Figure 1: Logo

Content

A social network application called CONNECT.

Project Describtion

CONNECT is simply a project that develops an application manages a straightforward social network.

CONNECT requires its users to create an account by entering a unique name, a name that does not exist before on CONNECT. Moreover, the users can change their status, picture, and even expand their relationships by adding friends existing on CONNECT! Additionally, the users can even pick their positive spot by clicking on a button that displays a random positive quote :)

Project Technologies

This project uses: - JAVA 21 - JavaFX SDK 21 - IntelliJ IDEA CE

Design Overview

To accomplish this remarkable results, the team had to divide the work to two classes: Connect and User.

1. Connect class

In the Connect class, there are several inner classes created and variables initialized within it to do the job, for example:

1.1 Variables:

- Stage stage;
- Scene connectScene;
- BorderPane borderPane;
- FlowPane flowPane;
- VBox vBox;
- TextField textFieldImage, textFieldStatus, textFieldName, textField-AddFriend;
- Label userNameLabel, statusLabel, friendsLabel, mainLabel, added-FriendLabel;
- Button statusChangeButton, imageChangeButton, addFriendButton, ad-dButton, deleteButton, lookupButton, quotesButton;
- ImageView imageView;
- static TreeSet setOfUsers = new TreeSet<>();
- static TreeSet setOfNames = new TreeSet<>();
- String userName;
- boolean profileIsCreated = false;
- public static File dataFile = new File("data.txt");

1.2 Methods:

- The **main** method in our code is the start point of the application. It reads user data from a file using readFromFile method, launches the application, and writes user data back to the file after the application stops using WriteInFile method.
- The **start** method is used to set up the two scenes we have. We also used it to set up the panes , vbox and hboxes we have, we also added a sound that launches when we open the application . And lastly we initialized and created the buttons we have in the application and connected them with their handlers using .setOnAction
- The **displayUI** method is made to update the user interface when a new profile is created or when an existing profile is displayed. We added in it several elements such as username , status and friends labels and images,

based on the user's choice of data (these elements are added in a FlowPane). It also contains a HBOX that is used for the mainLabel and can be changed when needed.

- The **writeInFile** (TreeSet listOfUsers) is a method that receives a set of users and save their information in an external file.
- The **readFromFile**(File file) is a method that receives a file object that contains users information and save the information in setOfUsers and setOfNames. while running the program, extracting data from collection is much faster and more efficient, and this is the main objective of this method.
- The **getUser** method is a method made to return a user object based on a provided username. It iterates through the setOfUsers to find the user with a matching username and it returns that user's object. If no matching user is found, it returns a new user object with the provided username.
- The **displayAlert** (Button button, boolean disable, String title, String content) is a method that receives a button object, boolean value, and two strings. The boolean value, if it was true, functions as in case no profile is created yet and the users tried to use the application features then it will prevent them of doing so. Additionally, the title and content are for what we want to display in the error alert in many different sitiuations.

1.3 Inner classes:

- The **NextSceneHandler** class is designed to handle the action event when the user presses next button in the welcoming scene. Inside the handle method, it sets the scene of the main stage to the connectScene which will change the scene to the next scene which contains the UI
- **CreateProfileHandler class**, which is the handler for the addButton, was created specially to deal with users interactions when adding their name and show the default state to the program when the profile is first created.
- **ChangeImageHandler class**, which is the handler for the imageChangeButton, was created to enable the user to change the picture of the profile through browsing their desired photo, and to ease it more, only allowed image extensions were specified in the code already.
- **ChangeStatusHandler class**, which is the handler for the statusChangeButton, was created to allow the users to change their status by typing their current status in the textfield and update it by pressing change status button.
- **AddFriendHandler class**, which is the handler for the addFriendButton, was designed in order to handle how the users add their friends' profiles

and to check that the friend that the user wants to add has already a profile in CONNECT.

- **LookupHandler class**, which is the handler for the lookupButton, was made to enable the user to search for other users by their name and be able to view their profiles information.
- **DeleteProfileHandler class**, which is the handler for the deleteButton, was created to provide the users with the choice of deleting their account if they no longer want it.
- **PositiveQuoteHandler class**, which is the handler for the quotesButton, was created to display a positive quote to the user if they clicked on pick positive quote button.

2. User class

In the User class, there are several methods created and variables initialized within it to do the job.

2.1 Variables:

- private final String userName;
- private String status = "No current status";
- private String imagePath = "noImage.png";
- public TreeSet setOfFriends = new TreeSet<>();
- PrintWriter output = new PrintWriter(dataFile)

2.2 Methods:

- **User constructor** which increment the number of users and set the name of the user to the name entered.
- **getUserName** which is a getter method that returns the user name.
- **setStatus** (String status) is a setter method that sets the user status.
- **getStatus** which is a getter method that returns the user status.
- **setUserImage** (String imagePath) is a a setter method that sets the user image.
- **getUserImage** is a getter method that returns the user image as an Image object.
- **toString** is an overridden method to return a string representing the name, status, image path, and friendsAsString, the set of friends written in (csv) format by iterating over the set using StringBuilder.

- **compareTo** (User user2) is an overridden method from comparable interface that compares the the current user with an other user based on their names. # Demo

  // A method to read from the data file that contains users information in order to use them while running the program public static void readFromFile(File file) throws FileNotFoundException { try (Scanner scanner = new Scanner(file)) { while (scanner.hasNext()) { // read each line and assign it to an array of string by splitting the string line by commas String[] line = scanner.nextLine().split(“,”);

```
          // add the userName, index[0], in the set of userNames
          setOfNames.add(line[0]);

          for (String userName : setOfNames) {
              // create a new user with this userName
              User newUser = new User(userName);

              // Add this user to the listOfUsers
              setOfUsers.add(newUser);

              // set her/his status, index[1]
              newUser.setStatus(line[1]);

              // set her/his image, index[2]
              newUser.setUserImage(line[2]);

              // add her/his friends to setOFFriends, index[3] until the end
              TreeSet<String> friends = new TreeSet<>(Arrays.asList(line).subList(3, li
              newUser.setOfFriends.addAll(friends);

          }
      }
  }
  catch (IOException e){
      System.out.println(e.getMessage());
  }
```

  }

  // A method to write in the file in order to store the data after closing the program public static void writeInFile(TreeSet setOffUsers){ try (PrintWriter output = new PrintWriter(dataFile)){ for (User user : setOffUsers){ output.println(user.toString()); } } catch (FileNotFoundException exception){ System.out.println(“no such file”); } }

  // A method that returns User object by specifying user’s name private static User getUser(String userName) { User user1 = new User(userName);

for (User user : setOfUsers) { if (user.getUserName().equals(userName)) { user1 = user; } } return user1; }

// A method to display an alert when needed public static void displayAlert(Button button, boolean disable, String title, String content){ button.setDisable(disable); Alert alert = new Alert(Alert.AlertType.ERROR); alert.setTitle(title); alert.setHeaderText(null); alert.setContentText(content); alert.showAndWait(); } # How to use CONNECT? 1 - Click on the 'NEXT' button to start the program.

2 - Enter your name in the upper text field between 'Name' label and 'Add' button and click on "Add" button in order to create your CONNECT profile.

3 - Make sure you add your name, with at most 20 characters, and you see 'New Profile Created' at the bottom of the page, otherwise the rest of buttons will be disabled to be clicked on.

4 - Feel free to update your status by typing whatever you want in the most upper text field in the left and clicking on 'Change Status' button.

5 - Set your profile image by clicking on the 'Change Image' button and select a photo from your files, you can ensure that you select the correct image by checking the photo name in the textfield.

6 - Expand your network and add your friends by typing a friend name in the most lower text field and clicking on 'Add friend' button.

7 - Try to write your friend's name in the the upper text field between 'Name' label and 'Add' button and then click on 'lookup' button, now you can see you become a friend in your friend friends' list!

8 - Click on 'pick positive quote' to pick your today's positive spot.

9 - Delete your profile by clicking on 'Delete' button.

Screenshots

What is displayed when launching the program:

This error occurs when trying to press a button related to user properties, in this case change status, before creating a profile. Notice that this button becoms disable to be clicked!

After creating a profile, the user can change the default photo by picking an image file from the device library, notice that only image extensions could to be chosen!

After creating the profile, the change status button is now able to be clicked!

The user can add friends that exist on CONNECT by typing the friend's name in the textfield above 'add friend' button and then either pressing enter from

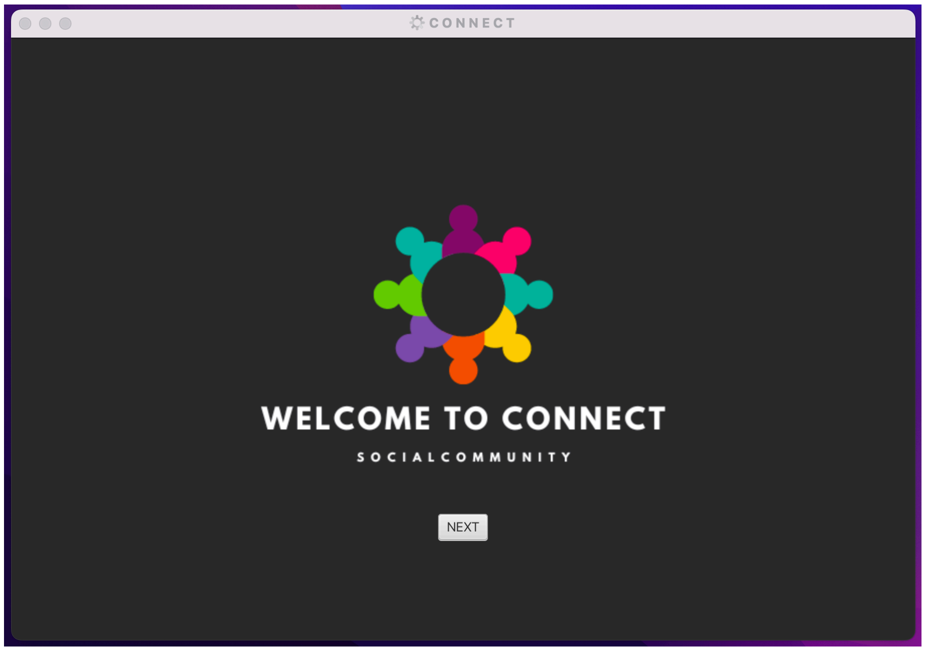
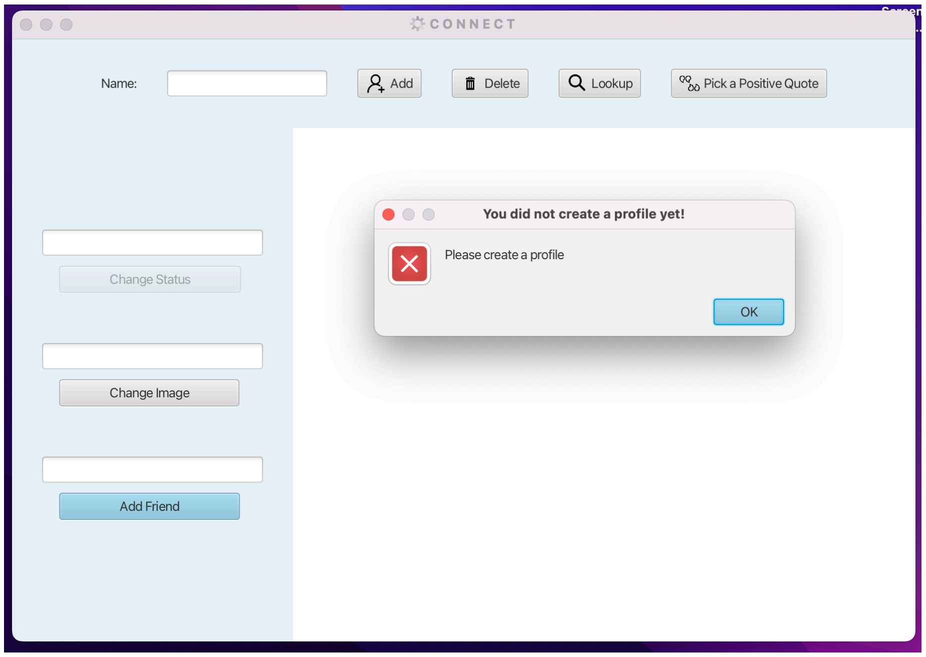


Figure 2: welcoming scene



You did not create a profile yet!

Please create a profile

Figure 3: when changing status before creating a profile

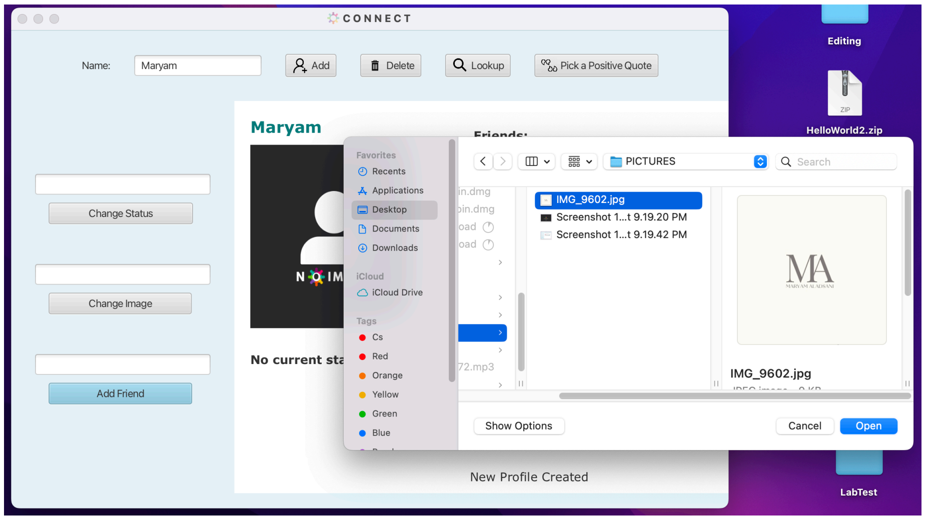


Figure 4: Choosing a profile image

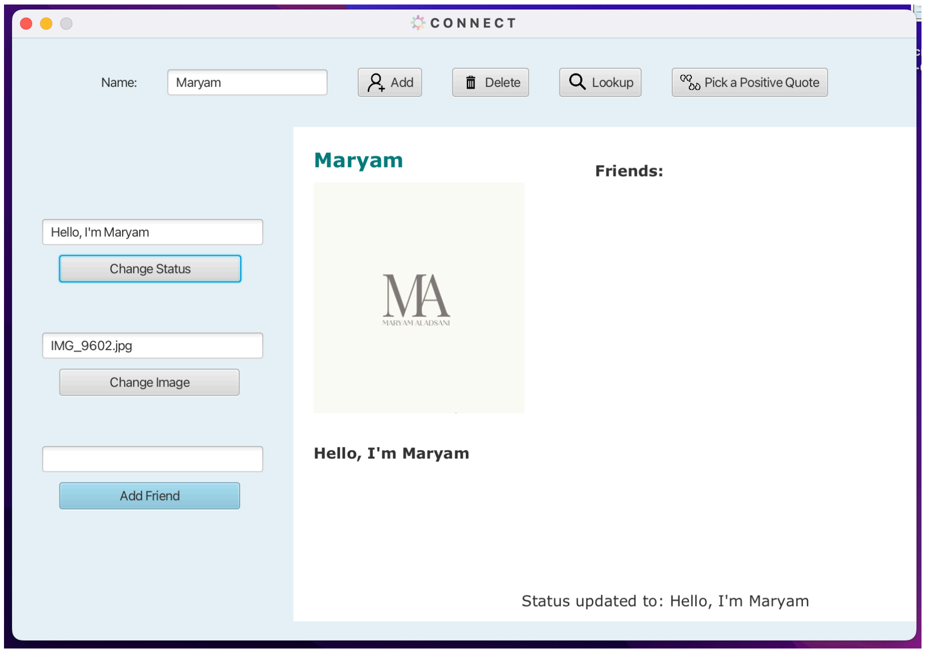
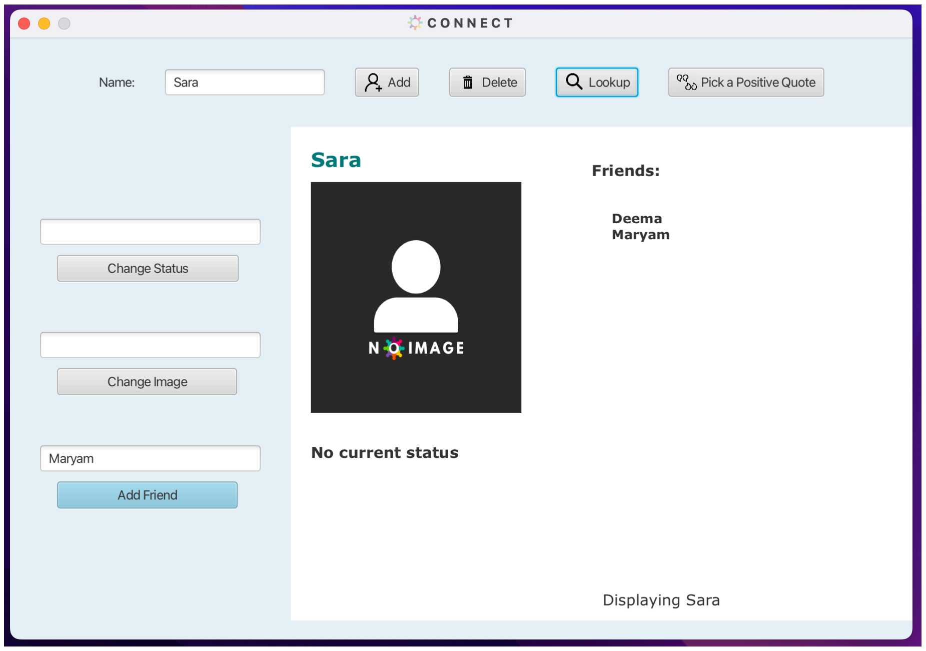


Figure 5: changing status after creating a profile

the keyboard or simply clicking the ‘add friend’ button. The name of friends are shown in the UI even when lookup for the user later.



Sara

Friends:

Deema
Maryam

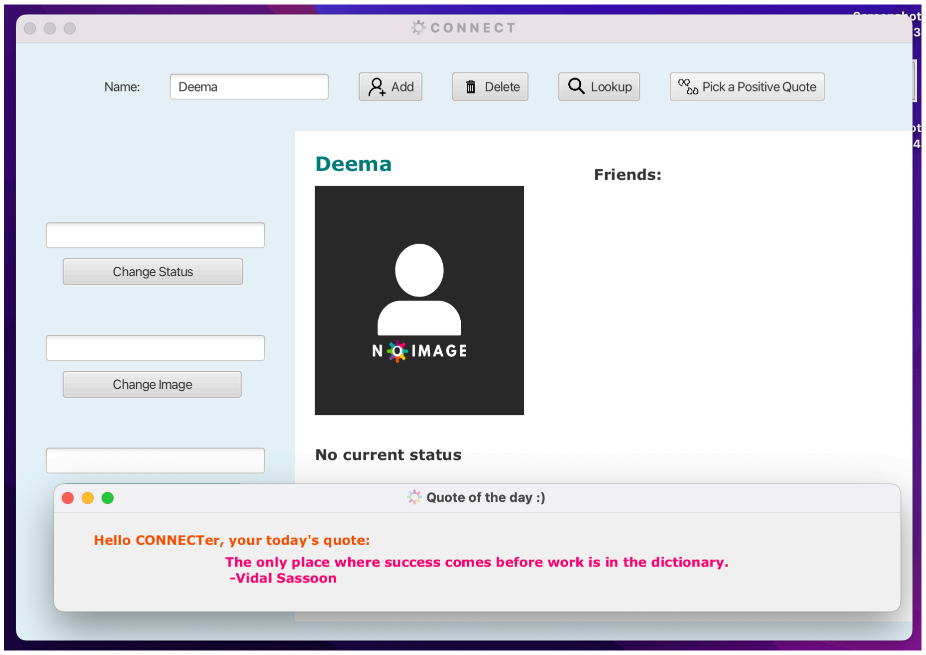
No current status

Displaying Sara

Figure 6: Adding friends

A random quote displayed to the CONNECT user when ‘pick a positive quote’ button is clicked!

And finally, profiles can be deleted by typing the profile name in the upper textfield and clicking delete button.

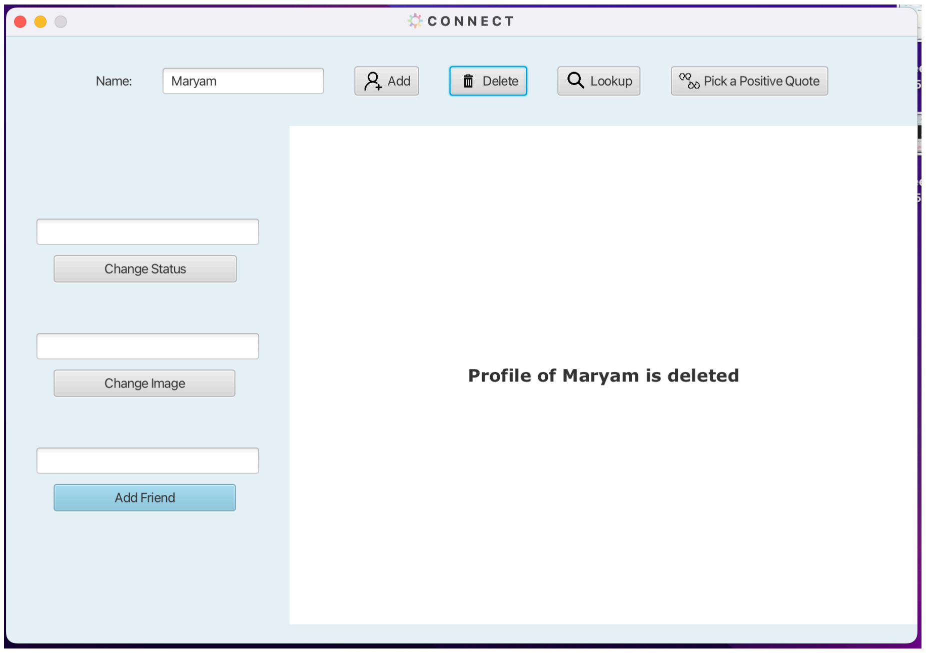


Deema

No current status

Hello CONNECTer, your today's quote:
The only place where success comes before work is in the dictionary.
-Vidal Sassoon

Figure 7: Quotes



Profile of Maryam is deleted

Authors

- @Maryam Aladsani
- @Deema Almousa
- @Sara Esmaeil