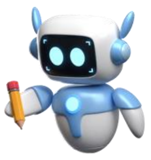
HDC – HGP- Assignment 02
UI Design Document



Student Name: Chukwuemeka Widom Arinze
Student Number: 2970177

1. Using an application such as draw.io or similar, create a wireframe of the layout of your UI, including explanation of what containers you used and why, as well as the naming convention used for each of your components.

Wireframe and explanation:

Start Method
The main layout I used was the BorderPane, the border pane allowed me to align layout in different directions around the main user interface. Each sub layouts were placed inside the main layout and the main layout allowed me to place each sub layout in a position in the main user interface. I named the BorderPane object border which is in all lowercase.

The first sub layout I used is GridPane, GridPane allowed me to align the labels side by side at the top left corner using rows and columns. I named the GridPane object grid which is all in lowercase alphabets.

The second sub layout I used was the VBox, VBox allowed elements to be displayed vertically. So, I used VBox to align elements vertically. I named the VBox object vbox which is all in lowercase alphabets.

Show Dialog and Remove Dialog
The main layout I used in each method was the BorderPane, just as I stated in the start method, this layout allows me to place UIs or sub layouts in any position that I wish. BorderPane allowed me to place the first sub layout that contained the label and text area in the centre of the dialog, and it also allowed me to place the second sub layout that contains all the buttons at the bottom of the dialog. I named the BorderPane object border which is in all lowercase alphabets.

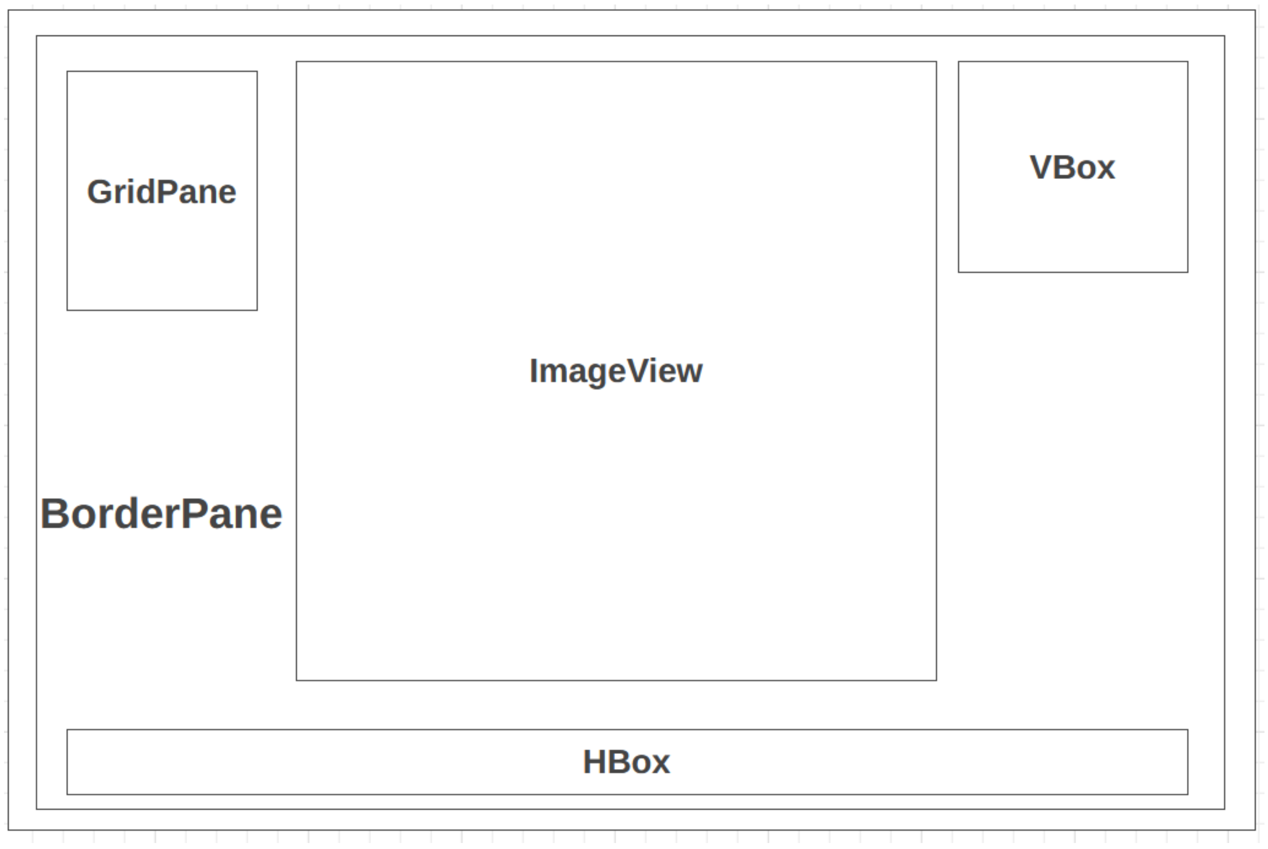
The first sub layout I used in both methods was a Hbox, HBox displays elements horizontally. Sort of like a queue. This layout held the label and the text area that will help the user input data for the application to process. The label tells the user what they need to input, and the text area collects the information entered by the user. I named the Hbox object hbox which is in all lowercase alphabets.

The second sub layout I created in both dialogs was also a HBox, this HBox arranges the elements horizontally. This layout will contain all the two buttons that are in both dialogs. I wanted them displayed horizontally because they will be equally spaced and allow the user to differentiate between two buttons. I named the Hbox object hbox which is in lowercase alphabets.

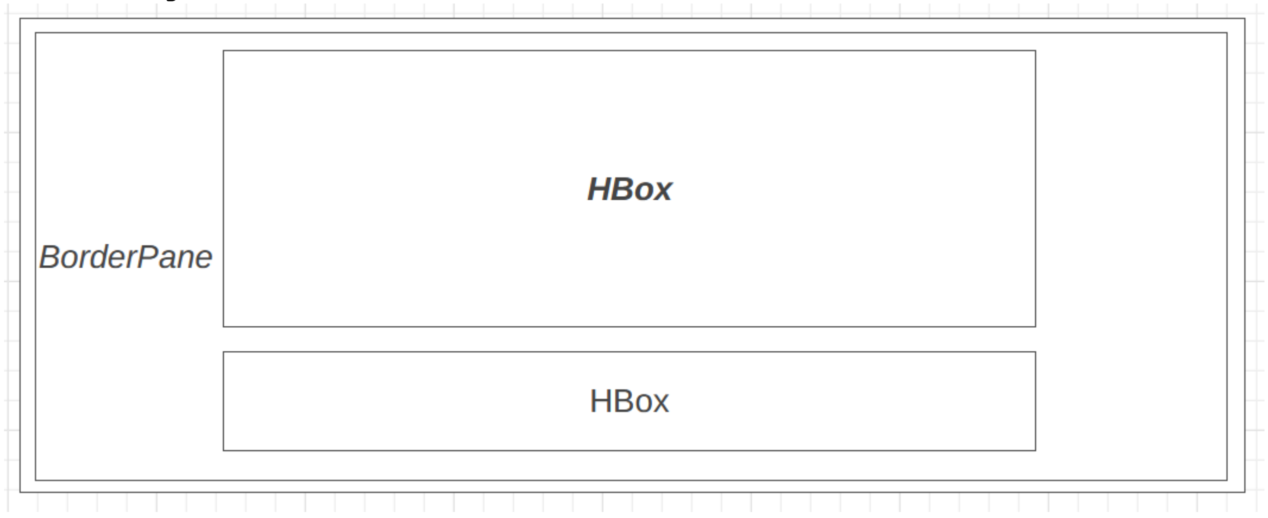
Display Problems Method
This method displays or alerts the user to any issues that they have encountered when entering an amount. I decided to use an alert because this takes the users focus from the dialog, they are currently on to the alert dialog. Inside this dialog, there will be at least one item that has occurred when the user tried to use the add dialog or the remove dialog. I named the Alert object alert which is in all lowercase

alphabets.

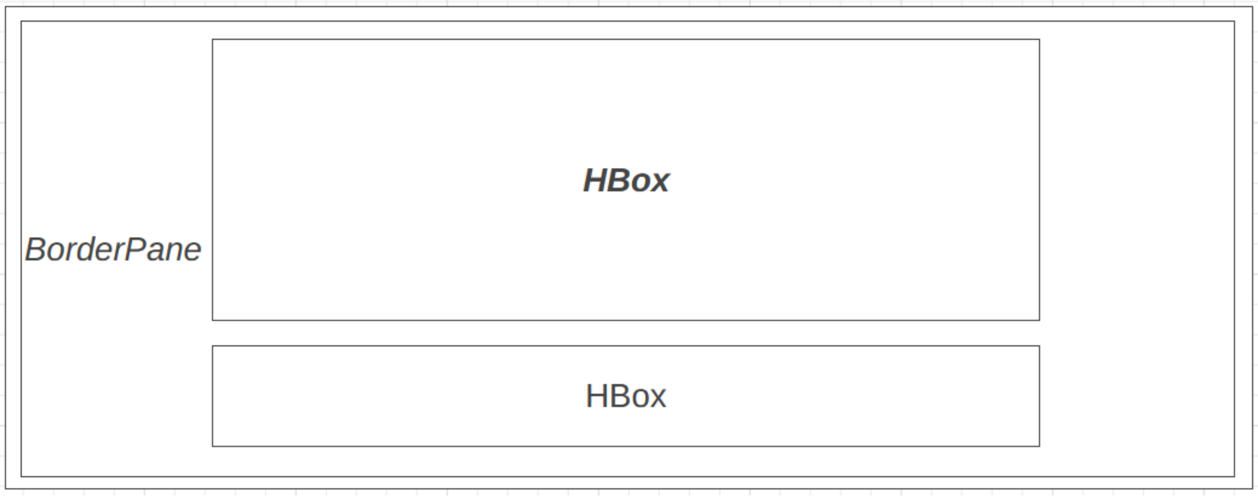
Main User Interface



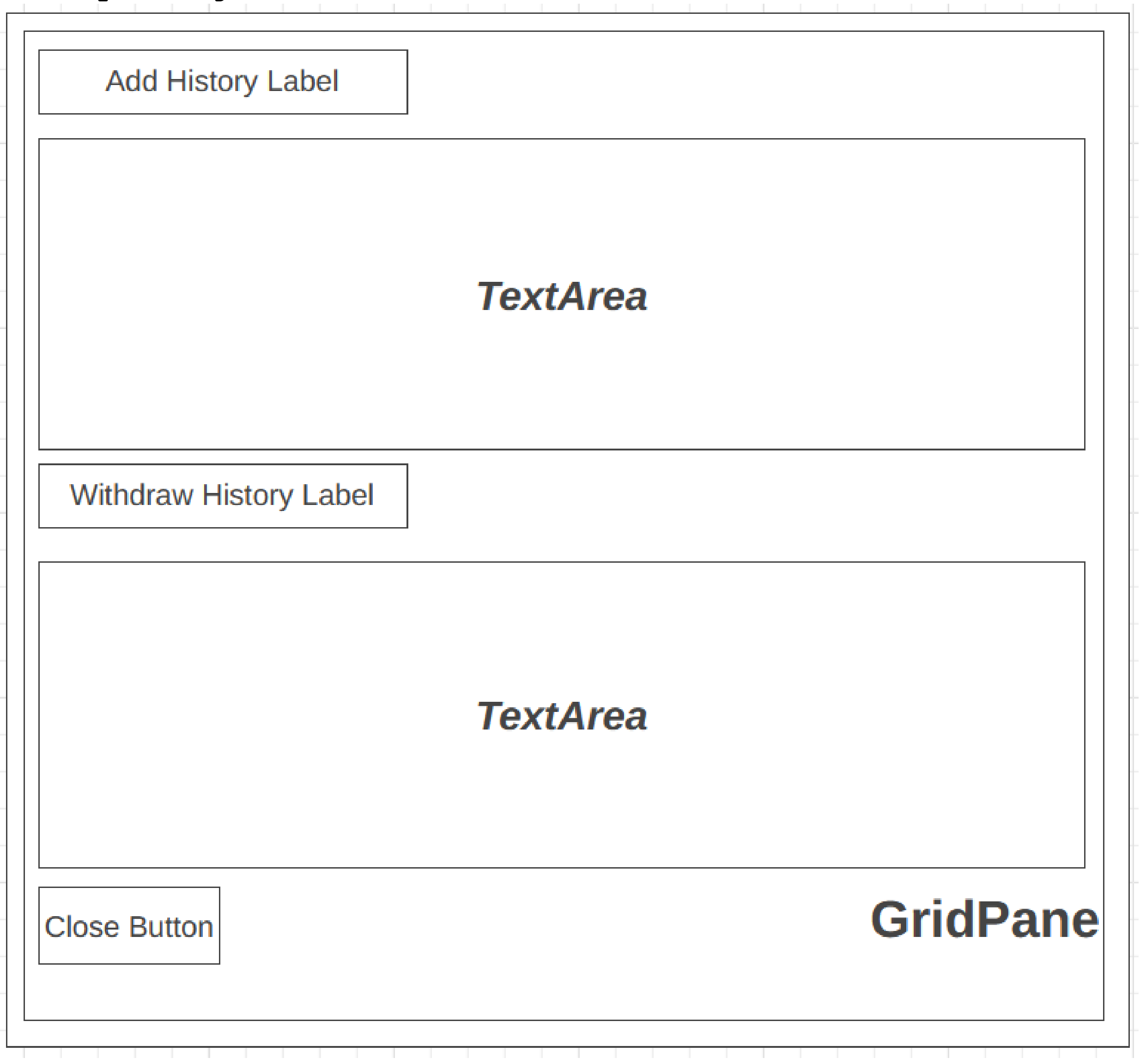
Add Dialog



Withdraw Dialog



History Dialog



2. Include screenshots of your Main User Interface. Why does your UI look the way it does? What design decisions did you make for this interface? Justify your decisions with a foundation of UI Design Principles.

Screenshots and explanation:

Labels

The labels are located at the top left corner of the main user interface because that position has space that can accommodate the labels and the information they contain. The text color for the labels is black because they are visible on a light background and the text size is bigger to increase visibility for the user.

Piggy Bank Image

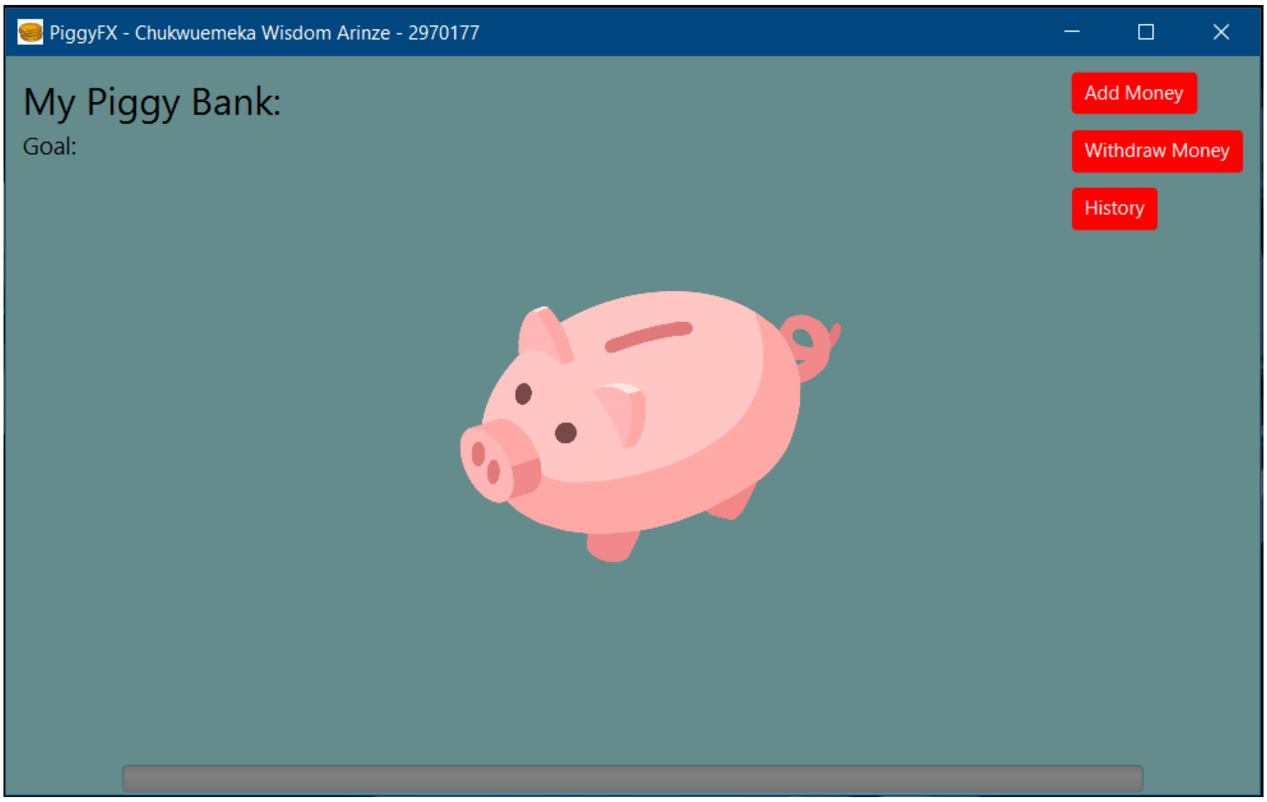
This image is located at the centre of the main user interface, I choose this location because it has the most space and it was a convenient location so that the user can see the progress. I made the image big enough to fit the centre of the main user interface.

Buttons

The buttons are located at the top right corner of the main user Interface. They are in that location because that location could fit all the buttons vertically. I made the buttons red because it will be easy for the user to spot them, and the colour made the application more appealing to the eye. I also increased the padding of the buttons so that they look bigger and easy to spot. The buttons also have the name and functionality on top of the button, and they are white. I chose white because they were visible on the red background that the buttons have.

Progress Bar

The progress bar was placed at the bottom of the main user interface. The progress bar was placed there so that the user can see the correlation of the progress and the change of the piggy bank image as money is increased and decreased. The progress bar should be big enough so that the user can see the bar increase and decrease and the bottom has enough space to contain the progress bar. I changed the progress bar colour so the user can see the increment and decrement of the progress bar. I chose red because it looked appealing for application.



3. Include screenshots of the 'Add Money' dialog. What components did you choose to include in the dialog? What design decisions did you make for this dialog? Justify your decisions with a foundation of UI Design Principles.

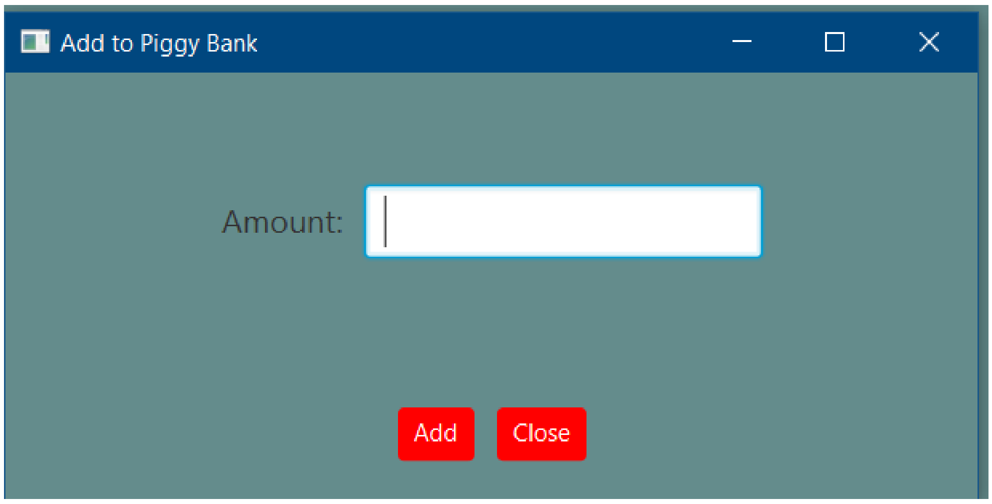
Screenshots and explanation:

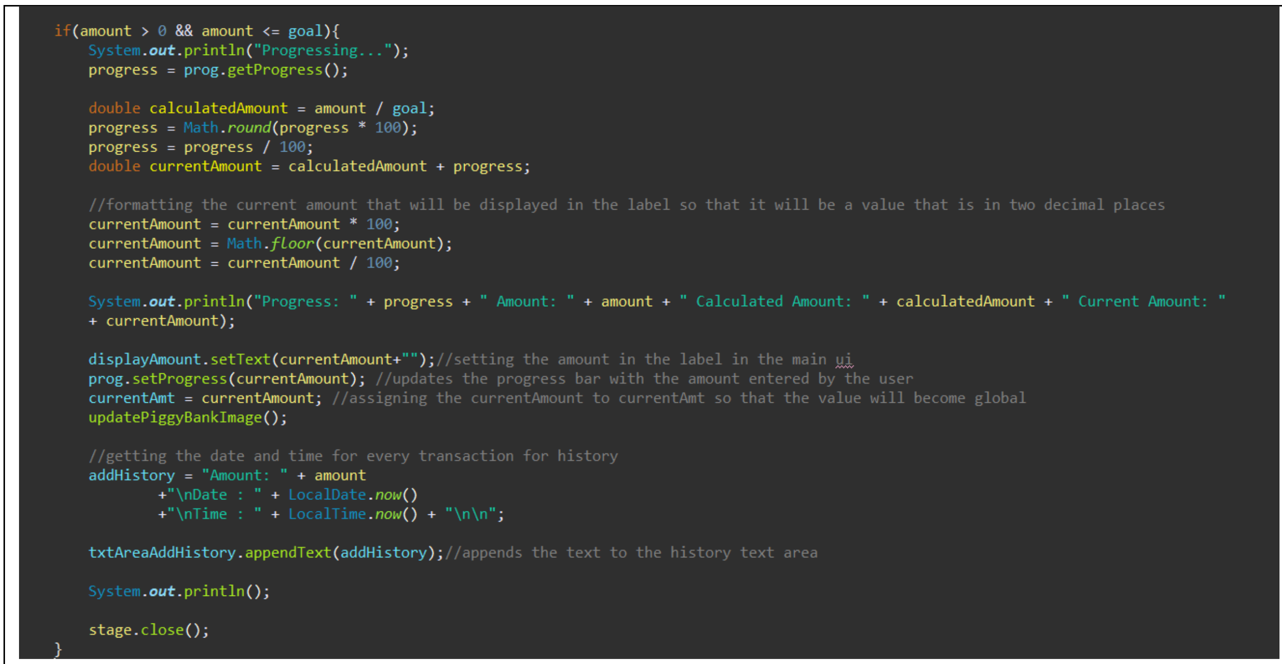
The first component I chose for this dialog was a label. The label allows me to place a text for the user to see. This text tells the user what type of information is required from them to be able to use the add money dialog without having error or problems.

The second component of this dialog I used was a text area. This text area will allow the user to input the type of information required from them. I choose to use a text area because I am familiar with it, and it is easy to use for the user. I also validated all the input entered by the user inside the text area. The text area also recovers when the wrong information is inserted into the text area. This text area looks, and it is placed in the same location in all the dialogs that have a text area.

The third and fourth component I used in this dialog was two buttons. The first button is the Add button. This button when pressed or clicked by the user passes the information that the user entered in the text area to the application for further processing. This button looks and feels the same as all the buttons in this application.

The fourth component or the second button cancels the dialog. If the user does not want to be in the add money dialog anymore, all they have to do is to click that button and it closes the add money dialog. This button is placed in the same other and place throughout the application. There is text on each button that will tell the user each buttons functionality.





```
    if(amount > 0 && amount <= goal){
        System.out.println("Progressing...");
        progress = prog.getProgress();

        double calculatedAmount = amount / goal;
        progress = Math.round(progress * 100);
        progress = progress / 100;
        double currentAmount = calculatedAmount + progress;

        //formatting the current amount that will be displayed in the label so that it will be a value that is in two decimal places
        currentAmount = currentAmount * 100;
        currentAmount = Math.floor(currentAmount);
        currentAmount = currentAmount / 100;

        System.out.println("Progress: " + progress + " Amount: " + amount + " Calculated Amount: " + calculatedAmount + " Current Amount: "
        + currentAmount);

        displayAmount.setText(currentAmount+"");//setting the amount in the label in the main ui
        prog.setProgress(currentAmount); //updates the progress bar with the amount entered by the user
        currentAmt = currentAmount; //assigning the currentAmount to currentAmt so that the value will become global
        updatePiggyBankImage();

        //getting the date and time for every transaction for history
        addHistory = "Amount: " + amount
                +"\nDate : " + LocalDate.now()
                +"\nTime : " + LocalTime.now() + "\n\n";

        txtAreaAddHistory.appendText(addHistory);//appends the text to the history text area

        System.out.println();

        stage.close();
    }
```

4. Include screenshots of the ‘Withdraw Money’ dialog. What components did you choose to include in the dialog? What design decisions did you make for this dialog? Justify your decisions with a foundation of UI Design Principles.

Screenshots and explanation:

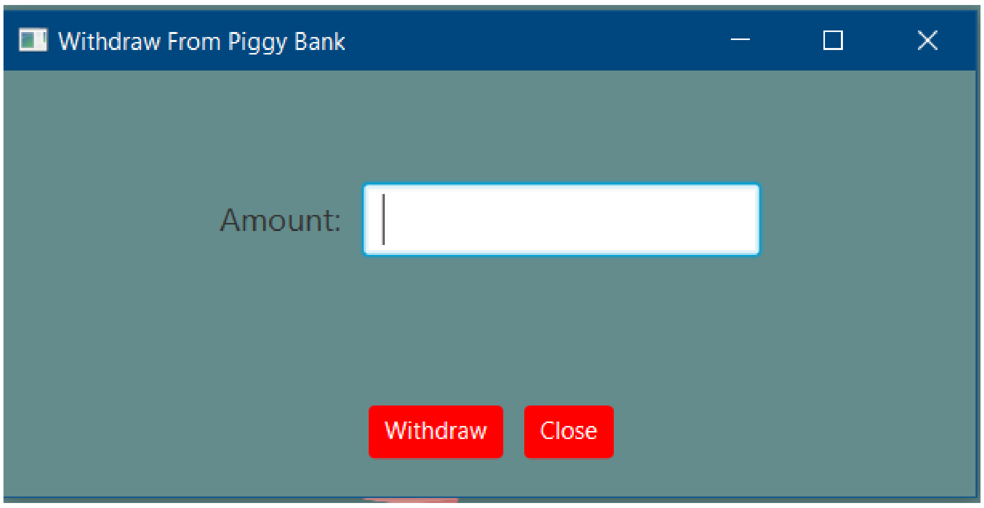
The first component in this dialog is a label. This label contains information about the type of information that is required in this dialog. The label was the most appropriate component to communicate what type of information is required from the user because it cannot be changed by the user, and it displays text.

The second component that I used in this component was a text area. This was appropriate because we need the user to enter information for the application to process it. The text area was placed close to the label, so we do not have the user going some lace else to input information. The text area allows the user to recover from errors when they occur. The type of error that occurs will be communicated to user when they occur so that they can recover. The text area is also in totally controlled by the user.

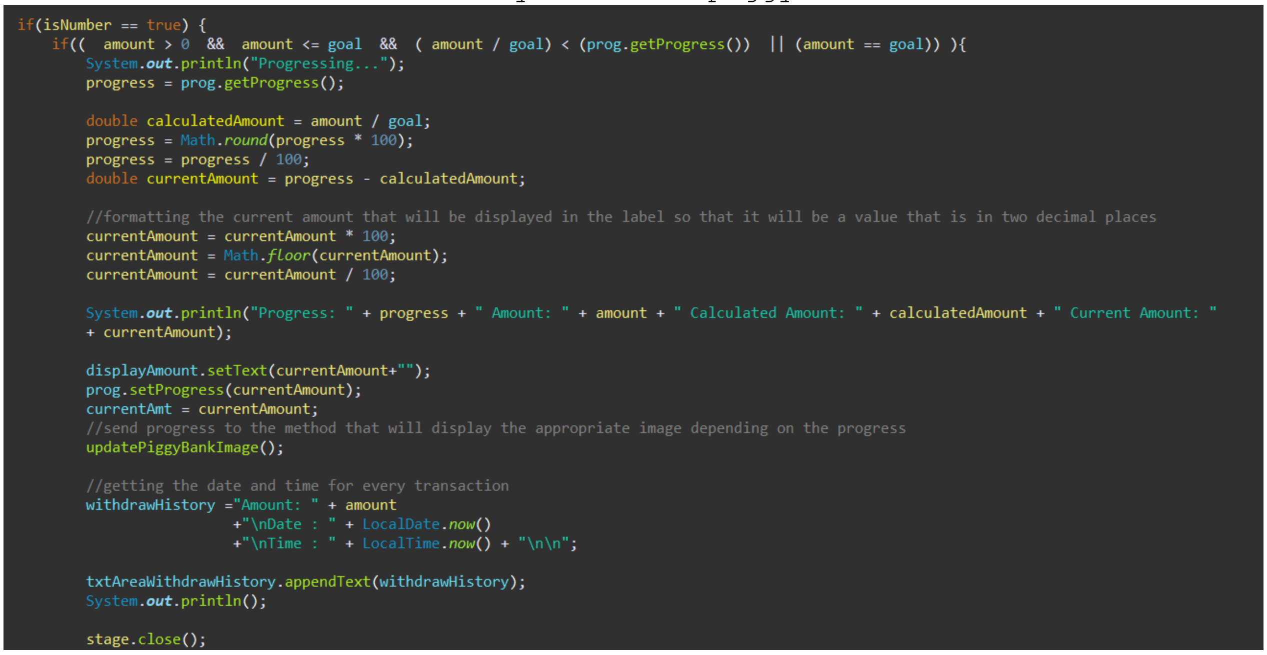
The third and fourth components in this dialog are buttons. The first button is the withdraw button. It looks and feels, and it is in the same location as the add money dialog. This button when clicked will collect the information the user enters and further processes it. The button contains text on it which tells the user the functionality of the button.

The fourth component which is the second button is the cancel button.

This button is consistent with all the other cancel buttons throughout the application. The button has a text on it, which tells the user the functionality of the button. This button just closes the remove money dialog. The button is totally controlled by the user.



Main code that withdraws money from the piggy bank.



```
if(isNumber == true) {
    if((  amount > 0  &&  amount <= goal  &&  ( amount / goal) < (prog.getProgress())  || (amount == goal)) ){
        System.out.println("Progressing...");
        progress = prog.getProgress();

        double calculatedAmount = amount / goal;
        progress = Math.round(progress * 100);
        progress = progress / 100;
        double currentAmount = progress - calculatedAmount;

        //formatting the current amount that will be displayed in the label so that it will be a value that is in two decimal places
        currentAmount = currentAmount * 100;
        currentAmount = Math.floor(currentAmount);
        currentAmount = currentAmount / 100;

        System.out.println("Progress: " + progress + " Amount: " + amount + " Calculated Amount: " + calculatedAmount + " Current Amount: "
        + currentAmount);

        displayAmount.setText(currentAmount+"");
        prog.setProgress(currentAmount);
        currentAmt = currentAmount;
        //send progress to the method that will display the appropriate image depending on the progress
        updatePiggyBankImage();

        //getting the date and time for every transaction
        withdrawHistory ="Amount: " + amount
                         +"\nDate : " + LocalDate.now()
                         +"\nTime : " + LocalTime.now() + "\n\n";

        txtAreaWithdrawHistory.appendText(withdrawHistory);
        System.out.println();

        stage.close();
```

5. Briefly explain the structure of your application, including what methods are included and what their functionality is.

Screenshots and explanation:

Start Method
When the application starts, the start method is executed and that displays the main user interface. All the buttons, the image, and labels are placed in the appropriate location.

Init Method
This method handles all the events in this application. An event is triggered in the application when the user performs an operation on an element in the application. There are just three events in this application. The add money dialog, the remove money dialog, and the history dialog.

Add Money Method
When the user presses the add money button, the add money dialog appears. Inside the add money dialog, when the user enter an amount and it passes all the validation then the image is updated, and the amount of money in the label at the top left corner of the main user interface is increased. That transaction is then recorded and played in the add money text area.

Remove Money Method
When the user presses the remove money dialog, the remove dialog appears. Then the user enters an amount, if the amount passes all the validation, then the main image at the centre of the main user interface is updated and then the value in the label at the top left of the main user interface is reduced. Then the transaction is recorded and placed in the withdraw transaction text area. If an error occurred when the user enters an amount, then an alert will be displayed to the user telling them about the error so they can recover. The

History Method
When the user clicks on the history button, every transaction in the add money dialog and the remove money dialog is displayed for the user to see.

Alert Problems method.
This method is triggered if there is a problem when the user enters an amount of money in the add money dialog or withdraw money dialog. If the money entered by the user failed any of the validation, then an alert is displayed to the user telling them about the error that occurred so that they can recover.

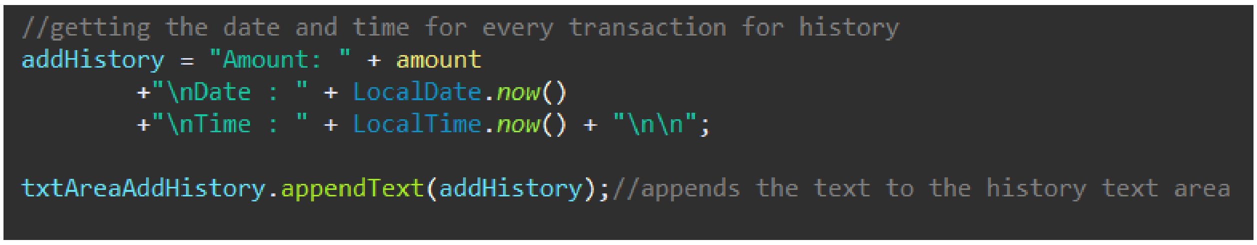
6. EXTRA FEATURE: Explain what your extra feature is, why you chose it, how it works etc. Include any relevant screenshots of your extra feature in action. Remember you must complete this section in detail to receive marks for your extra feature.

Screenshots and explanation:

The additional element I added to the application was a way for the application to keep track of each transaction. When the user adds money to the piggy bank, the amount entered, the date it was entered and the time it was entered is recorded and appended to a string then that string is appended to the add history text area.
The same process happens when the user withdraws money from the piggy bank. The amount of money entered is recorded, the date it was entered is recorded, and the time is also recorded and appended to a string. Then this string is appended to the withdraw history text area.

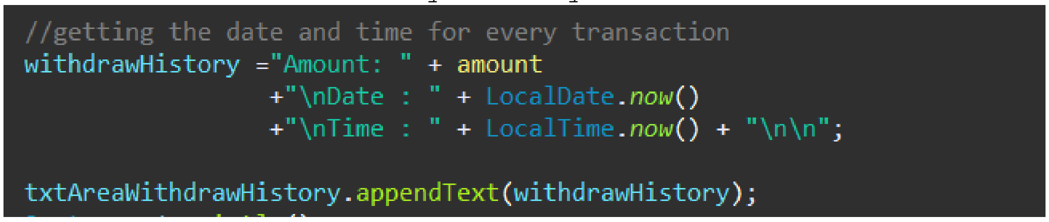
Button to trigger the display history functionality.

Code for the add money history.



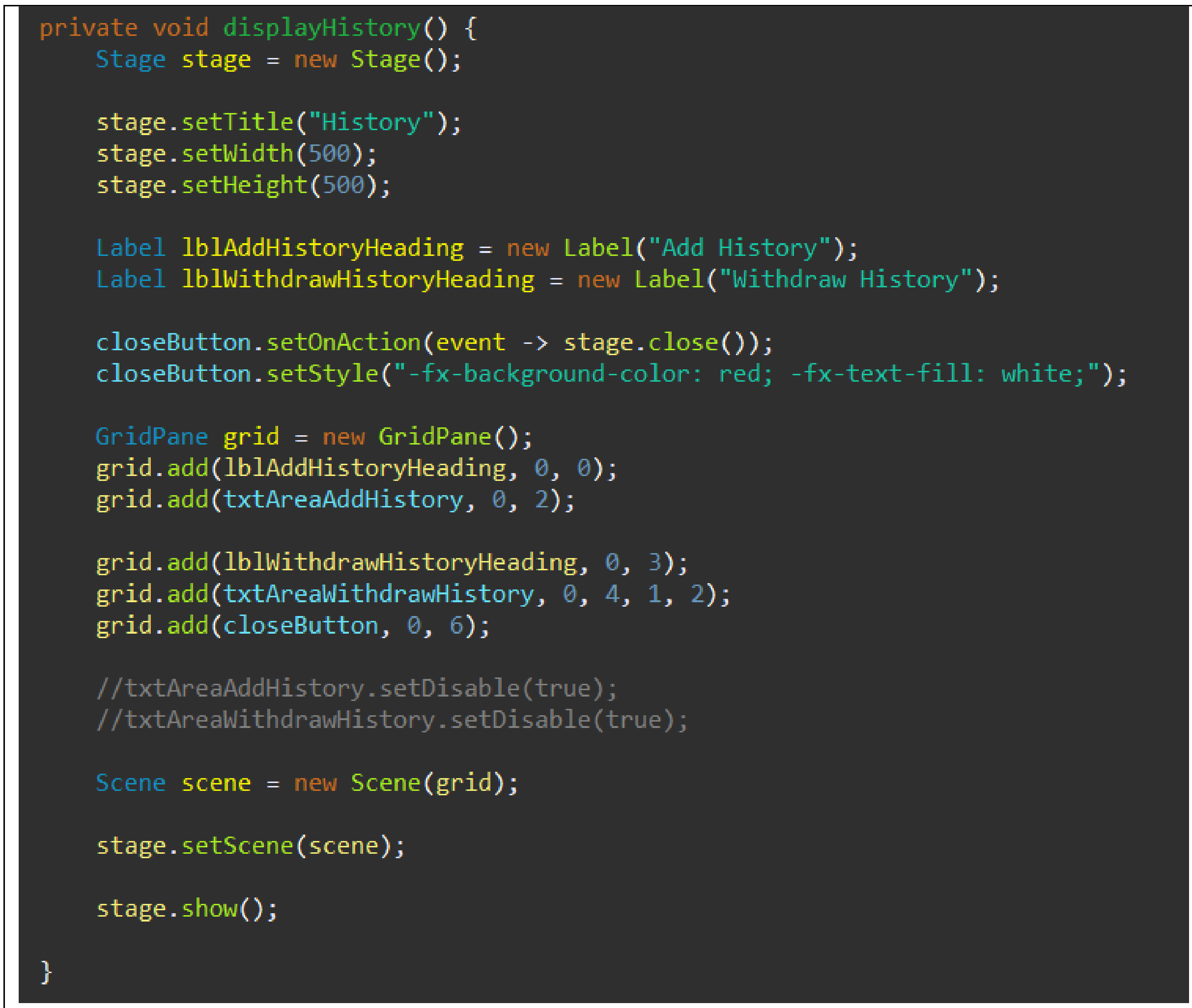
```
//getting the date and time for every transaction for history
addHistory = "Amount: " + amount
        +"\nDate : " + LocalDate.now()
        +"\nTime : " + LocalTime.now() + "\n\n";

txtAreaAddHistory.appendText(addHistory);//appends the text to the history text area
```

Code for the remove money history.



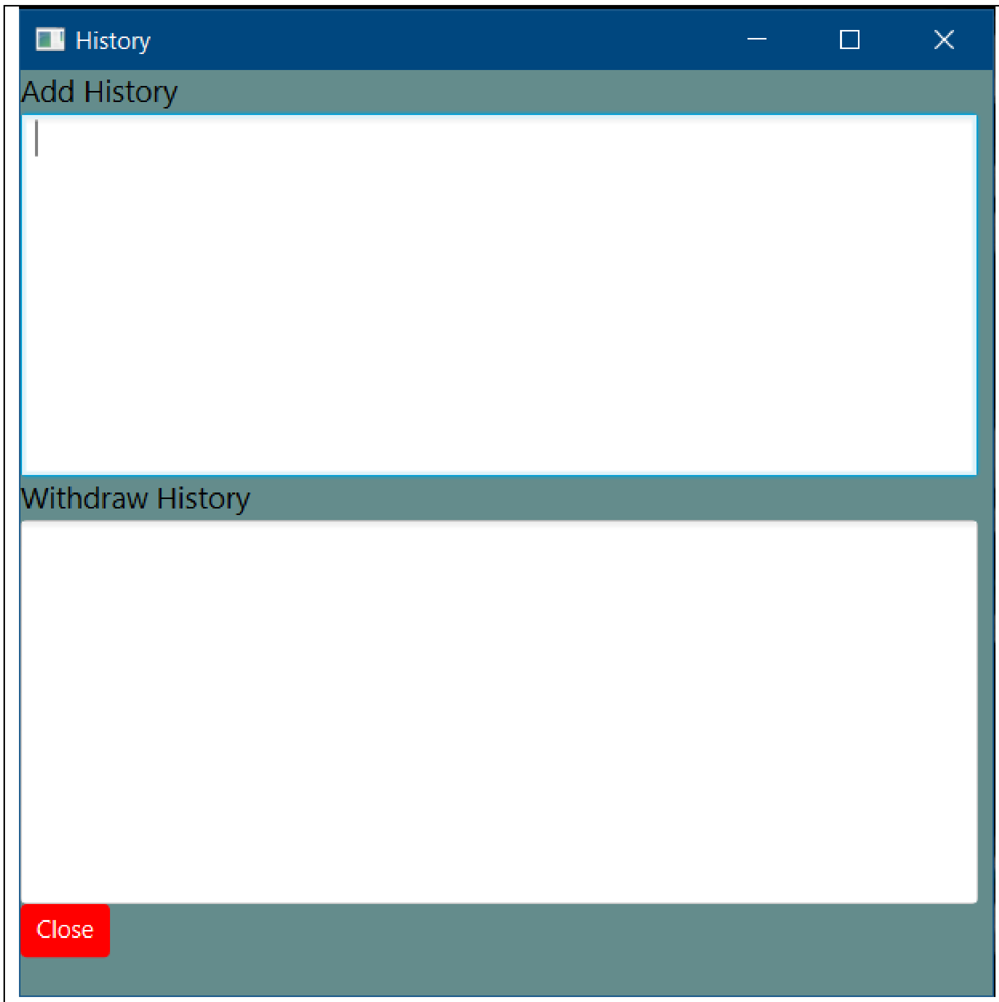
```
//getting the date and time for every transaction
withdrawHistory ="Amount: " + amount
                  +"\nDate : " + LocalDate.now()
                  +"\nTime : " + LocalTime.now() + "\n\n";

txtAreaWithdrawHistory.appendText(withdrawHistory);
```

Code for the add and remove history.



```
private void displayHistory() {
    Stage stage = new Stage();

    stage.setTitle("History");
    stage.setWidth(500);
    stage.setHeight(500);

    Label lblAddHistoryHeading = new Label("Add History");
    Label lblWithdrawHistoryHeading = new Label("Withdraw History");

    closeButton.setOnAction(event -> stage.close());
    closeButton.setStyle("-fx-background-color: red; -fx-text-fill: white;");

    GridPane grid = new GridPane();
    grid.add(lblAddHistoryHeading, 0, 0);
    grid.add(txtAreaAddHistory, 0, 2);

    grid.add(lblWithdrawHistoryHeading, 0, 3);
    grid.add(txtAreaWithdrawHistory, 0, 4, 1, 2);
    grid.add(closeButton, 0, 6);

    //txtAreaAddHistory.setDisable(true);
    //txtAreaWithdrawHistory.setDisable(true);

    Scene scene = new Scene(grid);

    stage.setScene(scene);

    stage.show();

}
```

GUI for the add money history and GUI for the remove money history.



7. Describe the interactive elements in your UI. How did you make sure they are intuitive and easy to use?

Explanation:

All the buttons have text on them, which tells the user the functionality of the button when they are presses or clicked.

In respect to the styling of the buttons, there is a drop shadow that is triggered when the user hovers around the buttons in the main user interface.

When the user adds or removes money from the piggy bank the progress bar increases or decreases and the piggy bank image updates in the same manner.

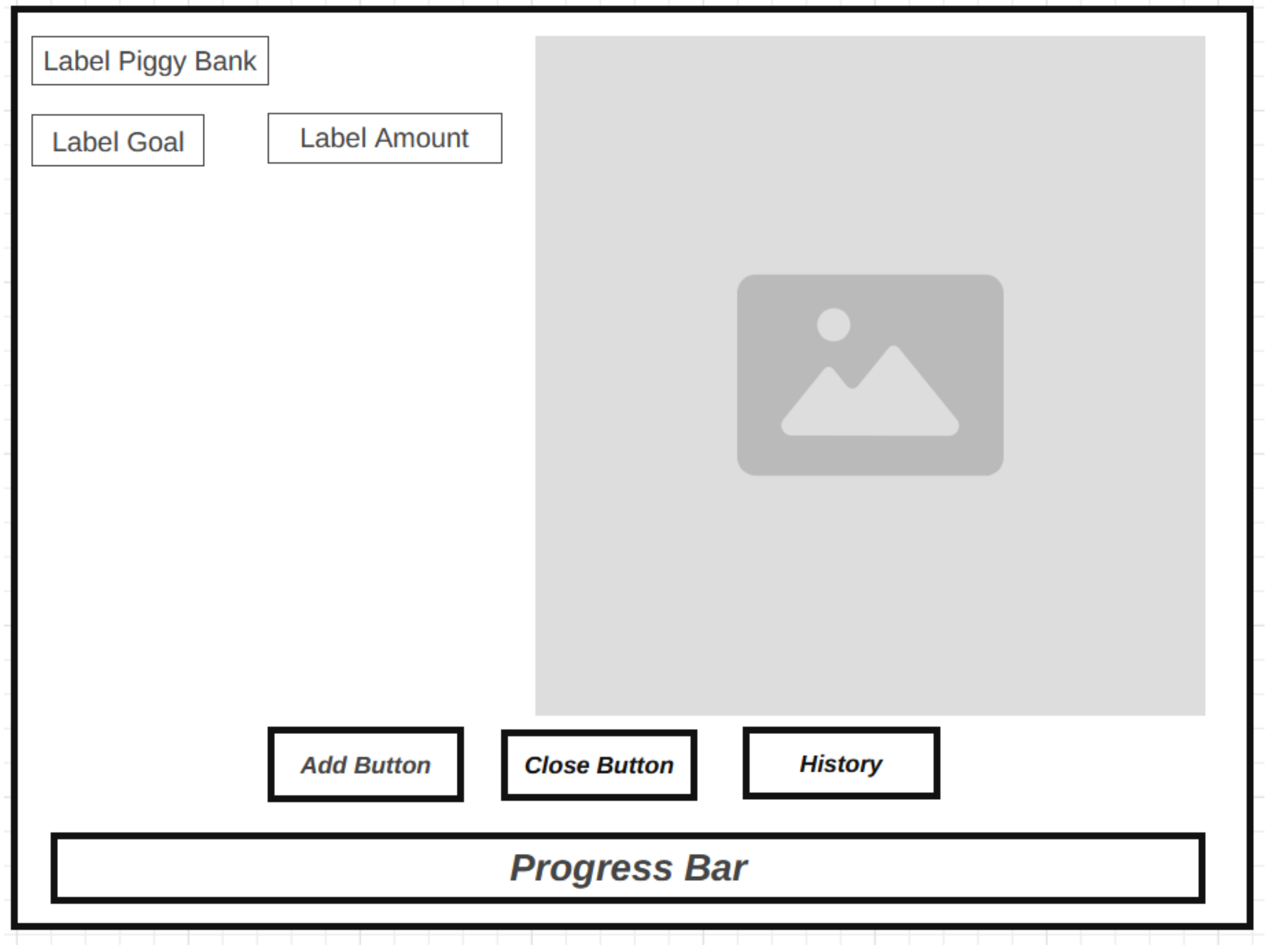
Also, the text area for both the add money and remove money dialog all have text appended to them when the user performs a transaction.

8. What do you envision to be the user base for this application? Justify your response. Describe the typical user of this application.

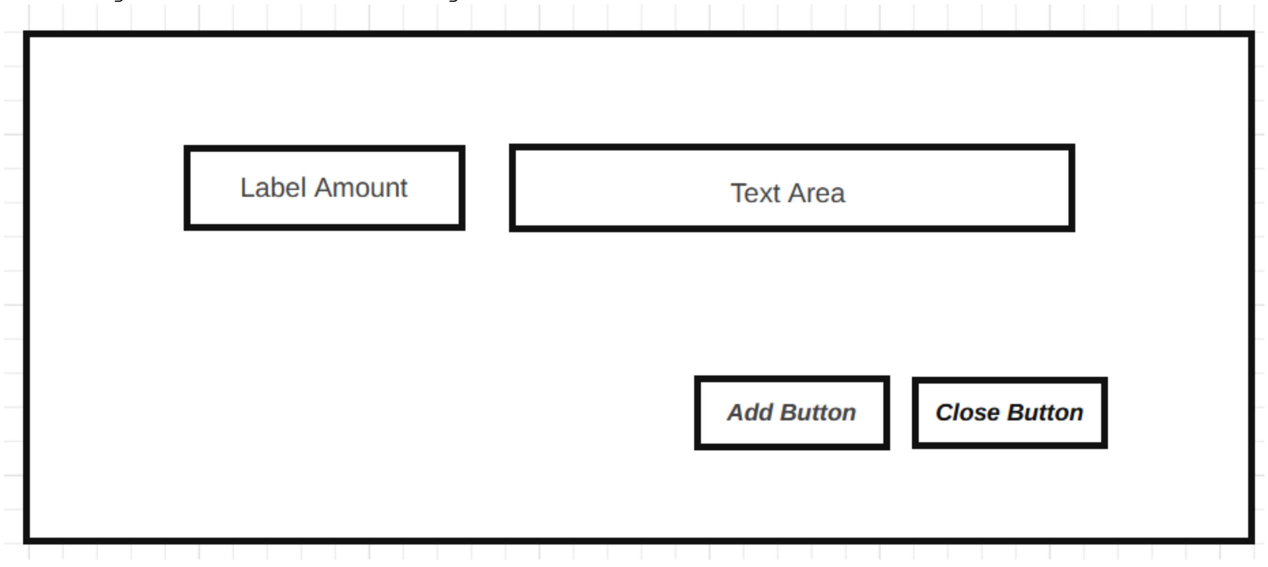
Explanation:
This application is like a smaller of less complex version of a banking system. This application did not have things like interest, loans etc. So, compared to banks and financial institutions, this application is less complex. I can see this application been given to kids to teach them about how the banking works. So, this can be great way to teach kids about the banking system and how to save, withdraw money. What happens when your savings is more than your withdrawal and what happens when your withdrawals is more than your savings. The purpose of receipts and how to keep track of all the products your buy.

9. How would you redesign this application? Draw a new wireframe detailing what you believe the ideal design to be for the PiggyFX application.

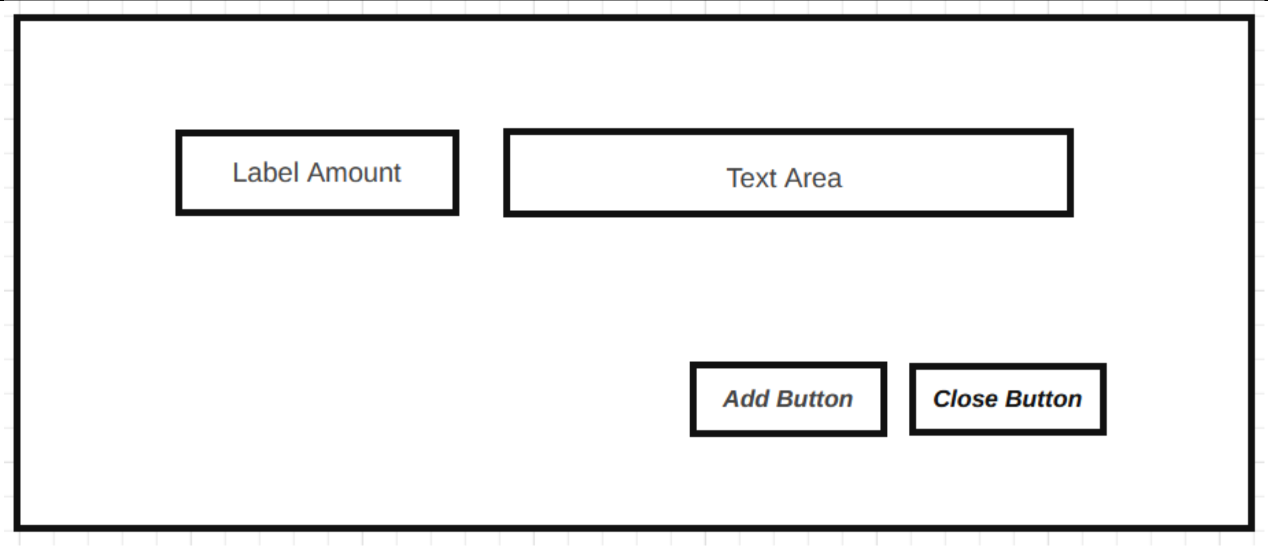
Wireframe and explanation:
Redesign Main User Interface



Redesign Of the Add Dialog



Redesign of the Remove Dialog



10. If you were to continue working on improving this application, what enhancements or updates would you add and why? Briefly explain what you would incorporate in future work, justify your reasoning.

Explanation:
I would implement backing up every transaction made in application to the clou. This will serve as a means of preserving transactions for later use in case something happens to the system. There is no system that is one hundred percent protection proof. So, placing uploading transactions to the cloud will add another layer of security and if something should happen to your machine, you could easily recover because of your backups.

11. If any part of your submission isn't working, please detail this below including relevant screenshots.

Screenshots and explanation: