E1900340_BIT203_ASSIGNMENT 2

by I KETUT MAHENDRA -

Submission date: 09-Jan-2022 10:56PM (UTC+0800)

Submission ID: 1739098175

File name: 16106_I_KETUT_MAHENDRA_-_E1900340_BIT203_ASSIGNMENT2_248658_1657454143.docx (5.02M)

Word count: 11522

Character count: 98129

Assignment No.: __2__

HELP University
university of achievers

Assignment Cover Sheet

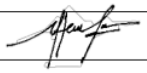
Student Information (For group assignment, please state names of all members)		Grade/Marks
Name	**ID**	
I Ketut Mahendra	E1900340	

Module/Subject Information		Office Acknowledgement
Module/Subject Code	BIT203	
Module/Subject Name	Advance OO Programming	
Lecturer/Tutor/Facilitator	Kok Chye Hock	
Due Date	9th January 2022	
Assignment Title/Topic	Assignment 2	
Intake (where applicable)	Semester September 2021	
Word Count	11.460	**Date/Time**

Declaration

- I/We have read and understood the Programme Handbook that explains on **plagiarism**, and I/we testify that, unless otherwise acknowledged, the work submitted herein is entirely my/our own.
- I/We declare that no part of this assignment has been written for me/us by any other person(s) except where such collaboration has been authorized by the lecturer concerned.
- I/We authorize the University to test any work submitted by me/us, using text comparison software, for instances of plagiarism. I/We understand this will involve the University or its contractors copying my/our work and storing it on a database to be used in future to test work submitted by others.

Note: 1) The attachment of this statement on any electronically submitted assignments will be deemed to have the same authority as a signed statement.
2) The Group Leader signs the declaration on behalf of all members.

Signature: 	Date: 9th January 2021
E-mail: 190030382@stikom-bali.ac.id	

Assignment No.: __2__

Feedback/Comments*
Main Strengths
Main Weaknesses
Suggestions for improvement

	Student acknowledge feedback/comments
Grader's signature	Student's signature:
Date:	Date:

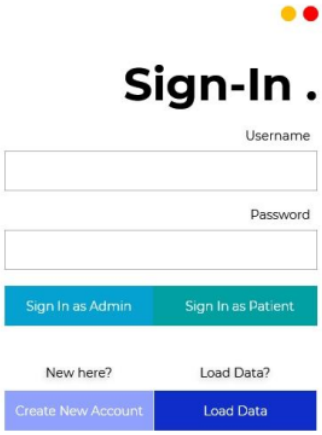
Note:
1) A soft and hard copy of the assignment shall be submitted.
2) The signed copy of the assignment cover sheet shall be retained by the marker.
3) If the Turnitin report is required, students have to submit it with the assignment. However, departments may allow students up to **THREE** (3) working days after submission of the assignment to submit the Turnitin report. The assignment shall only be marked upon the submission of the Turnitin report.

*Use additional sheets if required.

Table of Contents

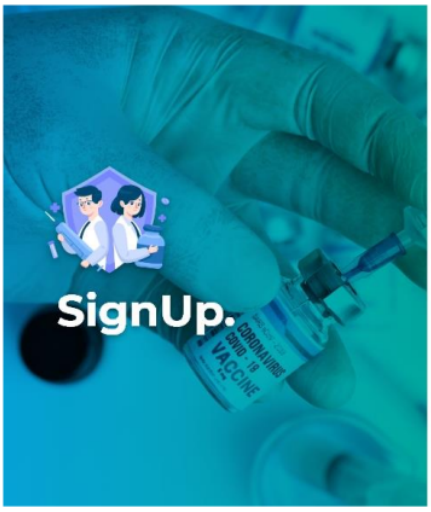
User Interface Design

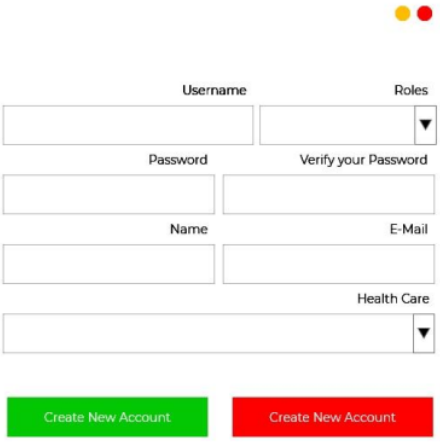
1. JFrame Login - JPanel loginPanel



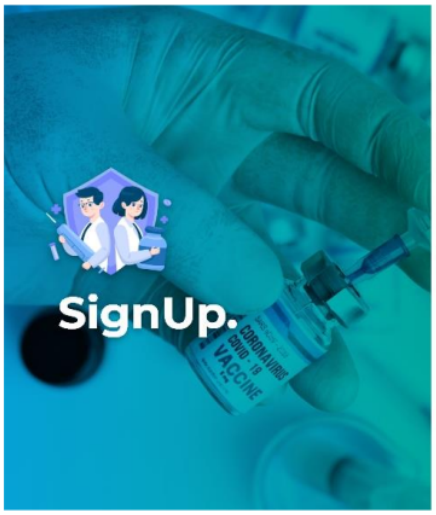
2. JFrame Sign Up - JPanel signUpPanel

Role: Administrator

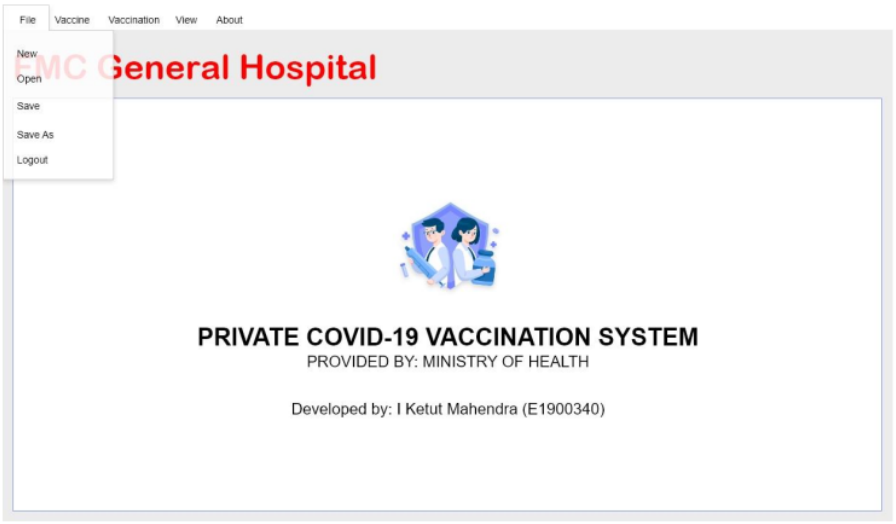




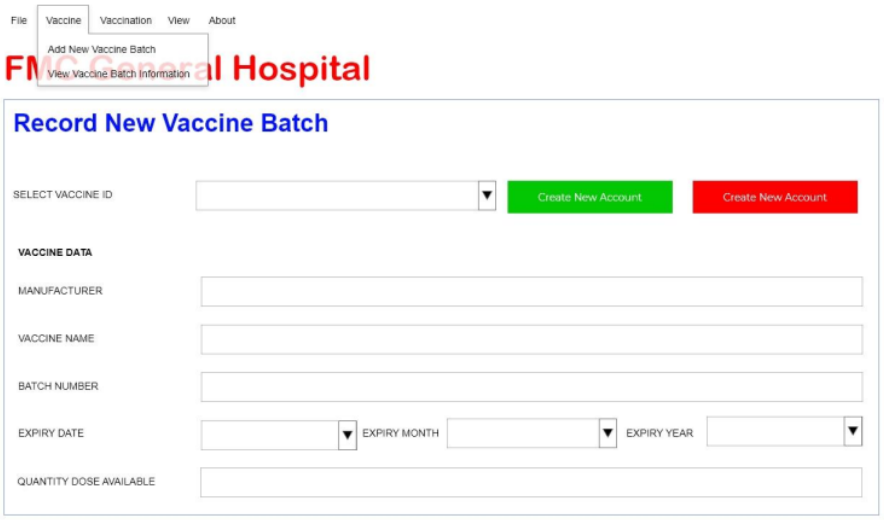
Role: Patient



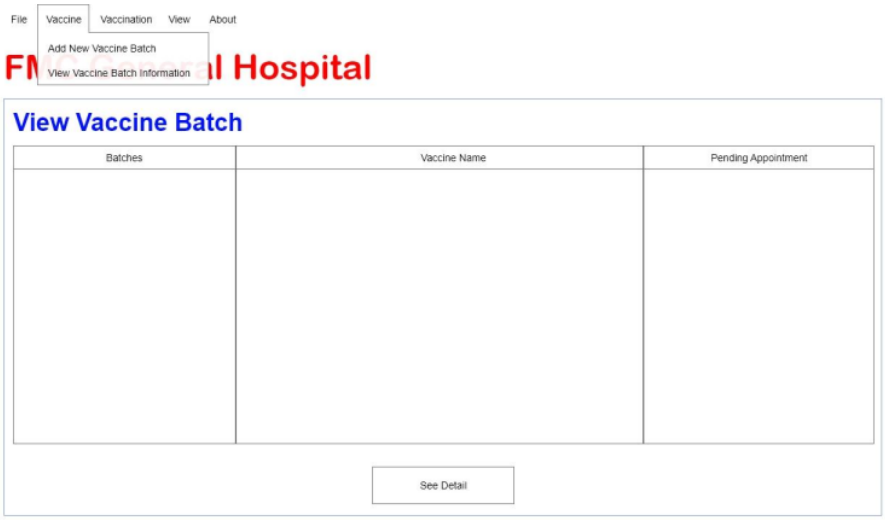
3. JFrame Admin Dashboard - JPanel Home Page



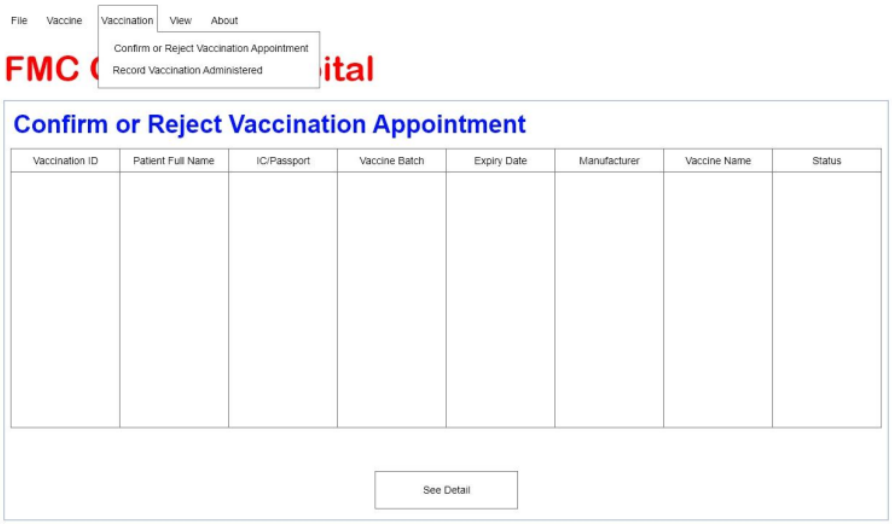
4. JFrame Admin Dashboard - JPanel Add New Vaccine Batch



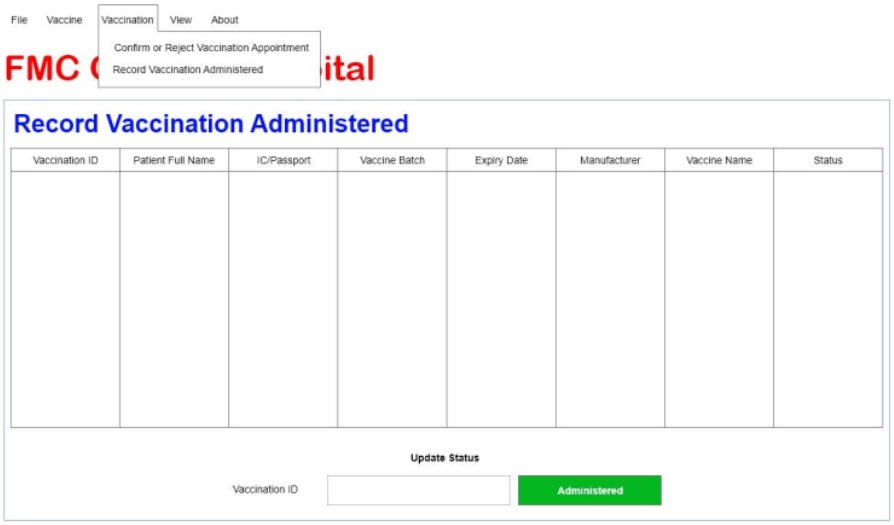
5. JFrame Admin Dashboard - View Vaccine Batch Information



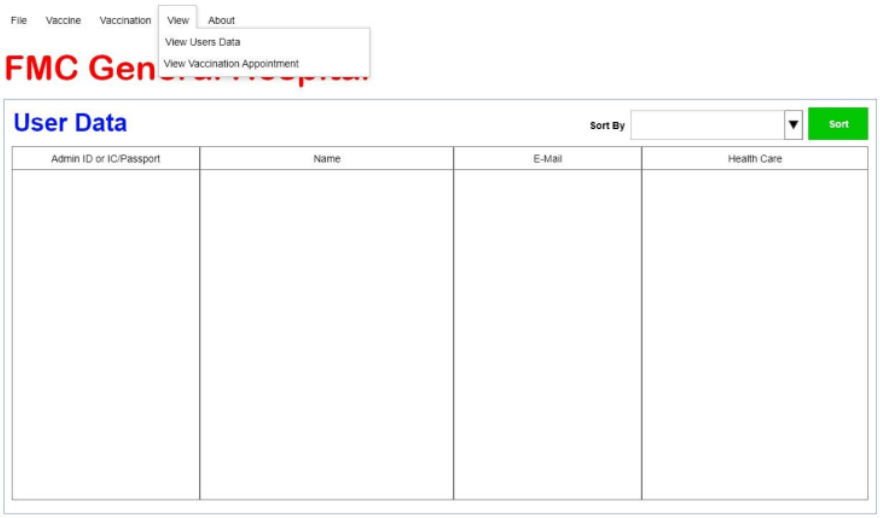
6. JFrame Admin Dashboard - JPanel Confirm or Reject Vaccination Appointment



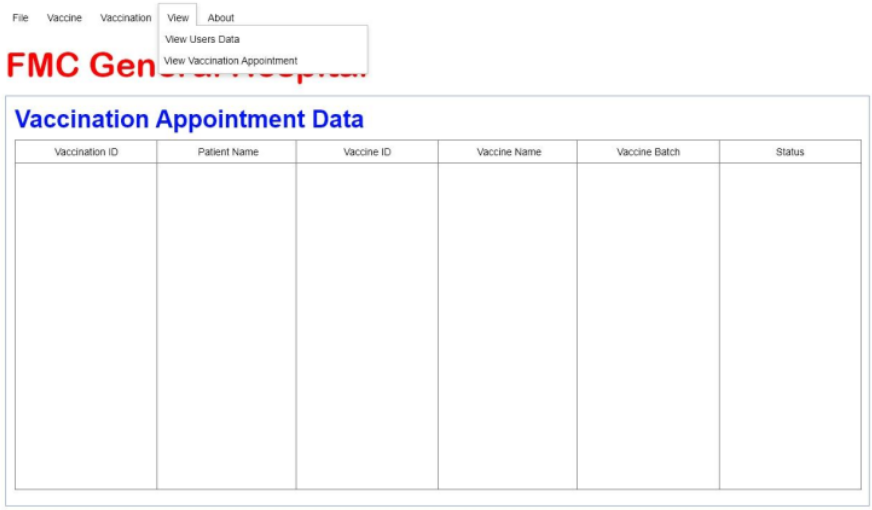
7. JFrame Admin Dashboard - JPanel Record Vaccination Administered



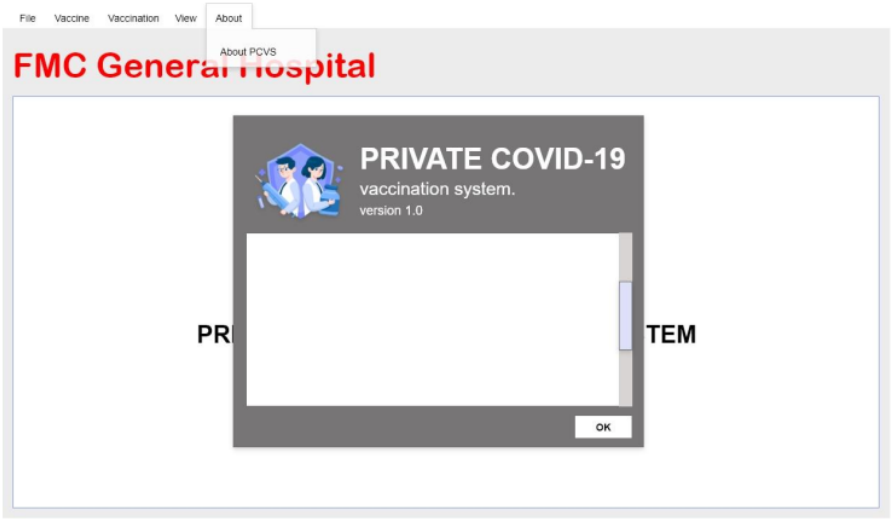
8. JFrame Admin Dashboard - JPanel View Users Data



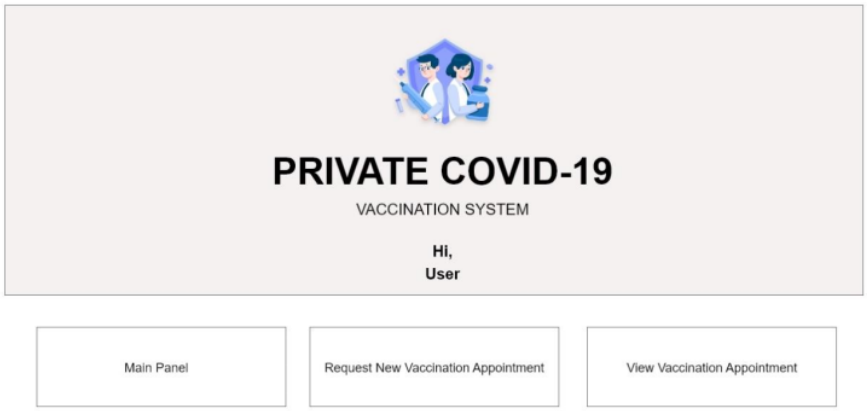
9. JFrame Admin Dashboard - JPanel View Vaccination Appointment



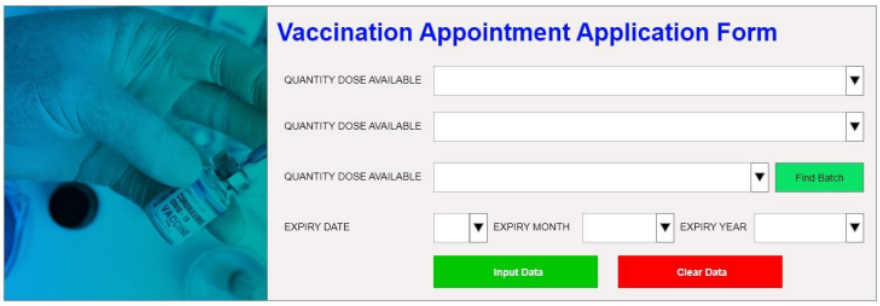
10. JDialog About PCVS



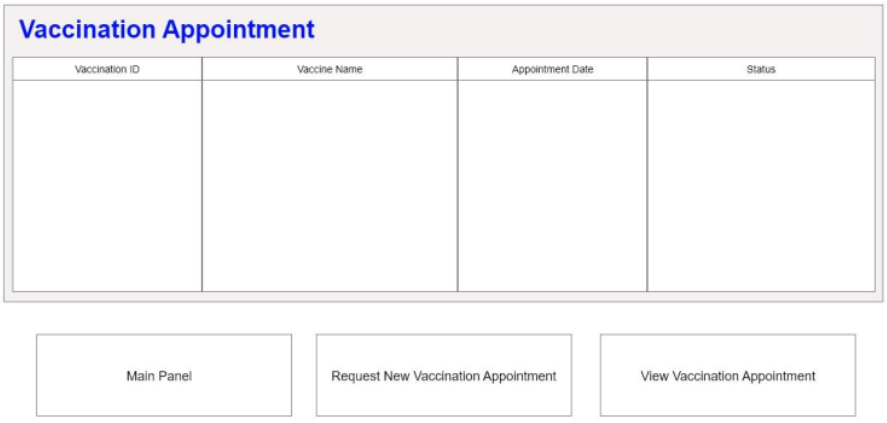
11. JFrame Patient Dashboard - Main Panel



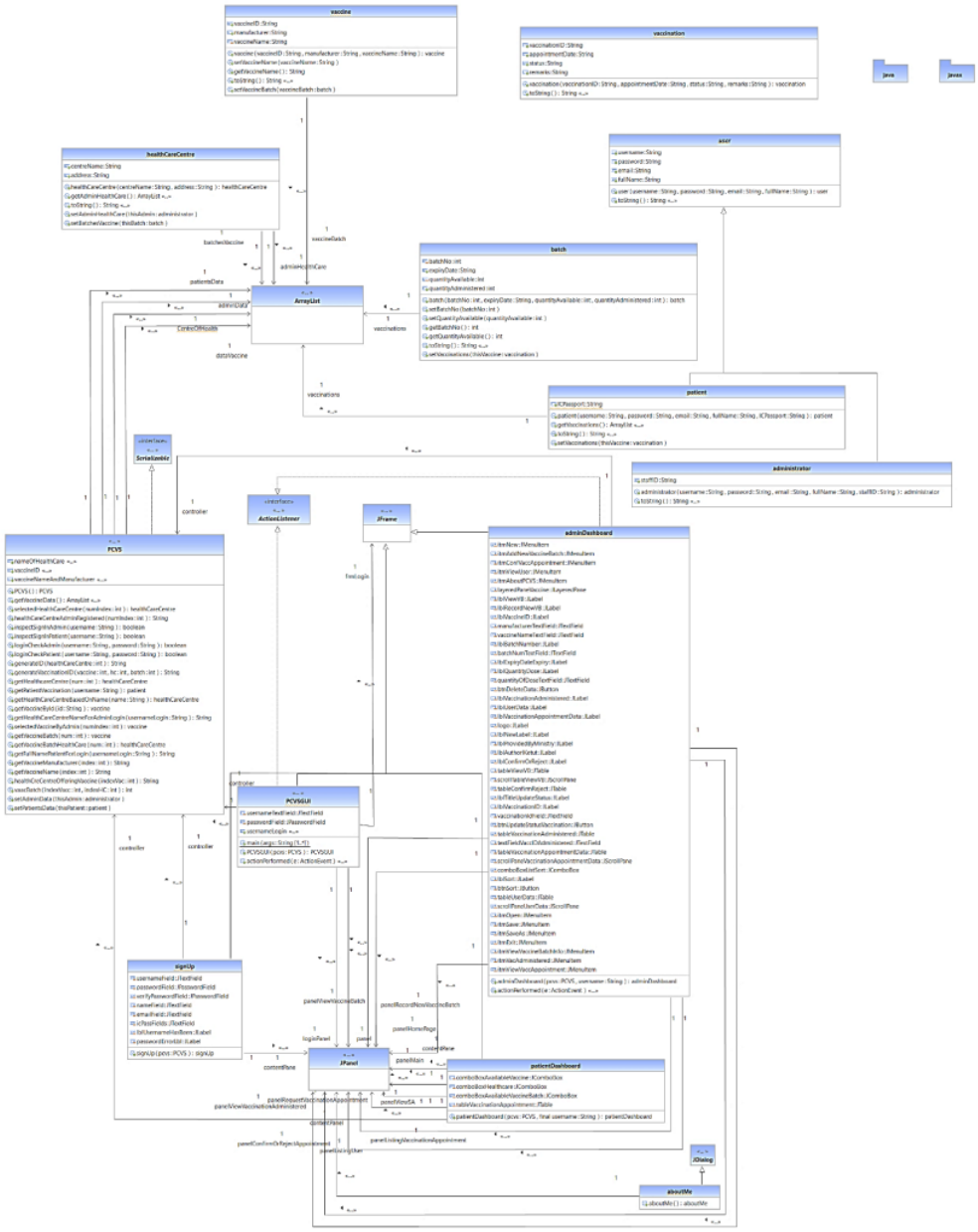
12. JFrame Patient Dashboard - Request Vaccination Appointment



13. JFrame Patient Dashboard - View Vaccination Appointment



Class Diagram



Java Source Code

vs.java

```
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Iterator;

import javax.swing.ComboBoxModel;

/**
 * PCVS.java is a class controller that contains a whole method that
 * will be used to process the input of the users
 *
 * @author I Ketut Mahendra
 */
public class PCVS implements Serializable
{
    //instance variable
    private ArrayList<healthCareCentre> CentreOfHealth;
    private ArrayList<vaccine> dataVaccine;
    private ArrayList<administrator> adminData;
    private ArrayList<patient> patientsData;

    /**
     * constructor of PCVS.java
     */
    public PCVS()
    {
        CentreOfHealth = new ArrayList<healthCareCentre>();
        CentreOfHealth.add(new healthCareCentre
                  ("FMC General Hospital", "Jalan Arjuna 18, "
                                + "Gianyar, Bali"));
        CentreOfHealth.add(new healthCareCentre
                  ("SMCDenpasar", "Jalan P.B. Sudirman, Denpasar, "
                                + "Bali"));

        dataVaccine = new ArrayList<vaccine>();
        dataVaccine.add(new vaccine
                  ("mRNA-1273", "Moderna", "Cov-19 Moderna"));
        dataVaccine.add(new vaccine
                  ("BNT162B2", "BioNTech, Pfizer", "Pfizer"));

        this.adminData = new ArrayList<>();

        this.patientsData = new ArrayList<>();
    }

    /**
     * getter method of CentreOfHealth.
     * the method will return the CentreOfHealth object.
     */
    public ArrayList<healthCareCentre> getCentreOfHealth()
    {
        return CentreOfHealth;
    }

    /**
     *getter method of dataVaccine
     * the method will return the dataVaccine object.
     */
    public ArrayList<vaccine> getVaccineData()
```

```
    {
        return dataVaccine;
    }

    /**
     * getter method of AdminData.
     * the method will return the adminData object.
     */
    public ArrayList<administrator> getAdminData()
    {
        return adminData;
    }

    /**
     * getter method of patientsData.
     * the method will return the patientsData object.
     */
    public ArrayList<patient> getPatientsData()
    {
        return patientsData;
    }

    /**
     * setter method of adminData.
     * @param thisAdmin it is the parameter of the method
     * with data type: administrator
     */
    public void setAdminData(administrator thisAdmin)
    {
        this.adminData.add(thisAdmin);
    }

    /**
     * setter method of patientsData.
     * @param thisPatient it is the parameter of the method
     *  with data type: patient
     */
    public void setPatientsData(patient thisPatient)
    {
        this.patientsData.add(thisPatient);
    }

    /**
     * method to get only the name of the healthcare centre
     * in CentreOfHealth.
     * the method will return String data of healthcare centre
     * name
     */
    public String getNameOfHealthCare()
    {
        String nameHealthCare = "";
        for (int i = 0; i < CentreOfHealth.size(); i++)
        {
            nameHealthCare = nameHealthCare + (i + 1) + (". ") +
                        CentreOfHealth.get(i).getCentreName() +
                        ("\n");
        }
        return nameHealthCare;
    }

    /**
```

```
     * method that is used for getting the data of the healthcare
     * centre using the index of the healthcare centre array after
     * the administrator select the healthcare centre name before
     * the administrator wants to register himself.
     * @param numIndex is the parameter of the method that will
     * contain integer data of the Healthcare Centre Index.
     * @return CentreOfHealth it will return the data from an
     * index of CentreOfHealth.
     */
    public healthCareCentre selectedHealthCareCentre(int numIndex)
    {
        return CentreOfHealth.get(numIndex);
    }

    /**
     * method that is used for returning a notice that the
     * administrator has been registered at a healthcare
     * centre.
     * @param numIndex is the parameter of the method that
     * will contain the index number of healthcare centre
     * with data type: int
     * @return registeredAdminHealthCare it will return the
     * notice.
     */
    public String healthCareCentreAdminRegistered(int numIndex) {
        String registeredAdminHealthCare = "";
        for (int i = 0; i < CentreOfHealth.size(); i++)
        {
            if (i == numIndex) {
                registeredAdminHealthCare =
                        registeredAdminHealthCare +
                        "You are successfully registered" +
                        "Healthcare Centre  : "+ CentreOfHealth.
                        get(i).getCentreName() + "Address    : " +
                        CentreOfHealth.
                        get(i).getAddress();
            }
        }
        return registeredAdminHealthCare;
    }

    /**
     * method used to verify the admin username when
     * the admin wants to sign-in to the system.
     * the method will return false if the data entered
     * by the user (according to the parameters)
     * is the same as the username data that has been
     * recorded in the system,
     * the method will return true if the opposite happens.
     * @param username is the parameter of the method that
     * will contain the username that will be compared.
     * username has data type: String
     * @return boolean type (true/false)
     */
    public boolean inspectSignInAdmin(String username)
    {
        for (int i = 0; i < adminData.size(); i++)
        {
            if (adminData.get(i).getUsername().
                        compareTo(username) == 0)
                return true;
```

```
        }
        return false;
    }

    /**
     * method used to verify the patient username
     * when the patient wants to sign-in to the system.
     * the method will return false if the data entered
     * by the user (according to the parameters) is the
     * same as the username data that has been recorded in
     * the system, the method will return true if the
     * opposite happens.
     * @param username is the parameter of the method that
     * will contain the username that will be compared.
     * username has data type: String
     * @return boolean type (true/false)
     */
    public boolean inspectSignInPatient(String username)
    {
        for (int i = 0; i < patientsData.size(); i++)
        {
            if (patientsData.get(i).getUsername().
                        compareTo(username) == 0)
                return true;
        }
        return false;
    }

    /**
     * method used to verify the username and password
     * when the administrator wants to log into the system,
     * this method will check the suitability of the username
     * and password entered by the administrator with the data
     * owned by the system, if appropriate then the method will
     * return true, and vice versa if the method does not find
     * the appropriate username and password data,
     * the method will return false.
     * @param username is the parameter of the method that will
     * contain the username that will be compared.
     * @param password is the parameter of the method that will
     * contain the password that will be compared.
     * @return boolean type (true/false)
     */
    public boolean loginCheckAdmin(String username,
            String password) {
        for (int i = 0; i < adminData.size(); i++) {
            if (adminData.get(i).getUsername().equals(username)
                        && adminData.get(i).getPassword().
                        equals(password))
                return true;
        }
        return false;
    }

    /**
     * method used to verify the username and password when
     * the patient wants to log into the system, this method will
     * check the suitability of the username and password entered
     * by the patient with the data owned by the system, if
     * appropriate then the method will return true, and vice versa
     * if the method does not find the appropriate username and
```

```
     * password data, the method will return false.
     * @param username is the parameter of the method that will
     * contain the username that will be compared.
     * @param password is the parameter of the method that will
     * contain the password that will be compared.
     * @return boolean type (true/false)
     */
    public boolean loginCheckPatient(String username,
            String password) {
        for (int i = 0; i < patientsData.size(); i++) {
            if (patientsData.get(i).getUsername().
                        equals(username) && patientsData.get(i).
                        getPassword().equals(password))
                return true;
        }
        return false;
    }

    /**
     * method used to generate the staffID automatically for
     * the Administrator.
     * the staffID pattern is A + HealthCareIndex + 4 digits
     * index number of administrator
     * character "A" : Administrator
     * @param healthCareCentre is the parameter of the method
     * that will contain the healthcare index
     * @return Staff ID : Sample: A10001
     */
    public String generateID(int healthCareCentre)
    {
        String StaffId = "";
        for (int i = 0; i < adminData.size(); i++)
        {
            String range = String.format("%04d", i+1);
            StaffId = "A" + healthCareCentre + range;
        }
        return StaffId;
    }

    /**
     * method used to generate the vaccination ID
     * automatically
     * @param vaccine is the parameter vaccine that
     * selected by patient
     * @param hc is the parameter health care centre
     * that selected by patient
     * @param batch is the parameter vaccine batch
     * that selected by patient
     * @return this method will return the vaccination id
     */
    public String generateVaccinationID(int vaccine, int hc,
            int batch) {
            String vaccinationID = "";
            for (int i = 0; i < 100; i++) {
                  String range = String.format("%04d", i+1);
                  vaccinationID = "VA"+(vaccine+1)+"iHC"+(hc+1)+
                              "sBt"+(batch+1);
            }
            return vaccinationID;
      }
```

```
    /**
     * method used to return the Healthcare Centre based on
     * index of health care
     * @param num index of healthcare centre
     * @return healthcare centre object based on the index
     */
    public healthCareCentre getHealthcareCentre(int num) {
        for (int i = 0; i < CentreOfHealth.size(); i++) {
            if (i == num)
                return CentreOfHealth.get(i);
        }
        return null;
    }

    /**
     * method used to return the patient data based on
     * username data
     * @param username parameter username of the patient
     * @return patient data object based on the username
     */
    public patient getPatientVaccination(String username) {
      for (int i = 0; i < patientsData.size(); i++) {
                  if(patientsData.get(i).getUsername()==username) {
                        return patientsData.get(i);
                  }
            }
      return null;
    }

    /**
     * method used to return the health care centre based
     * on healthcare centre login
     * @param name is the name of health care centre
     * @return health care centre object based on the name
     */
    public healthCareCentre getHealthCareCentreBasedOnName(String
            name) {
            for(int i = 0; i < CentreOfHealth.size();i++) {
                  if(CentreOfHealth.get(i).getCentreName().
                              equals(name))
                        return CentreOfHealth.get(i);
            }
            return null;
      }
    /**
     * method used to return the vaccine based on the
     * vaccine id
     * @param id is the vaccine id
     * @return it will return the vaccine object based
     * on the vaccine id
     */
    public vaccine getVaccineById(String id) {
      for(int i = 0; i < dataVaccine.size();i++) {
            if(dataVaccine.get(i).getVaccineID().
                        equals(id))
                  return dataVaccine.get(i);
      }
      return null;
    }
    /**
     * method used to return the name of healthcare
```

```
     * centre when the username login to system.
     * @param usernameLogin the parameter that contain
     * username of the administrator that will be compared,
     * and return the healthcare centre name where the method
     * found the same username
     * that inputted by administrator and the arraylist
     * @return healthCareAdmin: name of the health care
     * with "String" data type
     */
    public String getHealthCareCentreNameForAdminLogin(String
            usernameLogin){
        String healthCareAdmin = "";

        20 loop to get the centre
        for (int i = 0; i < getCentreOfHealth().size(); i++){
            healthCareCentre hc = getCentreOfHealth().get(i);

            1/ loop to get the admin
            for (int j = 0; j < hc.getAdminHealthCare().size();
                        j++)
            {
                if (hc.getAdminHealthCare().get(j).getUsername().
                        equals(usernameLogin))
                {
                    healthCareAdmin =  hc.getCentreName();
                }
            }
        }
        return healthCareAdmin;
    }

    /**
     * method that used for returning the vaccine id.
     * @return vaccine id i vaccine arraylist.
    1*/
    public String getVaccineID()
    {
        String vaccineID = "";
        for (int i = 0; i < dataVaccine.size(); i++)
        {
            vaccineID = vaccineID + (i+1) + (". ") + dataVaccine.
                        get(i).getVaccineID() + ("\n");
        }
        return vaccineID;
    }



    /**
     * method that used for displaying the detail of vaccine
     * that chosen by administrator
     * @param numIndex is the parameter that contain vaccine
     * index with data type: int
     * @return detail vaccine based on the vaccine index
     */
    public vaccine selectedVaccineByAdmin(int numIndex)
    {
        return dataVaccine.get(numIndex);
    }

    /**
```

```
     * method that used to get a vaccine object in vaccine
     * class based on the index number of the vaccine.
     * @param num is the parameter of the vaccine index that
     * inputted by the administrator
     * @return it will return null if the vaccine index is
     * not same with the input, and it will return
     *          the data if the vaccine index is same with
     *          the data in vaccine arraylist in class vaccine
     */
    public vaccine getVaccineBatch(int num) {
        for (int i = 0; i < dataVaccine.size(); i++) {
            if (i == num)
                return dataVaccine.get(i);
        }
        return null;
    }

    /**
     * method used to get a vaccine object in healthcare
     * class based on the index number of the vaccine.
     * @param num is the parameter of the vaccine index
     * that inputted by the administrator
     * @return it will return null if the vaccine index
     * is not same with the input, and it will return
     * the data
     * if the vaccine index same with data in vaccine
     * arraylist in class healthcare centre
     */
    public healthCareCentre getVaccineBatchHealthCare(int num)
    {
        for (int i = 0; i < CentreOfHealth.size(); i++)
        {
            if (i == num)
                return CentreOfHealth.get(i);
        }
        return null;
    }

    /**
     * method used for getting the full name of user and
     * displaying it when the patient successfully logged
     * in to the system
     * @param usernameLogin is the parameter of username
     * that will be used for comparing the data inputted
     * with the data in the ArrayList
     * @return patientName: the patient name based on the
     * same username on arraylist and parameter
     */
    public String getFullNamePatientForLogin(String
            usernameLogin){
        String patientName = "";
        for (int i = 0; i < getPatientsData().
                    size(); i++)
        {
            patient pName = getPatientsData().
                        get(i);
            if (pName.getUsername().
                        equals(usernameLogin))
            {
                patientName =  pName.
                        getFullName();
```

```
            }
        }
        return patientName;
    }

    /**
     * method used for return the vaccine name and
     * the manufacturer from vaccine arraylist
     * @return vaccine name and manufacturer
     */
    public String getVaccineNameAndManufacturer()
    {
        String vaccine = "";
        for (int i = 0; i < dataVaccine.size(); i++)
        {
            vaccine = vaccine + (i + 1) + ". Vaccine name: "
        + dataVaccine.get(i).getVaccineName() +
        ", Manufactured by: " +
        dataVaccine.get(i).getManufacturer() + "\n";
        }
        return vaccine;
    }

    /**
     * method used for return the vaccine manufacturer
     * based on the vaccine id that selected by user
     * @param index index number of vaccine id that
     * selected from combo box
     * @return the manufacturer name
     */
    public String getVaccineManufacturer(int index) {
      for(int i = 0; i < getVaccineData().size();i++) {
            if(i==index) {
                  return dataVaccine.get(i).getManufacturer();
            }
      }
            return null;
    }

    public String getVaccineName(int index) {
      for (int i = 0; i < getVaccineData().size(); i++) {
                  if (i==index) {
                        return dataVaccine.get(i).getVaccineName();
                  }
            }
      return null;
    }

    /**
     * method used for getting the healthcare centre that
     * offering the vaccine that selected by patient.
     * The way this method works is by looping the vaccine
     * batch in the vaccine batch arraylist in the vaccine
     * class. if it is found it will be entered into the
     * healthcareVaccineBatch variable. then looping again
     * through the vaccine batch at the healthcare center class.
     * then the data will be entered into the healthcareVaccineDataBatch
     * variable. then a comparison will be made between the results in
     * the healthcareVaccineBatch variable and the
     * healthcareVaccineDataBatch variable using if.
     * if a match is found, healthcareVaccineDataBatch
```

```
     * will be added to the storeHealthCareProvideVaccine arraylist.
     * then a for loop is done to enter the data into the
     * centerHealthOfferingVac variable and then return it
     * @param indexVac
     * @return
     */
    public String healthCreCentreOfferingVaccine(int indexVac)
    {
        String centreHealthOfferingVac = "";
        ArrayList<healthCareCentre> storeHealthCareProvideVaccine
        = new ArrayList<>();
        vaccine centreHealthVaccine = getVaccineBatch(indexVac);
        for (int i = 0; i < centreHealthVaccine.getVaccineBatch().
                size(); i++)
        {
            batch healthcareVaccineBatch = centreHealthVaccine.
                    getVaccineBatch().get(i);

            for (int j = 0; j < getCentreOfHealth().size(); j++)
            {
                healthCareCentre healthcareVaccineDataBatch =
                        getCentreOfHealth().get(j);

                if(healthcareVaccineDataBatch.getBatchesVaccine().
                        contains(healthcareVaccineBatch))
                {
                    storeHealthCareProvideVaccine.
                    add(healthcareVaccineDataBatch);
                }
            }
        for (int k = 0; k < storeHealthCareProvideVaccine.size(); k++)
        {
            centreHealthOfferingVac = centreHealthOfferingVac +
                    (k + 1) + ". " + storeHealthCareProvideVaccine.
                    get(k) + "\n";
        }
        return centreHealthOfferingVac;
    }

    /**
     * method used for getting the batch number of vaccine based on
     * healthcare centre and type of vaccine that selected by patient
     * @param indexVacc is the index of vaccine that selected by the
     * user in JCombobox
     * @param indexHC is the index of health care that selected by
     * the user in JCombobox
     * @return batch number of vaacine
     */
    public ArrayList<String> getVaccineBatchForAppoimtment(int
            indexHC, int indexVacc)
      ArrayList<String>batchData=new ArrayList<>();
      for(int i = 0; i < getHealthcareCentre(indexHC).
                  getBatchesVaccine().size();i++) {
            batch newBatchInHC = getHealthcareCentre(indexHC).
                    getBatchesVaccine().get(i);
            for(int j = 0; j < getVaccineData().get(indexVac).
                    getVaccineBatch().size();j++) {
                  batch newBatchInVacc = getVaccineData().
                          get(indexVac).getVaccineBatch().get(j);
```

```
      if(newBatchInHC.getBatchNo()==newBatchInVacc.getBatchNo()) {
                        batchData.add(String.valueOf(newBatchInHC.
                                    getBatchNo()));
                  }
            }
      }
      return batchData;
    }
}
```

user.java

```
/**
 * user.java is an abstract class that will be
 * inherited in child class patient.java and
 * administrator.java
 *
 * @author I Ketut Mahendra
 */
public abstract class user
{
    //instance variable
    private String username;
    private String password;
    private String email;
    private String fullName;

    /**
     * the constructor of user.java
     * @param username is the username of the user
     * @param password is the password of the user
     * @param email is the email address of the user
     * @param fullName is the full name of the user
     */
    public user(String username, String password,
                String email, String fullName)
    {
        this.username = username;
        this.password = password;
        this.email = email;
        this.fullName = fullName;
    }

    /**
     * setter method for username of the user
     * @param username is the parameter of the method
     *                 that contain the username of
     *                 the user
     */
    public void setUsername(String username)
    {
        this.username = username;
    }

    /**
     * setter method for password of the user
     * @param password is the parameter of
     *                 this method that contain
     *                 the password of the user
     */
```

```
public void setPassword(String password)
{
    this.password = password;
}

/**
 * setter method for email of the user
 * @param email is the parameter of this
 *              method that contain the
 *              email of the user
 */
public void setEmail(String email)
{
    this.email = email;
}

/**
 * setter method for full name of the user
 * @param fullName is the parameter of this
 *                 method that contain the full
 *                 name of the user
 */
public void setFullName(String fullName)
{
    this.fullName = fullName;
}

/**
 * getter method for username of the user
 * @return username of the user
 */
public String getUsername()
{
    return username;
}

/**
 * getter method for password of the user
 * @return password of the user
 */
public String getPassword()
{
    return password;
}

/**
 * getter method for email of the user
 * @return email of the user
 */
public String getEmail()
{
    return email;
}

/**
 * getter method for full name of the user
 * @return full name of the user
 */
public String getFullName()
{
    return fullName;
```

```
    }

    /**
     * to string method for user.java
     * @return the username, password, email, and
     * full name of the user
     */
    @Override
    public String toString()
    {
        return "user{" +
                "username='" + username + '\'' +
                ", password='" + password + '\'' +
                ", email='" + email + '\'' +
                ", fullName='" + fullName + '\'' +
                '}';
    }

}
```

patient.java

```
//import java arraylist library
import java.util.ArrayList;

/**
 * patient.java is an extends class of user.java.
 * this class will inherit the abstract class method
 *
 * @author I Ketut Mahendra
 */
public class patient extends user
{
    //instance variable
    private String ICPassport;
    private ArrayList<vaccine> vaccinations;

    /**
     * constructor of patient.java. this constructor
     * refers to member of the superclass with super
     * keyword
     * @param username is the username of the patient
     * @param password is the password of the patient
     * @param email is the email of the patient
     * @param fullName is the full name of the patient
     * @param ICPassport is the IC Number or Passport of
     *                   the patient
     */
    public patient(String username, String password,
                   String email, String fullName,
                   String ICPassport)
    {
        super(username, password, email, fullName);
        this.ICPassport = ICPassport;
        this.vaccinations = new ArrayList<>();
    }

    /**
     * setter method for ICPassport of the user
     * @param ICPassport is the ic passport of the
     *                   patient
```

```
    */
    public void setICPassport(String ICPassport)
    {
        this.ICPassport = ICPassport;
    }

    /**
     * set method for arraylist vaccination
     * @param thisVaccine parameter of this
     *                    method it will contain
     *                    the object of the arraylist
     */
    public void setVaccinations(vaccine thisVaccine)
    {
        this.vaccinations.add(thisVaccine);
    }

    /**
     * getter method for IC Number or Passport
     * @return the IC Number or Passport of the patient
     */
    public String getICPassport()
    {
        return ICPassport;
    }

    /**
     * getter method for vaccination object in
     * arraylist vaccinations
     * @return the vaccination object of the
     * vaccination arraylist
     */
    public ArrayList<vaccine> getVaccinations()
    {
        return vaccinations;
    }

    /**
     * toString method for patient class
     * @return the IC Passport of the patient
     */
    @Override
    public String toString() {
        return "patient{" +
                "ICPassport='" + ICPassport + '\'' +
                ", vaccinations=" + vaccinations +
                '}';
    }
}
```

administrator.java

```
/**
 * administrator.java is an extends class of user.java.
 * this class will inherit the abstract class method
 *
 * @author I Ketut Mahendra
 */
public class administrator extends user
{
    //instance variable
```

```
    private String staffID;

    /**
     * constructor of administrator.java. this constructor
     * refers to member of the superclass with super keyword
     * @param username is the username of the administrator
     * @param password is the password of the administrator
     * @param email is the email of the administrator
     * @param fullName is the full name of the administrator
     * @param staffID is the staff id of the administrator
     */
    public administrator(String username, String password,
                         String email, String fullName,
                         String staffID)
    {
        super(username, password, email, fullName);
        this.staffID = staffID;
    }

    /**
     * setter method for staff id of the administrator
     * @param staffID is the staff id of the administrator
     */
    public void setStaffID(String staffID)
    {
        this.staffID = staffID;
    }

    /**
     * getter method for staff id of the administrator
     * @return the staff id of the administrator
     */
    public String getStaffID()
    {
        return staffID;
    }

    //toString

    /**
     * toString method for administrator class
     * @return staff id of the administrator
     */
    @Override
    public String toString() {
        return "administrator{" +
                "staffID=" + staffID +
                '}';
    }
}
```

healthCareCentre.java

```
//import java arraylist library
import java.util.ArrayList;

/**
 * healthCareCentre class is a class that will be a class for
 * healthcare center data
 *
 * @author I Ketut Mahendra
```

```
*/
public class healthCareCentre
{
    //instance variable
    private String centreName;
    private String address;
    private ArrayList<administrator> adminHealthCare;
    private ArrayList<batch> batchesVaccine;

    /**
     * constructor for class healthCareCentre
     * @param centreName is the parameter for name of
     *                   healthcare centre
     * @param address is the parameter for the address of
     *                healthcare centre
     */
    public healthCareCentre(String centreName, String address)
    {
        this.centreName = centreName;
        this.address = address;
        this.adminHealthCare = new ArrayList<>();
        this.batchesVaccine = new ArrayList<>();
    }

    /**
     * setter for centre of health name
     * @param centreName name of the healthcare centre
     */
    public void setCentreName(String centreName)
    {
        this.centreName = centreName;
    }

    /**
     * setter for centre of health address
     * @param address address of the healthcare centre
     */
    public void setAddress(String address)
    {
        this.address = address;
    }

    /**
     * set object arraylist of administrator in healthcare centre
     * @param thisAdmin object of the administrator arraylist
     */
    public void setAdminHealthCare(administrator thisAdmin)
    {
        this.adminHealthCare.add(thisAdmin);
    }

    /**
     * set object arraylist of vaccine batches in healthcare centre
     * @param thisBatch object of the vaccine batch arraylist
     */
    public void setBatchesVaccine(batch thisBatch)
    {
        this.batchesVaccine.add(thisBatch);
    }

    //getter
```

```
    /**
     * getter method for healthcare centre name
     * @return name of healthcare centre
     */
    public String getCentreName()
    {
        return centreName;
    }

    /**
     * getter method for healthcare centre address
     * @return address of the healthcare centre
     */
    public String getAddress()
    {
        return address;
    }

    /**
     * getter method for admin healthcare centre arraylist
     * @return object of healthcare centre admin arraylist
     */
    public ArrayList<administrator> getAdminHealthCare()
    {
        return adminHealthCare;
    }

    /**
     * getter method for vaccine in healthcare centre arraylist
     * @return object of vaccine batch arraylist
     */
    public ArrayList<batch> getBatchesVaccine() {
        return batchesVaccine;
    }

    /**
     * to string method for healthCareCentre class
     * @return healthcare centre name and healthcare
     * centre address
     */
    @Override
    public String toString()
    {
        return "Health Care Centre Name: " +
                centreName + ", Address:" + address ;
    }
}
```

vaccine.java

```
//import java arraylist library
import java.util.ArrayList;

/**
 * class vaccine is a class that will be a class for vaccine data
 *
 * @author I Ketut Mahendra
 */
public class vaccine
```

```
{
    //instance variable
    private String vaccineID;
    private String manufacturer;
    private String vaccineName;
    private ArrayList<batch>vaccineBatch;

    /**
     * constructor for class vaccine
     * @param vaccineID is the parameter for id of the vaccine
     * @param manufacturer is the parameter for vaccine
     *                     manufacturer
     * @param vaccineName is the parameter for vaccine name
     */
    public vaccine(String vaccineID, String manufacturer,
                   String vaccineName)
    {
        this.vaccineID = vaccineID;
        this.manufacturer = manufacturer;
        this.vaccineName = vaccineName;
        this.vaccineBatch = new ArrayList<>();
    }

    /**
     * setter method for vaccine id
     * @param vaccineID is the parameter of the vaccine id
     */
    public void setVaccineID(String vaccineID)
    {
        this.vaccineID = vaccineID;
    }

    /**
     * setter method for vaccine name
     * @param vaccineName is the parameter of the vaccine name
     */
    public void setVaccineName(String vaccineName)
    {
        this.vaccineName = vaccineName;
    }

    /**
     * setter method for vaccine manufacturer
     * @param manufacturer is the parameter of the vaccine
     *                     manufacturer
     */
    public void setManufacturer(String manufacturer)
    {
        this.manufacturer = manufacturer;
    }

    /**
     * method to set the vaccine batch in vaccineBatch ArrayList
     * @param vaccineBatch is the parameter of the vaccine batch that
     *                     contain the vaccine batch object
     */
    public void setVaccineBatch(batch vaccineBatch)
    {
        this.vaccineBatch.add(vaccineBatch);
    }
```

```
    /**
     * getter method for vaccine id
     * @return it will return the vaccine id
     */
    public String getVaccineID()
    {
        return vaccineID;
    }

    /**
     * getter method for vaccine name
     * @return it will return the vaccine name
     */
    public String getVaccineName()
    {
        return vaccineName;
    }

    /**
     * getter method for vaccine manufacturer
     * @return it will return the vaccine manufacturer
     */
    public String getManufacturer()
    {
        return manufacturer;
    }

    public ArrayList<batch> getVaccineBatch()
    {
        return vaccineBatch;
    }

    /toString
    @Override
    public String toString()
    {
        return "Vaccine ID: " + vaccineID +
                ", Manufacturer: " + manufacturer +
                ", Vaccine Name: " + vaccineName;
    }
}
```

batch.java

```
/**
 * class batch is a class that will be a class
 * for vaccine batch data
 *
 * @author I Ketut Mahendra
 */

import java.util.ArrayList;
public class batch
{
    private int batchNo;
    private String expiryDate;
    private int quantityAvailable;
    private int quantityAdministered;
    private ArrayList<vaccination>vaccinations;
```

```
    /**
     * constructor for class batch
     * @param batchNo is the parameter for the number
     * of batch of the vaccine
     * @param expiryDate is the parameter for expiry
     * date of the vaccine
     * @param quantityAvailable is the parameter for
     * the quantity of the vaccine that available
     * @param quantityAdministered is the parameter
     * for the quantity of vaccine that has been
     * administered to the patient
     */
    public batch(int batchNo, String expiryDate,
            int quantityAvailable,int
            quantityAdministered)
    {
        this.batchNo = batchNo;
        this.expiryDate = expiryDate;
        this.quantityAvailable = quantityAvailable;
        this.quantityAdministered = quantityAdministered;
        this.vaccinations = new ArrayList<>();
    }

    /**
     * setter method for batch number of vaccine
     * @param batchNo parameter of batch number
     */
    public void setBatchNo(int batchNo)
    {
        this.batchNo = batchNo;
    }

    /**
     * setter method for expiry date of the vaccine
     * @param expiryDate parameter of expiry date of
     * the vaccine
     */
    public void setExpiryDate(String expiryDate)
    {
        this.expiryDate = expiryDate;
    }

    /**
     * setter method for quantity available of the
     * vaccine
     * @param quantityAvailable parameter of quantity
     * available of the vaccine
     */
    public void setQuantityAvailable(int
            quantityAvailable)
    {
        this.quantityAvailable = quantityAvailable;
    }

    /**
     * setter method for quantity of the vaccine
     * that has been administered
     * @param quantityAdministered parameter of
     * quantity of the vaccine that has been
     * administered
     */
```

```
    public void setQuantityAdministered(int
            quantityAdministered)
    {
        this.quantityAdministered = quantityAdministered;
    }

    /**
     * set method for arraylist vaccination
     * @param thisVaccine parameter of this method
     * it will contain the object of the arraylist
     */
    public void setVaccinations(vaccination thisVaccine)
    {
        this.vaccinations.add(thisVaccine);
    }

    /**
     * getter method for vaccination object in
     * arraylist vaccinations
     * @return the vaccination object of the
     * vaccination arraylist
     */
    public ArrayList<vaccination> getVaccinations()
    {
        return vaccinations;
    }

    /**
     * getter method for batch number of the vaccine
     * @return the batch number of the vaccine will
     * be returned
     */
    public int getBatchNo()
    {
        return batchNo;
    }

    /**
     * getter method for expiry date of the vaccine
     * @return the expiry date of the vaccine will
     * be returned
     */
    public String getExpiryDate()
    {
        return expiryDate;
    }

    /**
     * getter method for quantity vaccine that available
     * @return the available quantity of vaccine will
     * be returned
     */
    public int getQuantityAvailable()
    {
        return quantityAvailable;
    }

    /**
     * getter method for quantity vaccine that administered
     * @return the administered quantity of vaccine will
     * be returned
```

```
     */
    public int getQuantityAdministered()
    {
        return quantityAdministered;
    }

    //toString

    /**
     * toString method for class batch
     * @return it will return the batch number of the vaccine,
     * expiry date, quantity vaccine that available,
     * and quantity vaccine that has administered
     */
    @Override
    public String toString()
    {
        return "batch{" +
                "batchNo=" + batchNo +
                ", expiryDate='" + expiryDate +
                ", quantityAvailable=" + quantityAvailable +
                ", quantityAdministered=" + quantityAdministered +
                '}';
    }
}
```

vaccination.java

```
/**
 * class vaccination is a class that will
 * be a class for vaccination data
 *
 * @author I Ketut Mahendra
 */
public class vaccination
{
    private int vaccinationID;
    private String appointmentDate;
    private String status;
    private String remarks;

    /**
     * constructor for class vaccination
     * @param vaccinationID is the parameter for
     *                      the id of vaccination
     * @param appointmentDate is the parameter for
     *                        the appointment date
     *                        inputted by the user
     *                        for vaccination
     * @param status is the parameter for vaccination
     *               status
     * @param remarks is the parameter for the remarks
     *                of the vaccination progress/condition
     */
    public vaccination(int vaccinationID, String appointmentDate,
                       String status, String remarks)
    {
        this.vaccinationID = vaccinationID;
        this.appointmentDate = appointmentDate;
        this.status = status;
        this.remarks = remarks;
```

```
    }

    /**
     * setter method for vaccination id
     * @param vaccinationID is the parameter for
     *                      vaccination id
     */
    public void setVaccinationID(int vaccinationID)
    {
        this.vaccinationID = vaccinationID;
    }

    /**
     * setter method for appointment date of vaccination
     * @param appointmentDate is the parameter for
     *                        appointment date for vaccination
     */
    public void setAppointmentDate(String appointmentDate)
    {
        this.appointmentDate = appointmentDate;
    }

    /**
     * setter method for vaccination status
     * @param status is the parameter for vaccination status
     */
    public void setStatus(String status)
    {
        this.status = status;
    }

    /**
     * setter method for vaccination remarks
     * @param remarks is the parameter for vaccination remarks
     */
    public void setRemarks(String remarks)
    {
        this.remarks = remarks;
    }

    /**
     * getter method for vaccination id
     * @return the vaccination id will be returned
     */
    public int getVaccinationID()
    {
        return vaccinationID;
    }

    /**
     * getter method for vaccination appointment date
     * @return the appointment date that inputted by patient will be
returned
     */
    public String getAppointmentDate()
    {
        return appointmentDate;
    }

    /**
     * getter method for vaccination status
```

```
     * @return the vaccination status will be returned
     */
    public String getStatus()
    {
        return status;
    }

    /**
     * getter method for vaccination remarks
     * @return the vaccination remarks will be returned
     */
    public String getRemarks()
    {
        return remarks;
    }

    /**
     * toString method for vaccination class
     * @return the vaccination id, appointment date, status,
     * and remarks will be returned
     */
    @Override
    public String toString()
    {
        return "vaccination{" +
                "vaccinationID=" + vaccinationID +
                ", appointmentDate='" + appointmentDate + '\'' +
                ", status='" + status + '\'' +
                ", remarks='" + remarks + '\'' +
                '}';
    }
}
```

PCVSGUI.java

```
/**
 * This is the main class of Private Covid-19
 * Vaccination System using Graphical User Interface(GUI)
 */

/**
 * @author I Ketut Mahendra
 * Student ID: E1900340
 * IDE: Eclipse IDE for Java Developers
 * (includes Incubating components)
 * Version: 2021-12 (4.22.0)
 * Java runtime version: 16.0.1+9-24
 * Java version: 16.0.1
 */

import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;
import javax.swing.SwingConstants;
import java.awt.SystemColor;
import java.awt.Font;
import java.awt.Color;
```

```
import javax.swing.JTextField;
import javax.swing.JPasswordField;
import javax.swing.JButton;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

55blic class PCVSGUI extends JFrame
implements ActionListener{
	//create PCVS Object
	private PCVS controller;
	JFrame frmLogin;
	private JPanel loginPanel;
	private JTextField usernameTextField;
	28ivate JPasswordField passwordField;
	private final JPanel panel = new JPanel();

	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		PCVS controller = new PCVS();
		JFrame pcvsGUI = new PCVSGUI(controller);
		pcvsGUI.setVisible(true);
	}

	/**
	 * Create the frame.
	 */
	public PCVSGUI(PCVS pcvs)
	{
		//initialize PCVS Object
		controller = pcvs;

		setUndecorated(true);
		10tTitle("PRIVATE COV-19 VACCINATION SYSTEM");
		setIconImage(Toolkit.getDefaultToolkit().
				getImage(PCVSGUI.class.getResource
					("/img/pcvsIcon.png")));
		setResizable(false);
		setBounds(200, 150, 1053, 532);
		loginPanel 13new JPanel();
		loginPanel.setBorder(new EmptyBorder(5, 5, 5, 5));
		setContentPane(loginPanel);
		loginPanel.setLayout(null);

		JLabel welcomeToLbl = new JLabel("Welcome to");
		welcomeToLbl.setFont(new Font
				("Montserrat", Font.BOLD, 48));
		welcomeToLbl.setForeground(SystemColor.text);
		welcomeToLbl.setHorizontalAlignment
		(SwingConstants.LEFT);
		welcomeToLbl.setBounds(222, 193, 362, 68);
		loginPanel.add(welcomeToLbl);

		JLabel privateC19Lbl = new JLabel
				("Private COVID-19");
		privateC19Lbl.setHorizontalAlignment
		(SwingConstants.LEFT);
```

```
privateC19Lbl.setForeground(Color.WHITE);
privateC19Lbl.setFont(new Font
            ("Montserrat", Font.PLAIN, 36));
privateC19Lbl.setBounds(222, 250, 362, 50);
loginPanel.add(privateC19Lbl);

JLabel VaccinationSystemLbl = new JLabel
            ("Vaccination System");
VaccinationSystemLbl.setHorizontalAlignment
(SwingConstants.LEFT);
VaccinationSystemLbl.setForeground(Color.WHITE);
VaccinationSystemLbl.setFont(new Font
            ("Montserrat", Font.PLAIN, 24));
VaccinationSystemLbl.setBounds
(222, 289, 362, 50);
loginPanel.add(VaccinationSystemLbl);

JLabel logo = new JLabel("");
logo.setHorizontalAlignment
(SwingConstants.CENTER);
logo.setIcon(new ImageIcon(PCVSGUI.
            class.getResource("/img/pcvsIcon.png")));
logo.setBounds(37, 193, 169, 169);
loginPanel.add(logo);

JLabel loginLbl = new JLabel("Login.");
loginLbl.setHorizontalAlignment
(SwingConstants.RIGHT);
loginLbl.setFont(new Font
            ("Montserrat", Font.BOLD, 38));
loginLbl.setBounds(631, 53, 376, 54);
loginPanel.add(loginLbl);

JLabel usernameLabel = new JLabel("Username");
usernameLabel.setFont(new Font
            ("Montserrat", Font.PLAIN, 14));
usernameLabel.setHorizontalAlignment
(SwingConstants.RIGHT);
usernameLabel.setBounds(701, 117, 306, 33);
loginPanel.add(usernameLabel);

usernameTextField = new JTextField();
usernameTextField.setHorizontalAlignment
(SwingConstants.RIGHT);
usernameTextField.setFont(new Font
            ("Montserrat", Font.PLAIN, 18));
usernameTextField.setBounds
(631, 147, 376, 41);
loginPanel.add(usernameTextField);
usernameTextField.setColumns(10);

JLabel passwordLabel = new JLabel("Password");
passwordLabel.setHorizontalAlignment
(SwingConstants.RIGHT);
passwordLabel.setFont(new Font
            ("Montserrat", Font.PLAIN, 14));
passwordLabel.setBounds(701, 198, 306, 33);
loginPanel.add(passwordLabel);

passwordField = new JPasswordField();
passwordField.setHorizontalAlignment
```

```
            SwingConstants.RIGHT);
            passwordField.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            passwordField.setBounds(631, 224, 376, 41);
            loginPanel.add(passwordField);

            final JLabel lblUsernameNotFound =
                        new JLabel("Username not found");
            lblUsernameNotFound.setVisible(false);
            lblUsernameNotFound.setForeground(Color.RED);
            lblUsernameNotFound.setHorizontalAlignment
            (SwingConstants.LEFT);
            lblUsernameNotFound.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            lblUsernameNotFound.setBounds
            (631, 117, 306, 33);
            loginPanel.add(lblUsernameNotFound);

            final JLabel lblPasswordNotFound =
                        new JLabel("Password not found");
            lblPasswordNotFound.setVisible(false);
            lblPasswordNotFound.setHorizontalAlignment
            (SwingConstants.LEFT);
            lblPasswordNotFound.setForeground
            (Color.RED);
            lblPasswordNotFound.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            lblPasswordNotFound.setBounds(631, 198, 306, 33);
            loginPanel.add(lblPasswordNotFound);

      /**
       * login as administrator trying
       */
            JButton loginAdminBtn = new JButton("Login as Admin");
            loginAdminBtn.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        if ((controller.loginCheckAdmin
                                    (usernameTextField.getText(),
                                                passwordField.getText()))){
                              adminDashboard adminPage = new
                                          adminDashboard(controller,
                                                      usernameTextField.
                                                      getText());
                              adminPage.setVisible(true);
                              dispose();
                        }
                        else if(usernameTextField.getText().
                                    isEmpty()) {
                              JOptionPane.showMessageDialog
                              (usernameTextField, "Username is Blank");
                        }
                        else if(passwordField.getText().isEmpty()) {
                              JOptionPane.showMessageDialog(passwordField,
                                          "Password is Blank");
                        }
                        else {
                              lblUsernameNotFound.setVisible(true);
                              lblPasswordNotFound.setVisible(true);
                        }
                  }
```

```
            });
            loginAdminBtn.setFont(new Font("Montserrat",
                        Font.PLAIN, 16));
            loginAdminBtn.setBackground(new Color(100, 149, 237));
            loginAdminBtn.setForeground(new Color(230, 230, 250));
            loginAdminBtn.setBounds(631, 291, 185, 53);
            loginPanel.add(loginAdminBtn);

            JButton loginPatientBtn = new JButton
                        ("Login as Patient");
            loginPatientBtn.addMouseListener
            (new MouseAdapter() {
                  @Override
                  public void mouseClicked
                  (MouseEvent e) {

                        //if the text in user name text field and password
                        //in the password field is founded in arraylist,
                        //the method from object PCVS that called below
                        //will return true
                        if(controller.loginCheckPatient
                                    (usernameTextField.getText(),
                                                passwordField.getText())) {
                              patientDashboard patient = new
                                          patientDashboard(controller,
                                                      usernameTextField.
                                                      getText());
                              patient.setVisible(true);
                              dispose();
                        }
                        //if the user name field is empty,
                        //it will displaying a message dialog
                        else if(usernameTextField.getText().isEmpty()) {
                              JOptionPane.showMessageDialog
                              (usernameTextField, "Username is Blank");
                              lblUsernameNotFound.setVisible(false);
                              lblPasswordNotFound.setVisible(false);
                        }
                        //if the password field is empty
                        //it will displaying a message dialog
                        else if(passwordField.getText().isEmpty()) {
                              JOptionPane.showMessageDialog
                              (passwordField, "Password is Blank");
                              lblUsernameNotFound.setVisible(false);
                              lblPasswordNotFound.setVisible(false);
                        }
                        //if the data isn't found a label will appear
                        else {
                              lblUsernameNotFound.setVisible(true);
                              lblPasswordNotFound.setVisible(true);
                        }
                  }
            });
            loginPatientBtn.setForeground(new Color
                        (230, 230, 250));
            loginPatientBtn.setFont(new Font
                        ("Montserrat", Font.PLAIN, 16));
            loginPatientBtn.setBackground
            (new Color(30, 144, 255));
            loginPatientBtn.setBounds
            (814, 291, 193, 53);
```

```
            loginPanel.add(loginPatientBtn);

            JLabel registerLabel = new JLabel("New here?");
            registerLabel.setBackground(new Color
                        (32, 144, 255));
            registerLabel.setHorizontalAlignment
            (SwingConstants.CENTER);
            registerLabel.setForeground(new Color
                        (30, 144, 255));
            registerLabel.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            registerLabel.setBounds(631, 376, 185, 33);
            loginPanel.add(registerLabel);

      /**
       * registering trying
       */
            JButton registerBtn = new JButton("Create Account");
            registerBtn.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        signUp reg = new signUp(controller);
                        reg.setVisible(true);
                        dispose();
                  }
            });
            registerBtn.setForeground(new Color(230, 230, 250));
            registerBtn.setFont(new Font
                        ("Montserrat", Font.PLAIN, 16));
            registerBtn.setBackground(new Color
                        (32, 178, 170));
            registerBtn.setBounds
            (631, 410, 185, 53);
            loginPanel.add(registerBtn);
            panel.setBounds(0, 0, 552, 547);
            loginPanel.add(panel);
            panel.setLayout(null);

            JLabel devInfoLabel = new JLabel
                        ("Developed by:\r\nI Ketut Mahendra "
                        + "(E1900340) \r\nfor BIT205-Advance OOP\r\n "
                        + "Semester September 2021");
            devInfoLabel.setBounds(10, 484, 457, 33);
            panel.add(devInfoLabel);
            devInfoLabel.setForeground(new Color(255, 255, 255));
            devInfoLabel.setFont(new Font("Montserrat", Font.PLAIN, 9));

            JLabel Background = new JLabel("");
            Background.setIcon(new ImageIcon(PCVSGUI.
                        class.getResource("/img/signInBg.png")));
            Background.setBounds(0, 0, 552, 534);
            panel.add(Background);

            /**
             * action listener to close the panel
             */
            JLabel Close = new JLabel("\u2022");
            Close.setVerticalTextPosition(SwingConstants.TOP);
            Close.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
```

```
                        int closing = JOptionPane.
                                    showConfirmDialog(getParent(),
                                    "Are you sure to exit this program?",
                                    "Program Close",
                                    JOptionPane.YES_NO_CANCEL_OPTION,
                                    JOptionPane.CLOSED_OPTION);
                        if(closing==JOptionPane.YES_OPTION) {
                              System.exit(0);
                        }

                  }
            });
            Close.setFont(new Font("Montserrat", Font.BOLD, 72));
            Close.setForeground(Color.RED);
            Close.setHorizontalAlignment(SwingConstants.RIGHT);
            Close.setBounds(1017, 0, 26, 43);
            loginPanel.add(Close);

      /**
       * Action Listener to minimize the panel
       */
            JLabel Minimize = new JLabel("\u2022");
            Minimize.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        setState(JFrame.ICONIFIED);
                  }
            });
            Minimize.setVerticalTextPosition
            (SwingConstants.TOP);
            Minimize.setHorizontalAlignment
            (SwingConstants.RIGHT);
            Minimize.setForeground(Color.ORANGE);
            Minimize.setFont(new Font("Montserrat",
                        Font.BOLD, 72));
            Minimize.setBounds(981, 0, 26, 43);
            loginPanel.add(Minimize);

            JLabel lblHaveAData = new JLabel
                        ("Have a Data?");
            lblHaveAData.setHorizontalAlignment
            (SwingConstants.CENTER);
            lblHaveAData.setForeground(new Color
                        (30, 144, 255));
            lblHaveAData.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            lblHaveAData.setBackground(new Color
                        (30, 144, 255));
            lblHaveAData.setBounds(814, 376, 193, 33);
            loginPanel.add(lblHaveAData);

            JButton btnLoadData = new JButton
                        ("Load Data");
            btnLoadData.setForeground(new Color
                        (30, 230, 250));
            btnLoadData.setFont(new Font
                        ("Montserrat", Font.PLAIN, 16));
            btnLoadData.setBackground(new Color
                        (50, 205, 50));
            btnLoadData.setBounds(814, 410, 193, 53);
            loginPanel.add(btnLoadData);
```

```
	}

	public String getUsernameLogin() {
		return usernameTextField.getText();
	}
	@Override
	public void actionPerformed(ActionEvent e) {
		// TODO Auto-generated method stub

	}
}
```

signUp.java

```
import java.awt.BorderLayout;
import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import java.awt.Color;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.ComboBoxModel;
import javax.swing.ImageIcon;
import java.awt.Font;
import javax.swing.SwingConstants;
import javax.swing.JTextField;
import javax.swing.JComboBox;
import javax.swing.JPasswordField;
import javax.swing.JButton;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Toolkit;
import java.beans.PropertyChangeListener;
import java.beans.PropertyChangeEvent;
import javax.swing.JRadioButton;

//public class signUp extends JFrame {
public class signUp extends JFrame {
	private PCVS controller;
	private JPanel signUpPanel;
	private JTextField usernameField;
	private JPasswordField passwordField;
	private JPasswordField verifyPasswordField;
	private JTextField nameField;
	private JTextField emailField;
	private JTextField icPassFields;
	/**
	 * Create the frame.
	 * @param pcvs
	 */
	public signUp(PCVS pcvs) {
		//initialize controller object
		controller = pcvs;
```

```
        setTitle("Sign Up Form");
        setIconImage(Toolkit.getDefaultToolkit().
                    getImage(signUp.class.getResource
                                ("/img/pcvsIcon.png")));
        setUndecorated(true);
        setResizable(false);
        setBackground(new Color(255, 255, 255));
        setDefaultCloseOperation
        (JFrame.EXIT_ON_CLOSE);
        setBounds(200, 100, 1117, 562);
        signUpPanel = new JPanel();
        signUpPanel.setBorder(new EmptyBorder
                    (5, 5, 5, 5));
        setContentPane(signUpPanel);
        signUpPanel.setLayout(null);

        JPanel panelBackground = new JPanel();
        panelBackground.setBounds(0, 0, 488, 562);
        signUpPanel.add(panelBackground);
        panelBackground.setLayout(null);

        JLabel SignUp = new JLabel("SignUp.");
        SignUp.setForeground(new Color
                    (255, 255, 255));
        SignUp.setFont(new Font
                    ("Montserrat", Font.BOLD, 72));
        SignUp.setBounds(50, 258, 381, 160);
        panelBackground.add(SignUp);

        JLabel pcvsIcon = new JLabel("");
        pcvsIcon.setIcon(new ImageIcon
                    (signUp.class.getResource
                                ("/img/pcvsIcon.png")));
        pcvsIcon.setBounds(21, 147, 169, 169);
        panelBackground.add(pcvsIcon);

        JLabel background = new JLabel("");
        background.setIcon(new ImageIcon
                    (signUp.class.getResource
                                ("/img/signInBg.png")));
        background.setBounds(0, 0, 488, 562);
        panelBackground.add(background);

        JLabel Close = new JLabel("\u2022");

    /**
     * exit button
     */
        Close.addMouseListener(new MouseAdapter()
        {
            @Override
            public void mouseClicked(MouseEvent e)
            {
                int closing = JOptionPane.
                            showConfirmDialog
                            (getParent(),
                            "Are you sure to exit "
                            + "this process?",
                            "Program Close",
                            JOptionPane.YES_NO_OPTION,
```

```
                                    JOptionPane.CLOSED_OPTION);
                        if(closing==JOptionPane.YES_OPTION) {
                              PCVSGUI loginBack =
                                          new PCVSGUI(controller);
                              loginBack.setVisible(true);
                              dispose();
                        }
                  }
            });
            Close.setVerticalTextPosition
            (SwingConstants.TOP);
            Close.setHorizontalAlignment
            (SwingConstants.RIGHT);
            Close.setForeground(Color.RED);
            Close.setFont(new Font
                        ("Montserrat", Font.BOLD, 72));
            Close.setBounds(1081, 0, 26, 43);
            signUpPanel.add(Close);


      /**
       * minimize
       */
            JLabel Minimize = new JLabel("\u2022");
            Minimize.addMouseListener
            (new MouseAdapter()
            {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        setState(JFrame.ICONIFIED);
                  }
            });
            Minimize.setVerticalTextPosition
            (SwingConstants.TOP);
            Minimize.setHorizontalAlignment
            (SwingConstants.RIGHT);
            Minimize.setForeground
            (Color.ORANGE);
            Minimize.setFont(new Font
                        ("Montserrat", Font.BOLD, 72));
            Minimize.setBounds(1045, 0, 26, 43);
            signUpPanel.add(Minimize);

            JLabel usernameLbl = new JLabel("Username");
            usernameLbl.setHorizontalAlignment
            (SwingConstants.RIGHT);
            usernameLbl.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            usernameLbl.setBounds(630, 82, 218, 13);
            signUpPanel.add(usernameLbl);
            
            usernameField = new JTextField();
            usernameField.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            usernameField.setBounds(520, 98, 328, 48);
            signUpPanel.add(usernameField);
            usernameField.setColumns(10);

            JLabel rolesLbl = new JLabel("Roles");
            rolesLbl.setHorizontalAlignment
            (SwingConstants.RIGHT);
```

```
            rolesLbl.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            rolesLbl.setBounds(858, 80, 218, 13);
            signUpPanel.add(rolesLbl);

            String[] roles = {"Administrator",
                        "Patient"};
            final JComboBox comboBoxRoles =
                        new JComboBox(roles);
            comboBoxRoles.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            comboBoxRoles.setBounds(858, 98, 219, 48);
            signUpPanel.add(comboBoxRoles);

            JLabel passwordLbl = new JLabel
                        ("Password");
            passwordLbl.setHorizontalAlignment
            (SwingConstants.RIGHT);
            passwordLbl.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            passwordLbl.setBounds(630, 172, 163, 13);
            signUpPanel.add(passwordLbl);

            passwordField = new JPasswordField();
            passwordField.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            passwordField.setBounds(520, 190, 273, 46);
            signUpPanel.add(passwordField);

            JLabel verifyPassLbl = new JLabel
                        ("Verify your Password");
            verifyPassLbl.setHorizontalAlignment
            (SwingConstants.RIGHT);
            verifyPassLbl.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            verifyPassLbl.setBounds(913, 172, 163, 13);
            signUpPanel.add(verifyPassLbl);

            verifyPasswordField = new JPasswordField();
            verifyPasswordField.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            verifyPasswordField.setBounds(804, 190, 273, 46);
            signUpPanel.add(verifyPasswordField);

            JLabel nameLbl = new JLabel("Name");
            nameLbl.setHorizontalAlignment
            (SwingConstants.RIGHT);
            nameLbl.setFont(new Font
                        ("Montserrat", Font.PLAIN, 14));
            nameLbl.setBounds(630, 266, 159, 13);
            signUpPanel.add(nameLbl);

            nameField = new JTextField();
            nameField.setFont(new Font
                        ("Montserrat", Font.PLAIN, 18));
            nameField.setColumns(10);
            nameField.setBounds(520, 282, 269, 48);
            signUpPanel.add(nameField);

            JLabel emailLabel = new JLabel("E-Mail");
            emailLabel.setHorizontalAlignment
```

```
(SwingConstants.RIGHT);
emailLabel.setFont(new Font
            ("Montserrat", Font.PLAIN, 14));
emailLabel.setBounds(913, 266, 163, 13);
signUpPanel.add(emailLabel);

emailField 9 new JTextField();
emailField.setFont(new Font
            ("Montserrat", Font.PLAIN, 18));
emailField.setColumns(10);
emailField.setBounds(803, 282, 273, 48);
signUpPanel.add(emailField);

final JLabel ICPassLbl = new JLabel
            ("IC/Passport Number");
ICPassLbl.12tVisible(false);
ICPassLbl.setHorizontalAlignment
(SwingConstants.RIGHT);
ICPassLbl.setFont(new Font
            ("Montserrat", Font.PLAIN, 14));
ICPassLbl.setBounds(914, 366, 163, 13);
signUpPanel.add(ICPassLbl);

icPassFields = new JTextField();
icPassFields.9etVisible(false);
icPassFields.setFont(new Font
            ("Montserrat", Font.PLAIN, 18));
icPassFields.setColumns(10);
icPassFields.setBounds(520, 382, 557, 48);
signUpPanel.add(icPassFields);
             39
comboBoxRoles.addActionListener(new
            ActionListener() {
      public void actionPerformed
      (ActionEvent e) {
            if(comboBoxRoles.getSelectedItem().
                  toString() == "Administrator")
            {
                  icPassFields.setVisible(false);
                  ICPassLbl.setVisible(false);
            }else if(comboBoxRoles.getSelectedItem().
                  toString() == "Patient") {
                  icPassFields.setVisible(true);
                  ICPassLbl.setVisible(true);
            }
      }
});

final JLabel healthCareLbl = new JLabel
            ("12alth Care");
healthCareLbl.setHorizontalAlignment
(SwingConstants.RIGHT);
healthCareLbl.setFont(new Font
            ("Montserrat", Font.PLAIN, 14));
healthCareLbl.setBounds(914, 367, 163, 13);
signUpPanel.add(healthCareLbl);


String[] healthCare = {"FMC General Hospital, "
            + "Address: Jalan Arjuna 18, Gianyar, Bali",
            "SMCDenpasar, Address: Jalan P.B. Sudirman, "
```

```
                    + "Denpasar, Bali"};
            final JComboBox comboBoxHealthCare =
                    new JCo43oBox(healthCare);
            comboBoxHealthCare.setFont(new Font
                    ("Montserrat", Font.PLAIN, 18));
            comboBoxHealthCare.setBounds(520, 382, 557, 48);
            signUpPanel.add(comboBoxHealthCare);
                        39
            comboBoxRoles.addActionListener
            (new ActionListener()
            {
                  public void actionPerformed
                  (ActionEvent e) {
                        if(comboBoxRoles.getSelectedItem().
                                    toString()=="Administrator") {
                              comboBoxHealthCare.setVisible(true);
                              healthCareLbl.setVisible(true);
                        }else if(comboBoxRoles.getSelectedItem().
                                    toString()=="Patient") {
                              comboBoxHealthCare.setVisible(false);
                              healthCareLbl.setVisible(false);
                        }
                  }
            });

            JButton registe37utton = new JButton("Register");
            registerButton.addActionListener(new ActionListener() {
                  public void actionPerformed(ActionEvent e) {
                        String usernameInput = usernameField.getText();
                        String passwordInput = passwordField.getText();
                        String emailInput = emailField.getText();
                        String fullNameInput = nameField.getText();
                        String ICPASSInput = icPassFields.getText();
                        String staffID = controller.generateID

      (comboBoxHealthCare.getSelectedIndex()+1);
                        if(controller.inspectSignInAdmin(usernameInput)) {
                              JOptionPane.showMessageDialog(passwordField,
                                          "Your Username has been taken "
                                          + ""
                                          + "before");
                              usernameField.setText("");
                        }                          65
                        else if (usernameField.getText().isEmpty()||
                                    passwordField.getText().isEmpty()||
                                    emailField.getText().isEmpty()||
                                    nameField.getText().isEmpty()) {
                              JOptionPane.showMessageDialog
                              (passwordField,
                                "please fill the blank field");
                        }
                        else if (!(passwordField.getText().equals
                                    (verifyPasswordField.getText()))) {
                              JOptionPane.showMessageDialog(passwordField,
                                          "Password did not match,
                                          try again");
                              passwordField.setText("");
                              verifyPasswordField.setText("");
                        }
                        else if((comboBoxRoles.getSelectedItem().
                                    toString()=="Administrator")) {
```

```
                        administrator newAdmin = new administrator
                                    (usernameInput, passwordInput,
                                                emailInput,
                                                fullNameInput,
                                                staffID);
                        controller.setAdminData(newAdmin);
                        healthCareCentre centre = controller.
                                    getHealthcareCentre
                                    (comboBoxHealthCare.
                                                getSelectedIndex());
                        centre.setAdminHealthCare(newAdmin);
                        JOptionPane.showMessageDialog(passwordField,
                                    "You are succesfully registered "
                                    + "with: "+
                                    "\nUsername: "+
                                    usernameInput+
                                    "\nName: "+
                                    fullNameInput+
                                    "\nHealth Care Centre: "+
                                                comboBoxHealthCare.
                                                getSelectedItem()+
                                                "\n\nYour Staff ID: "
                                                +
                                                staffID);
                        PCVSGUI gui = new PCVSGUI(controller);
                        gui.setVisible(true);
                        dispose();
                  }
                  else if(comboBoxRoles.getSelectedItem().
                              toString()=="Patient") {
                        patient newPatient = new patient
                                    (usernameInput,
                                                passwordInput,
                                                emailInput,
                                                fullNameInput,
                                                ICPASSInput);
                        PCVSGUI gui = new PCVSGUI(controller);
                        JOptionPane.showMessageDialog(passwordField,
                                    "You are succesfully registered "
                                    + "with: "+"\nUsername: "+
                                                usernameInput+
                                                "\nName: "+
                                                fullNameInput+
                                                "\nIC/Passport: "+
                                                ICPASSInput);
                        controller.setPatientsData(newPatient);
                        gui.setVisible(true);
                        dispose();
                  }
                  else {
                        JOptionPane.showMessageDialog(passwordField,
                                    "all data is Blank");
                  }
            }
      });
      registerButton.setForeground(new Color(255, 255, 255));
      registerButton.setBackground(new Color(154, 205, 50));
      registerButton.setFont(new Font("Montserrat",
                  Font.PLAIN, 18));
      registerButton.setBounds(520, 480, 273, 48);
      signUpPanel.add(registerButton);
```

```
            JButton btnClearData = new JButton("Clear Data");
            btnClearData.addMouseListener(new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        usernameField.setText("");
                        passwordField.setText("");
                        verifyPasswordField.setText("");
                        emailField.setText("");
                        nameField.setText("");

      if(comboBoxRoles.getSelectedItem().toString()=="Patient")
                              icPassFields.setText("");
                  }
            });
            btnClearData.setForeground(new Color(255, 255, 255));
            btnClearData.setFont(new Font("Montserrat", Font.PLAIN, 18));
            btnClearData.setBackground(new Color(255, 0, 0));
            btnClearData.setBounds(804, 480, 273, 48);
            signUpPanel.add(btnClearData);
      }
}
```

adminDashboard.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.EmptyBorder;

import java.io.*;
import java.security.PublicKey;

import javax.swing.border.CompoundBorder;
import javax.swing.border.LineBorder;
import javax.swing.border.TitledBorder;
import javax.swing.event.MenuKeyListener;
import javax.swing.event.MenuKeyEvent;
import javax.swing.border.BevelBorder;
import javax.swing.table.DefaultTableModel;

public class adminDashboard
extends JFrame implements ActionListener
{
      private PCVS controller;
      private JMenuItem itmNew, itmOpen,
      itmSave, itmSaveAs, itmExit;
      private JMenuItem itmAddNewVaccineBatch,
      itmViewVaccineBatchInfo;
      private JMenuItem itmConfVaccAppointment,
      itmVacAdministered;
      private JMenuItem itmViewUser,
      itmViewVaccAppointment;
      private JMenuItem itmAboutPCVS;
      private JLayeredPane layeredPaneVaccine;
      private JPanel panelViewVaccineBatch;
      private JPanel panelRecordNewVaccineBatch;
      private JPanel panelConfirmOrRejectAppointment;
      private JPanel panelHomePage;
```

```
	private JLabel lblViewVB;
	private JLabel lblRecordNewVB;
	private JLabel lblVaccineID;
	private JTextField manufacturerTextField;
	private JTextField vaccineNameTextField;
	private JLabel lblBatchNumber;
	private JTextField batchNumTextField;
	private JLabel lblExpiryDateExpiry;
	private JLabel lblQuantityDose;
	private JTextField quantityOfDoseTextField;
	private JButton btnDeleteData;
	private JPanel panelViewVaccinationAdministered;
	private JPanel panelListingUser;
	private JPanel panelListingVaccinationAppointment;
	private JLabel lblVaccinationAdministered;
	private JLabel lblUserData;
	private JLabel lblVaccinationAppointmentData;
	private JLabel logo;
	private JLabel lblNewLabel;
	private JLabel lblProvidedByMinistry;
	private JLabel lblAuthorIKetut;
	private JLabel lblConfirmOrReject;
	private JTable tableViewVB;
	private JScrollPane scrollTableViewVB;
	private JTable tableConfirmReject;
	private JLabel lblTitleUpdateStatus;
	private JLabel lblVaccinationID;
	private JTextField vaccinationIdField;
	private JButton btnUpdateStatusVaccination;
	private JTable tableVaccinationAdministered;
	private JTextField textFieldVaccIDAdministered;
	private JTable tableVaccinationAppointmentData;
	private JScrollPane
	scrollPaneVaccinationAppointmentData;
	private JComboBox comboBoxListSort;
	private JLabel lblSort;
	private JButton btnSort;
	private JTable tableUserData;
	private JScrollPane scrollPaneUserData;

	/**
	 * Create the frame.
	 */
	public adminDashboard(PCVS pcvs, String username)
	{
		controller = pcvs;
		setVisible(true);
		setResizable(false);
		addWindowListener(new WindowAdapter() {
			@Override
			public void windowClosing(WindowEvent e) {
				int closing = JOptionPane.showConfirmDialog
					(getParent(),
					"Are you sure to exit "
					+ "this program?\n"
					+ "Make sure you has Saving "
					+ "this Project",
					"Program Close",
					JOptionPane.YES_NO_OPTION,
					JOptionPane.CLOSED_OPTION);
				if(closing==JOptionPane.YES_OPTION) {
```

```
                        PCVSGUI loginBack = new PCVSGUI
                                    (controller);
                        loginBack.setVisible(true);
                        dispose();
                    }
                }
            });

            10tTitle("PCVS - Admin");
            setIconImage(Toolkit.getDefaultToolkit().
                        getImage(adminDashboard.class.
                                    getResource
                                    ("/img/pcvsIcon.png")));
            setBounds(100, 50, 1240, 720);

            /**
             * Menu Bar
             */
            JMenuBar menuListing = new JMenuBar();
            menuListing.setBorderPaint56(false);
            menuListing.setBackground(new Color
                        (255, 255, 255));
            setJMenuBar(menuListing);
            JMenu fileMenu = new JMenu("File");
            fileMenu.setBackground(new Color
                      14 (255, 255, 255));
            fileMenu.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));
            menuListing.add(fileMenu);

            //menu new
            itmNew = new JMenuItem("New");
            fileMen3.add(itmNew);
            itmNew.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));

            //menu open
            itmOpen = new JMenuItem("Open");
            fileMenu3dd(itmOpen);
            itmOpen.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));

            //menu save
            itmSave = new JMenuItem("Save");
            fileMenu3dd(itmSave);
            itmSave.setFont(new Font("Segoe UI",
                        Font.PLAIN, 14));

            //menu save as
            itmSaveAs = new JMenuItem("Save As");
            fileMenu.a3(itmSaveAs);
            itmSaveAs.setFont(new Font("Segoe UI",
                        Font.PLAIN, 14));

            //menu logout
            itmExit 54new JMenuItem("Logout");
            itmExit.addActionListener(new
                        ActionListener()
            {
                  public void actionPerformed(ActionEvent e)
                  {
```

```
                int closing = JOptionPane.
                            showConfirmDialog
                            (getParent(),
                            "Logout from PCVS?",
                            Logout",
                            JOptionPane.YES_NO_OPTION,
                            JOptionPane.CLOSED_OPTION);
                if(closing==JOptionPane.YES_OPTION)
                {
                      PCVSGUI loginBack = new
                                  PCVSGUI(controller);
                      loginBack.setVisible(true);
                      dispose();
                }
          }
});
fileMenu.add(itmExit);
itmExit.setFont(new Font("Segoe UI",
                Font.PLAIN, 14));

/**
 * Menu Bar Vaccine
 */
JMenu vaccineMenu = new JMenu("Vaccine");
vaccineMenu.setBackground(new Color
                (255, 255, 255));
vaccineMenu.setFont(new Font("Segoe UI",
                Font.PLAIN, 14));
menuListing.add(vaccineMenu);

//menu record new vaccine batch
itmAddNewVaccineBatch = new JMenuItem
                ("Add New Vaccine Batch");
itmAddNewVaccineBatch.addActionListener
(new ActionListener() {
      @Override
      public void actionPerformed(ActionEvent e)
      {
            panelRecordNewVaccineBatch.setVisible
            (true);
            panelViewVaccineBatch.setVisible
            (false);
            panelConfirmOrRejectAppointment.
            setVisible(false);
            panelViewVaccinationAdministered.
            setVisible(false);
            panelListingUser.setVisible(false);
            panelListingVaccinationAppointment.
            setVisible(false);
      }
});
vaccineMenu.add(itmAddNewVaccineBatch);
itmAddNewVaccineBatch.setFont(new Font
                ("Segoe UI", Font.PLAIN, 14));

//menu view vaccine batch information
itmViewVaccineBatchInfo = new JMenuItem
                ("View Vaccine Batch Information");
itmViewVaccineBatchInfo.addActionListener
(new ActionListener() {
      public void actionPerformed(ActionEvent e)
```

```
                {
                        panelRecordNewVaccineBatch.setVisible
                        (false);
                        panelViewVaccineBatch.setVisible
                        (true);
                        panelConfirmOrRejectAppointment.
                        setVisible(false);
                        panelViewVaccinationAdministered.
                        setVisible(false);
                        panelListingUser.setVisible(false);
                        panelListingVaccinationAppointment.
                        setVisible(false);
                }
        });
        vaccineMenu.add(itmViewVaccineBatchInfo);
        itmViewVaccineBatchInfo.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));

        /**
         * Menu Bar Vaccination
         */
        JMenu vaccinationMenu = new JMenu
                        ("Vaccination");
        vaccinationMenu.setBackground(new Color
                        (255, 255, 255));
        vaccinationMenu.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));
        menuListing.add(vaccinationMenu);

        //menu confirm or reject vaccination appointment
        itmConfVaccAppointment = new JMenuItem
                        ("Confirm or Reject Vaccination"
                                        + " Appointment");
        itmConfVaccAppointment.addActionListener
        (new ActionListener()
        {
                public void actionPerformed
                (ActionEvent e) {
                        panelViewVaccineBatch.
                        setVisible(false);
                        panelRecordNewVaccineBatch.
                        setVisible(false);
                        panelConfirmOrRejectAppointment.
                        setVisible(true);
                        panelViewVaccinationAdministered.
                        setVisible(false);
                        panelListingUser.
                        setVisible(false);
                        panelListingVaccinationAppointment.
                        setVisible(false);
                }
        });
        vaccinationMenu.add(itmConfVaccAppointment);
        itmConfVaccAppointment.setFont(new Font
                        ("Segoe UI", Font.PLAIN, 14));

        //menu record vaccination administered
        itmVacAdministered = new JMenuItem
                        ("Record Vaccination Administered");
        itmVacAdministered.addActionListener
        (new ActionListener() {
```

```
            public void actionPerformed(ActionEvent e)
            {
                  panelViewVaccineBatch.
                  setVisible(false);
                  panelRecordNewVaccineBatch.
                  setVisible(false);
                  panelConfirmOrRejectAppointment.
                  setVisible(false);
                  panelViewVaccinationAdministered.
                  setVisible(true);
                  panelListingUser.
                  setVisible(false);
                  panelListingVaccinationAppointment.
                  setVisible(false);
            }
      });
      vaccinationMenu.add(itmVacAdministered);
      itmVacAdministered.setFont(new Font
                  ("Segoe UI", Font.PLAIN, 14));

      /**
       * Menu Bar View
       */
      JMenu viewMenu = new JMenu
                  ("View");
      viewMenu.setBounds(186, -21, 43, 24);
      menuListing.add(viewMenu);
      viewMenu.setBackground(new Color
                  (255, 255, 255));
      viewMenu.setFont(new Font
                  ("Segoe UI", Font.PLAIN, 14));

      //menu view users data
      itmViewUser = new JMenuItem
                  ("View Users Data");
      itmViewUser.addActionListener
      (new ActionListener() {
            public void actionPerformed
            (ActionEvent e)
            {
                  panelViewVaccineBatch.
                  setVisible(false);
                  panelRecordNewVaccineBatch.
                  setVisible(false);
                  panelConfirmOrRejectAppointment.
                  setVisible(false);
                  panelViewVaccinationAdministered.
                  setVisible(false);
                  panelListingUser.
                  setVisible(true);
                  panelListingVaccinationAppointment.
                  setVisible(false);
            }
      });
      viewMenu.add(itmViewUser);
      itmViewUser.setFont(new Font
                  ("Segoe UI", Font.PLAIN, 14));

      //menu view vaccination appointment
      itmViewVaccAppointment = new JMenuItem
                  ("View Vaccination Appointment");
```

```
itmViewVaccAppointment.addActionListener
(new ActionListener() {
      public void actionPerformed(ActionEvent e)
      {
            panelViewVaccineBatch.
            setVisible(false);
            panelRecordNewVaccineBatch.
            setVisible(false);
            panelConfirmOrRejectAppointment.
            setVisible(false);
            panelViewVaccinationAdministered.
            setVisible(false);
            panelListingUser.
            setVisible(false);
            panelListingVaccinationAppointment.
            setVisible(true);
      }
});
viewMenu.add(itmViewVaccAppointment);
itmViewVaccAppointment.setFont(new Font
            ("Segoe UI", Font.PLAIN, 14));

/**
 * Menu Bar About
 */
JMenu aboutMenu = new JMenu("About");
aboutMenu.setBackground(new Color
            (255, 255, 255));
aboutMenu.setFont(new Font
            ("Segoe UI", Font.PLAIN, 14));
menuListing.add(aboutMenu);

//menu about PCVS
itmAboutPCVS = new JMenuItem("About PCVS");
itmAboutPCVS.addActionListener
(new ActionListener() {
      public void actionPerformed
      (ActionEvent e)
      {
            aboutMe me = new aboutMe();
            me.setVisible(true);
      }
});
aboutMenu.add(itmAboutPCVS);
itmAboutPCVS.setFont(new Font
            ("Segoe UI", Font.PLAIN, 14));
getContentPane().setLayout(null);

//panel health care centre
JPanel panelHealthCareName = new JPanel();
panelHealthCareName.setBounds
(10, 10, 893, 51);
getContentPane().add
(panelHealthCareName);
panelHealthCareName.setLayout(null);

//label health care name
//it will find the health care name
//based on the user name who has
//logged in to the system
String hcName = controller.
```

```
                getHealthCareCentreNameForAdminLogin
                (username);
        final JLabel healthCareName = new JLabel
                ("New label");
        //display the health care name using setTex
        healthCareName.setText(hcName);
        healthCareName.setForeground(Color.RED);
        healthCareName.setFont(new Font
                ("Arial Rounded MT Bold",
                                Font.BOLD, 36));
        healthCareName.setBounds(0, 0, 893, 51);
        panelHealthCareName.add(healthCareName);

        //layered pane
        layeredPaneVaccine = new JLayeredPane();
        layeredPaneVaccine.setBorder(new TitledBorder
                (null, "", TitledBorder.LEADING,
                                TitledBorder.TOP, null, null));
        layeredPaneVaccine.setBounds(10, 71, 1206, 576);
        getContentPane().add(layeredPaneVaccine);

        //vaccine ID
        String[]vaccine = {"", "mRNA-1273", "BNT162B2"};

        //array of dates, it will be the data in combo box
        String[]datesExpiry = {"","1","2","3","4","5",
                "6","7","8","9","10","11","12","13",
                "14","15","16","17","18","19","20",
                "21","22","23","24","25","26","27",
                "28","29","30","31"};

        //array of month, it will be the data in
        //combo box
        String[]monthExpiry = {"","January","February",
                "March","April","May","June","July",
                "August","September","October",
                "November","December"};

        //array of year, it will be the data
        //in combo box. i assume that the range
        //of the year was started from 2019 until 2025
        String[]yearExpiry = {"","2019","2020","2021",
                "2022","2023","2024","2025"};

        //panel view vaccination appointment
        panelListingVaccinationAppointment =
                new JPanel();
        layeredPaneVaccine.setLayer
        (panelListingVaccinationAppointment, 3);
        panelListingVaccinationAppointment.
        setVisible(false);

        //panel record vaccination administered
        panelViewVaccinationAdministered = new JPanel();
        layeredPaneVaccine.setLayer
        (panelViewVaccinationAdministered, 2);
        panelViewVaccinationAdministered.
        setVisible(false);

        //panel view vaccinee batch
        panelViewVaccineBatch = new JPanel();
```

```
            layeredPaneVaccine.setLayer
            (panelViewVaccineBatch, 6);
            panelViewVaccineBatch.setBackground
            (Color.WHITE);
            panelViewVaccineBatch.setBorder
            (new TitledBorder(null, "",
                        TitledBorder.LEADING,
                        TitledBorder.TOP, null,
                        null));
            panelViewVaccineBatch.
            setVisible(false);

            //panel cinfirm or rejcet vaccination
            //appointment
            panelConfirmOrRejectAppointment =
                        new JPanel();
            layeredPaneVaccine.setLayer
            (panelConfirmOrRejectAppointment, 1);
            panelConfirmOrRejectAppointment.
            setBackground(Color.WHITE);
            panelConfirmOrRejectAppointment.
            setBorder(new TitledBorder
                        (null, "", TitledBorder.LEADING,
                                    TitledBorder.TOP, null,
                                    null));
            panelConfirmOrRejectAppointment.
            setVisible(false);
            layeredPaneVaccine.setLayout
            (new CardLayout(0, 0));

            //panel homepage admin dashboard
            panelHomePage = new JPanel();
            panelHomePage.setBackground
            (Color.WHITE);
            panelHomePage.setBorder
            (new TitledBorder(null, "", TitledBorder.
                        LEADING, TitledBorder.TOP,
                        null, null));
            layeredPaneVaccine.add(panelHomePage,
                        "name_340232002684400");
            panelHomePage.setLayout(null);

            //icon pcvs
            logo = new JLabel("");
            logo.setIcon(new ImageIcon(adminDashboard.
                        class.getResource
                        ("/img/pcvsIcon.png")));
            logo.setBounds(518, 125, 169, 169);
            panelHomePage.add(logo);

            //label title application
            lblNewLabel = new JLabel("PRIVATE "
                        + "COVID-19 VACCINATION SYSTEM");
            lblNewLabel.setHorizontalAlignment
            (SwingConstants.CENTER);
            lblNewLabel.setFont(new Font
                        ("Arial", Font.BOLD, 28));
            lblNewLabel.setBounds(301, 319, 603, 41);
            panelHomePage.add(lblNewLabel);

            lblProvidedByMinistry = new JLabel
```

```
                ("PROVIDED BY: MINISTRY OF HEALTH");
            lblProvidedByMinistry.setHorizontalAlignment
            (SwingConstants.CENTER);
            lblProvidedByMinistry.setFont(new Font
                ("Arial", Font.PLAIN, 18));
            lblProvidedByMinistry.setBounds
            (301, 350, 603, 40);
            panelHomePage.add(lblProvidedByMinistry);

            lblAuthorIKetut = new JLabel("Developed by: "
                + "I Ketut Mahendra (E1900340)");
            lblAuthorIKetut.setHorizontalAlignment
            (SwingConstants.CENTER);
            lblAuthorIKetut.setFont(new Font
                ("Arial", Font.PLAIN, 18));
            lblAuthorIKetut.setBounds(301, 400, 603, 40);
            panelHomePage.add(lblAuthorIKetut);
            panelConfirmOrRejectAppointment.setLayout(null);
            layeredPaneVaccine.add(
                panelConfirmOrRejectAppointment,
                "name_340232029253800");

            //label confirm or reject vaccination
            //appointment
            lblConfirmOrReject = new JLabel
            ("Confirm or Reject Vaccination Appointment");
            lblConfirmOrReject.setForeground
            (new Color(30, 144, 255));
            lblConfirmOrReject.setFont(new Font
                ("Arial", Font.BOLD, 28));
            lblConfirmOrReject.setBounds
            (10, 10, 608, 55);
            panelConfirmOrRejectAppointment.
            add(lblConfirmOrReject);

            Object[][]objTableConfirmReject = {};
            String[] titleConfirmTable =
                  {"Vaccination ID","Patient Full Name",
                              "IC/Passport","Vaccine Batch",
                              "Expiry Date","Manufacturer",
                              "Vaccine Name","Status"};

            JScrollPane scrollTableConfirmVaccination =
                new JScrollPane();
            scrollTableConfirmVaccination.
            setBounds(10, 75, 1182, 382);
            panelConfirmOrRejectAppointment.
            add(scrollTableConfirmVaccination);
            tableConfirmReject = new JTable
                (objTableConfirmReject,
                                titleConfirmTable);
            tableConfirmReject.setFont(new Font
                ("Arial", Font.PLAIN, 14));
            scrollTableConfirmVaccination.
            setViewportView(tableConfirmReject);

            lblTitleUpdateStatus = new JLabel
                ("Update Status");
            lblTitleUpdateStatus.setHorizontalAlignment
            (SwingConstants.CENTER);
            lblTitleUpdateStatus.setFont(new Font
```

```
                ("Arial", Font.BOLD, 14));
            lblTitleUpdateStatus.setBounds
            (548, 478, 104, 32);
            panelConfirmOrRejectAppointment.
            add(lblTitleUpdateStatus);

            lblVaccinationID = new JLabel
                ("Vaccination ID");
            lblVaccinationID.setFont(new Font
                ("Arial", Font.PLAIN, 14));
            lblVaccinationID.setBounds
            (177, 516, 104, 32);
            panelConfirmOrRejectAppointment.
            add(lblVaccinationID);

            vaccinationIdField = new JTextField();
            vaccinationIdField.setBounds
            (286, 516, 218, 32);
            panelConfirmOrRejectAppointment.
            add(vaccinationIdField);
            vaccinationIdField.setColumns(10);

            JLabel lblSetStatusVaccination = new
                JLabel("Set Status");
            lblSetStatusVaccination.setFont(new
                Font("Arial", Font.PLAIN, 14));
            lblSetStatusVaccination.setBounds
            (514, 516, 74, 32);
            panelConfirmOrRejectAppointment.
            add(lblSetStatusVaccination);

            String[]setStatusVaccination =
                  {"Confirm","Reject"};
            JComboBox comboBoxVaccinationStatus = new
                JComboBox(setStatusVaccination);
            comboBoxVaccinationStatus.setModel
            (new DefaultComboBoxModel
                (new String[]
                            {"", "Confirm",
                                "Reject"}));
            comboBoxVaccinationStatus.setFont
            (new Font("Arial", Font.PLAIN, 14));
            comboBoxVaccinationStatus.setBounds
            (598, 516, 218, 32);
            panelConfirmOrRejectAppointment.
            add(comboBoxVaccinationStatus);

            btnUpdateStatusVaccination =
                new JButton("Update");
            btnUpdateStatusVaccination.
            setForeground(new Color
                (255, 255, 255));
            btnUpdateStatusVaccination.
            setBackground(new Color
                (50, 205, 50));
            btnUpdateStatusVaccination.
            setFont(new Font("Arial",
                Font.BOLD, 14));
            btnUpdateStatusVaccination.
            setBounds(819, 516, 143, 32);
            panelConfirmOrRejectAppointment.
```

```
            add(btnUpdateStatusVaccination);
            layeredPaneVaccine.add
            (panelViewVaccineBatch,
                        "name_340232052609700");
            panelViewVaccineBatch.
            setLayout(null);

            lblViewVB = new JLabel
                        ("View Vaccine Batch");
            lblViewVB.setBounds
            (10, 10, 623, 55);
            lblViewVB.setForeground(new Color
                        (30, 144, 255));
            lblViewVB.setFont(new Font
                        ("Arial", Font.BOLD, 28));
            panelViewVaccineBatch.
            add(lblViewVB);

            //table data
            Object[][]batchInfo = {};

            String[]title = {"Batches","Vaccine Name",
                        "Pending Appointment"};

            scrollTableViewVB = new JScrollPane();
            scrollTableViewVB.setEnabled(false);
            scrollTableViewVB.setAlignmentY
            (Component.BOTTOM_ALIGNMENT);
            scrollTableViewVB.setFont(new Font
                        ("Arial", Font.BOLD, 14));
            scrollTableViewVB.setBounds
            (10, 69, 1182, 423);
            panelViewVaccineBatch.add
            (scrollTableViewVB);
            tableViewVB = new JTable
                        (batchInfo, title);
            scrollTableViewVB.setViewportView
            (tableViewVB);
            tableViewVB.setRowHeight(30);
            tableViewVB.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            tableViewVB.setSelectionMode
            (ListSelectionModel.SINGLE_SELECTION);

            JButton seeDetailVBButton = new
                        JButton("See Detail");
            seeDetailVBButton.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            seeDetailVBButton.setBounds
            (522, 502, 158, 60);
            panelViewVaccineBatch.
            add(seeDetailVBButton);

            panelRecordNewVaccineBatch =
                        new JPanel();
            layeredPaneVaccine.setLayer
            (panelRecordNewVaccineBatch, 5);
            panelRecordNewVaccineBatch.
            setBackground(Color.WHITE);
            panelRecordNewVaccineBatch.
            setBorder(new TitledBorder
```

```
                (null, "", TitledBorder.
                                LEADING,
                                TitledBorder.
                                TOP, null, null));
        panelRecordNewVaccineBatch.
        setLayout(null);
        layeredPaneVaccine.add
        (panelRecordNewVaccineBatch,
                "name_340232072685100");

        lblRecordNewVB = new JLabel
                ("Record New Vaccine Batch");
        lblRecordNewVB.setForeground
        (new Color(30, 334, 255));
        lblRecordNewVB.setFont(new Font
                ("Arial", Font.BOLD, 28));
        lblRecordNewVB.setBounds
        (10, 10, 608, 55);
        panelRecordNewVaccineBatch.
        add(lblRecordNewVB);

        lblVaccineID = new JLabel
                (9SELECT VACCINE ID");
        lblVaccineID.setFont(new Font
                ("Arial", Font.PLAIN, 14));
        lblVaccineID.setBounds(10, 87, 200, 48);
        panelRecordNewVaccineBatch.
        add(lblVaccineID);
        47
        final JComboBox comboBoxVaccineID
        = new JComboBox(vaccine);
        comboBoxVaccineID.addActionListener
        (new ActionListener() {
              public void actionPerformed
              (ActionEvent e) {
                    if(comboBoxVaccineID.
                                getSelectedIndex()==0)
                    {
                          manufacturerTextField.
                          setText("");
                          vaccineNameTextField.
                          setText("");
                    }
                    else if(comboBoxVaccineID.
                                getSelectedIndex()==1) {
                          manufacturerTextField.
                          setText
                          (controller.
                                      getVaccineManufacturer
                                      (0));
                          vaccineNameTextField.
                          setText
                          (controller.
                                      getVaccineName
                                      (0));
                    }else if(comboBoxVaccineID.
                                getSelectedIndex()
                                ==2)
                    {
                          manufacturerTextField.
                          setText
```

```
                        (controller.
                                    getVaccineManufacturer
                                    (1));
                        vaccineNameTextField.
                        setText
                        (controller.
                                    getVaccineName
                                    (1));
                }
            }
});
comboBoxVaccineID.
setBackground(Color.WHITE);
comboBoxVaccineID.setFont
(new Font("Arial",
            Font.PLAIN, 14));
comboBoxVaccineID.setBounds
(264, 88, 346, 47);
panelRecordNewVaccineBatch.
add(comboBoxVaccineID);

JLabel lblVaccineData =
            new JLabel("VACCINE DATA");
lblVaccineData.setFont
(new Font("Arial", Font.BOLD, 14));
lblVaccineData.setBounds(10, 180, 440, 26);
panelRecordNewVaccineBatch.
add(lblVaccineData);

JLabel lblManufacturer =
            new JLabel
            ("MANUFACTURER");
lblManufacturer.setFont
(new Font("Arial",
            Font.PLAIN, 14));
lblManufacturer.setBounds
(10, 216, 244, 47);
panelRecordNewVaccineBatch.
add(lblManufacturer);

manufacturerTextField =
            new JTextField();
manufacturerTextField.
setEditable(false);
manufacturerTextField.
setBackground(Color.WHITE);
manufacturerTextField.
setFont(new Font("Arial",
            Font.PLAIN, 14));
manufacturerTextField.
setBounds(264, 216, 932, 47);
panelRecordNewVaccineBatch.
add(manufacturerTextField);
manufacturerTextField.setColumns
(10);

//panel view vaccination administered
panelViewVaccinationAdministered.
setBackground(Color.WHITE);
panelViewVaccinationAdministered.
setBorder(new TitledBorder
```

```
                (null, "",
                                TitledBorder.
                                LEADING,
                                TitledBorder.
                                TOP,
                                null,
                                null));
        layeredPaneVaccine.add
        (panelViewVaccinationAdministered,
                "name_340232089176700");
        panelViewVaccinationAdministered.
        setLayout(null);

        //Vaccine name label
        JLabel lblVaccineName =
                new JLabel("VACCINE NAME");
        lblVaccineName.setFont
        (new Font("Arial", Font.
                PLAIN, 14));
        lblVaccineName.setBounds
        (10, 282, 244, 47);
        panelRecordNewVaccineBatch.
        add(lblVaccineName);

        //Vaccine name text field
        vaccineNameTextField = new
                JTextField();
        vaccineNameTextField.
        setEditable(false);
        vaccineNameTextField.
        setBackground(Color.WHITE);
        vaccineNameTextField.
        setFont(new Font("Arial",
                Font.PLAIN, 14));
        vaccineNameTextField.
        setColumns(10);
        vaccineNameTextField.
        setBounds(264, 282, 932, 47);
        panelRecordNewVaccineBatch.
        add(vaccineNameTextField);

        //Batch number label
        lblBatchNumber = new
                JLabel("BATCH NUMBER");
        lblBatchNumber.setFont
        (new Font("Arial",
                Font.PLAIN, 14));
        lblBatchNumber.setBounds
        (10, 347, 244, 47);
        panelRecordNewVaccineBatch.
        add(lblBatchNumber);

        //Batch number text field
        batchNumTextField = new
                JTextField();
        batchNumTextField.
        setBackground(Color.WHITE);
        batchNumTextField.setFont
        (new Font("Arial", Font.PLAIN, 14));
        batchNumTextField.
        setColumns(10);
```

```
            batchNumTextField.
            setBounds(264, 347, 932, 47);
            panelRecordNewVaccineBatch.
            add(batchNumTextField);

            //Label quantity of dose available
            lblQuantityDose = new
                        JLabel
                        ("QUANTITY DOSE AVAILABLE");
            lblQuantityDose.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblQuantityDose.setBounds
            (10, 483, 244, 47);
            panelRecordNewVaccineBatch.
            add(lblQuantityDose);

            //text field quantity of dose
            quantityOfDoseTextField =
                        new JTextField();
            quantityOfDoseTextField.
            setBackground(Color.WHITE);
            quantityOfDoseTextField.
            setFont(new Font("Arial",
                        Font.PLAIN, 14));
            quantityOfDoseTextField.
            setColumns(10);
            quantityOfDoseTextField.
            setBounds(264, 483, 932, 47);
            panelRecordNewVaccineBatch.
            add(quantityOfDoseTextField);

            //combobox expiry date
            final JComboBox comboBoxDateExpiry =
                        new JComboBox(datesExpiry);
            comboBoxDateExpiry.
            setBackground(Color.WHITE);
            comboBoxDateExpiry.
            setBounds(264, 415, 228, 48);
            panelRecordNewVaccineBatch.
            add(comboBoxDateExpiry);

            //Label expiry date
            lblExpiryDateExpiry =
                        new JLabel
                        ("EXPIRY DATE");
            lblExpiryDateExpiry.
            setFont(new Font("Arial",
                        Font.PLAIN, 14));
            lblExpiryDateExpiry.
            setBounds(10, 416, 244, 47);
            panelRecordNewVaccineBatch.
            add(lblExpiryDateExpiry);

            //Label expiry month
            JLabel lblExpiryMonthExpiry =
                        new JLabel("EXPIRY MONTH");
            lblExpiryMonthExpiry.setFont(new
                        Font("Arial", Font.PLAIN, 14));
            lblExpiryMonthExpiry.setBounds
            (502, 417, 109, 47);
            panelRecordNewVaccineBatch.
```

```
            add(lblExpiryMonthExpiry);

            //combobox expiry month
            final JComboBox comboBoxMonthExpiry =
                        new JComboBox(monthExpiry);
            comboBoxMonthExpiry.setBackground
            (Color.WHITE);
            comboBoxMonthExpiry.
            setBounds(620, 416, 228, 48);
            panelRecordNewVaccineBatch.
            add(comboBoxMonthExpiry);

            //Label expiry year
            JLabel lblExpiryYearExpiry =
                        new JLabel("EXPIRY YEAR");
            lblExpiryYearExpiry.setFont
            (new Font("Arial", Font.PLAIN, 14));
            lblExpiryYearExpiry.setBounds
            (861, 416, 94, 47);

            //combobox expiry year
            final JComboBox comboBoxYearExpiry
            = new JComboBox(yearExpiry);
            comboBoxYearExpiry.
            setBackground(Color.WHITE);
            comboBoxYearExpiry.setBounds
            (968, 416, 228, 48);
            panelRecordNewVaccineBatch.
            add(comboBoxYearExpiry);
            panelRecordNewVaccineBatch.
            add(lblExpiryYearExpiry);

            //Save data JButton
            JButton btnNewButton = new
                        JButton("SAVE DATA");
            btnNewButton.addMouseListener
            (new MouseAdapter() {
                  @Override
                  public void mouseClicked
                  (MouseEvent e) {
                        //parsing the text in
                        //batch number text field
                        String batchNum =
                        batchNumTextField.getText().
                        toString();
                        int batch = Integer.
                                    parseInt(batchNum);

                        //make a data format of
                        //vaccine expiry date
                        String expiry = (String)
                        comboBoxDateExpiry.
                        getSelectedItem()+
                        "/"+comboBoxMonthExpiry.
                        getSelectedIndex()+
                        "/"+comboBoxYearExpiry.
                        getSelectedItem();

                        //parsing the text
                        //in vaccine quantity
                        //text field
```

```
                        String availableQty =
                                    quantityOfDoseTextField.
                                    getText().toString();
                        int availableQuantity = Integer.
                                    parseInt(availableQty);

                        //make a vaccine batch object
                        batch vaccineBatch = new batch
                                    (batch, expiry,
                                                availableQuantity,
                                                0);

                        //record the vaccine batch
                        //to the arraylist in class
                        //health care centre
                        healthCareCentre centre =
                                    controller.
                                    getHealthCareCentreBasedOnName
                                    (healthCareName.getText());
                        centre.setBatchesVaccine
                        (vaccineBatch);

                        //record the vaccine batch
                        //to the arraylist in class vaccine
                        String vacID = comboBoxVaccineID.
                                    getSelectedItem().
                                    toString();
                        vaccine batchVac = controller.
                                    getVaccineById(vacID);
                        batchVac.setVaccineBatch
                        (vaccineBatch);

                        //displaying a message dialog
                        //due to the successful of
                        //the process
                        JOptionPane.showMessageDialog
                        (getParent(),
                                    "Your vaccine batch is "
                                    + "succesfully registered "
                                    + "");

                        //Set the text field to blank if
                        //the data has been successfully
                        //recorded to arraylist
                        batchNumTextField.
                        setText("");
                        comboBoxVaccineID.
                        setSelectedItem("");
                        comboBoxDateExpiry.
                        setSelectedItem("");
                        comboBoxMonthExpiry.
                        setSelectedItem("");
                        comboBoxYearExpiry.
                        setSelectedItem("");
                        quantityOfDoseTextField.
                        setText("");
                    }
            });
            btnNewButton.setForeground(new Color
                        (255, 255, 255));
            btnNewButton.setBackground(new Color
```

```
                (254, 205, 50));
        btnNewButton.setFont(new Font
                ("Arial", Font.BOLD, 16));
        btnNewButton.setBounds
        (620, 87, 283, 48);
        panelRecordNewVaccineBatch.
        add(btnNewButton);

        //DELETE DATA JBUTTON
        btnDeleteData = new JButton
                ("DELETE DATA");
        btnDeleteData.addMouseListener
        (new MouseAdapter() {
              @Override
              public void mouseClicked(MouseEvent e) {
                    //set the text field to
                    //empty if this jbutton is clicked
                    batchNumTextField.
                    setText("");
                    comboBoxVaccineID.
                    setSelectedItem("");
                    comboBoxDateExpiry.
                    setSelectedItem("");
                    comboBoxMonthExpiry.
                    setSelectedItem("");
                    comboBoxYearExpiry.
                    setSelectedItem("");
                    quantityOfDoseTextField.
                    setText("");

              }
        });
        btnDeleteData.setForeground(new Color
                (255, 255, 255));
        btnDeleteData.setFont(new Font
                ("Arial", Font.BOLD, 16));
        btnDeleteData.setBackground(new Color
                (255, 0, 0));
        btnDeleteData.setBounds
        (913, 87, 283, 48);
        panelRecordNewVaccineBatch.
        add(btnDeleteData);

        //Label view vaccination administered
        lblVaccinationAdministered = new JLabel
                ("Record Vaccination Administered");
        lblVaccinationAdministered.
        setForeground(new Color(30, 144, 255));
        lblVaccinationAdministered.
        setFont(new Font("Arial", Font.BOLD, 28));
        lblVaccinationAdministered.
        setBounds(10, 10, 608, 55);
        panelViewVaccinationAdministered.
        add(lblVaccinationAdministered);

        JScrollPane
        scrollPaneTableVaccineAdministered
        = new JScrollPane();
        scrollPaneTableVaccineAdministered.
        setBounds(10, 74, 1182, 387);
        panelViewVaccinationAdministered.
```

```
            add(scrollPaneTableVaccineAdministered);

            Object[][]objectAdministered= {};
            String[]titleAdministered= {"Vaccination ID",
                        "Patient Full Name",
                        "IC/Passport",
                        "Vaccine Batch",
                        "Expiry Date",
                        "Manufacturer",
                        "Vaccine Name",
                        "Status"};
            tableVaccinationAdministered =
                        new JTable(objectAdministered,
                                    titleAdministered);
            scrollPaneTableVaccineAdministered.
            setViewportView
            (tableVaccinationAdministered);

            JLabel lblUpdateStatusAdministered =
                        new JLabel("Update Status");
            lblUpdateStatusAdministered.
            setHorizontalAlignment
            (SwingConstants.CENTER);
            lblUpdateStatusAdministered.
            setFont(new Font("Arial",
                        Font.BOLD, 14));
            lblUpdateStatusAdministered.
            setBounds(540, 471, 121, 31);
            panelViewVaccinationAdministered.
            add(lblUpdateStatusAdministered);

            JLabel lblVaccinationId = new
                        JLabel("Vaccination ID");
            lblVaccinationId.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblVaccinationId.setBounds
            (382, 502, 121, 43);
            panelViewVaccinationAdministered.
            add(lblVaccinationId);

            textFieldVaccIDAdministered = new
                        JTextField();
            textFieldVaccIDAdministered.
            setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            textFieldVaccIDAdministered.
            setBounds(504, 502, 194, 43);
            panelViewVaccinationAdministered.
            add(textFieldVaccIDAdministered);
            textFieldVaccIDAdministered.
            setColumns(10);

            JButton btnSetAdministered = new
                        JButton("Administered");
            btnSetAdministered.setForeground
            (new Color(255, 255, 255));
            btnSetAdministered.
            setBackground(new Color
                        (50, 205, 50));
            btnSetAdministered.setFont
            (new Font("Arial", Font.BOLD, 14));
```

```
            btnSetAdministered.setBounds
            (707, 502, 163, 43);
            panelViewVaccinationAdministered.
            add(btnSetAdministered);
            15nelListingVaccinationAppointment.
            setBorder(new TitledBorder
                        (null, "", TitledBorder.
                                    LEADING, TitledBorder.
                                    TOP, null, null));
            panelListingVaccinationAppointment.
            setBackground(Color.WHITE);
            layeredPaneVaccine.add
            (panelListingVaccinationAppointment,
                        "name_340232109726500");
            panelListingVaccinationAppointment.
            setLayout(null);

            //Label vaccination appointment
            lblVaccinationAppointmentData =
                        new JLabel
                        ("Vaccination Appointment Data");
            6blVaccinationAppointmentData.
            setForeground(new Color(30, 144, 255));
            lblVaccinationAppointmentData.setFont
            (new Font("Arial", Font.BOLD, 28));
            lblVaccinationAppointmentData.
            setBounds(10, 10, 608, 55);
            panelListingVaccinationAppointment.
            add(lblVaccinationAppointmentData);

            Object[][]objectVaccinationAppointment = {};
            String[]titleAppointmentView =
                  {"Vaccination ID",
                              "Patient Name",
                              "Vaccine ID",
                              "Vaccine Name",
                              "Vaccine Batch",
                              "Status"};

            scrollPaneVaccinationAppointmentData =
                        new JScrollPane();
            scrollPaneVaccinationAppointmentData.
            setBounds(10, 64, 1182, 498);
            panelListingVaccinationAppointment.
            add(scrollPaneVaccinationAppointmentData);
            tableVaccinationAppointmentData =
                        new JTable
                        (objectVaccinationAppointment,
                                    titleAppointmentView);
            scrollPaneVaccinationAppointmentData.
            setViewportView
            (tableVaccinationAppointmentData);


            //Panel listing user
            panelListingUser = new JPanel();
            layeredPaneVaccine.setLayer
            (panelListingUser, 4);
            panelListingUser.setVisible
            (false);          15
            panelListingUser.setBackground
```

```
            (Color.WHITE);
            panelListingUser.setBorder(new TitledBorder
                        (null, "", TitledBorder.
                                    LEADING, TitledBorder.
                                    TOP, null, null));
            layeredPaneVaccine.add
            (panelListingUser, "name_340232131084700");
            panelListingUser.setLayout(null);

            //Label user data
            lblUserData 6 new JLabel("User Data");
            lblUserData.setForeground(new Color
                        (30, 144, 255));
            lblUserData.setFont(new Font("Arial",
                        Font.BOLD, 28));
            lblUserData.setBounds
            (10, 10, 608, 55);
            panelListingUser.
            add(lblUserData);

            //combobox type of sorting data user
            String[]listSort = {"","Patient","Administrator"};
            comboBoxListSort = new JComboBox(listSort);
            comboBoxListSort.setBounds(943, 23, 156, 31);
            panelListingUser.add(comboBoxListSort);
            57
            lblSort = new JLabel("Sort By");
            lblSort.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblSort.setBounds(893, 23, 47, 31);
            panelListingUser.add(lblSort);

            btnSort 2 new JButton("Sort");
            btnSort.setForeground(new Color(255, 255, 255));
            btnSort.8etBackground(new Color(50, 205, 50));
            btnSort.setFont(new Font("Arial", Font.BOLD, 14));
            btnSort.setBounds(1109, 23, 83, 31);
            panelListingUser.add(btnSort);

            scrollPaneUserData = new JScrollPane();
            scrollPaneUserData.setBounds(10, 75, 1182, 487);
            panelListingUser.add(scrollPaneUserData);

            Object[][]objUserData = {};
            String[]titleUserData =
                  {"Admin ID or IC/Passport",
                        "Name","E-Mail","Health Care"};
            tableUserData = new JTable
                        (o8jUserData, titleUserData);
            tableUserData.setModel(new
                        DefaultTableModel(
                  new Object[][] {},
                  new String[] {
                        "Admin ID or IC/Passport",
                        "Name", "E-Mail", "Health Care"
                  }
            ));
            scrollPaneUserData.setViewportView
            (tableUserData);

      }
```

```
34
	@Override
	public void actionPerformed(ActionEvent e) {
		// TODO Auto-generated method stub

	}
}
```

patientDashboard.java

```
11
import java.awt.Color;
import java.awt.Component;
import java.awt.EventQueue;
import java.awt.Toolkit;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.util.ArrayList;

import javax.swing.JFrame;
import javax.swing.JMenuBar;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JButton;
import javax.swing.SwingConstants;
import javax.swing.JDesktopPane;
import javax.swing.JInternalFrame;
import java.awt.Font;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import javax.swing.border.CompoundBorder;
import javax.swing.JToggleButton;
import java.awt.SystemColor;
import javax.swing.border.BevelBorder;
import javax.swing.border.TitledBorder;
import javax.swing.JLayeredPane;
import 21vax.swing.JTextField;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.ActionEvent;
import javax.swing.JComboBox;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import java.awt.CardLayout;
import java.awt.event.ActionListener;
41port java.awt.event.ItemListener;
import java.awt.event.ItemEvent;
import javax.swing.JScrollPane;

public class patientDashboard extends JFrame {
	private PCVS controller;
	private JPanel contentPane;
	private JTable tableVaccinationAppointment;


	/**
	 * Create the frame.
	 */
	public patientDashboard(PCVS pcvs,
			final String username) {
		controller = pcvs;
```

```
setResizable(false);
tLocationByPlatform(true);
addWindowListener(new WindowAdapter() {
      @Override
      public void windowClosing(WindowEvent e) {
            int closing = JOptionPane.
                        showConfirmDialog(getParent(),
                        "Logout from this program?",
                        "Logout",
                        JOptionPane.YES_NO_OPTION,
                        JOptionPane.CLOSED_OPTION);
            if(closing==JOptionPane.YES_OPTION) {
                  PCVSGUI loginBack = new PCVSGUI
                              (controller);
                  loginBack.setVisible(true);
                  dispose();
            }
      }
});

tTitle("PCVS - Patient");
setIconImage(Toolkit.getDefaultToolkit().
            getImage(patientDashboard.
                        class.getResource
                        ("/img/pcvsIcon.png")));
setBounds(100, 50, 1240, 625);
contentPane = new JPanel();
ntentPane.setBackground(SystemColor.text);
contentPane.setBorder(null);
setContentPane(contentPane);
contentPane.setLayout(null);

JLayeredPane layeredPane = new JLayeredPane();
layeredPane.setBorder(new TitledBorder
            (null, "", TitledBorder.LEADING,
                        TitledBorder.TOP, null, null));
layeredPane.setBounds(10, 10, 1206, 407);
contentPane.add(layeredPane);

final JPanel
panelRequestVaccinationAppointment_1 =
new JPanel();
panelRequestVaccinationAppointment_1.
setVisible(false);

final JPanel
panelViewVaccinationAppointment =
new JPanel();
panelViewVaccinationAppointment.
setVisible(false);
layeredPane.setLayout(new CardLayout(0, 0));

final JPanel panelMain_1 = new JPanel();
panelMain_1.setBorder(new TitledBorder
            (null, "", TitledBorder.
                        LEADING, TitledBorder.
                        TOP, null, null));
layeredPane.add(panelMain_1,
            "name_397551532585500");
panelMain_1.setLayout(null);
```

```
JLabel logo = new JLabel("");
logo.setIcon(new ImageIcon(patientDashboard.
            class.getResource("/img/pcvsIcon.png")));
logo.setBounds(519, 22, 169, 147);
panelMain_1.add(logo);

JLabel lblTitlePC = new
            JLabel("PRIVATE COVID-19");
lblTitlePC.setHorizontalAlignment
(SwingConstants.CENTER);
lblTitlePC.setFont(new Font
            ("Arial", Font.BOLD, 48));
lblTitlePC.setBounds(299, 179, 609, 63);
panelMain_1.add(lblTitlePC);

JLabel lblTitleVS = new JLabel
            ("VACCINATION SYSTEM");
lblTitleVS.setHorizontalAlignment
(SwingConstants.CENTER);
lblTitleVS.setFont(new Font
            ("Arial", Font.PLAIN, 24));
lblTitleVS.setBounds(299, 240, 609, 36);
panelMain_1.add(lblTitleVS);

JLabel lblHi = new JLabel("Hi,");
lblHi.setHorizontalAlignment
(SwingConstants.CENTER);
lblHi.setForeground
(Color.DARK_GRAY);
lblHi.setFont(new Font
            ("Arial", Font.BOLD, 20));
lblHi.setBounds(299, 298, 609, 36);
panelMain_1.add(lblHi);

String user = controller.
            getFullNamePatientForLogin
            (username);
JLabel lblNameUser = new JLabel("User");
lblNameUser.setText(user);
lblNameUser.setForeground(Color.DARK_GRAY);
lblNameUser.setHorizontalAlignment
(SwingConstants.CENTER);
lblNameUser.setFont(new Font
            ("Arial", Font.PLAIN, 20));
lblNameUser.setBounds(299, 329, 609, 36);
panelMain_1.add(lblNameUser);
panelViewVaccinationAppointment.
setBorder(new TitledBorder
            (null, "", TitledBorder.LEADING,
                        TitledBorder.TOP, null, null));
layeredPane.add(panelViewVaccinationAppointment,
            "name_397551556887600");
panelViewVaccinationAppointment.setLayout(null);

JLabel lblVaccinationAppointment =
            new JLabel("Vaccination Appointment");
lblVaccinationAppointment.
setForeground(new Color(30, 144, 255));
lblVaccinationAppointment.
setFont(new Font("Arial", Font.BOLD, 24));
lblVaccinationAppointment.
```

```
            setBounds(10, 10, 857, 58);
            panelViewVaccinationAppointment.
            add(lblVaccinationAppointment);

            JScrollPane
            scrollPaneVaccinationAppointment =
            new JScrollPane();
            scrollPaneVaccinationAppointment.
            setBounds(10, 67, 1182, 326);
            panelViewVaccinationAppointment.
            add(scrollPaneVaccinationAppointment);

            Object[][]objVaccinationAppointment= {};
            String[]titleVaccinationAppointment =
                  {"Vaccination ID", "Vaccine Name",
                              "Appointment Date", "Status"};
            tableVaccinationAppointment = new JTable
                        (objVaccinationAppointment,
                                    titleVaccinationAppointment);
            scrollPaneVaccinationAppointment.
            setViewportView(tableVaccinationAppointment);
            panelRequestVaccinationAppointment_1.
            setBorder(new TitledBorder(null, "",
                        TitledBorder.LEADING, TitledBorder.
                        TOP, null, null));
            layeredPane.add
            (panelRequestVaccinationAppointment_1,
                        "name_397551581382100");
            panelRequestVaccinationAppointment_1.
            setLayout(null);

            JLabel lblRequestVA = new JLabel
                        ("Vaccination Appointment "
                                    + "Application Form");
            lblRequestVA.setForeground(new Color
                        (30, 144, 255));
            lblRequestVA.setFont(new Font
                        ("Arial", Font.BOLD, 24));
            lblRequestVA.setBounds(411, 10, 765, 58);
            panelRequestVaccinationAppointment_1.
            add(lblRequestVA);

            JLabel lblAvailableVaccine = new
                        JLabel("Available Vaccine");
            lblAvailableVaccine.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAvailableVaccine.setBounds
            (411, 81, 125, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAvailableVaccine);

            //populate the item of the combo
            //box based on the array list data vaccine
            String[]availableVaccine =
                        new String[controller.
                                    getVaccineData().
                                    size()];
            for (int i = 0; i < controller.
                        getVaccineData().size(); i++)
      {
                  availableVaccine[i] = "Vaccine name: "
```

```
        + controller.getVaccineData().
                              get(i).getVaccineName() +
                              ", Manufactured by: " +
                              controller.
                              getVaccineData().
                              get(i).getManufacturer();
        }
            final JComboBox comboBoxAvailableVaccine =
                        new JComboBox(availableVaccine);
            comboBoxAvailableVaccine.setBounds
            (590, 81, 586, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxAvailableVaccine);

            JLabel lblAvailableAtHealthcare =
                        new JLabel("Healthcare Centre");
            lblAvailableAtHealthcare.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAvailableAtHealthcare.setBounds
            (411, 142, 126, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAvailableAtHealthcare);

            String[]centreDataOffering = new
                        String[controller.getCentreOfHealth().
                                size()];
      new ArrayList<>();
        controller.getVaccineBatch
        (comboBoxAvailableVaccine.
                  getSelectedIndex());

      for(int k = 0; k < controller.
                  getCentreOfHealth().size();k++) {
            centreDataOffering[k]= "Centre Name: "
                        + ""+controller.
                        getCentreOfHealth().
                        get(k).getCentreName()+
                        ", Address:"+controller.
                        getCentreOfHealth().get(k).
                        getAddress();
      }

            final JComboBox comboBoxHealthcare =
                        new JComboBox(centreDataOffering);
            comboBoxHealthcare.setBounds
            (590, 143, 586, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxHealthcare);

            JLabel lblAvailableVaccineBatch = new
                        JLabel("Available Vaccine Batch");
            lblAvailableVaccineBatch.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAvailableVaccineBatch.setBounds
            (411, 205, 169, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAvailableVaccineBatch);

            final int availableVacSelected =
                        comboBoxAvailableVaccine.
                        getSelectedIndex();
```

```
            final int availableHCSelected =
                        comboBoxHealthcare.
                        getSelectedIndex();
            final JComboBox
            comboBoxAvailableVaccineBatch =
            new JComboBox();
            comboBoxAvailableVaccineBatch.
            setBounds(590, 205, 462, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxAvailableVaccineBatch);

            JLabel lblAppointmentDate = new
                        JLabel("Appointment Date");
            lblAppointmentDate.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAppointmentDate.setBounds
            (411, 267, 169, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAppointmentDate);

            String[]datesAppointment = {"","1","2",
                        "3","4","5","6","7","8","9","10",
                        "11","12","13","14","15","16",
                        "17","18","19","20","21","22",
                        "23","24","25","26","27","28",
                        "29","30","31"};

            String[]monthAppointment = {"","January",
                        "February","March","April","May",
                        "June","July","August","September",
                        "October","November",
                        "December"};

            String[]yearAppointment= {"","2019","2020",
                        "2021","2022","2023","2024","2025"};

            final JComboBox comboBoxAppointmentDate =
                        new JComboBox(datesAppointment);
            comboBoxAppointmentDate.setBounds(590,
                        268, 70, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxAppointmentDate);

            JLabel lblAppointmentMonth = new JLabel
                        ("Appointment Month");
            lblAppointmentMonth.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAppointmentMonth.setBounds
            (670, 267, 125, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAppointmentMonth);

            final JComboBox comboBoxAppointmentMonth =
                        new JComboBox(monthAppointment);
            comboBoxAppointmentMonth.setBounds
            (805, 268, 145, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxAppointmentMonth);

            JLabel lblAppointmentYear = new
                        JLabel("Appointment Year");
```

```
            lblAppointmentYear.setFont(new Font
                        ("Arial", Font.PLAIN, 14));
            lblAppointmentYear.setBounds
            (960, 267, 114, 32);
            panelRequestVaccinationAppointment_1.
            add(lblAppointmentYear);

            final JComboBox comboBoxAppointmentYear =
                        new JComboBox(yearAppointment);
            comboBoxAppointmentYear.
            setBounds(1084, 267, 92, 32);
            panelRequestVaccinationAppointment_1.
            add(comboBoxAppointmentYear);
            61
            JPanel panel = new JPanel();
            panel.setBounds(0, 0, 390, 407);
            16nelRequestVaccinationAppointment_1.
            add(panel);
            panel.setLayout(null);

            JLabel lblNewLabel = new JLabel("");
            lblNewLabel.setIcon(new ImageIcon
                        (patientDashboard.class.
                                    getResource
                                    ("/img/signInBg.png")));
            lblNewLabel.setBounds(-155, 0, 545, 407);
            panel.add(lblNewLabel);

            JButton btnNewButton = new JButton
                        ("Input Data");
            final int availableBatchSelected =
                        comboBoxAvailableVaccineBatch.
            5           getSelectedIndex()+1;
            btnNewButton.addMouseListener
            (new MouseAdapter() {
                  @Override
                  public void mouseClicked(MouseEvent e) {
                        String dateVaccApp =
                                    comboBoxAppointmentDate.
                                    getSelectedIndex() + "/" +
                                                comboBoxAppointmentMonth.
                                                getSelectedIndex() + "/" +
                                                comboBoxAppointmentYear.
                                                getSelectedItem();
                        String vaccID = controller.
                                    generateVaccinationID
                                    (availableVacSelected,
                                                availableHCSelected,
                                                availableBatchSelected);
                        vaccination newVaccination =
                                    new vaccination(vaccID,
                                                dateVaccApp,
                                                "Pending", "");
                        patient pat = controller.
                                    getPatientVaccination(username);
                        pat.getVaccinations().add(newVaccination);
                  }
            });         2
            btnNewButton.setForeground(new Color
                        (255, 255, 255));
            btnNewButton.setBackground(new Color
```

```
                (154, 205, 50));
        btnNewButton.setFont(new Font("Arial",
                Font.BOLD, 14));
        btnNewButton.setBounds(552, 340, 223, 43);
        panelRequestVaccinationAppointment_1.
        add(btnNewButton);

        JButton btnCleardata = new JButton
                ("Clear Data");
        btnCleardata.addMouseListener
        (new MouseAdapter() {
              @Override
              public void mouseClicked
              (MouseEvent e) {
                    comboBoxAppointmentDate.
                    setSelectedItem("");
                    comboBoxAppointmentMonth.
                    setSelectedItem("");
                    comboBoxAppointmentYear.
                    setSelectedItem("");
              }
        });
        btnCleardata.setForeground(new Color
                (255, 255, 255));
        btnCleardata.setBackground(new Color
                (255, 0, 0));
        btnCleardata.setFont(new Font("Arial",
                Font.BOLD, 14));
        btnCleardata.setBounds(813, 340, 223, 43);
        panelRequestVaccinationAppointment_1.add
        (btnCleardata);

        JButton btnFindBatch = new JButton
                ("Find Batch");

        btnFindBatch.setFont(new Font
                ("Arial", Font.PLAIN, 14));
        btnFindBatch.setBackground(new Color
                (102, 205, 170));
        btnFindBatch.setBounds(1062, 205, 114, 32);
        panelRequestVaccinationAppointment_1.
        add(btnFindBatch);

        btnFindBatch.addMouseListener
        (new MouseAdapter() {
              @Override
              public void mouseClicked
              (MouseEvent e) {
                    comboBoxAvailableVaccineBatch.
                    removeAllItems();
                    for(int i = 0; i < controller.
                                getVaccineBatchForAppoimtment
                                (availableHCSelected,
                                            availableVacSelected).
                                size();i++) {
                          comboBoxAvailableVaccineBatch.
                          addItem(controller.
                                      getVaccineBatchForAppoimtment
                                      (availableHCSelected,

availableVacSelected));
```

```
                }
            }
        });

        JButton btnMainButton = new JButton
                ("Main Panel");
        btnMainButton.addMouseListener
        (new MouseAdapter() {
            @Override
            public void mouseClicked
            (MouseEvent e) {
                panelMain_1.
                setVisible(true);
                panelRequestVaccinationAppointment_1.
                setVisible(false);
                panelViewVaccinationAppointment.
                setVisible(false);
            }
        });
        btnMainButton.setFont(new Font
                ("Arial", Font.PLAIN, 18));
        btnMainButton.setBounds
        (33, 456, 356, 96);
        contentPane.add(btnMainButton);

        JButton btnRequestNewVaccination =
                new JButton
                ("Request New Vaccination Appointment");
        btnRequestNewVaccination.addMouseListener
        (new MouseAdapter() {
            @Override
            public void mouseClicked
            (MouseEvent e) {
                panelMain_1.setVisible(false);
                panelRequestVaccinationAppointment_1.
                setVisible(true);
                panelViewVaccinationAppointment.
                setVisible(false);
            }
        });
        btnRequestNewVaccination.setFont
        (new Font("Arial", Font.PLAIN, 18));
        btnRequestNewVaccination.setBounds
        (422, 455, 369, 97);
        contentPane.add(btnRequestNewVaccination);

        JButton btnViewVaccinationAppointment =
                new JButton
                ("View Vaccination Appointment");
        btnViewVaccinationAppointment.
        addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                panelMain_1.setVisible(false);
                panelRequestVaccinationAppointment_1.
                setVisible(false);
                panelViewVaccinationAppointment.
                setVisible(true);
            }
        });
        btnViewVaccinationAppointment.setFont
```

```
            (new Font("Arial", Font.PLAIN, 18));
            btnViewVaccinationAppointment.
            setBounds(824, 455, 369, 97);
            contentPane.add
            (btnViewVaccinationAppointment);
      }
}
```

aboutMe.java

```
import java.awt.BorderLayout;
import java.awt.FlowLayout;

import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import java.awt.Font;
import javax.swing.UIManager;
import javax.swing.JScrollPane;
import javax.swing.JTextPane;
import javax.swing.JSeparator;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.SystemColor;
import java.awt.Color;
import java.awt.Toolkit;

public class aboutMe extends JDialog {

      private final JPanel contentPanel =
                  new JPanel();

      /**
       * Create the dialog.
       */
      public aboutMe() {
            setIconImage(Toolkit.getDefaultToolkit().
                        getImage(aboutMe.class.
                                    getResource
                                    ("/img/pcvsIcon.png")));
            setResizable(false);
            getContentPane().setBackground
            (SystemColor.controlDkShadow);
            setUndecorated(true);
            setDefaultCloseOperation(JDialog.
                        DISPOSE_ON_CLOSE);
            setBounds(100, 100, 574, 464);
            getContentPane().setLayout
            (new BorderLayout());
            contentPanel.setBackground
            (SystemColor.controlDkShadow);
            contentPanel.setBorder
            (new EmptyBorder(5, 5, 5, 5));
            getContentPane().add
            (contentPanel, BorderLayout.CENTER);
            contentPanel.setLayout(null);
            {
```

```
            JLabel lblNewLabel = new JLabel("");
            lblNewLabel.setIcon(new ImageIcon
                        (aboutMe.class.getResource
                                    ("/img/pcvsIcon.png")));
            lblNewLabel.setBounds(10, 10, 175, 143);
            contentPanel.add(lblNewLabel);
      }
      {
            JLabel lblVaccinationSystem =
                        new JLabel("PRIVATE COVID-19");
            lblVaccinationSystem.setForeground
            (SystemColor.menu);
            lblVaccinationSystem.setFont(new Font
                        ("Arial", Font.BOLD, 38));
            lblVaccinationSystem.setBounds
            (195, 30, 355, 63);
            contentPanel.add(lblVaccinationSystem);
      }
      {
            JLabel lblPrivateCovid19 = new JLabel
                        ("vaccination system.");
            lblPrivateCovid19.setForeground
            (SystemColor.menu);
            lblPrivateCovid19.setFont(new Font
                        ("Arial", Font.PLAIN, 24));
            lblPrivateCovid19.setBounds
            (195, 84, 355, 41);
            contentPanel.add(lblPrivateCovid19);
      }

      JScrollPane scrollPane = new JScrollPane();
      scrollPane.setEnabled(false);
      scrollPane.setBounds(10, 193, 554, 193);
      contentPanel.add(scrollPane);
      {
            JTextPane txtpnDevelopedByStudent =
                        new JTextPane();
            txtpnDevelopedByStudent.setEditable
            (false);
            txtpnDevelopedByStudent.setFont
            (new Font("Tahoma", Font.PLAIN, 14));
            txtpnDevelopedByStudent.setText
            ("Developed by:\r\nStudent Name\t  :"
                        + " I Ketut Mahendra\r\nStudent ID"
                        + "\t\t  : E1900340\r\nSubject\t\t "
                        + " : BIT203 - Advance Object Oriented"
                        + " Programming\r\n");
            scrollPane.setColumnHeaderView
            (txtpnDevelopedByStudent);
      }

      JTextPane txtpnTheCovidVaccination =
                  new JTextPane();
      txtpnTheCovidVaccination.
      setEditable(false);
      txtpnTheCovidVaccination.
      setFont(new Font("Arial", Font.PLAIN, 12));
      txtpnTheCovidVaccination.
      setText("The Covid-19 vaccination rollout "
                  + "in Malaysia and most countries "
                  + "have been underway for a\r\nfew "
```

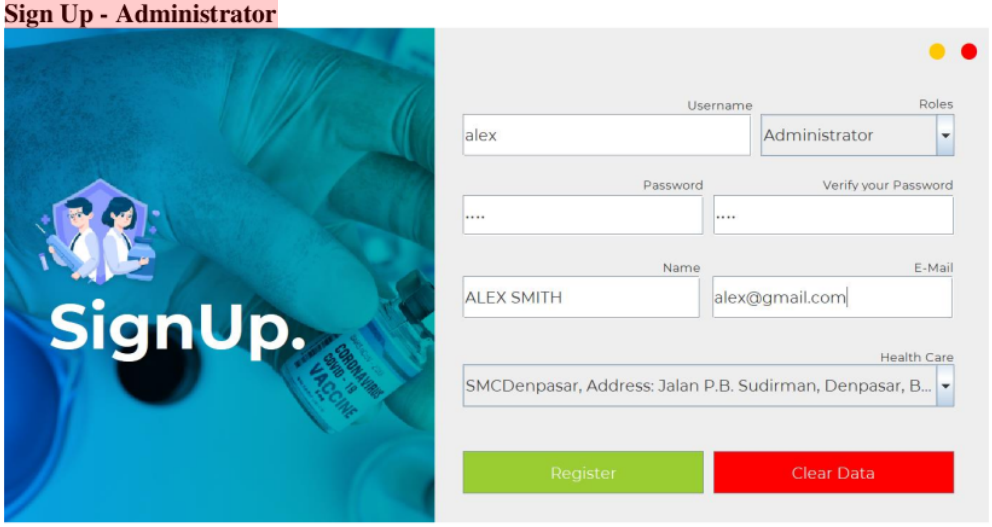
```
                    + "months. "
                    + "As more and more of the "
                    + "population "
                    + "have been vaccinated, private "
                    + "healthcare\r\ncentres have been "
                    + "allowed to purchase and "
                    + "administer "
                    + "vaccines, especially to "
                    + "patients who "
                    + "want\r\nto choose the "
                    + "type of vaccine "
                    + "that they receive. However, the "
                    + "vaccinations that have been\r\n"
                    + "administered "
                    + "by the private healthcare "
                    + "centres have to "
                    + "be recorded in the national "
                    + "vaccination"
                    + "\r\ncommittee database. "
                    + "A system is required for "
                    + "the Private Covid-19 Vaccination "
                    + "Scheme, known\r\nas PCVS.");
            scrollPane.setViewportView
            (txtpnTheCovidVaccination);
            {
                  JLabel lblVersion = new
                              JLabel("Version 1.0");
                  lblVersion.setForeground
                  (SystemColor.menu);
                  lblVersion.setFont(new
                              Font("Arial",
                                          Font.PLAIN, 14));
                  lblVersion.setBounds(12,
                              122, 355, 31);
                  contentPanel.add(lblVersion);
            }
            {
                  JPanel buttonPane = new
                              JPanel();
                  buttonPane.setBackground
                  (SystemColor.controlDkShadow);
                  buttonPane.setLayout(new
                              FlowLayout(FlowLayout.
                                          RIGHT));
                  getContentPane().add(buttonPane,
                              BorderLayout.SOUTH);
                  {
                        JButton okButton = new
                                    JButton("OK");
                        okButton.addMouseListener
                        (new MouseAdapter() {
                              @Override
                              public void mouseClicked
                              (MouseEvent e) {
                              dispose();
                              }
                        });
                        okButton.setActionCommand
                        ("OK");
                        buttonPane.add
                        (okButton);
```

```
                        getRootPane().
                        setDefaultButton(okButton);
                    }
                }
            }
}
```

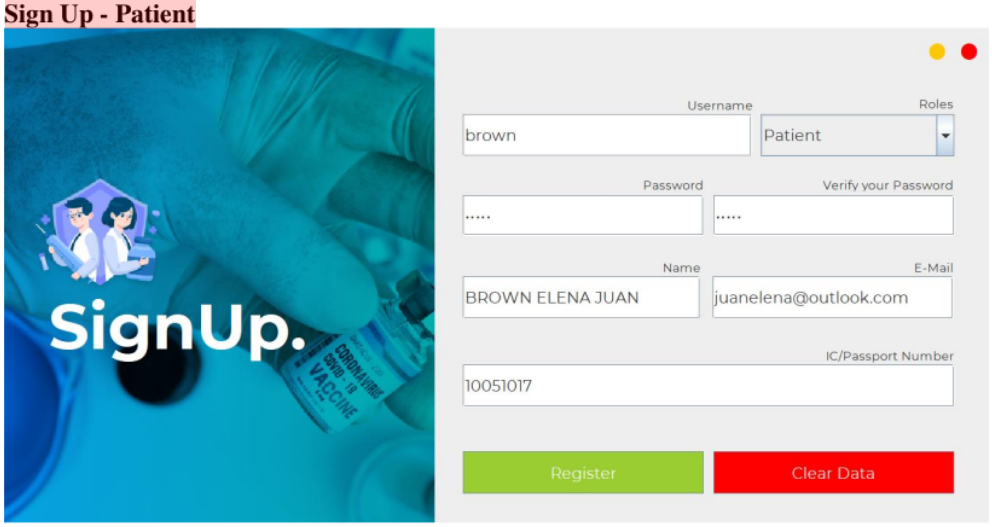
1

Sample Output

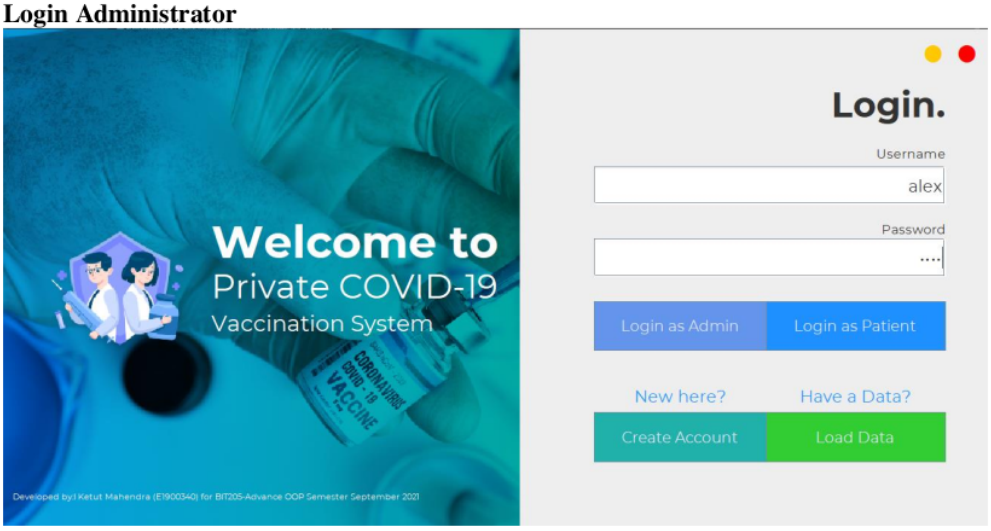
Sign Up - Administrator



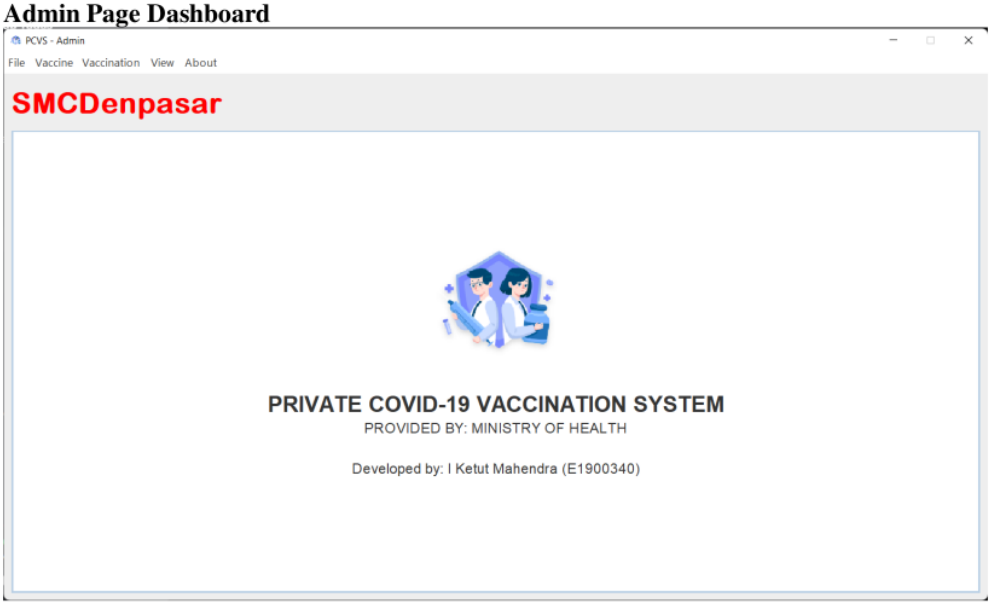
Sign Up - Patient



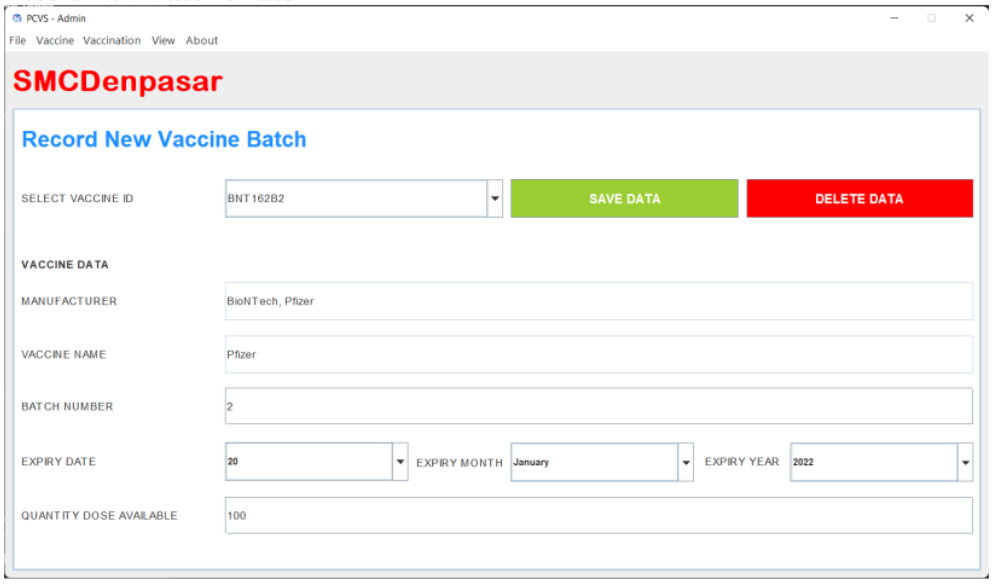
Login Administrator



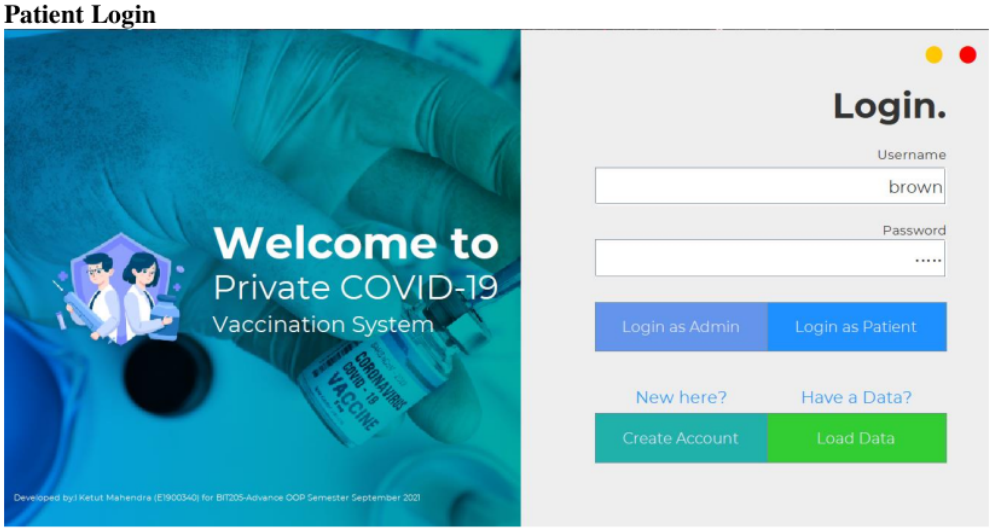
Admin Page Dashboard



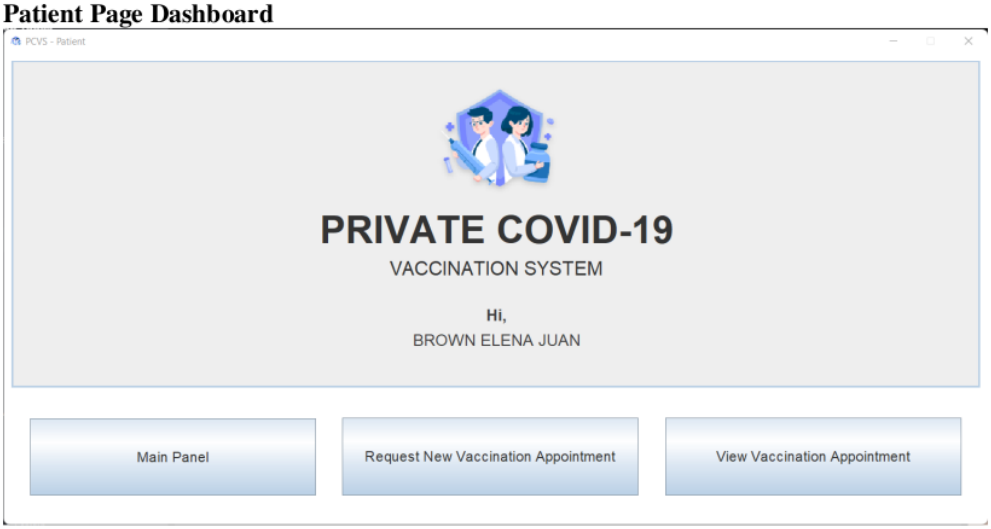
Record New Vaccine Batch



Patient Login

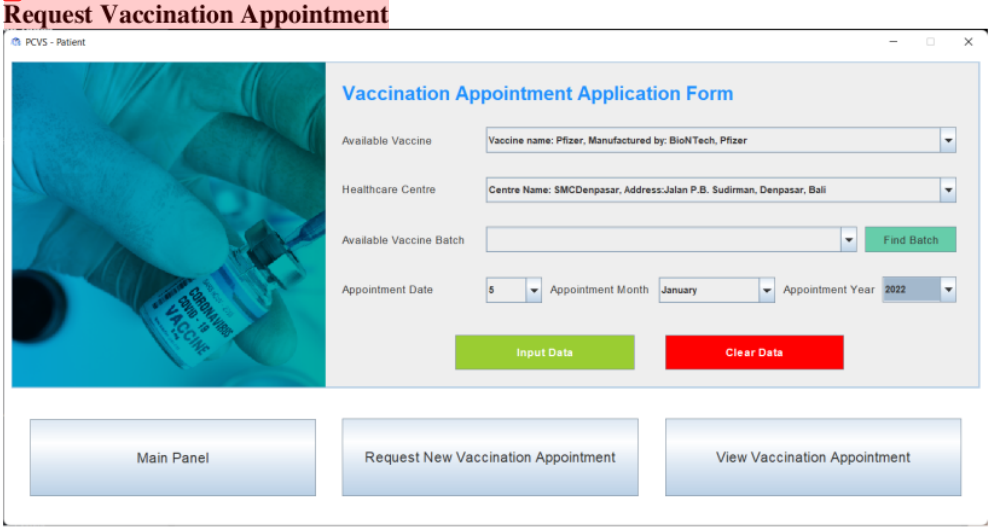


Patient Page Dashboard

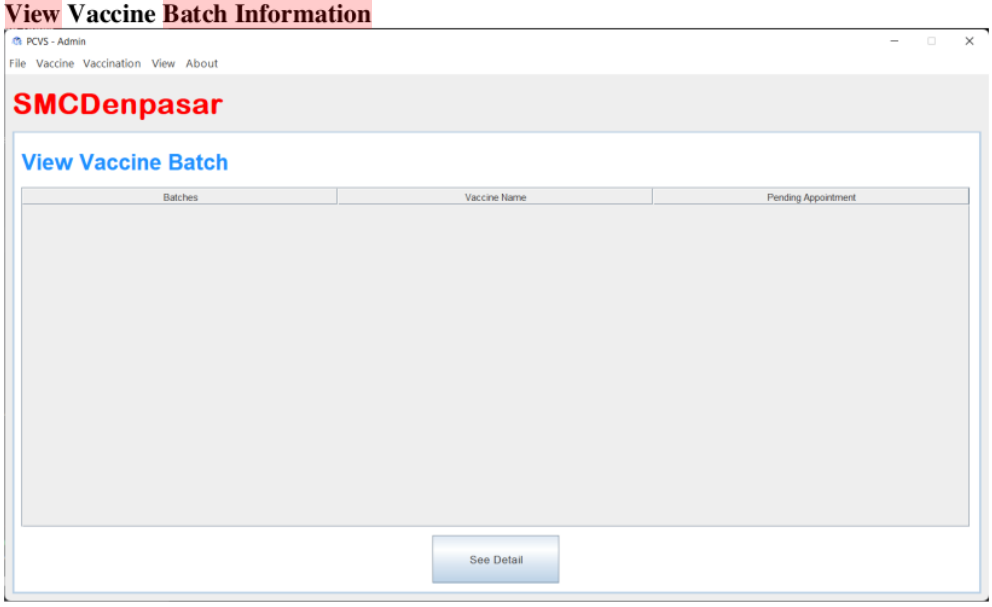


1

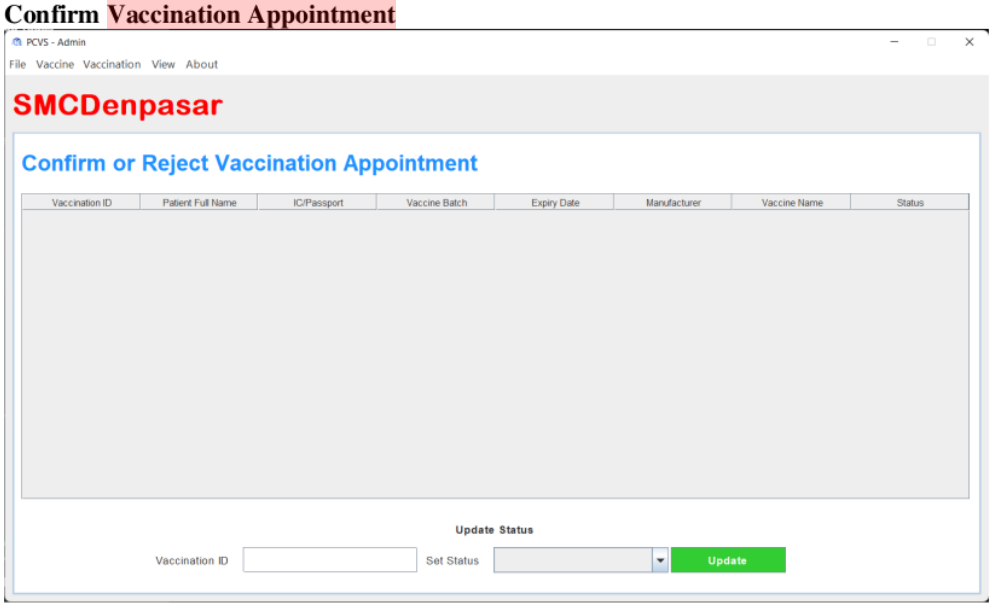
Request Vaccination Appointment



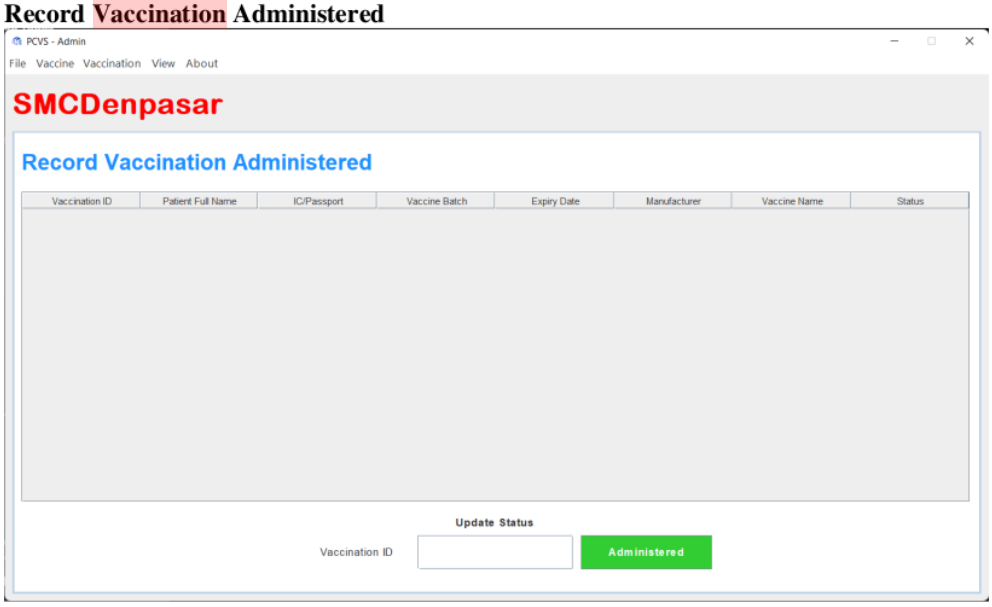
View Vaccine Batch Information



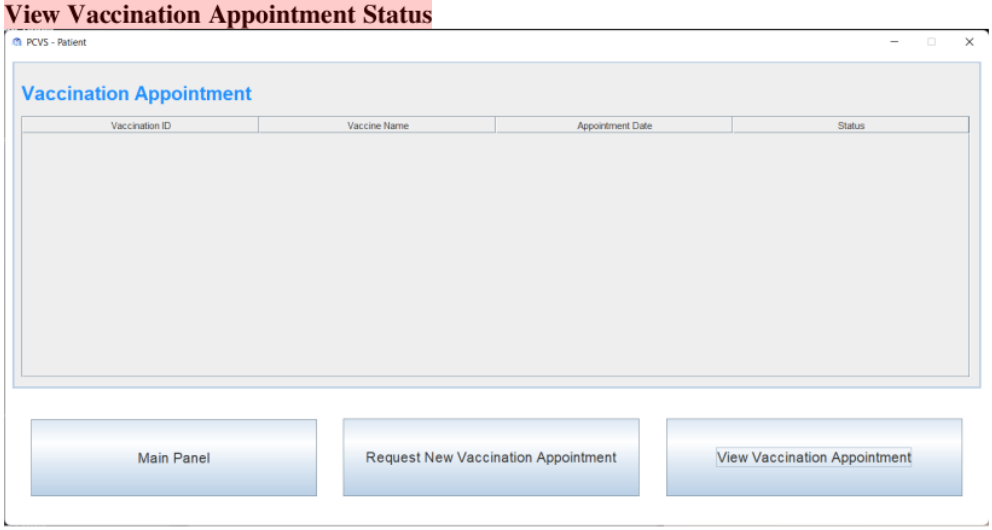
Confirm Vaccination Appointment



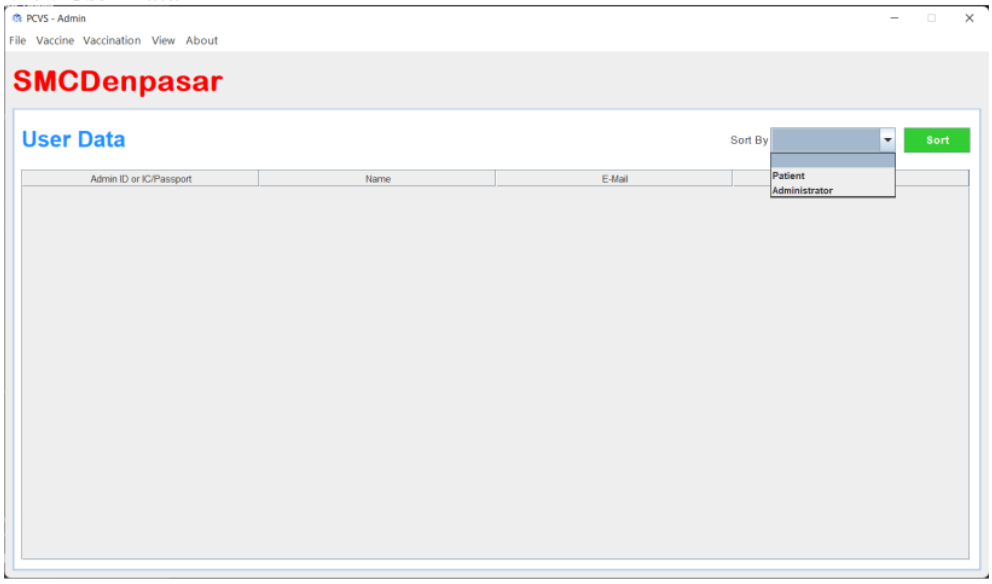
Record Vaccination Administered



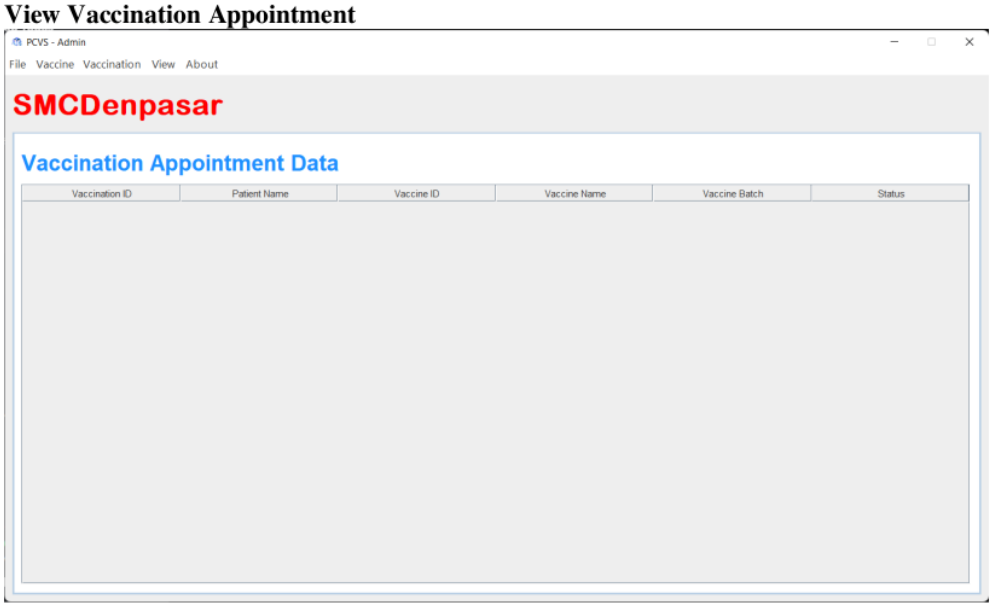
View Vaccination Appointment Status



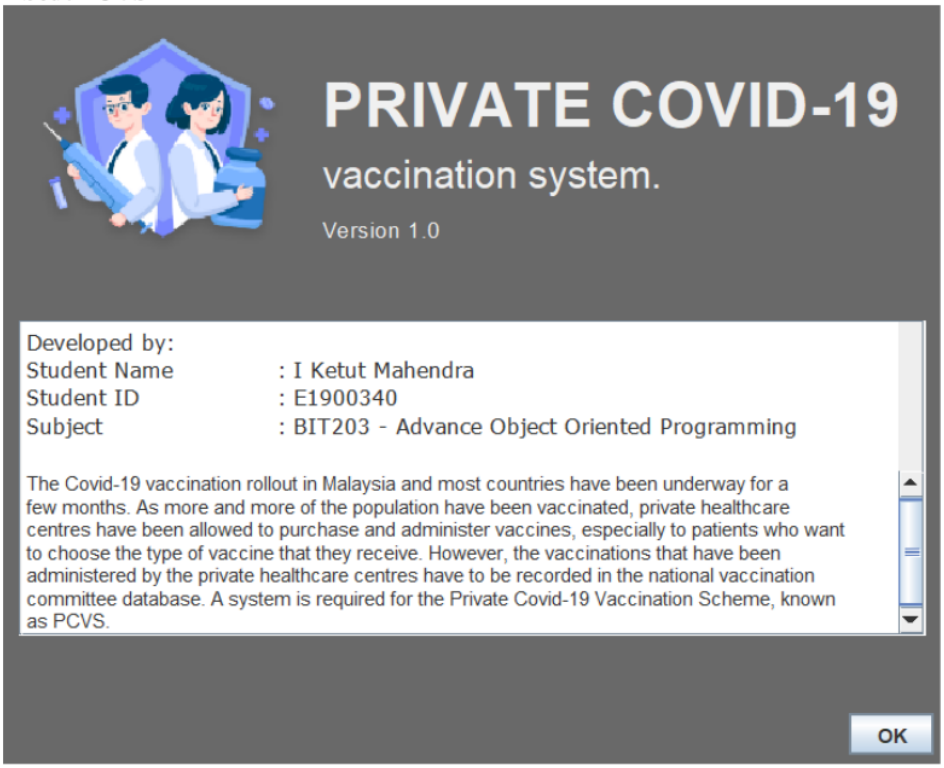
View User Data



View Vaccination Appointment



About PCVS


PRIVATE COVID-19
vaccination system.
Version 1.0
Developed by:
Student Name : I Ketut Mahendra
Student ID : E1900340
Subject : BIT203 - Advance Object Oriented Programming
The Covid-19 vaccination rollout in Malaysia and most countries have been underway for a few months. As more and more of the population have been vaccinated, private healthcare centres have been allowed to purchase and administer vaccines, especially to patients who want to choose the type of vaccine that they receive. However, the vaccinations that have been administered by the private healthcare centres have to be recorded in the national vaccination committee database. A system is required for the Private Covid-19 Vaccination Scheme, known as PCVS.
OK

E1900340_BIT203_ASSIGNMENT2

ORIGINALITY REPORT

35%	25%	8%	23%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to HELP UNIVERSITY Student Paper	13%
2	java.hotexamples.com Internet Source	2%
3	gitlab.devops.ifrn.edu.br Internet Source	2%
4	www.coursehero.com Internet Source	2%
5	earthconquest.tistory.com Internet Source	1%
6	Submitted to University of Central Lancashire Student Paper	1%
7	www.challenge.nm.org Internet Source	1%
8	coned.utcluj.ro Internet Source	1%
9	www.atlashymenoptera.net Internet Source	1%

10	github.com Internet Source	1 %
11	jexp.ru Internet Source	1 %
12	codegur.com Internet Source	1 %
13	key-to-programming.blogspot.com Internet Source	1 %
14	pablosuzarte.wordpress.com Internet Source	<1 %
15	Submitted to Queen Mary and Westfield College Student Paper	<1 %
16	es.scribd.com Internet Source	<1 %
17	okeanis.lib.teipir.gr Internet Source	<1 %
18	ojuba.org Internet Source	<1 %
19	git.unilim.fr Internet Source	<1 %
20	transfer.kg Internet Source	<1 %
21	giridhar-mb.blogspot.com	

	Internet Source	<1%
22	Submitted to National Institute of Technology, Kurukshetra Student Paper	<1%
23	Submitted to unina Student Paper	<1%
24	Submitted to University of Wales Institute, Cardiff Student Paper	<1%
25	stars.library.ucf.edu Internet Source	<1%
26	Submitted to City University Student Paper	<1%
27	Submitted to Sunway Education Group Student Paper	<1%
28	www.10qianwan.com Internet Source	<1%
29	Submitted to University of Huddersfield Student Paper	<1%
30	www.evoljava.com Internet Source	<1%
31	www.java-forum.org Internet Source	<1%

32	gitext.gfz-potsdam.de Internet Source	<1 %
33	veetmoradiyaprojects.wordpress.com Internet Source	<1 %
34	www.shiningboys.cn Internet Source	<1 %
35	idus.us.es Internet Source	<1 %
36	rvscript.com Internet Source	<1 %
37	Submitted to Informatics Education Limited Student Paper	<1 %
38	jnjsite.com Internet Source	<1 %
39	pastebin.com Internet Source	<1 %
40	5302414021nining.blogspot.com Internet Source	<1 %
41	computrachos.com Internet Source	<1 %
42	hemamaliweragama.info Internet Source	<1 %
43	Dietmar Abts. "Grundkurs JAVA", Springer Science and Business Media LLC, 2016	<1 %

	Publication	
44	community.oracle.com Internet Source	<1 %
45	yuanmacha.com Internet Source	<1 %
46	Submitted to Arab Open University Student Paper	<1 %
47	bluemini.com Internet Source	<1 %
48	mv-voice.com Internet Source	<1 %
49	Christopher Dillis, Connor McIntee, Ted Grantham, Van Butsic, Lance Le, Kason Grady. "Water storage and irrigation practices associated with cannabis production drive seasonal patterns of water extraction and use in Northern California watersheds", Cold Spring Harbor Laboratory, 2019 Publication	<1 %
50	Submitted to Technological Institute of the Philippines Student Paper	<1 %
51	Submitted to The Manchester College Student Paper	<1 %
52	blog.csdn.net Internet Source	<1 %

53	gitlab.engr.ship.edu Internet Source	<1 %
54	www.outi.be Internet Source	<1 %
55	fac.ksu.edu.sa Internet Source	<1 %
56	forum.byte-welt.net Internet Source	<1 %
57	sc.blogsite.org Internet Source	<1 %
58	www.qiushurong.cn Internet Source	<1 %
59	www.slideshare.net Internet Source	<1 %
60	documents.mx Internet Source	<1 %
61	mkyong.com Internet Source	<1 %
62	openlab.ncl.ac.uk Internet Source	<1 %
63	Barry Burd. "Java® For Dummies®", Wiley, 2011 Publication	<1 %

64	David Parsons. "Foundational Java", Springer Science and Business Media LLC, 2020 Publication	<1%
65	repository.its.ac.id Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		