README FILE

FaceLite Application

ICS108 Project T231

By: Atheer Al-Momtan & Dena Al-Harbi

Content

Classes

Methods

Usage

Technology used:
intellij and basic CSS styling

FaceLite is a JavaFX social app that allows users to create profiles, make changes to them, add friends and pictures, and more. They can then add friends to their profile and share pictures and updates with them. The app also includes features such as search functionality to help users connect with others, and create an event blast. With FaceLite, users can stay connected with their friends and family and share their updates with others.

Class1: Parent

This is the main class, it is where the program starts from.

This class contains the panes and their styling, the buttons, their setOnAction statements, and their text fields and the labels. Moreover, the action to be executed when a button is clicked which is in an inner class called ButtonHandler. It also includes the design of the program itself and handles any error that might happen when the user types wrong input in any of the buttons' text fields.

Variables of Class1

ArrayLists:

```
ArrayList<profileBase> users = new ArrayList<>();
2 usages
ArrayList<String[]> Retriever = new ArrayList<>();
2 usages
ArrayList<String> quotes = new ArrayList<String>()
```

Texts & Labels:

```
String topLast = "", addLast = "", pathLast = "", statusLast = "", addGuestLast = "";
3 usages
Text date = new Text(), welcome = new Text( s: "Welcome"), numOfAccounts = new Text(), updateText = new Text();

2 usages
Label NoimageLabel = new Label( s: "No Image"), usernameLabel = new Label( s: "What is your name (The host)?"),
        1 usage
        nameEventLabel = new Label( s: "What is the name of the event?"), bottomStatus,
        44 usages
        errorLabel = new Label( s: "Error - You have to pick a profile first"), nameLabel = new Label( s: "Name:");
```

Buttons:

```
Button addFriendEvent = new Button( s: "Add Friend to event"), addButton = new Button( s: "Add"), deleteButton = new Button( s: "Delete");

12 usages
Button lookupButton = new Button( s: "Lookup"), DismissButton = new Button( s: "X"), changeStatus = new Button( s: "Change Status");
6 usages
Button changePicture = new Button( s: "Change Picture"), returnButton = new Button( s: "Return to profile"), addFriend = new Button( s: "Add Friend");

5 usages
Button images = new Button( s: "Image Gallery"), statusGenerator = new Button( s: "Quote of the day"), createGroup = new Button( s: "Create an event blast"),
        5 usages
        addEvent = new Button( s: "Add Friend"), submit = new Button( s: "Submit");
4 usages
Button Users = new Button( s: "FaceLite users");
```

Text fields:

```
TextField changePicText = new TextField(), changeStatusText = new TextField(), addFriendText = new TextField(), userHost = new TextField(),
        9 usages
        EventName = new TextField(), addEventText = new TextField(), topText = new TextField();
```

Panes:

```
BorderPane borderPane = new BorderPane();
72 usages
GridPane center = new GridPane();
2 usages
int numberOfAccounts = 0;

17 usages
StackPane PicFrame = new StackPane(), stackPane = new StackPane();
3 usages
HBox horizontal = new HBox( v: 10, nameLabel, topText, addButton, deleteButton, lookupButton), errorMessage = new HBox();
12 usages
VBox vertical = new VBox( v: 30, statusGenerator, createGroup, images, Users, new VBox(addFriendText, addFriend),
        new VBox(changePicText, changePicture), new VBox(changeStatusText, changeStatus)), friendsVbox = new VBox();
```

Methods of Class1

Method #1

```
public void start(Stage stage) throws IOException {
```

The method “start” is responsible of the following:

1. It retrieves data from a text file (FILE_NAME) and stores it in memory. It reads the file line by line using a Scanner, splits each line using the "@" delimiter, and adds the resulting array to a Retriever object. The data represent user profiles.
2. It creates instances of profileBase based on the retrieved data. It initializes the profileBase objects with values from the retrieved data, such as the user's name, profile picture, current status, and friends list.
3. It sets the minimum width for various elements, such as buttons and text fields, to specific values.
4. It sets the onAction event handlers for several buttons by assigning instances of a ButtonHandler class to them.
5. It styles the UI elements using basic CSS styling
6. It sets up event handlers for text fields (addFriendText, changePicText, changeStatusText) so that when the Enter key is pressed, the corresponding button, say, addFriend, is activated.
7. It creates Various types of panes and add the text fields and buttons to them.
8. It creates the scene with it root node “borderPane”
9. It finally shows the stage.

To summarize:

this method is responsible for initializing and setting up the user interface for the FaceLite application, including loading user profiles from a text file, configuring UI elements, and displaying the UI on a stage.

Methods of Class1

Method #2

```
public void display(profileBase user) {
```

The method “display” is responsible of the following:

1. Checks if the profile has a default profile picture and displays it in PicFrame, or displays a placeholder image if there is no picture.
2. Updates the interface for the center pane with along with styling the background.
3. Positions various elements within the gridpane.
4. Updates the style and formatting of specific elements.
5. Displays the profile's friends' names in the friendsVbox.
6. Creates an HBox with several elements and styles it.

To summarize:
the method updates the UI to show the profile information, including the profile picture, friends' names, and status.

Method #3

```
public void ErrorPane(Label errorLabel) {
```

The method “ErrorPane” is responsible of the following:

1. It creates an HBox called errorMessage that contains the DismissButton and the errorLabel passed as parameters.
2. It clears the children of errorMessage.
3. It adds the errorLabel and DismissButton to the errorMessage.
4. It sets the minimum and maximum size of the errorMessage to 200x50 and 300x50 respectively.
5. It clears the stackPane.
6. It adds the center and errorMessage to the stackPane.
7. It sets the stackPane as the center of the borderPane.

To summarize:
the method creates and displays a errorMessage that contains an error message (errorLabel) and a dismiss button (DismissButton). The errorMessage is then displayed in the center of the borderPane.

Methods of Class1

Method #4

```
public void stop() throws Exception {
    //Save user data
    writer.saveData(users);
}
```

The method “stop” is responsible of the following:

1. It is called when the application is closed.
2. It saves user data by calling the saveData method
3. The saveData method is passed the users array list that was previously created.

Method #5

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

The method “main” is where the program starts.

Class2: ButtonHandler

This is an inner class of the class Parent. it implements the EventHandler<ActionEvent> interface.

This class contains the panes and their styling, the buttons, their setOnAction statements, and their text fields and the labels. Moreover, the action to be executed when a button is clicked which is in an inner class called ButtonHandler. It also includes the design of the program itself and handles any error that might happen when the user types wrong input in any of the buttons' text fields.

Variables of Class2

Array of All the pictures' paths in the image gallery

```
String[] paths
```

variables that store the text fields' data

```
topLast = topText.getText();
 addLast = addFriendText.getText();
pathLast = changePicText.getText();
addGuestLast = addEventText.getText();
statusLast = changeStatusText.getText();
```

A button used to check if certain requirements are met

```
 boolean checker;
```

Methods of Class2

Method #1

```
private void copyImagePath(String imgPath) {
    Clipboard clipboard = Clipboard.getSystemClipboard();
    ClipboardContent content = new ClipboardContent();
    content.putString(imgPath);
    clipboard.setContent(content);
}
```

The method “comyImagePath” is responsible of the following:

1. It retrieves the system clipboard using the Clipboard.getSystemClipboard() method.
2. It creates a new instance of ClipboardContent.
3. It sets the content of the clipboard to the imgPath string by calling content.putString(imgPath).
4. It sets the content of the system clipboard to the content object by calling clipboard.setContent(content).

To summarize:

this method copies the provided image path (imgPath) to the system clipboard.

Methods of Class2

Method #3

```
public void handle(ActionEvent e) {
```

The method “handle” is responsible of the following:

1. This method is triggered when an event is fired, and it contains various actions to handle different button events.
2. It checks if certain text fields (topText, addFriendText, changePicText, changeStatusText, addEventText) are empty or not. If they are not empty, it stores the contents in corresponding variables (topLast, addLast, pathLast, statusLast, addGuestLast) and clears the text fields.
3. It initializes a boolean variable checker to true.
4. Two Examples of what action is executed when a certain button is clicked:

1. If the event source is the addButton, it performs the following actions:
It checks if topLast is not empty and if the user is not already in the system. If the user is already in the system, it displays an error message. Otherwise, it creates a new profileBase object with the topLast as the username and adds it to the users array list.
It updates various UI elements, such as nameLabelUpdated, PicFrame, friendsVbox, and center, based on the new user profile data.
It updates the borderPane and displays the updated UI.

2. If the event source is the deleteButton, it performs the following actions:
It checks if topLast is empty and displays an error message if it is.
It searches for a user in the users array list with the same username as topLast. If found, it removes the user from the array list and updates the UI accordingly.
It displays a status message at the bottom of the app indicating the deletion of the profile.

To summarize:
The method class handles various button events in the user interface. It performs actions such as updating the elements, and displaying status messages based on the button events triggered by the user.

Methods of Class2

Method #3

```
public void display(profileBase user) {
```

The method “display” is responsible of the following:

1. Checks if the profile has a default profile picture and displays it in PicFrame, or displays a placeholder image if there is no picture.
2. Updates the interface for the center pane with along with styling the background.
3. Positions various elements within the gridpane.
4. Updates the style and formatting of specific elements.
5. Displays the profile's friends' names in the friendsVbox.
6. Creates an HBox with several elements and styles it.

To summarize:
the method updates the UI to show the profile information, including the profile picture, friends' names, and status.

Class3: profileBase

This class extends the main class parent.

This class represents a user profile and provides methods to access and update various profile-related information such as name, profile picture, status, and friends list.

Variables of Class3

class fields

```
private Label nameLabel;
2 usages
private String nameForButton, picPath, myFriendsRR = "[", finStatus;

2 usages
private ArrayList<String> myFriends = new ArrayList<>();
Label labelFriends = new Label( s: "Friends ");

3 usages
ImageView profileDefault = null;
10 usages
Label statusDefault = new Label( s: "No current status");
```

Constructor

```
public profileBase(String name) {
    this.nameLabel = new Label(name);
    this.nameForButton = name;
}
```

This is an arg constructor that takes the name of the user as an argument and initializes its value.

Methods of Class3

Method #1

```
public Label getLabelFriends() {
    return labelFriends;
}
```

The method “getLabelFriends” is a getter for friends label

Method #2

```
public ImageView getProfileDefault() {
    return profileDefault;
}
```

The method “getProfileDefault” returns the profile’s picture of the user

Method #3

```
public void getProfileDefault(String s) {
    profileDefault = new ImageView(new Image(s));
}
```

The method “getProfileDefault” is taking the path of the image as an argument and setting it as the user’s profile picture.

Method #4

```
public Label getStatusDefault() {
    return statusDefault;
}
```

The method “getStatusDefault” is a getter method that returns the status of the user

Method #5

```
public void getStatusDefault(String s) {
    statusDefault = new Label(s);
}
```

The method “getStatusDefault” is taking the status of the user as an argument and setting it as the user’s new status.

Methods of Class3

Method #6

```
public Label getNameLabel() {
    return nameLabel;
}
```

The method “getNameLabel” is a getter for the user’s name label

Method #7

```
public String getUserName() {
    return nameForButton;
}
```

The method “getUserName” is a getter that returns the name of the user

Method #8

```
public ArrayList<String> getMyFriends() {
    return myFriends;
}
```

The method “getMyFriends” is a getter that returns the list of the user’s freinds

Method #9

```
public void getMyFriends(ArrayList<String> s) {
    myFriends = s;
}
```

The method “getMyFriends” is taking the array list of friends of type string and assigning it to myFriends

Method #10

```
public void getMyFriendsUpdate(String NEW) {
    if (NEW.equals("No-Friends")) {
        myFriendsRR = "No-Friends";
    } else if (myFriendsRR.equals("["))
        myFriendsRR = myFriendsRR + NEW + "]";
    else if (!myFriendsRR.equals("[") && !myFriendsRR.endsWith("]"))
        myFriendsRR = myFriendsRR + "," + NEW;
    else if (myFriendsRR.endsWith("]")) {
        myFriendsRR = myFriendsRR.substring(0, myFriendsRR.length() - 1);
        myFriendsRR = myFriendsRR + "," + NEW + "]";
    }
```

is a method that take a string as an argument ad return or update the friend’s list depending of the string NEW

Methods of Class3

The following methods are methods with the same idea of previously explained methods.

```
public String getMyFriendsRR() { return myFriendsRR; }

1 usage
public void getMyFriendsupdateRR(String s) { myFriendsRR = s; }


3 usages
public void getPicPath(String path) { picPath = path; }

2 usages
public String getPicPathRR() {
    return picPath;

}

8 usages
public void getFinStatus(String stat) { finStatus = stat; }

2 usages
public String getFinStatusRR() {
    return finStatus;

}
```

Class4: writer

This class extends the main class parent.

This class saves the users' data to a file

Variables of Class4

class fields

```
static String FILE_NAME = "data.txt";
```

Methods

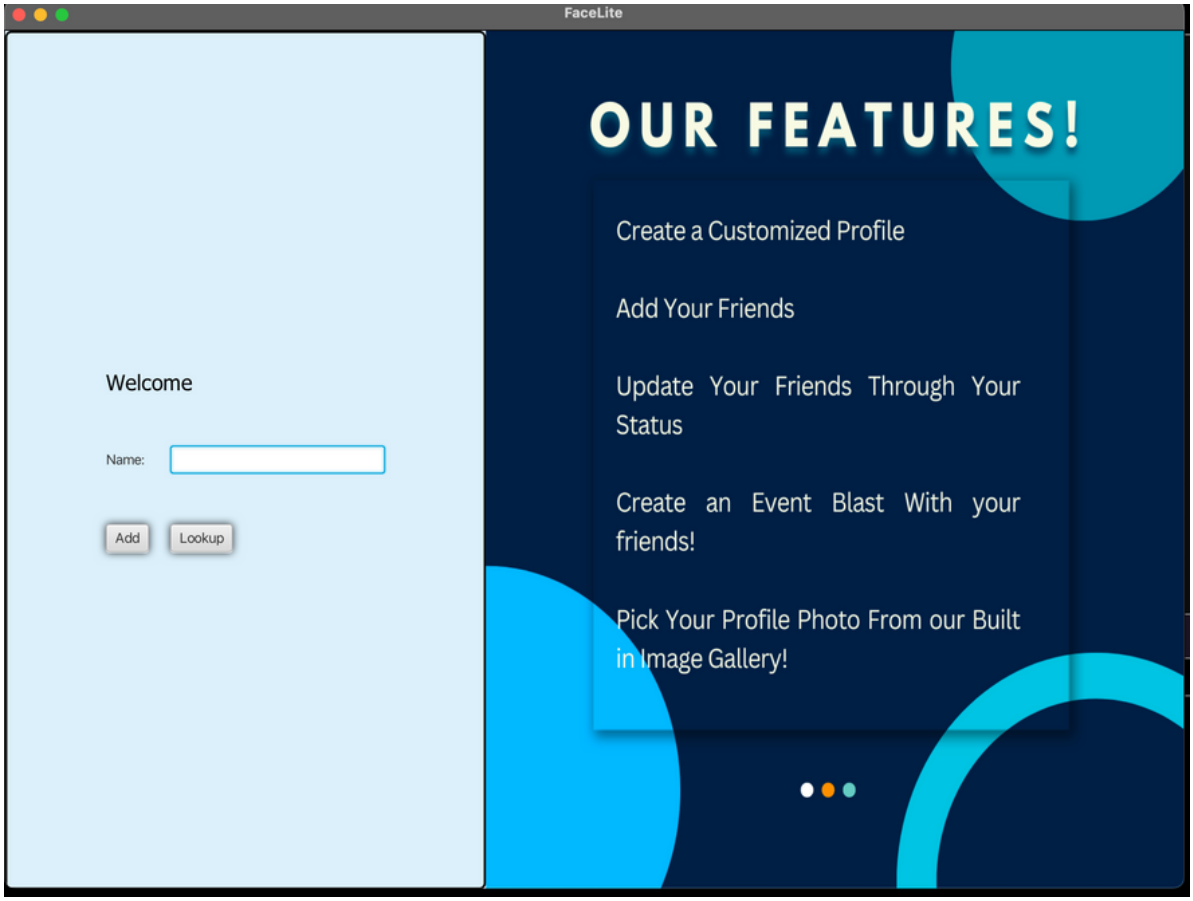
```
public static void saveData(ArrayList<profileBase> users) {
```

This method takes the array list of users as an argument. It then iterates over the list to extract the information of each user and add it to the file data.txt. If the file has previous information, the new information does not override the previous one but rather adds to it. It also adds default values for noncustomized profile information.

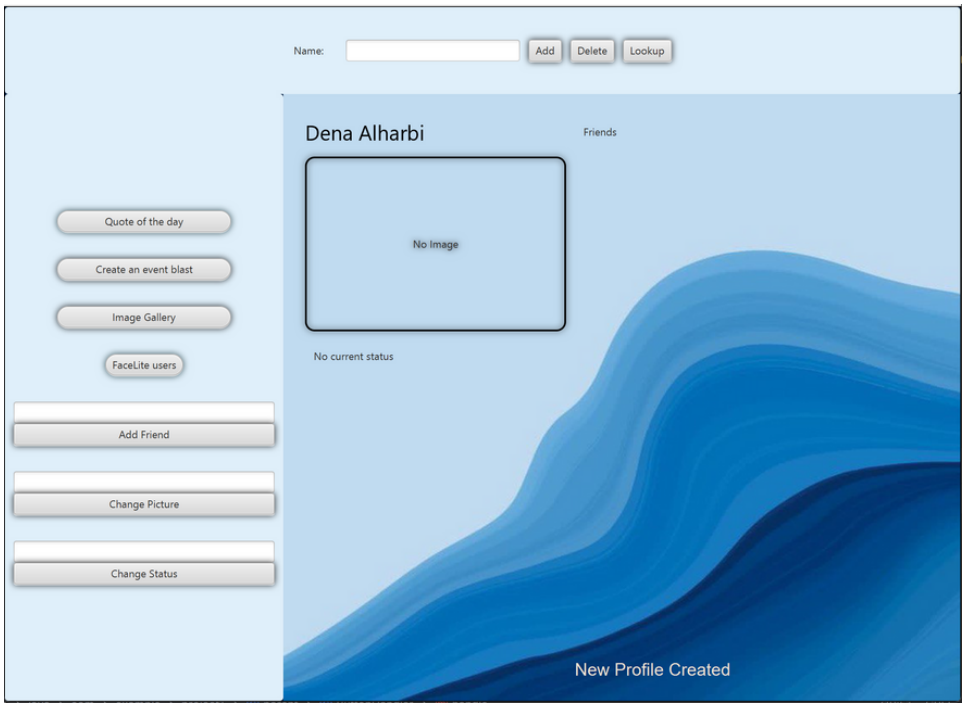
Usage

To use the code, simply open the following java files:
parent, profilebase, and writer. make the sure the images are also in the resources folder.

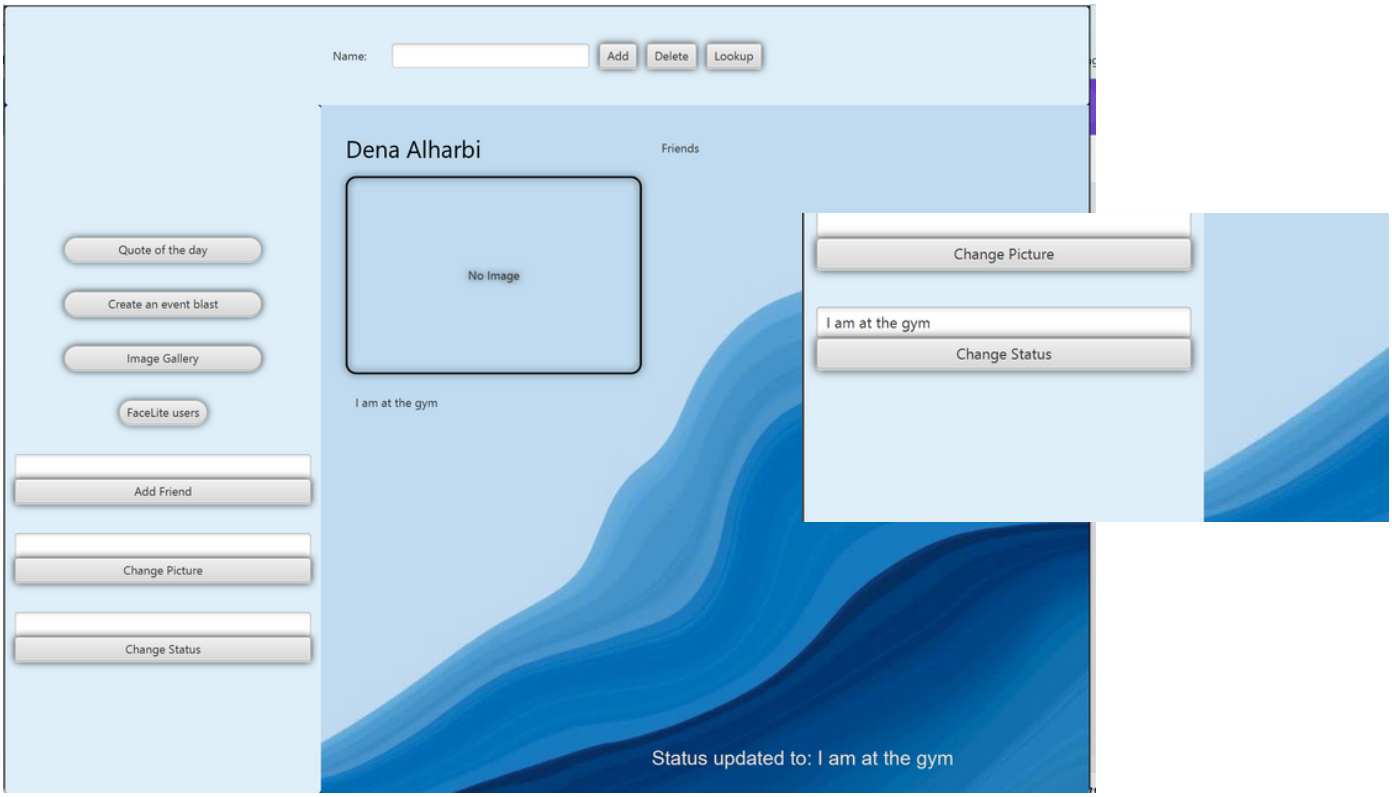
In the welcome pane when the code first runs, the user can either lookup a profile that already exists or create a new one. In the event of looking up a profile that does not exist or creating a profile that does exist an error message will pop up with a button to dismiss the error.



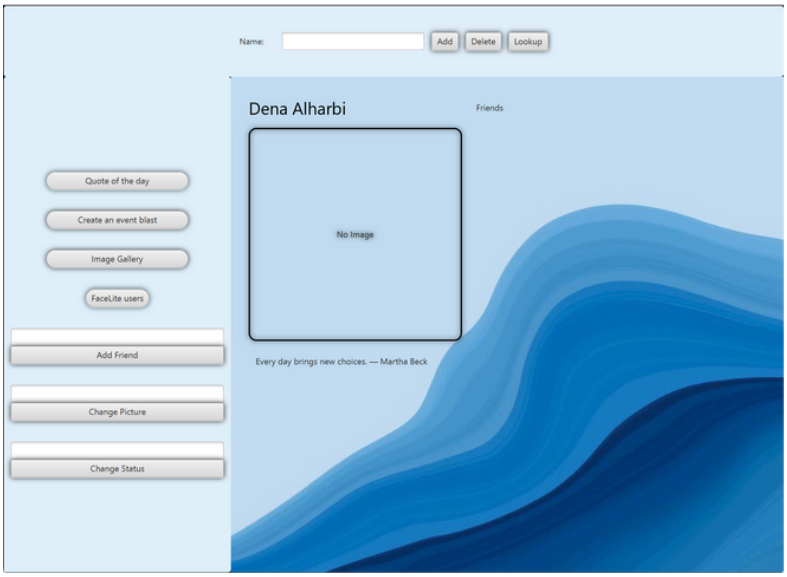
Creating a profile would make this window appear, each user can have a customized profile with a picture, status, and a list of friends. At the left and top of the interface, the user has many buttons that allow the user to make changes.



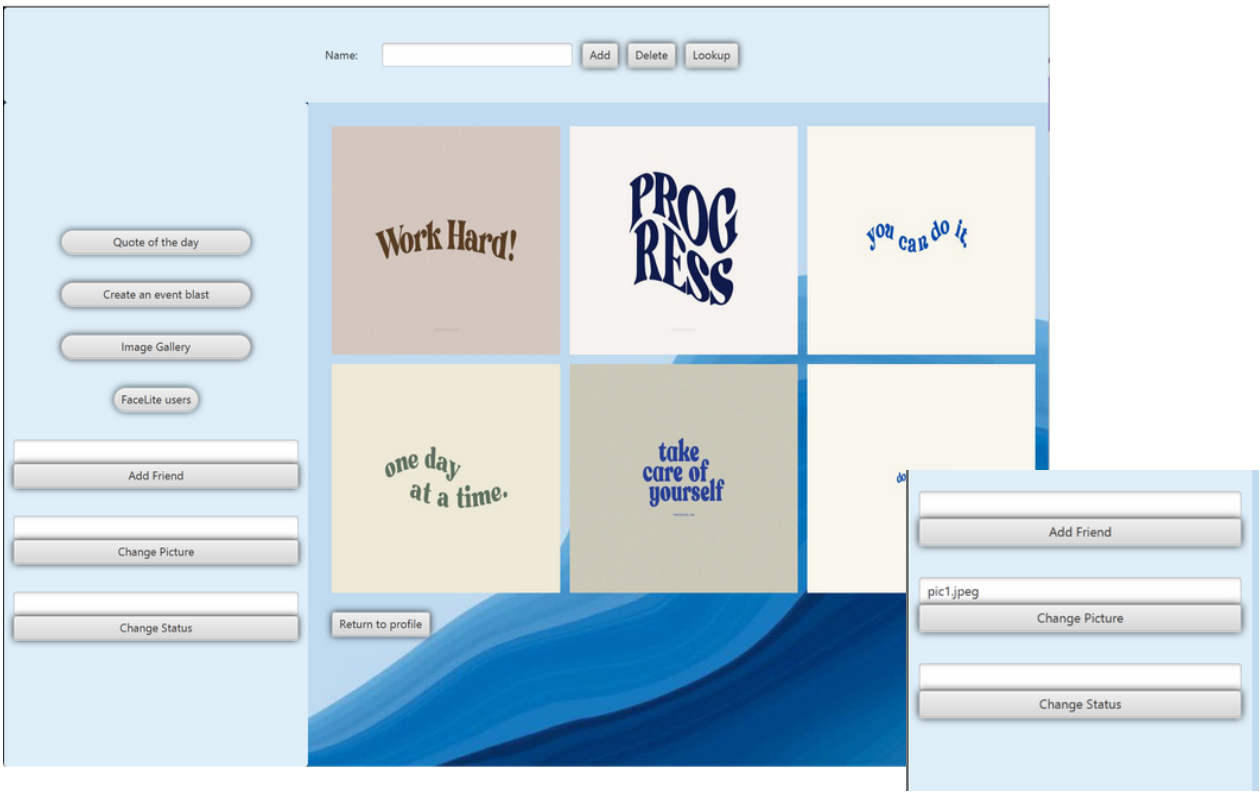
For example, the user can change his status by typing in the text field the preferred status, and then by pressing the button or pressing the enter key on the laptop keyboard the status will automatically be changed and the text field will be cleared up. Also, a message will appear at the bottom stating that the status has changed.



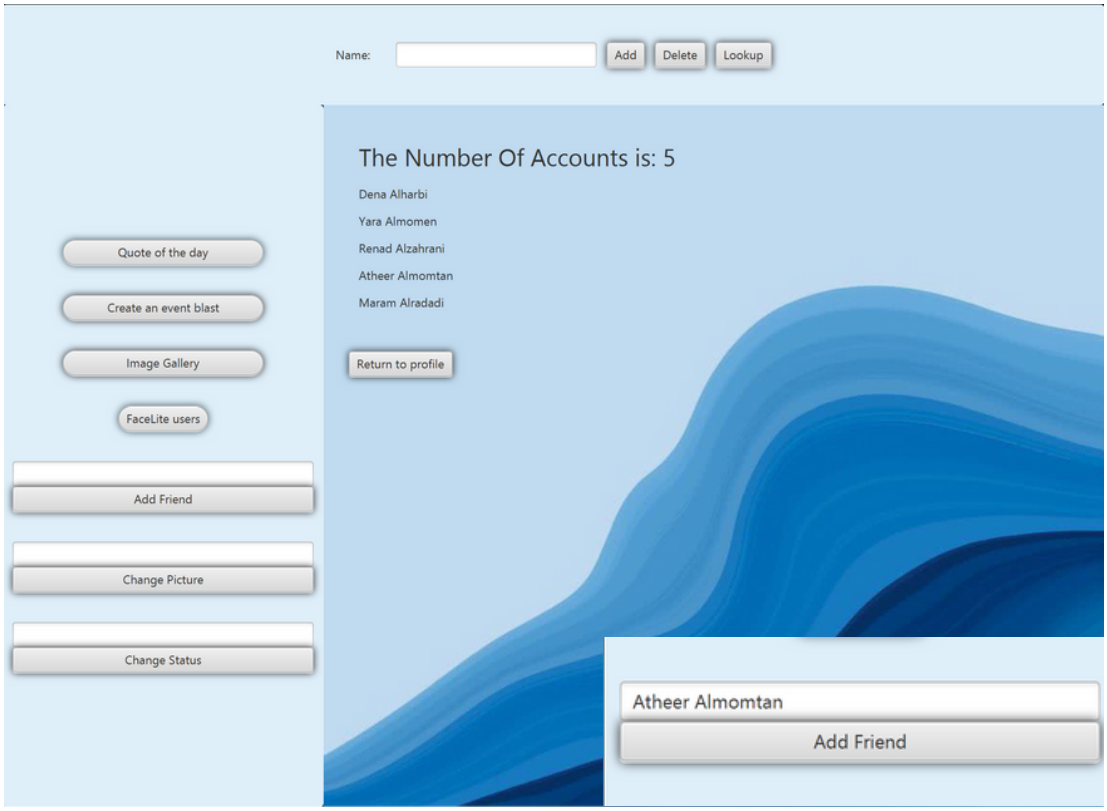
Another way of changing the status is by adding the “Quote of the Day” button, which generates a random positive quote that is automatically updated when pressed to the displayed profile. This allows users who do not have a specific status in mind to have an easier way of choosing one.

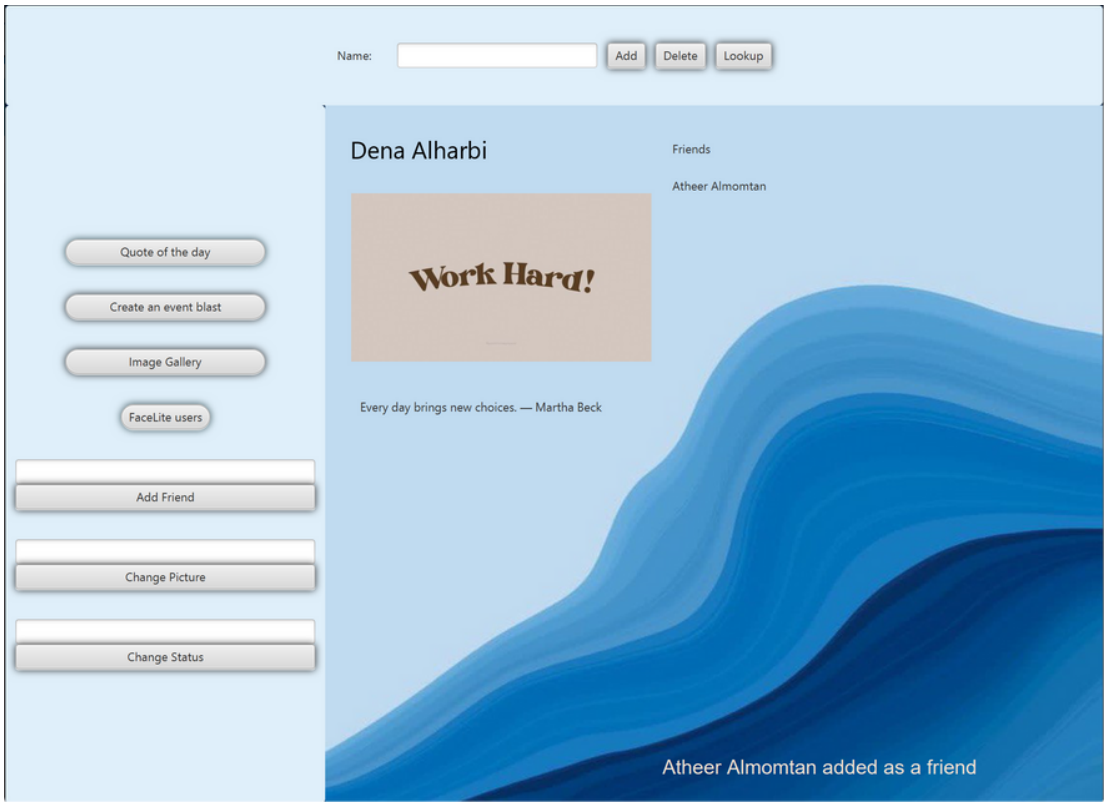


The user also has the option to add a picture or change it, and to make that process easier we added another feature which is called the “Image Gallery” where the user can view the options before changing the picture. The most notable thing is that the picture path can be copied just by mouse-clicking on the picture. Then the path is pasted and when the button or the enter key is pressed the picture will be added to the profile. There is also a “Return to profile” button in this window in case the user decides against changing the picture.

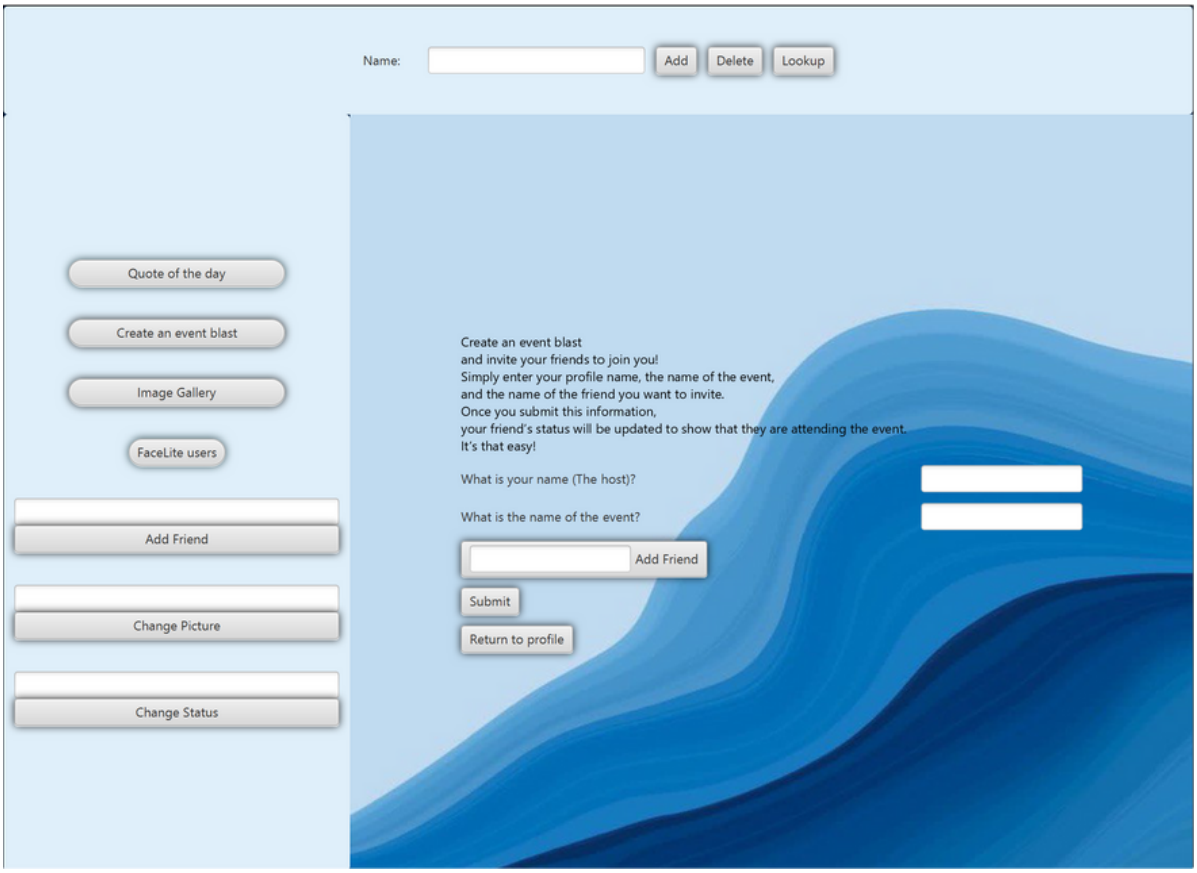


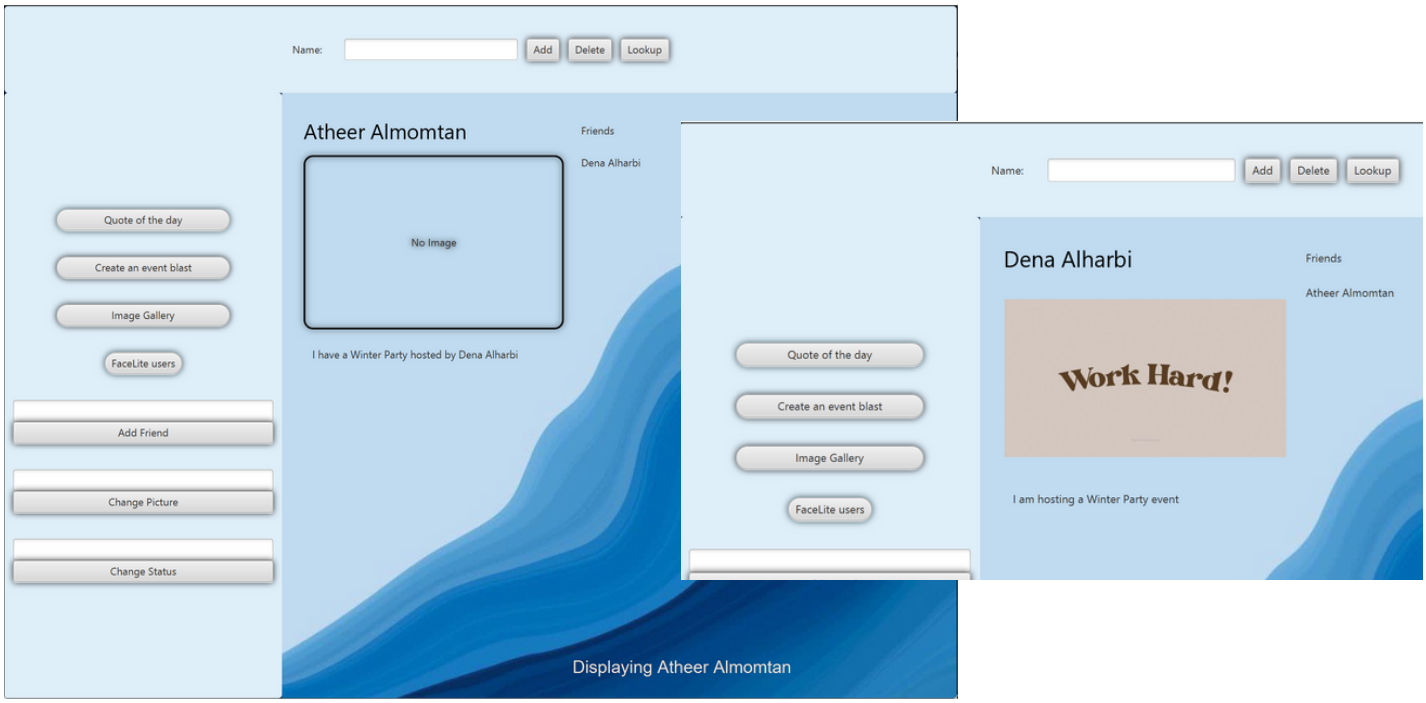
In addition, the user can add a friend or more to his list of friends on the side by typing the name of the profile in the add friend text field, in the case of the user not existing an error message will pop up. In order to make the process easier we added a button “FaceLite users” that displayed the number and names of all the profiles that exist to choose from. When for example “Dena Alharbi” adds “Atheer Almomtan” as a friend, “Dena Alharbi” is automatically added to “Atheer Almomtan” 's list of friends.





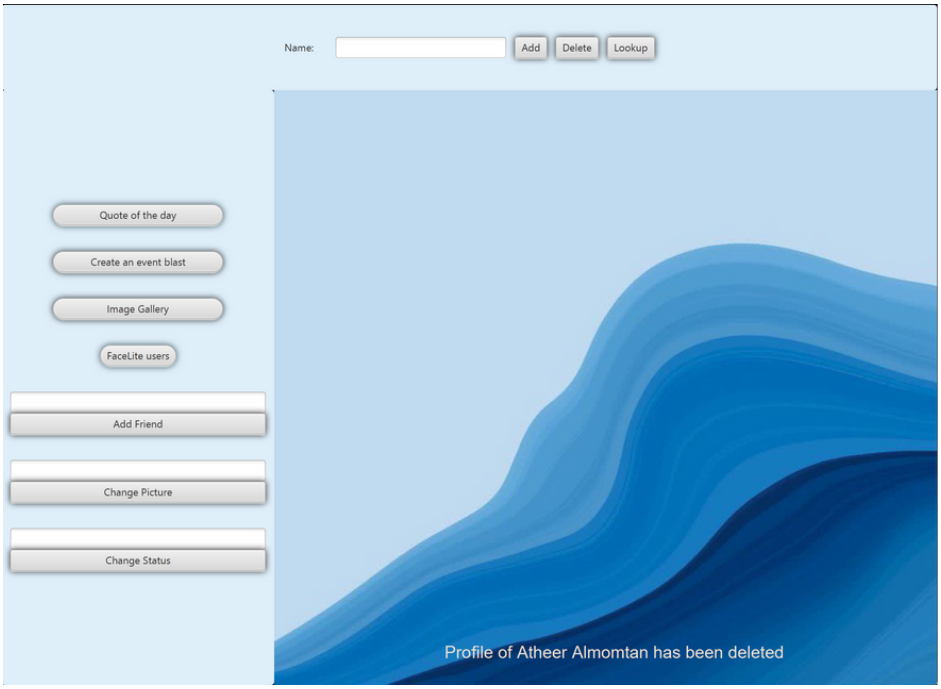
Another notable feature is the “ Create an event blast” option. If the user has an event that he wants to invite friends to he can go to the form by pressing “ Create an event blast” button and filling his name “The host” and the name of the event and add as many friends as needed then pressing the submit button. If any input was wrong an error message will pop up.

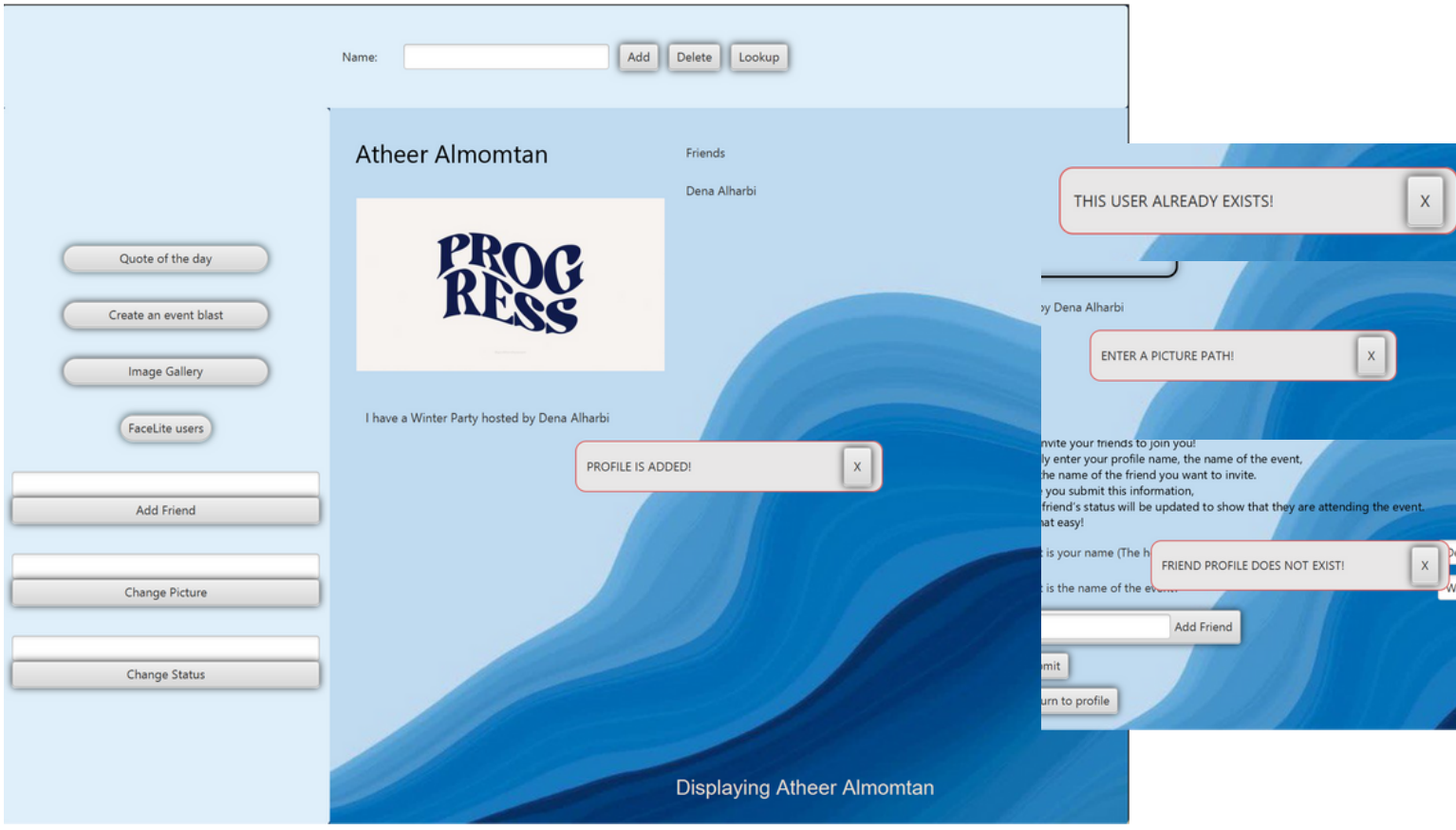




The status of the host and the friends invited will be updated accordingly.

At the top of the app the user has the option of deleting a profile, looking up an old one, or creating a new one by entering the name in name the text field.





Throughout the app, many error messages were implemented to pop up in the middle of the screen whenever the user enters a wrong input. And by clicking the dismiss button the error message will disappear from the screen.