



# **Group Assignment**

**TECHNOLOGY PARK MALAYSIA**

**CT038-3-2-ODJ**

**OBJECT ORIENTED DEVELOPMENT WITH JAVA**

**APU2F2109SE/ APD2F2109SE**

**HAND OUT DATE: 6 October 2021**

**HAND IN DATE: 6 December 2021**

**Group Member:**

**Brian Joseph Keyrupan      TP060173**

**Elden Zheng                      TP058070**

## Table of Contents

<b>Introduction</b>	4
<b>Design</b>	5
<b>Class Diagram</b>	5
<b>Use case Diagram</b>	6
<b>Use case specification</b>	7
Modify participant	7
Search participant	7
Register participant	7
View participant	8
Search Appointment	8
Add Appointment	8
Remove Appointment	9
Modify Appointment	9
View Appointment	9
View Supply	10
Modify Supply	10
Add Supply	10
Search Supply	11
Remove Supply	11
Login	12
Logout	12
Register	12
View Account	13
Participant View Appointment	13
Participant Cancel Appointment	13
Participant Register Appointment	14
Participant Search Available Vaccination	14
Participant View Available Vaccination	14
<b>Sample Code</b>	15
Variable	15
Control structure	16
If	16

---

If else.....	16
Nested if else.....	17
Looping structure .....	18
While Loop .....	18
For Loop.....	19
Object-Oriented code .....	20
Class.....	20
Object.....	20
Method .....	21
Constructor.....	22
Overloading.....	22
Encapsulation.....	23
Inheritance.....	24
Exception handling .....	24
Array .....	25
ArrayList .....	25
File Concept.....	26
Write data.....	26
Read data.....	27
Search data .....	27
Modify data.....	28
<b>Additional features.....</b>	<b>29</b>
Lambda .....	29
Jtable .....	30
SetLocation .....	31
LocalDate.....	31
<b>Sample output.....</b>	<b>32</b>
Login Form .....	32
Register Form.....	33
Participant Home Form.....	34
Participant Profile Form.....	35
Submit Appointment Form .....	36
Available Vaccine Form .....	37

---

---

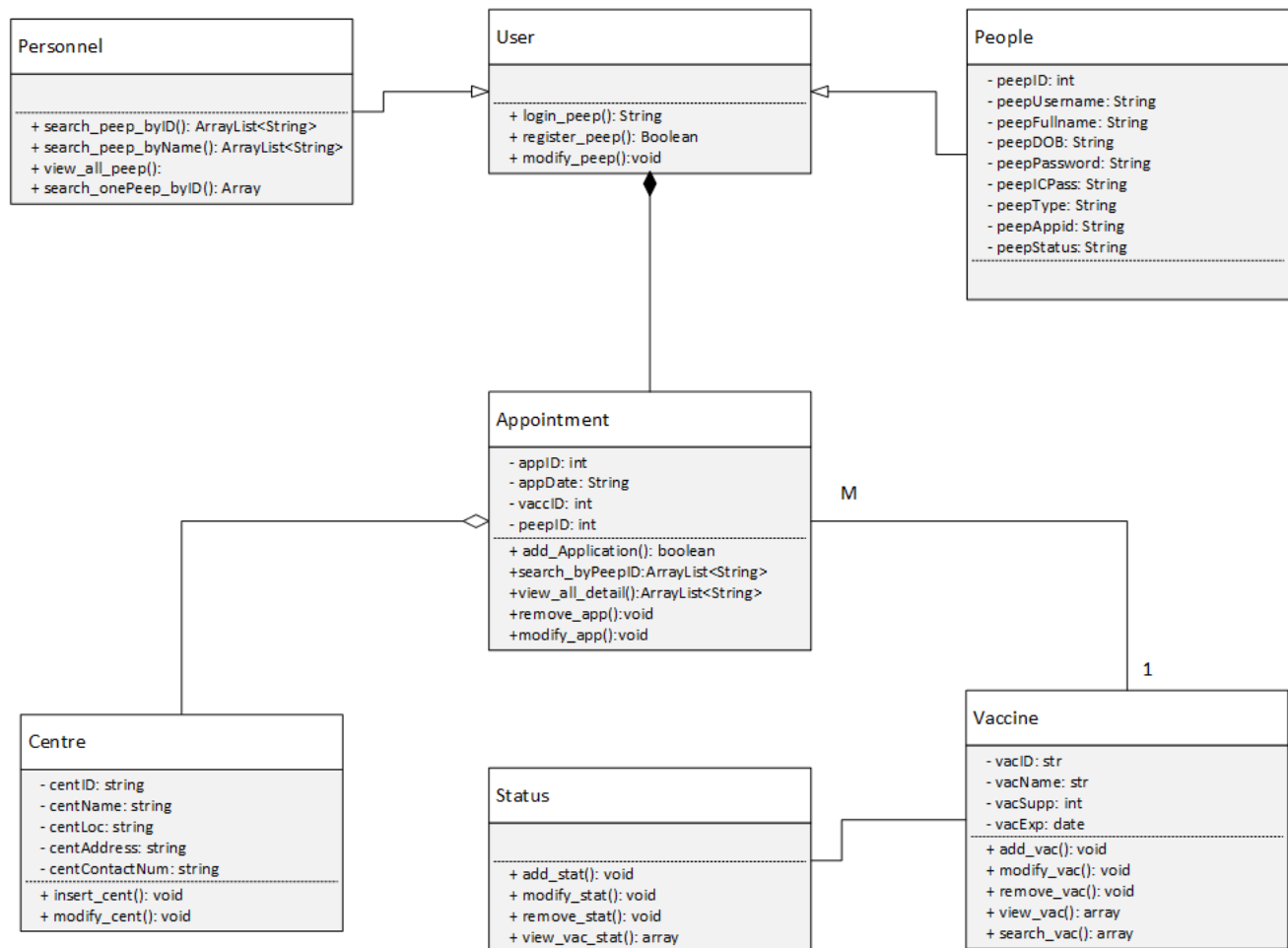
Personnel Home Form .....	38
View Participant Form .....	39
Register New Participant Form.....	40
Appointment Form.....	41
Supply Form.....	42
<b>Conclusion</b> .....	43
<b>References</b> .....	45

## Introduction

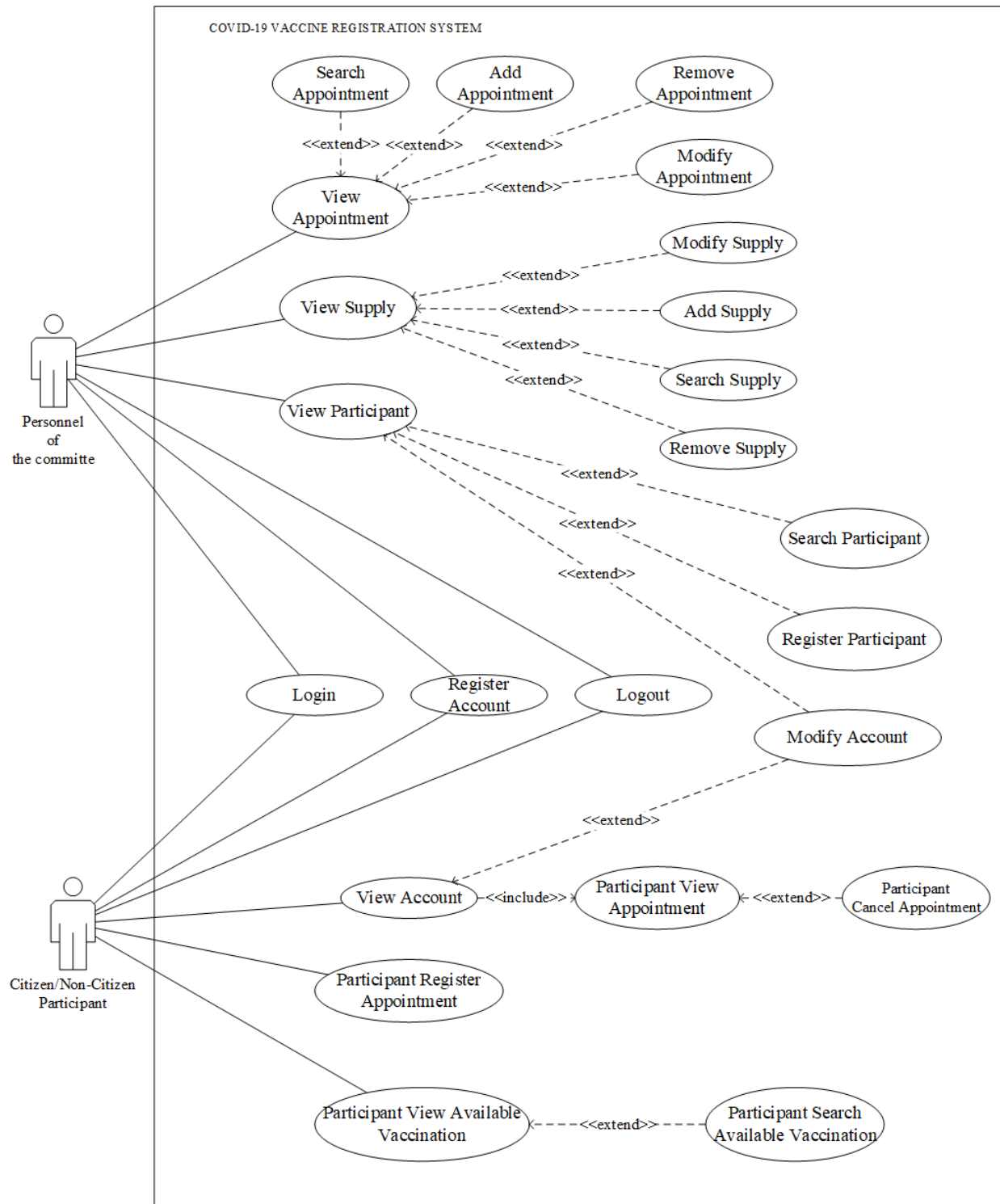
The following document consist of the documentation of the COVID-19 Vaccine supply and appointment in an object-oriented system. The system shall be in the form of a windows application. The system will be used by personnel and participants. System will be able to identify the user using the login function created, this will allow access rights according to the user type. Personnel of the system will be able to manage the participants, manage the vaccination appointments, and manage the supply of the COVID-19 vaccines in the centers. While the participants can be either citizen or non-citizen which both will be able to create an account in the system, make an appointment for a vaccination, and view the vaccine status of where and what type of vaccine is available. System will be able to run the function without any issues occurring.

## Design

### Class Diagram



## Use case Diagram



## Use case specification

### Modify participant

Use Case	Modify Participant
Brief Description	Personnel will be able to modify the data of the participant
Actors	Personnel
Preconditions	The Personnel choose which participant they want to modify by choosing the participant's data from the table
Main Flow	(a) This use case begins when the personnel clicked one of the participants from the table (b) The participant's data will be filled in the panel on the right (c) The personnel can change the data accordingly in the panel on the right (d) After the modification is done and the modify data is pressed the data will be updated in the text file
Alternative Flow	(d)(I) The modified data is the same as the old data, then the personnel cannot modify the data

### Search participant

Use Case	Search Participant
Brief Description	Personnel can search for a specific participant by inputting their ID
Actors	Personnel
Preconditions	The Personnel wants to find specific people
Main Flow	(a) The personnel input the ID of the participant they want to find (b) The system will put their details on the table
Alternative Flow	(a) (I) the participant ID does not exist there will be a pop-up window stating that the ID does not exist (II) the input ID is not a number there will be a pop-up window stating that the format of the ID is incorrect

### Register participant

Use Case	Register Participant
Brief Description	Personnel can register a participant manually
Actors	Personnel
Preconditions	There is a participant who went to ask the personnel for help
Main Flow	(a) The Personnel input the participant's data (b) The system checks if all the necessary information is filled in or not (c) The data will be updated in the file
Alternative Flow	(b) If one of the information is not filled out, a popout box will state that the personnel need to input the necessary data



## View participant

Use Case	View Participant
Brief Description	Personnel will be able to see the all the participant along with their details
Actors	Personnel
Preconditions	The personnel have already logged in as an admin
Main Flow	(a) The system inserts all of the participant's information into the table.
Alternative Flow	(a) (I) if there is no participant there will be a pop-out window stating there are no participant found

## Search Appointment

Use Case	Search Appointment
Brief Description	Personnel will be able to see the appointment according to the participant
Actors	Personnel
Preconditions	The personnel want to find a specific participant appointment
Main Flow	(a) The personnel input the ID of the participant's appointment they want to find (b) The system will put their details in the table
Alternative Flow	(a) (I) the participant has not booked a meeting yet, there will be a pop-out window stating ID not found

## Add Appointment

Use Case	Add Appointment
Brief Description	Personnel will be able to add an appointment to a specific participant
Actors	Personnel
Preconditions	The personnel have chosen a participant to add the appointment to
Main Flow	(a) The personnel input all the necessary details, such as vaccine name, vaccine location and the date of the appointment (b) The system will save the details in the file (c) The system will save the appointment ID to the participant file accordingly (d) The system will remove 1 supply to the chosen vaccine name and location
Alternative Flow	(a) (I) the vaccine supply is 0, so there will be a pop-out window stating the supply is empty please rechoose vaccine and the location

## Remove Appointment

Use Case	Remove Appointment
Brief Description	Personnel will be able to delete the appointment
Actors	Personnel
Preconditions	The personnel have an appointment they want to remove
Main Flow	(a) The personnel will pick they want to remove from the table (b) The personnel clicked the remove button (c) The system will delete that specific row from the file (d) The system will add 1 vaccine supply accordingly (e) The system will remove the appointment ID from the participant file
Alternative Flow	(a) (I) there is no appointment, the table will be empty, and a pop-up box will appear and state that there are no appointments.

## Modify Appointment

Use Case	Modify Appointment
Brief Description	Personnel can modify the appointment of the participants
Actors	Personnel
Preconditions	Personnel have chosen which appointment they want to modify
Main Flow	(a) The personnel choose will choose the appointment (b) The appointment's data will be filled in the panel on the right (c) The personnel can modify the date of the appointment in the given combo box (d) After the modify button has been pressed the modified data will be saved accordingly in the file
Alternative Flow	(d)(I) The modified data is the same as the old data, then the personnel cannot modify the data

## View Appointment

Use Case	View Appointment
Brief Description	Personnel can view the booked appointment from all the participants
Actors	Personnel
Preconditions	The personnel already logged in as admin
Main Flow	(a) The system inserts all of the appointments into the table.
Alternative Flow	(a) (I) if there is no appointment there will be a pop-out window stating there is no appointment found

### View Supply

Use Case	View Supply
Brief Description	Personnel can view all the supply from all the location
Actors	Personnel
Preconditions	The personnel already logged in as admin
Main Flow	(a) The system inserts the vaccine name, center location, and number of supply in the table.
Alternative Flow	(a) (I) if the vaccine from the location does not have any supply it will not be shown in the table (II) if there is no vaccine there will be a pop-out window stating there is no appointment found

### Modify Supply

Use Case	Modify Supply
Brief Description	Personnel can modify the number of supplies from the specific vaccine type and location
Actors	Personnel
Preconditions	The personnel have an incorrect input of the supply
Main Flow	(a) The personnel choses from the table which vaccine and location needs to be modified (b) The data details will be shown in the panel on the right (c) The personnel can change the number of supplies (d) The updated data will be saved to the file
Alternative Flow	(c)(I) if the input supply is not a number the system will show a pop-up window saying invalid input

### Add Supply

Use Case	Add Supply
Brief Description	Personnel will be able to add the supplies to the specific vaccine type and location
Actors	Personnel
Preconditions	The personnel need to specify the vaccine type and center location
Main Flow	(a) The personnel input vaccine type and center location using the combo box (b) The personnel input the number of supplies (c) The supplies input will be added with the old supplies data (d) The data will be saved to the file
Alternative Flow	(c)(I) if the input supply is not a number the system will show a pop-up window saying invalid input

## Search Supply

Use Case	Search Supply
Brief Description	Personnel will be able to search the existing supply by its ID, vaccine name, or by the location.
Actors	Personnel
Preconditions	Personnel wants to search for supply details.
Main Flow	<p>(a) The use case begins when the personnel reach the vaccine supply form.</p> <p>(b) System will display all the available vaccine supply detail on the table in the form.</p> <p>(c) They can search by ID by typing in the text box.</p> <p>(d) They can search by the vaccine name with the dedicated combo box.</p> <p>(e) They can search by the location with the location dedicated combo box.</p> <p>(f) System will search the vaccine supply according to the details given and display all the supply in the table on the form.</p>
Alternative Flow	(b)(I) If there is no supply available, then the system will display nothing to the table of the form.

## Remove Supply

Use Case	Remove Supply
Brief Description	Personnel will be able to remove the supply according to the vaccine supply detail
Actors	Personnel
Preconditions	Personnel wants the supply in the location and vaccine name is to be removed in the system because it is no more.
Main Flow	<p>(a) This use case is when the supply of a vaccine in a location wants to be removed.</p> <p>(b) After selecting the vaccine and location detail, personnel can remove it, and the supply of that vaccine in that location is no more in the system.</p>
Alternative Flow	(b)(I) If the Supply is not selected, then supply will not be able to be removed

## Login

Use Case	Login
Brief Description	This case will ask the user to enter a username and password and will be able to identify if the user is personnel or a citizen or a non-citizen, then log into the according account.
Actors	Personnel, Citizen/Non-Citizen
Preconditions	When the user wants to access the system, appointment functions and needs to register
Main Flow	(a) After inputting the username and password of the user, the system will be able to identify if the user is personnel, citizen, or non-citizen. (b) Correct username and password will direct the application to the home form accordingly.
Alternative Flow	(b)(I) Incorrect username and password will not allow any user to go into use the functions

## Logout

Use Case	Logout
Brief Description	This case logs the user out of the system.
Actors	Personnel, Citizen/Non-Citizen
Preconditions	Users must be logged into the system.
Main Flow	(a) After using the system's functions the users can log out of the system, and it brings back to the login form.
Alternative Flow	-

## Register

Use Case	Search Supply
Brief Description	Allows users to create an account by inputting their data in order to use the system.
Actors	Citizen/Non-Citizen
Preconditions	User wants to make an account to use the vaccination appointment system.
Main Flow	(a) System will receive the data from the user, then system will check if all the data required is filled and then system will insert the data into the data storage of the system.
Alternative Flow	(a)(I) When the input data is empty, password and the confirm password different or data inputted not accordingly then registration will fail and shows a pop-up box

## View Account

Use Case	View Account
Brief Description	Users will be able to see their own data.
Actors	Citizen/Non-Citizen
Preconditions	Requires users to log into their account, they are only able to see their own data and only then can they reach this function.
Main Flow	(a) System will show their data in the computer application form and allow them to see their personal data and appointment ID along with the data to determine if they are vaccinated or not.
Alternative Flow	-

## Participant View Appointment

Use Case	Participant View Appointment
Brief Description	The participants can only see their own appointment.
Actors	Citizen/Non-Citizen (Participant)
Preconditions	Must already have made an appointment.
Main Flow	(a) Here the system will show the participant's appointment. If the user has not made any appointment.
Alternative Flow	(a)(i) If the user has no appointment then the system will show none on the GUI.

## Participant Cancel Appointment

Use Case	Participant Cancel Appointment
Brief Description	Allows users to cancel the appointment they made,
Actors	Citizen/Non-Citizen (Participant)
Preconditions	Must already make an appointment in the system.
Main Flow	(a) Patients can cancel the appointment they set, or the personnel made. (b) This will allow them to make a new appointment or just cancel the appointment.
Alternative Flow	(a)(I) If there is no appointment then there is nothing to cancel and system will not allow user to cancel.

**Participant Register Appointment**

Use Case	Participant Register Appointment
Brief Description	The participant will be able to register their appointment
Actors	Citizen/Non-Citizen (Participant)
Preconditions	They will be able to register for an appointment only if they don't have an appointment
Main Flow	(a) They will enter the details on the appointment whether it is the date, selection of the vaccine, and the selection of the location. (b) After inserting the data in the system, then system will update their data storage to add or update the appointment in the text files.
Alternative Flow	-

**Participant Search Available Vaccination**

Use Case	Participant Search Vaccination Availability
Brief Description	Participants will be able to see the list of where and what vaccine is available to make appointments on.
Actors	Citizen/Non-Citizen (Participant)
Preconditions	Must be logged into the system.
Main Flow	(a) Citizen/Non-Citizen (Participant) can only search the available vaccination detail (b) They can search by the name of the vaccine, name of location, or the id of the available vaccination detail (c) After searching the detail then system will show the data in the table.
Alternative Flow	(c)(I) When searching if the data is not found then it will show none

**Participant View Available Vaccination**

Use Case	Participant View Vaccination Availability
Brief Description	The participant can search the available vaccine and location by the vaccine, location, or the id.
Actors	Citizen/Non-Citizen (Participant)
Preconditions	Only registered user can see the available vaccines.
Main Flow	(a) The registered user will be able to see all the available vaccination detail from the name of the vaccine, the location of the vaccine and the id of the vaccine.
Alternative Flow	(a)(I) The user will not be able to see any of the available vaccines if there are no available supply details.

## Sample Code

### Variable

Variables are like boxes or containers to store several types of data values. Some types of variables in Java include String, int, float, char, and Boolean. The ones in the following are the type of variables used to create the system.

```
private int peepID;  
private String peepUsername;  
private String peepFullname;  
private String peepDOB;  
private String peepPassword;  
private String peepICPass;  
private String peepType;  
private String peepAppid;  
private String peepStatus;
```

Strings are used to store text such as the following and have double quotes before and after the text.

```
private String peepUsername;
```

In the assignment string is used to store variable with words like name or password.

Int is short for integer, which stores numbers that are not decimals

```
private int peepID;
```

In the assignment integer is used to store variable with numbers that are not decimals, such as ID or supply.



Boolean stores only 2 values which are either true or false.

```
boolean noduplicate=true;
```

In the assignment boolean is used to store variable that has a specific condition, like we use it to make sure if the username already exist or not, if it exists then nonduplicate will be false and if it does not exist then nonduplicate will be true.

### Control structure

In simple terms, it means that it is a programming block that is utilized to control the flow of the program with instruction or condition. (Baeldung, 2019)

#### If

The if statement only executes the code inside it if the condition is true. (JavaTPoint, 2021b)

```
if(! (user.getPeepAppid().equals("none"))){  
    btnAppoint.setEnabled(false);  
    btnCancel.setEnabled(true);  
}
```

The code above checks if the variable “PeepAppid” is not none, if the variable “PeepAppid” is not none then it executes the code below. Which is to set the appointment button to be un interactable and set the cancel button to be interactable

#### If else

The if else statement is similar to the if statement but have more layers, after the if statement it will have the else if statement. The else if statement basically after checking if the if statement is false then it will go down and check if the else if condition is true or not, if its true then it will execute the code inside it, the code can have multiple else if statement. If its false then it will go down to the else statement. The else statement executes when all the if and else if statement is false.

```
People log = new People(Username,Password);
if (log.login_peep()=="admin"){ //enters admin information
    this.dispose();
    new PersonnelHome_jframe().setVisible(true);
}
else if(log.login_peep()=="false") //incorrect information
{
    JOptionPane.showMessageDialog(null,"Invalid Information, Please re-enter");
}
else{ //enters user information
    this.dispose();
    //open partGUI
    new ParticipantHome_jframe(Integer.parseInt(log.login_peep())).setVisible(true); //pass the participant id
}
```

The code above checks if the function returns the string “admin” if it's true then it launches the personnel home jframe, if its false then it checks below, the else if statement check if the function returns “false”, if it's true then a pop-out window will state that is invalid information, if its false it will go to the else statement. The else statement will execute if all the if and else if statement is false and will launch the participant home jframe.

#### Nested if else

The nested if else statement is basically an if statement with another if else statement inside of it, the if statement is checked when the outer if statement is true. (JavaTPoint, 2021b)

```
if(app.add_Application(app_date, vac_name, vac_location, id)==true){ //if code is successfull returns user
    if(user.equals("admin")){ //if the user is admin return to admin
        new PersonnelViewParticipant_jframe().setVisible(true);
        this.dispose();
    }else{ //if the user is participant return to participant
        new ParticipantHome_jframe(id).setVisible(true);
        this.dispose();
    }
}
else{ //if booking fails clear the data
    cmbVaccTemp.setSelectedIndex(0);
    cmbLocationTemp.setSelectedIndex(0);
    cmbDate.setSelectedIndex(0);
    cmbYear.setSelectedIndex(0);
    cmbMonth.setSelectedIndex(0);
}
```

The code above checks if the method runs successfully then it goes inside and check another if statement if the user is “admin” then it launches the view participant jframe, and if its false then it went to the else statement which launches the participant home jframe. If the method from the first if is not successful, then it will set all of the combo box to their original state.

## Looping structure

Loops structures are a block of code that will be executed when the specified condition(s) is/are fulfilled. The block of code will be executed over and over if the condition remains true, it will stop if and only if the condition returns false, only then will the code run out of the loop block. (W3School, 2021g)

### While Loop

While loops will run the code if the condition is true and will only stop if the condition is false. The following is one example of a while loop used in the code.

```
boolean noduplicate=true;
String line;
BufferedReader file = new BufferedReader(new FileReader("People.txt"));
while ((line=file.readLine())!= null){
    String [] data = line.split(",");
    if(peepUsername.equals(data[1])){
        JOptionPane.showMessageDialog(null,"Username already taken, Please re-enter username");
        noduplicate=false;
        break;
    }
}
```

The snippet shows a while loop used to check if the selected username exists in the text file. The while loop checks each time the BufferedReader object to call the readLine() function and if it returns not null then the while loop block will be executed. Then once the condition is null then it stops. The while loop block of code will check if the variable peepUsername is equals to the first variable of the array of the text file, and if the condition of the if is true then it will show a pop-up form and set the noduplicate boolean variable is false, then break from the while loop.

## For Loop

For loops are loop structures that you use when you know how many times you want to loop the body of the statement of the for a loop. There are 2 types of for loop used in the creation of the system, which are normal for loops and foreach loop. For-each loop loops through an Array or an ArrayList. (W3School, 2021g)

```
int Month=cmbMonth.getSelectedIndex();  
int[] one = { 0, 2, 4, 6, 7, 9, 11};  
int[] thirteen = { 3, 5, 8, 10};  
boolean foundOne = false;  
boolean foundThre=false;  
for (int n : one) {  
    if (n == Month) {  
        foundOne = true;  
        break;  
    }  
}  
for (int n : thirteen) {  
    if (n == Month) {  
        foundThre = true;  
        break;  
    }  
}
```

This for loop checks if the variable Month, which is the selected index of a combo box, is in the Array one or Array thirteen. So, the first for loop the condition is to loop each variable n in the Array one. Then with the if statement the condition is if the n is equal to the Month variable, then it sets the foundOne boolean to true. The second for loop is the same as the first for loop, difference is that it loops the integer Array thirteen and sets the boolean variable foundThre to true when the Month is in the Array.

## Object-Oriented code

Java is an object-oriented programming language, which means that everything is related to classes and objects, also with its attributes and its methods.

## Class

A class is a constructor of an object, it is a “blueprint” for creating objects (W3School, 2021a). A class contains the attributes and the methods of an object.

```
public class People extends User{  
    private int peepID;  
    private String peepUsername;  
    private String peepFullname;  
    private String peepDOB;  
    private String peepPassword;  
    private String peepICPass;  
    private String peepType;  
    private String peepAppid;  
    private String peepStatus;
```

To create a class, use the keyword class. In the above example public is the access modifier is public and class along with the name of the class. The name of the class in the snippet is People, extends to User is an inheritance, meaning this class is a subclass of User class. The variables declared are the attributes of People class. Which are used for the methods in it.

## Object

An object is a member of a java class, an object is used to call a class. Inside of the object there is a parameter which can be used to determine the constructor. (Techopedia, 2021)

```
Vaccine vac = new Vaccine(selected_data.get(0));
```

The example code above is a class being declared using an object, first type in the class then the object then followed by an equal sign after that the keyword “new” and the name of the class again along with the respective parameter.

## Method

A method is a block of code that will run when the method is called. Data can be passed into the method and are called parameters. So, method can be created with or without a parameter depending on how it is used and made. To create a method in java, the essential things needed would be to identify the access modifier of the method, the return value of the method, and lastly the name of the method. (W3School, 2021f)

```
public String return_string() {  
    String vac_data = VacID+", "+  
                      VacName+", "+  
                      VacLoc+", "+  
                      VacSupp;  
    return vac_data;  
}
```

The snippet is an example of a method. This method will return String as typed on the first line after stating the access modifier, then the name which in this case is return\_string(). Inside the method is what will be executed when the code is called. Then as stated what the return type of the method, this method will return a String vac\_data. To call this method in another method or class, just simply state return\_string().

### Constructor

A constructor is called when an object is instantiated, then it will check for the matching parameters. (Jenkov, 2021)

```
public People(String peepUsername, String peepPassword) {  
    this.peepUsername = peepUsername;  
    this.peepPassword = peepPassword;  
}
```

The example code above is a constructor of the class “people” with 2 string as its parameter then assign the input with the respective variable.

### Overloading

Constructor overloading basically means having multiple constructors with different parameters whether it's the amount or type. (*Removing repetitive code (overloading methods and constructors)* - Java Programming, 2021)

```
new ParticipantHome_jframe(Integer.parseInt(log.login_peep())) .setVisible(true);
```

```
public ParticipantHome_jframe() { //if run file  
    System.out.println("Please run Project");  
}  
public ParticipantHome_jframe(int Id) { //if run project  
    initComponents();  
  
    Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();  
    this.setLocation(dim.width/2-this.getSize().width/2, dim.height/2-this.getSize().height/2);  
    setID(Id);  
}
```

The two code examples on top put an integer as a parameter, then when the jframe launches it actually have 2 constructors. When it is passed an integer from the previous jframe it passes the value from the old frame to the new one.

## Encapsulation

Encapsulation is to make data encapsulated or hidden from the users and the implementation of encapsulation is using get set methods. By declaring private variables in the class which are the attributes, the variables are only accessed within the class, and to make it public is by using public get and set methods. Get methods are used to return the value of the variable from the class, whereas set method is to change the value of the variable. In short, get is made for read-only and set is for write-only. (W3School, 2021c)

```
public class People extends User{
    private int peepID;
    private String peepUsername;
    private String peepFullname;
    private String peepDOB;
    private String peepPassword;
    private String peepICPass;
    private String peepType;
    private String peepAppid;
    private String peepStatus;

    //-----Get Set Methods-----
    public int getPeepID() {
        return peepID;}
    public void setPeepID(int peepID) {
        this.peepID = peepID;}
    public String getPeepFullname() {
        return peepFullname;}
    public void setPeepFullname(String peepFullname) {
        this.peepFullname = peepFullname;}
    public String getPeepDOB() {
        return peepDOB;}
    public void setPeepDOB(String peepDOB) {
        this.peepDOB = peepDOB;}
    public String getPeepPassword() {
        return peepPassword;}
    public void setPeepPassword(String peepPassword) {
        this.peepPassword = peepPassword;}
    public String getPeepUsername() {
        return peepUsername;}
    public void setPeepUsername(String peepUsername) {
        this.peepUsername = peepUsername;}
    public String getPeepICPass() {
        return peepICPass;}
    public void setPeepICPass(String peepICPass) {
        this.peepICPass = peepICPass;}
    public String getPeepType() {
        return peepType;}
    public void setPeepType(String peepType) {
        this.peepType = peepType;}
    public String getPeepAppid() {
        return peepAppid;}
    public void setPeepAppid(String appid) {
        this.peepAppid = appid;}
    public String getPeepStatus() {
        return peepStatus;}
    public void setPeepStatus(String status) {
        this.peepStatus = status;}
}
```

The snippet code of the class people has attributes in it, and to be able to access those attributes it uses the get and set methods. Each attribute has its own get and set method as shown in the image.



## Inheritance

Inheritance basically passes down the attributes and method of a class to another class. (W3School, 2021d)

```
public class Personnel extends User{
```

This code above passes down the attributes and method from the “User” class to the “Personnel” class.

## Exception handling

Programs or lines of code that are prone to error are put in a try block, if an error occurs in the try block it will be thrown to the catch block according to the exception type. (GeeksforGeeks, 2018)

```
try(FileReader fr = new FileReader(participant_file)){
    //Search for the data changed in the array list
    BufferedReader br = new BufferedReader(fr);
    String data;
    ArrayList<String> peoples_data = new ArrayList<>();
    if((data=br.readLine())!=null){
        peoples_data.add(data);
        while ((data=br.readLine())!=null){
            peoples_data.add(data);
        }
        br.close();
        return peoples_data;
    }
    br.close();
    return null;
} catch (Exception e) {
    System.out.println(e);
}
```

In the sample code above, there is a try block that has a code that is prone to error like the file reader and buffered reader, so it is put in a try block. If the file reader or buffered reader fails, then the program will be thrown into the catch block which have “Exception” as their exception type and executes the program below it.

## Array

Arrays are used for storing multiple values within a set amount within one variable.

```
String [] date = new String[30];
```

The example code above shows an array with “date” as the variable name being declared with string as the data type and 30 is the set number of values of the array.

## ArrayList

An ArrayList is basically a resizable array. It is a class from the java.util package, therefore requires the class using this class to import java.util.ArrayList to use the ArrayList. (W3School, 2021)

```
ArrayList<String> vacArrayList= new ArrayList<>();  
while ((data=br.readLine()) != null){  
    Vaccine vacc = new Vaccine(data);  
    if(vacc.getVacLoc().toLowerCase().contains(vacloc.toLowerCase())){  
        vacArrayList.add(data);  
    }  
}
```

The snippet code above shows that an ArrayList is created on the first line. It creates an ArrayList object that will have a String in the ArrayList and the name of the variable is vacArrayList.

Then in the if block there is vacArrayList.add(data). That is how to add an item into the ArrayList. The parameter of the add must also be a String since the vacArrayList can only contain string.

```
String current_data=(new Personnel().search_peep_byID(person.getPeepID())).get(0);
```

The snippet above is creating a String variable. search\_peep\_byID(person.getPeepID()) will return an ArrayList, this function will accept an int with the get method of the Person object. The ArrayList returned then will have .get(0) function. This .get(0) will return the element of the index 0 of the ArrayList. The .get() function will get the element of the selected index number from the ArrayList.

## File Concept

PrintWriter enables the user to write formatted data (Jenkov, 2021b).

BufferedWriter adds buffering to the writer which makes the performance faster (JavaTPoint, 2021a).

FileWriter is used for writing streams of characters and creates an output file, if not already exist (JavaTPoint, 2021b).

FileReader is used for reading streams of character from a text file (GeeksforGeeks, 2019).

## Write data

```
try (PrintWriter regis = new PrintWriter(new BufferedWriter(new FileWriter("People.txt", true)))) {
    regis.append(peepID+", "+
        peepUsername+", "+
        peepPassword+", "+
        peepFullname+", "+
        peepDOB+", "+
        peepType+", "+
        peepICPass+", "+ "none"+", "+ "Non-Vaccinated"+ "\n");
    regis.close();
    JOptionPane.showMessageDialog(null, "Register Successfully");
    return true;
} catch (IOException ioe) {
    ioe.printStackTrace();
}
```

This method belongs to the people class and it appends the information like peepID, peepUsername, peepPassword, peepFullname, peepDOB, peepType, peepICPass to the people.txt and it is file reader is closed after the append is done.

### Read data

```
public ArrayList<String> view_all_peep(){ //should return the array which contains all the people
    File participant_file = new File("People.txt");
    try(FileReader fr = new FileReader(participant_file)){
        //Search for the data changed in the array list
        BufferedReader br = new BufferedReader(fr);
        String data;
        ArrayList<String> peoples_data = new ArrayList<>();
        if((data=br.readLine())!=null){
            peoples_data.add(data);
            while ((data=br.readLine())!=null){
                peoples_data.add(data);
            }
            br.close();
            return peoples_data;
        }
        br.close();
        return null;
    }catch(Exception e){
        System.out.println(e);
    }
    return null; //means no data
}
```

This method belongs to the personnel class, and its function is to get all the row and store it in inside an arraylist then return the arraylist.

### Search data

```
public ArrayList<String> search_peep_byID(int id){ //will return the full data of that id
    File file = new File("People.txt");
    try{
        String data;
        BufferedReader br=new BufferedReader(new FileReader(file));
        People person;
        while ((data=br.readLine())!= null){
            person = new People(data); //creates the people object
            if(id==person.getPeepID()){ //checks if the id is the one looking for
                ArrayList<String> peep_data= new ArrayList<>(); //returns the searched person
                peep_data.add(data);
                br.close();
                return peep_data;
            }
        }
    } catch (IOException ioe){
        //ioe.printStackTrace();
        System.out.println("Login Failed");
    } return null; //if not found will return null
}
```

The method above belongs to the Personnel Class. This method is to search for data in the text file with the first character of the string is the same as the parameter input in the method. After finding the first character with the same ID the string will be stored in the arraylist and return it outside the method.

## Modify data

```
public void modify_vacc(String new_data){
    //the People object with new data
    Vaccine new_vac= new Vaccine(new_data);
    ArrayList<String> peepArrList = new ArrayList<>();
    File participant_file = new File("Vaccine.txt"); //peopletxtfile
    try{
        try(FileReader fr = new FileReader(participant_file)){
            //Search for the data changed in the array list
            BufferedReader br = new BufferedReader(fr);
            String data;
            Vaccine tempVacc;
            while ((data=br.readLine())!=null){
                tempVacc = new Vaccine(data);
                if (tempVacc.getVacID()==new_vac.getVacID()){
                    //change the data
                    tempVacc.setVacName(new_vac.getVacName());
                    tempVacc.setVacLoc(new_vac.getVacLoc());
                    tempVacc.setVacSupp(new_vac.getVacSupp());
                }
                peepArrList.add(tempVacc.return_string());
            }
            try (PrintWriter pw= new PrintWriter(new FileWriter(participant_file, false))){
                //rewrite the whole textfile with the arraylist
                peepArrList.forEach((String)->{ //Lambda basically short form of function
                    pw.println(String);
                });
                JOptionPane.showMessageDialog(null, "Vaccine Supply Data Updated");
            }catch (IOException ioe){
                JOptionPane.showMessageDialog(null, ioe.getMessage());
            }
        }catch (FileNotFoundException ex){
            JOptionPane.showMessageDialog(null, ex.getMessage());
        }
    } catch (IOException ioe) {
        JOptionPane.showMessageDialog(null, ioe.getMessage());
    }
}
```

This method belongs to vaccine class, and it starts by putting the inputted string as a parameter so that it could be split into the data necessary by the constructor. Then it declares an arraylist. Then it initialized a while loop with an if statement that checks if the old vaccine ID is the same with the new vaccine ID and then if the condition is true then it sets the data according to the new vaccine data. All the data will be stored in the arraylist. After that it uses print writer to write all the arraylist into the file.

## Additional features

### Lambda

Lambda expressions are a block of code that takes in a parameter and executes the block of code. It is like a method, but the difference is that it does not require a name and it implements directly inside a method. To declare a lambda expression, it takes in a parameter then the expression is written after the “->” symbol. (W3School, 2021d)

```
peepArrList.forEach( (String) ->{  
    pw.println(String);  
});
```

As the snippet shows a lambda expression used in the code. The code loops the arraylist without the need for a for loop or a while loop, and for each time it loops the parameter for the lambda is the String and each time it goes into the lambda expression it print write the object and calls the function println the String of the parameter. The String here is the String value of the ArrayList peepArrList. Lambda expression can pass with multiple parameters.

## Jtable

Jtable shows data that is 2 dimensional which have rows and columns and arranges it in a table like spreadsheet. (GeeksforGeeks, 2021b)

AppointmentID	AppointmentDate	SupplyID	PeopleID
3	6/6/2021	11	3
4	31/12/2021	19	1
5	5/3/2024	10	2
6	20/12/2021	16	4
7	3/3/2021	1	5
8	13/4/2021	16	6
9	1/2/2021	14	7
11	29/10/2021	10	8
12	21/6/2021	17	9
13	20/11/2021	8	10

```
private void input_table(ArrayList<String> appArray) { //call this function whenever need to update the table
    //use the arraylist to input table
    DefaultTableModel model = (DefaultTableModel)Table_app.getModel();//The table variable
    model.setRowCount(0);
    for(String line:appArray){
        //insert a data into table
        String[] data_row = line.split(",");
        String[] selected_data={data_row[0],data_row[1],data_row[2],data_row[3]}; //must select the data in each array into the table
        model.addRow(selected_data);
    }
}
```

In the example output screen, the jtable is used to display the AppointmentID, AppointmentDate, SupplyID, PeopleID

In the example code above, there is a DefaultTableModel being used to create 0 rows and 0 columns (Oracle, 2020). After that it sets the row to 0 rows. After that it declared an array selected\_data which is the split version of the input arraylist. Then the array number 0,1,2,3 will be put in the jtable row by addrow function.

### SetLocation

The dimension class holds the height and width in integer (GeeksforGeeks, 2021). Getscreensize basically get the size of the current display and put it in the dimension class. Set location is used to set the location of the jframe.

```
Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();  
this.setLocation(dim.width/2-this.getSize().width/2,dim.height/2-this.getSize().height/2);//make jframe center
```

In the example code above the dimension is used to get the default screen size and holds the height and width to integer, then it's set the location of the jframe depending on the width and height half of the value of the currents screen size.

### LocalDate

Local Date class is from a java.time package to use date and time API. LocalDate represents the date in a format yyyy-MM-dd. (W3School, 2021b)

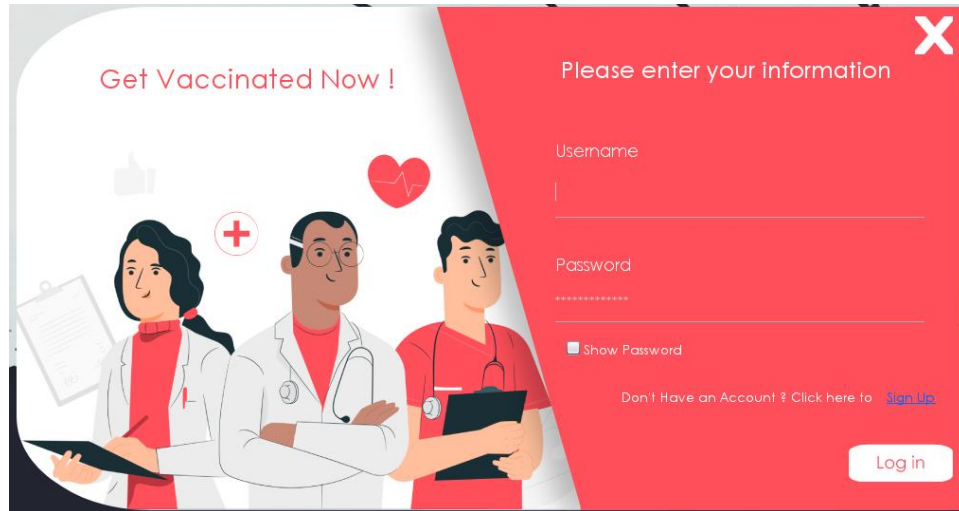
```
LocalDate current_date = LocalDate.now();
```

In the snippet the method now() from the LocalDate class to create a LocalDate object. The now() method returns the current date.



## Sample output

### Login Form



Get Vaccinated Now !

Please enter your information

Username

Password

☐ Show Password

Don't Have an Account ? Click here to [Sign Up](#)

Log in

This is the login form of the system. It accepts the username and password then click the “Log in” button to be able to login. There is also a show password check box to show the password. Then there is a blue “Sign Up” text that will allow users to register their own account. By pressing the text, it will direct the user to the register form. This form uses custom images which is also transparent along with custom buttons that change according to the user’s mouse, this is to make the form more professional looking.

## Register Form

Please enter your information

Fullname

User Name

Date of Birth

1 January 1900

Date Month Year

Are you a:

☒ Citizen

☐ Non-Citizen

Number

Enter Username

Password

\*\*\*\*\*

☐ Show Password

Confirm Password

\*\*\*\*\*

Enter Number

Register

This form will accept the data to register a new user. It accepts the full name, date of birth, a radio button to identify if you are a citizen or a non-citizen, your ic number or passport number accordingly, your username, password and a confirm password. There is also a “Show Password” check box to show the password and confirm password. After inputting all the data then click the Register button to register. This form uses custom images along with custom buttons that change according to the user’s mouse, this is to make the form more professional looking.

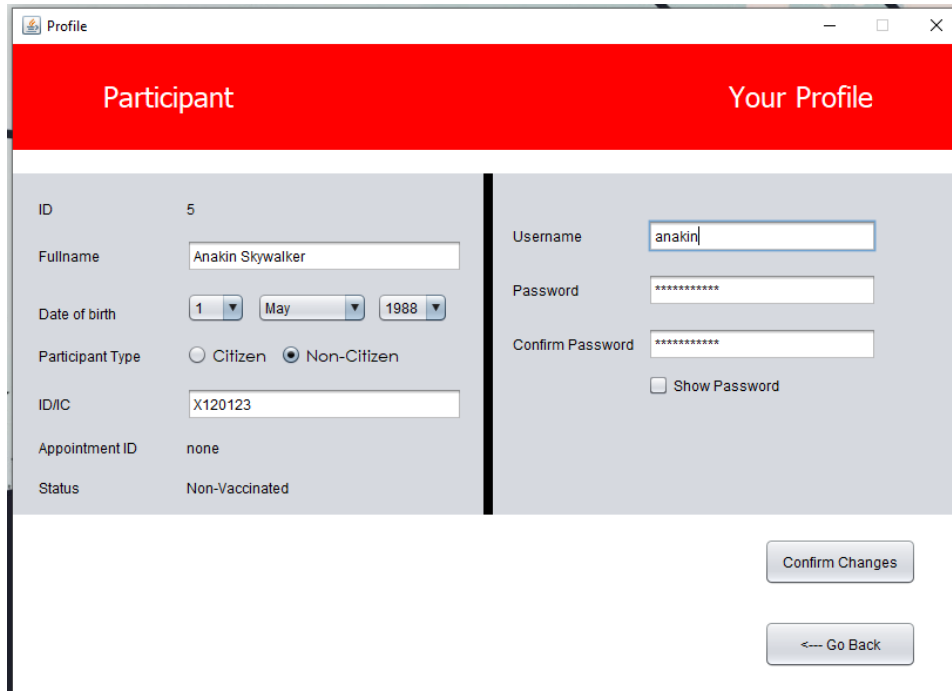
## Participant Home Form

The screenshot shows a web application window titled "Home" with standard window controls (minimize, maximize, close). The main content area has a red header bar with the text "Participant" on the left and "Home" on the right. Below the header, the text "Hi, Elden" is displayed in a large, bold font. To the left of this text is a vertical stack of four buttons: "Profile", "Submit Appointment", "Vaccine Status", and "Log Out". To the right of the text is a grey box titled "Appointment Info" containing the following details: Appointment ID: 7, Appointment Date: 3/3/2021, Vaccine Name: CoronaVac, and Vaccine Location: NorthWood. A "Cancel" button is located at the bottom of this box.

Appointment Info	
Appointment ID	7
Appointment Date	3/3/2021
Vaccine Name	CoronaVac
Vaccine Location	NorthWood

This form acts as a main hub for the participant in here the participant can go to their respective function. If the profile button is pressed, then the participant goes to the participant profile form GUI. If they do have an appointment then the submit appointment button is greyed out and cannot be pressed, and they have the appointment information filled in along with a pressable cancel button. If they pressed the cancel button then the appointment is cancelled, if they press the vaccine button then they will be taken to the view vaccine form GUI. If they press the log out button, then they will be taken to the main login menu and be logged out. Lastly the GUI have a label that says “hi, and their name”, to make it more personal

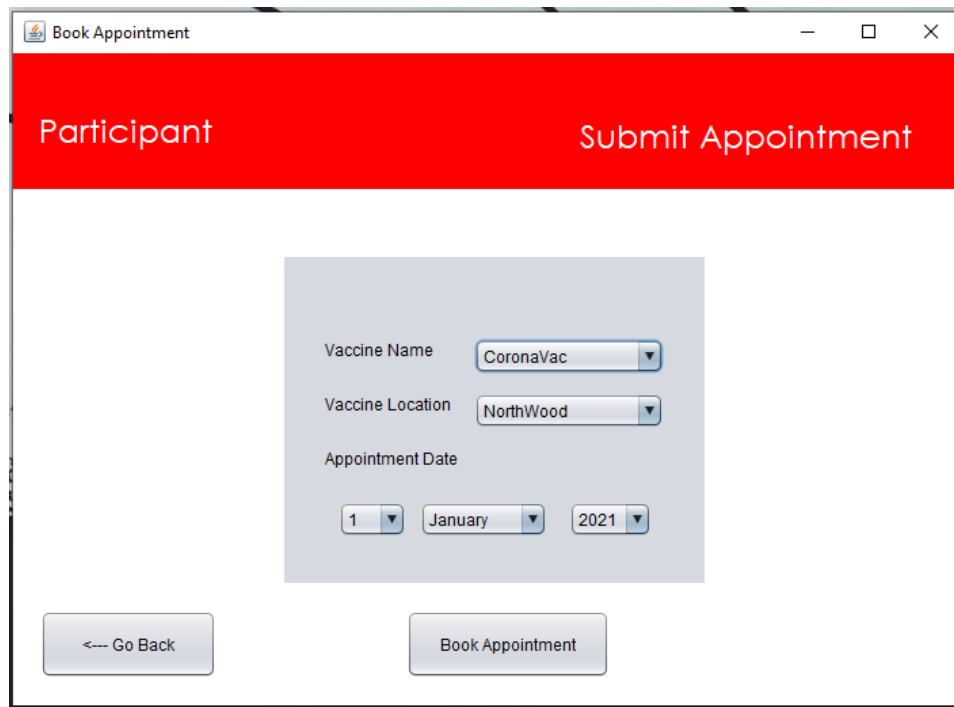
## Participant Profile Form



The screenshot shows a web application window titled "Profile". The window has a red header bar with the text "Participant" on the left and "Your Profile" on the right. The main content area is divided into two columns. The left column contains the following fields: ID (5), Fullname (Anakin Skywalker), Date of birth (1 May 1988), Participant Type (Citizen, Non-Citizen), ID/IC (X120123), Appointment ID (none), and Status (Non-Vaccinated). The right column contains the following fields: Username (anakin), Password (masked with asterisks), Confirm Password (masked with asterisks), and a checkbox for Show Password. At the bottom right of the form, there are two buttons: "Confirm Changes" and "<--- Go Back".

This is the profile form, there will be the data inputted when registered to the system. The user will be able to modify the full name, date of birth, the type of person whether they are a citizen or a non-citizen, the ic/passport number, username, and password. There is also their ID, Appointment ID, and Status if they are vaccinated or not which is not able to be modified here. Then after modifying the data, they can hit the “Confirm Changes” button to change the data. And lastly the back button to go back to the home form.

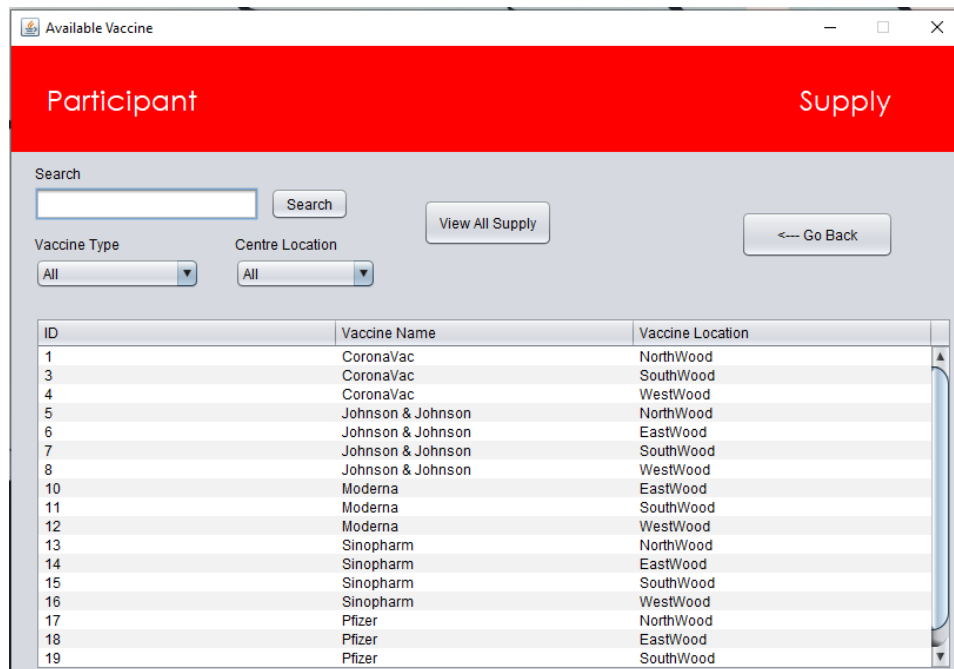
## Submit Appointment Form



The screenshot shows a Java Swing window titled "Book Appointment". The window has a red header bar with the text "Participant" on the left and "Submit Appointment" on the right. Below the header, there is a light gray rectangular area containing a form. The form has three sections: "Vaccine Name" with a dropdown menu showing "CoronaVac", "Vaccine Location" with a dropdown menu showing "NorthWood", and "Appointment Date" with three dropdown menus showing "1", "January", and "2021". Below the form area, there are two buttons: "<--- Go Back" on the left and "Book Appointment" on the right.

This form will let user to have them submit their appointment according to the vaccine name and the location along with the date when the user wants to book the vaccination. They can input the vaccine name, location, and the appointment all in the combo box so the user does not need to enter anything. After selecting the data then hit the “Book Appointment” button to add the appointment to the system. The go back button brings the user back to the home form.

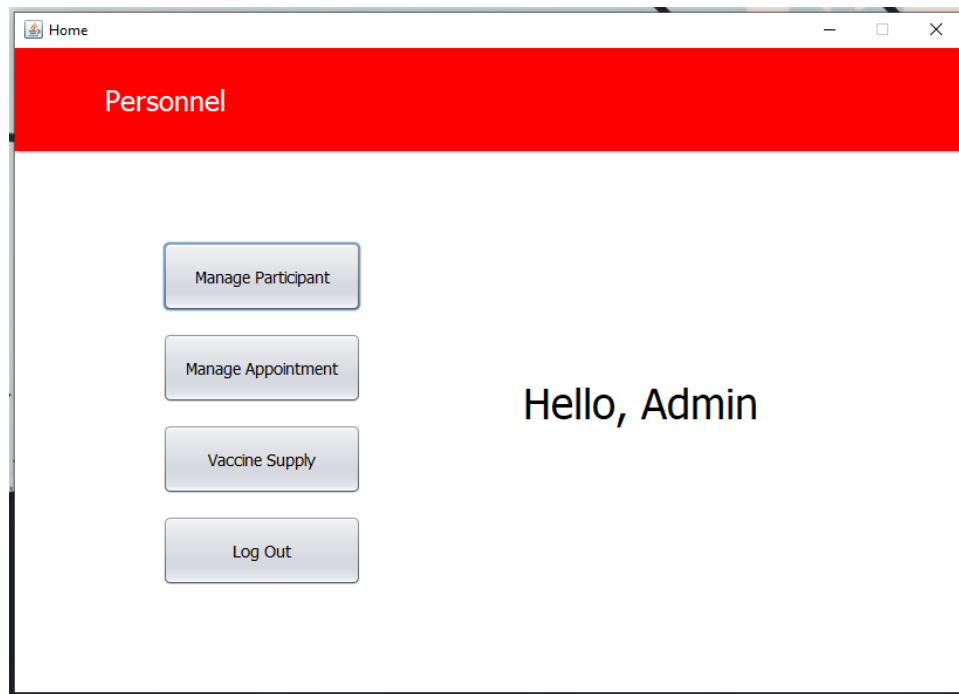
## Available Vaccine Form



ID	Vaccine Name	Vaccine Location
1	CoronaVac	NorthWood
3	CoronaVac	SouthWood
4	CoronaVac	WestWood
5	Johnson & Johnson	NorthWood
6	Johnson & Johnson	EastWood
7	Johnson & Johnson	SouthWood
8	Johnson & Johnson	WestWood
10	Moderna	EastWood
11	Moderna	SouthWood
12	Moderna	WestWood
13	Sinopharm	NorthWood
14	Sinopharm	EastWood
15	Sinopharm	SouthWood
16	Sinopharm	WestWood
17	Pfizer	NorthWood
18	Pfizer	EastWood
19	Pfizer	SouthWood

This form is where user can see where the available location according to the vaccine. The search button to let user search the vaccine detail according to the id. Then there is the vaccine type combo box and the location combo box so that they can find the detail they want to see on the table. Again, go back button to bring the user to the home form.

## Personnel Home Form



This is the home form for the Personnel. Here Personnel can access form back to the login, to manage the people/participants, manage the vaccine appointments and manage the vaccine supply. To reach those forms with the buttons on the home form shown in the image. Along with a label that says “Hello, Admin” to make it user friendly.

## View Participant Form

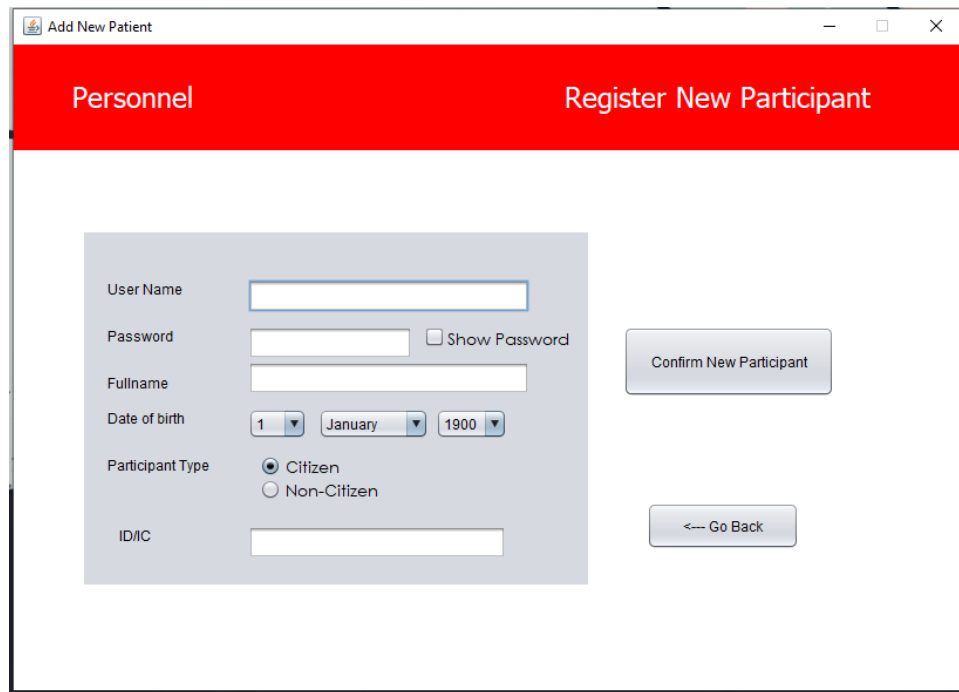
The screenshot shows a web application window titled "Participant". The main content area has a red header with "Personnel" on the left and "View Participant Data" on the right. Below the header, there are search filters: "Search by" with radio buttons for "ID" (selected) and "Name", and a text input "Enter Name/ID". There are two buttons: "All Participant" and "Search Participant". Below the search filters is a table with 4 columns: "ID", "Fullname", "AppointmentID", and "Vaccination Sta...". The table has 5 rows of data. Below the table is a button "Register New Participant". To the right of the table is a detailed view for the selected participant (ID 1, Brian jk). It includes fields for "ID", "Fullname", "Date of birth" (with dropdowns for day, month, and year), "Participant Type" (radio buttons for "Citizen" and "Non-Citizen", with "Non-Citizen" selected), "IC/Passport Number", "Appointment ID", and "Vaccine Status" (radio buttons for "Non-Vaccinated" and "Vaccinated", with "Non-Vaccinated" selected). There are buttons "Confirm Change", "Clear Data", and "<--- Go Back".

ID	Fullname	AppointmentID	Vaccination Sta...
1	Brian jk	2	Non-Vaccinated
2	Zheng	2	Non-Vaccinated
3	joseph	3	Non-Vaccinated
4	ora the seco	3	Vaccinated
5	Anakin Skywalk...	none	Non-Vaccinated

In this form, personnel will be able to see all the participants at the table. Personnel can search a specific participant by the ID or the name according to the radio button selected, after typing the ID or name then use the “Search Participant” button to search the participant and the data will be shown in the table. Personnel will also be able to edit the data of each participant and change their status from vaccinated into non-vaccinated. Personnel can change the data of the participant and hit the “Confirm Change” button to change the data. They can submit an appointment for the selected participant. The “Clear Data” button will clear the selected data on the panel. Here personnel will also be able to register a new participant, by clicking the button it will open another form to register the new participant’s data.



## Register New Participant Form



The screenshot shows a window titled "Add New Patient" with a red header bar. The header bar contains the text "Personnel" on the left and "Register New Participant" on the right. Below the header, there is a form with the following fields and controls:

- User Name:
- Password:  ☐ Show Password
- Fullname:
- Date of birth:
- Participant Type: ☒ Citizen ☐ Non-Citizen
- ID/IC:

There are two buttons on the right side of the form:

- Confirm New Participant
- <--- Go Back

This form is dedicated for personnel to register a new participant into the system so that participants can submit their own appointment and see their own data in the forms shown and explained above. Personnel will enter the data of the participant like username, password, full name, date of birth, participant type, and the ic/passport number.

## Appointment Form

The screenshot shows a web application window titled "Appointment". The interface is divided into two main sections: "Personnel" on the left and "Appointment" on the right, both highlighted with a red header. The "Personnel" section contains a search bar with the text "Search by People's ID" and a "Search" button. Below the search bar is a table with four columns: AppointmentID, AppointmentDate, SupplyID, and PeopleID. The table contains four rows of data. The "Appointment" section contains a form with fields for Appointment ID, Appointment Date (with a date picker set to 1 January 1900), Vaccine Name, Vaccine Location, People Name, and People Status. Below the form are buttons for "Modify", "Delete", "<--- Go Back", and "Clear Data".

AppointmentID	AppointmentDate	SupplyID	PeopleID
1	17/8/2021	14	2
2	5/2/2021	11	1
3	6/6/2021	11	3
4	7/7/2021	9	1

This appointment form will allow personnel to see all the appointments on the table. The table will show the appointment id, appointment date, supply id for the appointment, and the participant of that appointment. The panel on the right side of the form will show the selected data and allow personnel to modify or delete the appointment. The “Clear Data” button will clear the selected data on the panel.

## Supply Form

ID	Vaccine Name	Vaccine Location	Supply
1	CoronaVac	NorthWood	70
3	CoronaVac	SouthWood	400
4	CoronaVac	WestWood	400
5	Johnson & John...	NorthWood	400
6	Johnson & John...	EastWood	400
7	Johnson & John...	SouthWood	400
8	Johnson & John...	WestWood	400
10	Moderna	EastWood	400
11	Moderna	SouthWood	398
12	Moderna	WestWood	400
13	Sinopharm	NorthWood	400
14	Sinopharm	EastWood	399
15	Sinopharm	SouthWood	400
16	Sinopharm	WestWood	400
17	Pfizer	NorthWood	400
18	Pfizer	EastWood	400
19	Pfizer	SouthWood	400

This form is the form for personnel to see all the supply available, the search id will be used to find the supply with the searched id. The “Vaccine Type” and “Centre Location” combo box will filter the table according to the selected data in the combo box. “View All Supply” button will display all the data into the table. When a data is selected from the table, it will show all the data in the panel. In the panel when a data is selected in the table, personnel can modify the number of supply or add the number of supplies of the vaccine according to the location. Personnel is also able to remove supply. Lastly, the back button will direct the personnel back to the home form.

## Conclusion

The system will be able to:

1. I conclude that the personnel and participant can login to the system by entering their username and password.
2. I conclude that with the help of the system that the participant can register and have their data store in the textile.
3. I conclude that with the help of the system the personnel can modify the participant's full name, date of birth, participant type, ic/passport number, and vaccine status.
4. I conclude that with the help of the system that the personnel can search participant either by ID or name.
5. I conclude that with the help of the system that the personnel can register participant.
6. I conclude that with the help of the system the personnel can view all participant in a table format.
7. I conclude that with the help of the system that the personnel can search for appointment by typing the participant's ID.
8. I conclude that with the help of the system that the personnel can add appointment to the specific participant.
9. I conclude that with the help of the system that the personnel can remove appointment by choosing the table.
10. I conclude that with the help of the system the personnel can modify the appointment's date.
11. I conclude that with the help of the system the personnel can view all appointment in a table format.
12. I conclude that with the help of the system the personnel can view all the vaccine supply from each location and type.
13. I conclude that with the help of the system the personnel can modify the amount of vaccine supply from each location and type.
14. I conclude that with the help of the system the personnel can add the vaccine supply from each location and type.

15. I conclude that with the help of the system the personnel can search vaccine supply by ID, vaccine type, or centre location.
16. I conclude that with the help of the system the personnel can remove the vaccine supply of a specific type and location by turning the supply amount to 0.
17. I conclude that with the help of the system the participant may view their own account.
18. I conclude that with the help of the system the participant may modify their own account's full name, IC/Passport number, username, password, citizen type, and date of birth.
19. I conclude that with the help of the system the participant can view their own appointment from their home page.
20. I conclude that with the help of the system the participant can cancel their own appointment from their home page.
21. I conclude that with the help of the system the participant may register for their own appointment by picking the vaccine type, centre location, and the date of the appointment.
22. I conclude that with the help of the system the participant may view vaccination availability of each vaccine type and centre location
23. I conclude that with the help of the system the participant may search vaccination availability by vaccine type and centre location

## References

Baeldung, B. (2019, October 9). *Control Structures in Java*. Baeldung.

<https://www.baeldung.com/java-control-structures>

GeeksforGeeks. (2018, September 6). *Exceptions in Java*.

<https://www.geeksforgeeks.org/exceptions-in-java/>

GeeksforGeeks. (2019, January 3). *File handling in Java using FileWriter and FileReader*.

<https://www.geeksforgeeks.org/file-handling-java-using-filewriter-filereader/>

GeeksforGeeks. (2021a, August 27). *Java AWT / Dimension Class*.

<https://www.geeksforgeeks.org/java-awt-dimension-class/>

GeeksforGeeks. (2021b, October 12). *Java Swing / JTable*. [https://www.geeksforgeeks.org/java-](https://www.geeksforgeeks.org/java-swing-jtable/)

[swing-jtable/](https://www.geeksforgeeks.org/java-swing-jtable/)

*Java For Loop*. (2021). W3school.Com. [https://www.w3schools.com/java/java\\_for\\_loop.asp](https://www.w3schools.com/java/java_for_loop.asp)

JavaTPoint. (2021a). *Java BufferedWriter Class - javatpoint*. WwW.Javatpoint.Com.

<https://www.javatpoint.com/java-bufferedwriter-class>

JavaTPoint. (2021b). *Java BufferedWriter Class - javatpoint*. WwW.Javatpoint.Com.

<https://www.javatpoint.com/java-bufferedwriter-class>

JavaTPoint. (2021c). *Java If else - Javatpoint*. WwW.Javatpoint.Com.

<https://www.javatpoint.com/java-if-else>

Jenkov, J. (2021a). *Java Constructors*. Tutorials.Jenkov.Com.

<http://tutorials.jenkov.com/java/constructors.html>

Jenkov, J. (2021b). *Java IO: PrintWriter*. Tutorials.Jonkov.Com.

[http://tutorials.jenkov.com/java-io/printwriter.html#:~:text=PrintWriter%20\)%20enables%20you%20to%20write,to%20mix%20text%20and%20numbers](http://tutorials.jenkov.com/java-io/printwriter.html#:~:text=PrintWriter%20)%20enables%20you%20to%20write,to%20mix%20text%20and%20numbers)

Oracle. (2020, June 24). *DefaultTableModel (Java Platform SE 7 )*. Oracle.Com.

<https://docs.oracle.com/javase/7/docs/api/javax/swing/table/DefaultTableModel.html>

*Removing repetitive code (overloading methods and constructors) - Java Programming*. (2021).

java-programming.mooc.fi. <https://java-programming.mooc.fi/part-5/2-method-and-constructor-overloading>

Techopedia. (2021, September 9). *Java Object*. Techopedia.Com.

<https://www.techopedia.com/definition/24339/java-object>

W3school. (2021). *Java ArrayList*. W3school.Com.

[https://www.w3schools.com/java/java\\_arraylist.asp](https://www.w3schools.com/java/java_arraylist.asp)

W3School. (2021a). *Java Classes and Objects*. W3school.Com.

[https://www.w3schools.com/java/java\\_classes.asp](https://www.w3schools.com/java/java_classes.asp)

W3School. (2021b). *Java Date and Time*. W3school.Com.

[https://www.w3schools.com/java/java\\_date.asp](https://www.w3schools.com/java/java_date.asp)

W3School. (2021c). *Java Encapsulation and Getters and Setters*. W3school.Com.

[https://www.w3schools.com/java/java\\_encapsulation.asp](https://www.w3schools.com/java/java_encapsulation.asp)

W3School. (2021d). *Java Inheritance (Subclass and Superclass)*. W3school.Com.

[https://www.w3schools.com/java/java\\_inheritance.asp](https://www.w3schools.com/java/java_inheritance.asp)

W3School. (2021e). *Java Lambda Expressions*. W3school.Com. Retrieved December 5, 2021,

from [https://www.w3schools.com/java/java\\_lambda.asp](https://www.w3schools.com/java/java_lambda.asp)

W3School. (2021f). *Java Methods*. W3school.Com.

[https://www.w3schools.com/java/java\\_methods.asp](https://www.w3schools.com/java/java_methods.asp)

W3School. (2021g). *Java While Loop*. W3schools.Com.

[https://www.w3schools.com/java/java\\_while\\_loop.asp](https://www.w3schools.com/java/java_while_loop.asp)