

Comp 132 Project

Emirhan Çakır 76409

Contents

Overview of Class	2
Login Page.....	2
Login Page: Login Execution	2
Login Page: Sign in Execution	3
Homepage	4
Homepage: Create Content.....	5
Homepage: Display Content.....	5
Profile Page.....	6
Profile Page: Display Content	7
Profile Page: Edit, Remove Content	7
Profile Page: Display Followers, Following List.....	8
Profile Page: Display Groups	9
Profile Page: Modify Profile.....	10
Profile Page: Log Out Account.....	10
Profile Page: Delete Account.....	11
Profile Page: Create Group.....	11
Group Page	12
Group Page: Display Group Information	12
Group Page: Leave.....	12
Group Page: Edit Group and Display Users	13
Notes	13
References	13

Overview of Class

Content

Content class hold information and methods of contents.

Group

This class holds all information and methods could be used groups.

GroupPage

GroupPage class has all information and methods could be used in group pages.

launcher

This class is made to launch program.

LoginPage

LoginPage class holds all information and methods of launch page It enables to login or sign in.

Mainpage

This class holds all information and methods will be used in main page.

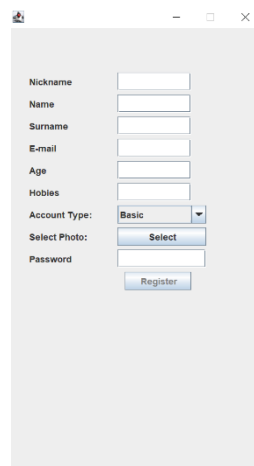
Profile

Profile class holds all information and methods which can be used in profile page.

User

This class holds all information and methods about users.

Login Page



A registration form with the following fields and controls:

- Nickname:
- Name:
- Surname:
- E-mail:
- Age:
- Hobbies:
- Account Type: (dropdown menu)
- Select Photo:
- Password:
-



Users are welcomed with this frame. They have two options in this step. They can sign in or log in. Both buttons activate different executions.

Login Page: Login Execution

```

if (e.getSource() == loginBut) {
    user = User.findUser(String.valueOf(passwordField.getPassword()), nickField.getText());
    if (!user.equals(null)) {
        successLabel.setText("Successfull");
        Mainpage mainframe = new Mainpage();

    } else {
        successLabel.setText("Try again!");
    }
}

```

After login button is activated, program enters the if block which is given below. findUser() method checks whether the password and nickname matches in database. If findUser() finds any user it returns User object. Program set static user variable to this returned object to use other executions. If user is null, program understand that password or nick field is incorrect. findUser() method is explained below

```

/**
 * This method is used to check password and nickname whether they are entered
 * correct or not.
 *
 * @param password - entered password in login page
 * @param nickname - entered nickname in login page
 * @return User or null is returned after this method. The returned User used to
 *         hold information of logged user.
 */
public static User findUser(String password, String nickname) {
    for (User user : allUsers) {
        if (user.nickname.equals(nickname)) {
            if (user.password.equals(password)) {
                return user;
            }
        }
    }
    return null;
}

```

Login Page: Sign in Execution

```

} else if (e.getSource() == signBut) {

    frame.getContentPane().remove(LoginPanel);
    getSignInPage();
    frame.getContentPane().add(signinPanel);
    frame.invalidate();
    frame.repaint();
    frame.setVisible(true);
}

```

If sign button is activated, it removes login panel and call getSignInPage(). Sign in panel is created in getSignInPage() and added to frame in if block. Frame is invalidate and repaint after panel added. getSignInPage() is shown and explained below.

```

/**
 * getSigninPage()- method prepare the frame for sign in operations.
 */
public void getSigninPage() {
    signinPanel = new JPanel();
    signinPanel.setLayout(null);

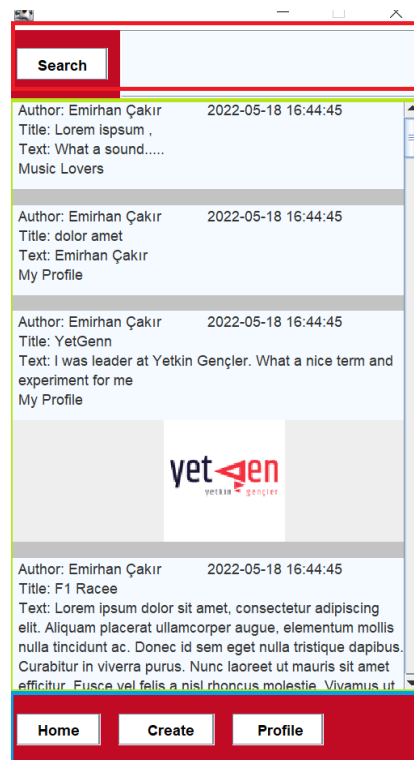
    nicklabel = new JLabel("Nickname");
    nicklabel.setBounds(30, 60, 120, 25);
    signinPanel.add(nicklabel);

    nickField = new JTextField();
    nickField.setBounds(150, 60, 100, 25);
    signinPanel.add(nickField);

    namelabel = new JLabel("Name");
    namelabel.setBounds(30, 90, 120, 25);
    signinPanel.add(namelabel);
}

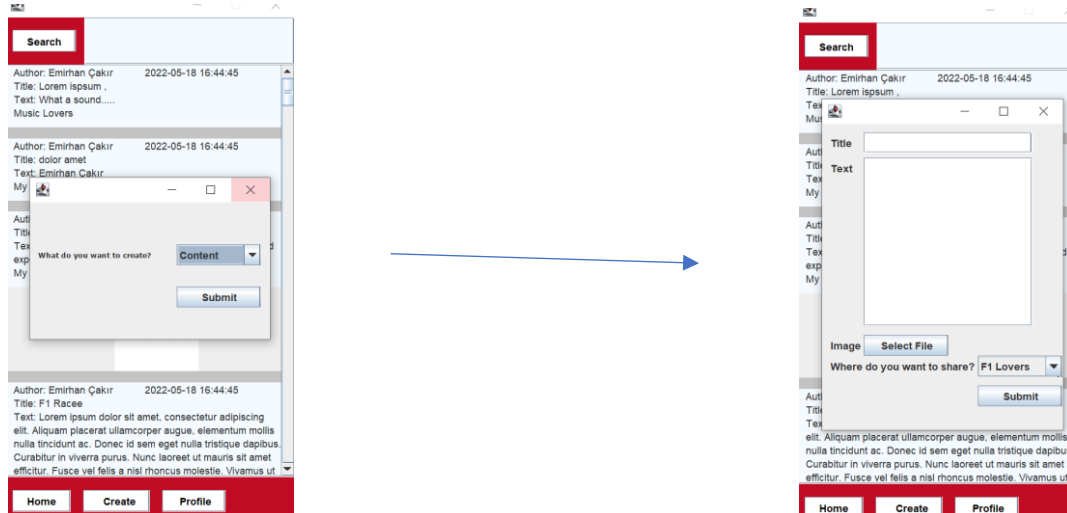
```

Homepage



After login homepage welcomes users. Homepage is consist of three panel. The top panel contains search field and button. Center pane is scroll pane which enables to hold all contents. This panel is changing at group, profile and main page. Bottom panel enables us to visit profile page, create group or content or visit home page.

Homepage: Create Content



After pressed create button new frame is opened via `openCreationOptions()` method, it ask users what they want to create. They have two options to create group or content. Program collect information via text fields, combo boxes.

```
// It calls openCreationOptions() method
else if (e.getSource() == create) {
    openCreationOptions();
}

// This method construct a new frame which shows options to create. They are
// basically contents and groups.
private void openCreationOptions() {
    CreationOptionsFrame = new JFrame();
    CreationOptionsFrame.setDefaultCloseOperation(CreationOptionsFrame.DISPOSE_ON_CLOSE);
    CreationOptionsFrame.setSize(300, 200);

    JLabel creationOption = new JLabel("What do you want to create?");
    creationOption.setFont(new Font("Helvetica", Font.BOLD, 12));
    creationOption.setBounds(10, 50, 150, 25);
    CreationOptionsFrame.add(creationOption);

    String[] contentOptions = { "Content", "Group" };
    JComboBox creationOptionsBox = new JComboBox(contentOptions);
}
```

Above codes prepare panel to create content. Below "else if" block create content according to the values of selected image. If image is selected it has path to address image.

```
// It creates new content
} // It calls openCreationOptions() method
else if (e.getSource() == submit) {
    if (!(Content.allContentTitles().contains(titleField.getText()))) {
        Content newContent;
        if (image == null) {
            newContent = new Content(titleField.getText(), currentUser.getNickname(), textArea.getText(),
                shareoptionList.getSelectedItem().toString(), currentUser);
            currentUser.addContent(newContent);
        } else {
            newContent = new Content(titleField.getText(), textArea.getText(), image.toPath(),
                shareoptionList.getSelectedItem().toString(), currentUser);
            currentUser.addContent(newContent);
        }
    }
}
```

Homepage: Display Content

When midPanel -which is at the center of frame- is wanted to create putContents() method is called by at JScrollPane constructor. putContents() creates an ArrayList takes all contents of attended group and followed users, after that

```
// creates midPanel
midPanel = new JScrollPane(putContents(), JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);
midPanel.setPreferredSize(new Dimension(168, 528));
frame.add(midPanel, BorderLayout.CENTER);
```



```
/**
 * This method calls when JScrollPane is started to construct. It creates a
 * panel which includes all contents of attended group or the user which is
 * followed by current user.
 * @return content - content is JPanel which holds all contents whose we follow
 * or attended.
 */
public JPanel putContents() {
    JPanel content = new JPanel();
    content.setLayout(new BorderLayout());
    ArrayList<Content> allContents = new ArrayList<>();
    allContents.addAll(currentUser.getAllContent());
    for (Group group : currentUser.getGroupList()) {
        allContents.addAll(group.getContent());
    }
}
```

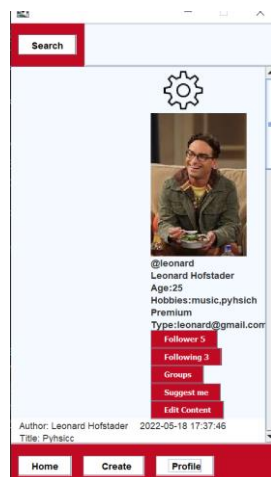
After adding all contents to ArrayList, putContent() method creates JTextAreas for each content and add them to content panel. After adding all contents putContent() method returns content panel. This panel is used to create scroll pane during initialization of midPanel.

```
//It creates text area for each content and put them to content panel which will be added to scroll pane
for (Content eachcontent : allContents) {
    JTextArea contentText = new JTextArea(String.format("Author: %s \t %s \nTitle: %s\nText: %s \n%s",
        eachcontent.getAuthor(), eachcontent.getTimestamp().toString().substring(0, 19),
        eachcontent.getTitle(), eachcontent.getText(), eachcontent.getLocation()));

    // arrange features of contentText
    contentText.setMargin(new Insets(0, 10, 10, 30));
    contentText.setBackground(new Color(244, 250, 255));
    contentText.setLineWrap(true);
    contentText.setWrapStyleWord(true);
    contentText.setEditable(false);

    content.add(contentText);
}
```

Profile Page



The main difference between homepage and profile page is center panel. Bottom and top panel is accessible through the application. After profile button is activated, it removes center panel and creates profile object.

```
@Override
public void actionPerformed(ActionEvent e) {
    // It opens profile page
    if (e.getSource() == profilebut) {
        frame.remove(midPanel);
        Profile refreshProfile = new Profile();
    }
}
```

Profile Page: Display Content

In profile constructor scroll pane calls putMyContents() method which shares common features with putContent() method. The difference between these two method. putMyContents() also put some buttons -followers, following, groups, suggest me and edit content buttons-

```
/**
 * The constructor basically prepare system to put profile elements. Firstly it
 * removes possible panels and adds related scroll pane.
 */
public Profile() {
    try {
        frame.remove(groupPage.groupScrollPane());
    } catch (NullPointerException e) {
        frame.remove(profileScrollPane());
    }

    if (!ScrollPane.isScrollPane(profileScrollPane())) {
        // ScrollPane is created
        profileScrollPane = new JScrollPane(profileContents(), JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
            JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);
        frame.getContentPane().add(profileScrollPane());
    }
}
```

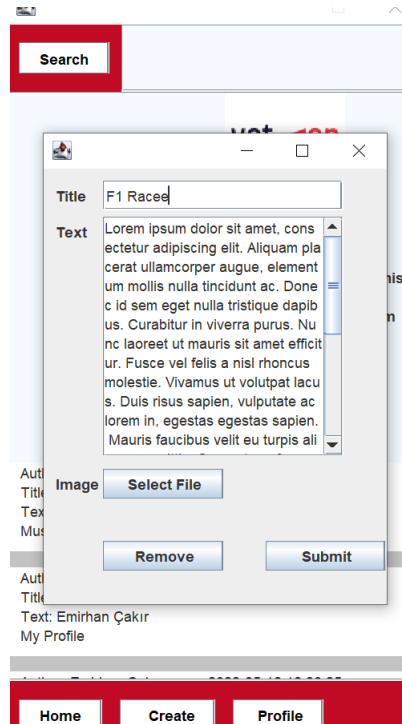
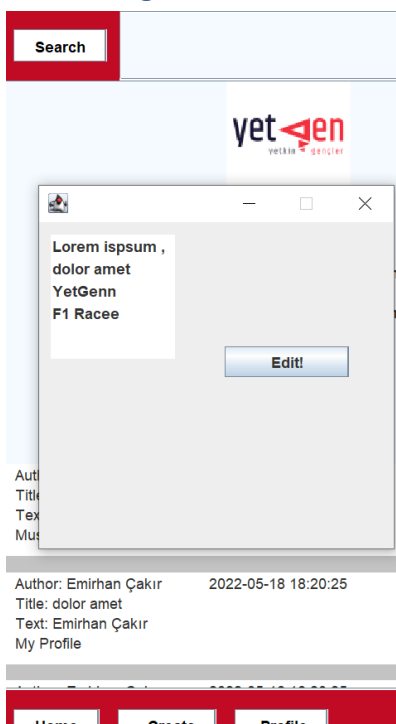


```
/**
 * The method is called in constructor to put all contents to the scroll pane.
 * Firstly it creates basic panel than put information of
 * user: name, nickname, ... After that it takes all contents of user to put
 * panel.
 * @return profilePanel - this panel hold all contents, buttons, informations of
 * user.
 */
public JPanel putMyContents() {
    profilePanel = new JPanel();
    profilePanel.setBackground(new Color(244, 244, 255));
    profilePanel.setLayout(new BorderLayout(profilePanel, BorderLayout.VK_KEYS));

    settings = new JButton();
    settings.setIcon(new ImageIcon(System.getProperty("user.dir") + "/comp122-project/img/settings.png"));
    settings.setSize(new Dimension(40, 40));
    settings.setText("Settings");
    settings.setBackground(new Color(244, 244, 255));
}
```

After adding the panel which is returned by putMyContent() method , profile page is set up.

Profile Page: Edit, Remove Content



Users select which content to edit, then edit frame is opened to make some change. If users want to remove content, they can click to remove button.

```
// It shows content with their names, then selected content is edited.
else if (e.getSource() == editContentBut) {
    ArrayList<Content> contents = new ArrayList<Content>();
    ArrayList<String> contentstitle = new ArrayList<String>();

    for (Content content : currentUser.getAllContent()) {
        contents.add(content);
        contentstitle.add(content.getTitle());
    }

    editFrame = new JFrame();

    editFrame.setLayout(null);

    editFrame.setDefaultCloseOperation(openSettingFrame.DISPOSE_ON_CLOSE);
    editFrame.setSize(300, 300);

    contentList = new JList(contentstitle.toArray());
}
```

```

} //It calls openeditPage method to edit content
else if (e.getSource() == contentEditBut) {
    openeditPage(contentList.getSelectedValue());
}

```

[illegible]

```

//It removes content from current user and content lists in Content class
else if (e.getSource() == removeContent) {
    currentUser.getAllContent().remove(editedContent);
    Content.getAllContentTitles().remove(editedContent.getTitle());
    Content.getAllContent().remove(editedContent.getTitle());
}

```

```

//It takes all input is entered to fields for edit operation.
else if (e.getSource() == submit) {
    if (image != null) {
        editedContent.setSelectedPhoto(image.toPath().toString());
    }
    editedContent.setTitle(titleField.getText());
    editedContent.setText(textArea.getText());
    contentEditFrame.dispose();
}

```

Search

Type:leonard@gmail.com

Follower 5

Following 3

Groups

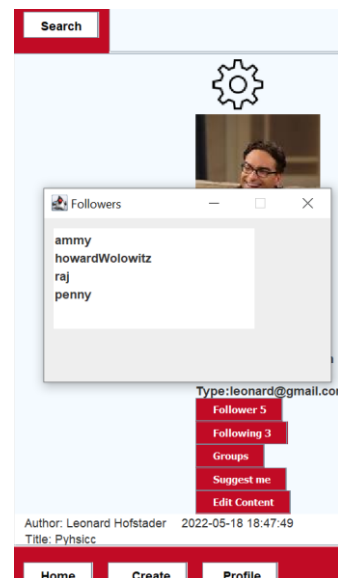
Suggest me

Edit Content

Author: Leonard Hofstadter
 Title: Physisic

2022-05-18 18:47:49

Home Create Profile



```
} // It calls openfollowers() method to show follower list.  
else if (e.getSource() == follower) {  
    openfollowers();  
}  
// It calls openfollowings() method to show following list.  
else if (e.getSource() == following) {  
    openFollowings();  
}
```

In `openFollowings()` method creates new frame put `JList` which contains followers' nickname. Almost same steps followed in `openFollowers()`.


```

/**
 * The method constructs a new frame which have information of following list.
 * It creates new frame firstly, then puts JList onto the frame.
 */
private void openFollowing() {
    followingListFrame = new JFrame("Following");
    followingListFrame.setLayout(null);

    followingListFrame.setDefaultCloseOperation(followingListFrame.DISPOSE_ON_CLOSE);
    followingListFrame.setSize(300, 200);

    groupList = new JList(currentUser.getFollowingList().keySet().toArray());
    groupList.setBounds(10, 10, 200, 100);
    followingListFrame.add(groupList);

    followingListFrame.setVisible(true);
    followingListFrame.setResizable(false);
    followingListFrame.setLocationRelativeTo(null);
}

```

```

/**
 * The method constructs new frame to put the nickname of all followers. It
 * takes nickname via followerList which is HashMap, the puts into the JList.
 */
private void openFollowers() {
    followerListFrame = new JFrame("Followers");
    followerListFrame.setLayout(null);

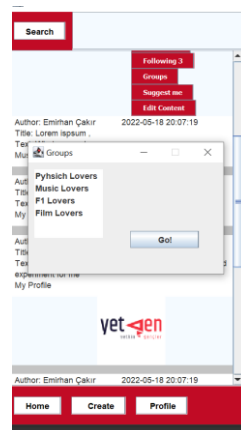
    followerListFrame.setDefaultCloseOperation(followerListFrame.DISPOSE_ON_CLOSE);
    followerListFrame.setSize(300, 200);

    groupList = new JList(currentUser.followerList.keySet().toArray());
    groupList.setBounds(10, 10, 200, 100);
    followerListFrame.add(groupList);

    followerListFrame.setVisible(true);
    followerListFrame.setResizable(false);
    followerListFrame.setLocationRelativeTo(null);
}

```

Profile Page: Display Groups



After groups button is clicked the codes which is given bellow executed. Users have an ability to visit group via this frame.

```

// If the related button is activated it calls openGroupList which shows group
// list which is attended.
if (e.getSource() == joinedGroup) {
    openGroupList();
}

```



```

// The method creates new frame which has view of the all attended groups. It
// takes name with for loop and then put into JList. Also method sets
// 'Go!' button to visit related group.
private void openGroupList() {
    groupListFrame = new JFrame("Groups");
    groupListFrame.setLayout(null);

    groupListFrame.setDefaultCloseOperation(groupListFrame.DISPOSE_ON_CLOSE);
    groupListFrame.setSize(300, 200);

    ArrayList<String> list = new ArrayList<>();

    for (Group eachGroup : currentUser.getGroupList()) {
        list.add(eachGroup.getName());
    }

    groupList = new JList(list.toArray());
}

```

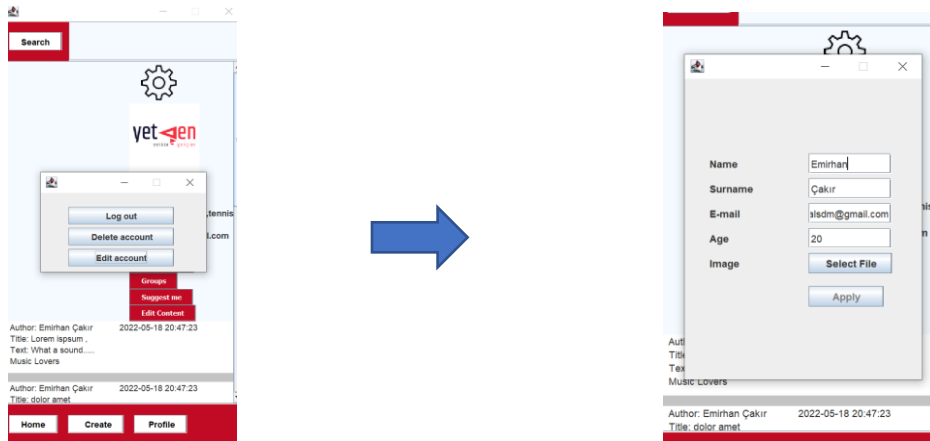
When "Groups" button is clicked openGroupList() is called, it opens frame with names of attended groups. There is "Go" button to visit group page. If this button is clicked, below codes executed, explanation gives in image.

```

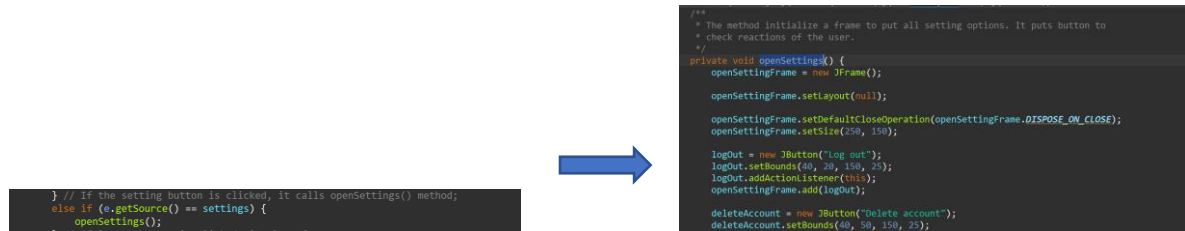
public void actionPerformed(ActionEvent e) {
    // If the related button is activated it calls openGroupList which shows group
    // list which is attended.
    if (e.getSource() == joinedGroup) {
        openGroupList();
    } // If the related group button is activated, it gets selected group from JList
    // then creates Group.
    else if (e.getSource() == goGroup) {
        Group selectedGroup = null;
        // JList doesn't give group object, for loop to find group in allgroup list
        for (Group eachGroup : Group.getAllGroups()) {
            if (eachGroup.getName() == groupList.getSelectedValuesList().get(0)) {
                selectedGroup = eachGroup;
                break;
            }
        }
        groupListFrame.dispose();
        GroupPage openGroupPage = new GroupPage(selectedGroup);
    }
}

```

Profile Page: Modify Profile



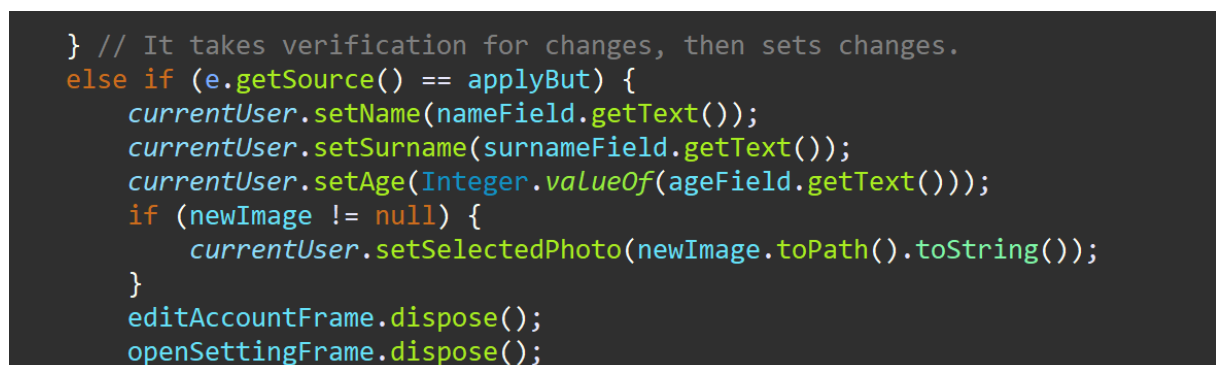
After clicking edit buttons, settings options are showed. “Edit Account” button triggers to build frame to edit. After apply edition process is done.



If setting button is clicked openSettings() is called. openSettings() show all setting options such as edit , delete account or log out.



If “Edit Account” button is clicked openEditAccount() method is called. It has text fields and labels to take change of user. After “Apply” button changes are done.



Profile Page: Log Out Account

Users access settings panel and choose “log out” button. This process is explained below how settings panel is opened.

```

        openSettings();
    } // If logout button is clicked, it close frames.
    else if (e.getSource() == logOut) {
        openSettingFrame.dispose();
        frame.dispose();
        LoginPage restart = new LoginPage();
    }
}

```

After logout button is activated, current frame is disposed, and login page is created again. Execution continues from login page.

Profile Page: Delete Account

```

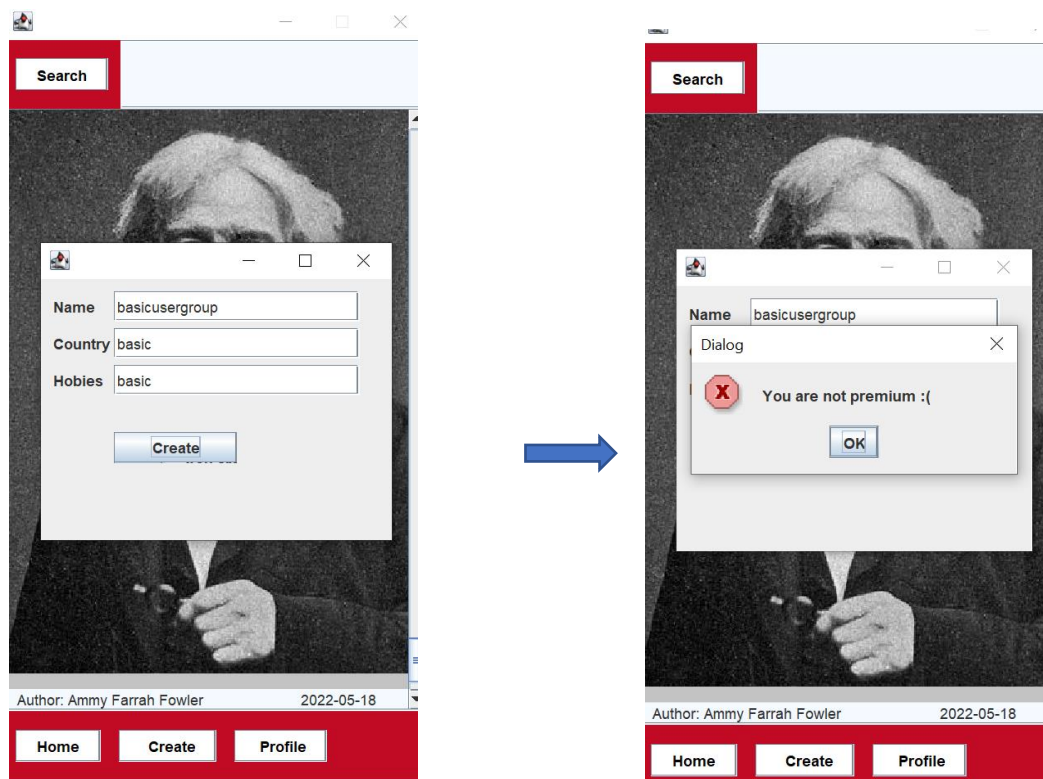
    } // If deleteAccount is activated, it removes user from all user.
    else if (e.getSource() == deleteAccount) {
        User.allUsers.remove(currentUser);

        openSettingFrame.dispose();
        frame.dispose();
        LoginPage restart = new LoginPage();
    }
}

```

User access to settings panel and selects “Delete Account” button. Delete Account button removes current user from allusers which hold all registered users. It creates login page object to continue execution.

Profile Page: Create Group



After create button is clicked if the user select group from combo box. The frame to create group opens. If user is premium, it cannot create group. If else block checks whether user is premium or not.

```

else if (e.getSource() == groupCreateButton) {
    if (currentUser.isPremium == false) {
        JOptionPane.showMessageDialog(new JFrame(), "You are not premium :", "Dialog",
            JOptionPane.ERROR_MESSAGE);
    } else {
        Group newGroup = new Group(titleField.getText(), countryField.getText(), hobbiesField.getText(),
            currentUser);
        currentUser.attendGroup(newGroup);
        createGroupFrame.dispose();
    }
}

```

Group Page

Group Page: Display Group Information



Group page is built according to the argument in constructor. The parameter is selected by current user. Almost same steps of profile page are followed in this class. Just instead of taking information of current user's information or contents, program takes group's contents and information.

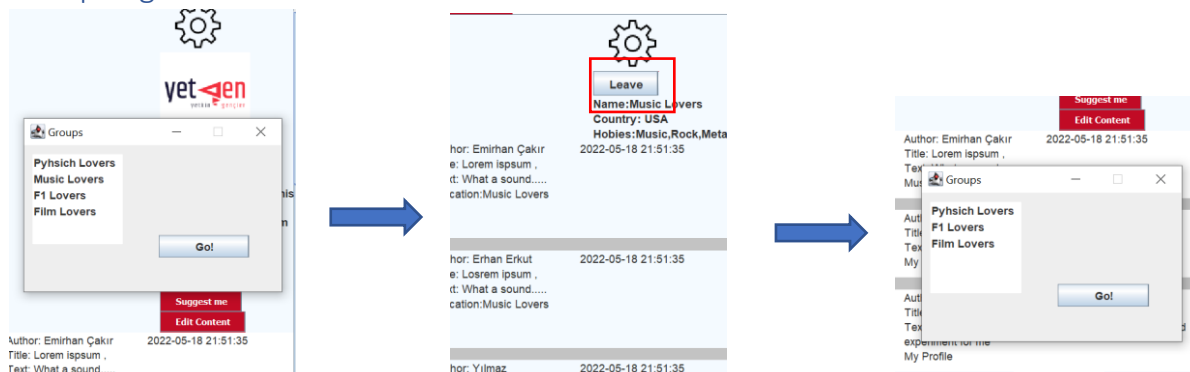
```

// This constructor takes Group object to build group page. It removes potential
// panels with build() method, then initialize pane to holds elements of the
// group.
// @param selectedGroup - selectedGroup parameter indicates which group is
// wanted to open.
//
public GroupPage(Group selectedGroup) {
    currentGroup = selectedGroup;
    build();
    groupScrollPane = new JScrollPane(addGroupElements(), JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
        JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);
    frame.add(groupScrollPane);
    frame.invalidate();
    frame.repaint();
}

// It initialize panel to elements. Then add firstly buttons and information of
// groups. After that contents which is shared in this group placed onto panel.
// JScrollPane uses this method generally.
//
// @return groupPane - groupPane holds all buttons, labels, fields will be used
// Panel
private JPanel addGroupElements() {
    groupPane = new JPanel();
    groupPane.setBackground(new Color(244, 250, 255));
    groupPane.setLayout(new BorderLayout(groupPane, BorderLayout.PAGE_AXIS));
    settings = new JButton();
    settings.setIcon(new ImageIcon(system.getProperty("user.dir") + "/compl32-project/img/settings.png"));
    settings.setBounds(380, 5, 30, 30);
    settings.setBackground(new Color(244, 250, 255));
    settings.setBorderPainted(false);
    settings.addActionListener(this);
    groupPane.add(settings);
}

```

Group Page: Leave



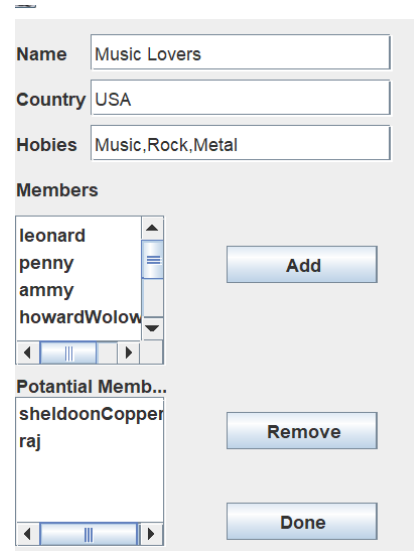
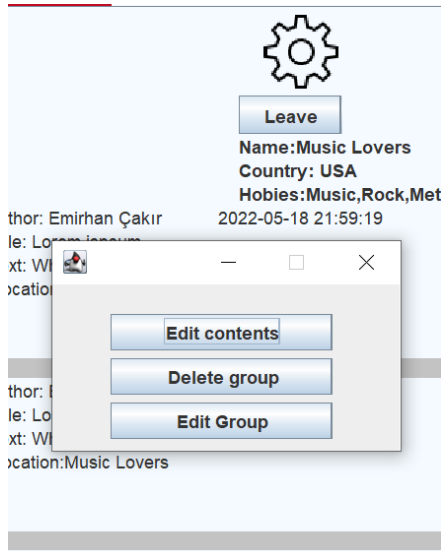
After leave button is pressed below codes are executed. Leave group method is called, it removes group from the group list of user and remove user from the member of group.

```
// users leave groups with this button
else if (e.getSource() == leaveBut) {
    currentUser.leaveGroup(currentGroup);
}
```



```
public void leaveGroup(Group group) {
    groupList.remove(group);
    group.getMembers().remove(this);
}
```

Group Page: Edit Group and Display Users



After “Edit Group” button is pressed, creator could display users, edit information of the group. If they are not creator they will not able open edit page.

```
// It enables group creator to edit group
else if (e.getSource() == editGroupBut) {
    if (currentUser.equals(currentGroup.getCreator())) {
        openEditGroupPage();
    } else {
        JOptionPane.showMessageDialog(new JFrame(), "You are not creator :)", "Dialog",
            JOptionPane.ERROR_MESSAGE);
    }
}
```



```
/**
 * It creates new frame to edit group. Before that close frames to make easier
 * to use program for user.
 */
private void openEditGroupPage() {
    try {
        editGroupFrame.dispose();
    } catch (NullPointerException e) {
        openSettingFrame.dispose();
    }
    editGroupFrame = new JFrame();
    editGroupFrame.setLayout(new BorderLayout());
    editGroupFrame.setDefaultCloseOperation(editGroupFrame.DISPOSE_ON_CLOSE);
    editGroupFrame.setSize(300, 400);

    JLabel groupTitle = new JLabel("Name");
    groupTitle.setBounds(10, 10, 50, 25);
    editGroupFrame.add(groupTitle);

    JTextField titleField = new JTextField(currentGroup.getName());
```



```
// It takes verification to edit
else if (e.getSource() == groupEditDoneButton) {
    currentGroup.setName(titleField.getText());
    currentGroup.setCountry(countryField.getText());
    currentGroup.setHobbies(hobbiesField.getText());
    editGroupFrame.dispose();
}
```

Notes

More information for classes and methods Javadoc could be checked which locates in same file.

References

1. <https://docs.oracle.com/javase/7/docs/api/javax/swing/JPanel.html>

2. <https://docs.oracle.com/javase/7/docs/api/javax/swing/JScrollPane.html>
3. <https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html>
4. <https://docs.oracle.com/javase/8/docs/api/java/awt/Color.html>