(**mouseClicked**.**getClickCount**()** **==** 2**)** **{**

selectedIndex **=** tableMain**.**getSelectionModel**().**getSelectedIndex**();**

modelMain **=** tableMain**.**getSelectionModel**().**getSelectedItem**();**

txtId**.**setText**(**modelMain**.**getID**().**toString**());**

txtStudentName**.**setText**(**modelMain**.**getName**());**

txtqMark**.**setText**(**Double**.**toString**(**modelMain**.**getQMarks**()));**

txtA1**.**setText**(**Double**.**toString**(**modelMain**.**getA1**()));**

txtA2**.**setText**(**Double**.**toString**(**modelMain**.**getA2**()));**

txtA3**.**setText**(**Double**.**toString**(**modelMain**.**getA3**()));**

txtExamMarks**.**setText**(**Double**.**toString**(**modelMain**.**getExamMarks**()));**

actionToPerform **=** "UPDATE"**;**

**}**

**});**

// Clear textfield

btnClear**.**setOnAction**(**clear **->** **{**

ClearAllTextField**();**

**});**

**}**

// Update student record

private void updateStudentRecord**()** **throws** NumberFormatException**,** Exception **{**

**if** **(**isCompleted**())** **{**

Double Result **=** getStudentResult**(**Double**.**parseDouble**(**txtqMark**.**getText**()),**

Double**.**parseDouble**(**txtA1**.**getText**()),** Double**.**parseDouble**(**txtA2**.**getText**()),**

Double**.**parseDouble**(**txtA3**.**getText**()),** Double**.**parseDouble**(**txtExamMarks**.**getText**()));**

dataAccess**.**updateStudentDetails**(**modelMain**.**getIdStudent**(),** Integer**.**parseInt**(**txtId**.**getText**()),**

txtStudentName**.**getText**(),** Double**.**parseDouble**(**txtqMark**.**getText**()),**

Double**.**parseDouble**(**txtA1**.**getText**()),** Double**.**parseDouble**(**txtA2**.**getText**()),**

Double**.**parseDouble**(**txtA3**.**getText**()),** Double**.**parseDouble**(**txtExamMarks**.**getText**()),** Result**,**

getStudentGrade**(**Result**));**

refreshTable**();**

tableMain**.**getSelectionModel**().**select**(**selectedIndex**);**

ClearAllTextField**();**

**}**

**}**

// Add the record in table view and save in database

private void addStudentDetails**()** **{**

**if** **(**isCompleted**())** **{**

Double Result **=** getStudentResult**(**Double**.**parseDouble**(**txtqMark**.**getText**()),**

Double**.**parseDouble**(**txtA1**.**getText**()),** Double**.**parseDouble**(**txtA2**.**getText**()),**

Double**.**parseDouble**(**txtA3**.**getText**()),** Double**.**parseDouble**(**txtExamMarks**.**getText**()));**

**try** **{**

dataAccess**.**saveStudent**(**Integer**.**parseInt**(**txtId**.**getText**()),** txtStudentName**.**getText**(),**

Double**.**parseDouble**(**txtqMark**.**getText**()),** Double**.**parseDouble**(**txtA1**.**getText**()),**

Double**.**parseDouble**(**txtA2**.**getText**()),** Double**.**parseDouble**(**txtA3**.**getText**()),**

Double**.**parseDouble**(**txtExamMarks**.**getText**()),** Result**,** getStudentGrade**(**Result**));**

**}** **catch** **(**Exception e1**)** **{**

alertMessage**(**e1**.**getMessage**());**

**}**

refreshTable**();**

tableMain**.**getSelectionModel**().**select**(**tableMain**.**getItems**().**size**()** **-** 1**);**

ClearAllTextField**();**

**}**

**}**

// Refresh the table after adding and updating student records

private void refreshTable**()** **{**

**try** **{**

mainModel **=** FXCollections**.**observableArrayList**(**dataAccess**.**getAllStudents**());**

tableMain**.**setItems**(**mainModel**);**

**}** **catch** **(**Exception e**)** **{**

alertMessage**(**e**.**getMessage**());**

**}**

**}**

// Add and returns input value to main model

public static ObservableList**<**MainModel**>** addStudentDetails**(**int id**,** String name**,** Double qMarks**,** Double a1**,** Double a2**,**

Double a3**,** Double examMarks**,** Double result**,** String grade**)** **{**

mainModel**.**addAll**(new** MainModel**(**id**,** name**,** qMarks**,** a1**,** a2**,** a3**,** examMarks**,** result**,** grade**));**

**return** mainModel**;**

**}**

// Calculate Result

public Double getStudentResult**(**Double QMarks**,** Double A1Marks**,** Double A2Marks**,** Double A3Marks**,** Double ExamMarks**)** **{**

DecimalFormat df**=new** DecimalFormat**(**"##.##"**);**

**return** Double**.**parseDouble**(**df**.**format**((**QMarks **\*** 0.05**)** **+** **(**A1Marks **\*** 0.15**)** **+** **(**A2Marks **\*** 0.2**)** **+** **(**A3Marks **\*** 0.10**)** **+** **(**ExamMarks **\*** 0.5**)));**

**}**

// Calculate Grade

public String getStudentGrade**(**Double result**)** **{**

**if** **(**result **>=** 85**)** **{**

**return** "HD"**;**

**}** **else** **if** **(**result **>=** 75 **&&** result **<** 85**)** **{**

**return** "DI"**;**

**}** **else** **if** **(**result **>=** 65 **&&** result **<** 75**)** **{**

**return** "CR"**;**

**}** **else** **if** **(**result **>=** 50 **&&** result **<** 65**)** **{**

**return** "PS"**;**

**}** **else**

**return** "FL"**;**

**}**

// Clear all text field

public void ClearAllTextField**()** **{**

txtId**.**clear**();**

txtStudentName**.**clear**();**

txtqMark**.**clear**();**

txtA1**.**clear**();**

txtA2**.**clear**();**

txtA3**.**clear**();**

txtExamMarks**.**clear**();**

txtId**.**requestFocus**();**

actionToPerform **=** "ADD"**;**

**}**

// validate checkbox empty or complete

public boolean isCompleted**()** **{**

int i **=** 0**;**

**if** **(**txtId**.**getText**().**isEmpty**())** **{**

lblIDmsg**.**setText**(**"Please enter student id."**);**

lblIDmsg**.**getStyleClass**().**add**((**"error-font"**));**

txtId**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtStudentName**.**getText**().**isEmpty**())** **{**

lblNamemsg**.**setText**(**"Please enter student name."**);**

lblNamemsg**.**getStyleClass**().**add**((**"error-font"**));**

txtStudentName**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtqMark**.**getText**().**isEmpty**())** **{**

lblQuizemsg**.**setText**(**"Please enter quize marks."**);**

lblQuizemsg**.**getStyleClass**().**add**((**"error-font"**));**

txtqMark**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtA1**.**getText**().**isEmpty**())** **{**

lblA1msg**.**setText**(**"Please enter assignment 1 marks."**);**

lblA1msg**.**getStyleClass**().**add**((**"error-font"**));**

txtA1**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtA2**.**getText**().**isEmpty**())** **{**

lblA2msg**.**setText**(**"Please enter assignment 2 marks."**);**

lblA2msg**.**getStyleClass**().**add**((**"error-font"**));**

txtA2**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtA3**.**getText**().**isEmpty**())** **{**

lblA3msg**.**setText**(**"Please enter assignment 3 marks."**);**

lblA3msg**.**getStyleClass**().**add**((**"error-font"**));**

txtA3**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**txtExamMarks**.**getText**().**isEmpty**())** **{**

lblExammsg**.**setText**(**"Please enter exam marks."**);**

lblExammsg**.**getStyleClass**().**add**((**"error-font"**));**

txtExamMarks**.**getStyleClass**().**add**((**"validation-error"**));**

i**++;**

**}**

**if** **(**i **>** 0**)** **{**

Toolkit**.**getDefaultToolkit**().**beep**();**

**return** **false;**

**}** **else** **{**

**return** **true;**

**}**

**}**

// Event for text changed property

private void setTextFieldErrorProperty**()** **{**

txtId**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtId**,** lblIDmsg**,** 1**);**

**});**

txtStudentName**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtStudentName**,** lblNamemsg**,** 2**);**

**});**

txtqMark**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtqMark**,** lblQuizemsg**,** 3**);**

**});**

txtA1**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtA1**,** lblA1msg**,** 3**);**

**});**

txtA2**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtA2**,** lblA2msg**,** 3**);**

**});**

txtA3**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtA3**,** lblA3msg**,** 3**);**

**});**

txtExamMarks**.**textProperty**().**addListener**(**e **->** **{**

setTextFieldError**(**txtExamMarks**,** lblExammsg**,** 3**);**

**});**

**}**

// Used to display information message in application

public void setTextFieldError**(**TextField textfield**,** Label lbl**,** int Flag**)** **{**

**switch** **(**Flag**)** **{**

**case** 1**:**

**if** **(!**textfield**.**getText**().**isEmpty**())** **{**

textfield**.**getStyleClass**().**removeAll**(**"validation-error"**);**

lbl**.**getStyleClass**().**removeAll**((**"error-font"**));**

lbl**.**setText**(**""**);**

**}**

**break;**

**case** 2**:**

**if** **(!**textfield**.**getText**().**isEmpty**())** **{**

textfield**.**getStyleClass**().**removeAll**(**"validation-error"**);**

lbl**.**getStyleClass**().**removeAll**((**"error-font"**));**

lbl**.**setText**(**""**);**

**}**

**break;**

**case** 3**:**// Check the value between 0-100

**try** **{**

**if** **(!**textfield**.**getText**().**trim**().**isEmpty**())** **{**

**if** **(**Double**.**parseDouble**(**textfield**.**getText**())** **<** 0 **||** Double**.**parseDouble**(**textfield**.**getText**())** **>** 100**)** **{**

textfield**.**getStyleClass**().**add**((**"validation-error"**));**

lbl**.**setText**(**"Please enter the value between 0 - 100."**);**

lbl**.**getStyleClass**().**add**((**"error-font"**));**

Toolkit**.**getDefaultToolkit**().**beep**();**// Alert with the

// sound

**}** **else** **{**

textfield**.**getStyleClass**().**removeAll**(**"validation-error"**);**

lbl**.**getStyleClass**().**removeAll**((**"error-font"**));**

lbl**.**setText**(**""**);**

**}**

**}**

**}** **catch** **(**Exception ex**)** **{**

alertMessage**(**ex**.**getMessage**());**

**}**

**break;**

**default:**

**break;**

**}**

**}**

// Alert user with the message

public void alertMessage**(**String alertMessage**)** **{**

Alert alert **=** **new** Alert**(**AlertType**.**ERROR**);**// Display information

// for invalid input

alert**.**setTitle**(**"Error Dialog: Invalid Input"**);**

// String[] message = alertMessage.split(":");

alert**.**setHeaderText**(**alertMessage**);**

alert**.**setContentText**(**"Ooops, there is an error!"**);**

Toolkit**.**getDefaultToolkit**().**beep**();**

alert**.**showAndWait**();**

**}**

**}**

**8.** **searchController.java**

**import** javafx**.**scene**.**control**.\*;**

**import** javafx**.**scene**.**layout**.**BorderPane**;**

public class searchController **{**

CustomTableView customTableView **=** **new** CustomTableView**();**

DataAccess dataAccess **=** **new** DataAccess**();**

private TableView**<**MainModel**>** tableMain **=** **new** TableView**<**MainModel**>();**

// Design Search Form

public void searchStudentGUI**(**BorderPane root**,** ComboBox**<**String**>** cmb**,** TextField txt**)** **throws** Exception **{**

tableMain**.**getColumns**().**addAll**(**customTableView**.**getColumn**(**tableMain**));**

**if** **(**cmb**.**getSelectionModel**().**getSelectedIndex**()** **>** **-**1**)** **{**

tableMain**.**getItems**().**addAll**(**dataAccess**.**searchStudent**(**cmb**.**getValue**(),** txt**.**getText**()));**

**}**

root**.**setCenter**(**tableMain**);**

**}**

**}**

**9.application.css**

**.**TopHBox**{**

-fx-border-color**: rgb(235.0,235.0,235.0) ;**

-fx-border-width**: 1.0 ;**

-fx-border-style**: solid outside ;**

-fx-padding**: 10.0,0.0,0.0,0.0;**

**}**

**.**validation-error **{**

-fx-text-box-border**: #DBB1B1 ;**

-fx-control-inner-background**: #FFF0F0 ;**

-fx-focus-color**: #FF2020 ;**

-fx-faint-focus-color**: #FF202020 ;**

**}**

/\*fill the color of label that displays information for invalid inputs\*/

**.**error-font**{**

-fx-text-fill**: red;**

**}**

/\*Footer color \*/

**.**hBoxBackGroundColor**{**

-fx-background-color**: #336699;**

**}**

**10.DBResource**

user=root

dbUrl = jdbc:mysql://localhost:3306/gradeprocess

**11.DBConnector.java**

**import** java**.**io**.**FileInputStream**;**

**import** java**.**sql**.**Connection**;**

**import** java**.**sql**.**DriverManager**;**

**import** java**.**util**.**Properties**;**

public class DBConnector **{**

public Connection conn**;**

//Connect database from the path provided in DBResource

public DBConnector**()** **throws** Exception **{**

Properties props **=** **new** Properties**();**

props**.**load**(new** FileInputStream**(**"DBResource"**));**

String dbUrl **=** props**.**getProperty**(**"dbUrl"**);**

String user **=** props**.**getProperty**(**"user"**);**

**try** **{**

Class**.**forName**(**"com.mysql.jdbc.Driver"**).**newInstance**();**

conn **=** DriverManager**.**getConnection**(**dbUrl **+** "?useSSL=false"**,** user**,** **null);**

**}** **catch** **(**Exception e**)** **{**

System**.**out**.**println**(**e**);**

**}**

**}**

**}**

**12.DataAccess.java**

**import** java**.**sql**.**CallableStatement**;**

**import** java**.**sql**.**ResultSet**;**

**import** java**.**sql**.**Statement**;**

**import** java**.**util**.**ArrayList**;**

**import** java**.**util**.**List**;**

public class DataAccess **{**

DBConnector db**;**

// Create Table in MySql Database

public boolean createTableName**(**String tableName**)** **throws** Exception **{**

CallableStatement myStmt **=** **null;**

**try** **{**

db **=** **new** DBConnector**();**

myStmt **=** db**.**conn**.**prepareCall**(**"{CALL createTable(?)}"**);**

myStmt**.**setString**(**1**,** tableName**);**

**return** myStmt**.**execute**();**

**}** **finally** **{**

myStmt**.**close**();**

**}**

**}**

// Save the data in table

public List**<**MainModel**>** saveStudent**(**int id**,** String Name**,** Double QMark**,** Double A1Mark**,** Double A2Mark**,** Double A3Mark**,**

Double ExamMark**,** Double Result**,** String Grade**)** **throws** Exception **{**

List**<**MainModel**>** list **=** **new** ArrayList**<>();**

ResultSet myRs **=** **null;**

Statement myStmt **=** **null;**

**try** **{**

db **=** **new** DBConnector**();**

myStmt **=** db**.**conn**.**createStatement**();**

String query **=** "INSERT INTO " **+** TableNameStatic**.**getTableName**()**

**+** " (StudentID,StudentName,QMark,Assignment1,Assignment2,Assignment3,Exam,Results,Grade)VALUES('"

**+** id **+** "','" **+** Name **+** "','" **+** QMark **+** "','" **+** A1Mark **+** "','" **+** A3Mark **+** "','" **+** A3Mark **+** "','"

**+** ExamMark **+** "','" **+** Result **+** "','" **+** Grade **+** "')"**;**

myStmt**.**executeUpdate**(**query**);**

myRs **=** myStmt**.**executeQuery**(**"SELECT \* FROM " **+** TableNameStatic**.**getTableName**());**

**while** **(**myRs**.**next**())** **{**

MainModel mainModel **=** **new** MainModel**(**myRs**.**getInt**(**"id"**),** myRs**.**getInt**(**"StudentID"**),**

myRs**.**getString**(**"StudentName"**),** myRs**.**getDouble**(**"QMark"**),** myRs**.**getDouble**(**"Assignment1"**),**

myRs**.**getDouble**(**"Assignment2"**),** myRs**.**getDouble**(**"Assignment3"**),** myRs**.**getDouble**(**"Exam"**),**

myRs**.**getDouble**(**"Results"**),** myRs**.**getString**(**"Grade"**));**

list**.**add**(**mainModel**);**

**}**

**return** list**;**

**}** **finally** **{**

myRs**.**close**();**

myStmt**.**close**();**

**}**

**}**

// Fetch all student record from database

public List**<**MainModel**>** getAllStudents**()** **throws** Exception **{**

List**<**MainModel**>** list **=** **new** ArrayList**<>();**

Statement myStmt **=** **null;**

ResultSet myRs **=** **null;**

**try** **{**

db **=** **new** DBConnector**();**

myStmt **=** db**.**conn**.**createStatement**();**

myRs **=** myStmt**.**executeQuery**(**"Select \* from " **+** TableNameStatic**.**getTableName**());**

**while** **(**myRs**.**next**())** **{**

MainModel mainModel **=** **new** MainModel**(**myRs**.**getInt**(**"id"**),** myRs**.**getInt**(**"StudentID"**),**

myRs**.**getString**(**"StudentName"**),** myRs**.**getDouble**(**"QMark"**),** myRs**.**getDouble**(**"Assignment1"**),**

myRs**.**getDouble**(**"Assignment2"**),** myRs**.**getDouble**(**"Assignment3"**),** myRs**.**getDouble**(**"Exam"**),**

myRs**.**getDouble**(**"Results"**),** myRs**.**getString**(**"Grade"**));**

list**.**add**(**mainModel**);**

**}**

**return** list**;**

**}** **finally** **{**

myStmt**.**close**();**

myRs**.**close**();**

**}**

**}**

// Search selected data

public List**<**MainModel**>** searchStudent**(**String cmb**,** String searchvalue**)** **throws** Exception **{**

List**<**MainModel**>** list **=** **new** ArrayList**<>();**

Statement myStmt **=** **null;**

ResultSet myRs **=** **null;**

**try** **{**

db **=** **new** DBConnector**();**

myStmt **=** db**.**conn**.**createStatement**();**

myRs **=** myStmt**.**executeQuery**(**

"SELECT \* FROM " **+** TableNameStatic**.**getTableName**()** **+** " WHERE StudentID=" **+** searchvalue**);**

**while** **(**myRs**.**next**())** **{**

MainModel mainModel **=** **new** MainModel**(**myRs**.**getInt**(**"id"**),** myRs**.**getInt**(**"StudentID"**),**

myRs**.**getString**(**"StudentName"**),** myRs**.**getDouble**(**"QMark"**),** myRs**.**getDouble**(**"Assignment1"**),**

myRs**.**getDouble**(**"Assignment2"**),** myRs**.**getDouble**(**"Assignment3"**),** myRs**.**getDouble**(**"Exam"**),**

myRs**.**getDouble**(**"Results"**),** myRs**.**getString**(**"Grade"**));**

list**.**add**(**mainModel**);**

**}**

**return** list**;**

**}** **finally** **{**

myStmt**.**close**();**

myRs**.**close**();**

**}**

**}**

// Update the student record

public void updateStudentDetails**(**int idStudent**,** int studentID**,** String Name**,** Double qMark**,** Double A1**,** Double A2**,**

Double A3**,** Double ExamMark**,** Double Result**,** String Grade**)** **throws** Exception **{**

Statement myStmt **=** **null;**

**try** **{**

db **=** **new** DBConnector**();**

myStmt **=** db**.**conn**.**createStatement**();**

String Query **=** "UPDATE " **+** TableNameStatic**.**getTableName**()** **+** " SET StudentID='" **+** studentID

**+** "',StudentName='" **+** Name **+** "',QMark='" **+** qMark **+** "',Assignment1='" **+** A1 **+** "',Assignment2='" **+** A2

**+** "',Assignment3='" **+** A3 **+** "',Exam='" **+** ExamMark **+** "',Results='" **+** Result **+** "',Grade='" **+** Grade

**+** "' where id='" **+** idStudent **+** "'"**;**

myStmt**.**executeUpdate**(**Query**);**

**}** **finally** **{**

myStmt**.**close**();**

**}**

**}**

**}**

**13.Image/Icon Used**

****java

* add
* search

4.infinite.jpg



**Database Schema**

-- MySQL dump 10.13 Distrib 5.7.9, for Win64 (x86\_64)

--

-- Host: 127.0.0.1 Database: gradeprocess

-- ------------------------------------------------------

-- Server version 5.7.9-log

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/**;**

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/**;**

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/**;**

/\*!40101 SET NAMES utf8 \*/**;**

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/**;**

/\*!40103 SET TIME\_ZONE='+00:00' \*/**;**

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/**;**

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/**;**

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/**;**

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/**;**

--

-- Table structure for table `itc521`

--

**DROP** **TABLE** **IF** **EXISTS** `itc521`**;**

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/**;**

/\*!40101 SET character\_set\_client = utf8 \*/**;**

**CREATE** **TABLE** `itc521` **(**

`id` **int(**11**)** unsigned **NOT** **NULL** AUTO\_INCREMENT**,**

`StudentID` **int(**11**)** **NOT** **NULL,**

`StudentName` **varchar(**50**)** **NOT** **NULL,**

`QMark` **double** **NOT** **NULL,**

`Assignment1` **double** **NOT** **NULL,**

`Assignment2` **double** **NOT** **NULL,**

`Assignment3` **double** **NOT** **NULL,**

`Exam` **double** **NOT** **NULL,**

`Results` **double** **NOT** **NULL,**

`Grade` **varchar(**50**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**)**

**)** ENGINE**=**InnoDB AUTO\_INCREMENT**=**10 **DEFAULT** CHARSET**=**utf8**;**

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/**;**

--

-- Dumping data for table `itc521`

--

**LOCK** TABLES `itc521` **WRITE;**

/\*!40000 ALTER TABLE `itc521` DISABLE KEYS \*/**;**

**INSERT** **INTO** `itc521` **VALUES** **(**6**,**11111111**,**'Bikash Katwal'**,**99**,**99**,**99**,**99**,**99**,**99**,**'HD'**),(**7**,**11111112**,**'Tom Cruise 1'**,**87**,**98**,**95**,**87**,**65**,**79.25**,**'DI'**),(**8**,**10000111**,**'Angelina Jolie'**,**99**,**66**,**88**,**88**,**68**,**73.05**,**'CR'**),(**9**,**12345678**,**'Susmita'**,**88**,**7**,**55**,**55**,**98**,**70.95**,**'CR'**);**

/\*!40000 ALTER TABLE `itc521` ENABLE KEYS \*/**;**

UNLOCK TABLES**;**

--

-- Dumping events for database 'gradeprocess'

--

--

-- Dumping routines for database 'gradeprocess'

--

/\*!50003 DROP PROCEDURE IF EXISTS `createTable` \*/**;**

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ **;**

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ **;**

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ **;**

/\*!50003 SET character\_set\_client = utf8 \*/ **;**

/\*!50003 SET character\_set\_results = utf8 \*/ **;**

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ **;**

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ **;**

/\*!50003 SET sql\_mode = 'STRICT\_TRANS\_TABLES,NO\_AUTO\_CREATE\_USER,NO\_ENGINE\_SUBSTITUTION' \*/ **;**

DELIMITER **;;**

**CREATE** DEFINER**=**`root`@`localhost` **PROCEDURE** `createTable`**(IN** tableName **varchar(**50**))**

**BEGIN**

**SET** @tableName**=**tableName**;**

**SET** @createquery**=CONCAT(**'

CREATE TABLE IF NOT EXISTS ' **,** @tableName**,** ' (

id INT(11) UNSIGNED NOT NULL AUTO\_INCREMENT,

StudentID INT NOT NULL,

StudentName VARCHAR(50) NOT NULL,

QMark DOUBLE NOT NULL,

Assignment1 DOUBLE NOT NULL,

Assignment2 DOUBLE NOT NULL,

Assignment3 DOUBLE NOT NULL,

Exam DOUBLE NOT NULL,

Results DOUBLE NOT NULL,

Grade VARCHAR(50) NOT NULL,

PRIMARY KEY (id)

)

'**);**

**PREPARE** stmt **FROM** @createquery**;**

**EXECUTE** stmt**;**

**DEALLOCATE** **PREPARE** stmt**;**

**SELECT** **TRUE;**

**END** **;;**

DELIMITER **;**

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ **;**

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ **;**

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ **;**

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ **;**

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/**;**

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/**;**

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/**;**

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/**;**

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/**;**

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/**;**

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/**;**

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/**;**

-- Dump completed on 2016-10-07 0:02:49