

**INDIVIDUAL PROJECT 2**

**SEMESTER 2 2019/2020**

|  |  |
| --- | --- |
|  |  |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**SEKSYEN           : 1**

**LECTURER   : DR. SAMRY@MOHD SHAMRIE SAININ**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Students name** | **Matrics No.** | **Phone No.** |
| 1 | Firman Bin Ridwan | BI19110024 | 012 810 2950 |

**1.0 Java Code**

This project was consist of 12 java class which is 1 Main class (MyApps.java), 1 Interface class (Master.java), 6 class consist of different modules (Login.java, Profile.java, Patient.java, Date.java, Prescription.java and Ward.java), and 4 java GUI classes (LoginPanel.java, ProfilePanel.java, PetPanel.java, and LastPanel.java).

All the java file was submitted via GitHub link as below:

<https://github.com/Manmansui/OOP-KK14203_FIRMAN_RIDWAN/tree/master/Project/Project2>

**2.0 Object-Oriented Programming Implementation**

The Object-Oriented Programming concept that use in this project was as below:

* 1. **Classes and object.**

All the modules in this project have their own classes and instances of it. For example, Patient.java have its own class and the object of it can be created outside the class to inherit the data as below:

class Patient{

}

class Prescription {

}

class Ward {

}

class PetPanel extends JPanel {   
 Profile a;  
 Patient b;  
 Prescription c;  
 Date d;  
 Ward ez;

. . . . .

}

The object of the Patient, Prescription, and Ward was created and act as instance inside PetPanel class.

* 1. **Encapsulation.**

By taking the Profile.java from the modules as examples, The Profile class below is fully encapsulated. It has three private fields and each of them has its own set of getter and setter methods.

class Profile{ //1st CLASS  
 private String name, ic, address, age;  
 void setName(String name){  
 this.name = name;  
 }  
 void setIc(String ic){  
 this.ic = ic;  
 }  
 String getName(){  
 return name;  
 }  
 String getIc(){  
 return ic;  
 }  
 String getAddress(){  
 return address;  
 }  
 String getAge(){  
 return age;  
 }   
}

The ProfilePanel class below set value for each field with the setter method, While the LastPanel class get the value with the getter method from Profile class as below:

class ProfilePanel extends JPanel{

Profile a;

public ProfilePanel(JFrame frame) {

a = new Profile();

. . . . . . . . . .

a.setName(J\_Name.getText());   
a.setIc(J\_IC.getText());   
a.setAddress(Jaddress.getText());   
a.setAge(J\_Age.getText());

. . . .

}

class LastPanel extends JPanel{

Profile a;

public LastPanel(Profile a, . . ){

this.a = a;

. . . . . . .

Result= "\nOwner: " + a.getName() +  
 "\nI/C: " + a.getIc() +  
 "\nAge: " + a.getAge() +  
 "\nAddress " + a.getAddress();

. . . . .

}

}

* 1. **Inheritance**

By taking the GUI classes as example, all the class inherit the properties of javax swing where the class act as the java swing itself using the “extends” keyword. For example:

class LoginPanel extends JPanel {

. . . . .

class ProfilePanel extends JPanel {

. . . .

class PetPanel extends JPanel {

. . . . .

class LastPanel extends JPanel{

. . . .

Some methods in java swing can be use inside all these classes as below:

class ProfilePanel extends JPanel {

//adjust size and set layout  
 setPreferredSize (new Dimension (520, 291));  
 setLayout (null);  
  
 //add components  
 add (header);  
 add (jtf\_user);

. . . . . . . . . .

}

* 1. **Polymorphism**

Polymorphism is the ability of an object to take on many forms. In this project, the use of static polymorphism was implemented as below:

public PetPanel(JFrame frame, Profile a){

. . . . .

}

public PetPanel(JFrame frame, Profile a, Patient b,

Prescription c, Date d, Ward ez){

. . . . .

}

The above example was two constructors from PetPanel.java where known as constructor overloading. Both constructors have same name but take different parameters.

* 1. **Interface**

Example below shows the interface implementation, where all the abstract method were override and define inside the Login class.

interface Master{

public boolean testLogin(String user, String pass);  
 public void failedTest();  
 public int test();  
}

class Login implements Master{

public boolean testLogin(String user, String pass);{

boolean test = false

. . . . . .

return false;

}

public void failedTest(){  
 trial--;

}  
public int test(){  
 return trial;  
}

}

**3.0 Read and write implementation**

There were three part in this project where read and write procedure being implemented.

* 1. The first part is at the LoginPanel class where user enter their username & password as below:

Class login implements Master{

. . . . . .

public boolean testLogin(String user, String pass){   
 boolean test = false;  
 try{  
 FileReader fr = new FileReader(filename);  
 BufferedReader br = new BufferedReader(fr);  
 String line;  
 while ((line = br.readLine()) != null) {  
 data = line.split(" ");  
 }  
 for(int i=0; i<data.length; i++){  
 if(user.equals(data[i]) && pass.equals(data[i+1])) {   
 test = true;  
 break;  
 }   
 }   
 }catch(Exception e){  
 System.out.println("Error: " + e.toString());  
 }  
 return test;   
 }

}

Class LoginPanel{

. . . . . . . .

status = log.testLogin(jtf\_user.getText(), jpf\_pass.getText());

. . . . . . . .

if(status){  
 System.out.println("test");  
 frame.getContentPane().removeAll();  
 frame.getContentPane().add (new ProfilePanel(frame));   
 frame.pack();  
 frame.setVisible (true);  
}

}

This system will read the “password.txt” file and search the entered username along with it corresponding password. If username and password found in the file and matches, the frame will be cleared, user will be able to proceed and otherwise.

* 1. View the data on selected filename. The code below will read a file and present it on the information panel so that user can see the data.

public void readFile(){  
 BufferedReader reader;  
 try {  
 reader = new BufferedReader(new FileReader(filename));  
 String line = reader.readLine();  
 String output="";  
 while (line != null) {  
 output += line + "\n";  
 // read next line  
 line = reader.readLine();  
 }  
 output += "\n";  
 info.setText(output);  
 reader.close();  
 } catch (IOException io) {  
 info.setText(io.toString());  
 }  
 }

* 1. Add the data to a selected filename directory. The method below is from LastPanel class as below:

public void writeInput(){  
 File file = new File(filename);  
 FileWriter fr = null;  
 BufferedWriter br = null;  
 PrintWriter pr = null;  
 try {  
 fr = new FileWriter(file, true);  
 br = new BufferedWriter(fr);  
 pr = new PrintWriter(br);  
 pr.println(finals);  
 printResult.setText("Successfully printing data to " + filename);  
 } catch (IOException e) {   
 printResult.setText(e.toString());  
 } finally {  
 try {  
 pr.close();  
 br.close();  
 fr.close();  
 } catch (IOException e) {  
 printResult.setText("Error! Missing file");  
 } catch(NullPointerException f){  
 printResult.setText("Error! filename cannot empty");  
 }  
 }  
 }

The combination of FileWriter, BufferedWritter, and printWritter can be used to create a file and append the data into file instead of deleting the initial data in the file if the file already exists.

**User Manual (How to use the system):**

First and foremost, the system run from the main file which is MyApps.java file. All informative

If the user is a new user, the user needs to add user’s username and password by clicking the Register item on the top right of the login panel as below.

A screenshot of a social media post

Description automatically generated

Figure 1: The Login Panel

User need to enter Administrator code to successfully register as below. User then need to click submit to add the new username and password to “password.txt” and a popup will show for successfully registered or else registration failed. The username and password were added and arranged side by side all in one line in the file.

A screenshot of a cell phone

Description automatically generated

Figure 2: The Register Panel

After that, the user which is the doctor in this case, need to submit the username and corresponding password on the login panel. If the username and password match and found in the “password.txt” file, user will be able to enter the system.

After successfully login, user will be lead to next panel which is for collecting pet’s owner information such as name, IC number, age, and address. User need to fill all the field or else not be able to submit and proceed. This panel provide reset button for resetting all the information as below:

A screenshot of a cell phone

Description automatically generated

Figure 3: The Profile Panel

Next, user will see a bigger panel consisting many information about the pet that need to be collected. User need to enter appropriate information about the pet and submit it to proceed. Same as before, user need to fill all the field or else not be able to submit and proceed. Reset button also provided in this panel for resetting all the data.

A screenshot of a social media post

Description automatically generated

Figure 4: The Pet Panel

Lastly, after submitting user will see all the information entered on the new panel. If the user wanted to edit the information, user just need to press the edit button on bottom left side to go back to previous panel. There was also a print button where user can click it to print the data to selected file name. The filename was initialized to “data.txt”. Upon successfully saving the data, there was a message on bottom right corner to show the successfully task. If the owner come with many pets, user will first need to print the data for first pet. After successfully saving the data, user can click the edit button for the second pet information and so on.

There was also a menu bar at the top left of the panel where user have three option which is new owner, view data, and change filename. The new owner option will create a new pet’s owner profile. The view data option will show the data contained in the selected filename on information panel. The change filename option will change the directory of the file path, with this user can view and save data on the selected file. All above information refer to figure below:

A screenshot of a cell phone

Description automatically generated

Figure 5: The Last Panel

A screenshot of a computer

Description automatically generated

Figure 6: The data saved on data.txt file