**Introduction:**

Accurate and timely immunization records play a critical role in monitoring and ensuring the health and well-being of children in Philippine barangays, the smallest administrative units in the country. However, the current lack of a comprehensive digital health system poses significant challenges in maintaining effective immunization tracking systems at the grassroots level. This deficiency compromises public health interventions and places the health of children in these communities at risk. To address this pressing issue, there is a compelling need to implement a digital immunization record system that can revolutionize the management and monitoring of child immunization data in barangays. This research highlights the transformative potential of such a system and emphasizes the urgency of its implementation to improve child health outcomes.

**Problem Statement:**

The healthcare sector in Philippine barangays, the smallest administrative units in the country, faces significant challenges in maintaining effective immunization records for children aged 12 and below. The absence of a comprehensive digital health system hampers the accurate tracking of vaccination statuses, posing a risk to public health interventions and jeopardizing the health of children in these communities. Despite the ongoing digitization efforts in the Philippine health system, there remains an urgent need for a digital solution to manage and monitor child immunization data at the barangay level.

**Scope and Limitations**

The scope of the proposed digital immunization record system is to facilitate the registration and immunization tracking of children aged 12 years and below in Philippine barangays. The system aims to provide a comprehensive platform for healthcare providers in barangay clinics to record and monitor the immunization status of children, ensuring timely and appropriate vaccinations. It includes features such as registration of children, scheduling immunization sessions. The system is intended to be used by healthcare providers specifically within the barangay clinics.

While the digital immunization record system offers significant benefits, there are certain limitations that need to be considered—Data Privacy and Security to be specific—since Safeguarding the confidentiality and security of personal health information is crucial. Implementing appropriate data protection measures and ensuring adherence to privacy regulations are essential considerations in the design and deployment of the system.

**Problem Background:**

The healthcare sector in Philippine barangays, the smallest administrative units in the country, struggles with maintaining effective immunization records for children 12 years old and below. The accurate tracking of vaccination statuses, a crucial health indicator, often suffers due to the lack of a comprehensive digital health system. This deficiency can compromise public health interventions, risking children's health in these communities.

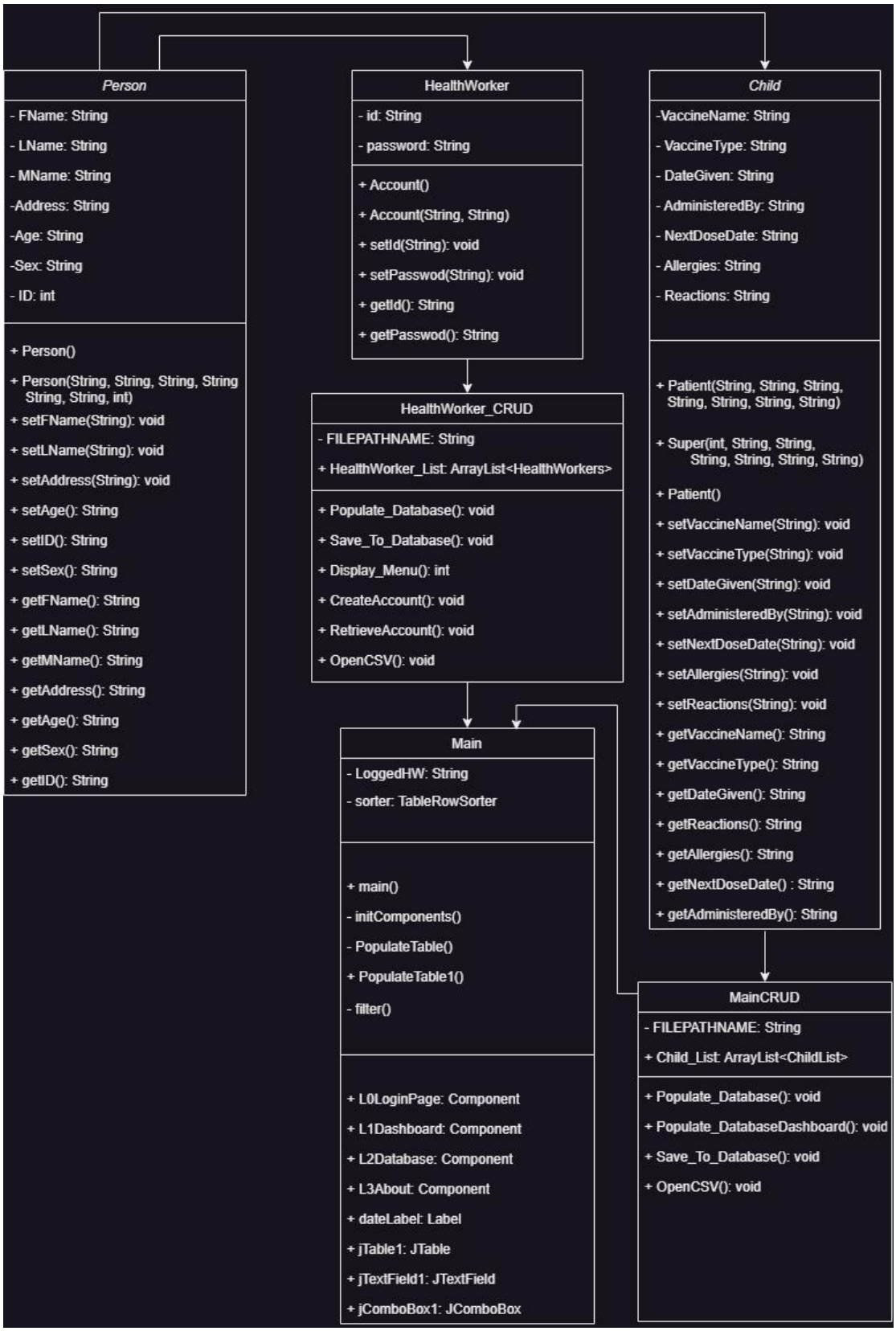
Labrique et al. (2018) highlighted the transformative potential of digital health systems. These systems can enhance health service delivery, improve patient-health worker communication, and facilitate faster access to essential health data [1]. This underscores the need for digital immunization record systems in barangays to manage and monitor child immunization data effectively.

A UNICEF report (2022) underscores the importance of efficient vaccination tracking systems. Such systems are crucial to ensure that children, particularly in remote and underserved areas, receive timely vaccinations [2].

Macabasag et al. (2022) discussed the ongoing digitization of the Philippine health system, including the normalization of electronic medical records. This indicates the potential for a digital immunization record system at the barangay level to efficiently track and manage child immunization data [3].

Considering these findings, there is an immediate need for a digital solution. A barangay-level child immunization monitoring system could significantly improve the management and monitoring of child immunization records, enhancing overall child health outcomes in these communities.

## Class Diagram (in UML notation):



**Functional Requirements:**

(a) Registration:

The system incorporates a registration feature that enables health workers to enroll children into the digital immunization record system. This process involves capturing essential demographic information, such as the child's name and other relevant details.

(b) Vaccine Scheduling:

The system provides a scheduling feature to assist healthcare workers and parents/guardians in tracking and planning upcoming immunizations based on the child's age and recommended vaccination schedule.

(c) Data Storage and Retrieval:

The system securely stores immunization records, allowing authorized healthcare providers to access and retrieve information as needed. This facilitates continuity of care and enables accurate monitoring of vaccination statuses.

(d) Login:

The digital immunization record system ensures that only authorized personnel with valid credentials can access the system, input data into the database, and retrieve information. These security measures safeguard the integrity and confidentiality of the immunization data, reducing the risk of unauthorized access or misuse.

(e) Database Viewing:

The system allows administrators to view the information of the children and accounts stored in the database.

(f) Medical Details:

The Clinitrack system's dashboard displays the medical details of children, including their names, vaccine types, dates of administration, and the next dose date.

(g) Search Filter:

The system's search bar filter enables users to efficiently browse data by allowing them to search using criteria such as ID, address, age, sex, and vaccine type.

(h) Authorized Health Workers:

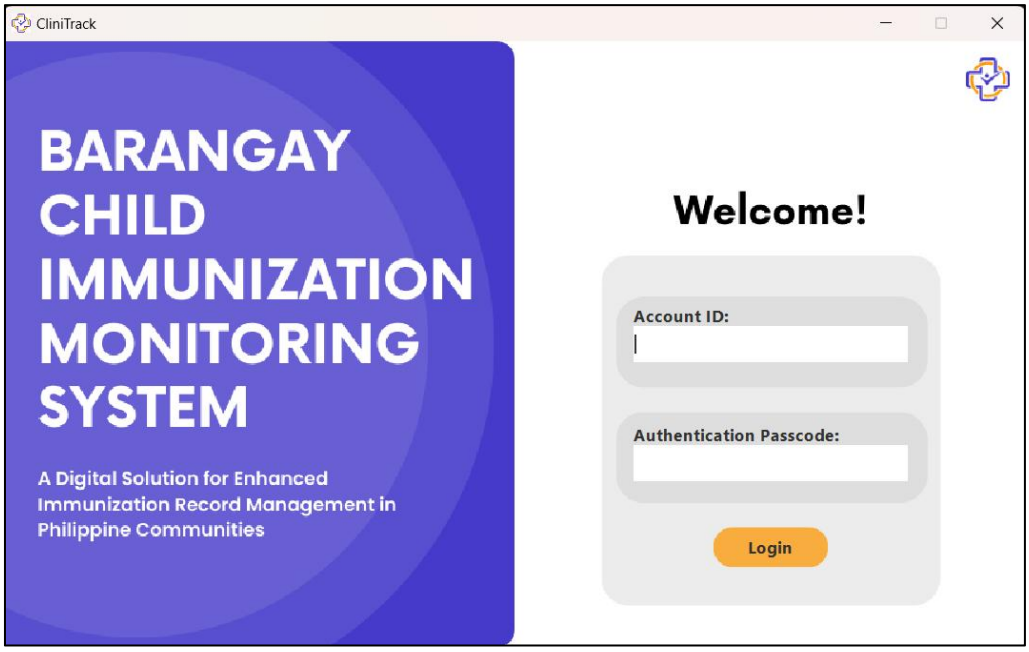
Only authorized personnel can access the data within the system.

(i) Notes

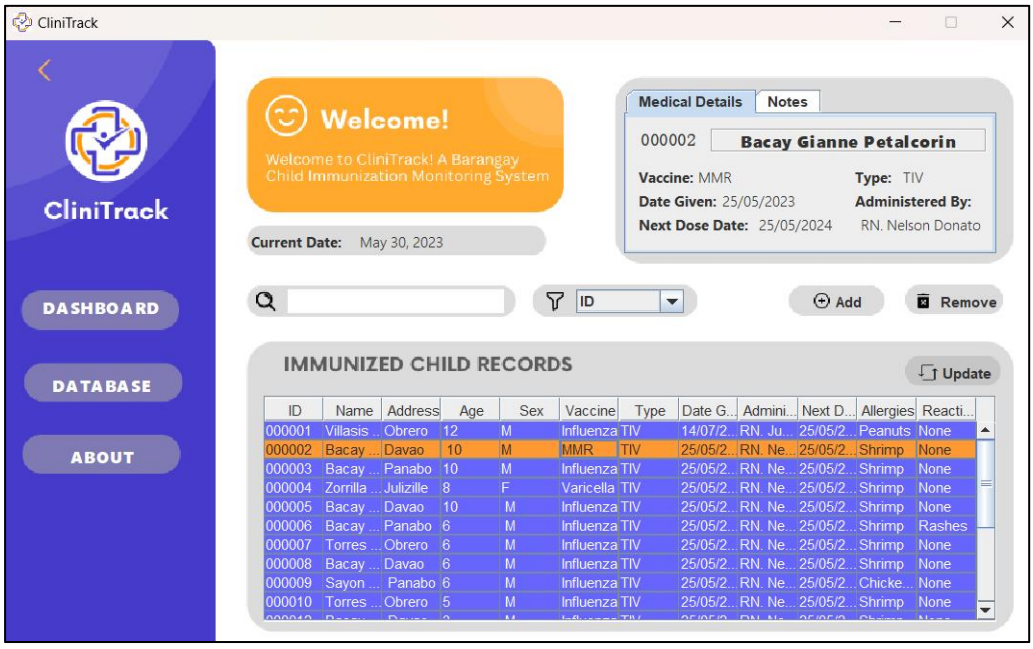
The system has a "Notes" function for healthcare workers to track allergies and reactions to vaccines in children. This feature ensures informed decisions and necessary precautions for future vaccinations.

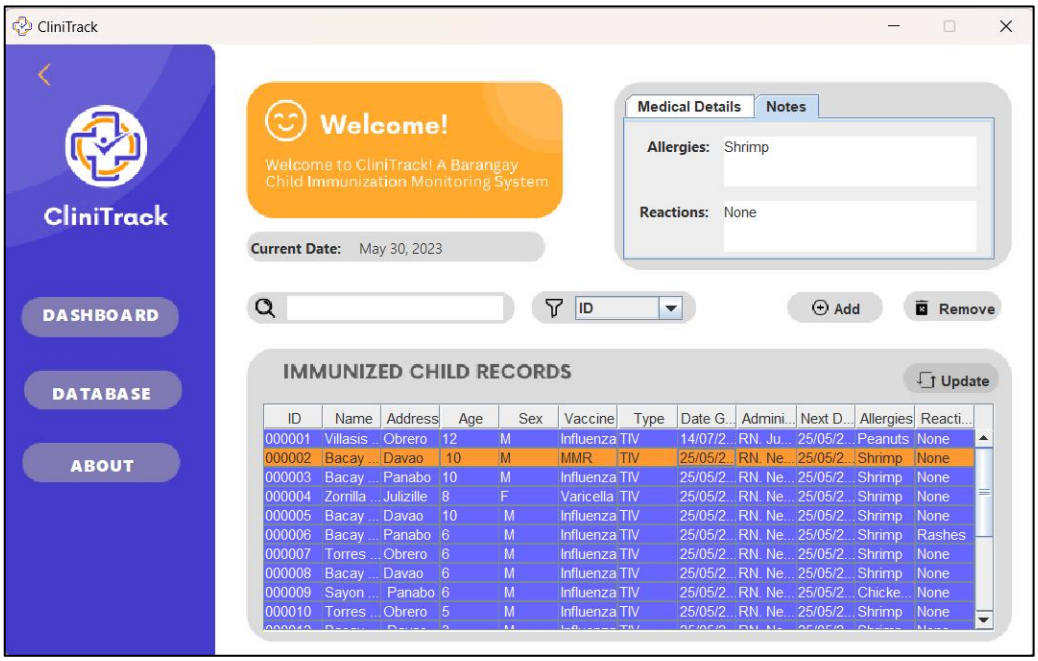
**User Interface Design:**

- **Log in Page**

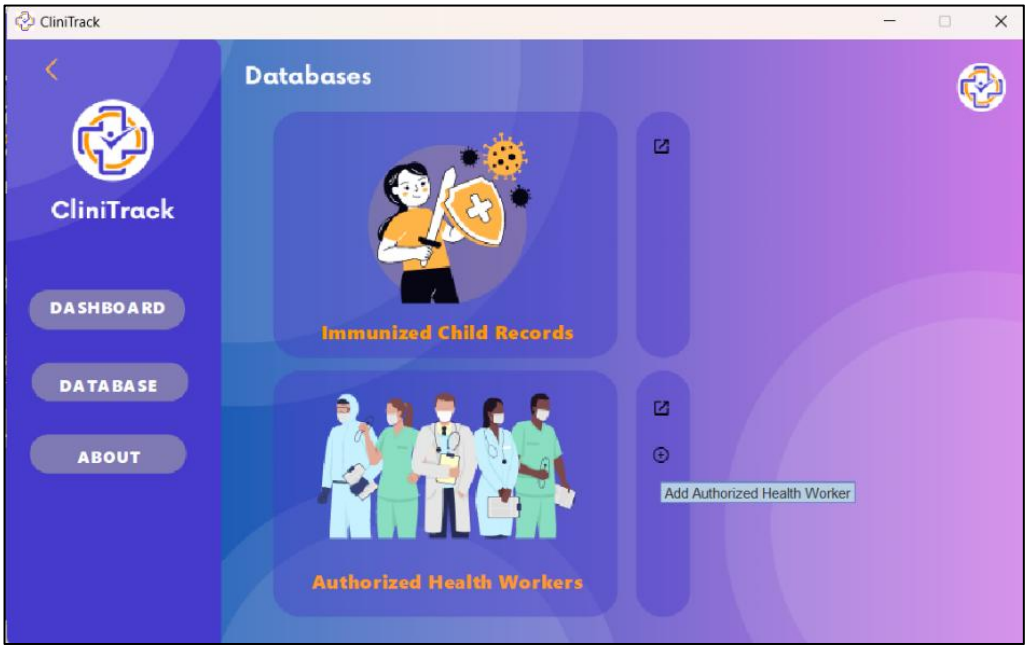



- **Dashboard**

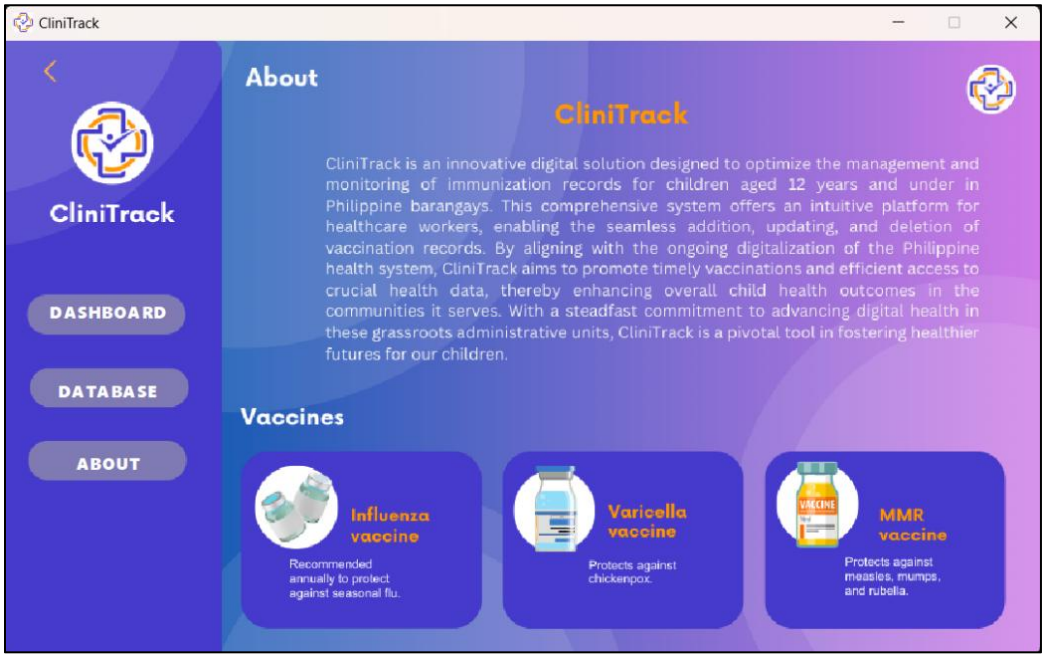





- **Database Page**



## ➢ About Page




## Program Coding:

### Package: Classes

#### Child.java

```
public class Child extends Person{
   private String VaccineName, VaccineType, DateGiven, AdministeredBy, NextDoseDate, Allergies, Reactions;

   public Child() {
      super();
      this.VaccineName = "";
      this.VaccineType = "";
      this.DateGiven = "";
      this.AdministeredBy = "";
      this.NextDoseDate = "";
      this.Allergies = "";
      this.Reactions = "";
   }

   public Child(int ID, String Name, String Address, String Age, String Sex, String VaccineName, String VaccineType, String DateGiven, String AdministeredBy, String NextDoseDate, String Allergies, String Reactions) {
      super(ID, Name, Address, Age, Sex);
      this.VaccineName = VaccineName;
      this.VaccineType = VaccineType;
      this.DateGiven = DateGiven;
      this.AdministeredBy = AdministeredBy;
      this.NextDoseDate = NextDoseDate;
      this.Allergies = Allergies;
      this.Reactions = Reactions;
   }

   //Setters
   public void setVaccineName(String VaccineName) {
      this.VaccineName = VaccineName;
   }

   public void setVaccineType(String VaccineType) {
      this.VaccineType = VaccineType;
   }

   public void setDateGiven(String DateGiven) {
      this.DateGiven = DateGiven;
```

```
    }

    public void setAdministeredBy(String AdministeredBy)
{
        this.AdministeredBy = AdministeredBy;
    }

    public void setNextDoseDate(String NextDoseDate) {
        this.NextDoseDate = NextDoseDate;
    }

    public void setAllergies(String Allergies) {
        this.Allergies = Allergies;
    }

    public void setReactions(String Reactions) {
        this.Reactions = Reactions;
    }


    //Getters

    public String getVaccineName() {
        return VaccineName;
    }

    public String getVaccineType() {
        return VaccineType;
    }

    public String getDateGiven() {
        return DateGiven;
    }

    public String getAdministeredBy() {
        return AdministeredBy;
    }

    public String getNextDoseDate() {
        return NextDoseDate;
    }

    public String getAllergies() {
        return Allergies;
    }

    public String getReactions() {
        return Reactions;
    }
}
```

## HealthWorker.java

```
package Classes;
public class HealthWorker extends Person{
    private String password;

    public HealthWorker() {
        super();
        this.password = "";
    }

    public HealthWorker(int ID, String Name, String
Address, String Age, String Sex, String password) {
        super(ID, Name, Address, Age, Sex);
        this.password = password;
    }

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

    public String getPassword() {
        return password;
    }
}
```

## HealthWorkerCRUD.java

```
public class HealthWorkerCRUD {
    //Database
    private static final String FILEPATHNAME =
"C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\HealthWorker.csv";
    //ArrayList
    public static ArrayList<HealthWorker>
HealthWorkerList = new ArrayList<>();

    //Populate Database
    public static void Populate_Database() {
        File file = new File(FILEPATHNAME);
        try (Scanner input = new Scanner(file)) {
            while (input.hasNextLine()) {
                String[] Line = input.nextLine().split(",");
                String ID = Line[0];
```

```
            String Name = Line[1];
            String Address = Line[2];
            String Age = Line[3];
            String Sex = Line[4];
            String Password = Line[5];
            HealthWorker hW = new
HealthWorker(Integer.parseInt(ID), Name, Address, Age,
Sex, Password);
            HealthWorkerList.add(hW);
          }
        }
        catch (FileNotFoundException e) {
          JOptionPane.showMessageDialog(null, "Database
file not found!");
        }
    }

    //Save to Database
    public static void Save_To_Database() {
      try (PrintWriter writer = new PrintWriter(new
FileWriter(FILEPATHNAME))) {
        for (HealthWorker hW : HealthWorkerList) {
          writer.println(String.format("%06d", hW.getID())
+ "," + hW.getName() + "," + hW.getAddress() + "," +
hW.getAge() + "," + hW.getSex() + "," +
hW.getPassword());
        }
        writer.close();
      }
      catch (IOException e) {
        JOptionPane.showMessageDialog(null, "Failed to
save data to file!");
      }
    }

    //Display Menu
    public static int Display_Menu() {
      String[] options = {"Create", "Retrieve", "Open",
"Exit"};
      return JOptionPane.showOptionDialog(null, "What do
you want to do?", "Health worker CRUD System",
JOptionPane.DEFAULT_OPTION,
JOptionPane.PLAIN_MESSAGE, null, options, options[0])
+ 1;
    }

    //Create
    public static void CreateAccount() {
      String ID = JOptionPane.showInputDialog(null,
"Enter ID number:");
      if(ID == null) {
        return;
      }

      String Name = JOptionPane.showInputDialog(null,
"Enter Name:");
      if(Name == null) {
        return;
      }

      String Address = JOptionPane.showInputDialog(null,
"Enter Address:");
      if(Address == null) {
        return;
      }

      String Age = JOptionPane.showInputDialog(null,
"Enter Age:");
      if(Age == null) {
        return;
      }

      String Sex = JOptionPane.showInputDialog(null,
"Enter Sex:");
      if(Sex == null) {
        return;
      }

      String Password = JOptionPane.showInputDialog(null,
"Enter Password:");
      if(Password == null) {
        return;
      }

      HealthWorker hW = new
HealthWorker(Integer.parseInt(ID), Name, Address, Age,
Sex, Password);
      HealthWorkerList.add(hW);
      Save_To_Database();
      JOptionPane.showMessageDialog(null, "Health worker
successfully added to the database!");
    }

    //Retrieve
    public static void RetrieveAccount() {
      String ID = JOptionPane.showInputDialog(null,
"Enter ID number:");
      if(ID == null) {
        return;
      }

      boolean hWFound = false;
      for (HealthWorker hW : HealthWorkerList) {
        if (ID.equalsIgnoreCase(String.valueOf(hW.getID())))
{
          JOptionPane.showMessageDialog(null, "Health
worker Found!\n\n" +
              "ID: " + hW.getID() + "\n" +
```

```
            "Password: " + hW.getPassword());
          hWFound = true;
          break;
        }
      }
      if (!hWFound && ID != null) {
        JOptionPane.showMessageDialog(null, "Health
worker not found!");
      }
    }
```

```
  //Open
  public static void OpenCSV() {
    File file = new File(FILEPATHNAME);
    try {
      Desktop.getDesktop().open(file);
    }
    catch (IOException e) {
      e.getMessage();
    }
  }
}
```

# MainCRUD.java

```
public class MainCRUD {
   //Database
  private static final String FILEPATHNAME =
"C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\Patients.csv";
  //ArrayList
  public static ArrayList<Child> ChildList = new
ArrayList<>();


//Populate Database
  public static void PopulateDatabase() {
    File file = new File(FILEPATHNAME);
    try (Scanner input = new Scanner(file)) {
      while (input.hasNextLine()) {
        String[] Line = input.nextLine().split(",");
        String ID = Line[0];
        String Name = Line[1];
        String Address = Line[2];
        String Age = Line[3];
        String Sex = Line[4];
        String VaccineName = Line[5];
        String VaccineType = Line[6];
        String DateGiven = Line[7];
        String AdministeredBy = Line[8];
        String NextDoseDate = Line[9];
        String Allergies = Line[10];
        String Reactions = Line[11];
        Child child = new Child(Integer.parseInt(ID), Name,
Address, Age, Sex, VaccineName, VaccineType, DateGiven,
AdministeredBy, NextDoseDate, Allergies, Reactions);
        ChildList.add(child);
      }
    }
    catch (FileNotFoundException e) {
      JOptionPane.showMessageDialog(null, "Database file
not found!");
    }
  }

  public static void SaveToDatabase() {
    try (PrintWriter writer = new PrintWriter(new
FileWriter(FILEPATHNAME))) {
      for (Child patient : ChildList) {
        writer.println(String.format("%06d",
patient.getID()) + "," + patient.getName() + "," +
patient.getAddress() + "," + patient.getAge() + "," +
patient.getSex() + "," +
            patient.getVaccineName() + "," +
patient.getVaccineType() + "," + patient.getVaccineType()
+ "," + patient.getDateGiven() +
patient.getAdministeredBy() + "," +
            patient.getNextDoseDate() + "," +
patient.getAllergies() + "," + patient.getReactions());
      }
      writer.close();
    }
    catch (IOException e) {
      JOptionPane.showMessageDialog(null, "Failed to
save data to file!");
    }
  }

  //Open Immunized Child Records
  public static void OpenCSV() {
    File file = new File(FILEPATHNAME);
    try {
      Desktop.getDesktop().open(file);
    }
    catch (IOException e) {
      e.getMessage();
    }
  }
  }
```

## Person

```
public class Person {
   private int ID;
   private String Name, Address, Age, Sex;

   public Person() {
      this.ID = 000000;
      this.Name = "";
      this.Address = "";
      this.Age = "";
      this.Sex = "";
   }

   public Person(int ID, String Name, String Address,
String Age, String Sex) {
      this.ID = ID;
      this.Name = Name;
      this.Address = Address;
      this.Age = Age;
      this.Sex = Sex;
   }

   //Setters
   public void setID(int ID) {
      this.ID = ID;
   }
   public void setName(String Name) {
     this.Name = Name;
   }
   public void setAddress(String Address) {
     this.Address = Address;
   }
   public void setAge(String Age) {
      this.Age = Age;
   }
   public void setSex(String Sex) {
      this.Sex = Sex;
   }

   //Getters
   public int getID() {
      return ID;
   }
   public String getName() {
     return Name;
   }
   public String getAddress() {
     return Address;
   }
   public String getAge() {
      return Age;
   }
   public String getSex() {
      return Sex;
   }
}
```

# Package: Main

## Main.java

```
public class Main extends javax.swing.JFrame {

   private String LoggedHW;

   public Main() {
      initComponents();
      SimpleDateFormat dateFormat = new
SimpleDateFormat("MMMM d, yyyy");
      String formattedDate = dateFormat.format(new
Date());
      dateLabel.setText(formattedDate);

      LoggedHW = "";
      HealthWorkerCRUD.Populate_Database();
      MainCRUD.PopulateDatabase();
      PopulateTable();

      L0LoginPage.setVisible(true);
      L1Dashboard.setVisible(false);
      L2Database.setVisible(false);
      L3About.setVisible(false);


      jComboBox1.setSelectedIndex(0);

      jTable1.setRowSelectionInterval(0, 0);
      int selectedRow = jTable1.getSelectedRow();
      String Id = jTable1.getValueAt(selectedRow,
0).toString();
      String Name = jTable1.getValueAt(selectedRow,
1).toString();
      String VaccineName =
jTable1.getValueAt(selectedRow, 5).toString();
      String DateGiven = jTable1.getValueAt(selectedRow,
7).toString();
      String NextDoseDate =
```

```
jTable1.getValueAt(selectedRow, 9).toString();
        String VaccineType =
jTable1.getValueAt(selectedRow, 6).toString();
        String AdministeredBy =
jTable1.getValueAt(selectedRow, 8).toString();
        String Allergies = jTable1.getValueAt(selectedRow,
10).toString();
        String Reactions = jTable1.getValueAt(selectedRow,
11).toString();
        jLabel23.setText(Id);
        jLabel12.setText(Name);
        jLabel15.setText(VaccineName);
        jLabel16.setText(DateGiven);
        jLabel17.setText(NextDoseDate);
        jLabel19.setText(VaccineType);
        jLabel20.setText(AdministeredBy);
        jTextArea1.setText(Allergies);
        jTextArea3.setText(Reactions);
        jButton5.setEnabled(false);
        jButton6.setEnabled(false);

        //Table Sorter
        TableRowSorter<TableModel> sorter = new
TableRowSorter<>(jTable1.getModel());
        jTable1.setRowSorter(sorter);

jTextField1.getDocument().addDocumentListener(new
javax.swing.event.DocumentListener() {
            @Override
            public void
changedUpdate(javax.swing.event.DocumentEvent e) {
                filter();
            }
            @Override
            public void
removeUpdate(javax.swing.event.DocumentEvent e) {
                filter();
            }
            @Override
            public void
insertUpdate(javax.swing.event.DocumentEvent e) {
                filter();
            }
            private void filter() {
                String text = jTextField1.getText();
                int columnIndex =
jComboBox1.getSelectedIndex();
                if (text.isEmpty()) {
                    sorter.setRowFilter(null);
                } else {

sorter.setRowFilter(RowFilter.regexFilter("(?i)" + text,
columnIndex));
                }
            }
        });

    }

    //Populate Table
    public static void PopulateTable() {
        DefaultTableModel model = (DefaultTableModel)
jTable1.getModel();
        //Patient_List into patientTable
        for (Child patient : ChildList) {
            int ID = patient.getID();
            String Name = patient.getName();
            String Address = patient.getAddress();
            String Age = patient.getAge();
            String Sex = patient.getSex();
            String VaccineName = patient.getVaccineName();
            String VaccineType = patient.getVaccineType();
            String DateGiven = patient.getDateGiven();
            String AdministeredBy =
patient.getAdministeredBy();
            String NextDoseDate = patient.getNextDoseDate();
            String Allergies = patient.getAllergies();
            String Reactions = patient.getReactions();

            model.addRow(new
Object[]{String.format("%06d", ID), Name, Address, Age,
Sex, VaccineName, VaccineType, DateGiven,
AdministeredBy, NextDoseDate, Allergies, Reactions});
        }
    }

private void
jButton5ActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        DefaultTableModel model = (DefaultTableModel)
jTable1.getModel();
        int rowCount = model.getRowCount();
        int columnCount = model.getColumnCount();

        try {
            try (FileWriter fileWriter = new
FileWriter("C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\Patients.csv")) {
                // Write the data rows
                for (int row = 0; row < rowCount; row++) {
                    for (int column = 0; column < columnCount;
column++) {
                        Object cellValue = model.getValueAt(row,
column);
                        fileWriter.append(cellValue.toString());
                        if (column < columnCount - 1) {
```

```
                    fileWriter.append(",");
                } else {
                    fileWriter.append("\n");
                }
            }
        }fileWriter.flush();
         // Display a message or perform any other
necessary actions after saving
          JOptionPane.showMessageDialog(null, "Records
successfully updated!");
        }
    } catch (IOException e) {
        e.printStackTrace();
        // Handle the exception appropriately
    }
  }
private void
jButton7ActionPerformed(java.awt.event.ActionEvent evt)
{
    // TODO add your handling code here:
    int confirmation =
JOptionPane.showConfirmDialog(null, "Are you sure you
want to log out?", "Confirmation",
JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE);
        if (confirmation == JOptionPane.YES_OPTION) {
            LoadingScreen.setVisible(false);
            L0LoginPage.setVisible(true);
            L1Dashboard.setVisible(false);
            L2Database.setVisible(false);
            L3About.setVisible(false);
        }
        else {}
  }

private void
jButton8ActionPerformed(java.awt.event.ActionEvent evt)
{
    // TODO add your handling code here:
    if (ID.getText().isEmpty() ||
Password.getText().isEmpty()) {
        JOptionPane.showMessageDialog(null, "Please fill
in all fields");
    }
    else {
        String identification = ID.getText().trim();
        String password = Password.getText().trim();

        boolean hWFound = false;
        for (HealthWorker hW : HealthWorkerList) {
            if(String.format("%06d",
hW.getID()).equalsIgnoreCase(identification) &&
password.equals(hW.getPassword())) {
                LoggedHW = hW.getName();
                L0LoginPage.setVisible(false);
                L1Dashboard.setVisible(true);
                L2Database.setVisible(false);
                L3About.setVisible(false);
                hWFound = true;
                break;
            }
        }
        if (hWFound == false) {
            JOptionPane.showMessageDialog(null, "Account
not found! Access denied!");
            ID.setText("");
            Password.setText("");
        }
    }
  }
private void
jButton6ActionPerformed(java.awt.event.ActionEvent evt)
{
    // TODO add your handling code here:
    int row = jTable1.getSelectedRow();

    if (row < 0) {
        JOptionPane.showMessageDialog(this,
            "No row is selected! Please select one row",
            "Select row",
            JOptionPane.ERROR_MESSAGE);
    } else {
        if (row >= 0 && row < ChildList.size()) {
            ChildList.remove(row);
        } else {}
        DefaultTableModel model =
(DefaultTableModel)jTable1.getModel();
        model.removeRow(row);

        int rowCount = model.getRowCount();
        int columnCount = model.getColumnCount();

        try {
            try (FileWriter fileWriter = new
FileWriter("C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\Patients.csv")) {
                // Write the data rows
                for (row = 0; row < rowCount; row++) {
                    for (int column = 0; column < columnCount;
column++) {
                        Object cellValue = model.getValueAt(row,
column);
                        fileWriter.append(cellValue.toString());
                        if (column < columnCount - 1) {
                            fileWriter.append(",");
                        } else {
```

```
                    fileWriter.append("\n");
                }
            }
        }fileWriter.flush();
        // Display a message or perform any other
necessary actions after saving
        JOptionPane.showMessageDialog(null,
"Record successfully deleted!");
        }
      } catch (IOException e) {
        e.printStackTrace();
        // Handle the exception appropriately
      }
    }
  }
private void
jButton21ActionPerformed(java.awt.event.ActionEvent
evt) {
    // TODO add your handling code here:
    DefaultTableModel model = (DefaultTableModel)
jTable1.getModel();
    String prevID = (String)
model.getValueAt(model.getRowCount()-1, 0);
    Child patientToAdd = new
Child(Integer.parseInt(String.format("%06d",
Integer.parseInt(prevID) +01)),null, null, null, null, null,
null, null, LoggedHW, null, null, null);
    ChildList.add(patientToAdd);
    model.addRow(new Object[]{String.format("%06d",
Integer.parseInt(prevID) +01),null, null, null, null, null,
null, null, LoggedHW, null, null, null});
  }

  private void
jTable1MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int selectedRow = jTable1.getSelectedRow();
    String Id = jTable1.getValueAt(selectedRow,
0).toString();
    String Name = jTable1.getValueAt(selectedRow,
1).toString();
    String VaccineName =
jTable1.getValueAt(selectedRow, 5).toString();
    String DateGiven = jTable1.getValueAt(selectedRow,
7).toString();
    String NextDoseDate =
jTable1.getValueAt(selectedRow, 9).toString();
    String VaccineType =
jTable1.getValueAt(selectedRow, 6).toString();
    String AdministeredBy =
jTable1.getValueAt(selectedRow, 8).toString();
    String Allergies = jTable1.getValueAt(selectedRow,
10).toString();
    String Reactions = jTable1.getValueAt(selectedRow,
11).toString();
    jLabel23.setText(Id);
    jLabel12.setText(Name);
    jLabel15.setText(VaccineName);
    jLabel16.setText(DateGiven);
    jLabel17.setText(NextDoseDate);
    jLabel19.setText(VaccineType);
    jLabel20.setText(AdministeredBy);
    jTextArea1.setText(Allergies);
    jTextArea3.setText(Reactions);
    jButton5.setEnabled(true);
    jButton6.setEnabled(true);
  }

  private void
jButton22ActionPerformed(java.awt.event.ActionEvent
evt) {
    // TODO add your handling code here:
    int confirmation =
JOptionPane.showConfirmDialog(null, "Are you sure you
want to log out?", "Confirmation",
JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE);
      if (confirmation == JOptionPane.YES_OPTION) {
        LoadingScreen.setVisible(false);
        L0LoginPage.setVisible(true);
        L1Dashboard.setVisible(false);
        L2Database.setVisible(false);
        L3About.setVisible(false);
      }
      else {}
  }

  private void
jButton24ActionPerformed(java.awt.event.ActionEvent
evt) {
    // TODO add your handling code here:
    int confirmation =
JOptionPane.showConfirmDialog(null, "Are you sure you
want to log out?", "Confirmation",
JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE);
      if (confirmation == JOptionPane.YES_OPTION) {
        LoadingScreen.setVisible(false);
        L0LoginPage.setVisible(true);
        L1Dashboard.setVisible(false);
        L2Database.setVisible(false);
        L3About.setVisible(false);
      }
      else {}
  }

  private void
jTable1MousePressed(java.awt.event.MouseEvent evt) {
```

```
        // TODO add your handling code here:
        int selectedRow = jTable1.getSelectedRow();
        String Id = jTable1.getValueAt(selectedRow,
0).toString();
        String Name = jTable1.getValueAt(selectedRow,
1).toString();
        String VaccineName =
jTable1.getValueAt(selectedRow, 5).toString();
        String DateGiven = jTable1.getValueAt(selectedRow,
7).toString();
        String NextDoseDate =
jTable1.getValueAt(selectedRow, 9).toString();
        String VaccineType =
jTable1.getValueAt(selectedRow, 6).toString();
        String AdministeredBy =
jTable1.getValueAt(selectedRow, 8).toString();
        String Allergies = jTable1.getValueAt(selectedRow,
10).toString();
        String Reactions = jTable1.getValueAt(selectedRow,
11).toString();
        jLabel23.setText(Id);
        jLabel12.setText(Name);
        jLabel15.setText(VaccineName);
        jLabel16.setText(DateGiven);
        jLabel17.setText(NextDoseDate);
        jLabel19.setText(VaccineType);
        jLabel20.setText(AdministeredBy);
        jTextArea1.setText(Allergies);
        jTextArea3.setText(Reactions);
        jButton5.setEnabled(true);
        jButton6.setEnabled(true);
    }

    private void
jTable1KeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        if (evt.isControlDown() && evt.getKeyCode() ==
java.awt.event.KeyEvent.VK_S) {
            DefaultTableModel model = (DefaultTableModel)
jTable1.getModel();
            int rowCount = model.getRowCount();
            int columnCount = model.getColumnCount();

            try {
                try (FileWriter fileWriter = new
FileWriter("C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\Patients.csv")) {
                    // Write the data rows
                    for (int row = 0; row < rowCount; row++) {
                        for (int column = 0; column < columnCount;
column++) {
                            Object cellValue = model.getValueAt(row,
column);
                            fileWriter.append(cellValue.toString());
                            if (column < columnCount - 1) {
                                fileWriter.append(",");
                            } else {
                                fileWriter.append("\n");
                            }
                        }
                    }fileWriter.flush();
                    // Display a message or perform any other
necessary actions after saving
                    JOptionPane.showMessageDialog(null,
"Records successfully updated!");
                }
            } catch (IOException e) {
                e.printStackTrace();
                // Handle the exception appropriately
            }
            // Add your code here
        } else if (evt.getKeyCode() ==
java.awt.event.KeyEvent.VK_ENTER) {
            DefaultTableModel model = (DefaultTableModel)
jTable1.getModel();
            int rowCount = model.getRowCount();
            int columnCount = model.getColumnCount();

            try {
                try (FileWriter fileWriter = new
FileWriter("C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Databases\\Patients.csv")) {
                    // Write the data rows
                    for (int row = 0; row < rowCount; row++) {
                        for (int column = 0; column < columnCount;
column++) {
                            Object cellValue = model.getValueAt(row,
column);
                            fileWriter.append(cellValue.toString());
                            if (column < columnCount - 1) {
                                fileWriter.append(",");
                            } else {
                                fileWriter.append("\n");
                            }
                        }
                    }fileWriter.flush();
                    // Display a message or perform any other
necessary actions after saving
                    JOptionPane.showMessageDialog(null,
"Records successfully updated!");
                }
            } catch (IOException e) {
                e.printStackTrace();
                // Handle the exception appropriately
            }
        }
```

```
    }
public static void main(String args[]) {
        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                Main frame = new Main();
                frame.setResizable(false);
                frame.setTitle("CliniTrack");
                frame.setIconImage(new
ImageIcon("C:\\Users\\Gianne
Bacay\\Documents\\NetBeansProjects\\LE1\\src\\main\\jav
a\\Resources\\Untitled design (11).png").getImage());
                frame.setLocationRelativeTo(null);
                frame.setVisible(true);

            }
        });
    }
```

## References:


1. Labrique, A., Vasudevan, L., Mehl, G., Rosskam, E., & Hyder, A. A. (2018). Digital health and health systems of the future. Global Health: Science and Practice, 6(Supplement 1), S1-S4.

2. Macabasag, R. L. A., Mallari, E. U., Pascual, P. J. C., & Fernandez-Marcelo, P. G. H. (2022). Normalisation of electronic medical records in routine healthcare work amidst ongoing digitalisation of the Philippine health system. Social Science & Medicine, 307, 115182.

3. Macabasag RV, Olaguer MA, Basher A, et al. Digital health transformation in the Philippines: progress and challenges. J Innov Health Inform. 2022;29(2):010. doi:10.14236/jhi.v29i2.010

4. UNICEF. Progress for Every Child in the SDG Era. 2022. Available at: https://www.unicef.org/media/94816/file/Progress-for-Every-Child-in-the-SDG-Era-2022.pdf. Accessed on 26 May 2023.

5. UNICEF. (2022). The Importance of Vaccination Tracking Systems. Retrieved from https://data.unicef.org/topic/child-health/immunization