```
package dlsu;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource("
SelectCard.fxml"));
        root.getStylesheets().addAll(this.getClass().
getResource("style.css").toExternalForm());
        primaryStage.setTitle("Fall Analyzer | Select Card");
        primaryStage.setResizable(false);
        primaryStage.setScene(new Scene(root, 1024, 575.5));
        primaryStage.show();
    }

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

```css
.root{
    -fx-background-color: #ecf0f1;
}

#setupPane{
    -fx-background-image: url("Main.png");
    -fx-background-size: 1030;
    -fx-background-repeat: stretch;
    -fx-background-position: center center;
}

#explorePane{
    -fx-background-image: url("Explore.png");
    -fx-background-size: 1030;
    -fx-background-repeat: stretch;
    -fx-background-position: center center;
}

#header1Label{
    -fx-font-family: Asimov;
    -fx-text-fill: orange;
    -fx-font-size: 30;
    -fx-font-position: center center;
}

#header2Label{
    -fx-font-family: Asimov;
    -fx-text-fill: #2c3e50;
    -fx-font-size: 15;
    -fx-font-position: center center;
    -fx-text-alignment: justify;
}

#header3Label{
    -fx-font-family: Asimov;
    -fx-text-fill: orange;
    -fx-font-size: 20;
    -fx-font-position: center center;
}

#fieldsLabelLabel{
    -fx-font-family: Asimov;
    -fx-text-fill: #2c3e50;
    -fx-font-size: 13;
    -fx-font-position: center center;
    -fx-text-alignment: justify;
}

#feedbackLabel{
    -fx-font-family: Asimov;
    -fx-text-fill: #2c3e50;
    -fx-font-size: 10;
    -fx-font-position: center center;
}

#errorLabel{
    -fx-font-family: Asimov;
    -fx-text-fill: red;
    -fx-font-size: 10;
    -fx-font-position: center center;
}

#start{
        -fx-font-family: Asimov;
        -fx-text-fill: orange;
        -fx-font-size: 20;
        -fx-font-position: center center;
        -fx-background-color: white;
}

#respondButtons{
        -fx-font-family: Asimov;
        -fx-text-fill: orange;
        -fx-font-size: 15;
        -fx-font-position: center center;
        -fx-background-color: white;
}

#selected{
        -fx-font-family: Asimov;
        -fx-text-fill: white;
        -fx-font-size: 20;
        -fx-font-position: center center;
        -fx-background-color: #009fe0;
        -fx-background-radius: 5em;
        -fx-max-height: 45px;
        -fx-max-width: 45px;
        -fx-min-height: 45px;
        -fx-min-width: 45px;
}

#idle{
        -fx-font-family: Asimov;
        -fx-text-fill: orange;
        -fx-font-size: 20;
        -fx-font-position: center center;
        -fx-background-color: white;
        -fx-background-radius: 5em;
        -fx-max-height: 45px;
        -fx-max-width: 45px;
        -fx-min-height: 45px;
        -fx-min-width: 45px;
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUp1Controller">
    <children>
      <Group layoutX="1.0" layoutY="42.0">
         <children>
              <Label id="header2Label" layoutX="380.0" layoutY=
"254.0" stylesheets="@style.css" text="Let's setup your login
credentials!" textAlignment="JUSTIFY" />
            <GridPane hgap="10.0" layoutX="380.0" layoutY="288.
0" prefHeight="90.0" prefWidth="263.0">
              <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" maxWidth="
115.0" minWidth="10.0" prefWidth="91.0" />
                <ColumnConstraints hgrow="ALWAYS" maxWidth="162
.0" minWidth="10.0" prefWidth="162.0" />
              </columnConstraints>
              <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
              </rowConstraints>
               <children>
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Username:" />
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Password:" GridPane.rowIndex="1" />
                   <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Confirm:" GridPane.rowIndex="2" />
                  <TextField fx:id="username" prefHeight="25.0"
 prefWidth="162.0" promptText="username" GridPane.columnIndex="
1" />
```

```
                  <PasswordField fx:id="password" promptText="
password" GridPane.columnIndex="1" GridPane.rowIndex="1" />
                  <PasswordField fx:id="confirm" prefHeight="25
.0" prefWidth="117.0" promptText="confirm password" GridPane.
columnIndex="1" GridPane.rowIndex="2" />
               </children>
            </GridPane>
              <Label id="errorLabel" fx:id="feedbackLabel"
layoutX="497.0" layoutY="387.0" stylesheets="@style.css"
textAlignment="CENTER" />
         </children>
      </Group>
        <Group layoutX="-33.0">
            <children>
                <HBox layoutX="376.0" layoutY="484.0"
prefHeight="45.0" prefWidth="310.0" spacing="10.0">
                    <children>
                        <Button id="selected" fx:id="first"
mnemonicParsing="false" onAction="#onFirst" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="1" />
                        <Button id="idle" fx:id="second"
mnemonicParsing="false" onAction="#onSecond" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="2" />
                        <Button id="idle" fx:id="third"
mnemonicParsing="false" onAction="#onThird" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="3" />
                        <Button id="idle" fx:id="fourth"
mnemonicParsing="false" onAction="#onFourth" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="4" />
                        <Button id="start" fx:id="finish"
mnemonicParsing="false" onAction="#onFinish" prefHeight="61.0"
prefWidth="88.0" stylesheets="@style.css" text="finish" />
                    </children>
                </HBox>
            </children>
        </Group>
    </children>
</Pane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUp2Controller">
    <children>
      <Group layoutX="-146.0" layoutY="50.0">
         <children>
              <Label id="header2Label" layoutX="566.0" layoutY=
"229.0" stylesheets="@style.css" text="Let's setup your profile
!" textAlignment="JUSTIFY" />
            <GridPane hgap="10.0" layoutX="380.0" layoutY="275.
0" prefHeight="90.0" prefWidth="263.0">
              <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" maxWidth="
115.0" minWidth="10.0" prefWidth="91.0" />
                <ColumnConstraints hgrow="ALWAYS" maxWidth="162
.0" minWidth="10.0" prefWidth="162.0" />
              </columnConstraints>
              <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
              </rowConstraints>
               <children>
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="First Name:" />
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Last Name:" GridPane.rowIndex="1" />
                   <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Gender:" GridPane.rowIndex="2" />
                  <TextField fx:id="firstName" prefHeight="25.0
" prefWidth="162.0" promptText="first name" GridPane.
```

```
columnIndex="1" />
                  <TextField fx:id="LastName" promptText="last
name" GridPane.columnIndex="1" GridPane.rowIndex="1" />
                  <ChoiceBox fx:id="gender" prefHeight="25.0"
prefWidth="82.0" GridPane.columnIndex="1" GridPane.rowIndex="2"
 />
               </children>
            </GridPane>
              <Label id="errorLabel" fx:id="feedbackLabel"
layoutX="496.0" layoutY="380.0" stylesheets="@style.css"
textAlignment="CENTER" />
            <GridPane hgap="10.0" layoutX="674.0" layoutY="317.
0" prefHeight="90.0" prefWidth="263.0">
               <children>
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Age:" />
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Address:" GridPane.rowIndex="1" />
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Contact Number:" GridPane.rowIndex="2" />
                  <TextField fx:id="age" prefHeight="25.0"
prefWidth="162.0" promptText="age" GridPane.columnIndex="1" />
                  <TextField fx:id="address" promptText="
address" GridPane.columnIndex="1" GridPane.rowIndex="1" />
                  <TextField fx:id="contactNumber" prefHeight="
25.0" prefWidth="97.0" promptText="contact number" text="+639"
GridPane.columnIndex="1" GridPane.rowIndex="2" />
               </children>
               <columnConstraints>
                  <ColumnConstraints hgrow="SOMETIMES" maxWidth
="123.0" minWidth="10.0" prefWidth="120.0" />
                  <ColumnConstraints hgrow="ALWAYS" maxWidth="
162.0" minWidth="10.0" prefWidth="133.0" />
               </columnConstraints>
               <rowConstraints>
                  <RowConstraints minHeight="10.0" prefHeight="
30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="
30.0" vgrow="SOMETIMES" />
                  <RowConstraints minHeight="10.0" prefHeight="
30.0" vgrow="SOMETIMES" />
               </rowConstraints>
            </GridPane>
         </children>
      </Group>
        <Group layoutX="-33.0">
            <children>
                <HBox layoutX="376.0" layoutY="484.0"
prefHeight="45.0" prefWidth="310.0" spacing="10.0">
                    <children>
                        <Button id="idle" fx:id="first"
mnemonicParsing="false" onAction="#onFirst" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="1" />
                        <Button id="selected" fx:id="second"
mnemonicParsing="false" onAction="#onSecond" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="2" />
                        <Button id="idle" fx:id="third"
mnemonicParsing="false" onAction="#onThird" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="3" />
                        <Button id="idle" fx:id="fourth"
mnemonicParsing="false" onAction="#onFourth" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="4" />
                        <Button id="start" fx:id="finish"
mnemonicParsing="false" onAction="#onFinish" prefHeight="61.0"
prefWidth="88.0" stylesheets="@style.css" text="finish" />
                    </children>
                </HBox>
            </children>
        </Group>
    </children>
</Pane>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUp3Controller">
    <children>
      <Group layoutX="-18.0" layoutY="-6.0">
         <children>
            <Group layoutX="-79.0" layoutY="16.0">
               <children>
                  <ListView fx:id="list" layoutX="708.0"
layoutY="310.0" prefHeight="119.0" prefWidth="200.0" />
                  <Label id="header2Label" layoutX="706.0"
layoutY="275.0" stylesheets="@style.css" text="Here are your
responders:" textAlignment="JUSTIFY" />
                  <Button layoutX="817.0" layoutY="445.0"
mnemonicParsing="false" onAction="#onEdit" text="edit" />
                  <Button layoutX="858.0" layoutY="445.0"
mnemonicParsing="false" onAction="#onDelete" text="delete" />
               </children>
            </Group>
            <Group layoutX="-146.0" layoutY="50.0">
               <children>
                    <Label id="header2Label" layoutX="441.0"
layoutY="243.0" stylesheets="@style.css" text="Let's add your
responders!" textAlignment="JUSTIFY" />
                  <GridPane hgap="10.0" layoutX="380.0" layoutY
="275.0" prefHeight="90.0" prefWidth="328.0">
                    <columnConstraints>
                      <ColumnConstraints hgrow="SOMETIMES"
maxWidth="137.6666259765625" minWidth="10.0" prefWidth="103.
33334350585938" />
                      <ColumnConstraints hgrow="ALWAYS"
maxWidth="196.66665649414062" minWidth="10.0" prefWidth="196.
66665649414062" />
                    </columnConstraints>
                    <rowConstraints>
                      <RowConstraints minHeight="10.0"
prefHeight="30.0" vgrow="SOMETIMES" />
                      <RowConstraints minHeight="10.0"
prefHeight="30.0" vgrow="SOMETIMES" />
                      <RowConstraints minHeight="10.0"
prefHeight="30.0" vgrow="SOMETIMES" />
                    </rowConstraints>
                     <children>
                        <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="First Name:" />
                        <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="Last Name:" GridPane.rowIndex="1
" />
                        <TextField fx:id="firstName" prefHeight
="25.0" prefWidth="162.0" promptText="first name" GridPane.
columnIndex="1" />
                        <TextField fx:id="LastName" prefHeight=
"25.0" prefWidth="182.0" promptText="last name" GridPane.
columnIndex="1" GridPane.rowIndex="1" />
                        <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="Contact Number:" GridPane.
rowIndex="2" />
                        <TextField fx:id="contactNumber"
prefHeight="25.0" prefWidth="117.0" promptText="contact number"
 text="+639" GridPane.columnIndex="1" GridPane.rowIndex="2" />
                     </children>
                  </GridPane>
                  <Button fx:id="add" layoutX="526.0" layoutY="
375.0" mnemonicParsing="false" onAction="#onAdd" text="add" />
               </children>
            </Group>
              <Label id="errorLabel" fx:id="feedbackLabel"
layoutX="382.0" layoutY="466.0" stylesheets="@style.css"
textAlignment="CENTER" />
         </children>
      </Group>
        <Group layoutX="-33.0">
            <children>
                <HBox layoutX="376.0" layoutY="484.0"
prefHeight="45.0" prefWidth="310.0" spacing="10.0">
                    <children>
```

```
                        <Button id="idle" fx:id="first"
mnemonicParsing="false" onAction="#onFirst" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="1" />
                        <Button id="idle" fx:id="second"
mnemonicParsing="false" onAction="#onSecond" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="2" />
                        <Button id="selected" fx:id="third"
mnemonicParsing="false" onAction="#onThird" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="3" />
                        <Button id="idle" fx:id="fourth"
mnemonicParsing="false" onAction="#onFourth" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="4" />
                        <Button id="start" fx:id="finish"
mnemonicParsing="false" onAction="#onFinish" prefHeight="61.0"
prefWidth="88.0" stylesheets="@style.css" text="finish" />
                    </children>
                </HBox>
            </children>
        </Group>
    </children>
</Pane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUp4Controller">
    <children>
      <Group layoutX="1.0" layoutY="40.0">
         <children>
              <Label id="header2Label" layoutX="409.0" layoutY=
"259.0" stylesheets="@style.css" text="To finish setup, enter
your" textAlignment="CENTER" />
            <GridPane hgap="10.0" layoutX="380.0" layoutY="315.
0" prefHeight="68.0" prefWidth="263.0">
              <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" maxWidth="
115.0" minWidth="10.0" prefWidth="91.0" />
                <ColumnConstraints hgrow="ALWAYS" maxWidth="162
.0" minWidth="10.0" prefWidth="162.0" />
              </columnConstraints>
              <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
              </rowConstraints>
               <children>
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Username:" />
                  <Label id="fieldsLabelLabel" stylesheets="@
style.css" text="Password:" GridPane.rowIndex="1" />
                  <TextField fx:id="username" prefHeight="25.0"
 prefWidth="162.0" promptText="username" GridPane.columnIndex="
1" />
                  <PasswordField fx:id="password" promptText="
password" GridPane.columnIndex="1" GridPane.rowIndex="1" />
               </children>
            </GridPane>
```

```
              <Label id="errorLabel" fx:id="feedbackLabel"
layoutX="497.0" layoutY="393.0" stylesheets="@style.css"
textAlignment="CENTER" />
            <Label id="header2Label" layoutX="389.0" layoutY="
278.0" stylesheets="@style.css" text="username and password
below:" textAlignment="CENTER" />
         </children>
      </Group>
      <Group layoutX="-33.0">
         <children>
            <HBox layoutX="376.0" layoutY="484.0" prefHeight="
45.0" prefWidth="310.0" spacing="10.0">
               <children>
                    <Button id="idle" fx:id="first"
mnemonicParsing="false" onAction="#onFirst" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="1" />
                  <Button id="idle" fx:id="second"
mnemonicParsing="false" onAction="#onSecond" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="2" />
                  <Button id="idle" fx:id="third"
mnemonicParsing="false" onAction="#onThird" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="3" />
                  <Button id="selected" fx:id="fourth"
mnemonicParsing="false" onAction="#onFourth" prefHeight="38.0"
prefWidth="81.0" stylesheets="@style.css" text="4" />
                  <Button id="start" fx:id="finish"
mnemonicParsing="false" onAction="#onFinish" prefHeight="61.0"
prefWidth="88.0" stylesheets="@style.css" text="finish" />
               </children>
            </HBox>
         </children>
      </Group>
    </children>
</Pane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.chart.*?>
<?import java.lang.*?>
<?import javafx.scene.image.*?>
<?import javafx.scene.shape.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="explorePane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
exploreController">
   <children>
      <Group layoutX="802.0" layoutY="91.0">
         <children>
            <Label id="header2Label" layoutX="154.0" layoutY="
10.0" stylesheets="@style.css" text="Hello," />
            <TilePane alignment="CENTER_RIGHT" layoutX="-227.0"
 layoutY="21.0" prefHeight="19.0" prefWidth="434.0">
               <children>
                  <Label id="header1Label" fx:id="usernameHello
" stylesheets="@style.css" text="Username!" textAlignment="
RIGHT" />
               </children>
            </TilePane>
         </children>
      </Group>
      <TabPane layoutY="144.0" prefHeight="438.0" prefWidth="
1025.0" tabClosingPolicy="UNAVAILABLE">
        <tabs>
          <Tab fx:id="profileTab" text="Profile">
            <content>
              <AnchorPane minHeight="0.0" minWidth="0.0"
prefHeight="424.0" prefWidth="1026.0">
                <children>
                        <Group layoutX="499.0" layoutY="-8.0">
                           <children>
                              <TilePane layoutX="97.0" layoutY=
"260.0" prefHeight="75.0" prefWidth="161.0">
                                 <children>
```

```
                                    <Line endX="100.0" startX="
-100.0" stroke="#009fe0" />
                                    <Label id="errorLabel" fx:
id="feedbackLabelLogin" stylesheets="@style.css" textAlignment=
"CENTER" />
                                    <HBox alignment="CENTER"
prefHeight="25.0" prefWidth="78.0" spacing="5.0">
                                       <children>
                                          <Button fx:id="
editLogin" mnemonicParsing="false" onAction="#onEditLogin" text
="edit" />
                                          <Button fx:id="
saveLogin" disable="true" mnemonicParsing="false" onAction="#
onSaveLogin" text="save" />
                                       </children>
                                    </HBox>
                                 </children>
                              </TilePane>
                          <GridPane hgap="10.0" layoutX="66.0"
layoutY="160.0" prefHeight="90.0" prefWidth="263.0">
                              <columnConstraints>
                                  <ColumnConstraints hgrow="
SOMETIMES" maxWidth="115.0" minWidth="10.0" prefWidth="91.0" />
                                  <ColumnConstraints hgrow="
ALWAYS" maxWidth="162.0" minWidth="10.0" prefWidth="162.0" />
                              </columnConstraints>
                              <rowConstraints>
                                  <RowConstraints minHeight="10
.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                  <RowConstraints minHeight="10
.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                  <RowConstraints minHeight="10
.0" prefHeight="30.0" vgrow="SOMETIMES" />
                              </rowConstraints>
                              <children>
                                  <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="Username:" />
                                  <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="Password:" GridPane.rowIndex="1"
 />
                                  <Label id="fieldsLabelLabel"
stylesheets="@style.css" text="Confirm:" GridPane.rowIndex="2"
```

```
/>
                                  <TextField fx:id="
usernameLogin" editable="false" prefHeight="25.0" prefWidth="
162.0" promptText="username" GridPane.columnIndex="1" />
                                  <PasswordField fx:id="
passwordLogin" editable="false" promptText="password" GridPane.
columnIndex="1" GridPane.rowIndex="1" />
                                  <PasswordField fx:id="
confirmLogin" editable="false" prefHeight="25.0" prefWidth="117
.0" promptText="confirm password" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
                              </children>
                          </GridPane>
                          <Label id="header3Label" layoutX="66.
0" layoutY="126.0" stylesheets="@style.css" text="Login
credentials:" textAlignment="JUSTIFY" />
                           </children>
                        </Group>
                    <Group layoutX="-235.0" layoutY="-168.0">
                       <children>
                           <Label id="header3Label" layoutX="
379.0" layoutY="248.0" stylesheets="@style.css" text="Profile:"
 textAlignment="JUSTIFY" />
                           <GridPane hgap="10.0" layoutX="380.0
" layoutY="275.0" prefHeight="90.0" prefWidth="278.0">
                               <columnConstraints>
                                   <ColumnConstraints hgrow="
SOMETIMES" maxWidth="128.66665649414062" minWidth="10.0"
prefWidth="119.33334350585938" />
                                   <ColumnConstraints hgrow="
ALWAYS" maxWidth="162.0" minWidth="10.0" prefWidth="133.
66665649414062" />
                               </columnConstraints>
                               <rowConstraints>
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                               </rowConstraints>
                               <children>
```

```
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="First Name:" />
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="Last Name:" GridPane.rowIndex="
1" />
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="Gender:" GridPane.rowIndex="2"
/>
                                   <TextField fx:id="
firstNameProfile" editable="false" prefHeight="25.0" prefWidth=
"162.0" promptText="first name" GridPane.columnIndex="1" />
                                   <TextField fx:id="
LastNameProfile" editable="false" prefHeight="25.0" prefWidth="
142.0" promptText="last name" GridPane.columnIndex="1" GridPane
.rowIndex="1" />
                                   <ChoiceBox fx:id="
genderProfile" disable="true" prefHeight="25.0" prefWidth="82.0
" GridPane.columnIndex="1" GridPane.rowIndex="2" />
                               </children>
                           </GridPane>
                           <GridPane hgap="10.0" layoutX="379.0
" layoutY="368.0" prefHeight="90.0" prefWidth="279.0">
                               <children>
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="Age:" />
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="Address:" GridPane.rowIndex="1"
 />
                                   <Label id="fieldsLabelLabel"
 stylesheets="@style.css" text="Contact Number:" GridPane.
rowIndex="2" />
                                   <TextField fx:id="ageProfile
" editable="false" prefHeight="25.0" prefWidth="162.0"
promptText="age" GridPane.columnIndex="1" />
                                   <TextField fx:id="
addressProfile" editable="false" prefHeight="25.0" prefWidth="
146.0" promptText="address" GridPane.columnIndex="1" GridPane.
rowIndex="1" />
                                   <TextField fx:id="
contactNumberProfile" editable="false" prefHeight="25.0"
prefWidth="117.0" promptText="contact number" text="+639"
GridPane.columnIndex="1" GridPane.rowIndex="2" />
```

```
                               </children>
                               <columnConstraints>
                                   <ColumnConstraints hgrow="
SOMETIMES" maxWidth="123.0" minWidth="10.0" prefWidth="120.0"
/>
                                   <ColumnConstraints hgrow="
ALWAYS" maxWidth="162.0" minWidth="10.0" prefWidth="133.0" />
                               </columnConstraints>
                               <rowConstraints>
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                   <RowConstraints minHeight="
10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                               </rowConstraints>
                           </GridPane>
                              <TilePane layoutX="419.0" layoutY
="467.0" prefHeight="57.0" prefWidth="201.0">
                                 <children>
                                    <Line endX="100.0" startX="
-100.0" stroke="#009fe0" />
                                 <Label id="errorLabel" fx:id="
feedbackLabel" stylesheets="@style.css" textAlignment="CENTER"
/>
                                    <HBox alignment="CENTER"
prefHeight="25.0" prefWidth="81.0" spacing="5.0">
                                       <children>
                                          <Button fx:id="
editProfile" mnemonicParsing="false" onAction="#onEditProfile"
text="edit" />
                                          <Button fx:id="
saveProfile" disable="true" mnemonicParsing="false" onAction="#
onSaveProfile" text="save" />
                                       </children>
                                    </HBox>
                                 </children>
                              </TilePane>
                       </children>
                    </Group>
                </children>
              </AnchorPane>
```

```
            </content>
          </Tab>
          <Tab fx:id="respondersTab" text="Responders">
            <content>
              <AnchorPane minHeight="0.0" minWidth="0.0"
prefHeight="180.0" prefWidth="200.0">
                     <children>
                        <TableView fx:id="respondersTable"
layoutX="-1.0" layoutY="-3.0" prefHeight="365.0" prefWidth="527
.0">
                          <columns>
                            <TableColumn fx:id="firstName"
prefWidth="163.3333282470703" text="First Name" />
                              <TableColumn fx:id="lastName"
prefWidth="173.0" text="Last Name" />
                              <TableColumn fx:id="contactNumber
" prefWidth="189.0" text="Contact Number" />
                          </columns>
                        </TableView>
                         <Group fx:id="editResponders" disable=
"true" layoutX="236.0" layoutY="-139.0">
                             <children>
                                 <Label id="header2Label"
layoutX="428.0" layoutY="247.0" stylesheets="@style.css" text="
Let's modify your responders!" textAlignment="JUSTIFY" />
                                 <GridPane hgap="10.0" layoutX=
"380.0" layoutY="275.0" prefHeight="90.0" prefWidth="328.0">
                                     <columnConstraints>
                                         <ColumnConstraints
hgrow="SOMETIMES" maxWidth="137.6666259765625" minWidth="10.0"
prefWidth="103.33334350585938" />
                                         <ColumnConstraints
hgrow="ALWAYS" maxWidth="196.66665649414062" minWidth="10.0"
prefWidth="196.66665649414062" />
                                     </columnConstraints>
                                     <rowConstraints>
                                         <RowConstraints
minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                         <RowConstraints
minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
                                         <RowConstraints
minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
```

```
                                     </rowConstraints>
                                     <children>
                                         <Label id="
fieldsLabelLabel" stylesheets="@style.css" text="First Name:"
/>
                                         <Label id="
fieldsLabelLabel" stylesheets="@style.css" text="Last Name:"
GridPane.rowIndex="1" />
                                         <TextField fx:id="
firstNameResponders" prefHeight="25.0" prefWidth="162.0"
promptText="first name" GridPane.columnIndex="1" />
                                         <TextField fx:id="
LastNameResponders" prefHeight="25.0" prefWidth="182.0"
promptText="last name" GridPane.columnIndex="1" GridPane.
rowIndex="1" />
                                         <Label id="
fieldsLabelLabel" stylesheets="@style.css" text="Contact Number
:" GridPane.rowIndex="2" />
                                         <TextField fx:id="
contactNumberResponders" prefHeight="25.0" prefWidth="117.0"
promptText="contact number" text="+639" GridPane.columnIndex="1
" GridPane.rowIndex="2" />
                                     </children>
                                 </GridPane>
                                 <Button fx:id="saveResponder"
layoutX="526.0" layoutY="375.0" mnemonicParsing="false"
onAction="#onSaveResponder" text="save" />
                              <Label id="errorLabel" fx:id="
allFieldsError" layoutX="491.0" layoutY="415.0" stylesheets="@
style.css" textAlignment="CENTER" />
                              <Label id="errorLabel" fx:id="
successful" layoutX="439.0" layoutY="415.0" stylesheets="@style
.css" textAlignment="CENTER" />
                             </children>
                         </Group>
                        <HBox layoutX="374.0" layoutY="371.0"
prefHeight="29.0" prefWidth="137.0" spacing="5.0">
                           <children>
                              <Button fx:id="deleteResponder"
mnemonicParsing="false" onAction="#onDeleteResponder" text="
delete" />
                              <Button fx:id="editResponder"
```

```
mnemonicParsing="false" onAction="#onEditResponder" text="edit"
 />
                              <Button fx:id="addResponder"
mnemonicParsing="false" onAction="#onAddResponder" text="add"
/>
                           </children>
                        </HBox>
                     </children>
                  </AnchorPane>
            </content>
          </Tab>
            <Tab fx:id="activityTab" text="Activity Log">
               <content>
                  <AnchorPane minHeight="0.0" minWidth="0.0"
prefHeight="180.0" prefWidth="200.0">
                     <children>
                        <TableView fx:id="activityTable"
prefHeight="402.0" prefWidth="659.0">
                           <columns>
                              <TableColumn fx:id="dateCell"
prefWidth="146.33333206176758" text="Date" />
                              <TableColumn fx:id="timeCell"
prefWidth="161.6666717529297" text="Time" />
                              <TableColumn fx:id="activityCell"
 prefWidth="171.33331298828125" text="Activity" />
                              <TableColumn fx:id="locationCell"
 prefWidth="176.66668701171875" text="Location" />
                           </columns>
                        </TableView>
                        <TilePane alignment="CENTER" layoutX="
671.0" layoutY="325.0" prefHeight="59.0" prefWidth="346.0">
                           <children>
                              <Label id="feedbackLabel" fx:id="
feedbackData" stylesheets="@style.css" textAlignment="CENTER"
/>
                              <HBox alignment="TOP_CENTER"
prefHeight="26.0" prefWidth="241.0" spacing="5.0">
                                 <children>
                                    <Button id="respondButtons"
 fx:id="deleteData" mnemonicParsing="false" onAction="#
onDeleteData" prefHeight="30.0" prefWidth="71.0" stylesheets="@
style.css" text="delete">
```

```
                                        <tooltip>
                                            <Tooltip text="
Delete high-lighted data log" />
                                        </tooltip>
                                    </Button>
                                     <Button id="respondButtons
" fx:id="deleteAllData" mnemonicParsing="false" onAction="#
onDeleteAllData" stylesheets="@style.css" text="clear all">
                                         <tooltip>
                                             <Tooltip text="
Clear the entire data log" />
                                         </tooltip>
                                     </Button>
                                    <Button id="respondButtons"
 fx:id="exportData" mnemonicParsing="false" onAction="#
onExportData" stylesheets="@style.css" text="export">
                                        <tooltip>
                                            <Tooltip text="
Export data log to an Excel File" />
                                        </tooltip>
                                    </Button>
                                 </children>
                              </HBox>
                           </children>
                        </TilePane>
                        <GridPane layoutX="726.0" layoutY="296.
0" prefHeight="30.0" prefWidth="246.0">
                          <columnConstraints>
                            <ColumnConstraints hgrow="SOMETIMES
" maxWidth="230.0" minWidth="10.0" prefWidth="206.0" />
                            <ColumnConstraints hgrow="SOMETIMES
" maxWidth="95.0" minWidth="10.0" prefWidth="13.0" />
                          </columnConstraints>
                          <rowConstraints>
                            <RowConstraints minHeight="10.0"
prefHeight="30.0" vgrow="SOMETIMES" />
                          </rowConstraints>
                           <children>
                              <Label id="header2Label" text="
Number of recorded falls:" />
                              <Label id="header3Label" fx:id="
numberOfFalls" text="0" GridPane.columnIndex="1" />
```

```
                           </children>
                        </GridPane>
                        <PieChart id="feedbackLabel" fx:id="
storageSpace" layoutX="666.0" layoutY="48.0" prefHeight="225.0"
 prefWidth="351.0" stylesheets="@style.css" />
                        <Label id="header3Label" layoutX="762.0
" layoutY="14.0" stylesheets="@style.css" text="Storage Usage:"
 />
                     </children></AnchorPane>
               </content>
            </Tab>
        </tabs>
      </TabPane>
   </children>
</Pane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUpHomeController">
    <children>
        <Label id="header1Label" layoutX="467.0" layoutY="288.0
" stylesheets="@style.css" text="Hello!" />
        <Label id="header2Label" layoutX="382.0" layoutY="337.0
" stylesheets="@style.css" text="Thank you for using fall
analyzer! " textAlignment="JUSTIFY" />
        <Label id="header2Label" layoutX="383.0" layoutY="357.0
" stylesheets="@style.css" text="To begin setup, please click
start." textAlignment="JUSTIFY" />
        <Button id="start" fx:id="onNext" layoutX="473.0"
layoutY="411.0" mnemonicParsing="false" onAction="#onNext"
prefHeight="38.0" prefWidth="81.0" stylesheets="@style.css"
text="start" />
    </children>
</Pane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.image.*?>
<?import java.lang.*?>
<?import javafx.geometry.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
selectCardController">
   <children>
      <TilePane alignment="CENTER" layoutX="352.0" layoutY="317
.0" prefHeight="152.0" prefWidth="321.0">
         <children>
            <Label id="header1Label" stylesheets="@style.css"
text="Hello!" />
            <Label id="header2Label" stylesheets="@style.css"
text="Please choose your card:" />
            <GridPane hgap="5.0">
              <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" minWidth="
10.0" prefWidth="150.0" />
                <ColumnConstraints hgrow="SOMETIMES" minWidth="
10.0" prefWidth="50.0" />
              </columnConstraints>
              <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30
.0" vgrow="SOMETIMES" />
              </rowConstraints>
               <children>
                  <ComboBox fx:id="selectCard" onAction="#
onSelect" prefHeight="25.0" prefWidth="154.0" promptText="
select your card">
                     <tooltip>
                        <Tooltip text="Select the microSD card
for your device." />
                     </tooltip></ComboBox>
                  <Button fx:id="onNext" mnemonicParsing="false
" onAction="#onNext" prefHeight="25.0" prefWidth="41.0" text="
Next" GridPane.columnIndex="1" />
```

```
                </children>
                <TilePane.margin>
                   <Insets />
                </TilePane.margin>
             </GridPane>
             <Label id="feedbackLabel" fx:id="feedbackLabel" 
stylesheets="@style.css" />
          </children>
       </TilePane>
          </children>
</Pane>
```

```java
package dlsu;

import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

import javafx.event.ActionEvent;

import java.io.IOException;

/**
 * Created by aenon on 11/11/2017.
 */

public class changeScene {

    public void setScene(String fxmlFile, String cssFile,
ActionEvent actionEvent, String title) throws IOException {
        Parent parent = FXMLLoader.load(getClass().getResource(
fxmlFile));
        parent.getStylesheets().addAll(this.getClass().
getResource(cssFile).toExternalForm());
        Scene scene = new Scene(parent);
        Stage stage = (Stage) ((Node) actionEvent.getSource()).
getScene().getWindow();
        stage.setScene(scene);
        stage.setTitle(title);
        stage.show();
    }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<Pane id="setupPane" prefHeight="575.5" prefWidth="1024"
stylesheets="@style.css" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="dlsu.
setUpFinishController">
    <children>
        <Label id="header1Label" layoutX="379.0" layoutY="288.0
" stylesheets="@style.css" text="Congratulations!"
textAlignment="CENTER" />
      <TilePane alignment="CENTER" layoutX="360.0" layoutY="347
.0" prefHeight="67.0" prefWidth="183.0" vgap="5.0">
         <children>
              <Label id="header2Label" stylesheets="@style.css"
 text="You have successfully setup your card!" textAlignment="
CENTER" />
              <Label id="header2Label" stylesheets="@style.css"
 text="Please safely remove the card," textAlignment="CENTER"
/>
            <Label id="header2Label" layoutX="114.0" layoutY="
71.0" stylesheets="@style.css" text="and insert it to your
device." textAlignment="CENTER" />
         </children>
      </TilePane>
        <Button id="start" fx:id="onFinish" layoutX="468.0"
layoutY="437.0" mnemonicParsing="false" onAction="#onFinish"
prefHeight="38.0" prefWidth="88.0" stylesheets="@style.css"
text="finish" />
    </children>
</Pane>
```

```
package dlsu;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;

import java.io.*;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUp1Controller implements Initializable {
    public Button first;
    public Button second;
    public Button third;
    public Button fourth;
    public Button finish;
    @FXML
    private TextField username;
    @FXML
    private PasswordField confirm;
    @FXML
    private PasswordField password;
    @FXML
    private Label feedbackLabel;

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        File loginFile = new File(selectCardController.
driveLetter + "login.txt");
        if(loginFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));

        username.setText(br.readLine());
        String temp = br.readLine();
        password.setText(temp);
        confirm.setText(temp);

      } catch (FileNotFoundException e) {
        e.printStackTrace();
      } catch (IOException e) {
        e.printStackTrace();
      } finally {
        try {
          br.close();
        } catch (IOException e) {
          e.printStackTrace();
        }
      }
    }

    File profileFile = new File(selectCardController.
driveLetter + "profile.txt");
    if (!profileFile.exists()){
      third.setDisable(true);
    }

    File respondentsFile = new File(selectCardController.
driveLetter + "respondents.txt");
    if (!respondentsFile.exists()){
      fourth.setDisable(true);
    }

    finish.setDisable(true);

    first.setOnMouseEntered(event -> first.setCursor(Cursor
.HAND));
    first.setOnMousePressed(event -> first.setCursor(Cursor
.CLOSED_HAND));
    first.setOnMouseReleased(event -> first.setCursor(
Cursor.HAND));
    first.setOnMouseExited(event -> first.setCursor(Cursor.
DEFAULT));
```

```
        second.setOnMouseEntered(event -> second.setCursor(
Cursor.HAND));
        second.setOnMousePressed(event -> second.setCursor(
Cursor.CLOSED_HAND));
        second.setOnMouseReleased(event -> second.setCursor(
Cursor.HAND));
        second.setOnMouseExited(event -> second.setCursor(
Cursor.DEFAULT));

        third.setOnMouseEntered(event -> third.setCursor(Cursor
.HAND));
        third.setOnMousePressed(event -> third.setCursor(Cursor
.CLOSED_HAND));
        third.setOnMouseReleased(event -> third.setCursor(
Cursor.HAND));
        third.setOnMouseExited(event -> third.setCursor(Cursor.
DEFAULT));

        fourth.setOnMouseEntered(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMousePressed(event -> fourth.setCursor(
Cursor.CLOSED_HAND));
        fourth.setOnMouseReleased(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMouseExited(event -> fourth.setCursor(
Cursor.DEFAULT));

        finish.setOnMouseEntered(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMousePressed(event -> finish.setCursor(
Cursor.CLOSED_HAND));
        finish.setOnMouseReleased(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMouseExited(event -> finish.setCursor(
Cursor.DEFAULT));
    }

    public void onFirst(ActionEvent actionEvent) {
    }

    public void onSecond(ActionEvent actionEvent) throws
```

```
IOException {
        if (username.getText().isEmpty() || password.getText().
isEmpty() || confirm.getText().isEmpty()){
            feedbackLabel.setText("All fields are required!");
        }else{
            if(password.getText().equals(confirm.getText())){
                feedbackLabel.setText("");
                BufferedWriter writer = null;

                try{
                    writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "login.txt"), "utf-8"));
                    writer.write(username.getText());
                    writer.newLine();
                    writer.write(password.getText());
                    writer.close();
                }catch(Exception e){
                }

                changeScene changeScene = new changeScene();
                changeScene.setScene("Setup2.fxml", "style.css"
, actionEvent, "Fall Analyzer | Profile Setup");

            }else {
                feedbackLabel.setText("Your passwords doesn't
match!");
            }
        }

    }

    public void onThird(ActionEvent actionEvent) throws
IOException {
        if(password.getText().equals(confirm.getText())){
            feedbackLabel.setText("");
            BufferedWriter writer = null;

            try{
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "login.txt"), "utf-8"));
```

```
                writer.write(username.getText());
                writer.newLine();
                writer.write(password.getText());
                writer.close();
            }catch(Exception e){
            }

            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup3.fxml", "style.css",
actionEvent, "Fall Analyzer | Responders Setup");

        }else {
            feedbackLabel.setText("Your passwords doesn't match
!");
        }
    }

    public void onFourth(ActionEvent actionEvent) throws
IOException {
        if(password.getText().equals(confirm.getText())){
            feedbackLabel.setText("");
            BufferedWriter writer = null;

            try{
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "login.txt"), "utf-8"));
                writer.write(username.getText());
                writer.newLine();
                writer.write(password.getText());
                writer.close();
            }catch(Exception e){
            }

            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup4.fxml", "style.css",
actionEvent, "Fall Analyzer | Finish Setup");

        }else {
            feedbackLabel.setText("Your passwords doesn't match
!");
        }
```

```java
    }

    public void onFinish(ActionEvent actionEvent) {
    }
}
```

```
package dlsu;

import dlsu.Utils.checkContactNumber;
import javafx.event.ActionEvent;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.*;

import java.io.*;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUp2Controller implements Initializable {
    public Button first;
    public Button second;
    public Button third;
    public Button fourth;
    public ChoiceBox gender;
    public TextField firstName;
    public TextField LastName;
    public TextField age;
    public TextField address;
    public TextField contactNumber;
    public Label feedbackLabel;
    public Button finish;

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        File file = new File(selectCardController.driveLetter +
 "profile.txt");
        if(file.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "profile.txt"));

                firstName.setText(br.readLine());
                LastName.setText(br.readLine());
                gender.setValue(br.readLine());
                age.setText(br.readLine());
                address.setText(br.readLine());
                contactNumber.setText(br.readLine());

            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        gender.getItems().add("male");
        gender.getItems().add("female");

        File respondentsFile = new File(selectCardController.
driveLetter + "respondents.txt");
        if (!respondentsFile.exists()){
            fourth.setDisable(true);
        }

        finish.setDisable(true);

        first.setOnMouseEntered(event -> first.setCursor(Cursor
.HAND));
        first.setOnMousePressed(event -> first.setCursor(Cursor
.CLOSED_HAND));
        first.setOnMouseReleased(event -> first.setCursor(
Cursor.HAND));
        first.setOnMouseExited(event -> first.setCursor(Cursor.
DEFAULT));

        second.setOnMouseEntered(event -> second.setCursor(
Cursor.HAND));
        second.setOnMousePressed(event -> second.setCursor(
Cursor.CLOSED_HAND));
```

```
        second.setOnMouseReleased(event -> second.setCursor(
Cursor.HAND));
        second.setOnMouseExited(event -> second.setCursor(
Cursor.DEFAULT));

        third.setOnMouseEntered(event -> third.setCursor(Cursor
.HAND));
        third.setOnMousePressed(event -> third.setCursor(Cursor
.CLOSED_HAND));
        third.setOnMouseReleased(event -> third.setCursor(
Cursor.HAND));
        third.setOnMouseExited(event -> third.setCursor(Cursor.
DEFAULT));

        fourth.setOnMouseEntered(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMousePressed(event -> fourth.setCursor(
Cursor.CLOSED_HAND));
        fourth.setOnMouseReleased(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMouseExited(event -> fourth.setCursor(
Cursor.DEFAULT));

        finish.setOnMouseEntered(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMousePressed(event -> finish.setCursor(
Cursor.CLOSED_HAND));
        finish.setOnMouseReleased(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMouseExited(event -> finish.setCursor(
Cursor.DEFAULT));
    }

    public void onFirst(ActionEvent actionEvent) throws
IOException {
            try{
                Integer.parseInt(age.getText());

                BufferedWriter writer = null;
                try{
                    writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "profile.txt"), "utf-8"));
                    writer.write(firstName.getText());
                    writer.newLine();
                    writer.write(LastName.getText());
                    writer.newLine();
                    writer.write(gender.getValue().toString());
                    writer.newLine();
                    writer.write(age.getText());
                    writer.newLine();
                    writer.write(address.getText());
                    writer.newLine();
                    writer.write(contactNumber.getText());
                    writer.close();
                } catch(Exception e){
                }
            }catch(Exception e){
                feedbackLabel.setText("Your age is invalid!");
            }finally {
                feedbackLabel.setText("");
                changeScene changeScene = new changeScene();
                changeScene.setScene("Setup1.fxml", "style.css"
, actionEvent, "Fall Analyzer | Login Credentials");
            }
    }

    public void onSecond(ActionEvent actionEvent) {
    }

    public void onThird(ActionEvent actionEvent) throws
IOException {
        if(checkContactNumber.validContactNumber(contactNumber
.getText())) {
            if (firstName.getText().isEmpty() || LastName.
getText().isEmpty() || age.getText().isEmpty() || address.
getText().isEmpty() || contactNumber.getText().isEmpty()) {
                feedbackLabel.setText("All fields are required
!");
            } else {
                try {
                    Integer.parseInt(age.getText());

                    BufferedWriter writer = null;
                    try {
                        writer = new BufferedWriter(new 
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "profile.txt"), "utf-8"));
                        writer.write(firstName.getText());
                        writer.newLine();
                        writer.write(LastName.getText());
                        writer.newLine();
                        writer.write(gender.getValue().toString
());

                        writer.newLine();
                        writer.write(age.getText());
                        writer.newLine();
                        writer.write(address.getText());
                        writer.newLine();
                        writer.write(contactNumber.getText());
                        writer.close();
                    } catch (Exception e) {
                    }
                } catch (Exception e) {
                    feedbackLabel.setText("Your age is invalid
!");
                } finally {
                    feedbackLabel.setText("");
                    changeScene changeScene = new changeScene()
;

                    changeScene.setScene("Setup3.fxml", "style.
css", actionEvent, "Fall Analyzer | Responders Setup");
                }
            }
        } else {
            feedbackLabel.setText("Your contact number is 
invalid!");
        }
    }

    public void onFourth(ActionEvent actionEvent) throws 
IOException {
        if(checkContactNumber.validContactNumber(contactNumber
.getText())) {
            try{
                Integer.parseInt(age.getText());
                BufferedWriter writer = null;
                try{
                    writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "profile.txt"), "utf-8"));
                    writer.write(firstName.getText());
                    writer.newLine();
                    writer.write(LastName.getText());
                    writer.newLine();
                    writer.write(gender.getValue().toString());
                    writer.newLine();
                    writer.write(age.getText());
                    writer.newLine();
                    writer.write(address.getText());
                    writer.newLine();
                    writer.write(contactNumber.getText());
                    writer.close();
                } catch(Exception e){
                }
            }catch(Exception e){
                feedbackLabel.setText("Your age is invalid!");
            }finally {
                feedbackLabel.setText("");
                changeScene changeScene = new changeScene();
                changeScene.setScene("Setup4.fxml", "style.css"
, actionEvent, "Fall Analyzer | Finish Setup");
            }
        } else {
            feedbackLabel.setText("Your contact number is
invalid!");
        }
    }

    public void onFinish(ActionEvent actionEvent) {

    }
}
```

```
package dlsu;

import dlsu.Utils.checkContactNumber;
import javafx.beans.property.SimpleStringProperty;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.*;

import java.io.*;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUp3Controller implements Initializable {
    public Button first;
    public Button second;
    public Button third;
    public Button fourth;
    public TextField firstName;
    public TextField LastName;
    public TextField contactNumber;
    public Label feedbackLabel;
    public ListView list;

    public final ObservableList <responders> data =
FXCollections.observableArrayList();
    public Button finish;

    public static class responders{
        private SimpleStringProperty firstName;
        private SimpleStringProperty lastName;
        private SimpleStringProperty contactNumber;

        private responders(String first, String last, String
contact){
            this.firstName = new SimpleStringProperty(first);
            this.lastName = new SimpleStringProperty(last);
            this.contactNumber = new SimpleStringProperty(
contact);
        }

        public String getFirstName(){
            return firstName.get();
        }

        public void setFirstName(String first){
            firstName.set(first);
        }

        public String getLastName(){
            return lastName.get();
        }

        public void setLastName(String last){
            lastName.set(last);
        }

        public String getContactNumber(){
            return contactNumber.get();
        }

        public void setContactNumber(String contact){
            contactNumber.set(contact);
        }
    }

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        File respondentsFile = new File(selectCardController.
driveLetter + "respondents.txt");
        if (respondentsFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "respondents.txt"));

                Boolean end = false;
                String line;
                String first = "";
                String last = "";
                String contact = "";
                while(end == false){
                    line = br.readLine();
                    if (line.equals("-end-")){
                        end = true;
                    } else if (line.equals("breakLine")){
                        //just do nothing
                    } else{
                        first = line;
                        last = br.readLine();
                        contact = br.readLine();
                        data.add(new responders(first, last,
contact));

                        list.getItems().add(first + " " + last)
;
                    }
                }
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        finish.setDisable(true);

        first.setOnMouseEntered(event -> first.setCursor(Cursor
.HAND));
        first.setOnMousePressed(event -> first.setCursor(Cursor
.CLOSED_HAND));
        first.setOnMouseReleased(event -> first.setCursor(
Cursor.HAND));
        first.setOnMouseExited(event -> first.setCursor(Cursor.
DEFAULT));
```

```
        second.setOnMouseEntered(event -> second.setCursor(
Cursor.HAND));
        second.setOnMousePressed(event -> second.setCursor(
Cursor.CLOSED_HAND));
        second.setOnMouseReleased(event -> second.setCursor(
Cursor.HAND));
        second.setOnMouseExited(event -> second.setCursor(
Cursor.DEFAULT));

        third.setOnMouseEntered(event -> third.setCursor(Cursor
.HAND));
        third.setOnMousePressed(event -> third.setCursor(Cursor
.CLOSED_HAND));
        third.setOnMouseReleased(event -> third.setCursor(
Cursor.HAND));
        third.setOnMouseExited(event -> third.setCursor(Cursor.
DEFAULT));

        fourth.setOnMouseEntered(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMousePressed(event -> fourth.setCursor(
Cursor.CLOSED_HAND));
        fourth.setOnMouseReleased(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMouseExited(event -> fourth.setCursor(
Cursor.DEFAULT));

        finish.setOnMouseEntered(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMousePressed(event -> finish.setCursor(
Cursor.CLOSED_HAND));
        finish.setOnMouseReleased(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMouseExited(event -> finish.setCursor(
Cursor.DEFAULT));
    }

    public void onAdd(ActionEvent actionEvent) {
        if (firstName.getText().isEmpty() || LastName.getText()
.isEmpty() || contactNumber.getText().isEmpty()){
            feedbackLabel.setText("All fields are required!");
        }else{
            if (checkContactNumber.validContactNumber(
contactNumber.getText())){
                list.getItems().add(firstName.getText() + " " +
 LastName.getText());
                data.add(new responders(firstName.getText(),
LastName.getText(), contactNumber.getText()));
                firstName.setText("");
                LastName.setText("");
                contactNumber.setText("+639");
                feedbackLabel.setText("");
            }else {
                feedbackLabel.setText("Your contact number is
invalid!");
                contactNumber.setText("+639");
            }
        }
    }

    public void onDelete(ActionEvent actionEvent) {
        try{
            int index   = list.getSelectionModel().
getSelectedIndex();
            list.getItems().remove(index);
            data.remove(index);
        }catch (Exception e){
        }
    }

    public void onEdit(ActionEvent actionEvent) {
        try{
            int index   = list.getSelectionModel().
getSelectedIndex();
            firstName.setText(String.valueOf(data.get(index).
getFirstName()));
            LastName.setText(String.valueOf(data.get(index).
getLastName()));
            contactNumber.setText(String.valueOf(data.get(index
).getContactNumber()));
            data.remove(index);
            list.getItems().remove(index);
        }catch (Exception e){
        }
    }

    public void onFirst(ActionEvent actionEvent) throws
IOException {
        if (data.isEmpty()){
            feedbackLabel.setText("You must set at least one (1
) responder!");
        }else {
            BufferedWriter writer = null;
            try {
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "respondents.txt"), "utf-8"));

                int size = data.size();
                for (int i = 0; i < size; i++) {
                    writer.write(data.get(i).getFirstName());
                    writer.newLine();
                    writer.write(data.get(i).getLastName());
                    writer.newLine();
                    writer.write(data.get(i).getContactNumber()
);

                    writer.newLine();
                    writer.write("breakLine");
                    writer.newLine();
                }
                writer.write("-end-");
                writer.close();
            } catch (Exception e) {
            }
            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup1.fxml", "style.css",
actionEvent, "Fall Analyzer | Login Credentials");
        }
    }

    public void onSecond(ActionEvent actionEvent) throws
IOException {
        if (data.isEmpty()){
            feedbackLabel.setText("You must set at least one (1
) responder!");
        }else {
            BufferedWriter writer = null;
            try {
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "respondents.txt"), "utf-8"));

                int size = data.size();
                for (int i = 0; i < size; i++) {
                    writer.write(data.get(i).getFirstName());
                    writer.newLine();
                    writer.write(data.get(i).getLastName());
                    writer.newLine();
                    writer.write(data.get(i).getContactNumber()
);
                    writer.newLine();
                    writer.write("breakLine");
                    writer.newLine();
                }
                writer.write("-end-");
                writer.close();
            } catch (Exception e) {
            }
            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup2.fxml", "style.css",
actionEvent, "Fall Analyzer | Profile Setup");
        }
    }

    public void onThird(ActionEvent actionEvent) {
    }

    public void onFourth(ActionEvent actionEvent) throws
IOException {
        if (data.isEmpty()){
            feedbackLabel.setText("You must set at least one (1
) responder!");
        }else{
            BufferedWriter writer = null;
            try{
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "respondents.txt"), "utf-8"));

                int size = data.size();
                for (int i = 0; i < size; i++){
                    writer.write(data.get(i).getFirstName());
                    writer.newLine();
                    writer.write(data.get(i).getLastName());
                    writer.newLine();
                    writer.write(data.get(i).getContactNumber()
);

                    writer.newLine();
                    writer.write("breakLine");
                    writer.newLine();
                }
                writer.write("-end-");
                writer.close();
            } catch(Exception e){
            }
            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup4.fxml", "style.css",
actionEvent, "Fall Analyzer | Finish Setup");
        }

    }

    public void onFinish(ActionEvent actionEvent) {
    }
}
```

```
package dlsu;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;

import java.io.*;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUp4Controller implements Initializable {
    public Button first;
    public Button second;
    public Button third;
    public Button fourth;
    public Button finish;
    @FXML
    private TextField username;
    @FXML
    private PasswordField password;
    @FXML
    private Label feedbackLabel;

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        first.setOnMouseEntered(event -> first.setCursor(Cursor
.HAND));
        first.setOnMousePressed(event -> first.setCursor(Cursor
.CLOSED_HAND));
        first.setOnMouseReleased(event -> first.setCursor(
Cursor.HAND));
        first.setOnMouseExited(event -> first.setCursor(Cursor.
DEFAULT));
```

```
        second.setOnMouseEntered(event -> second.setCursor(
Cursor.HAND));
        second.setOnMousePressed(event -> second.setCursor(
Cursor.CLOSED_HAND));
        second.setOnMouseReleased(event -> second.setCursor(
Cursor.HAND));
        second.setOnMouseExited(event -> second.setCursor(
Cursor.DEFAULT));

        third.setOnMouseEntered(event -> third.setCursor(Cursor
.HAND));
        third.setOnMousePressed(event -> third.setCursor(Cursor
.CLOSED_HAND));
        third.setOnMouseReleased(event -> third.setCursor(
Cursor.HAND));
        third.setOnMouseExited(event -> third.setCursor(Cursor.
DEFAULT));

        fourth.setOnMouseEntered(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMousePressed(event -> fourth.setCursor(
Cursor.CLOSED_HAND));
        fourth.setOnMouseReleased(event -> fourth.setCursor(
Cursor.HAND));
        fourth.setOnMouseExited(event -> fourth.setCursor(
Cursor.DEFAULT));

        finish.setOnMouseEntered(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMousePressed(event -> finish.setCursor(
Cursor.CLOSED_HAND));
        finish.setOnMouseReleased(event -> finish.setCursor(
Cursor.HAND));
        finish.setOnMouseExited(event -> finish.setCursor(
Cursor.DEFAULT));
    }

    public void onFirst(ActionEvent actionEvent) throws
IOException {
        String user = "";
        String pass = "";
        File loginFile = new File(selectCardController.
driveLetter + "login.txt");
        if (loginFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
                user = br.readLine();
                pass = br.readLine();
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup1.fxml", "style.css",
actionEvent, "Fall Analyzer | Login Credentials");

        }
    }

    public void onSecond(ActionEvent actionEvent) throws
IOException {
        String user = "";
        String pass = "";

        File loginFile = new File(selectCardController.
driveLetter + "login.txt");
        if (loginFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
```

```
                user = br.readLine();
                pass = br.readLine();
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup2.fxml", "style.css",
actionEvent, "Fall Analyzer | Profile Setup");

        }
    }

    public void onThird(ActionEvent actionEvent) throws
IOException {
        String user = "";
        String pass = "";

        File loginFile = new File(selectCardController.
driveLetter + "login.txt");
        if (loginFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
                user = br.readLine();
                pass = br.readLine();
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

            feedbackLabel.setText("");
            changeScene changeScene = new changeScene();
            changeScene.setScene("Setup3.fxml", "style.css",
actionEvent, "Fall Analyzer | Responders Setup");

        }
    }

    public void onFourth(ActionEvent actionEvent) throws
IOException {
    }

    public void onFinish(ActionEvent actionEvent) throws
IOException {
        String user = "";
        String pass = "";

        File loginFile = new File(selectCardController.
driveLetter + "login.txt");
        if (loginFile.exists()){
            BufferedReader br = null;
            try {
                br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
                user = br.readLine();
                pass = br.readLine();
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

            if (username.getText().isEmpty() || password.getText().isEmpty()){
                feedbackLabel.setText("All fields are required!");
            }else if(!username.getText().equals(user)){
                feedbackLabel.setText("Invalid username!");
            }else if(!password.getText().equals(pass)){
                feedbackLabel.setText("Invalid password!");
            }else{
                feedbackLabel.setText("");
                changeScene changeScene = new changeScene();
                changeScene.setScene("SetupFinish.fxml", "style.css", actionEvent, "Fall Analyzer | Remove card");
            }
        }
    }
}
```

```
package dlsu;

import com.sun.org.apache.xpath.internal.SourceTree;
import dlsu.Utils.checkContactNumber;
import dlsu.Utils.logInDialog;
import javafx.beans.property.SimpleStringProperty;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.Initializable;
import javafx.scene.Group;
import javafx.scene.chart.PieChart;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;

import javax.sound.midi.Soundbank;
import java.io.*;
import java.net.URL;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;

public class exploreController implements Initializable{

    public Label usernameHello;
    public TextField usernameLogin;
    public PasswordField passwordLogin;
    public PasswordField confirmLogin;
    public Label feedbackLabelLogin;
    public Button editLogin;
    public Button saveLogin;

    public TextField firstNameProfile;
    public TextField LastNameProfile;
    public ChoiceBox genderProfile;
    public TextField ageProfile;
    public TextField addressProfile;
    public TextField contactNumberProfile;
    public Label feedbackLabel;
    public Button editProfile;
    public Button saveProfile;

    public Group editResponders;
    public TableView respondersTable;
    public TableColumn firstName;
    public TableColumn lastName;
    public TableColumn contactNumber;

    public TextField firstNameResponders;
    public TextField LastNameResponders;
    public TextField contactNumberResponders;
    public Button saveResponder;
    public Label allFieldsError;
    public Label successful;
    public Button deleteResponder;
    public Button editResponder;
    public Button addResponder;

    public TableView activityTable;
    public TableColumn dateCell;
    public TableColumn timeCell;
    public TableColumn activityCell;
    public TableColumn locationCell;
    public Label feedbackData;
    public Button deleteData;
    public Button deleteAllData;
    public Button exportData;
    public Label numberOfFalls;
    public PieChart storageSpace;

    public int fallTimes = 0;
    public Tab profileTab;
    public Tab respondersTab;
    public Tab activityTab;

    dlsu.Utils.logInDialog logInDialog = new logInDialog();

    public final ObservableList<responders> data =
FXCollections.observableArrayList();
```

```
    public final ObservableList<activities> log = FXCollections
.observableArrayList();

    public static class activities{
        private SimpleStringProperty date;
        private SimpleStringProperty time;
        private SimpleStringProperty activity;
        private SimpleStringProperty location;

        private activities(String dateLog, String timeLog,
String activityLog, String locationLog){
            this.date = new SimpleStringProperty(dateLog);
            this.time = new SimpleStringProperty(timeLog);
            this.activity = new SimpleStringProperty(
activityLog);
            this.location = new SimpleStringProperty(
locationLog);
        }

        public String getDate() {
            return date.get();
        }

        public void setDate(String dateLog) {
            date.set(dateLog);
        }

        public String getTime() {
            return time.get();
        }

        public void setTime(String timeLog) {
            time.set(timeLog);
        }

        public String getActivity() {
            return activity.get();
        }

        public void setActivity(String activityLog) {
            activity.set(activityLog);
        }
        public String getLocation() {
            return location.get();
        }

        public void setLocation(String locationLog) {
            location.set(locationLog);
        }
    }

    public static class responders{
        private SimpleStringProperty firstName;
        private SimpleStringProperty lastName;
        private SimpleStringProperty contactNumber;

        private responders(String first, String last, String
contact){
            this.firstName = new SimpleStringProperty(first);
            this.lastName = new SimpleStringProperty(last);
            this.contactNumber = new SimpleStringProperty(
contact);
        }

        public String getFirstName(){
            return firstName.get();
        }

        public void setFirstName(String first){
            firstName.set(first);
        }

        public String getLastName(){
            return lastName.get();
        }

        public void setLastName(String last){
            lastName.set(last);
        }

        public String getContactNumber(){
            return contactNumber.get();
        }
        public void setContactNumber(String contact){
            contactNumber.set(contact);
        }
    }

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        initProfileTab();
        initRepondersTab();
        initActivityTab();
    }

    private void initActivityTab() {
        File dir = new File(selectCardController.driveLetter +
"Activities");
        if (dir.isDirectory()){
            for(File file : dir.listFiles()){
                if (file.isFile()){
                    BufferedReader br = null;
                    try {
                        br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "Activities\\" + file.
getName()));
                        String line = "";
                        String date = "";
                        String time = "";
                        String activity = "";
                        String location = "";
                        String toExtract = "";

                        while((line = br.readLine()) != null){
                            if (line.equals("-end_of_activity-"
)){
                                //do nothing
                            } else {
                                //extract rtc and coordinates
here.
                                String rtc = "";
                                String latitude = "";
                                String longitude = "";
                                br.readLine();
                                toExtract = br.readLine();

                                int comma = 0;
                                int g = 0;

                                while(comma < 5){
                                    if (toExtract.charAt(g) ==
','){
                                        comma++;
                                        g++;
                                    }
                                    if (comma == 2){
                                        rtc = rtc + toExtract.
charAt(g);
                                    }
                                    if (comma == 3){
                                        longitude = longitude +
 toExtract.charAt(g);
                                    }
                                    if (comma == 4){
                                        latitude = latitude +
toExtract.charAt(g);
                                    }

                                    g++;
                                }

                                String year = "";
                                String month = "";
                                String day = "";

                                String hour = "";
                                String min = "";
                                String sec = "";

                                //extract date and time here.
                                for (int i = 0; i < 14; i++){
                                    if (i < 8){
                                        if (i < 4){
                                            year = year + rtc.
charAt(i);
                                        }
                                        if (i > 3 && i < 6){
                                            month = month + rtc
.charAt(i);
                                        }
                                        if (i > 5 && i < 8){
                                            day = day + rtc.
charAt(i);
                                        }
                                    } else{
                                        if (i == 8 || i == 9){
                                            hour = hour + rtc.
charAt(i);
                                        } else if (i == 10 || i
 == 11){
                                            min = min + rtc.
charAt(i);
                                        } else if (i == 12 || i
 == 13){
                                            sec = sec + rtc.
charAt(i);
                                        }
                                    }
                                }

                                //convert numbered months to
words:
                                date = getDate(year, month, day
);

                                //convert 24hour format to
12hour format (AM, PM, NN, MN)here:
                                time = getTime(hour, min, sec);

                                //combine coordinates here:
                                location = longitude + ", " +
latitude;

                                br.readLine();
                                br.readLine();
                                br.readLine();
                                activity = br.readLine();

                                if(activity.contains("Falling!"
)){
                                    fallTimes++;
                                    numberOfFalls.setText(
String.valueOf(fallTimes));
                                }
                                log.add(new activities(date,
time, activity, location));
                                dateCell.setCellValueFactory(
new PropertyValueFactory<activities, String>("date"));
                                timeCell.setCellValueFactory(
new PropertyValueFactory<activities, String>("time"));
                                activityCell.
setCellValueFactory(new PropertyValueFactory<activities, String
>("activity"));
                                locationCell.
setCellValueFactory(new PropertyValueFactory<activities, String
>("location"));
                                activityTable.getItems().setAll
(this.log);
                            }
                        }

                    } catch (IOException e) {
                        System.out.println(e);
                    } finally {
                        try {
                            br.close();
                        } catch (IOException e) {
                            System.out.println(e);
                        }
                    }
                } else {
                    System.out.println("NOT A FILE!");
                }
            }
        } else {
            System.out.println("NOT A DIRECTORY!");
        }
        setStorageChart();
    }

    private void initRepondersTab(){
        BufferedReader br = null;
        try {
            br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "respondents.txt"));

            Boolean end = false;
            String line = "";
            String first = "";
            String last = "";
            String contact = "";
            while (end == false) {
                line = br.readLine();
                if (line.equals("-end-")) {
                    end = true;
                } else if (line.equals("breakLine")) {
                    //just do nothing
                } else {
                    first = line;
                    last = br.readLine();
                    contact = br.readLine();
                    data.add(new responders(first, last,
contact));
                    firstName.setCellValueFactory(new
PropertyValueFactory<responders, String>("firstName"));
                    lastName.setCellValueFactory(new
PropertyValueFactory<responders, String>("lastName"));
                    contactNumber.setCellValueFactory(new
PropertyValueFactory<responders, String>("contactNumber"));
                    respondersTable.getItems().setAll(this.data
);
                }
            }
        } catch (IOException e) {
        } finally {
            try {
                br.close();
            } catch (IOException e) {
            }
        }
    }

    private void initProfileTab() {
        BufferedReader br = null;
        try {
            br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
            String temp = br.readLine();
            usernameHello.setText(temp + "!");
            usernameLogin.setText(temp);
            temp = br.readLine();
            passwordLogin.setText(temp);
            confirmLogin.setText(temp);
        } catch (IOException e) {
        } finally {
            try {
                br.close();
            } catch (IOException e) {
            }
        }

        genderProfile.getItems().add("male");
        genderProfile.getItems().add("female");
        try {
            br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "profile.txt"));
            firstNameProfile.setText(br.readLine());
            LastNameProfile.setText(br.readLine());
            genderProfile.setValue(br.readLine());
            ageProfile.setText(br.readLine());
            addressProfile.setText(br.readLine());
            contactNumberProfile.setText(br.readLine());
        } catch (IOException e) {
        } finally {
            try {
                br.close();
            } catch (IOException e) {
            }
        }

    }
    private void setStorageChart() {
        File space = new File(selectCardController.driveLetter)
;
        long freespace = space.getFreeSpace() / (1024*1024*1024
); //outputs in GB. (1024*1024) if MB
        long totalspace = space.getTotalSpace() / (1024*1024*
1024); //outputs  in*GB (1024*1024) if MB
        long usedspace = (space.getTotalSpace() - space.
getFreeSpace()) / (1024*1024*1024); //outputs  in*GB (1024*1024
) if MB

        ObservableList<PieChart.Data> pieDate =
                FXCollections.observableArrayList(
                        new PieChart.Data("Free Space: " +
freespace + " GB", freespace),
                        new PieChart.Data("Used Space: " +
usedspace + " GB", usedspace)
                );
        storageSpace.setData(pieDate);
        storageSpace.setLabelLineLength(10);
        storageSpace.setLabelsVisible(false);

        storageSpace.getData().stream().forEach(data -> {
            Tooltip tooltip = new Tooltip();
            tooltip.setText(String.format("%.1f%%", 100*data.
getPieValue()/totalspace));
            Tooltip.install(data.getNode(), tooltip);
            data.pieValueProperty().addListener((observable,
oldValue, newValue) ->
                    tooltip.setText(newValue + "%"));
        });
    }

    private String getDate(String year, String month, String
day){
        switch (month){
            case "01":
                month = "January";
                break;
            case "02":
                month = "February";
                break;
            case "03":
                month = "March";
                break;
            case "04":
                month = "April";
                break;
            case "05":
                month = "May";
                break;
            case "06":
                month = "June";
                break;
            case "07":
                month = "July";
                break;
            case "08":
                month = "August";
                break;
            case "09":
                month = "September";
                break;
            case "10":
                month = "October";
                break;
            case "11":
                month = "November";
                break;
            case "12":
                month = "December";
                break;
        }

        String date = month + " " + day + ", " + year;
        return  date;
    }

    private String getTime(String hour, String min, String sec)
{
        String time = "";
        int hh = Integer.parseInt(hour);
        System.out.println("hh raw: " + hh);

        if (hh > 12){
            hh = hh - 12;
        }

        System.out.println("hh 12hr: " + hh);

        hh = hh + 8; //convert to UTC+8:00

        System.out.println("hh utc: " + hh);

        if (hh > 12){
            hh = hh - 12;

            if (hh == 12){
                time = time + hh + ":" + min + ":" + sec + " M.
N.";
            } else{
                time = time + hh + ":" + min + ":" + sec + " P.
M.";
            }

        } else if (hh < 13){
            if (hh == 12) {
                time = time + hh + ":" + min + ":" + sec + " N.
N.";
            } else if (hh == 0) {
                time = time + "12" + ":" + min + ":" + sec + "
A.M.";
            } else {
                time = time + hh + ":" + min + ":" + sec + " A.
M.";
            }
        }

        System.out.println("TIME: " + time);

        return time;
    }

    public void onEditLogin(ActionEvent actionEvent) {
        if(logInDialog.logIn()){
            usernameLogin.setEditable(true);
            passwordLogin.setEditable(true);
            confirmLogin.setEditable(true);
            saveLogin.setDisable(false);
            feedbackLabelLogin.setText("");
        }
    }

    public void onSaveLogin(ActionEvent actionEvent) {
        if(usernameLogin.getText().isEmpty() || passwordLogin.
getText().isEmpty() || confirmLogin.getText().isEmpty()){
            feedbackLabelLogin.setText("All fields are required
!");
        } else if (!passwordLogin.getText().equals(confirmLogin
.getText())){
            feedbackLabelLogin.setText("Passwords do not match
!");
        } else {
            usernameHello.setText(usernameLogin.getText() + "!"
);
            BufferedWriter writer = null;
            try {
                writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "login.txt"), "utf-8"));
                writer.write(usernameLogin.getText());
                writer.newLine();
                writer.write(passwordLogin.getText());
                writer.close();

                feedbackLabelLogin.setText("Login Credentials
was successfully modified!");
                usernameLogin.setEditable(false);
                passwordLogin.setEditable(false);
                confirmLogin.setEditable(false);
                saveLogin.setDisable(true);
                setStorageChart();
            } catch (Exception e) {
            }
        }
    }

    public void onEditProfile(ActionEvent actionEvent) {
        if(logInDialog.logIn()){

            firstNameProfile.setEditable(true);
            LastNameProfile.setEditable(true);
            genderProfile.setDisable(false);
            ageProfile.setEditable(true);
            addressProfile.setEditable(true);
            contactNumberProfile.setEditable(true);
            saveProfile.setDisable(false);
            feedbackLabel.setText("");
        }
    }

    public void onSaveProfile(ActionEvent actionEvent) {
        if(firstNameProfile.getText().isEmpty() ||
LastNameProfile.getText().isEmpty() || ageProfile.getText().
isEmpty() || addressProfile.getText().isEmpty() ||
contactNumberProfile.getText().isEmpty()){
            feedbackLabel.setText("All fields are required!");
        } else {
            if (checkContactNumber.validContactNumber(
contactNumberProfile.getText())) {
                try {
                    Integer.parseInt(ageProfile.getText());

                    BufferedWriter writer = null;
                    try {
                        writer = new BufferedWriter(new
OutputStreamWriter(new FileOutputStream(selectCardController.
driveLetter + "profile.txt"), "utf-8"));
                        writer.write(firstNameProfile.getText()
);
                        writer.newLine();
                        writer.write(LastNameProfile.getText())
;
                        writer.newLine();
                        writer.write(genderProfile.getValue().
toString());
                        writer.newLine();
                        writer.write(ageProfile.getText());
                        writer.newLine();
                        writer.write(addressProfile.getText());
                        writer.newLine();
                        writer.write(contactNumberProfile.
getText());
                        writer.close();

                        feedbackLabel.setText("Profile was
successfully modified!");
                        firstNameProfile.setEditable(false);
                        LastNameProfile.setEditable(false);
                        genderProfile.setDisable(true);
                        ageProfile.setEditable(false);
                        addressProfile.setEditable(false);
                        contactNumberProfile.setEditable(false)
;
                        saveProfile.setDisable(true);
                        setStorageChart();
                    } catch (Exception e) {
                    }
                } catch (Exception e) {
                    feedbackLabel.setText("Your age is invalid
!");
                }
            } else{
                feedbackLabel.setText("Your contact number is
invalid!");
            }
        }
    }

    public void saveResponder(){
        BufferedWriter writer = null;
        try{
            writer = new BufferedWriter(new OutputStreamWriter(
new FileOutputStream(selectCardController.driveLetter + "
respondents.txt"), "utf-8"));

            int size = data.size();
            for (int i = 0; i < size; i++){
                writer.write(data.get(i).getFirstName());
                writer.newLine();
                writer.write(data.get(i).getLastName());
                writer.newLine();
                writer.write(data.get(i).getContactNumber());
                writer.newLine();
                writer.write("breakLine");
                writer.newLine();
            }
            writer.write("-end-");
            writer.close();
            setStorageChart();
        } catch(Exception e){
        }
    }

    public void onSaveResponder(ActionEvent actionEvent) {
        if (firstNameResponders.getText().isEmpty() ||
LastNameResponders.getText().isEmpty() ||
contactNumberResponders.getText().isEmpty()){
            allFieldsError.setText("All fields are required!");
            successful.setText("");
        } else {
            if (checkContactNumber.validContactNumber(
contactNumberResponders.getText())){
                allFieldsError.setText("");

                data.add(new responders(firstNameResponders.
getText(), LastNameResponders.getText(),
contactNumberResponders.getText()));
                firstName.setCellValueFactory(new
PropertyValueFactory<responders, String>("firstName"));
                lastName.setCellValueFactory(new
PropertyValueFactory<responders, String>("lastName"));
                contactNumber.setCellValueFactory(new
PropertyValueFactory<responders, String>("contactNumber"));
                respondersTable.getItems().setAll(this.data);

                firstNameResponders.setText("");
                LastNameResponders.setText("");
                contactNumberResponders.setText("");

                successful.setText("Responders were
successfully modified!");
                editResponders.setDisable(true);
                saveResponder();
            } else {
                allFieldsError.setText("Contact number is
invalid!");
                successful.setText("");
            }

        }
    }

    public void onDeleteResponder(ActionEvent actionEvent) {
        int selectedIndex = respondersTable.getSelectionModel()
.getSelectedIndex();
        if (selectedIndex >= 0){
            if (logInDialog.logIn()){
                respondersTable.getItems().remove(selectedIndex
);
                data.remove(selectedIndex);
                allFieldsError.setText("");
                successful.setText("Responders were
successfully modified!");
                saveResponder();
            }
        }
    }

    public void onEditResponder(ActionEvent actionEvent) {
        if(logInDialog.logIn()){
            int selectedIndex = respondersTable.
getSelectionModel().getSelectedIndex();
            if (selectedIndex >= 0){
                respondersTable.getItems().remove(selectedIndex
);
                firstNameResponders.setText(data.get(
selectedIndex).getFirstName());
                LastNameResponders.setText(data.get(
selectedIndex).getLastName());
                contactNumberResponders.setText(data.get(
selectedIndex).getContactNumber());
                data.remove(selectedIndex);
            }

            allFieldsError.setText("");
            successful.setText("");
            editResponders.setDisable(false);
        }
    }

    public void onAddResponder(ActionEvent actionEvent) {
        if (logInDialog.logIn()){
            contactNumberResponders.setText("+639");
            allFieldsError.setText("");
            successful.setText("");
            editResponders.setDisable(false);
        }
    }

    public void saveLog(){
        BufferedWriter writer = null;
        try{
            writer = new BufferedWriter(new OutputStreamWriter(
new FileOutputStream(selectCardController.driveLetter + "
activity.txt"), "utf-8"));

            int size = log.size();
            for (int i = 0; i < size; i++){
                writer.write(log.get(i).getDate());
                writer.newLine();
                writer.write(log.get(i).getTime());
                writer.newLine();
                writer.write(log.get(i).getActivity());
                writer.newLine();
                writer.write(log.get(i).getLocation());
                writer.newLine();
                writer.write("-end_of_activity-");
                writer.newLine();
            }
            writer.close();
            setStorageChart();
        } catch(Exception e){
        }
    }
    public void onDeleteData(ActionEvent actionEvent){
        int selectedIndex = activityTable.getSelectionModel().
getSelectedIndex();
        if (selectedIndex >= 0){
            if (logInDialog.logIn()){
                if(activityCell.getCellObservableValue(
selectedIndex).getValue().toString().contains("Falling!")){
                    fallTimes--;
                    numberOfFalls.setText(String.valueOf(
fallTimes));
                }
                activityTable.getItems().remove(selectedIndex);
                log.remove(selectedIndex);
                feedbackData.setText("Data log was successfully
 modified!");
                saveLog();
            }
        }

    }

    public void onDeleteAllData(ActionEvent actionEvent) {
        if (logInDialog.logIn()){
            fallTimes = 0;
            numberOfFalls.setText(String.valueOf(fallTimes));
            activityTable.getItems().clear();
            log.clear();
            feedbackData.setText("Data log was successfully
modified!");
            saveLog();
        }
    }

    public void onExportData(ActionEvent actionEvent) throws
IOException {
        HSSFWorkbook workbook = new HSSFWorkbook();
        HSSFSheet sheet = workbook.createSheet("Fall Analyzer
Data Log");

        HSSFRow row = null;
        HSSFCell cell = null;
        row = sheet.createRow(0);
        cell = row.createCell(0);
        cell.setCellValue("DATE");
        cell = row.createCell(1);
        cell.setCellValue("TIME");
        cell = row.createCell(2);
        cell.setCellValue("ACTIVITY");
        cell = row.createCell(3);
        cell.setCellValue("LOCATION");

        row = sheet.createRow(1);
        cell = row.createCell(0);
        cell.setCellValue("try");

        row = sheet.createRow(2);
        cell = row.createCell(1);
        cell.setCellValue("tr2y");

        for (int rowIterate = 0; rowIterate < activityTable.
getItems().size(); rowIterate++){
            row = sheet.createRow(rowIterate+1);
            for (int columnIterate = 0; columnIterate <
activityTable.getColumns().size(); columnIterate++){
                cell = row.createCell(columnIterate);
                switch (columnIterate){
                    case 0: //DATE
                        cell.setCellValue(dateCell.
getCellObservableValue(rowIterate).getValue().toString());
                        break;
                    case 1: //TIME
                        cell.setCellValue(timeCell.
getCellObservableValue(rowIterate).getValue().toString());
                        break;
                    case 2: //ACTIVITY
                        cell.setCellValue(activityCell.
getCellObservableValue(rowIterate).getValue().toString());
                        break;
                    case 3: //LOCATION
                        cell.setCellValue(locationCell.
getCellObservableValue(rowIterate).getValue().toString());
                        break;
                }
            }
        }

        sheet.autoSizeColumn(0);
        sheet.autoSizeColumn(1);
        sheet.autoSizeColumn(2);
        sheet.autoSizeColumn(3);

        DateFormat dateFormat = new SimpleDateFormat("
yyyyMMdd_HHmmss");
        Date date = new Date();
        String fileName = "Fall Analyzer Data Log " +
dateFormat.format(date) + ".xls";

        workbook.write(new FileOutputStream(fileName));
        workbook.close();

        feedbackData.setText("Data log was successfully
exported!");
    }
}
```

```
package dlsu;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.Button;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUpHomeController implements Initializable {

    @FXML
    private Button onNext;

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        onNext.setOnMouseEntered(event -> onNext.setCursor(
Cursor.HAND));
        onNext.setOnMousePressed(event -> onNext.setCursor(
Cursor.CLOSED_HAND));
        onNext.setOnMouseReleased(event -> onNext.setCursor(
Cursor.HAND));
        onNext.setOnMouseExited(event -> onNext.setCursor(
Cursor.DEFAULT));
    }

    @FXML
    private void onNext(ActionEvent actionEvent) throws
IOException {
        changeScene changeScene = new changeScene();
        changeScene.setScene("Setup1.fxml", "style.css",
actionEvent, "Fall Analyzer | Login Credentials");
    }

}
```

```
package dlsu;


import dlsu.Utils.getExternalDevices;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.*;

import java.io.*;
import java.net.URL;
import java.util.*;

public class selectCardController implements Initializable{
    dlsu.Utils.getExternalDevices getExternalDevices = new
getExternalDevices();

    @FXML
    private Button onNext;

    @FXML
    private Label feedbackLabel;

    @FXML
    private ComboBox selectCard;

    public static String driveLetter;


    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        getExternalDevices.getDevices();
        System.out.println("Storage Devices: " +
getExternalDevices.storages);
        for (int i = 0; i < getExternalDevices.storages.size();
 i++){
            selectCard.getItems().add(getExternalDevices.
storages.get(i));
        }
        onNext.setOnMouseEntered(event -> onNext.setCursor(
Cursor.HAND));
        onNext.setOnMousePressed(event -> onNext.setCursor(
Cursor.CLOSED_HAND));
        onNext.setOnMouseReleased(event -> onNext.setCursor(
Cursor.HAND));
        onNext.setOnMouseExited(event -> onNext.setCursor(
Cursor.DEFAULT));

        selectCard.setOnMouseEntered(event -> selectCard.
setCursor(Cursor.HAND));
        selectCard.setOnMousePressed(event -> selectCard.
setCursor(Cursor.CLOSED_HAND));
        selectCard.setOnMouseReleased(event -> selectCard.
setCursor(Cursor.HAND));
        selectCard.setOnMouseExited(event -> selectCard.
setCursor(Cursor.DEFAULT));
    }

    @FXML
    private void onNext(ActionEvent actionEvent) throws
IOException{
        List<String> messages = new ArrayList<String>();

        messages.add("Please select your card first!");
        messages.add("You haven't selected your card yet.");
        messages.add("Ooopsie! Please select your card please
.");
        messages.add("Select your card first.");

        Collections.shuffle(messages);

        try {
            String drive = selectCard.getValue().toString();
            driveLetter = "";
            for (int i = 0; i < drive.length(); i++) {
                if (Objects.equals(drive.charAt(i), '(')) {
                    i++;
                    driveLetter = driveLetter + drive.charAt(i)
 + ":" + "\\";
                }
            }
            System.out.println("Selected Drive: " + drive);
            System.out.println("Selected Drive Letter: " +
driveLetter);

            File loginFile = new File(driveLetter + "login.txt"
);
            File profileFile = new File(driveLetter + "profile.
txt");
            File respondentsFile = new File(driveLetter + "
respondents.txt");
            File activityFile = new File(driveLetter + "
Activities");
            File fileLog = new File(driveLetter + "filelog.txt"
);
            if (loginFile.exists() && profileFile.exists() &&
respondentsFile.exists() && activityFile.exists() && fileLog.
exists()) {
                feedbackLabel.setText("");
                changeScene changeScene = new changeScene();
                changeScene.setScene("Explore.fxml", "style.css
", actionEvent, "Fall Analyzer");
            } else if (!loginFile.exists() && !profileFile.
exists() && !respondentsFile.exists() && !activityFile.exists()
 && !fileLog.exists()){
                feedbackLabel.setText("");
                changeScene changeScene = new changeScene();
                changeScene.setScene("SetupHome.fxml", "style.
css", actionEvent, "Fall Analyzer | Home Setup");
            }else {
                System.out.println("Some of the important files
 are missing!\nCard must be initialized again.");

                feedbackLabel.setText("Some of the important
files are missing!\nCard must be initialized again.");

                Alert alert = new Alert(Alert.AlertType.ERROR);
                alert.setTitle("ERROR!");
                String s = "Some of the important files are
missing!\nCard must be initialized again.";
                alert.setContentText(s);

                Optional<ButtonType> result = alert.showAndWait
();
```

```
                changeScene changeScene = new changeScene();
                changeScene.setScene("SetupHome.fxml", "style.
css", actionEvent, "Fall Analyzer | Home Setup");
            }

        } catch (Exception e){
            feedbackLabel.setText(messages.get(0));
            System.out.println(e);
        }

    }

    public void onSelect(ActionEvent actionEvent) {
        List<String> messages = new ArrayList<String>();

        messages.add("Great! Click next to proceed");
        messages.add("Ok! Now, click next to proceed");
        messages.add("Alright! To proceed, click next");
        messages.add("Let's proceed by clicking the next
button");

        Collections.shuffle(messages);

        feedbackLabel.setText(messages.get(0));
    }

}
```

```java
package dlsu;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.Cursor;
import javafx.scene.control.Button;
import javafx.stage.Stage;

import java.io.*;
import java.net.URL;
import java.util.ResourceBundle;

/**
 * Created by aenon on 11/11/2017.
 */
public class setUpFinishController implements Initializable {

    @FXML
    private Button onFinish;

    @Override
    public void initialize(URL location, ResourceBundle
resources) {
        onFinish.setOnMouseEntered(event -> onFinish.setCursor(
Cursor.HAND));
        onFinish.setOnMousePressed(event -> onFinish.setCursor(
Cursor.CLOSED_HAND));
        onFinish.setOnMouseReleased(event -> onFinish.
setCursor(Cursor.HAND));
        onFinish.setOnMouseExited(event -> onFinish.setCursor(
Cursor.DEFAULT));
    }

    @FXML
    private void onFinish(ActionEvent actionEvent) throws
IOException {
        BufferedWriter writer = null;
        try {
            writer = new BufferedWriter(new OutputStreamWriter(
new FileOutputStream(selectCardController.driveLetter + "
filelog.txt"), "utf-8"));
```

```
            writer.write("actFileCount = 0");
            writer.newLine();
            writer.write("actCounter = 0");
            writer.newLine();
            writer.close();
        } catch (Exception e){
        }

        File dir = new File(selectCardController.driveLetter +
"Activities");
        dir.mkdir();

        Stage stage = (Stage) onFinish.getScene().getWindow();
        stage.close();
    }

}
```

```
package dlsu.Utils;

import dlsu.selectCardController;
import javafx.application.Platform;
import javafx.geometry.Insets;
import javafx.scene.Node;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.util.Pair;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Optional;

/**
 * Created by aenon on 14/11/2017.
 */
public class logInDialog {
    public Boolean logIn(){
        String loginusername = "";
        String loginpassword = "";
        BufferedReader br = null;
        try {
            br = new BufferedReader(new FileReader(
selectCardController.driveLetter + "login.txt"));
            loginusername = br.readLine();
            loginpassword = br.readLine();
        } catch (IOException e) {
        } finally {
            try {
                br.close();
            } catch (IOException e) {
            }
        }

        // Create the custom dialog.
        Dialog<Pair<String, String>> dialog = new Dialog<>();
        dialog.setTitle("Login to continue!");
        dialog.setHeaderText("Please enter username and
password to continue.");
```

```
//// Set the icon (must be included in the project).
//            dialog.setGraphic(new ImageView(this.getClass().
getResource("Main.png").toString()));

// Set the button types.
        ButtonType loginButtonType = new ButtonType("Login",
ButtonBar.ButtonData.OK_DONE);
        dialog.getDialogPane().getButtonTypes().addAll(
loginButtonType, ButtonType.CANCEL);

// Create the username and password labels and fields.
        GridPane grid = new GridPane();
        grid.setHgap(10);
        grid.setVgap(10);
        grid.setPadding(new Insets(20, 150, 10, 10));

        TextField username = new TextField();
        username.setPromptText("Username");
        PasswordField password = new PasswordField();
        password.setPromptText("Password");

        grid.add(new Label("Username:"), 0, 0);
        grid.add(username, 1, 0);
        grid.add(new Label("Password:"), 0, 1);
        grid.add(password, 1, 1);

// Enable/Disable login button depending on whether a username
 was entered.
        Node loginButton = dialog.getDialogPane().lookupButton(
loginButtonType);
        loginButton.setDisable(true);

// Do some validation (using the Java 8 lambda syntax).
        username.textProperty().addListener((observable,
oldValue, newValue) -> {
            loginButton.setDisable(newValue.trim().isEmpty());
        });

        dialog.getDialogPane().setContent(grid);

// Request focus on the username field by default.
        Platform.runLater(() -> username.requestFocus());
```

```
// Convert the result to a username-password-pair when the
login button is clicked.
        dialog.setResultConverter(dialogButton -> {
            if (dialogButton == loginButtonType) {
                return new Pair<>(username.getText(), password.
getText());
            }
            return null;
        });

        Optional<Pair<String, String>> result = dialog.
showAndWait();

        final String[] user = {""};
        final String[] pass = {""};
        result.ifPresent(usernamePassword -> {
            user[0] = usernamePassword.getKey();
            pass[0] = usernamePassword.getValue();
            //System.out.println("Username=" + usernamePassword
.getKey() + ", Password=" + usernamePassword.getValue());
        });

        if (user[0].equals(loginusername) && pass[0].equals(
loginpassword)){
            return true;
        } else{
            return false;
        }
    }
}
```

```java
package dlsu.Utils;

/**
 * Created by aenon on 25/11/2017.
 */
public class checkContactNumber {
    public static Boolean validContactNumber(String
contactNumber){
        if (contactNumber.length() > 13){
            return false;
        } else{
            String num = contactNumber.substring(1);

            try{
                Long.parseLong(num);
                if (num.length() != 12){
                    return false;
                } else {
                    return true;
                }
            } catch (NumberFormatException e){
                return false;
            }
        }
    }
}
```

```
package dlsu.Utils;

import javax.swing.filechooser.FileSystemView;
import java.io.File;
import java.util.ArrayList;
import java.util.List;

/**
 * Created by aenon on 09/11/2017.
 */
public class getExternalDevices {

    public final List<String> storages = new ArrayList<String>(
);
    public final List<String> storageLetters = new ArrayList<
String>();

    public void getDevices(){

        FileSystemView fileSystemView = FileSystemView.
getFileSystemView();

        File[] f = File.listRoots();
        for (int i = 0; i < f.length; i++)
        {
            if (!fileSystemView.getSystemDisplayName(f[i]).
equals("")){
                storages.add(fileSystemView.
getSystemDisplayName(f[i])); //Disk name and Disk Letter
                storageLetters.add(String.valueOf(f[i])); //
Disk Letter
            }
        }
    }
}
```

```
Manifest-Version: 1.0
Main-Class: dlsu.Main
```