



Project Technical Report

Well-Being Monitor System

by Group 2

INSTRUCTOR: Professor Maherukh Akhtar

Member: Aemun Ahmar

NYIT ID: 1047508

Phone: 646-886-0346

Email: aahmar@nyit.edu

Member: Fernanda Tovar

NYIT ID: 1090913

Phone: 914-563-2211

Email: ftovar@nyit.edu

Leader: Neha Bala

NYIT ID: 1133176

Phone: 347-761-4437

Email: nbala02@nyit.edu

Member: Jeanne-Venus Aine

NYIT ID: 1051152

Phone: 929-327-8454

Email: jaine@nyit.edu

Member: Tiara Nurse

NYIT ID: 1114555

Phone: 929-271-8051

Email: tnurse@nyit.edu

Table of Contents

0.0 Abstract	2
1.0 Introduction	2
1.1 Purpose	2
1.2 Existing Applications	2
1.3 Proposed Idea	3
1.4 Software Engineering Model	3
1.5 Objective / Scope	4
1.6 User Access Level	4
1.7 Technologies	5
2.0 Feasibility Report	6
2.1 Activity List of Operations	6
Activity List of Operations for Patient:	6
Activity List of Operations for Doctor:	6
Activity List of Operations for Pharmacist:	7
3.0 Analysis of the Project	8
3.1 Context Diagram	8
3.2 Diagram	8
4.0 Data Modeling	9
4.1 Table Schema	9
4.2 Entity Relationship Diagram	10
5.0 Explanation of Files	11
5.1 Files	11
5.1.1 Request / Grant Access	11
5.1.2 Writing Prescriptions	13
5.1.3 Viewing Records	14
5.1.4 Approve & Decline Prescription	16
5.1.5 Review Prescriptions	17
5.1.6 Allergy Alert	19
6.0 Testing	20
6.1 Unit Testing	20
7.0 Conclusion	21
7.1 Limitations	21
7.2 Contributions	22
7.3 Learning Outcomes	26
8.0 References	28

0.0 Abstract

The main goal of the project is to develop a website application that will solve a defined problem in the medical field. The intended outcome is to allow patients, doctors, and pharmacists to have easier access to health records and be able to view any updates on the patients' files.

This website application keeps track of health records while providing cross-examination to ensure the accuracy of the updated patient records. A back-end system was set up to allow patients to register accounts and allow doctors, pharmacists to sign into their accounts.

Depending on the user identification, they have different dashboards that contain buttons to manage/prescribe medication, add/edit new patients, manage/view patient records and view health reports. The unique system functionality includes a feature that generates a drug allergy alert when a Doctor prescribes a medication and the ability for the Pharmacist to approve and decline a prescription.

1.0 Introduction

1.1 Purpose

The purpose is to further the development of the proposed software to enhance the medical system by giving patients, doctors, and pharmacists a common platform allowing for better communication and health system.

1.2 Existing Applications

In terms of existing systems, through research we have found a few different implementations of this idea. For the most part, all of the existing systems that we found could be sorted into two categories: personal health records (PHR) or health apps.

A PHR is an electronic application that can help you maintain and manage your health information in a private and confidential environment. They are similar to electronic health records that doctors keep except with PHRs we get to manage our information and control who has access to it. Some existing PHRs include MyChart and MyHealthRecord. Both of these applications allow users to manage their health information while also giving you the ability to choose who gets access to this information. Both of these websites also allow healthcare providers involved in your care to access information such as allergies, medications, medical conditions, and pathology results. PHRs give you control of your own health information meaning healthcare providers that can access to this information

cannot change it. The main purpose of PHRs is to help patients better communicate with their healthcare providers to better manage their care.

Health apps are different from PHRs in that they offer additional features such as fitness tracking and reminders for medications, tests, and appointments. In addition, they track other health information such as nutrition, sleep, and so on. Some health apps include: MyFitnessPal, FitBit and so on. Depending on the type of app that you choose, certain apps may have different features according to the purpose of the app.

1.3 Proposed Idea

For our project, we went with the PHR approach for several reasons. The main reason we chose to go with this approach is because the purpose of our project is to improve communication between patients, doctors, and pharmacies by providing them with a common platform to access patient records. In addition, health apps are mostly intended for personal use and not as a way to communicate with your doctor.

Our idea is very similar to what exists currently, but there were some differences. For example, patients will not be able to edit their information. Instead, doctors will be the ones editing all of the patient information while patients will be able to view it and make sure all of the information is correct. In addition, doctors are able to send electronic prescriptions directly to the hospital pharmacy they are associated with instead of using a third party application to do so. Patients are able to see the medications that they are prescribed to by their Doctor.

The main difference is that we wanted to focus on a particular feature that we thought is very important because it can be overlooked and not prioritized in the medical field. We wanted a common bridge for all three users when it comes to a life threatening matter. That feature is an allergy alert. The procedure starts with the doctor prescribing medication. When they try to submit the prescription, an alert will appear if the allergy input is the same as the medication input. Therefore, the Doctor has to re-submit. After the prescription goes through, the pharmacist then decides whether to approve or decline based on insurance coverage and any other reason they deemed necessary. After, the doctor receives a notification to resend a prescription. Then once approved, the patient has that medicine displayed on their health record and is ready for pick-up.

1.4 Software Engineering Model

The approach taken to complete the project in a timely manner is through the incremental or iterative model approach. The model is integrated around all the phases which allow for a thorough or detailed development of the system. The approach was used to develop the website structure and layout to determine where all the functions of the website will be added. The next part broke down the database into different tables for the patient records, doctors, pharmacists, and medicines.

1.5 Objective / Scope

The requirements of the website application had to include different accounts for each user whether it was a Doctor, Patient or Pharmacist. The main objective was to provide a uniform record and prescription form that is simultaneously updating on all ends. It will contain four entities and access levels: Patient, Doctor, Pharmacy, and an Admin. The Doctor and Patient must register and the Doctor can update the medical records. The important feature had to show a situation, such as an allergy information that would trigger the Doctor if a noted allergy in a medication was prescribed.

All the requirements were completely done. Unfortunately, with the limitations of time constraints we couldn't have an admin. If this project would continue, we would have added another user named Admin that would have access to all features. The admin would have its own login information and would have a rich security layer, since it has access to a lot of vital information. Our temporary admin consisted of using JSON files that held the data about the Doctor and Pharmacist accounts. We executed the JSON-to Table Function in the URL and it automatically inputted the Doctor and Pharmacist account info into the Database.

1.6 User Access Level

Depending on the user, they will have a different login that gives them access to different features as listed below:

User Access Level			
Action	Patient	Doctor	Pharmacist
Request access to health records	Allowed	Denied	Denied
View health records	Allowed	Allowed	Denied
View Medicine Prescribed	Allowed	Allowed	Allowed

Grant Access for patient to view records	Denied	Allowed	Denied
edit and update all performed tests from the patient	Denied	Allowed	Denied
Confirm prescription	Denied	Denied	Allow

Figure 1.6- Table of User Access Level

This table shows that depending on your account information it gives you access to different features on the dashboard. Patients can only view while Doctors and Pharmacist has features that can update the patient's records.

1.7 Technologies

For this project, we created a website application that used a combination of multiple programming languages for different aspects of the website. In terms of software, we used Brackets as our main source code editor. For the front-end of our website, We started with a pre-made template and adjusted the HTML , CSS, Javascript to make it work for our design/framework needs. We added register and login templates and also table templates to display and receive information to our databases. HTML and CSS was used to create the framework for the website while Javascript was used to help the user interact with the website. For the back-end we used PHPMyAdmin and MYSQL. MySQL was used to create our database and the different tables that we needed for this project. PHPMyAdmin was used to help establish a connection between the database and the website while also allowing the website to communicate with the database. In order to run the PHP code we used Xampp web server.

2.0 Feasibility Report

2.1 Activity List of Operations

Activity List of Operations for Patient:

1. Register as a patient with doctor code, first name, last name; email, new password. (The doctor's code has to pre-exist and has to match the patient with the respective Doctor.) (once the account is activated the patient will have a medical record number or MR number generated for them. This number will allow the doctor and pharmacist to view or update the patient's file.)
2. Once account is registered, you can log in with a valid email and password. If it's valid then it will take you to the dashboard where you can see buttons.
3. In one section there are two buttons.
 - a. If patient clicks on "grant access" then it will notify the Doctor that the Patient wants to see their records.
 - b. Once Doctor grants access, the other button allows the patient to "view the record", which they can't edit.
4. In the other section, the other button is the "view medication" button and once the prescription is approved the table will display the medication prescribed.
5. The patient can then log out.

Activity List of Operations for Doctor:

1. Register as a Doctor with name, hospital location, password and ID number.
2. Once account is registered, you can log in with an ID number and password. If it's valid then it will take you to the dashboard where you can see buttons.
3. In one section there are three buttons.
 - a. If the Doctor clicks on "view patients" then the Doctor can see what patients register with their Doctor's code.
 - b. If the Doctor clicks "add New Patient" then Doctor has to add the automatic generated patient id, first name, last name, dob, age, height, weight, allergies, current medication, address, phone number, and email.
 - c. If the Doctor clicks "Grant Access" then they can see the table filled with patients that requested access to see their records. Doctor can either approve or deny.
4. In the other section, there are two buttons.
 - a. If the Doctor clicks "Write prescription" then the Doctor must find the appropriate patient in the table and add the inputs: insurance, date, medicine, cause, strength, dosage, medication, allergies, and notes. Once this is done, you can prescribe. A pop up notification will appear if the medication causes them an allergy and the Doctor has to change it.
 - b. If the Doctor clicks "Declined Prescriptions" then the Doctor must resend the prescription to the pharmacist with the change of the medication.
5. The doctor can log out.

Activity List of Operations for Pharmacist:

1. Register as a Pharmacist with name, hospital location, password and ID number.
2. Once account is register, you can log in with an ID number and password. If it's valid then it will take you to the dashboard where you can see buttons.
3. In one section there is a button to "view prescriptions".
 - a. If the Pharmacist clicks on "view prescriptions" then the Pharmacist can see a table with the patient and and then a link that takes you to the prescription information of that particular patient. The Pharmacist can deny the prescription and the Doctor will be notified. If approved then it will appear on the patient's dashboard.
4. The pharmacist can log out.

3.0 Analysis of the Project

3.1 Context Diagram

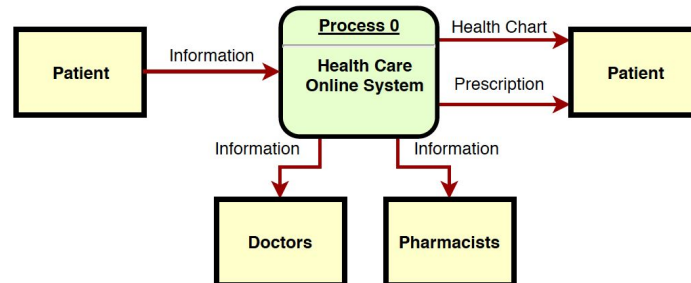


Figure 3.1 Context Diagram for Well-Being Monitor

3.2 Diagram

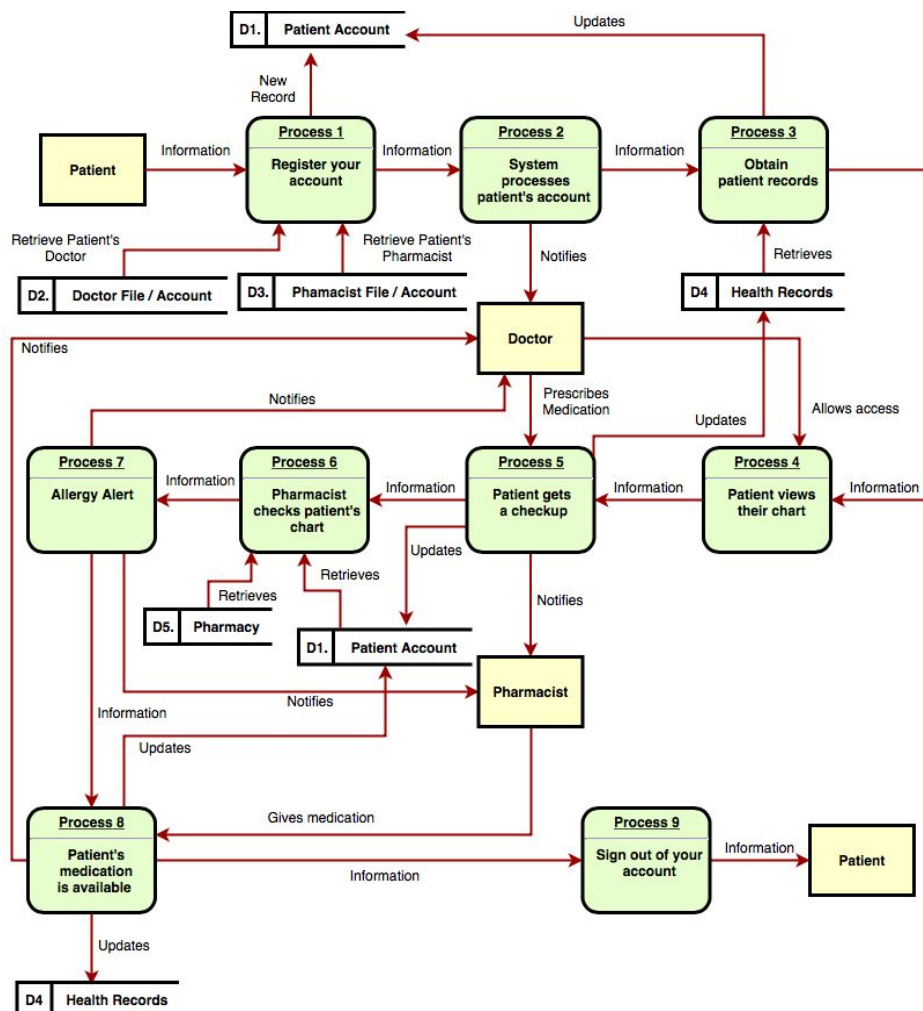


Figure 3.2 Data Flow Diagram for Well-Being Monitor

4.0 Data Modeling

This project was designed using MySQL for the purposes using and updating data between the users of this application. By using MySQL, respective tables were designed to hold information for each entity involved. In Figure 4.1, the tables show which of the primary keys would be used to find the required data to fulfill a request, i.e. knowing who one of out many doctors are by locating their doctor-id number. The table healthrecord claims three primary key to pinpoint and project the appropriate information related to the patient. In Figure 4.2, the entity-relationship diagram visually shows how the application works between the users and objects within the application.

4.1 Table Schema

healthrecord	patients	doctors	pharmacists	pharmacy
PK <u>patientid: int</u>	PK <u>patientid: int</u>	PK <u>doctorid: int</u>	PK <u>pharmacistidid: int</u>	PK <u>medid: int</u>
PK <u>doctorid: int</u>	fname: varchar(35) lname: varchar(50) password: varchar(20)	fname: varchar(35) lname: varchar(50) password: varchar(20)	fname: varchar(35) lname: varchar(50) password: varchar(20)	name: varchar(50) dosage: varchar(10) sideeffect: varchar(1000)
PK <u>pharmacistidid: int</u>				
doctorid: int pharmacistid: int lname: varchar(50) fname: varchar(35) birthday: date age: int gender: varchar(10) height: int weight: int allergies: varchar(1000) medication: varchar(2000) address: varchar(100) phone: varchar(10) email: varchar(100)				

Figure 4.1 Table Schema for Well-Being Monitor

4.2 Entity Relationship Diagram

Well-Being Monitor Application Entity-Relationship Diagram

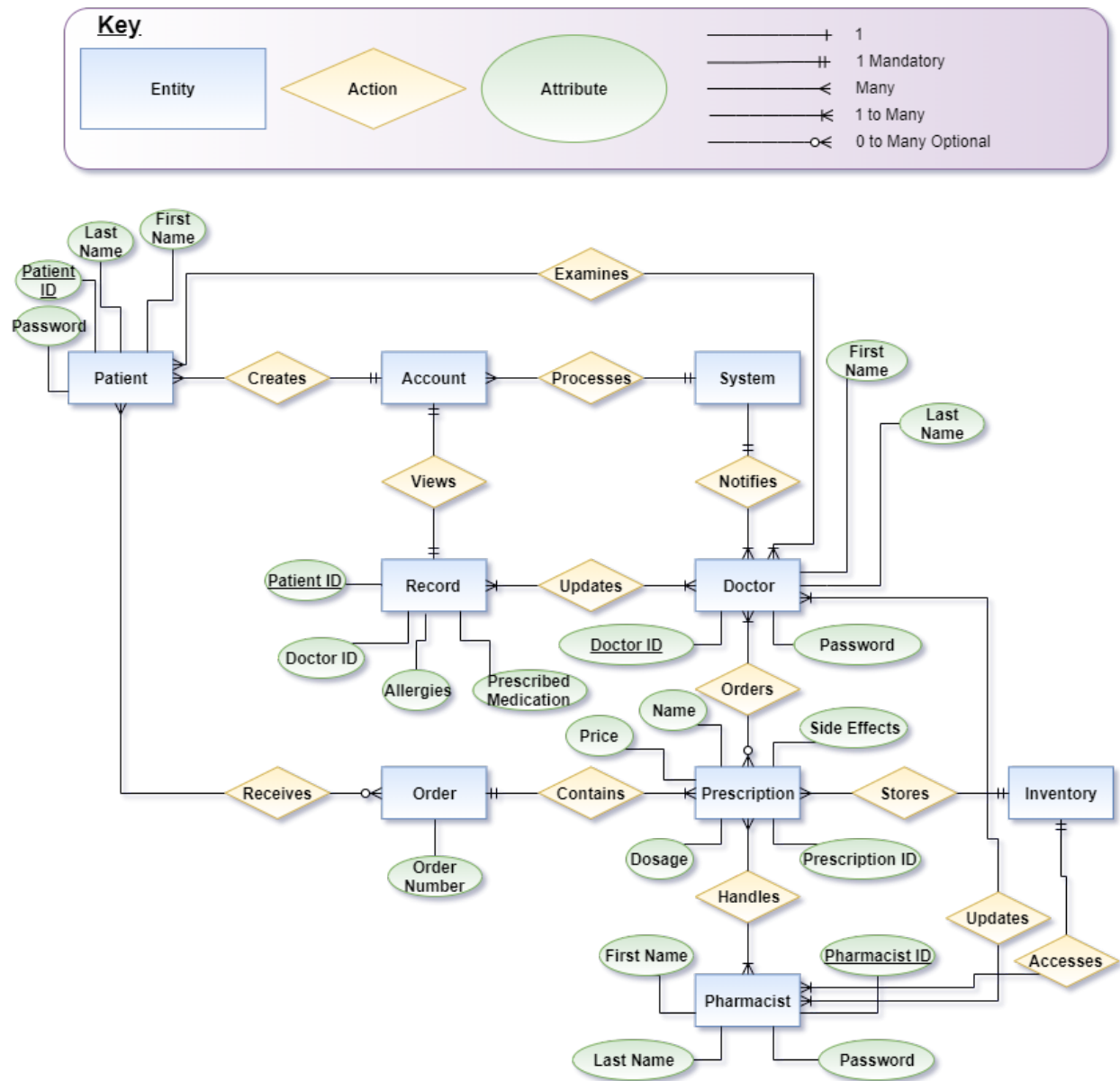


Figure 4.2 Entity Relationship Diagram for Well-Being Monitor

5.0 Explanation of Files

In this section we get a to see closer look into the files of the website. There are 6 different functions demonstrated. Each function was designed based on the users needs to ensure the website functions smoothly. The patient side needed access to their records as well as prescriptions. The doctor needed to modify health records and keep track of the allergies while writing prescriptions while the pharmacist needed to ensure the insurance would cover the medication being prescribed.

5.1 Files

5.1.1 Request / Grant Access

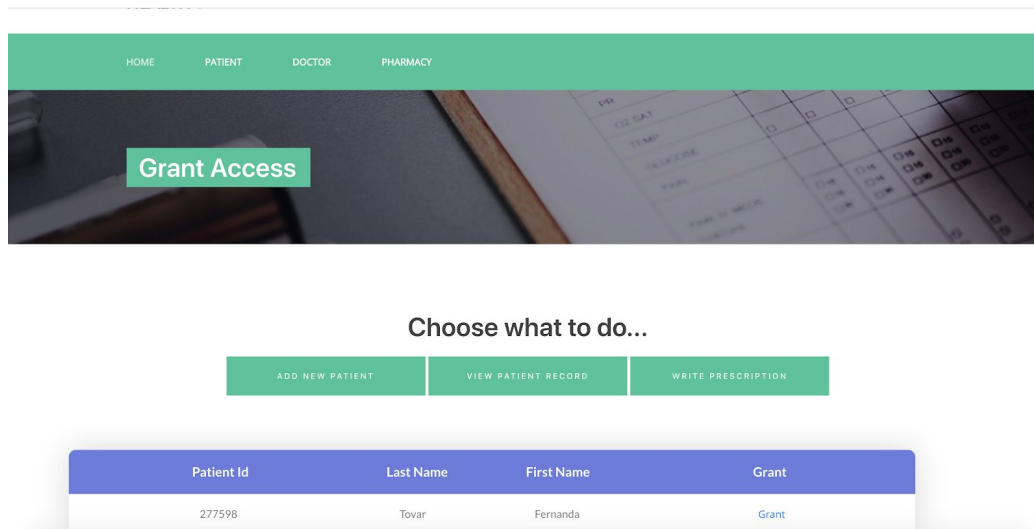


Figure 5.1 Grant Access function in doctor's account

```

1  <?php
2      session_start();
3
4      include('dbconnect.php');
5
6      $patientid = $_GET['PID'];
7      $sql = "UPDATE access SET granted=true WHERE patientid='$patientid'";
8
9      $output = $dbconnect->query($sql);
10
11     if ($dbconnect->query($sql) === TRUE)
12     {
13         //echo "Access Granted";
14         header('Location: grantaccess.php');
15     } else
16     {
17         //echo "Something went wrong" . "<br>" . $dbconnect->error;
18         header('Location: grantaccess.php');
19     }
20
21  ?>

```

Figure 5.2 PHP code for granting access to patient's request

Code Explanation

The code for granting access starts with retrieving the patient ID from the URL. As shown in figure 5.2, the retrieved patient ID is used to update the access table if the doctor has granted access to the patient. In the database, in the access table, there are two columns, one labeled request and one labeled granted. If the patient requests access the column will display a 1 while the granted column will display a 0, indicating that access has not been given. Once the doctor processes the patient's request, they grant access to the patient, updating the access table, column granted, from 0 to 1.

5.1.2 Writing Prescriptions

HEALTH+

Welcome Dr. Mike | Log Out

HOME

PATIENT

DOCTOR

PHARMACY

Prescribe

Choose what to do...

ADD NEW PATIENT

VIEW PATIENT RECORD

GRANT ACCESS

Patient Id	Doctor ID	First Name	Last Name	Insurance	Date	Medication
275913	945435	Tiara	Nurse	Insurance	Today's Date	Medication

Figure 5.3 Writing Prescription function in doctor's side

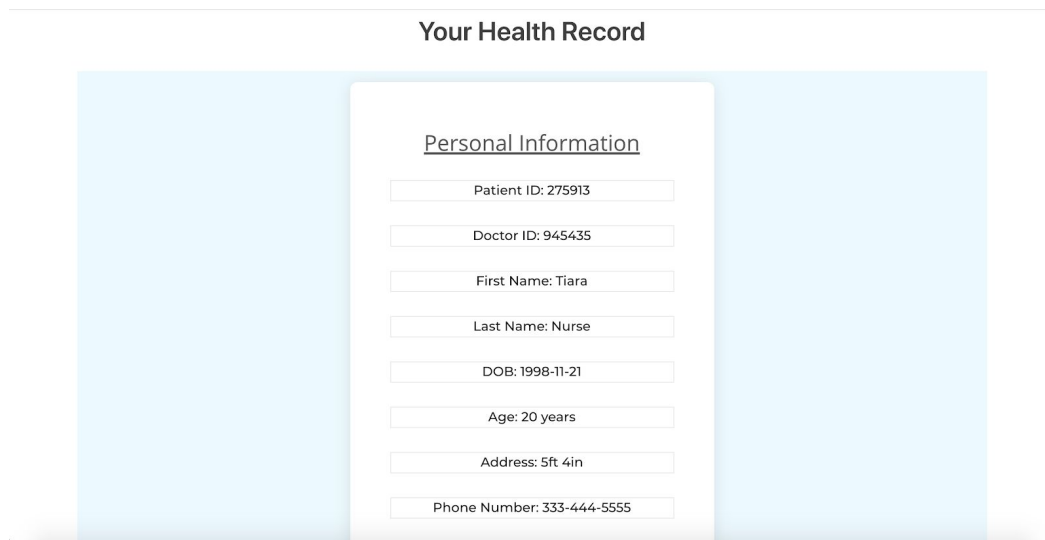
```
150 <?php
151     include('dbconnect.php');
152
153     $patientid = $_GET['PID'];
154
155     $sql = "SELECT patientid, doctorid, fname, lname, allergies, medication FROM healthrecord WHERE patientid =
156           '$patientid'";
157     $output = $dbconnect->query($sql);
158
159     while($result = mysqli_fetch_array($output)){ ?>
160     <form method=POST action=send_prescription.php onsubmit="return allergyAlert()">
161         <tr class="row100 body">
162             <td class="cell100 column1"><?php echo "<input style=text-align:center; type=text name=patientid
163               value='".$result['patientid']."'>"?></td>
164             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=doctorid
165               value='".$result['doctorid']."'>"?></td>
166             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=fname
167               value='".$result['fname']."'>"?></td>
168             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=lname
169               value='".$result['lname']."'>"?></td>
170             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=insurance
171               placeholder=Insurance required">"?></td>
172             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=date
173               placeholder=Today's Date required">"?></td>
174             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=medicine
175               placeholder=Medication Name required">"?></td>
176             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=cause
177               placeholder=Condition required">"?></td>
178             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=strength
179               placeholder=Strength required">"?></td>
180             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=dosage
181               placeholder=Dosage required">"?></td>
182             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=medication
183               value='".$result['medication']."'>"?></td>
184             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=allergies
185               value='".$result['allergies']."'>"?></td>
186             <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=notes
187               placeholder=Notes">"?></td>
188             <td class="cell100 column3"><?php echo "<input type=submit name=submit value=Prescribe">"?></td>
189         </tr>
190     </form>
191 <?php ?>
```

Figure 5.4 PHP code for writing prescriptions

Code Explanation

The code for writing prescriptions starts with retrieving the patient ID from the URL. The retrieved patient ID will be used to get the patient information from the health record table in the database. As shown in figure 5.4, the variable, \$result, is used to retrieve data from health record such as the patient ID, doctor ID, patient name, current medication and allergies. The rest of the form is used by the doctor fill out depending on the patient's condition.

5.1.3 Viewing Records



The screenshot displays a web form titled "Your Health Record". Inside the form, there is a section titled "Personal Information" with a list of fields containing patient data. The fields are as follows:

Field Label	Value
Patient ID:	275913
Doctor ID:	945435
First Name:	Tiara
Last Name:	Nurse
DOB:	1998-11-21
Age:	20 years
Address:	5ft 4in
Phone Number:	333-444-5555

Figure 5.5 patient viewing health record.


```

158 <?php if($output->num_rows != 0){
159     while($result = mysqli_fetch_assoc($output)) { ?>
160
161     <span class="login100-form-title p-b-33" style="text-decoration:underline">Personal Information</span>
162
163     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;"> Patient ID: <?php echo
164     $result['patientid']?></div>
165     <span class="login100-form-title p-b-33"> </span>
166     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;"> Doctor ID: <?php echo
167     $result['doctorid']?> </div>
168     <span class="login100-form-title p-b-33"> </span>
169     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">First Name: <?php echo
170     $result['fname']?> </div>
171     <span class="login100-form-title p-b-33"> </span>
172     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Last Name: <?php echo
173     $result['lname']?> </div>
174     <span class="login100-form-title p-b-33"> </span>
175     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">DOB: <?php echo
176     $result['birthday']?> </div>
177     <span class="login100-form-title p-b-33"> </span>
178     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Age: <?php echo
179     $result['age']?> years</div>
180     <span class="login100-form-title p-b-33"> </span>
181     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Address: <?php echo
182     $result['height']?></div>
183     <span class="login100-form-title p-b-33"> </span>
184     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Phone Number: <?php echo
185     $result['phone']?> </div>
186     <span class="login100-form-title p-b-33"> </span>
187     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">E-mail: <?php echo
188     $result['email']?></div>
189     <span class="login100-form-title p-b-33"> </span>
190     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Sex: <?php echo
191     $result['gender']?></div>
192
193     <br><span class="login100-form-title p-b-33"></span>
194     <span class="login100-form-title p-b-33" style="text-decoration:underline">Medication</span>
195     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;"><?php echo
196     $result['medication']?> </div>
197
198     <br><span class="login100-form-title p-b-33"> </span>
199     <span class="login100-form-title p-b-33" style="text-decoration:underline">Allergies</span>
200     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;"><?php echo
201     $result['allergies']?></div>
202
203     <br><span class="login100-form-title p-b-33"> </span>
204     <span class="login100-form-title p-b-33" style="text-decoration:underline">Vital Signs</span>
205     <div class="wrap-input100" style="text-align:center;font-family:Montserrat;font-size:18px;font-weight:500;">Your height in ft is : <?
206     php echo $result['height']?></div>

```

Figure 5.6 PHP code to view health record

Code explanation

For the functionality of this code to view the health record it was similar to writing prescriptions just the span class that was used, displayed the information differently.

5.1.4 Approve & Decline Prescription

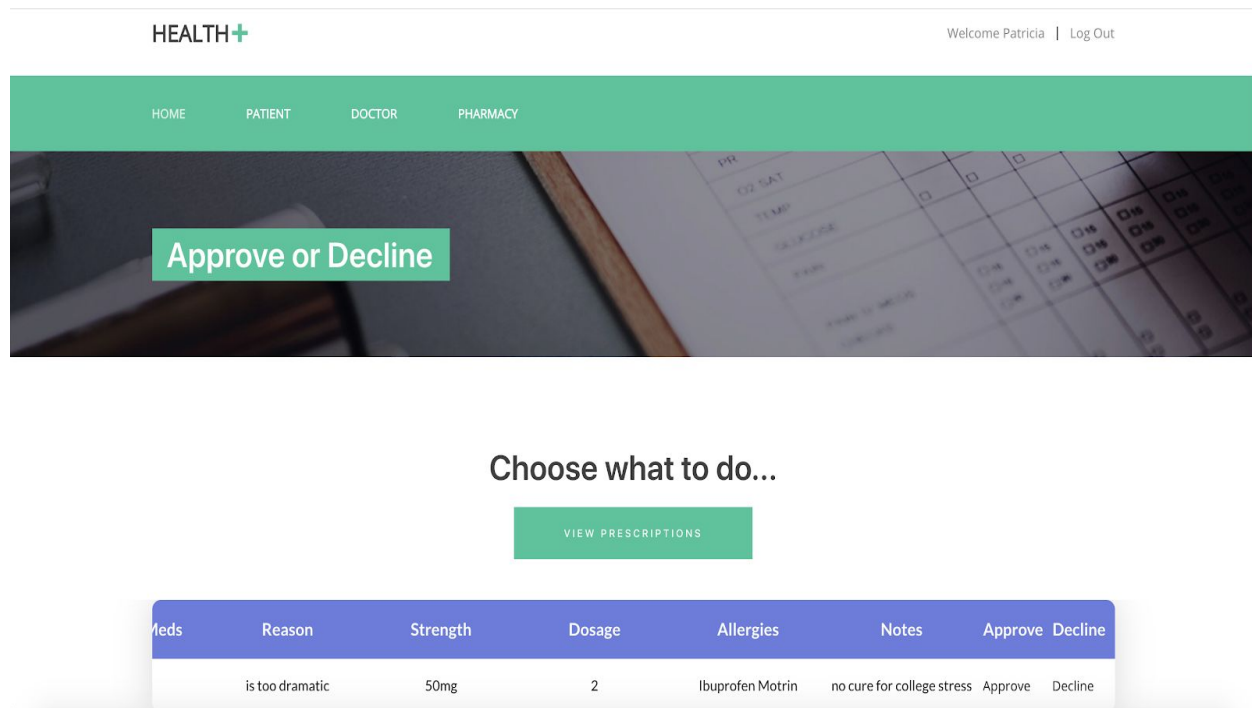


Figure 5.7 Approve & Decline functions for received prescriptions

```
24 //Retrieve patient ID from the URL
25 $patientid = $_GET['PID'];
26
27 //Action that needs to be performed in the MYSQL database
28 $sql = "INSERT INTO decline SELECT*FROM prescription WHERE patientid = $patientid";
29 $sql2 = "DELETE FROM prescription WHERE patientid = $patientid";
30
31 //Connect to the MYSQL database and check if the actions are successfully executed
32 if (($dbconnect->query($sql) === TRUE) && ($dbconnect->query($sql2) === TRUE)) {
33     echo "Prescription for patient: " . $patientid. " has been declined.";
34 } else {
35     echo "Error" . "<br>" . $dbconnect->error;
36 }
37 $dbconnect->close();
38 ?>
```

Figure 5.8 PHP code for declining prescriptions

Code Explanation

The code for approving and declining prescriptions is similar. The implementation begins with retrieving the patient ID from the URL. Using this information, the MYSQL database inserts the received prescription into either the decline table or the approve table, based on which function the pharmacist chooses to perform. In Figure 5.5, if the prescription is to be declined, the code allows the pharmacist to insert the received prescription into the decline table in the database. If the prescription was to be approved, the received prescription would be inserted into the approve table in the database. In addition to inserting into the decline or approve table, the received prescription is deleted from the prescription table where the doctor initially sent the form. This is because the prescription table only holds all prescriptions that are waiting to be approved or declined.

5.1.5 Review Prescriptions

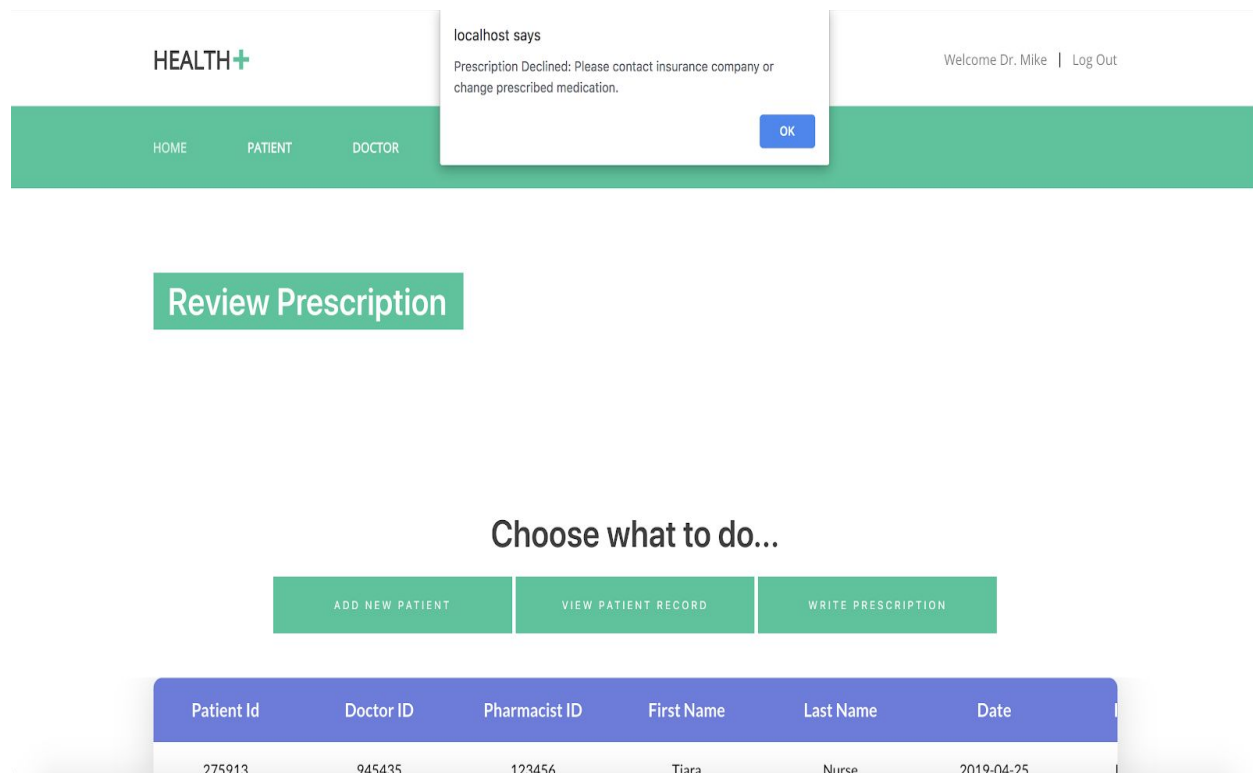


Figure 5.9 Review prescription page with an insurance alert

```

149 <!-- Prescription Decline Alert to Doctor-->
150 <script>
151     function declineAlert(){
152         alert ("Prescription Declined: Please contact insurance company or change prescribed medication.");
153     }
154 </script>
155 <tbody>
156     <?php
157         include('dbconnect.php');
158         //Retrieve data from the declined table to ensure that the doctor sees the declined patients
159         $patientid = $_GET['PID'];
160         $query = "SELECT * FROM decline WHERE patientid = '$patientid'";
161         $output = $dbconnect->query($query);
162         if($output->num_rows != 0){
163             while($result = mysqli_fetch_assoc($output)) {
164                 echo "<body onload=declineAlert()>";
165             }
166         }
167 <!-- Retrieve the form from the decline table and display all the information that is there for the declined patient-->
168 <form method=POST action=send_prescription.php onsubmit="return allergyAlert()" onload=declineAlert()>
169     <tr class="row100 body">
170         <td class="cell100 column1"><?php echo "<input style=text-align:center; type=text name=patientid
171         value='".$result['patientid']."'>"?></td>
172         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=doctorid
173         value='".$result['doctorid']."'>"?></td>
174         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=pharmacistid
175         value='".$result['pharmacistid']."'>"?></td>
176         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=fname
177         value='".$result['fname']."'>"?></td>
178         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=lname
179         value='".$result['lname']."'>"?></td>
180         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=date
181         value='".$result['date']."'>"?></td>
182         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=insurance
183         value='".$result['insurance']."'>"?></td>
184         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=medicine
185         value='".$result['medicine']."'>"?></td>
186         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=cause
187         value='".$result['cause']."'>"?></td>
188         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=strength
189         value='".$result['strength']."'>"?></td>
190         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=dosage
191         value='".$result['dosage']."'>"?></td>
192         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=allergies
193         value='".$result['allergies']."'>"?></td>
194         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=medication
195         value='".$result['medication']."'>"?></td>
196         <td class="cell100 column2"><?php echo "<input style=text-align:center; type=text name=notes
197         value='".$result['notes']."'>"?></td>
198         <td class="cell100 column3"><?php echo "<input type=submit name=submit value=Prescribe>"?></td>
199     </tr>

```

Figure 5.10 Reviewing prescriptions form and javascript code

Code Explanation

The code for reviewing prescriptions retrieves data from the declined table where all the prescriptions that the pharmacist rejected are located. This data is taken based on the patient ID and is then placed into the declined prescriptions button on the doctors end. In figure 5.7, the javascript code from lines 150-154, is the message that the local host displays when the doctor is reviewing the prescription. This alert states that the insurance was the cause of rejection. The variable, \$result, is used to fetch specific data from the database. The variable is used to display that selected data. This PHP code containing the variable is implemented onto the HTML form that is used for the review prescriptions.

5.1.6 Allergy Alert

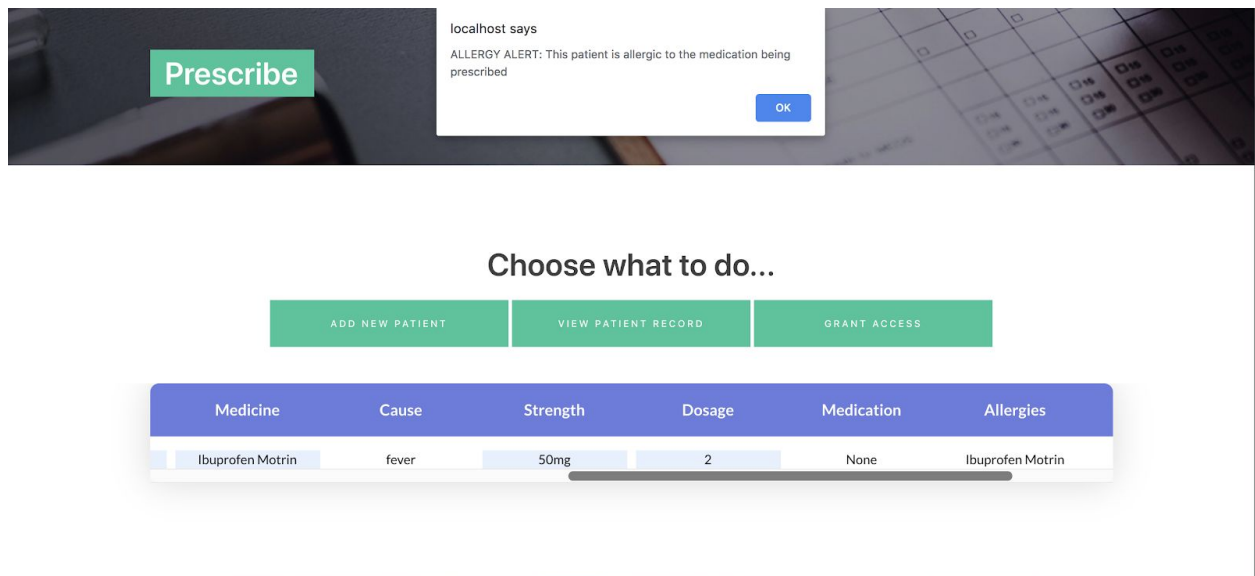


Figure 5.11 Allergy Alert function being triggered

```
227 <script>
228   function allergyAlert(){
229     if((document.forms[0].medicine.value) === (document.forms[0].allergies.value)){
230       alert("ALLERGY ALERT: This patient is allergic to the medication being prescribed");
231       return false;
232     } else {
233       alert("Prescription Sending");
234       return true;
235     }
236   }
237 </script>
```

Figure 5.12 Javascript code for allergy alert trigger

Code Explanation

Figure 5.12 displays the javascript code for the allergy alert. The code compares the input of the prescribing medication to the allergies indicated by the person. This allows the code to detect, if they are the same input, an allergy alert will be thrown. If the input is different then the prescription will be send.

6.0 Testing

By utilizing the sharing functionality Google Drive provides, a folder for testing the code and scripts was created to share amongst the team. First a main code testing file was created, which contained the basis of the website. Each team member then used the code testing file to develop and create several versions of code to be implemented into the program. Each member had different scripts dependant on the assigned tasks and was promptly updated on the functionality of the code when it is placed in the original code testing file. Once the functionality of a member's code yielded the right results, that code is added to the official code testing folder and the member would continue on the next task. A dummy database was also used to work with the team's data and code. As soon as the combined codes and scripts worked well together, that was immediately incorporated into the team's official health record database.

6.1 Unit Testing

```
21 <body>
22 <?php session_start(); ?>
23
24 <div class="super_container">
25 <!-- Home -->
26 <div class="home">
27 <div class="background_image" style="background-image:url(images/index_hero.jpg)"></div>
28
29 <!-- Header -->
30 <header class="header" id="header">
31 <div>
32 <div class="header_top">
33 <div class="container">
34 <div class="row">
35 <div class="col">
36 <div class="header_top_content d-flex flex-row align-items-center justify-content-start">
37 <div class="logo">
38 <a href="index.html">health</span></span></a>
39 </div>
40 <div class="header_top_extra d-flex flex-row align-items-center justify-content-start ml-auto">
41 <div class="header_top_nav">
42 <ul class="d-flex flex-row align-items-center justify-content-start">
43 <?php
44 if (isset($_SESSION['doctorid']))
45 {
46 echo ' <li><a href=doctoraccount.html style=text-decoration:none>Welcome Dr. ' . $_SESSION['fname']
47 . '</a></li>';
48 echo ' <li><a href=logout.php>Log Out</a></li>';
49 } else if(isset($_SESSION['email']) || isset($_SESSION['name']))
50 {
51 echo ' <li><a href=myaccount.html style=text-decoration:none>Welcome ' . $_SESSION['fname'] . '</a>
52 </li>';
53 echo ' <li><a href=logout.php>Log Out</a></li>';
54 } else if(isset($_SESSION['pharmacistid']))
55 {
56 echo ' <li><a href=pharmacyaccount.html style=text-decoration:none>Welcome ' . $_SESSION['fname'] .
57 . '</a></li>';
58 echo ' <li><a href=logout.php>Log Out</a></li>';
59 } else
60 {
61 echo ' <li><a href=register.html>Register</a></li>';
62 echo ' <li><a href=patient.html>Sign In </a></li>';
63 }
64 }
65 </ul>
66 </div>
67 </div>
68 </div>
69 </div>
```

Figure 6.1 HTML and PHP Code of the Website

Code Explanation

Figure 6.1 displays part of the main website code, which utilizes PHP code that uses data information from logging into either a patient, doctor, or pharmacist user account, which will provide a greeting of the user on the next updated page. If no user is logged in, the code will call for an action for registering a new account or signing into an existing account.

7.0 Conclusion

7.1 Limitations

Due to time constraints and how much more research and practice is needed to fully implement our remaining ideas, we faced a few limitations with our website application.

Our first limitation deals with notifications. What we all wanted to implement was a notification system that can send patients an email confirming when they have signed up for appointments and information about their prescriptions. But this was not included as more research needed to be done on how to code and include this aspect as well as this was not a multi device application.

The next limitation was for the patient allergy alert. For the alert to work correctly, the doctor must input each allergy in at a time for the system to check. Inputting multiple allergies separated by commas or spaces is not acceptable and read in.

Lastly, including an insurance transaction aspect was the last limitation. Our hopes was to include a feature that would calculate the amount of insurance coverage for different medication. But none of us had time to create another database for this option.

Even though we faced these limitations, we are hopeful to return to this project to fully include these aspects which go hand in hand with some future expectations. One future expectation would be the emailing and instant messaging feature for notifications. We would include a feature where patients can message doctors their questions, confirm appointments and send reminders, and a notice of medication pickup.

With appointments in mind, we would like to add a feature to schedule appointments based on patient request. The doctor would then approve it based on his/her schedule. But this would require another database to implement this feature.

And lastly, we would like to fix the website having no “real” admin. We are using PHPMyAdmin and JSON file currently. But, we would like to add a page to our website with an admin login to adjust everything from the patient to the doctor to the pharmacist.

7.2 Contributions

1. Leader: Neha Bala

➤ DOCTOR ACCOUNT

- **Adding New Patient:** Inserting the add new patient information into the health record table in the database.
- **View Patients:** Displaying the patients of the doctor from the patient table by matching the doctor ID. The patients are displayed only by their ID, last name and first name.
- **Edit Patient Records:** Allowing the doctors to edit the health record of patients. All the information of that patient is retrieved from the health record table using the patient ID. The ID's can't be edited.
- **Updating Patient Records:** Using the edited information to update the health record table in the database with the new information of the patient.
- **URL Information:** The URL will display the patient ID, patient last name and patient first name of the patient whose record is being edited.
- **Write Prescriptions:** The ID's of the doctor and patient, patient name, allergies and current medication is retrieved from the health record and onto the prescription. The doctor fills out rest of the information and sends the prescription to the prescription table in the database.
- **Allergy Alert:** There is an allergy alert thrown if the prescribing medication is the same as the allergies during the writing prescription.
- **Insurance Alert:** The insurance alert was thrown when the doctor would review the declined prescriptions.
- **Declined Prescriptions:** All prescriptions that the pharmacist declines goes into the decline table and that table information is displayed to the doctor to review.
- **Review Prescription:** Upon clicking the review prescription for the patient, the doctor immediately gets an alert indicating why the prescription was declined.

➤ PHARMACIST ACCOUNT

- **Decline Prescriptions:** If the pharmacist declines the prescription the information is removed from the prescription table and entered into the decline table for the doctor to review.
- **Approve Prescription:** If the pharmacist approves the prescription the information is removed from the prescription table and entered into the approve table where all final prescriptions are sent to the patient account.

2. Member: Aemun Ahmar

➤ PATIENT ACCOUNT

- **Patient Registration Form**
 - **Sign Up:** Inserting the registering information into the patient table in the database
 - **Doctor Code:** Checking whether the doctor code that the patient is registering with is a valid doctor code or not.
 - **Sign In After Register:** Signing directly into the patient's account once they are done registering.
- **Sign In:** Signing in with the email that the patient registered with.
- **Request Access:** Requesting access from doctor to view health record. Once requested, the patient does not have to request access again.

➤ DOCTOR ACCOUNT

- **Sign In:** Signing in with the Doctor ID that is assigned to them.
- **Grant Access:** Granting access to patients who want to view their health records. Once granted, the doctor does not have to do it again.

➤ OTHER PROGRAMMING CONTRIBUTIONS:

- **User Authentication:** Making sure that the user logging on is directed to the right account dashboard and corresponding functions. Also ensuring that the user remains logged in no matter the page they are on and that they are not able to sign into any other account or able to access the registration page.
- **Log Out:** Completely logging out of the account by destroying the session.
- **Table Schema:** Setting up all of the tables with the correct attributes and logic to ensure that all functions worked the way they should.

➤ TEAM CONTRIBUTIONS:

- **File Management:** Maintaining final code files and putting together everyone's code to ensure that they all worked together with the frontend, backend, and each other.

3. Member: Jeanne-Venus Aine

➤ RESEARCHED features:

- **Sign In generator:** I did some research on ways to keep a patient from accessing their health record. We had wanted there to be a 24 hour limit on the patients' accounts after signing up to ensure that the account was verified and matched the doctor's information.

- **Notification Generator:** I researched ways to generate a notification for the doctor and pharmacist to use for any updates using PHP and MySQL. This included looking into the trigger function in SQL to generate a pop up message if the doctor or pharmacist were to have pending alerts.

➤ **Front-End Pharmacist pages:**

- **Pharmacist sign in:** I worked on the creating the pharmacist sign in pages of the website. As it similar to the doctor sign in, all that was needed was a pharmacist ID and matching password to log in. Once logged in, the pharmacist should be able to view, approve, and/or deny any prescription updates received by the doctors on file. It would show the patient's information as well as the requested prescription and dosage and mainly the patient's allergy.
- **Connection to database:** The database was used to store the sign-in information of the doctors and pharmacists, which were created using JSON files. These files allowed for the usage of data stored in each "doctor" and "pharmacist" directly from the server to the website.

4. Member: Tiara Nurse

➤ **RESEARCHED feature:**

- **Assign random unique IDs:** I researched how to generate a 5 digit random ID number using PHP that will be assigned to the patients when they sign up. This ID number served as identification jut as a regular ID number does everywhere else.

➤ **Generating Unique Patient ID:**

- The ID number was generated at sign in and consisted of 5 numbers randomly chosen. The numbers range from 0 to 9 and the result was appended onto the PIN string. This was chosen because parsing this number as an integer or storing it in an INT database would lead to a loss of data. More specifically, if the ID had leading zeros, those leading zeros would be lost, throwing the ID number off.

➤ **Front End**

- I also had the task of writing all descriptions for our website. This meaning that any description text displayed that helps one navigate the website, state who we are, and what services are provided, was written by me. Research was done by viewing other hospital sites to view particular wording and explanations.

5. Member: Fernanda Tovar

➤ **Website Design:**

- **Templates :**

- Main layout: I focused on researching website templates for the main layout. For this I was looking for a medical layout that already implemented medical related photos. This helped us start with a foundation and focused on the database, which is what truly gave the website life.
- Login template: I researched a login template that had a clean and simple layout for the user to input their email or ID number and the password. This template also included a link that would direct them to register if they haven't yet. This login template also was used to display the patient's records by simply removing the button that "logged" the person in.
- Registration form: This also lead to the registration form that was manipulated to display what we required for the user to input.
- Table template: The last template that I found and was key to display the information in a clean way. This was a table that had slightly change in tone as the rows were altering. The table was used to display patients and prescriptions.
- **Adjusted Templates** : I had to now adjust these templates by editing the css and javascript files. Since multiple templates were combined, I had to make sure that each html file had the proper css and javascript calls in order to display the information correctly. I also adjusted the templates by moving the code around to create a layout that worked for our project requirements.
 - **Information Pages:** Specifically for this part, I created the tabs that directed the user to a page that corresponded to each user level. I moved around the code to created an ideal "advertising place" to display the features that they had access to. The login code was also added to each information page.
 - **Created Dashboards:** The dashboard pages layout was the same for all users expect that some had more sections with more buttons. The dashboard was an html page from the main layout template that contains the appropriate containers to still contain the tabs and the footer. Overall, it had the same layout as other pages with boxes.

➤ **View Records/View Prescriptions:**

- **PHP:** After having the desirable layout, it was up to the challenging part, which was to actually display the information that was inputted from all different user types. I learned how to call the proper php file to the the html file and then used "echo and .&result" to display the information.

7.3 Learning Outcomes

Neha Bala

Working on this project allowed me to learn more about the functionality of databases and how they interact with server side PHP. Through past projects and internships, I already had a deep understanding of the programming languages HTML, CSS and Javascript. Even though, I had previously encountered PHP and SQL, I never got to a chance to develop PHP code or create and interact with databases. Through this project, I got an opportunity to work closely with PHP and MYSQL database. I learned how to connect to the database using PHP and perform operations that allow me to edit the database depending on what a particular function required and the conditions they held. Using these languages, I learned how to implement many functions of a website that we encounter on daily basis, such as signing up or logging in or getting alerts. In addition to my prior knowledge, I got the opportunity to develop new skills that allowed me to go beyond the front-end of web development. Overall, being the team leader allowed me to work closely with all members in determining which functions to implement and which direction to take the overall project in. With many features overlapping each other, it was important to make sure everyone was being communicated all the details as the project went on.

Aemun Ahmar

Throughout this project, I learned more about UI design and the importance of it to enhance user experience. While I have worked with HTML and CSS before, I was not as involved in the process as I was for this project. Most of the time when doing projects like this I am always doing the backend portion because that is what I am mostly comfortable with. However, this project gave me a chance to explore a different aspect of web development and further develop the skills that I already had. I learned that the layout of your website is just as important as the backend functionality because a user friendly website is what most consumers look for when interacting with any sort of web application. Any site that is too complex or difficult to navigate will ultimately push users away. In addition, I also learned the importance of product management. Throughout the project, I found myself taking tasks that would normally be associated with the responsibilities of a product manager. Such tasks included building the project by coordinating with all of the other members in the group, designing the product, and so on. As a result, I learned that this is an area of tech that I am really interested in and would like to continue doing as a career path.

Jeanne-Venus Aine

Through this project, I learned a lot about the front and back-end of the application through PHP and HTML; I haven't learned PHP before this assignment, and, while I am by no means an expert, I learned how PHP ties in with developing websites through HTML and utilizing databases through MySQL. I also learned how to tie aspects of the databases to properly present information limited to specific users of the application. I hope to use this project as a starting point to branching out more into the field of databases and learning more of PHP and HTML and am grateful for my team's support and fortitude.

Tiara Nurse

Being unfamiliar with PHP and a beginner in using SQL, relying on my team played a huge role. Thanks to their patience and hours of research and testing, I can now say that I successfully understand the basics of PHP such as the syntax and coding small actions/commands. With SQL, learning how to use it in our Database class was helpful, but thanks to this project, I was successfully able to use what I learned and help produce a successful application. I realized how much more I have to learn and practice to better what projects I will create in the future, but one can only get better while knowing the basics.

Fernanda Tovar

This project proved itself to be quite challenging. Fortunately, thanks to my teammates' experience with database and my experience in website design, it became a perfect match. The website design aspect of the project was more for practicing my skills rather than learning it. Ultimately, this project still became a learning center for me. As a beginner with PHP and SQL, it was quite overwhelming to dwell into this, but with my team mate's guidance it became a bit more familiar and it gave me a chance to see how everything connects. The girls gave me quick explanations about how certain features would work and I would constantly ask questions. As well, the part that I am proud because I directly implemented it with guidance was the code that "echo" the results that was previously inputted. Also for SQL, I saw how the tables in the database were created and how to execute the statements. I definitely still need more practice, but I'm glad this project gave me the chance to get my feet wet.

8.0 References

SQL Tutorial, www.w3schools.com/sql/.

Aigars. "Fixed Header Table - Free HTML/CSS Table Template 2019." *Colorlib*, 21 Apr. 2018, colorlib.com/wp/template/fixed-header-table/.

Aigars. "Login Form 19 by Colorlib - Free HTML Login Form 2019." *Colorlib*, 21 Apr. 2018, colorlib.com/wp/template/login-form-v19/.

"HTML." *W3Schools Online Web Tutorials*, www.w3schools.com/.

Joefrey. "Health - Free Medical Clinic Website Template 2019." *Colorlib*, 11 Aug. 2018, colorlib.com/wp/template/health/.

Karthik, Girish. "66 Best Free Bootstrap Registration Forms For All Sites 2019." *Colorlib*, 15 Feb. 2019, colorlib.com/wp/free-bootstrap-registration-forms/.

"PHP Tutorials." *GeeksforGeeks*, www.geeksforgeeks.org/php/#basics.

"Rand." *Php*, www.php.net/manual/en/function.rand.php.