Course work 1(PP2)

K.A.D.S.T. Kumarapeli

2019952

W1790958

Contents

I used separate classes for each calculator, home window, Database, components and helping views. You are not allowed to input any minus values from this calculator and its better to use virtual keyboard for input data. There are separate help windows for each calculator and referring it before use better to get dome idea about this application. There is feedback area to user to feel free to enter their user experience and it will be helpful for further developing.

Home.java

```
public class Home extends Application {
    //variables for buttons and components
    public static Button btnFD;
    public static Button btnSavings;
    public static Button btnMortgage;
    public static Button btnLoan;
    public static Label lblHome;
    public static Button btnFDName;
    public static Button btnSavingsName;
    public static Button btnMortgageName;
    public static Button btnLoanName;
    public static Button btnClse;
    public static Button btnHelp;
    public static Button btnFeedback;
    //format decimal 2 numbers
    public static DecimalFormat df2 = new DecimalFormat("0.00");

    //home window calling
    @Override
    public void start(Stage primaryStage) {
        //calling method
        homeWindow();
    }

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

    public static void homeWindow() {
        //creating stage and disable title bar
        Stage primaryStage = new Stage();
        primaryStage.initStyle(StageStyle.UNDECORATED);

        //adding images and give sizes and (x,y)
        Image image1 = new Image("Pics/HomeBG.jpg");
        ImageView homeBG = new ImageView();
        homeBG.setImage(image1);
        homeBG.setFitWidth(609);
        homeBG.setFitHeight(416);

        Image imageGif = new Image("Pics/hmm.gif");
        ImageView homeBGGif = new ImageView();
        homeBGGif.setImage(imageGif);
        homeBGGif.setFitWidth(609);
        homeBGGif.setFitHeight(416);
        homeBGGif.setOpacity(.8);

        Image image2 = new Image("Pics/Dixed Deposit.png");
        ImageView btnFDPng = new ImageView();
        btnFDPng.setImage(image2);
        btnFDPng.setFitWidth(82);
        btnFDPng.setFitHeight(84);

        Image image3 = new Image("Pics/savings.png");
        ImageView btnSavingsPng = new ImageView();
        btnSavingsPng.setImage(image3);
        btnSavingsPng.setFitWidth(82);
        btnSavingsPng.setFitHeight(90);

        Image image4 = new Image("Pics/Mortgage.png");
        ImageView btnMortgagePng = new ImageView();
        btnMortgagePng.setImage(image4);
        btnMortgagePng.setFitWidth(82);
        btnMortgagePng.setFitHeight(84);

        Image image5 = new Image("Pics/loan.png");
        ImageView btnLoanPng = new ImageView();
        btnLoanPng.setImage(image5);
        btnLoanPng.setFitWidth(82);
        btnLoanPng.setFitHeight(84);

        Image image6 = new Image("Pics/Feedback.png");
        ImageView btnFedbk = new ImageView();
        btnFedbk.setImage(image6);
        btnFedbk.setFitWidth(40);
        btnFedbk.setFitHeight(40);

        Image image7 = new Image("Pics/Financial cal.png");
        ImageView nameImg = new ImageView();
        nameImg.setImage(image7);
```

```
nameImg.setFitWidth(200);
nameImg.setFitHeight(100);
nameImg.setLayoutX(190);
nameImg.setLayoutY(-15);
nameImg.setOpacity(0.5);

//Creating buttons from components class and giving parameter values
btnFD = Components.creatingButton("", 125., 63., 90., 82.);
btnFD.setGraphic(btnFDPng);
btnSavings = Components.creatingButton("", 357., 63., 90., 82.);
btnSavings.setGraphic(btnSavingsPng);
btnMortgage = Components.creatingButton("", 125., 222., 90., 82.);
btnMortgage.setGraphic(btnMortgagePng);
btnLoan = Components.creatingButton("", 357., 222., 90., 82.);
btnLoan.setGraphic(btnLoanPng);
btnClse = Components.createBtnClose();
btnHelp = Components.createBtnHelp();
btnFeedback = Components.createBtnHelp();
btnFeedback.setGraphic(btnFedbk);
btnFeedback.setLayoutY(350);
btnFeedback.setLayoutX(548);

//label for dark apperaence for home window
lblHome = Components.creatingLabelForBackground();
lblHome.setOpacity(0.65);
//creating labels for each buttons and set style for them
btnFDName = Components.creatingButton("Fixed Deposit", 74., 155., 50., 200.);
btnFDName.setStyle("-fx-background-color:transparent; -fx-text-fill:#f5f7f7;-fx-font-size:1.5em;");
btnSavingsName = Components.creatingButton("Savings", 306., 155., 50., 200.);
btnSavingsName.setStyle("-fx-background-color:transparent; -fx-text-fill:#f5f7f7;-fx-font-size:1.5em;");
btnMortgageName = Components.creatingButton("Mortgage", 74., 312., 50., 200.);
btnMortgageName.setStyle("-fx-background-color:transparent; -fx-text-fill:#f5f7f7;-fx-font-size:1.5em;");
btnLoanName = Components.creatingButton("Loan", 306., 312., 50., 200.);
btnLoanName.setStyle("-fx-background-color:transparent; -fx-text-fill:#f5f7f7;-fx-font-size:1.5em;");
//adding all elements in to homepane in relevant oder
Pane homePane = new Pane();
homePane.getChildren().add(homeBG);
homePane.getChildren().add(homeBGGif);
homePane.getChildren().add(lblHome);
homePane.getChildren().add(btnFeedback);
homePane.getChildren().add(nameImg);
homePane.getChildren().add(btnFDName);
homePane.getChildren().add(btnSavingsName);
homePane.getChildren().add(btnMortgageName);
homePane.getChildren().add(btnLoanName);
homePane.getChildren().add(btnFD);
homePane.getChildren().add(btnSavings);
homePane.getChildren().add(btnMortgage);
homePane.getChildren().add(btnLoan);
homePane.getChildren().add(btnClse);
homePane.getChildren().add(btnHelp);

//giving action methods for each buttons in homepage
btnFD.setOnAction(e -> {
    primaryStage.close();
    FixDeposit.fixDepositWindow();
});
btnSavings.setOnAction(e -> {
    primaryStage.close();
    Savings.savingsWindow();
});
btnMortgage.setOnAction(e -> {
    primaryStage.close();
    Mortgage.mortgageWindow();

});
btnLoan.setOnAction(e -> {
    primaryStage.close();
    Loans.loansWindow();

});
btnClse.setOnAction(e -> {
    primaryStage.close();
});
btnFeedback.setOnAction(e -> {
    primaryStage.close();
    Help.feedbackWindow();
});
btnFeedback.setOnAction(e -> {
    primaryStage.close();
    Help.feedbackWindow();
});
btnHelp.setOnAction(e -> {
    primaryStage.close();
    Help.helpWindow();
});
primaryStage.setScene(new Scene(homePane, 600, 400));
primaryStage.show();
```

```
    }
}
```

Components.java

```
public class Components {
    //all the useful components created in this class using static methods , for avoid code duplication
    //method for creating buttons
    public static Button creatingButton(String txtOnBtn, Double x, Double y, Double prefHeight, Double setPrefWidth) {
        Button btn = new Button(txtOnBtn);
        btn.setLayoutX(x);
        btn.setLayoutY(y);
        btn.setPrefHeight(prefHeight);
        btn.setPrefWidth(setPrefWidth);
        btn.setStyle("-fx-background-color:transparent; -fx-border-color:#F0F8FF;-fx-text-fill:#f5f7f7;-fx-font-size:2em;-fx-
border-radius:20");
        return btn;
    }

    //method for creating labels
    public static Label creatingLabel(String txtOnLbl, Double x, Double y, Double prefHeight, Double PrefWidth) {
        Label lbl = new Label(txtOnLbl);
        lbl.setLayoutX(x);
        lbl.setLayoutY(y);
        lbl.setPrefHeight(prefHeight);
        lbl.setPrefWidth(PrefWidth);
        lbl.setStyle("-fx-text-fill:#ffffff");
        lbl.setStyle("-fx-background-color:transparent;-fx-text-fill:#f5f7f7;-fx-font-size:1em;-fx-font-weight: Bold;");
        return lbl;

    }

    //creating label for get dark appearence in all windows
    public static Label creatingLabelForBackground() {
        Label lblBG = new Label();
        lblBG.setLayoutX(0);
        lblBG.setLayoutY(0);
        lblBG.setPrefHeight(500);
        lblBG.setPrefWidth(700);
        lblBG.setStyle("-fx-background-color: #000000;");
        lblBG.setOpacity(0.81);
        return lblBG;
    }

    //method for creating text fields
    public static TextField creatingTextField(String promptTxt, Double x, Double y, Double prefHeight, Double PrefWidth) {
        TextField txtField = new TextField();
        txtField.setPromptText(promptTxt);
        txtField.setLayoutX(x);
        txtField.setLayoutY(y);
        txtField.setPrefWidth(PrefWidth);
        txtField.setPrefHeight(prefHeight);
        txtField.setStyle("-fx-background-color:transparent; -fx-border-color:#B0C4DE;-fx-text-fill:#f5f7f7;-fx-font-
size:1em;-fx-border-radius:10");
        return txtField;
    }

    //method for create back button, back button appears in most of the windows and this will avoid code duplication
    public static Button createBtnBack() {
        Image imageBack = new Image("Pics/back.png");
        ImageView backPng = new ImageView();
        backPng.setImage(imageBack);
        backPng.setFitWidth(30);
        backPng.setFitHeight(30);
        Button btnBack = new Button();
        btnBack.setLayoutX(0);
        btnBack.setLayoutY(0);
        btnBack.setPrefHeight(20);
        btnBack.setPrefWidth(30);
        btnBack.setGraphic(backPng);
        btnBack.setStyle("-fx-background-color:transparent; -fx-background-radius:100;");
        btnBack.setOpacity(0.81);
        return btnBack;
    }

    //method for create close button, close button appears in all windows and this will avoid code duplication
    public static Button createBtnClose() {
        Image imageClose = new Image("Pics/Close.png");
        ImageView closePng = new ImageView();
        closePng.setImage(imageClose);
        closePng.setFitWidth(30);
        closePng.setFitHeight(29);

        Button btnClose = new Button();
        btnClose.setLayoutX(560);
        btnClose.setLayoutY(0);
        btnClose.setPrefHeight(19);
        btnClose.setPrefWidth(30);
        btnClose.setGraphic(closePng);
        btnClose.setStyle("-fx-background-color:transparent; -fx-background-radius:100;");
        btnClose.setOpacity(0.81);
```

```
        return btnClose;
    }

    //method for creating help button
    public static Button createBtnHelp() {
        Image imageHelp = new Image("Pics/help.png");
        ImageView helpPng = new ImageView();
        helpPng.setImage(imageHelp);
        helpPng.setFitWidth(40);
        helpPng.setFitHeight(40);

        Button btnClose = new Button();
        btnClose.setLayoutX(2);
        btnClose.setLayoutY(350);
        btnClose.setPrefHeight(19);
        btnClose.setPrefWidth(30);
        btnClose.setGraphic(helpPng);
        btnClose.setStyle("-fx-background-color:transparent; -fx-background-radius:100;");
        btnClose.setOpacity(0.81);
        return btnClose;
    }

    //method for keyboard keys, all keyboard keys wii created in this method
    public static Button btnForKeyBoard(String txtOnBtn, Double x, Double y, Double prefHeight, Double setPrefWidth) {
        Button key = new Button(txtOnBtn);
        key.setLayoutX(x);
        key.setLayoutY(y);
        key.setPrefHeight(prefHeight);
        key.setPrefWidth(setPrefWidth);
        key.setStyle("-fx-background-color:transparent;-fx-border-color:#66CDAA;-fx-text-fill:#f5f7f7;-fx-font-size:2em;-fx-
border-radius:20;");
        return key;

    }

    //method for keyboard, this method can be called with (x,y) values
    public static AnchorPane keyBoard(Double x, Double y) {
        //setting anchor pain and creating buttons and add all into that created anchor pain
        AnchorPane keyBoardPane = new AnchorPane();
        keyBoardPane.setStyle("-fx-border-color: green; -fx-border-width: 0px 0px 0px 1px");
        keyBoardPane.setPrefWidth(234);
        keyBoardPane.setPrefHeight(303);
        keyBoardPane.setLayoutX(x);
        keyBoardPane.setLayoutY(y);
        Button key0 = btnForKeyBoard("0", 36., 210., 46., 89.);
        Button key1 = btnForKeyBoard("1", 35., 164., 46., 45.);
        Button key2 = btnForKeyBoard("2", 80., 164., 46., 45.);
        Button key3 = btnForKeyBoard("3", 125., 164., 46., 45.);
        Button key4 = btnForKeyBoard("4", 35., 118., 46., 45.);
        Button key5 = btnForKeyBoard("5", 80., 118., 46., 45.);
        Button key6 = btnForKeyBoard("6", 125., 118., 46., 45.);
        Button key7 = btnForKeyBoard("7", 35., 72., 46., 45.);
        Button key8 = btnForKeyBoard("8", 80., 72., 46., 45.);
        Button key9 = btnForKeyBoard("9", 125., 72., 46., 45.);
        Button keyClearAll = btnForKeyBoard("C", 170., 164., 46., 45.);
        Button keyPoint = btnForKeyBoard(".", 125., 210., 46., 45.);
        Button keyGO = btnForKeyBoard("G\nO", 170., 73., 90., 45.);
        Button keyCE = btnForKeyBoard("CE", 170., 210., 46., 45.);
        TextField txtKeyBoard = creatingTextField("Enter Value here", 30., 25., 25., 191.);
        txtKeyBoard.setStyle("-fx-background-color:transparent;-fx-border-color:#66CDAA;-fx-text-fill:#f5f7f7;-fx-font-
size:1em;-fx-border-radius:10");
        keyBoardPane.getChildren().add(key0);
keyBoardPane.getChildren().add(key1);
keyBoardPane.getChildren().add(key2);
keyBoardPane.getChildren().add(key3);
keyBoardPane.getChildren().add(key4);
keyBoardPane.getChildren().add(key5);
keyBoardPane.getChildren().add(key6);
keyBoardPane.getChildren().add(key7);
keyBoardPane.getChildren().add(key8);
keyBoardPane.getChildren().add(key9);
keyBoardPane.getChildren().add(keyPoint);
keyBoardPane.getChildren().add(keyClearAll);
keyBoardPane.getChildren().add(keyGO);
keyBoardPane.getChildren().add(keyCE);
keyBoardPane.getChildren().add(txtKeyBoard);
//giving action events for all buttons
key0.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "0");
    }
});
key1.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "1");
    }
});
```

```
key2.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "2");
    }
});
key3.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "3");
    }
});
key4.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "4");
    }
});
key5.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "5");
    }
});
key6.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "6");
    }
});
key7.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "7");
    }
});
key8.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "8");
    }
});
key9.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + "9");
    }
});
keyPoint.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText(txtKeyBoard.getText() + ".");
    }
});
keyCE.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        String number = txtKeyBoard.getText();
        if (!(number.equals(""))) {
            txtKeyBoard.setText(number.substring(0, (number.length() - 1)));
        }
    }
});
keyClearAll.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        txtKeyBoard.setText("");

    }
});
//this will help to find the selected text field in each calculator
keyGO.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        if (FixDeposit.selectedTxtDA == true) {
            FixDeposit.txtDepositAmnt.setText(txtKeyBoard.getText());
        } else if (FixDeposit.selectedTxRate == true) {
            FixDeposit.txtIRate.setText(txtKeyBoard.getText());
        } else if (FixDeposit.selectedTxtTime == true) {
            FixDeposit.txtTime.setText(txtKeyBoard.getText());
        } else if (FixDeposit.selectedTxtFV == true) {
            FixDeposit.txtFtureValue.setText(txtKeyBoard.getText());
        } else if (Loans.selectedTxtLoanAmnt == true) {
            Loans.txtLoanAmnt.setText(txtKeyBoard.getText());
        } else if (Loans.selectedTxtIntrstRate == true) {
            Loans.txtIntrstRate.setText(txtKeyBoard.getText());
        } else if (Loans.selectedTxtLoanPeriod == true) {
            Loans.txtLoanPeriod.setText(txtKeyBoard.getText());
        } else if (Loans.selectedTxtMonthlyPayment == true) {
```

```
                Loans.txtMonthlyPayment.setText(txtKeyBoard.getText());
            } else if (Loans.selectedTxtPaybackAmnt == true) {
                Loans.txtPaybackAmnt.setText(txtKeyBoard.getText());
            } else if (Mortgage.selectedTxtMortgageAmnt == true) {
                Mortgage.txtMortgageAmnt.setText(txtKeyBoard.getText());
            } else if (Mortgage.selectedTxtMortgageIRate == true) {
                Mortgage.txtMortgageIRate.setText(txtKeyBoard.getText());
            } else if (Mortgage.selectedTxtMortgagePeriod == true) {
                Mortgage.txtMortgagePeriod.setText(txtKeyBoard.getText());
            } else if (Mortgage.selectedTxtMonthlyPayment == true) {
                Mortgage.txtMonthlyPayment.setText(txtKeyBoard.getText());
            } else if (Mortgage.selectedTxtDownPayment == true) {
                Mortgage.txtDownPayment.setText(txtKeyBoard.getText());
            } else if (Savings.selectedTxtSavingsAmnt == true) {
                Savings.txtSavingsAmnt.setText(txtKeyBoard.getText());
            } else if (Savings.selectedTxtSavingsIRate == true) {
                Savings.txtSavingsIRate.setText(txtKeyBoard.getText());
            } else if (Savings.selectedTxtSavingsPeriod == true) {
                Savings.txtSavingsPeriod.setText(txtKeyBoard.getText());
            } else if (Savings.selectedTxtSavingsPayment == true) {
                Savings.txtSavingsMonthlyPayment.setText(txtKeyBoard.getText());
            } else if (Savings.selectedTxtSavingsFv == true) {
                Savings.txtSavingsFValue.setText(txtKeyBoard.getText());
            }
        }
    });

    return keyBoardPane;
    }

    //method for information alert
    public static Alert information(String message) {
        Alert infoAlert = new Alert(Alert.AlertType.NONE);
        infoAlert.setAlertType(Alert.AlertType.INFORMATION);
        infoAlert.setContentText(message);
        infoAlert.showAndWait();
        return infoAlert;
    }

    //method for error alert and this will pop up with a sound
    public static Alert error(String message) {
        Toolkit.getDefaultToolkit().beep();
        Alert errorAlert = new Alert(Alert.AlertType.NONE);
        errorAlert.setAlertType(Alert.AlertType.ERROR);
        errorAlert.setContentText(message);
        errorAlert.showAndWait();
        return errorAlert;
    }

    //if input values are less than 0 this will popup with a sound
    public static Alert errorInput() {
        Toolkit.getDefaultToolkit().beep();
        Alert errorAlert = new Alert(Alert.AlertType.NONE);
        errorAlert.setAlertType(Alert.AlertType.ERROR);
        errorAlert.setContentText("Please make sure not to  enter (-) values as inputs");
        errorAlert.showAndWait();
        return errorAlert;
    }


}
```

FixDeposit.java

```
public class FixDeposit {
    //variables for calculations , validations and components
    public static Button btnClse;
    public static Button btnBk;
    public static Button btnCalculateFix;
    public static Button btnHelp;
    public static Label lblDpsitAmnt;
    public static Label lblIntrstRate;
    public static Label lblTime;
    public static Label lblFtureValue;
    public static TextField txtDepositAmnt;
    public static TextField txtIRate;
    public static TextField txtTime;
    public static TextField txtFtureValue;
    public static AnchorPane keyBoard;
    public static Label lblFD;
    private static double pV, fV, r, t;
    public static boolean selectedTxtDA, selectedTxRate, selectedTxtTime, selectedTxtFV;
    private static boolean validatedData = false;

    //connection with database and creating collection for add data
    private static MongoCollection getCollection() {
        MongoClient connectDB = new databaseInitialization().connectingDB();
        MongoDatabase database = connectDB.getDatabase("FinancialCalculator");
        MongoCollection mongoCollection = database.getCollection("FixDeposit");
        return mongoCollection;
    }

    //creating Fd window
    public static void fixDepositWindow() {
        Stage fDStage = new Stage();
        fDStage.initStyle(StageStyle.UNDECORATED);

        //adding image to background
        Image image1 = new Image("Pics/FDBG.jpg");
        ImageView FDBg = new ImageView();
        FDBg.setImage(image1);
        FDBg.setFitWidth(600);
        FDBg.setFitHeight(420);
        //creating labels, buttons
        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        btnHelp = Components.createBtnHelp();
        btnCalculateFix = Components.creatingButton("Calculate", 210., 308., 25., 100.);
        btnCalculateFix.setStyle("-fx-background-color:transparent; -fx-border-color:#F0F8FF;-fx-text-fill:#f5f7f7;-fx-font-
size:1.5em;-fx-border-radius:10;");


        lblDpsitAmnt = Components.creatingLabel("Deposit Amount (P)", 42., 97., 25., 149.);
        lblIntrstRate = Components.creatingLabel("Interest Rate (r)", 42., 148., 25., 149.);
        lblTime = Components.creatingLabel("Time (t)", 42., 200., 25., 149.);
        lblFtureValue = Components.creatingLabel("Future Value (A)", 42., 252., 25., 149.);

        txtDepositAmnt = Components.creatingTextField("LKR", 191., 97., 25., 149.);
        txtIRate = Components.creatingTextField("% (Monthly)", 191., 148., 25., 149.);
        txtTime = Components.creatingTextField("Years", 191., 200., 25., 149.);
        txtFtureValue = Components.creatingTextField("LKR", 191., 252., 25., 149.);

        //reading fiddeposit collection to load previous data into text fields
        Document lastInsert = new MongoClient().getDatabase("FinancialCalculator").getCollection("FixDeposit").find().sort(new
BasicDBObject("_id", -1)).first();
        if (lastInsert != null) {

            txtDepositAmnt.setText(String.valueOf(lastInsert.get("Present Value (LKR)")));
            txtIRate.setText(String.valueOf(lastInsert.get("Interest")));
            txtTime.setText(String.valueOf(lastInsert.get("Time Period (years)")));
            txtFtureValue.setText(String.valueOf(lastInsert.get("Future Value (LKR)")));

        }

        lblFD = Components.creatingLabelForBackground();


        keyBoard = Components.keyBoard(352., 72.);
        //adding all components into anchorpain
        Pane fDPane = new Pane();
        fDPane.getChildren().add(FDBg);
        fDPane.getChildren().add(lblFD);
        fDPane.getChildren().add(lblDpsitAmnt);
        fDPane.getChildren().add(lblIntrstRate);
        fDPane.getChildren().add(lblTime);
        fDPane.getChildren().add(lblFtureValue);
        fDPane.getChildren().add(btnClse);
        fDPane.getChildren().add(btnBk);
        fDPane.getChildren().add(btnCalculateFix);
        fDPane.getChildren().add(btnHelp);
```

```
fDPane.getChildren().add(txtDepositAmnt);
fDPane.getChildren().add(txtIRate);
fDPane.getChildren().add(txtTime);
fDPane.getChildren().add(txtFtureValue);
fDPane.getChildren().add(keyBoard);

//setting action methods to all buttons
btnBk.setOnAction(e -> {
    fDStage.close();
    homeWindow();
});
btnClse.setOnAction(e -> {
    fDStage.close();

});
btnCalculateFix.setOnAction(e -> {
    calculate();

});
btnHelp.setOnAction(e -> {
    Help.helpWindowFix();

});
//boolean condition to select the selected textfield in this fd calculator
txtDepositAmnt.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtDA = true;
        selectedTxRate = false;
        selectedTxtTime = false;
        selectedTxtFV = false;
    }
});
txtIRate.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtDA = false;
        selectedTxRate = true;
        selectedTxtTime = false;
        selectedTxtFV = false;
    }
});
txtTime.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtDA = false;
        selectedTxRate = false;
        selectedTxtTime = true;
        selectedTxtFV = false;
    }
});
txtFtureValue.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtDA = false;
        selectedTxRate = false;
        selectedTxtTime = false;
        selectedTxtFV = true;
    }
});
fDStage.setScene(new Scene(fDPane, 600, 400));
fDStage.show();
    }

    //method for calculate button
    public static void calculate() {
        try { //compound interest is calculated by monthly
            //checking condition to get empty textfield
            if (txtDepositAmnt.getText().equals("") && !txtIRate.getText().equals("") && !txtTime.getText().equals("") &&
!txtFtureValue.getText().equals("")) {
                //getting values and convert them into doubble
                r = Double.parseDouble(txtIRate.getText());
                t = Double.parseDouble(txtTime.getText());
                fV = Double.parseDouble(txtFtureValue.getText());
                //condition to check (-) values, if user enters (-) values calculation will not be execute
                if (r > 0 && t > 0 && fV > 0) {
                    pV = fV / (Math.pow((1 + r / 1200), 12 * t));
                    //set output into relevent textfield and display information alert
                    txtDepositAmnt.setText(df2.format(pV));
                    Components.information("Present value is :  " + df2.format(pV));
                    //validated input for save into database
                    validatedData = true;
                    //error message
                } else {
                    Components.errorInput();
                }
            } else if (!txtDepositAmnt.getText().equals("") && txtIRate.getText().equals("") && !txtTime.getText().equals("")
```

```
&& !txtFtureValue.getText().equals("")) {
                pV = Double.parseDouble(txtDepositAmnt.getText());
                t = Double.parseDouble(txtTime.getText());
                fV = Double.parseDouble(txtFtureValue.getText());
                if (pV > 0 && t > 0 && fV > 0) {
                    r = 100 * (12 * (Math.pow((fV / pV), 1 / (12 * t)) - 1));
                    txtIRate.setText(df2.format(r));
                    Components.information("Interest Rate is :  " + df2.format(r));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else if (!txtDepositAmnt.getText().equals("") && !txtIRate.getText().equals("") && txtTime.getText().equals("")
&& !txtFtureValue.getText().equals("")) {
                pV = Double.parseDouble(txtDepositAmnt.getText());
                r = Double.parseDouble(txtIRate.getText());
                fV = Double.parseDouble(txtFtureValue.getText());
                if (pV > 0 && r > 0 && fV > 0) {
                    t = (Math.log(fV / pV)) / (12 * Math.log(1 + (r / 1200)));
                    txtTime.setText(df2.format(t));
                    Components.information("Duration  is :  " + df2.format(t) + " years");
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else if (!txtDepositAmnt.getText().equals("") && !txtIRate.getText().equals("") && !txtTime.getText().equals("")
&& txtFtureValue.getText().equals("")) {
                pV = Double.parseDouble(txtDepositAmnt.getText());
                r = Double.parseDouble(txtIRate.getText());
                t = Double.parseDouble(txtTime.getText());
                if (pV > 0 && r > 0 && t > 0) {
                    fV = pV * (Math.pow((1 + r / 1200), 12 * t));
                    txtFtureValue.setText(df2.format(fV));
                    Components.information("Future Value is :  " + df2.format(fV));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else {
                //if user filled all textfields this error message wil popup
                Components.error("Please make sure to Empty one text Field which you want to calculate");
            }
        } catch (Exception e) {
            //if user enter string as an input this message will popup
            Components.error("Please make sure the entered data is valid");
        }
        //if there be a validated input it will save into database in relevant
        if (validatedData == true) {
            //creating document and adding validated data into it
            Document fdData = new Document();
            fdData.put("Present Value (LKR)", pV);
            fdData.put("Interest", r);
            fdData.put("Time Period (years)", t);
            fdData.put("Future Value (LKR)", fV);
            getCollection().insertOne(fdData);
            validatedData = false;
        }


    }
}
```

Loans.java

```
public class Loans {
    public static Button btnClse;
    public static Button btnBk;
    public static Button btnCalculateLoan;
    public static Button btnHelp;
    public static AnchorPane keyBoard;
    public static Label lblLoan;
    public static Label lblLoanAmnt;
    public static Label lblIntrstRate;
    public static Label lblLoanPeriod;
    public static Label lblMonthlyPayment;
    public static Label lblPaybackAmnt;
    public static TextField txtLoanAmnt;
    public static TextField txtIntrstRate;
    public static TextField txtLoanPeriod;
    public static TextField txtMonthlyPayment;
    public static TextField txtPaybackAmnt;
    public static boolean selectedTxtLoanAmnt, selectedTxtIntrstRate, selectedTxtLoanPeriod, selectedTxtMonthlyPayment,
selectedTxtPaybackAmnt;
    private static double mP, T, lA, r;
    private static boolean validatedData = false;

    private static MongoCollection getCollection() {
        MongoClient connectDB = new databaseInitialization().connectingDB();
        MongoDatabase database = connectDB.getDatabase("FinancialCalculator");
        MongoCollection mongoCollection = database.getCollection("Loans");
        return mongoCollection;

    }


    public static void loansWindow() {
        Stage loanStage = new Stage();
        loanStage.initStyle(StageStyle.UNDECORATED);

        Image image1 = new Image("Pics/LoansBG.jpg");
        ImageView loanBG = new ImageView();
        loanBG.setImage(image1);
        loanBG.setFitWidth(600);
        loanBG.setFitHeight(420);


        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        btnHelp = Components.createBtnHelp();
        btnCalculateLoan = Components.creatingButton("Calculate", 210., 308., 25., 100.);
        btnCalculateLoan.setStyle("-fx-background-color:transparent; -fx-border-color:#F0F8FF;-fx-text-fill:#f5f7f7;-fx-font-
size:1.5em;-fx-border-radius:10");


        lblLoanAmnt = Components.creatingLabel("Loan Amount", 42., 50., 25., 149.);
        lblIntrstRate = Components.creatingLabel("Interest Rate", 42., 97., 25., 149.);
        lblLoanPeriod = Components.creatingLabel("Loan period", 42., 148., 25., 149.);
        lblMonthlyPayment = Components.creatingLabel("Monthly Payment", 42., 200., 25., 149.);
        lblPaybackAmnt = Components.creatingLabel("Payback Amount", 42., 252., 25., 149.);

        txtLoanAmnt = Components.creatingTextField("LKR", 191., 50., 25., 149.);
        txtIntrstRate = Components.creatingTextField("%", 191., 97., 25., 149.);
        txtLoanPeriod = Components.creatingTextField("months", 191., 148., 25., 149.);
        txtMonthlyPayment = Components.creatingTextField("LKR", 191., 200., 25., 149.);
        txtPaybackAmnt = Components.creatingTextField("LKR", 191., 252., 25., 149.);

        Document lastInsert = new MongoClient().getDatabase("FinancialCalculator").getCollection("Loans").find().sort(new
BasicDBObject("_id", -1)).first();
        if (lastInsert != null) {

            txtLoanAmnt.setText(String.valueOf(lastInsert.get("Loan Amount (LKR)")));
            txtIntrstRate.setText(String.valueOf(lastInsert.get("Interest")));
            txtLoanPeriod.setText(String.valueOf(lastInsert.get("Time Period (months)")));
            txtMonthlyPayment.setText(String.valueOf(lastInsert.get("Monthly Payment (LKR)")));
            txtPaybackAmnt.setText(String.valueOf(lastInsert.get("Payback Amount (LKR)")));

        }

        lblLoan = Components.creatingLabelForBackground();


        keyBoard = Components.keyBoard(352., 30.);


        Pane loanPane = new Pane();
        loanPane.getChildren().add(loanBG);
        loanPane.getChildren().add(lblLoan);
        loanPane.getChildren().add(lblLoanAmnt);
        loanPane.getChildren().add(lblIntrstRate);
        loanPane.getChildren().add(lblLoanPeriod);
        loanPane.getChildren().add(lblMonthlyPayment);
```

```
loanPane.getChildren().add(lblPaybackAmnt);
loanPane.getChildren().add(btnClse);
loanPane.getChildren().add(btnBk);
loanPane.getChildren().add(btnCalculateLoan);
loanPane.getChildren().add(btnHelp);
loanPane.getChildren().add(txtLoanAmnt);
loanPane.getChildren().add(txtIntrstRate);
loanPane.getChildren().add(txtLoanPeriod);
loanPane.getChildren().add(txtMonthlyPayment);
loanPane.getChildren().add(txtPaybackAmnt);
loanPane.getChildren().add(keyBoard);


btnBk.setOnAction(e -> {
    loanStage.close();
    homeWindow();
});
btnClse.setOnAction(e -> {
    loanStage.close();
});
btnCalculateLoan.setOnAction(e -> {
    calculate();

});
btnHelp.setOnAction(e -> {
    Help.helpWindowLoan();

});

txtLoanAmnt.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtLoanAmnt = true;
        selectedTxtIntrstRate = false;
        selectedTxtLoanPeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtPaybackAmnt = false;
    }
});
txtIntrstRate.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtLoanAmnt = false;
        selectedTxtIntrstRate = true;
        selectedTxtLoanPeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtPaybackAmnt = false;
    }
});
txtLoanPeriod.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtLoanAmnt = false;
        selectedTxtIntrstRate = false;
        selectedTxtLoanPeriod = true;
        selectedTxtMonthlyPayment = false;
        selectedTxtPaybackAmnt = false;
    }
});
txtMonthlyPayment.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtLoanAmnt = false;
        selectedTxtIntrstRate = false;
        selectedTxtLoanPeriod = false;
        selectedTxtMonthlyPayment = true;
        selectedTxtPaybackAmnt = false;
    }
});
txtPaybackAmnt.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtLoanAmnt = false;
        selectedTxtIntrstRate = false;
        selectedTxtLoanPeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtPaybackAmnt = true;
    }
});


loanStage.setScene(new Scene(loanPane, 600, 400));
loanStage.show();
}


public static void calculate() {
    try {
        if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") &&
```

```
!txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("")) {
                lA = Double.parseDouble(txtLoanAmnt.getText());
                r = Double.parseDouble(txtIntrstRate.getText());
                T = Double.parseDouble(txtLoanPeriod.getText());
                if (lA > 0 && r > 0 && T > 0) {
                    double i = r / 1200;
                    mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
                    txtMonthlyPayment.setText(df2.format(mP));
                    txtPaybackAmnt.setText(df2.format(mP * T));
                    Components.information("Monthly Payment :  " + df2.format(mP) + "\n" + "Payback Amount  :  " +
df2.format(mP * T));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }

            } else if (txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") &&
!txtLoanPeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("")) {
                r = Double.parseDouble(txtIntrstRate.getText());
                T = Double.parseDouble(txtLoanPeriod.getText());
                mP = Double.parseDouble(txtMonthlyPayment.getText());
                if (r > 0 && T > 0 && mP > 0) {
                    double i = r / 1200;
                    lA = (mP / i) * (1 - (1 / (Math.pow((1 + i), T))));
                    txtLoanAmnt.setText(df2.format(lA));
                    txtPaybackAmnt.setText(df2.format(mP * T));
                    Components.information("Loan Amount  :  " + df2.format(lA) + "\n" + "Payback Amount  :  " + df2.format(mP
* T));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }

            } else if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") &&
txtLoanPeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("")) {
                lA = Double.parseDouble(txtLoanAmnt.getText());
                r = Double.parseDouble(txtIntrstRate.getText());
                mP = Double.parseDouble(txtMonthlyPayment.getText());
                double i = r / 1200;
                if (lA > 0 && r > 0 && mP > 0) {
                    T = ((Math.log((mP / i) / ((mP / i) - lA))) / (Math.log(1 + i)));
                    txtLoanPeriod.setText(df2.format(T));
                    txtPaybackAmnt.setText(df2.format(mP * T));
                    Components.information("Time Period  :  " + df2.format(T) + "\n" + "Payback Amount  :  " + df2.format(mP *
T));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }

            } else if (!txtLoanAmnt.getText().equals("") && txtIntrstRate.getText().equals("") &&
!txtLoanPeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("") || !txtPaybackAmnt.getText().equals("")) {
                Components.error("You are Not allowed to get this output in ths Version");

            } else if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") &&
!txtLoanPeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("") && txtPaybackAmnt.getText().equals("")) {
                lA = Double.parseDouble(txtLoanAmnt.getText());
                r = Double.parseDouble(txtIntrstRate.getText());
                mP = Double.parseDouble(txtMonthlyPayment.getText());
                T = Double.parseDouble(txtLoanPeriod.getText());
                if (lA > 0 && r > 0 && mP > 0 && T > 0) {
                    txtPaybackAmnt.setText(df2.format(mP * T));
                    Components.information("Payback Amount :  " + df2.format(mP * T));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }

            } else {
                Components.error("Please make sure to Empty one text Field which you want to calculate");
            }
        } catch (Exception e) {
            Components.error("Please make sure the entered data is valid");

        }
        if (validatedData == true) {
            Document loanData = new Document();
            loanData.put("Loan Amount (LKR)", lA);
            loanData.put("Interest", r);
            loanData.put("Time Period (months)", T);
            loanData.put("Monthly Payment (LKR)", mP);
            loanData.put("Payback Amount (LKR)", mP * T);
            getCollection().insertOne(loanData);
            validatedData = false;
        }
    }


}
```

Mortgage.java

```
public class Mortgage {
    public static Button btnClse;
    public static Button btnBk;
    public static Button btnCalculateMortgage;
    public static Button btnHelp;
    public static Label lblMortgage;
    public static Label lblMortgageAmnt;
    public static Label lblMortgageIRate;
    public static Label lalMortgagePeriod;
    public static Label lblMonthlyPayment;
    public static Label lblDownpayment;
    public static TextField txtMortgageAmnt;
    public static TextField txtMortgageIRate;
    public static TextField txtMortgagePeriod;
    public static TextField txtMonthlyPayment;
    public static TextField txtDownPayment;
    public static AnchorPane keyBoard;
    public static boolean selectedTxtMortgageAmnt, selectedTxtMortgageIRate, selectedTxtMortgagePeriod,
selectedTxtMonthlyPayment, selectedTxtDownPayment;
    private static double mP, T, mA, r, dP = 0;
    private static boolean validatedData = false;

    private static MongoCollection getCollection() {
        MongoClient connectDB = new databaseInitialization().connectingDB();
        MongoDatabase database = connectDB.getDatabase("FinancialCalculator");
        MongoCollection mongoCollection = database.getCollection("Mortgage");
        return mongoCollection;

    }


    public static void mortgageWindow() {
        Stage mortgageStage = new Stage();
        mortgageStage.initStyle(StageStyle.UNDECORATED);

        Image image1 = new Image("Pics/MortgageBG.jpg");
        ImageView mortBG = new ImageView();
        mortBG.setImage(image1);
        mortBG.setFitWidth(600);
        mortBG.setFitHeight(420);

        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        btnHelp = Components.createBtnHelp();
        btnCalculateMortgage = Components.creatingButton("Calculate", 210., 308., 25., 100.);
        btnCalculateMortgage.setStyle("-fx-background-color:transparent; -fx-border-color:#F0F8FF;-fx-text-fill:#f5f7f7;-fx-
font-size:1.5em;-fx-border-radius:10");

        lblMortgageAmnt = Components.creatingLabel("Mortgage Amount", 42., 50., 25., 149.);
        lblMortgageIRate = Components.creatingLabel("Interest Rate", 42., 97., 25., 149.);
        lalMortgagePeriod = Components.creatingLabel("Mortgage period", 42., 148., 25., 149.);
        lblMonthlyPayment = Components.creatingLabel("Monthly Payment", 42., 200., 25., 149.);
        lblDownpayment = Components.creatingLabel("DownPayment", 42., 252., 25., 149.);

        txtMortgageAmnt = Components.creatingTextField("LKR", 191., 50., 25., 149.);
        txtMortgageIRate = Components.creatingTextField("%", 191., 97., 25., 149.);
        txtMortgagePeriod = Components.creatingTextField("months", 191., 148., 25., 149.);
        txtMonthlyPayment = Components.creatingTextField("LKR", 191., 200., 25., 149.);
        txtDownPayment = Components.creatingTextField("LKR", 191., 252., 25., 149.);

        Document lastInsert = new MongoClient().getDatabase("FinancialCalculator").getCollection("Mortgage").find().sort(new
BasicDBObject("_id", -1)).first();
        if (lastInsert != null) {

            txtMortgageAmnt.setText(String.valueOf(lastInsert.get("Mortgage Amount (LKR)")));
            txtMortgageIRate.setText(String.valueOf(lastInsert.get("Interest")));
            txtMortgagePeriod.setText(String.valueOf(lastInsert.get("Time Period (months)")));
            txtMonthlyPayment.setText(String.valueOf(lastInsert.get("Monthly Payment (LKR)")));
            txtDownPayment.setText(String.valueOf(lastInsert.get("Downpayment (LKR)")));

        }

        lblMortgage = Components.creatingLabelForBackground();


        keyBoard = Components.keyBoard(352., 30.);

        Pane mortgagePane = new Pane();
        mortgagePane.getChildren().add(mortBG);
        mortgagePane.getChildren().add(lblMortgage);
        mortgagePane.getChildren().add(btnClse);
        mortgagePane.getChildren().add(btnBk);
        mortgagePane.getChildren().add(btnHelp);
        mortgagePane.getChildren().add(btnCalculateMortgage);
        mortgagePane.getChildren().add(lblMortgageAmnt);
        mortgagePane.getChildren().add(lblMortgageIRate);
        mortgagePane.getChildren().add(lalMortgagePeriod);
```

```
mortgagePane.getChildren().add(lblMonthlyPayment);
mortgagePane.getChildren().add(lblDownpayment);
mortgagePane.getChildren().add(txtMortgageAmnt);
mortgagePane.getChildren().add(txtMortgageIRate);
mortgagePane.getChildren().add(txtMortgagePeriod);
mortgagePane.getChildren().add(txtMonthlyPayment);
mortgagePane.getChildren().add(txtDownPayment);
mortgagePane.getChildren().add(keyBoard);


btnBk.setOnAction(e -> {
    mortgageStage.close();
    homeWindow();
});
btnClse.setOnAction(e -> {
    mortgageStage.close();
});
btnCalculateMortgage.setOnAction(e -> {
    calculate();
});
btnHelp.setOnAction(e -> {
    Help.helpWindowMortgage();
});
txtMortgageAmnt.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtMortgageAmnt = true;
        selectedTxtMortgageIRate = false;
        selectedTxtMortgagePeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtDownPayment = false;
    }
});
txtMortgageIRate.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtMortgageAmnt = false;
        selectedTxtMortgageIRate = true;
        selectedTxtMortgagePeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtDownPayment = false;
    }
});
txtMortgagePeriod.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtMortgageAmnt = false;
        selectedTxtMortgageIRate = false;
        selectedTxtMortgagePeriod = true;
        selectedTxtMonthlyPayment = false;
        selectedTxtDownPayment = false;
    }
});
txtMonthlyPayment.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtMortgageAmnt = false;
        selectedTxtMortgageIRate = false;
        selectedTxtMortgagePeriod = false;
        selectedTxtMonthlyPayment = true;
        selectedTxtDownPayment = false;
    }
});
txtDownPayment.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtMortgageAmnt = false;
        selectedTxtMortgageIRate = false;
        selectedTxtMortgagePeriod = false;
        selectedTxtMonthlyPayment = false;
        selectedTxtDownPayment = true;
    }
});


mortgageStage.setScene(new Scene(mortgagePane, 600, 400));
mortgageStage.show();


}

public static void calculate() {

    try {
        if (!txtMortgageAmnt.getText().equals("") && !txtMortgageIRate.getText().equals("") &&
!txtMortgagePeriod.getText().equals("") && txtMonthlyPayment.getText().equals("")) {
            if (txtDownPayment.getText().equals("")) {
                mA = Double.parseDouble(txtMortgageAmnt.getText());
                r = Double.parseDouble(txtMortgageIRate.getText());
```

```
                    T = Double.parseDouble(txtMortgagePeriod.getText());
                    double i = r / 1200;
                    if (mA > 0 && r > 0 && T > 0) {
                        mP = ((mA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
                        txtMonthlyPayment.setText(df2.format(mP));
                        Components.information("Monthly Payment :  " + df2.format(mP) + "\n" + "Payback Amount :  " +
df2.format(mP * T));
                        validatedData = true;
                    } else {
                        Components.errorInput();
                    }
                } else {
                    mA = Double.parseDouble(txtMortgageAmnt.getText());
                    r = Double.parseDouble(txtMortgageIRate.getText());
                    T = Double.parseDouble(txtMortgagePeriod.getText());
                    dP = Double.parseDouble(txtDownPayment.getText());
                    double i = r / 1200;
                    if (mA > 0 && r > 0 && T > 0 && dP > 0) {
                        mP = (((mA - dP) * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
                        txtMonthlyPayment.setText(df2.format(mP));
                        Components.information("Monthly Payment :  " + df2.format(mP) + "\n" + "Payback Amount with
downpayment :  " + df2.format((mP * T) + dP));
                        validatedData = true;
                    } else {
                        Components.errorInput();
                    }
                }
            } else if (txtMortgageAmnt.getText().equals("") && !txtMortgageIRate.getText().equals("") &&
!txtMortgagePeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("")) {
                if (txtDownPayment.getText().equals("")) {
                    r = Double.parseDouble(txtMortgageIRate.getText());
                    T = Double.parseDouble(txtMortgagePeriod.getText());
                    mP = Double.parseDouble(txtMonthlyPayment.getText());
                    double i = r / 1200;
                    if (r > 0 && T > 0 && mP > 0) {
                        mA = ((mP / i) * (1 - (1 / (Math.pow((1 + i), T)))));
                        txtMortgageAmnt.setText(df2.format(mA));
                        Components.information("Mortgage Amount :  " + df2.format(mA) + "\n" + "Payback Amount :  " +
df2.format(mP * T));
                        validatedData = true;
                    } else {
                        Components.errorInput();
                    }
                } else {
                    r = Double.parseDouble(txtMortgageIRate.getText());
                    T = Double.parseDouble(txtMortgagePeriod.getText());
                    dP = Double.parseDouble(txtDownPayment.getText());
                    mP = Double.parseDouble(txtMonthlyPayment.getText());
                    double i = r / 1200;
                    if (r > 0 && T > 0 && dP > 0 && mP > 0) {
                        mA = (((mP / i) * (1 - (1 / (Math.pow((1 + i), T))))) + dP);
                        txtMortgageAmnt.setText(df2.format(mA));
                        Components.information("Mortgage Amount :  " + df2.format(mA) + "\n" + "Payback Amount with
downpayment :  " + df2.format((mP * T) + dP));
                        validatedData = true;
                    } else {
                        Components.errorInput();
                    }

                }
            } else if (!txtMortgageAmnt.getText().equals("") && !txtMortgageIRate.getText().equals("") &&
txtMortgagePeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("")) {
                if (txtDownPayment.getText().equals("")) {
                    r = Double.parseDouble(txtMortgageIRate.getText());
                    mA = Double.parseDouble(txtMortgageAmnt.getText());
                    mP = Double.parseDouble(txtMonthlyPayment.getText());
                    double i = r / 1200;
                    if (r > 0 && mA > 0 && mP > 00) {
                        T = ((Math.log((mP / i) / ((mP / i) - mA)) / (Math.log(1 + i))));
                        txtMortgagePeriod.setText(df2.format(T));
                        Components.information("Mortgage Period :  " + df2.format(T) + "\n" + "Payback Amount :  " +
df2.format(mP * T));
                        validatedData = true;
                    } else {
                        Components.errorInput();
                    }

                } else {
                    r = Double.parseDouble(txtMortgageIRate.getText());
                    mA = Double.parseDouble(txtMortgageAmnt.getText());
                    mP = Double.parseDouble(txtMonthlyPayment.getText());
                    dP = Double.parseDouble(txtDownPayment.getText());
                    double i = r / 1200;
                    if (r > 0 && mA > 0 && mP > 0 && dP > 0) {
                        T = ((Math.log((mP / i) / ((mP / i) - (mA - dP)))) / (Math.log(1 + i)));
                        txtMortgagePeriod.setText(df2.format(T));
                        Components.information("Mortgage Period :  " + df2.format(T) + "\n" + "Payback Amount with downpayment
:  " + df2.format((mP * T) + dP));
                        validatedData = true;
```

```
                    } else {
                        Components.errorInput();
                    }

                }

            } else if (!txtMortgageAmnt.getText().equals("") && !txtMortgageIRate.getText().equals("") &&
!txtMortgagePeriod.getText().equals("") && !txtMonthlyPayment.getText().equals("") && txtDownPayment.getText().equals("")) {
                r = Double.parseDouble(txtMortgageIRate.getText());
                mA = Double.parseDouble(txtMortgageAmnt.getText());
                mP = Double.parseDouble(txtMonthlyPayment.getText());
                T = Double.parseDouble(txtMortgagePeriod.getText());
                double i = r / 1200;
                if (r > 0 && mA > 0 && mP > 0 && T > 0) {
                    dP = mA - ((mP / i) * (1 - (1 / (Math.pow((1 + i), T)))));
                    txtDownPayment.setText(df2.format(dP));
                    Components.information("Mortgage downpayment :  " + df2.format(dP) + "\n" + "Payback Amount with
downpayment :  " + df2.format((mP * T) + dP));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else {
                Components.error("Please make sure to Empty one text Field which you want to calculate");
            }
        } catch (Exception e) {

            Components.error("Please make sure the entered data is valid");
        }
        if (validatedData == true) {
            Document mortgageData = new Document();
            mortgageData.put("Mortgage Amount (LKR)", mA);
            mortgageData.put("Interest", r);
            mortgageData.put("Time Period (months)", T);
            mortgageData.put("Monthly Payment (LKR)", mP);
            mortgageData.put("Downpayment (LKR)", dP);
            getCollection().insertOne(mortgageData);
            validatedData = false;
        }
    }
}
```

Savings.java

```
public class Savings {
    public static Button btnClse;
    public static Button btnBk;
    public static Button btnCalculateSavings;
    public static Button btnHelp;
    public static AnchorPane keyBoard;
    public static Label lblSavingsAmnt;
    public static Label lblSavingsIRate;
    public static Label lalSavingsPeriod;
    public static Label lblSavingsMonthlyPayment;
    public static Label lblSavingsFValue;
    public static Label lblSavings;
    public static TextField txtSavingsAmnt;
    public static TextField txtSavingsIRate;
    public static TextField txtSavingsPeriod;
    public static TextField txtSavingsMonthlyPayment;
    private static double pV, fV, r, t, mP;
    public static TextField txtSavingsFValue;
    public static boolean selectedTxtSavingsAmnt, selectedTxtSavingsIRate, selectedTxtSavingsPeriod,
selectedTxtSavingsPayment, selectedTxtSavingsFv;
    private static boolean validatedData = false;

    private static MongoCollection getCollection() {
        MongoClient connectDB = new databaseInitialization().connectingDB();
        MongoDatabase database = connectDB.getDatabase("FinancialCalculator");
        MongoCollection mongoCollection = database.getCollection("Savings");
        return mongoCollection;

    }

    public static void savingsWindow() {
        Stage savingsStage = new Stage();
        savingsStage.initStyle(StageStyle.UNDECORATED);

        Image image1 = new Image("Pics/SavingsBg.jpg");
        ImageView savingsBg = new ImageView();
        savingsBg.setImage(image1);
        savingsBg.setFitWidth(600);
        savingsBg.setFitHeight(420);

        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        btnHelp = Components.createBtnHelp();
        btnCalculateSavings = Components.creatingButton("Calculate", 210., 308., 25., 100.);
        btnCalculateSavings.setStyle("-fx-background-color:transparent; -fx-border-color:#F0F8FF;-fx-text-fill:#f5f7f7;-fx-
font-size:1.5em;-fx-border-radius:10;");

        lblSavingsAmnt = Components.creatingLabel("Savings Amount", 42., 50., 25., 149.);
        lblSavingsIRate = Components.creatingLabel("Interest Rate", 42., 97., 25., 149.);
        lalSavingsPeriod = Components.creatingLabel("Savings period", 42., 148., 25., 149.);
        lblSavingsMonthlyPayment = Components.creatingLabel("Monthly Payment", 42., 200., 25., 149.);
        lblSavingsFValue = Components.creatingLabel("Future Value", 42., 252., 25., 149.);

        txtSavingsAmnt = Components.creatingTextField("LKR", 191., 50., 25., 149.);
        txtSavingsIRate = Components.creatingTextField("%", 191., 97., 25., 149.);
        txtSavingsPeriod = Components.creatingTextField("Years", 191., 148., 25., 149.);
        txtSavingsMonthlyPayment = Components.creatingTextField("LKR", 191., 200., 25., 149.);
        txtSavingsFValue = Components.creatingTextField("LKR", 191., 252., 25., 149.);

        Document lastInsert = new MongoClient().getDatabase("FinancialCalculator").getCollection("Savings").find().sort(new
BasicDBObject("_id", -1)).first();
        if (lastInsert != null) {

            txtSavingsAmnt.setText(String.valueOf(lastInsert.get("Loan Amount (LKR)")));
            txtSavingsIRate.setText(String.valueOf(lastInsert.get("Interest")));
            txtSavingsPeriod.setText(String.valueOf(lastInsert.get("Time Period (years)")));
            txtSavingsMonthlyPayment.setText(String.valueOf(lastInsert.get("Monthly Payment (LKR)")));
            txtSavingsFValue.setText(String.valueOf(lastInsert.get("Future Value (LKR)")));

        }


        lblSavings = Components.creatingLabelForBackground();
        lblSavings.setOpacity(.5);
        keyBoard = Components.keyBoard(352., 30.);

        Pane savingsPain = new Pane();
        savingsPain.getChildren().add(savingsBg);
        savingsPain.getChildren().add(lblSavings);
        savingsPain.getChildren().add(btnClse);
        savingsPain.getChildren().add(btnBk);
        savingsPain.getChildren().add(btnHelp);
        savingsPain.getChildren().add(btnCalculateSavings);
        savingsPain.getChildren().add(lblSavingsAmnt);
        savingsPain.getChildren().add(lblSavingsIRate);
        savingsPain.getChildren().add(lalSavingsPeriod);
        savingsPain.getChildren().add(lblSavingsMonthlyPayment);
```

```
savingsPain.getChildren().add(lblSavingsFValue);
savingsPain.getChildren().add(txtSavingsAmnt);
savingsPain.getChildren().add(txtSavingsIRate);
savingsPain.getChildren().add(txtSavingsPeriod);
savingsPain.getChildren().add(txtSavingsMonthlyPayment);
savingsPain.getChildren().add(txtSavingsFValue);
savingsPain.getChildren().add(keyBoard);


btnBk.setOnAction(e -> {
    savingsStage.close();
    homeWindow();
});
btnClse.setOnAction(e -> {
    savingsStage.close();

});
btnCalculateSavings.setOnAction(e -> {
    Savings.calculate();

});
btnHelp.setOnAction(e -> {
    Help.helpWindowSavings();

});
txtSavingsAmnt.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtSavingsAmnt = true;
        selectedTxtSavingsIRate = false;
        selectedTxtSavingsPeriod = false;
        selectedTxtSavingsPayment = false;
        selectedTxtSavingsFv = false;
    }
});
txtSavingsIRate.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtSavingsAmnt = false;
        selectedTxtSavingsIRate = true;
        selectedTxtSavingsPeriod = false;
        selectedTxtSavingsPayment = false;
        selectedTxtSavingsFv = false;
    }
});
txtSavingsPeriod.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtSavingsAmnt = false;
        selectedTxtSavingsIRate = false;
        selectedTxtSavingsPeriod = true;
        selectedTxtSavingsPayment = false;
        selectedTxtSavingsFv = false;
    }
});
txtSavingsMonthlyPayment.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtSavingsAmnt = false;
        selectedTxtSavingsIRate = false;
        selectedTxtSavingsPeriod = false;
        selectedTxtSavingsPayment = true;
        selectedTxtSavingsFv = false;
    }
});
txtSavingsFValue.setOnMouseClicked(new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent event) {
        selectedTxtSavingsAmnt = false;
        selectedTxtSavingsIRate = false;
        selectedTxtSavingsPeriod = false;
        selectedTxtSavingsPayment = false;
        selectedTxtSavingsFv = true;
    }
});


savingsStage.setScene(new Scene(savingsPain, 600, 400));
savingsStage.show();


}

public static void calculate() {
    try {
        if (txtSavingsAmnt.getText().equals("") && !txtSavingsIRate.getText().equals("") &&
!txtSavingsPeriod.getText().equals("") && !txtSavingsMonthlyPayment.getText().equals("") &&
!txtSavingsFValue.getText().equals("")) {
            r = Double.parseDouble(txtSavingsIRate.getText());
```

```
                t = Double.parseDouble(txtSavingsPeriod.getText());
                fV = Double.parseDouble(txtSavingsFValue.getText());
                mP = Double.parseDouble(txtSavingsMonthlyPayment.getText());
                if (r > 0 && t > 0 && fV > 0 && mP > 0) {
                    double pow = Math.pow((1 + (r / 1200)), (12 * t));
                    pV = (fV - (mP * ((pow - 1) / (r / 1200)))) / (pow);
                    txtSavingsAmnt.setText(df2.format(pV));
                    Components.information("Present value is :  " + df2.format(pV));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else if (!txtSavingsAmnt.getText().equals("") && !txtSavingsIRate.getText().equals("") &&
txtSavingsPeriod.getText().equals("") && !txtSavingsMonthlyPayment.getText().equals("") &&
!txtSavingsFValue.getText().equals("")) {
                r = Double.parseDouble(txtSavingsIRate.getText());
                fV = Double.parseDouble(txtSavingsFValue.getText());
                mP = Double.parseDouble(txtSavingsMonthlyPayment.getText());
                if (r > 0 && fV > 0 && mP > 0) {
                    t = Math.log(((fV * r) / (1200 * mP)) + 1) / (12 * Math.log(1 + (r / 1200)));
                    txtSavingsPeriod.setText(df2.format(t));
                    Components.information("Duration  is :  " + df2.format(t) + " months");
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else if (!txtSavingsAmnt.getText().equals("") && !txtSavingsIRate.getText().equals("") &&
!txtSavingsPeriod.getText().equals("") && txtSavingsMonthlyPayment.getText().equals("") &&
!txtSavingsFValue.getText().equals("")) {
                r = Double.parseDouble(txtSavingsIRate.getText());
                t = Double.parseDouble(txtSavingsPeriod.getText());
                fV = Double.parseDouble(txtSavingsFValue.getText());
                pV = Double.parseDouble(txtSavingsAmnt.getText());
                if (r > 0 && t > 0 && fV > 0 && pV > 0) {
                    double pow = Math.pow((1 + (r / 1200)), (12 * t));
                    mP = (fV - pV * (pow)) / ((pow - 1) / (r / 1200));
                    txtSavingsMonthlyPayment.setText(df2.format(mP));
                    Components.information("Monthly Payment is :  " + df2.format(mP));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }
            } else if (!txtSavingsAmnt.getText().equals("") && !txtSavingsIRate.getText().equals("") &&
!txtSavingsPeriod.getText().equals("") && !txtSavingsMonthlyPayment.getText().equals("") &&
txtSavingsFValue.getText().equals("")) {
                r = Double.parseDouble(txtSavingsIRate.getText());
                mP = Double.parseDouble(txtSavingsMonthlyPayment.getText());
                t = Double.parseDouble(txtSavingsPeriod.getText());
                pV = Double.parseDouble(txtSavingsAmnt.getText());
                if (r > 0 && mP > 0 && t > 0 && pV > 0) {
                    double pow = Math.pow((1 + (r / 1200)), (12 * t));
                    fV = (mP * ((pow - 1) / (r / 1200))) + pV * (pow);
                    txtSavingsFValue.setText(df2.format(fV));
                    Components.information("Future Value is :  " + df2.format(fV));
                    validatedData = true;
                } else {
                    Components.errorInput();
                }

            } else if (!txtSavingsAmnt.getText().equals("") && txtSavingsIRate.getText().equals("") &&
!txtSavingsPeriod.getText().equals("") && !txtSavingsMonthlyPayment.getText().equals("") &&
!txtSavingsFValue.getText().equals("")) {
                Components.error("You are Not allowed to get this output in ths Version");
            } else {
                Components.error("Please make sure to Empty one text Field which you want to calculate");
            }
        } catch (Exception e) {
            Components.error("Please make sure the entered data is valid");
        }
        if (validatedData == true) {
            Document savingsData = new Document();
            savingsData.put("Loan Amount (LKR)", pV);
            savingsData.put("Interest", r);
            savingsData.put("Time Period (years)", t);
            savingsData.put("Monthly Payment (LKR)", mP);
            savingsData.put("Future Value (LKR)", fV);
            getCollection().insertOne(savingsData);
            validatedData = false;
        }
    }


}
```

Help.java

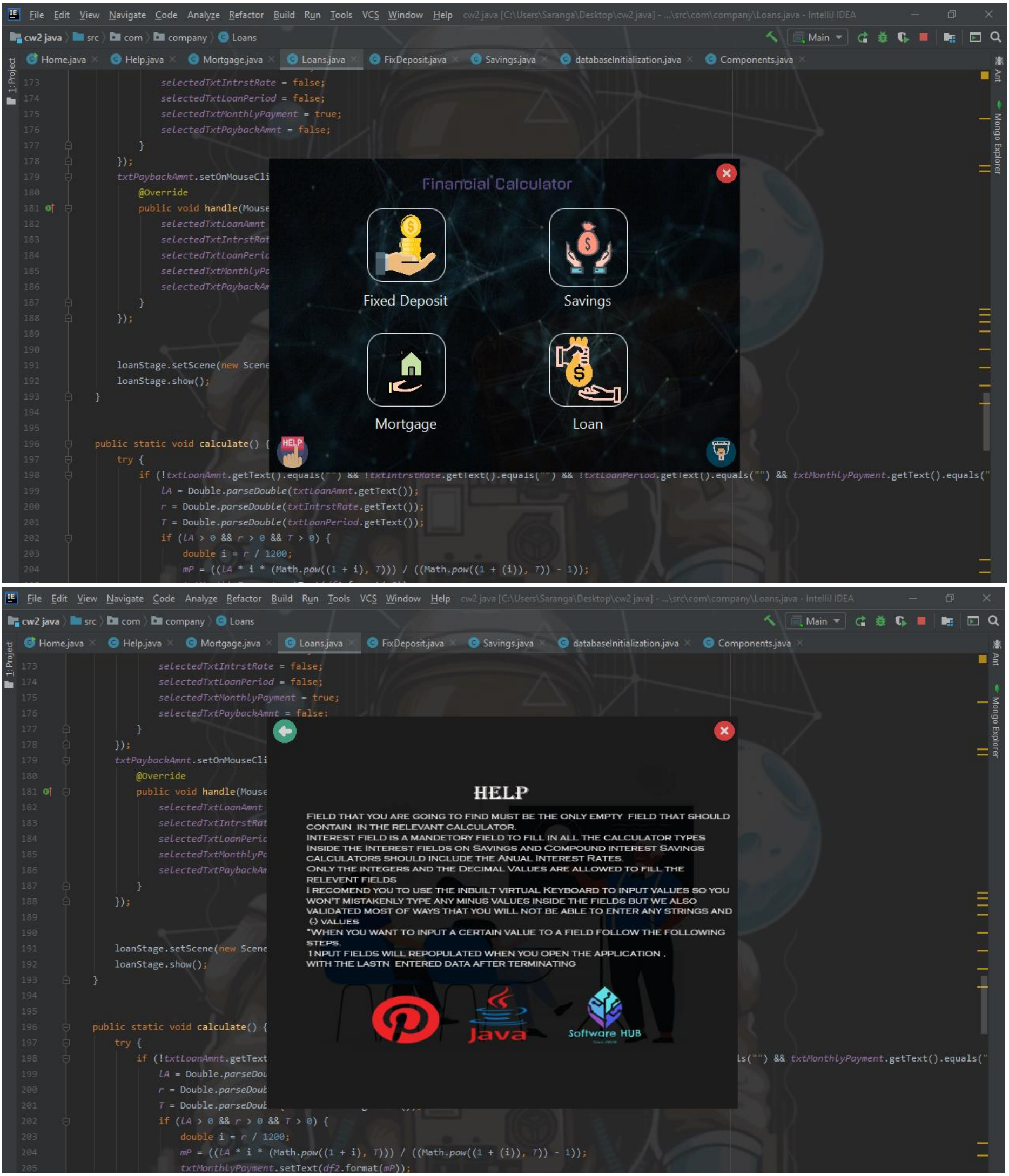
```
public class Help {
    public static Label lblFeedbackBG;
    public static TextField txtFeeback;
    public static Button btnClse;
    public static Button btnBk;
    public static Button btnSubmit;

    private static MongoCollection getCollection() {
        MongoClient connectDB = new databaseInitialization().connectingDB();
        MongoDatabase database = connectDB.getDatabase("FinancialCalculator");
        MongoCollection mongoCollection = database.getCollection("Feedback");
        return mongoCollection;
    }

    //method to get feedback from users , , creating window and adding components and input feedback details into database
    public static void feedbackWindow() {
        Stage fdbkStage = new Stage();
        fdbkStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/feedback.jpg");
        ImageView bgfd = new ImageView();
        bgfd.setImage(image1);
        bgfd.setFitWidth(609);
        bgfd.setFitHeight(500);
        lblFeedbackBG = Components.creatingLabelForBackground();
        txtFeeback = Components.creatingTextField("your text here", 50., 50., 200., 450.);
        btnSubmit = Components.creatingButton("Submit", 228., 320., 25., 100.);
        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        Pane feedbackPane = new Pane();
        feedbackPane.getChildren().add(bgfd);
        feedbackPane.getChildren().add(lblFeedbackBG);
        feedbackPane.getChildren().add(txtFeeback);
        feedbackPane.getChildren().add(btnSubmit);
        feedbackPane.getChildren().add(btnClse);
        feedbackPane.getChildren().add(btnBk);
        btnBk.setOnAction(e -> {
            fdbkStage.close();
            Home.homeWindow();
        });
        btnClse.setOnAction(e -> {
            fdbkStage.close();
        });
        btnSubmit.setOnAction(e -> {
            submitting();
        });
        fdbkStage.setScene(new Scene(feedbackPane, 600, 400));
        fdbkStage.show();
    }

    //method for submit button
    public static void submitting() {
        String fv = txtFeeback.getText();
        Help.getCollection();
        Document feedbkData = new Document();
        feedbkData.put("Present Value (LKR)", fv);
        getCollection().insertOne(feedbkData);
        Components.information("Feedback Submitted");
        txtFeeback.clear();
    }

    //methods for all help windows in each calculators
    public static void helpWindow() {
        Stage helpStage = new Stage();
        helpStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/help.jpg");
        ImageView bghelp = new ImageView();
        bghelp.setImage(image1);
        bghelp.setFitWidth(609);
        bghelp.setFitHeight(500);
        btnClse = Components.createBtnClose();
        btnBk = Components.createBtnBack();
        Pane helpPane = new Pane();
        helpPane.getChildren().add(bghelp);
        helpPane.getChildren().add(btnClse);
        helpPane.getChildren().add(btnBk);
        btnBk.setOnAction(e -> {
            helpStage.close();
            Home.homeWindow();
        });
        btnClse.setOnAction(e -> {
            helpStage.close();
        });
        helpStage.setScene(new Scene(helpPane, 600, 500));
        helpStage.show();
    }

    public static void helpWindowFix() {
```

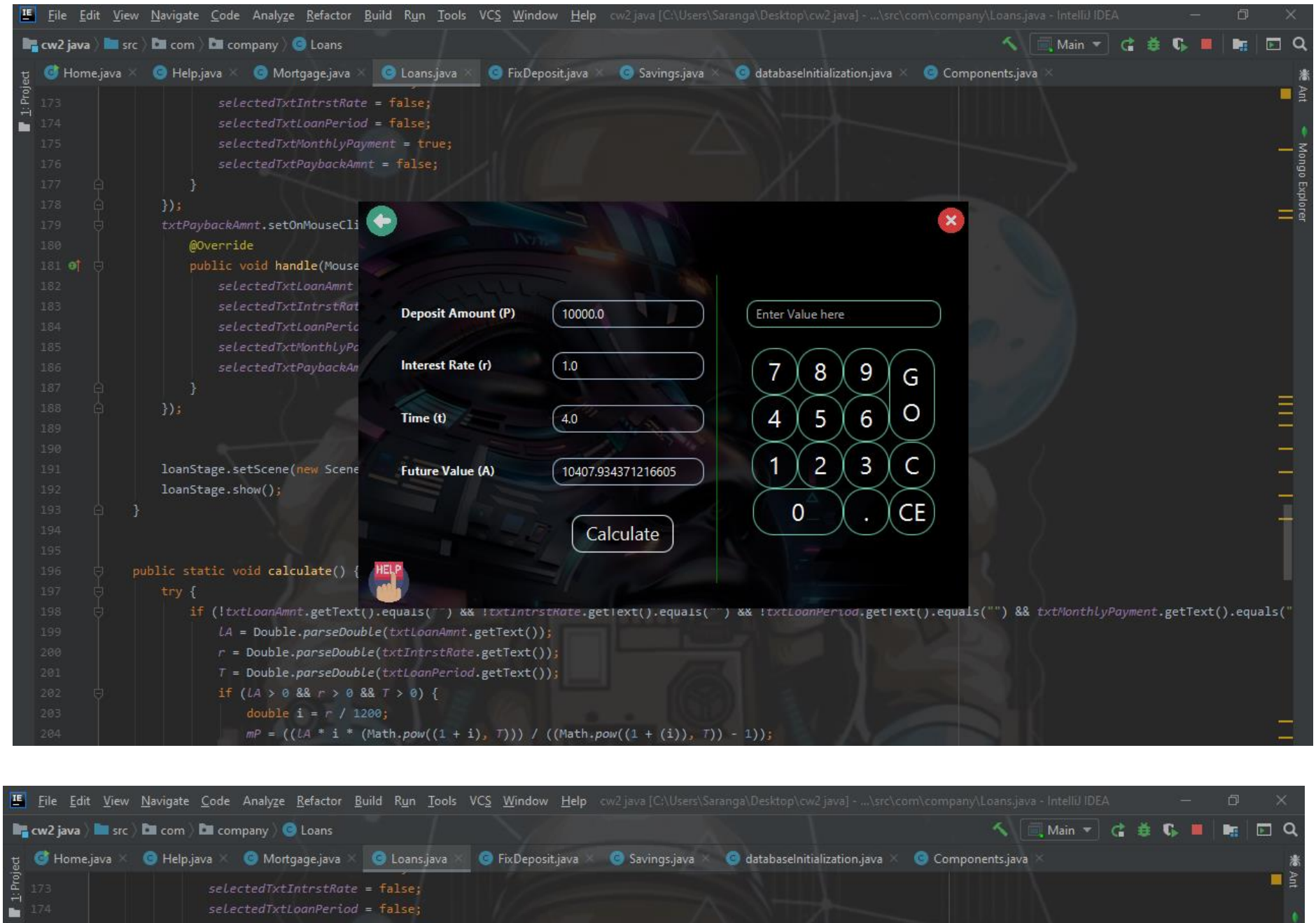
```
        Stage helpStage = new Stage();
        helpStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/helpFixed.jpg");
        ImageView bghelp = new ImageView();
        bghelp.setImage(image1);
        bghelp.setFitWidth(609);
        bghelp.setFitHeight(500);
        btnClse = Components.createBtnClose();
        Pane helpPane = new Pane();
        helpPane.getChildren().add(bghelp);
        helpPane.getChildren().add(btnClse);
        btnClse.setOnAction(e -> {
            helpStage.close();
        });
        helpStage.setScene(new Scene(helpPane, 600, 500));
        helpStage.show();
    }

    public static void helpWindowLoan() {
        Stage helpStage = new Stage();
        helpStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/helpLoans.jpg");
        ImageView bghelp = new ImageView();
        bghelp.setImage(image1);
        bghelp.setFitWidth(609);
        bghelp.setFitHeight(500);
        btnClse = Components.createBtnClose();
        Pane helpPane = new Pane();
        helpPane.getChildren().add(bghelp);
        helpPane.getChildren().add(btnClse);
        btnClse.setOnAction(e -> {
            helpStage.close();
        });
        helpStage.setScene(new Scene(helpPane, 600, 500));
        helpStage.show();
    }

    public static void helpWindowSavings() {
        Stage helpStage = new Stage();
        helpStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/helpSavings.jpg");
        ImageView bghelp = new ImageView();
        bghelp.setImage(image1);
        bghelp.setFitWidth(609);
        bghelp.setFitHeight(500);
        btnClse = Components.createBtnClose();
        Pane helpPane = new Pane();
        helpPane.getChildren().add(bghelp);
        helpPane.getChildren().add(btnClse);
        btnClse.setOnAction(e -> {
            helpStage.close();
        });
        helpStage.setScene(new Scene(helpPane, 600, 500));
        helpStage.show();
    }

    public static void helpWindowMortgage() {
        Stage helpStage = new Stage();
        helpStage.initStyle(StageStyle.UNDECORATED);
        Image image1 = new Image("Pics/helpMortgage.jpg");
        ImageView bghelp = new ImageView();
        bghelp.setImage(image1);
        bghelp.setFitWidth(609);
        bghelp.setFitHeight(500);
        btnClse = Components.createBtnClose();
        Pane helpPane = new Pane();
        helpPane.getChildren().add(bghelp);
        helpPane.getChildren().add(btnClse);
        btnClse.setOnAction(e -> {
            helpStage.close();
        });
        helpStage.setScene(new Scene(helpPane, 600, 500));
        helpStage.show();
    }

}
```

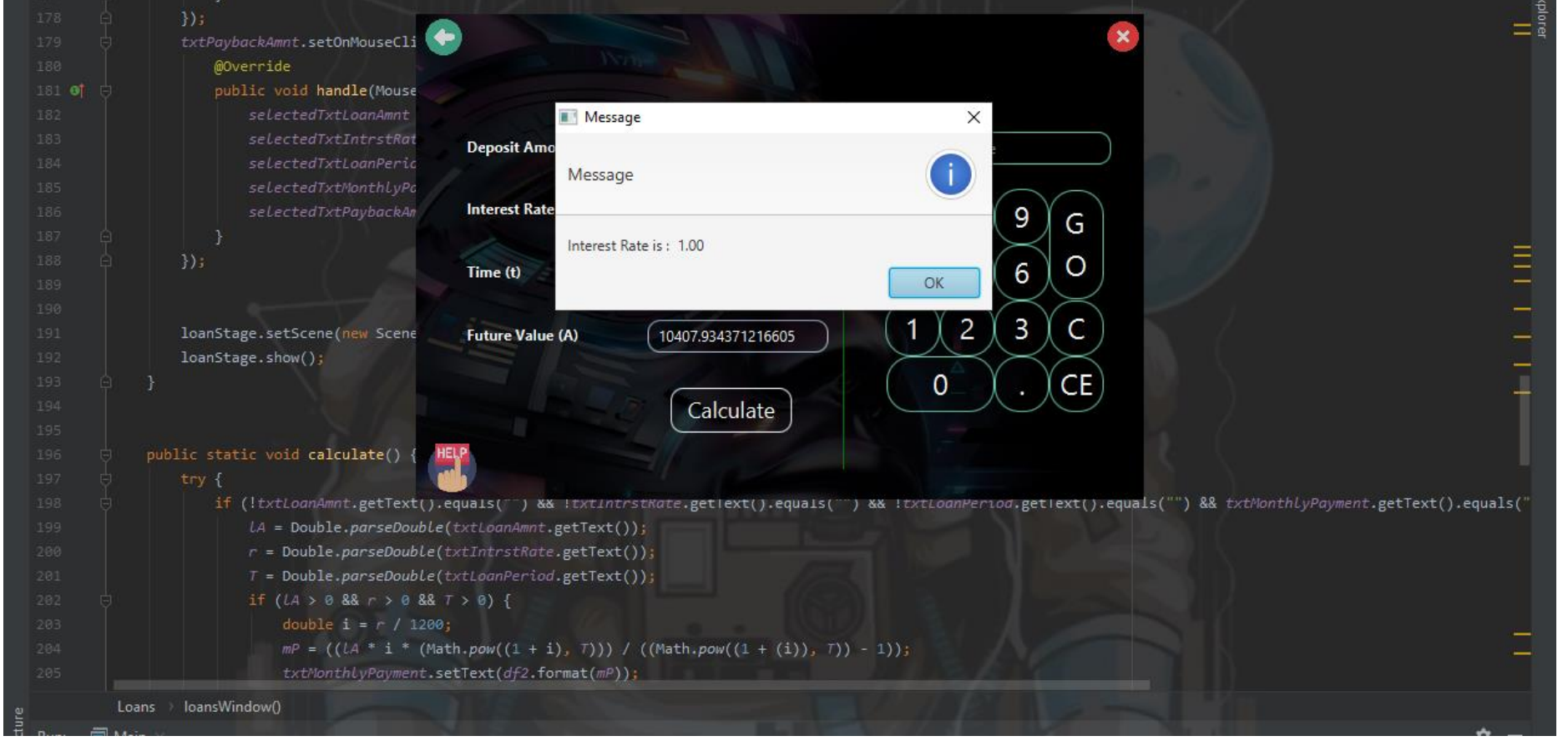
DatabaseInitiliation.java

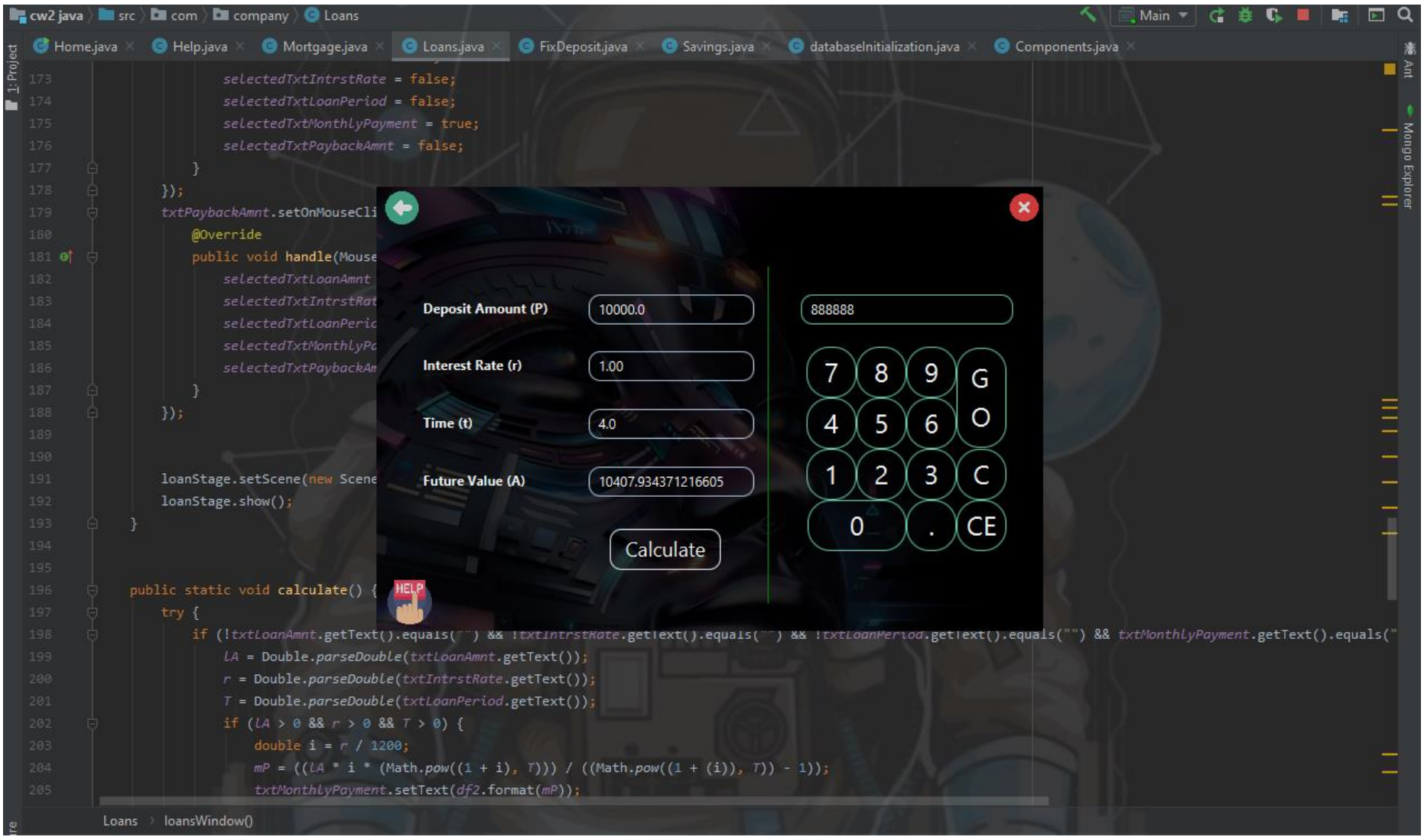
```
public class databaseInitialization {
    //class for connecting database
    public MongoClient connectingDB() {
        MongoClient mongoClient = new MongoClient("localhost", 27017);
        return mongoClient;
    }
}
```

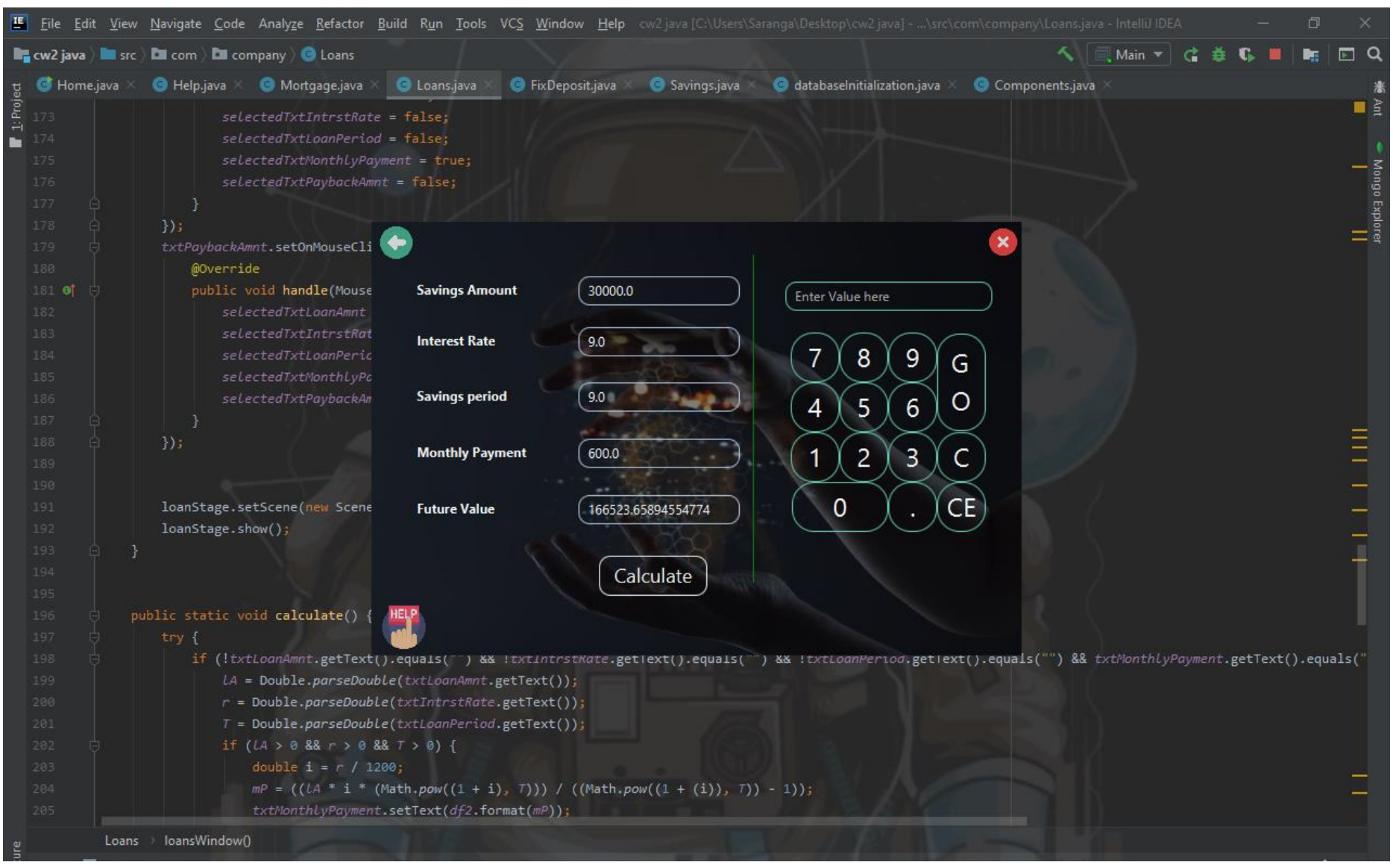
views

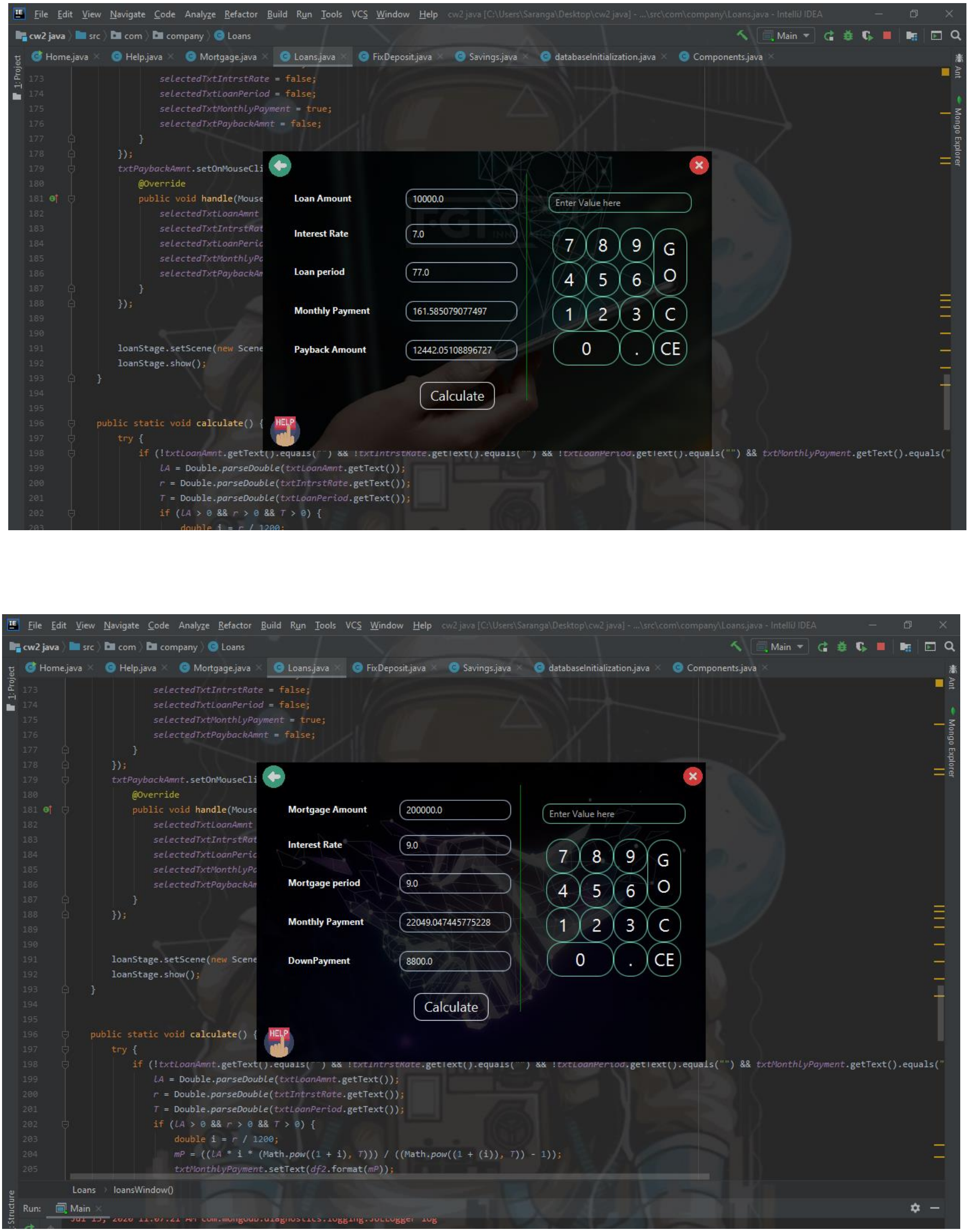

Financial Calculator
Fixed Deposit
Savings
Mortgage
Loan
HELP
HELP
FIELD THAT YOU ARE GOING TO FIND MUST BE THE ONLY EMPTY FIELD THAT SHOULD CONTAIN IN THE RELEVANT CALCULATOR.
INTEREST FIELD IS A MANDETORY FIELD TO FILL IN ALL THE CALCULATOR TYPES
INSIDE THE INTEREST FIELDS ON SAVINGS AND COMPOUND INTEREST SAVINGS CALCULATORS SHOULD INCLUDE THE ANUAL INTEREST RATES.
ONLY THE INTEGERS AND THE DECIMAL VALUES ARE ALLOWED TO FILL THE RELEVENT FIELDS
I RECOMEND YOU TO USE THE INBUILT VIRTUAL KEYBOARD TO INPUT VALUES SO YOU WON'T MISTAKENLY TYPE ANY MINUS VALUES INSIDE THE FIELDS BUT WE ALSO VALIDATED MOST OF WAYS THAT YOU WILL NOT BE ABLE TO ENTER ANY STRINGS AND (-) VALUES
*WHEN YOU WANT TO INPUT A CERTAIN VALUE TO A FIELD FOLLOW THE FOLLOWING STEPS.
1 NPUT FIELDS WILL REPOPULATED WHEN YOU OPEN THE APPLICATION .
WITH THE LASTN ENTERED DATA AFTER TERMINATING
Java
Software HUB

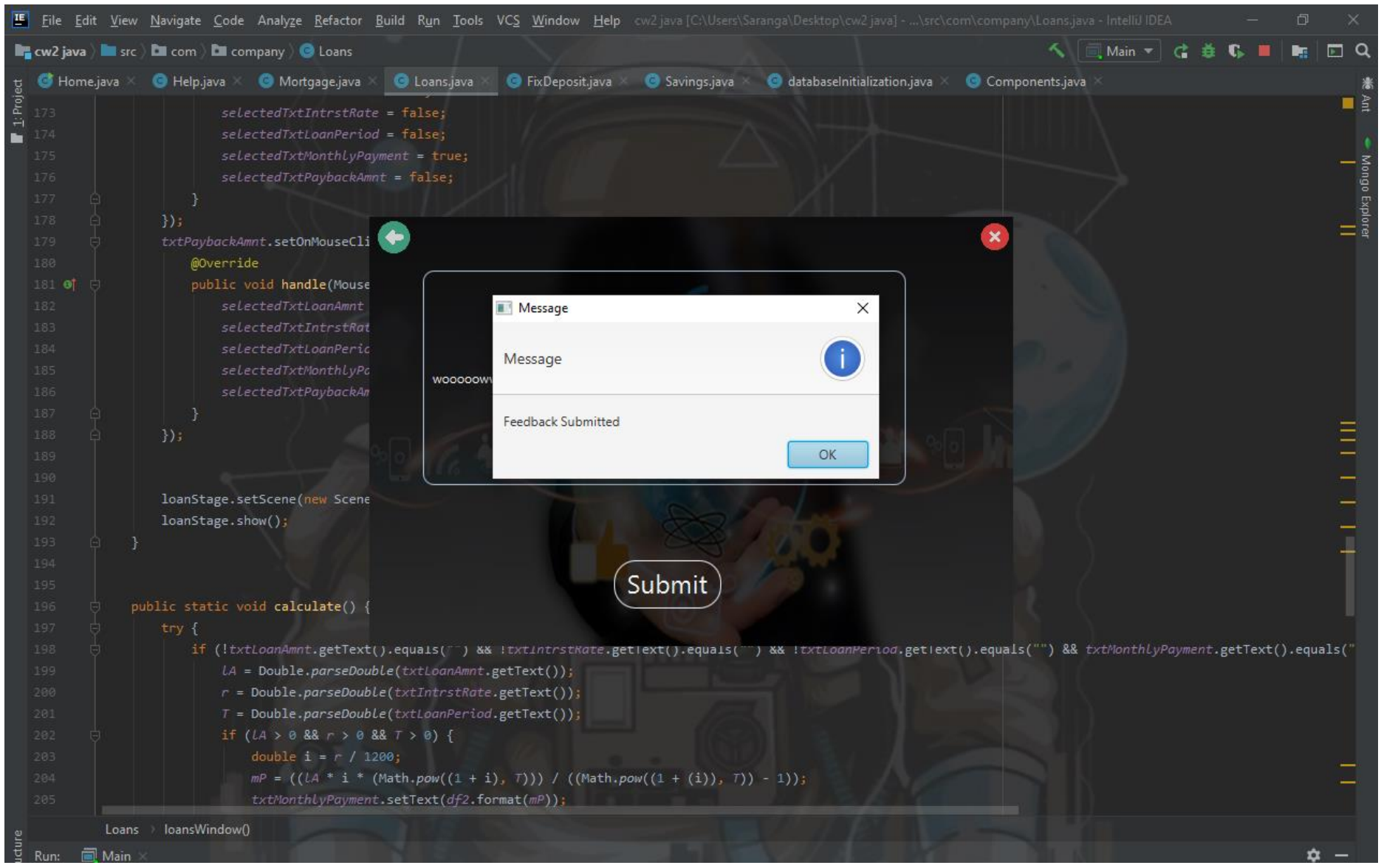

Deposit Amount (P)
10000.0
Interest Rate (r)
1.0
Time (t)
4.0
Future Value (A)
10407.934371216605
Calculate
Enter Value here
7 8 9 G
4 5 6 O
1 2 3 C
0 . CE
HELP

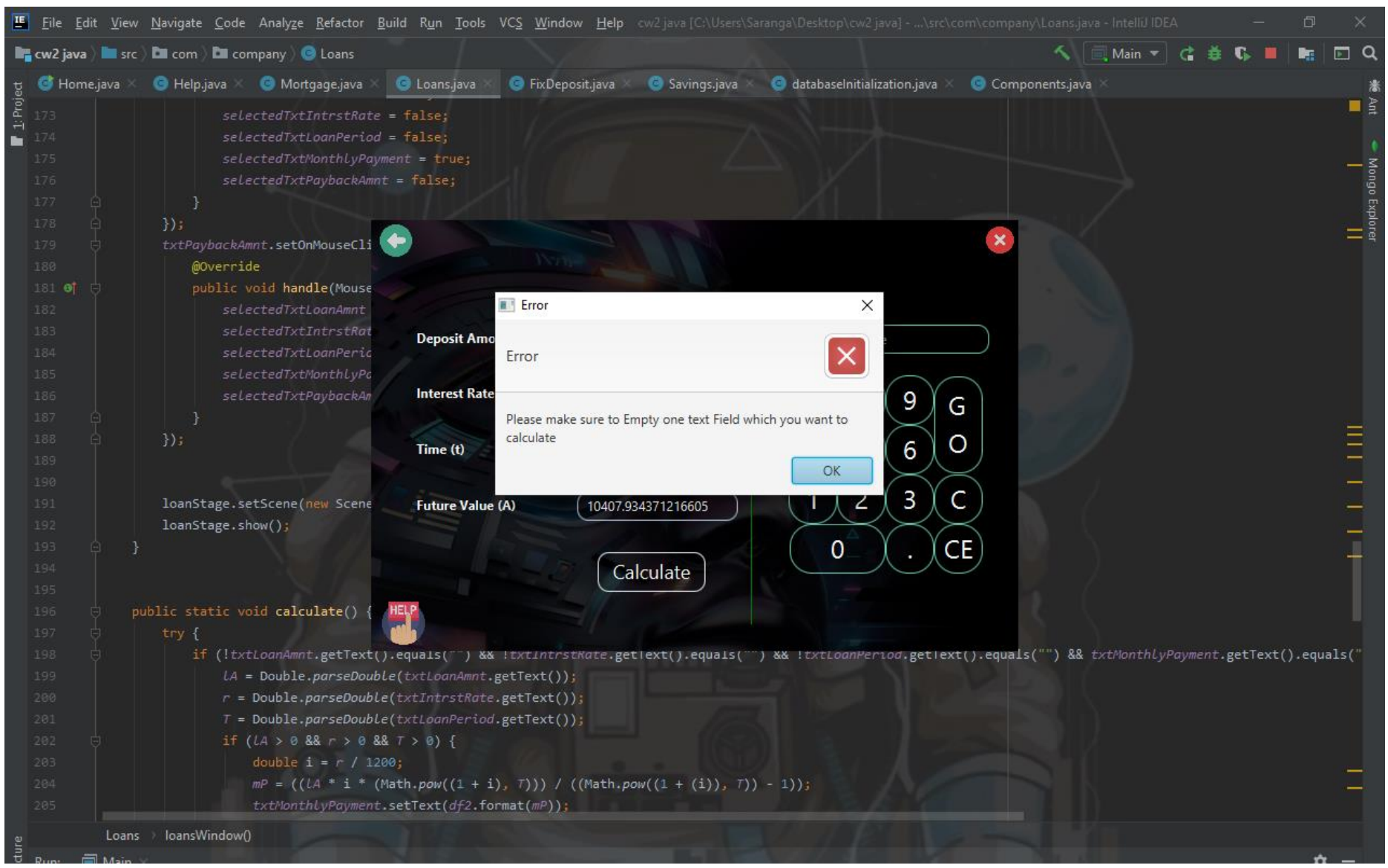

Message
Message
Interest Rate is : 1.00
OK
Deposit Amo
Interest Rate
Time (t)
Future Value (A)
10407.934371216605
Calculate
HELP

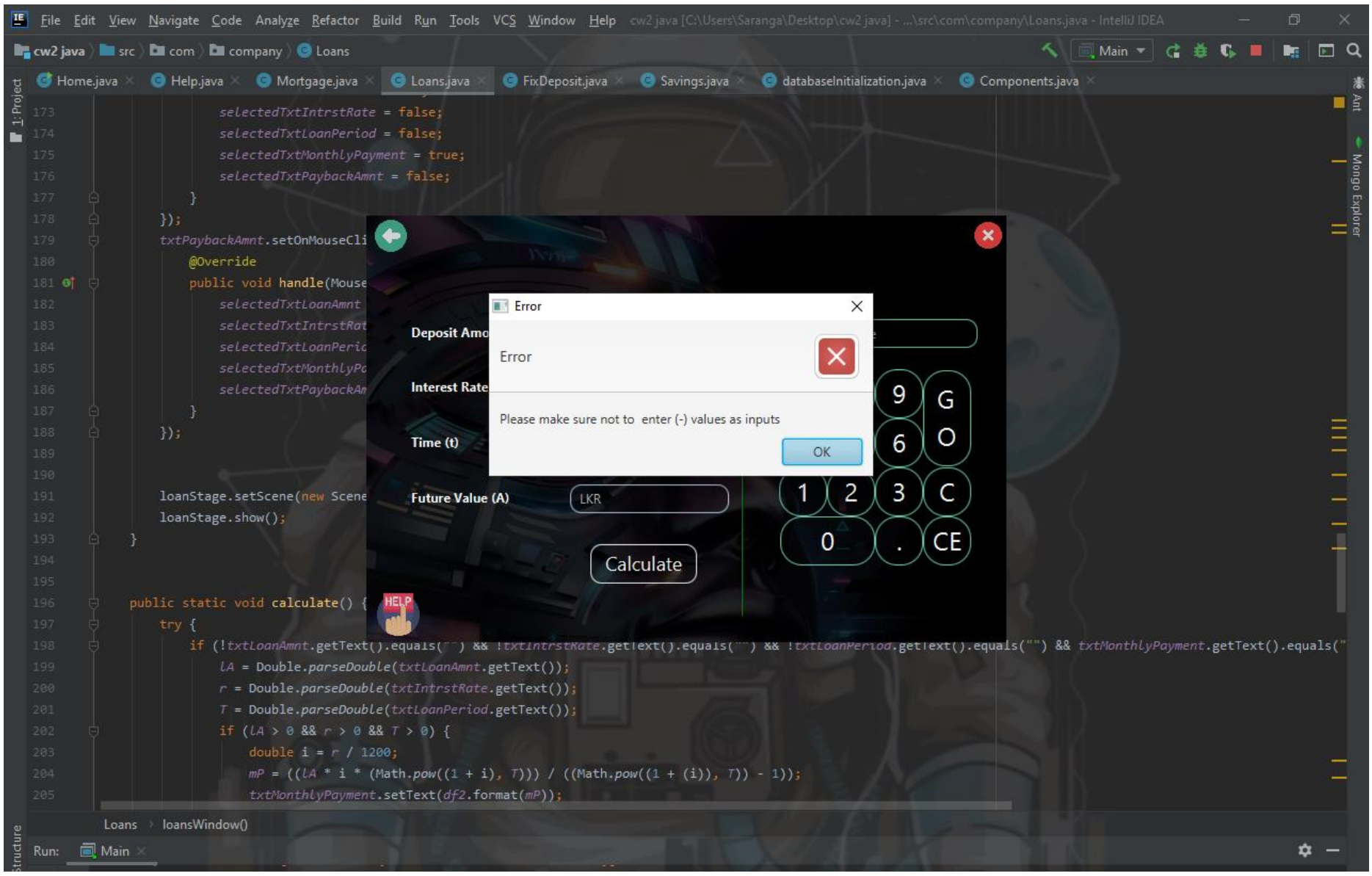

Deposit Amount (P)
10000.0
Interest Rate (r)
1.00
Time (t)
4.0
Future Value (A)
10407.934371216605
Calculate
888888
7 8 9 G O
4 5 6
1 2 3 C
0 . CE
HELP

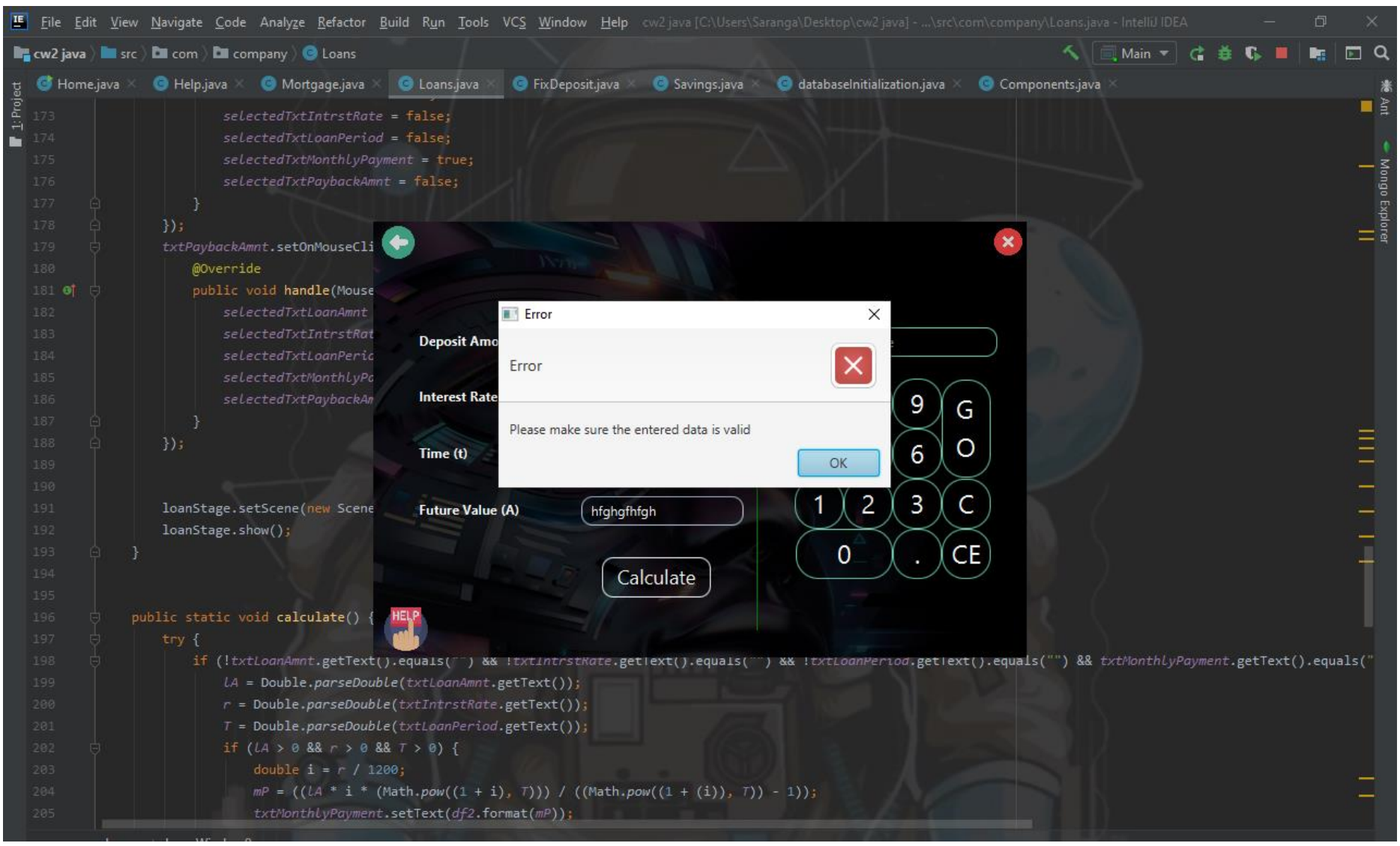

Savings Amount
30000.0
Interest Rate
9.0
Savings period
9.0
Monthly Payment
600.0
Future Value
166523.65894554774
Calculate
Enter Value here
7 8 9 G O
4 5 6
1 2 3 C
0 . CE
HELP

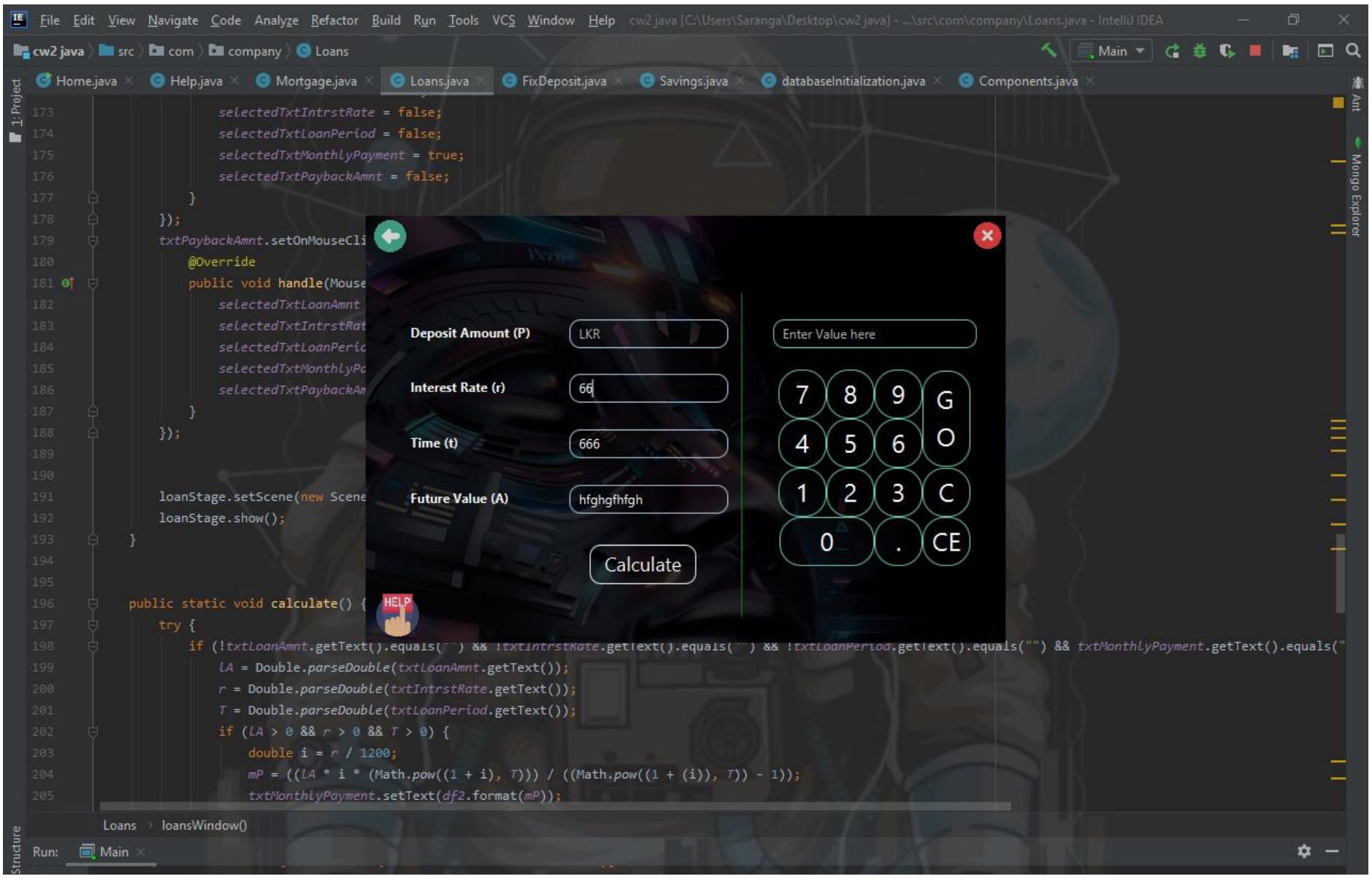

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
cw2 java [C:\Users\Saranga\Desktop\cw2 java] - ...\src\com\company\Loans.java - IntelliJ IDEA
cw2 java src com company Loans
Main
Home.java Help.java Mortgage.java Loans.java FixDeposit.java Savings.java databaseInitialization.java Components.java
selectedTxtIntrstRate = false;
selectedTxtLoanPeriod = false;
selectedTxtMonthlyPayment = true;
selectedTxtPaybackAmnt = false;
txtPaybackAmnt.setOnMouseCli
@Override
public void handle(Mouse
loanStage.setScene(new Scene
loanStage.show();
public static void calculate() {
try {
if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") && !txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("
lA = Double.parseDouble(txtLoanAmnt.getText());
r = Double.parseDouble(txtIntrstRate.getText());
T = Double.parseDouble(txtLoanPeriod.getText());
if (lA > 0 && r > 0 && T > 0) {
Loan Amount
10000.0
Interest Rate
7.0
Loan period
77.0
Monthly Payment
161.585079077497
Payback Amount
12442.05108896727
Calculate
Enter Value here
7 8 9 G
4 5 6 O
1 2 3 C
0 . CE
HELP
Mortgage Amount
200000.0
Interest Rate
9.0
Mortgage period
9.0
Monthly Payment
22049.047445775228
DownPayment
8800.0
Calculate
Enter Value here
7 8 9 G
4 5 6 O
1 2 3 C
0 . CE
HELP
double i = r / 1200;
mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
txtMonthlyPayment.setText(df2.format(mP));
Loans loansWindow()
Run: Main

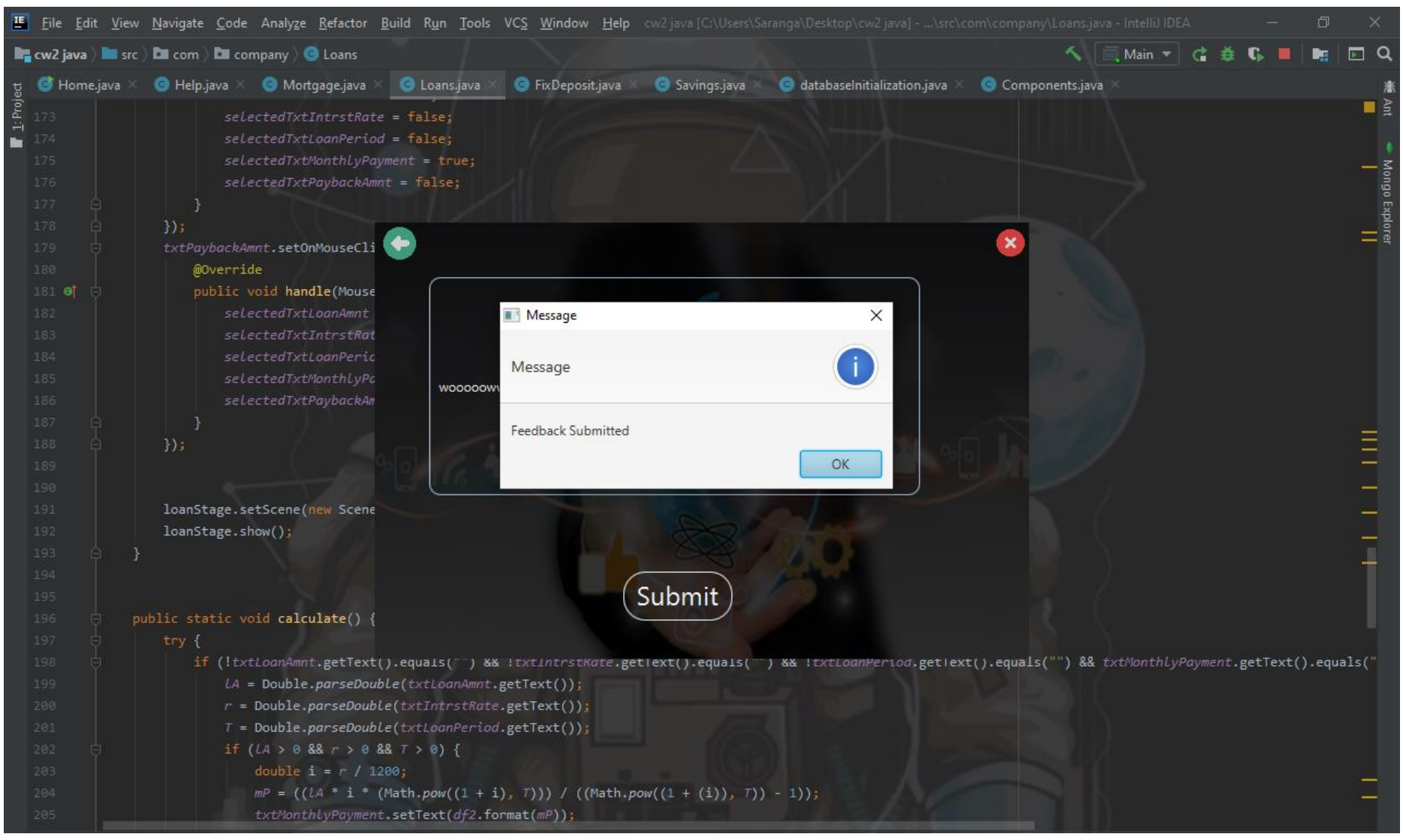

Message
Message
Feedback Submitted
OK
Submit

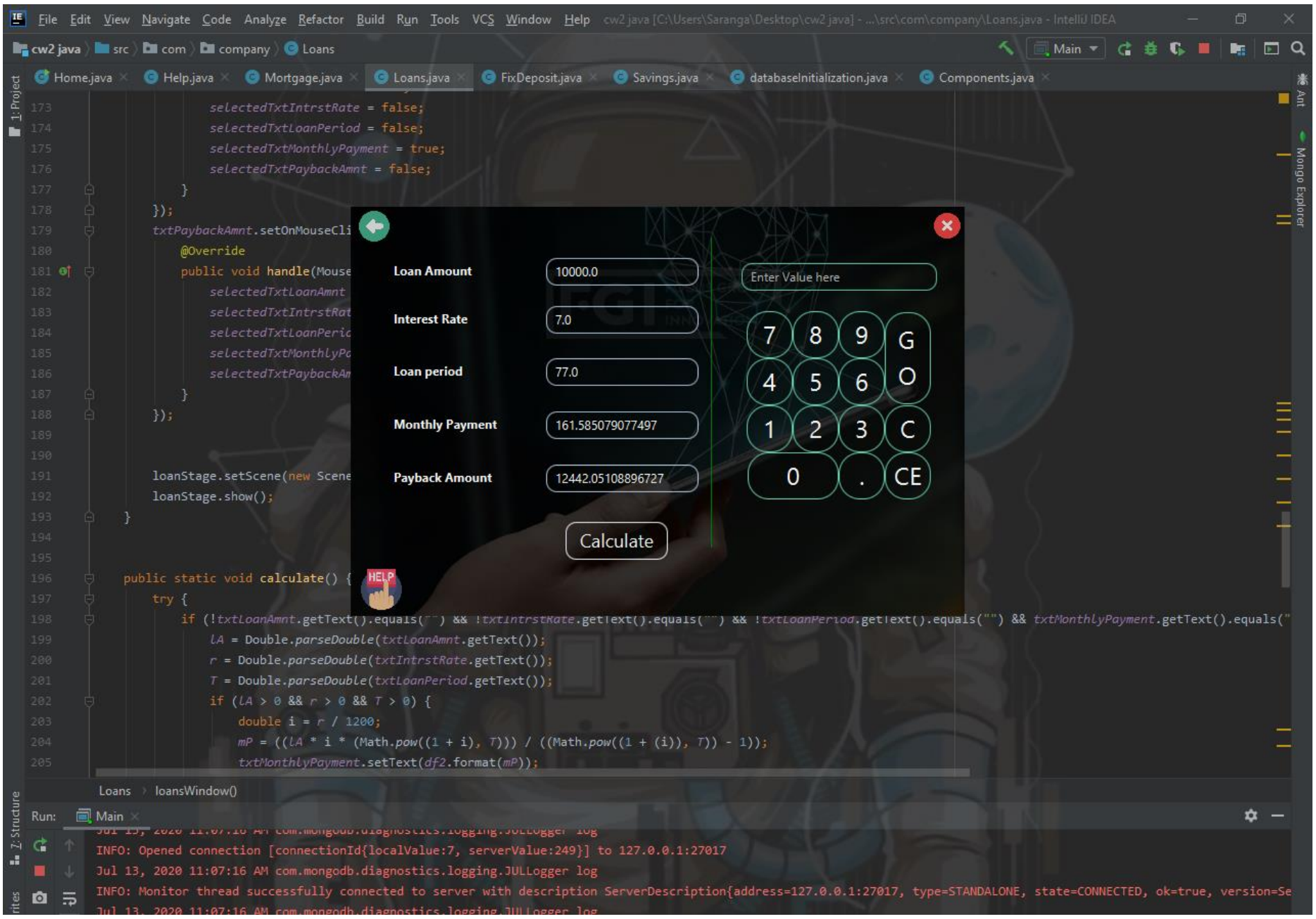

Error
Error
Please make sure to Empty one text Field which you want to calculate
OK
Deposit Amo
Interest Rate
Time (t)
Future Value (A)
10407.934371216605
Calculate
HELP

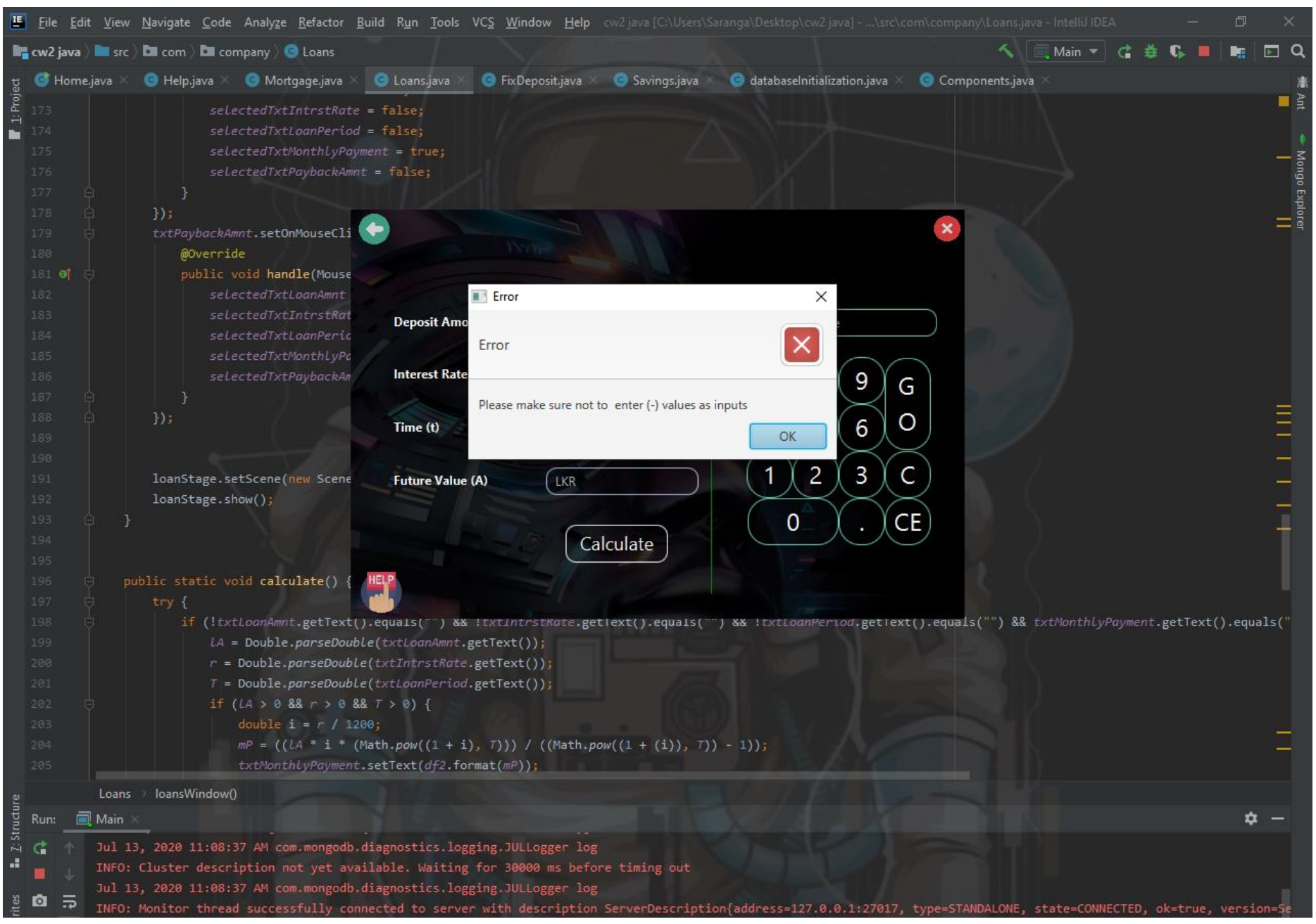

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
cw2 java [C:\Users\Saranga\Desktop\cw2 java] - ...\src\com\company\Loans.java - IntelliJ IDEA
cw2 java src com company Loans
Main
Home.java Help.java Mortgage.java Loans.java FixDeposit.java Savings.java databaseInitialization.java Components.java
selectedTxtIntrstRate = false;
selectedTxtLoanPeriod = false;
selectedTxtMonthlyPayment = true;
selectedTxtPaybackAmnt = false;
txtPaybackAmnt.setOnMouseCli
@Override
public void handle(Mouse
selectedTxtLoanAmnt
selectedTxtIntrstRat
selectedTxtLoanPeric
selectedTxtMonthlyPa
selectedTxtPaybackAm
loanStage.setScene(new Scene
loanStage.show();
public static void calculate() {
try {
if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") && !txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("
lA = Double.parseDouble(txtLoanAmnt.getText());
r = Double.parseDouble(txtIntrstRate.getText());
T = Double.parseDouble(txtLoanPeriod.getText());
if (lA > 0 && r > 0 && T > 0) {
double i = r / 1200;
mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
txtMonthlyPayment.setText(df2.format(mP));
Loans loansWindow()
Run: Main
Error
Error
Please make sure not to enter (-) values as inputs
OK
Deposit Amo
Interest Rate
Time (t)
Future Value (A)
LKR
Calculate
HELP
9 G
6 O
1 2 3 C
0 . CE


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
cw2 java [C:\Users\Saranga\Desktop\cw2 java] - ...\src\com\company\Loans.java - IntelliJ IDEA
cw2 java src com company Loans
Main
Home.java Help.java Mortgage.java Loans.java FixDeposit.java Savings.java databaseInitialization.java Components.java
selectedTxtIntrstRate = false;
selectedTxtLoanPeriod = false;
selectedTxtMonthlyPayment = true;
selectedTxtPaybackAmnt = false;
txtPaybackAmnt.setOnMouseCli
@Override
public void handle(Mouse
selectedTxtLoanAmnt
selectedTxtIntrstRat
selectedTxtLoanPeric
selectedTxtMonthlyPa
selectedTxtPaybackAm
loanStage.setScene(new Scene
loanStage.show();
public static void calculate() {
try {
if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") && !txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("
lA = Double.parseDouble(txtLoanAmnt.getText());
r = Double.parseDouble(txtIntrstRate.getText());
T = Double.parseDouble(txtLoanPeriod.getText());
if (lA > 0 && r > 0 && T > 0) {
double i = r / 1200;
mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
txtMonthlyPayment.setText(df2.format(mP));
Error
Error
Please make sure the entered data is valid
OK
Deposit Amo
Interest Rate
Time (t)
Future Value (A)
hfghgfhfgh
Calculate
HELP
9 G
6 O
1 2 3 C
0 . CE


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
cw2 java [C:\Users\Saranga\Desktop\cw2 java] - ...\src\com\company\Loans.java - IntelliJ IDEA
cw2 java src com company Loans
Main
Home.java Help.java Mortgage.java Loans.java FixDeposit.java Savings.java databaseInitialization.java Components.java
selectedTxtIntrstRate = false;
selectedTxtLoanPeriod = false;
selectedTxtMonthlyPayment = true;
selectedTxtPaybackAmnt = false;
txtPaybackAmnt.setOnMouseCli
@Override
public void handle(Mouse
loanStage.setScene(new Scene
loanStage.show();
public static void calculate() {
try {
if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") && !txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("
lA = Double.parseDouble(txtLoanAmnt.getText());
r = Double.parseDouble(txtIntrstRate.getText());
T = Double.parseDouble(txtLoanPeriod.getText());
if (lA > 0 && r > 0 && T > 0) {
double i = r / 1200;
mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
txtMonthlyPayment.setText(df2.format(mP));
Deposit Amount (P)
LKR
Interest Rate (r)
66
Time (t)
666
Future Value (A)
hfghgfhfgh
Calculate
Enter Value here
7 8 9 GO
4 5 6
1 2 3 C
0 . CE
HELP
Loans loansWindow()
Run: Main


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
cw2 java [C:\Users\Saranga\Desktop\cw2 java] - ...\src\com\company\Loans.java - IntelliJ IDEA
cw2 java src com company Loans
Main
Home.java Help.java Mortgage.java Loans.java FixDeposit.java Savings.java databaseInitialization.java Components.java
selectedTxtIntrstRate = false;
selectedTxtLoanPeriod = false;
selectedTxtMonthlyPayment = true;
selectedTxtPaybackAmnt = false;
txtPaybackAmnt.setOnMouseCli
@Override
public void handle(Mouse
loanStage.setScene(new Scene
loanStage.show();
public static void calculate() {
try {
if (!txtLoanAmnt.getText().equals("") && !txtIntrstRate.getText().equals("") && !txtLoanPeriod.getText().equals("") && txtMonthlyPayment.getText().equals("
lA = Double.parseDouble(txtLoanAmnt.getText());
r = Double.parseDouble(txtIntrstRate.getText());
T = Double.parseDouble(txtLoanPeriod.getText());
if (lA > 0 && r > 0 && T > 0) {
double i = r / 1200;
mP = ((lA * i * (Math.pow((1 + i), T))) / ((Math.pow((1 + (i)), T)) - 1));
txtMonthlyPayment.setText(df2.format(mP));
Message
Message
Feedback Submitted
OK
Submit
Loans loansWindow()
Run: Main


Loan Amount
10000.0
Interest Rate
7.0
Loan period
77.0
Monthly Payment
161.585079077497
Payback Amount
12442.05108896727
Enter Value here
Calculate
HELP
7 8 9 G O
4 5 6
1 2 3 C
0 . CE
Loans › loansWindow()
Run: Main
INFO: Opened connection [connectionId{localValue:7, serverValue:249}] to 127.0.0.1:27017
Jul 13, 2020 11:07:16 AM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=127.0.0.1:27017, type=STANDALONE, state=CONNECTED, ok=true, version=Se


Error
Error
Please make sure not to enter (-) values as inputs
OK
Future Value (A)
LKR
Calculate
HELP
Loans › loansWindow()
Run: Main
Jul 13, 2020 11:08:37 AM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster description not yet available. Waiting for 30000 ms before timing out
Jul 13, 2020 11:08:37 AM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=127.0.0.1:27017, type=STANDALONE, state=CONNECTED, ok=true, version=Se