

FINAL REPORT

1. Problem Definition & Requirements

Problem Statement

The Vaccination Portal is designed to address the need for efficient and organized tracking of vaccination records for patients. With increasing demand for accessible healthcare data, this system ensures secure storage, retrieval, and management of information related to patients' vaccinations, providers, and insurance. It aims to eliminate redundancy, streamline operations, and enhance data accuracy for better decision-making.

Requirements Gathering

The Vaccination Portal includes the following functional requirements:

1. **User Login and Authentication:**
 - Secure access to the portal for authorized users.
 - Login credentials are required to access patient data.
2. **Patient Record Management:**
 - Ability to add new patient records, update existing ones, and delete incorrect data.
 - Store demographic details such as name, address, gender, race, and ethnicity.
3. **Vaccination Data Management:**
 - Record vaccination details, including type, dose, manufacturer, and administration site.
 - Include vaccination outcomes such as missed appointments or adverse reactions.
4. **Provider and Insurance Information:**
 - Track vaccination providers' details (name, address, and contact information).
 - Maintain insurance information for patients.
5. **Data Retrieval:**
 - Allow users to query patient records based on criteria like name, date of birth, or vaccination type.
6. **Comprehensive Reporting:**
 - Generate reports to display aggregated vaccination data.

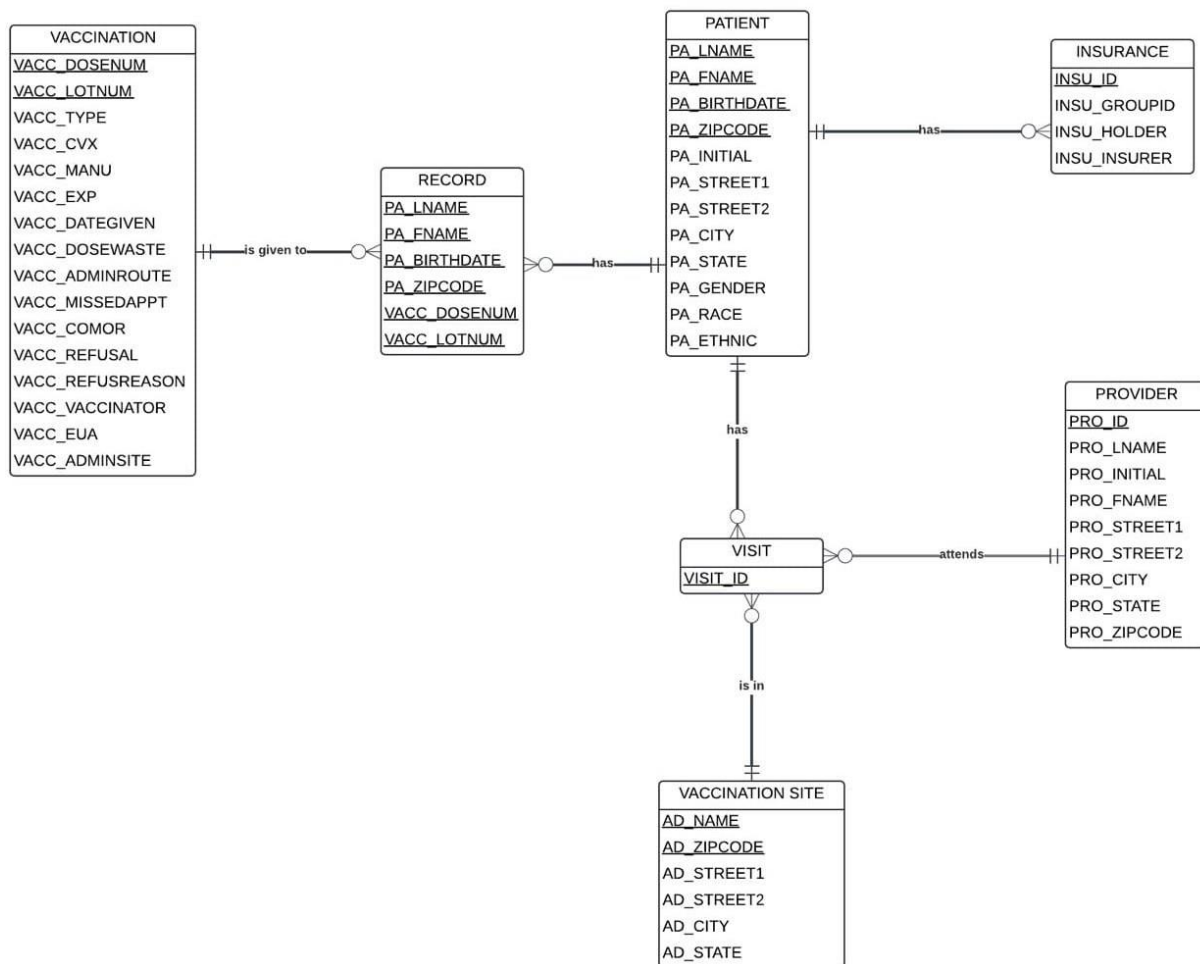
Scope and Feasibility

- **Scope:**
 - The project focuses on vaccination record management for healthcare providers and patients. It includes functionalities such as patient record management, vaccination tracking, and insurance provider linkage.
 - Supports CRUD operations for all entities and ensures the integrity of relationships between patients, providers, and vaccination sites.
- **Feasibility:**
 - Technologically feasible using a relational database (e.g., MySQL or PostgreSQL) and a front-end framework (e.g., React or Angular).
 - Scalable design to include new features like real-time updates or integration with national vaccination registries.

2. Database Design

Entity-Relationship Diagram (ERD)

The database is designed using a relational schema to represent patients, vaccination records, providers, and associated data. Below is the ER diagram for the Vaccination Portal:



Description of Entities:

1. **Patient:**
 - Stores details like name, date of birth, gender, race, and address.
2. **Vaccination:**
 - Tracks details like vaccine type, dose number, manufacturer, and administration site.
3. **Insurance:**
 - Maintains information about patient insurance, including the holder and insurer details.
4. **Provider:**
 - Stores information about vaccination providers such as name, address, and contact information.
5. **Visit:**
 - Links patients to vaccination providers during specific visits.
6. **Vaccination Site:**

- Captures location details for where vaccinations are administered.

7. **Record:**

- Keeps the records of all the vaccinations and their doses

Entities and Attributes

1. Patient Table

- **Primary Key:** PA_ID (unique identifier for each patient)
- **Attributes:**
 - PA_LNAME (Last Name)
 - PA_FNAME (First Name)
 - PA_INITIAL (Middle Name/Initial)
 - PA_BIRTHDATE (Date of Birth)
 - PA_GENDER (Gender)
 - PA_RACE (Race)
 - PA_ETHNIC (Ethnicity)
 - PA_STREET1, PA_STREET2 (Address Lines)
 - PA_CITY, PA_STATE, PA_ZIPCODE (City, State, and Zip Code)

2. Vaccination Table

- **Primary Key:** (VACC_LOTNUM, VACC_DOSENUM)
(Composite Key to uniquely identify each vaccination record.)
- **Attributes:**
 - VACC_TYPE (e.g., COVID-19, Polio)
 - VACC_CVX (CVX Code)
 - VACC_MANU (Manufacturer)
 - VACC_EXP (Expiration Date)
 - VACC_DATEGIVEN (Date Administered)
 - VACC_DOSEWASTE (Number of Wasted Doses)
 - VACC_ADMINROUTE (Route of Administration)
 - VACC_COMOR (Comorbidity details)
 - VACC_REFUSAL (Refusal Status)
 - VACC_REFUSEREASON (Reason for Refusal)
 - VACC_VACCINATOR (Vaccinator's Name)
 - VACC_EUA (EUA Fact Sheet Provided)
 - VACC_ADMINSITE (Administration Site)

3. Provider Table

- **Primary Key:** PRO_ID (unique identifier for each provider)
- **Attributes:**
 - PRO_LNAME, PRO_FNAME, PRO_INITIAL (Provider's Name Details)
 - PRO_STREET1, PRO_STREET2 (Address Lines)
 - PRO_CITY, PRO_STATE, PRO_ZIPCODE (City, State, and Zip Code)

4. Insurance Table

- **Primary Key:** INSU_ID
- **Attributes:**
 - INSU_GROUPID (Group ID of the insurance plan)
 - INSU_HOLDER (Insurance Holder Name)
 - INSU_INSURER (Name of the Insurance Provider)

Relationships:

- **Foreign Key:** PA_ID links to Patient.PA_ID (to associate insurance details with a specific patient).

5. Visit Table

- **Primary Key:** VISIT_ID
- **Attributes:**
 - PA_ID (Foreign Key from the Patient Table)
 - PRO_ID (Foreign Key from the Provider Table)
 - VACC_LOTNUM (Foreign Key from the Vaccination Table)
 - VACC_DOSENUM (Foreign Key from the Vaccination Table)

Relationships:

- PA_ID → Patient.PA_ID

- PRO_ID → Provider.PRO_ID
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

6. Vaccination Site Table

- **Primary Key:** AD_ID
- **Attributes:**
 - AD_NAME (Name of the Vaccination Site)
 - AD_STREET1, AD_STREET2 (Address Lines)
 - AD_CITY, AD_STATE, AD_ZIPCODE (City, State, and Zip Code)

Relationships:

- VISIT_ID → Visit.VISIT_ID (Links the vaccination site to the specific visit where the vaccination was administered.)

7. Record Table

- **Primary Key:** (PA_FNAME, PA_LNAME, PA_BIRTHDATE, PA_ZIPCODE, VACC_LOTNUM, VACC_DOSENUM)
(Composite key used to uniquely identify a patient vaccination record.)
- **Attributes:**
 - PA_FNAME, PA_LNAME, PA_BIRTHDATE, PA_ZIPCODE (Linked to Patient details)
 - VACC_LOTNUM, VACC_DOSENUM (Linked to Vaccination details)

Relationships:

- PA_FNAME, PA_LNAME, PA_BIRTHDATE, PA_ZIPCODE → Patient.PA_FNAME, Patient.PA_LNAME, Patient.PA_BIRTHDATE, Patient.PA_ZIPCODE
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

Normalization

Step 1: First Normal Form (1NF)

1. Ensure all attributes are atomic (contain only a single value).
2. Remove repeating groups.

Changes:

- Split composite attributes (e.g., PA_NAME → PA_FNAME, PA_LNAME, etc.).
- Separate multivalued attributes (e.g., vaccination details like VACC_TYPE, VACC_LOTNUM, etc.) into their own rows.

Step 2: Second Normal Form (2NF)

1. Remove partial dependencies (attributes depending only on part of a composite primary key).
2. Ensure all attributes depend on the entire primary key.

Changes:

- Split Record table to avoid partial dependency. Moved VACC_LOTNUM and VACC_DOSENUM to the Vaccination table.
- Moved insurance attributes (INSU_GROUPID, INSU_HOLDER, etc.) into a separate Insurance table linked to Patient via PA_ID.
- Created Provider and Visit tables for relationships between patients and vaccination providers.

Step 3: Third Normal Form (3NF)

1. Remove transitive dependencies (non-prime attributes depending on non-key attributes).
2. Ensure all attributes depend directly on the primary key.

Changes:

- Moved vaccination site details (e.g., AD_NAME, AD_ZIPCODE) to a separate Vaccination_Site table linked to Visit.
- Ensured Vaccination table only contains attributes directly related to vaccination (e.g., VACC_LOTNUM, VACC_TYPE).
- All tables now adhere to 3NF.

Final Normalized Schema

Below is the **final schema** after normalization, with foreign keys clearly specified:

1. Patient Table

- **Primary Key:** PA_ID (Unique identifier for each patient).
- **Attributes:**
 - PA_LNAME (Last Name)
 - PA_FNAME (First Name)
 - PA_INITIAL (Middle Name/Initial)
 - PA_BIRTHDATE (Date of Birth)
 - PA_GENDER, PA_RACE, PA_ETHNIC
 - PA_STREET1, PA_STREET2 (Address Lines)
 - PA_CITY, PA_STATE, PA_ZIPCODE

2. Vaccination Table

- **Primary Key:** (VACC_LOTNUM, VACC_DOSENUM)
(Composite Key to uniquely identify each vaccination record.)
- **Attributes:**
 - VACC_TYPE (e.g., COVID-19, Polio)
 - VACC_CVX (CVX Code)
 - VACC_MANU (Manufacturer)
 - VACC_EXP (Expiration Date)
 - VACC_DATEGIVEN (Date Administered)
 - VACC_DOSEWASTE (Number of Wasted Doses)

- VACC_ADMINROUTE (Route of Administration)
- VACC_REFUSAL, VACC_REFUSEREASON
- VACC_VACCINATOR (Vaccinator Name)

3. Provider Table

- **Primary Key:** PRO_ID (Unique identifier for each provider).
- **Attributes:**
 - PRO_LNAME, PRO_FNAME, PRO_INITIAL (Provider's Name Details)
 - PRO_STREET1, PRO_STREET2 (Address Lines)
 - PRO_CITY, PRO_STATE, PRO_ZIPCODE (City, State, Zip Code)

4. Insurance Table

- **Primary Key:** INSU_ID
- **Attributes:**
 - INSU_GROUPID (Group ID of the insurance plan)
 - INSU_HOLDER (Insurance Holder Name)
 - INSU_INSURER (Name of the Insurance Provider)
- **Foreign Key:** PA_ID → Patient.PA_ID (Links insurance details to a specific patient).

5. Visit Table

- **Primary Key:** VISIT_ID
- **Attributes:**
 - PA_ID (Foreign Key from the Patient Table)
 - PRO_ID (Foreign Key from the Provider Table)
 - VACC_LOTNUM (Foreign Key from the Vaccination Table)
 - VACC_DOSENUM (Foreign Key from the Vaccination Table)

Foreign Keys:

- PA_ID → Patient.PA_ID
- PRO_ID → Provider.PRO_ID
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

6. Vaccination Site Table

- **Primary Key:** AD_ID
- **Attributes:**
 - AD_NAME (Name of the Vaccination Site)
 - AD_STREET1, AD_STREET2 (Address Lines)
 - AD_CITY, AD_STATE, AD_ZIPCODE (City, State, Zip Code)
- **Foreign Key:** VISIT_ID → Visit.VISIT_ID (Links the vaccination site to the visit where the vaccination was administered).

7. Record Table

- **Primary Key:** (PA_ID, VACC_LOTNUM, VACC_DOSENUM)
(Composite key uniquely identifying a patient's vaccination record.)
- **Attributes:**
 - PA_ID (Foreign Key linking to Patient.PA_ID)
 - VACC_LOTNUM, VACC_DOSENUM (Foreign Keys linking to Vaccination)

Foreign Keys:

- PA_ID → Patient.PA_ID
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

Summary of Foreign Keys

1. **Insurance Table:**
 - PA_ID → Patient.PA_ID

2. Visit Table:

- PA_ID → Patient.PA_ID
- PRO_ID → Provider.PRO_ID
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

3. Vaccination Site Table:

- VISIT_ID → Visit.VISIT_ID

4. Record Table:

- PA_ID → Patient.PA_ID
- VACC_LOTNUM, VACC_DOSENUM → Vaccination.VACC_LOTNUM, Vaccination.VACC_DOSENUM

3. Implementation (30 points)

SQL Queries

- Overview: The implementation involves designing SQL queries to handle data operations effectively. These include creating tables, inserting new records, updating existing data, retrieving information, and deleting records as necessary.
- Purpose: The queries ensure data consistency, support project functionalities, and demonstrate how the database interacts with the application.
- Focus Areas: Queries address CRUD (Create, Read, Update, Delete) operations to manage vaccination records, patient information, insurance details, and provider relationships.

Database Table Creation

- Tables for major entities like Patient, Vaccination, Provider, Insurance, Visit, and Vaccination_Site are created, with primary keys and foreign key relationships ensuring data integrity.
- Normalization techniques were applied to avoid redundancy and improve scalability.

CRUD Operations

- CRUD functionalities were implemented to:
- Create: Add records for patients, vaccinations, and providers.
- Read: Retrieve vaccination details for specific patients or providers.
- Update: Modify existing records, such as vaccination statuses or patient addresses.
- Delete: Remove obsolete or incorrect records.

Data Integrity and Constraints

- Primary Keys: Ensure unique identification of each record across tables.
- Foreign Keys: Establish relationships between entities (e.g., a patient linked to their vaccination through the Visit table).

Constraints:

- NOT NULL constraints for mandatory fields.
- CHECK constraints for data validation (e.g., valid gender values, date ranges).
- Composite keys for uniquely identifying complex records (e.g., vaccination dose and lot number).

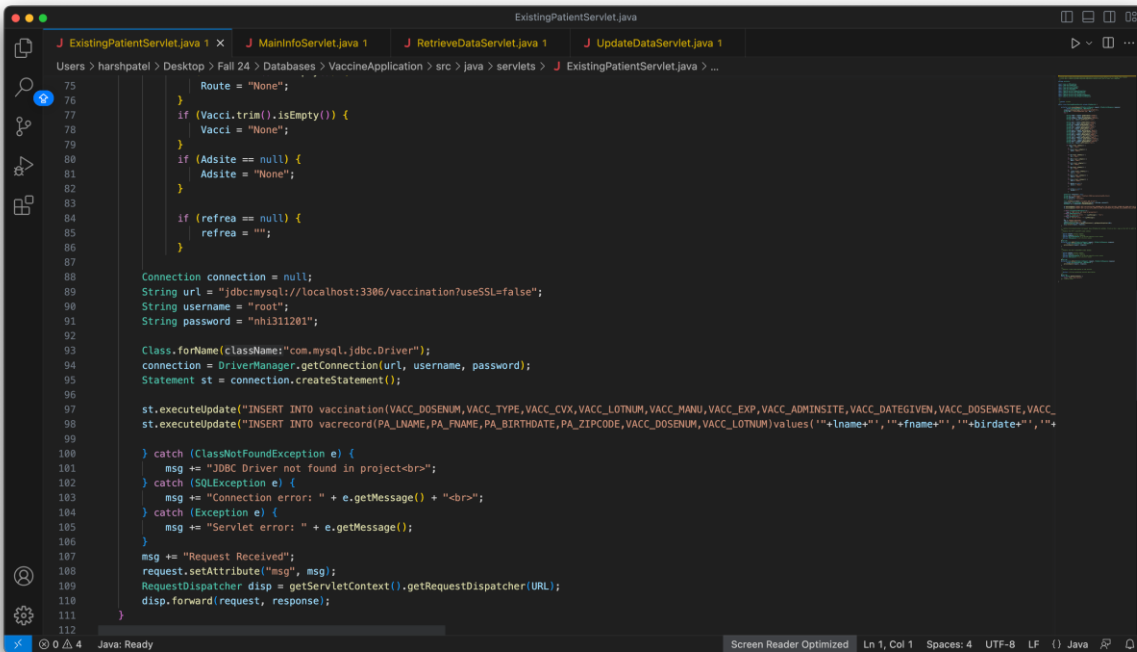
Testing and Validation

- Mock Data:
- Representative patient and vaccination records were used to test database functionality.
- Scenarios included valid operations, edge cases, and constraint violations.
- Validation:
- Ensured proper handling of cascading updates and deletions using foreign key constraints.
- Verified that all queries produced accurate and expected outputs.

Screenshots and Evidence

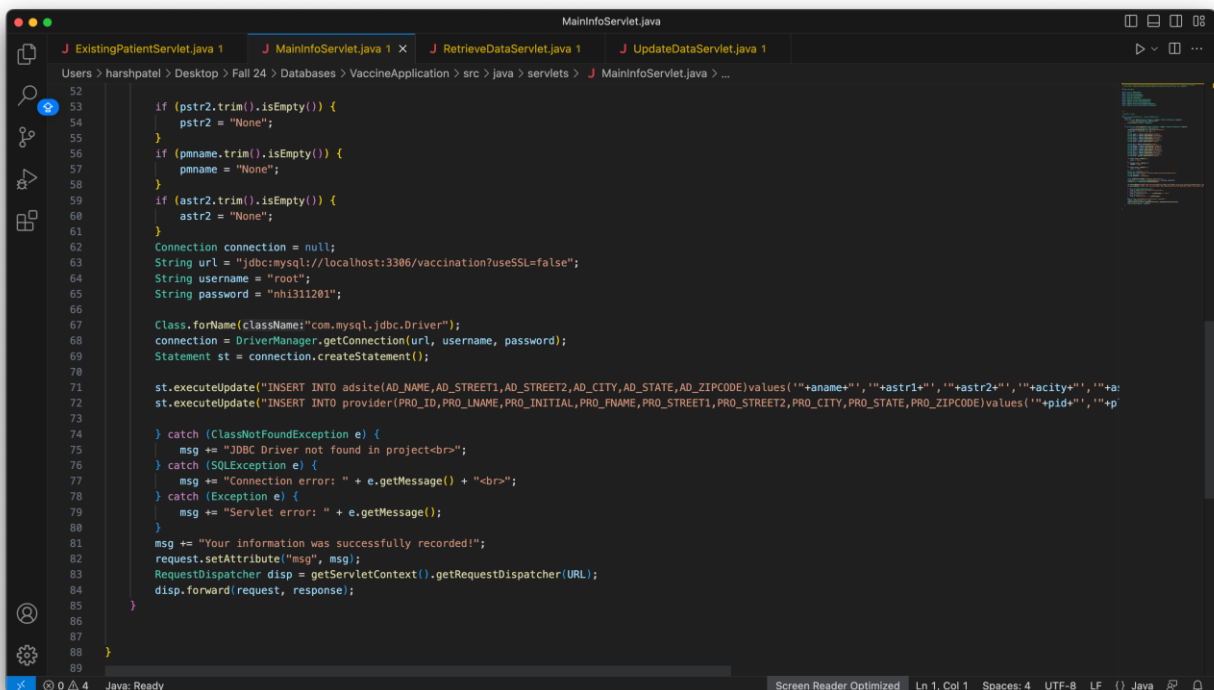
Screenshots capture:

EXISTING PATIENT SERVLET :



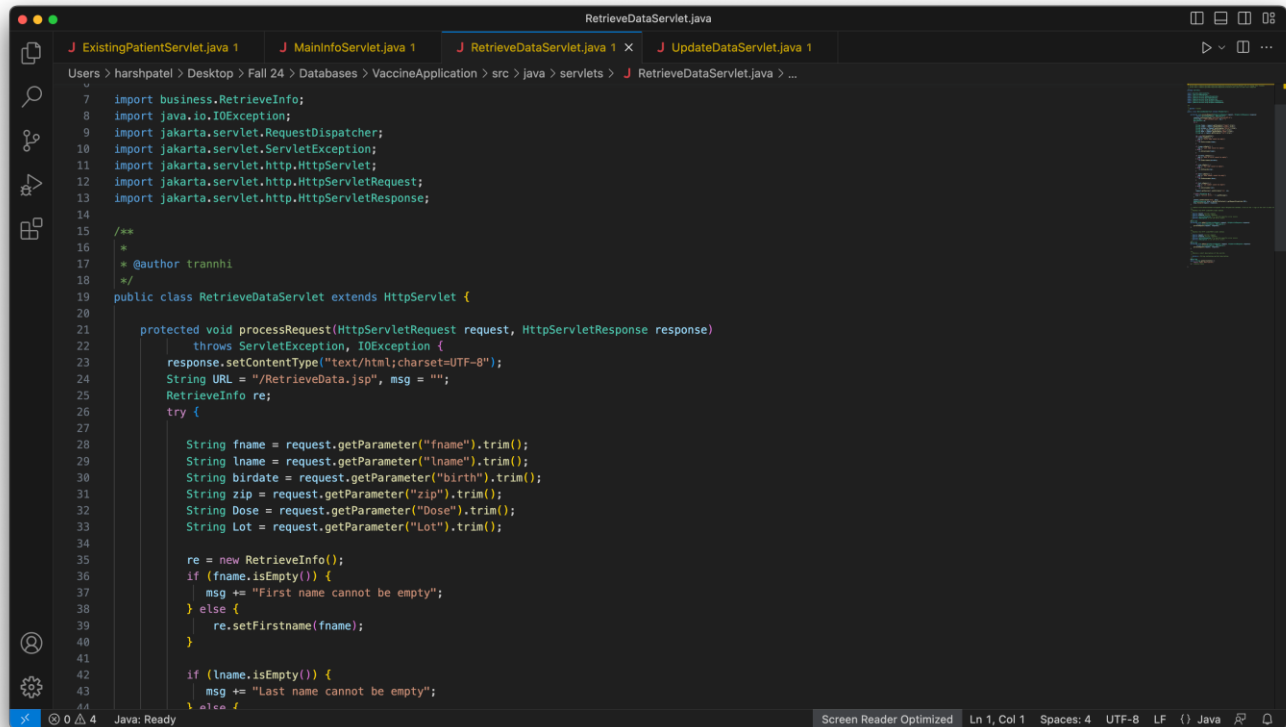
```
ExistingPatientServlet.java
J ExistingPatientServlet.java 1 X J MainInfoServlet.java 1 J RetrieveDataServlet.java 1 J UpdateDataServlet.java 1
Users > harshpatel > Desktop > Fall 24 > Databases > VaccineApplication > src > java > servlets > J ExistingPatientServlet.java > ...
75 Route = "None";
76 }
77 if (Vacci.trim().isEmpty()) {
78     Vacci = "None";
79 }
80 if (Adsite == null) {
81     Adsite = "None";
82 }
83
84 if (refrea == null) {
85     refrea = "";
86 }
87
88 Connection connection = null;
89 String url = "jdbc:mysql://localhost:3306/vaccination?useSSL=false";
90 String username = "root";
91 String password = "nh1311201";
92
93 Class.forName(className:"com.mysql.jdbc.Driver");
94 connection = DriverManager.getConnection(url, username, password);
95 Statement st = connection.createStatement();
96
97 st.executeUpdate("INSERT INTO vaccination(VACC_DOSENUM, VACC_TYPE, VACC_CVX, VACC_LOTNUM, VACC_MANU, VACC_EXP, VACC_ADMINSITE, VACC_DATEGIVEN, VACC_DOSEWASTE, VACC_
98 st.executeUpdate("INSERT INTO vacrecord(PA_LNAME, PA_FNAME, PA_BIRTHDATE, PA_ZIPCODE, VACC_DOSENUM, VACC_LOTNUM) values('"+lname+"','"+fname+"','"+birthdate+"','"+
99
100 } catch (ClassNotFoundException e) {
101     msg += "JDBC Driver not found in project<br>";
102 } catch (SQLException e) {
103     msg += "Connection error: " + e.getMessage() + "<br>";
104 } catch (Exception e) {
105     msg += "Servlet error: " + e.getMessage();
106 }
107
108 msg += "Request Received";
109 request.setAttribute("msg", msg);
110 RequestDispatcher disp = getServletContext().getRequestDispatcher(URL);
111 disp.forward(request, response);
112
113 }
114
115 }
Java: Ready Screen Reader Optimized Ln 1, Col 1 Spaces: 4 UTF-8 LF {} Java
```

MAIN INFO SERVLET :



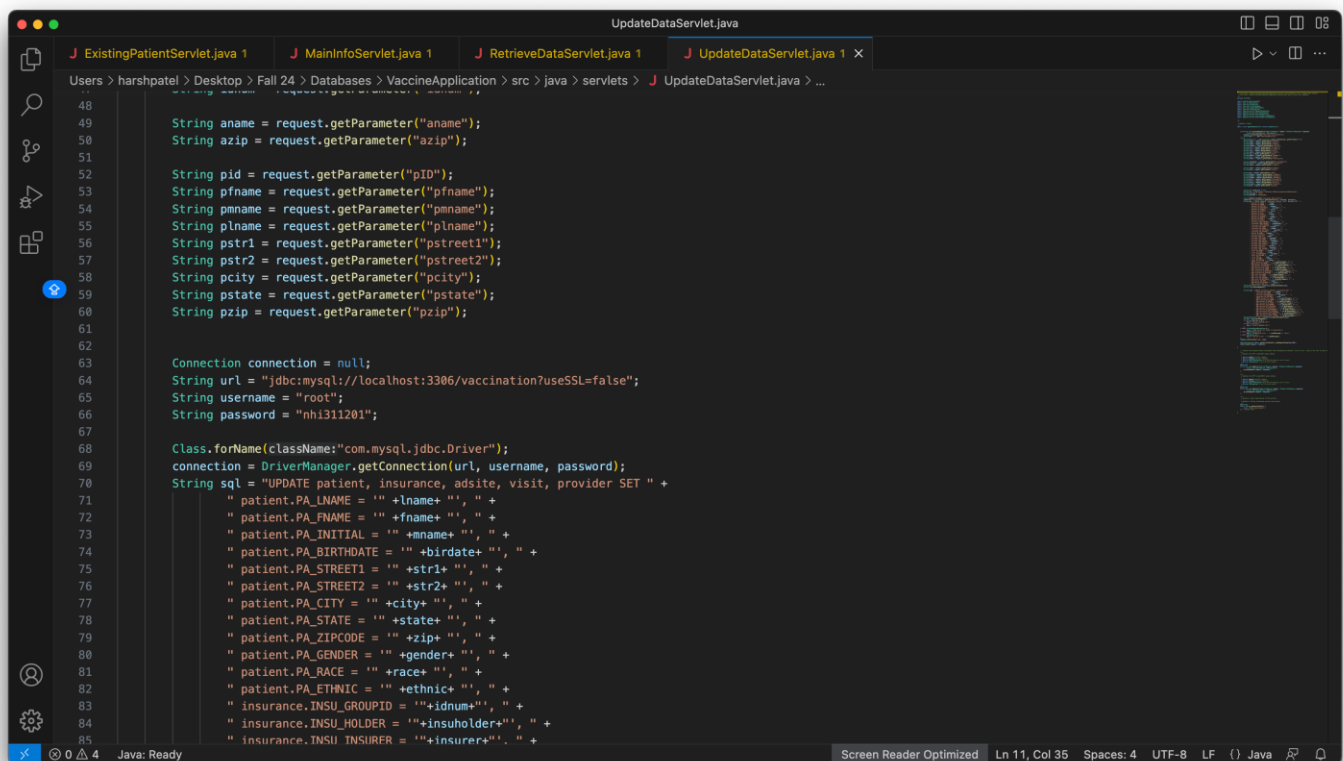
```
MainInfoServlet.java
J ExistingPatientServlet.java 1 J MainInfoServlet.java 1 X J RetrieveDataServlet.java 1 J UpdateDataServlet.java 1
Users > harshpatel > Desktop > Fall 24 > Databases > VaccineApplication > src > java > servlets > J MainInfoServlet.java > ...
52
53 if (pstr2.trim().isEmpty()) {
54     pstr2 = "None";
55 }
56 if (pmname.trim().isEmpty()) {
57     pmname = "None";
58 }
59 if (astr2.trim().isEmpty()) {
60     astr2 = "None";
61 }
62
63 Connection connection = null;
64 String url = "jdbc:mysql://localhost:3306/vaccination?useSSL=false";
65 String username = "root";
66 String password = "nh1311201";
67
68 Class.forName(className:"com.mysql.jdbc.Driver");
69 connection = DriverManager.getConnection(url, username, password);
70 Statement st = connection.createStatement();
71
72 st.executeUpdate("INSERT INTO adsite(AD_NAME, AD_STREET1, AD_STREET2, AD_CITY, AD_STATE, AD_ZIPCODE) values('"+aname+"','"+astr1+"','"+astr2+"','"+acity+"','"+a
73 st.executeUpdate("INSERT INTO provider(PRO_ID, PRO_LNAME, PRO_INITIAL, PRO_FNAME, PRO_STREET1, PRO_STREET2, PRO_CITY, PRO_STATE, PRO_ZIPCODE) values('"+pid+"','"+p
74
75 } catch (ClassNotFoundException e) {
76     msg += "JDBC Driver not found in project<br>";
77 } catch (SQLException e) {
78     msg += "Connection error: " + e.getMessage() + "<br>";
79 } catch (Exception e) {
80     msg += "Servlet error: " + e.getMessage();
81 }
82
83 msg += "Your information was successfully recorded!";
84 request.setAttribute("msg", msg);
85 RequestDispatcher disp = getServletContext().getRequestDispatcher(URL);
86 disp.forward(request, response);
87
88 }
89
90 }
Java: Ready Screen Reader Optimized Ln 1, Col 1 Spaces: 4 UTF-8 LF {} Java
```

RETRIEVE DATA SERVLET :



```
RetrieveDataServlet.java
J ExistingPatientServlet.java 1 J MainInfoServlet.java 1 J RetrieveDataServlet.java 1 X J UpdateDataServlet.java 1
Users > harshpatel > Desktop > Fall 24 > Databases > VaccineApplication > src > java > servlets > J RetrieveDataServlet.java > ...
7 import business.RetrieveInfo;
8 import java.io.IOException;
9 import jakarta.servlet.RequestDispatcher;
10 import jakarta.servlet.ServletException;
11 import jakarta.servlet.http.HttpServlet;
12 import jakarta.servlet.http.HttpServletRequest;
13 import jakarta.servlet.http.HttpServletResponse;
14
15 /**
16  *
17  * @author tranhhi
18  */
19 public class RetrieveDataServlet extends HttpServlet {
20
21     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
22         throws ServletException, IOException {
23         response.setContentType("text/html;charset=UTF-8");
24         String URL = "/RetrieveData.jsp", msg = "";
25         RetrieveInfo re;
26         try {
27
28             String fname = request.getParameter("fname").trim();
29             String lname = request.getParameter("lname").trim();
30             String birthdate = request.getParameter("birthdate").trim();
31             String zip = request.getParameter("zip").trim();
32             String Dose = request.getParameter("Dose").trim();
33             String Lot = request.getParameter("Lot").trim();
34
35             re = new RetrieveInfo();
36             if (fname.isEmpty()) {
37                 msg += "First name cannot be empty";
38             } else {
39                 re.setFirstname(fname);
40             }
41
42             if (lname.isEmpty()) {
43                 msg += "Last name cannot be empty";
44             } else {
45                 re.setLastname(lname);
46             }
47
48             re.setBirthdate(birthdate);
49             re.setZip(zip);
50             re.setDose(Dose);
51             re.setLot(Lot);
52
53             RequestDispatcher dispatcher = request.getRequestDispatcher(URL);
54             dispatcher.forward(request, response);
55         } catch (Exception ex) {
56             msg += ex.getMessage();
57         }
58     }
59 }
Java: Ready Screen Reader Optimized Ln 1, Col 1 Spaces: 4 UTF-8 LF {} Java
```

UPDATE DATA SERVLET :



```
UpdateDataServlet.java
J ExistingPatientServlet.java 1 J MainInfoServlet.java 1 J RetrieveDataServlet.java 1 J UpdateDataServlet.java 1 X
Users > harshpatel > Desktop > Fall 24 > Databases > VaccineApplication > src > java > servlets > J UpdateDataServlet.java > ...
48
49 String aname = request.getParameter("aname");
50 String azip = request.getParameter("azip");
51
52 String pid = request.getParameter("pid");
53 String pname = request.getParameter("pname");
54 String pmname = request.getParameter("pmname");
55 String plname = request.getParameter("plname");
56 String pstr1 = request.getParameter("pstreet1");
57 String pstr2 = request.getParameter("pstreet2");
58 String pcity = request.getParameter("pcity");
59 String pstate = request.getParameter("pstate");
60 String pzip = request.getParameter("pzip");
61
62
63 Connection connection = null;
64 String url = "jdbc:mysql://localhost:3306/vaccination?useSSL=false";
65 String username = "root";
66 String password = "nh1311201";
67
68 Class.forName(className:"com.mysql.jdbc.Driver");
69 connection = DriverManager.getConnection(url, username, password);
70 String sql = "UPDATE patient, insurance, adsite, visit, provider SET " +
71     " patient.PA_LNAME = '" + lname + "', " +
72     " patient.PA_FNAME = '" + fname + "', " +
73     " patient.PA_INITIAL = '" + mname + "', " +
74     " patient.PA_BIRTHDATE = '" + birthdate + "', " +
75     " patient.PA_STREET1 = '" + str1 + "', " +
76     " patient.PA_STREET2 = '" + str2 + "', " +
77     " patient.PA_CITY = '" + city + "', " +
78     " patient.PA_STATE = '" + state + "', " +
79     " patient.PA_ZIPCODE = '" + zip + "', " +
80     " patient.PA_GENDER = '" + gender + "', " +
81     " patient.PA_RACE = '" + race + "', " +
82     " patient.PA_ETHNIC = '" + ethnic + "', " +
83     " insurance.INSU_GROUPID = '" + idnum + "', " +
84     " insurance.INSU_HOLDER = '" + insuholder + "', " +
85     " insurance.INSU_INSURER = '" + insurer + "'";
Java: Ready Screen Reader Optimized Ln 11, Col 35 Spaces: 4 UTF-8 LF {} Java
```

Tools and Technologies

- SQL for database design and management.
- MySQL Workbench for schema creation and testing.
- NetBeans IDE for integration with application logic.

Key Outcomes

- The database supports seamless management of patient records, vaccination history, and associated entities.
- Successfully integrated security features and validation mechanisms ensure robust performance and scalability.

4. Application Integration (15 Points)

User Interface

- Overview: The application provides an interactive and user-friendly interface to interact with the database, enabling healthcare providers to manage vaccination records efficiently.

Key Features:

- Secure Login Page for provider authentication.
- Main Dashboard with intuitive navigation for accessing core functionalities.

Forms segmented into sections for:

- Patient information.
- Insurance details.
- Vaccination records.
- Search and Query Interface: Simplified retrieval of data using patient identifiers or vaccination details.

Integration Details

Backend Tools:

- Built using Java Servlets for dynamic server-side processing.
- MySQL database integration to support CRUD operations and secure data handling.

Frontend Tools:

- Developed using HTML, CSS, and JavaScript for responsive design.
- JSP (JavaServer Pages) for dynamic content rendering.

Functionality

CRUD Operations:

- Create: Add new patient records, insurance details, and vaccination data.
- Read: Retrieve and display patient vaccination history and associated details.
- Update: Modify patient or vaccination information as needed.
- Delete: Remove outdated or incorrect entries securely.

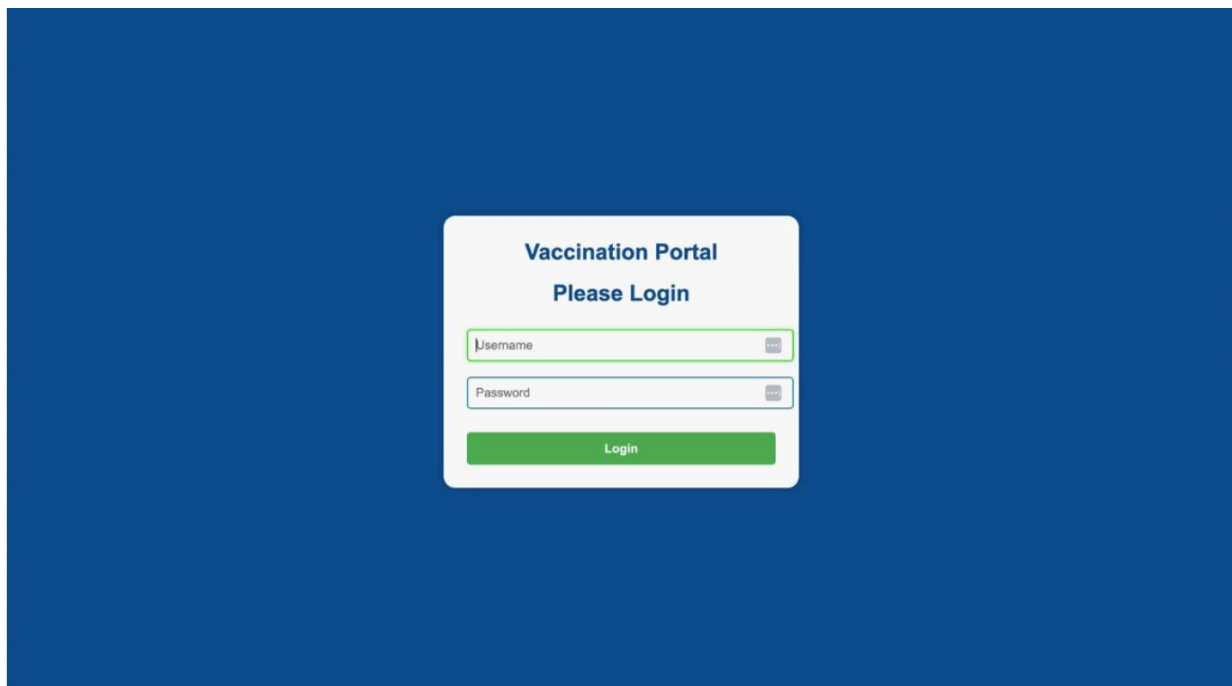
Validation and Security:

- Real-time validation ensures data accuracy during form submissions.
- Secure login mechanism prevents unauthorized access.

Screenshots

Include screenshots to demonstrate:

- Login interface.



- Dashboard navigation.

COVID-19 Vaccination Tracking Application

[Profile](#)[New Patient](#)[Existing Patient](#)[Retrieve Data](#)[Log Out](#)

Account Information:

Administration Information:

The Administration Name*:

Street 1*:

Street 2*:

City*:

State*:
MO

Zip Code*:
nnnnn(-nnnn)

Provider Information:

Provider ID*:
0XXXXX

First Name*:

Initial:

Last Name*:

Does the provider have the same address?
☐ Yes ☒ No

Street 1*:

Street 2*:

City*:

State*:
MO

Zip Code*:
nnnnn(-nnnn)

- Form inputs for adding, updating, and retrieving records.

Adding New Patient :

Adding A New Record for New Patients

Profile
New Patient
Existing Patient
Retrieve Data
Log Out

1st Form:

Patient Information

First Name*:

Initial:

Last Name*:

Date of Birth*:

Street 1*:

Street 2:

City*:

State*: MO

Zip Code*:

Gender*: ☒ Male ☐ Female ☐ Unknown

Race* (select all that apply)

☐ American Indian/Alaska Native

☐ White

☐ Asian

☐ Black/African American

☐ Native Hawaiian/Pacific Islander

☐ Other Race

☐ Unknown

Ethnicity*

☐ Hispanic or Latino

☐ Not Hispanic or Latino

☐ Unknown

2nd Form:

Insurance Information

Does patient carry insurance? ☐ Yes ☒ No

Primary Insurance holder:

Insurer:

Group/Individual ID number:

Administration Information:

The Administration Name*:

Zip Code*:

Provider Information:

Provider ID*:

3rd Form:

Vaccination Information

Vaccine Type:

CVX code:

Manufacturer: N/A

Lot Number*:

Expiration Date:

Dose Number*: ☒ 1st Dose ☐ 2nd Dose ☐ 3rd Dose

Administration Site: ☐ Left Arm ☐ Right Arm

(Lower Extremity) ☐ Left ☐ Right

Date Administered:

Number of Dose Wasted:

Administration Route:

Missed Appointment*: ☐ Yes ☒ No

Comorbidity*: ☐ Yes ☒ No

Refused Vaccination*: ☒ NO

☐ YES Reason:

Vaccinator:

Received EUA Fact Sheet for Recipients*: ☐ Yes ☒ No

Report an Adverse Event to VAERS: [Click here to report](#)

Submit
Reset

Adding Updating Patient :

Adding A New Record for Existing Patients

[Profile](#) [New Patient](#) **[Existing Patient](#)** [Retrieve Data](#) [Log Out](#)

Last Name*:

First Name*:

Date of Birth*:

Zip Code*:

1st Form:

Patient Information
First Name*:

Middle Name:

Last Name*:

Date of Birth*:

Street 1*:

Street 2:

City*:

State*:

Zip Code*:

Gender*:

Race*:

Birthdate*:

2nd Form:

Insurance Information
Primary Insurance Number:

Insurer:

Group/Individual ID Number:

Provider Information:
The Administration Name*:

Street 1*:

Street 2:

City*:

State*:

Zip Code*:

Provider Information:
Provider ID#:

First Name*:

Middle Name:

Last Name*:

Street 1*:

Street 2:

City*:

State*:

Zip Code*:

3rd Form:

Vaccination Information
Vaccine Type:

CVX Code:

Manufacturer:
Lot Number*:

Expiration Date:

Dose Number*: ☐ 1st Dose ☐ 2nd Dose ☐ 3rd
Dose:
Administration Site: ☐ Left Arm ☐ Right Arm
(Lower Extremity) ☐ Left ☐ Right
Date Administered:

Number of Dose Wished:

Administration Route:

Wished Appointment*: ☐ Yes ☐ No
Comorbidity*: ☐ Yes ☐ No
Refused Vaccination*:
☒ NO
☐ YES
Vaccination:

Received EMA Fact Sheet for Recipients*: ☐ Yes ☐ No
Report on Adverse Event to VAERS: [Click here to report](#)

Adding Retrieving Patient :

Retrieving Records

Profile New Patient Existing Patient **Retrieve Data** Log Out

Last Name*:

First Name*:

Date of Birth*:

Zip Code*:

Dose:

Lot Number*:

1st Form:

Patient Information

First Name*:

Middle Name*:

Last Name*:

Date of Birth*:

Street 1*:

Street 2*:

City*:

State*:

Zip Code*:

Gender*: ☐ Male ☐ Female ☐ Unknown

Race*:

Ethnicity*

- ☐ Hispanic or Latino
- ☐ Not Hispanic or Latino
- ☒ Unknown

2nd Form:

Insurance Information

Primary Insurance Holder*:

Insurance*:

Group/Individual ID number*:

Administration Information:

The Administration Name*:

Street 1*:

Street 2*:

City*:

State*:

Zip Code*:

Provider Information:

Provider ID*:

First Name*:

Middle Name*:

Last Name*:

Street 1*:

Street 2*:

City*:

State*:

Zip Code*:

3rd Form:

Vaccination Information

Vaccine Type*:

CVX code*:

Manufacturer*:

Lot Number*:

Registration Date*:

Dose Number*: ☐ 1st Dose ☐ 2nd Dose ☐ 3rd Dose

Administration Site: ☐ Left Arm ☐ Right Arm

(Lower Extremity) ☐ Left ☐ Right

Date Administered*:

Number of Dose Vials*:

Administration Method:

Missed Appointment? ☐ Yes ☐ No

Consentability? ☐ Yes ☐ No

Refused Vaccination?

- ☐ NO
- ☐ YES Reason:

Vaccinator:

Received EUA Post Street for Recipients? ☐ Yes ☐ No

Report on Address Went to ADDRESS: [CLICK HERE TO ADD](#)

Tools and Frameworks

- NetBeans IDE: Used for backend development and database integration.
- MySQL Workbench: For managing and testing database connectivity.
- Browser-based Testing: Validation of the UI using tools like Chrome or Firefox Developer Tools.

Key Outcomes

- The integrated application successfully connects the database with the front end to perform intended operations.
- Providers can manage patient data, vaccination history, and insurance details seamlessly.
- The interface is scalable and adaptable for future enhancements, such as adding vaccination reminders or dashboards for analytics.

Teamwork & Collaboration (5 Points)

Team Contribution

- Overview: The project was completed through collaborative efforts, with each team member taking on specific responsibilities to ensure the successful delivery of the Vaccination Management System.

Individual Contributions:

- Harshil Sharma:
Designed the **UI/UX** to ensure an intuitive and responsive interface for seamless user interaction. Worked on the **Main Page**, integrating navigation with database connectivity to handle patient and vaccination records effectively.
- Harsh Patel:
Contributed to the **UI/UX design**, with a focus on creating clean and user-friendly forms. Worked on the **Existing Patient Page**, developing features to retrieve and update records in the database, ensuring accurate data consistency.
- Ngoc Yen Nhi Tran:
Focused on the **logical backend functionality**, ensuring smooth communication between forms and the database. Designed and implemented the **New Patient Page**, ensuring proper data validation and mapping patient, insurance, and vaccination details to the database.
- Syeda Bushra Fatima:
Developed the **logical backend** for database querying and retrieval operations. Worked on the **Retrieve Data Page**, ensuring efficient database searches for vaccination records and implementing robust data-handling logic.

Collaboration Process

Communication Tools:

- Team meetings were held weekly to discuss progress and resolve challenges.
- Tools like Slack, Zoom, and Google Drive were used for collaboration and resource sharing.

Task Allocation:

- Tasks were divided based on expertise, ensuring efficient project management.
- Progress tracking was conducted using a shared Google Sheet.

Peer Evaluation

- Feedback: Team members provided constructive feedback on individual contributions during peer evaluation.

Evaluation Forms:

- Each team member completed a peer evaluation form detailing their contributions and assessing the performance of others.

Challenges and Resolutions

- Challenge: Synchronizing database updates with the front-end interface during integration.
- Resolution: Conducted additional testing sessions and debugged the code collaboratively.
- Challenge: Ensuring all members were proficient in the required technologies.
- Resolution: Knowledge-sharing sessions and resource sharing to bridge skill gaps.

Key Outcomes

- Effective collaboration ensured the timely completion of project milestones.
- Each team member contributed meaningfully, and peer evaluations confirmed the equitable distribution of work.