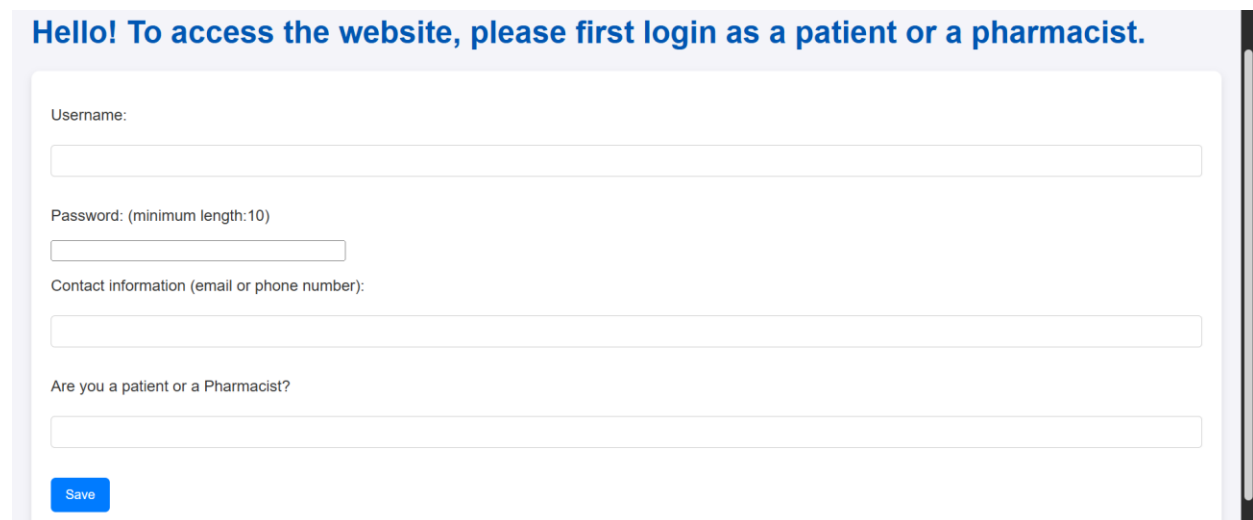


Gino Rojas

CIS 344

ADDITIONS:

For the final project, I first began by working on the login page, as I have created a form that requires the user four inputs in order to login and receive access to the home page the username, the password, the user's contact information, whether it's their phone number or email, and to login as a patient or a pharmacist. For the patient and pharmacist column, it provides role-based access control of the website, as depending on which type the user login as, it will lead them to one of two home pages, the home page for patients and the home page for pharmacist. I have also provided password security by making the minimum length of the password be 10 characters long, which if not reached, would display an error message "You have not entered all of the required details, please try again."



Hello! To access the website, please first login as a patient or a pharmacist.

Username:

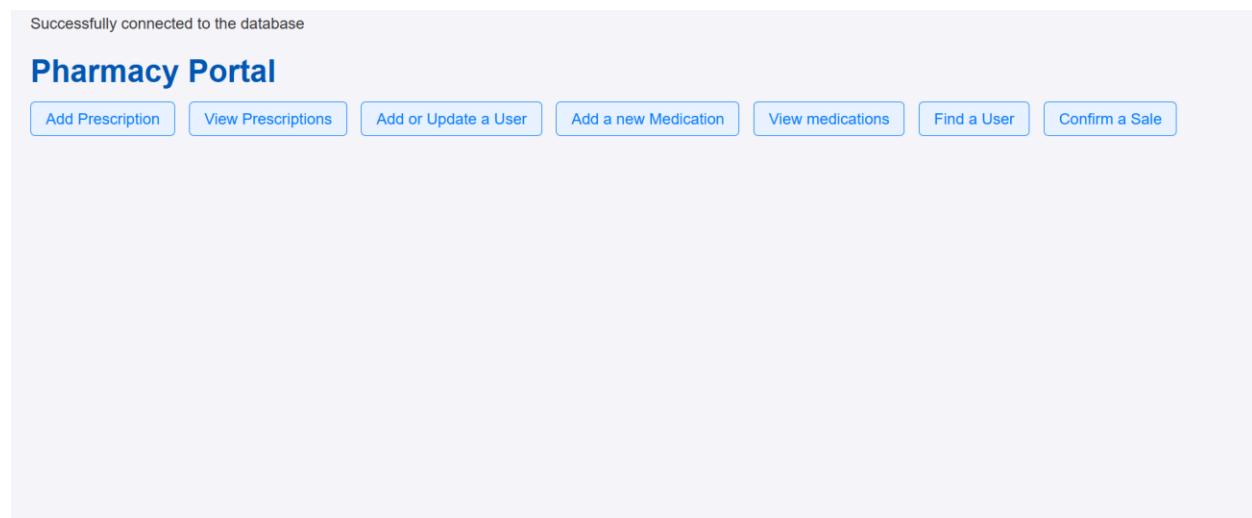
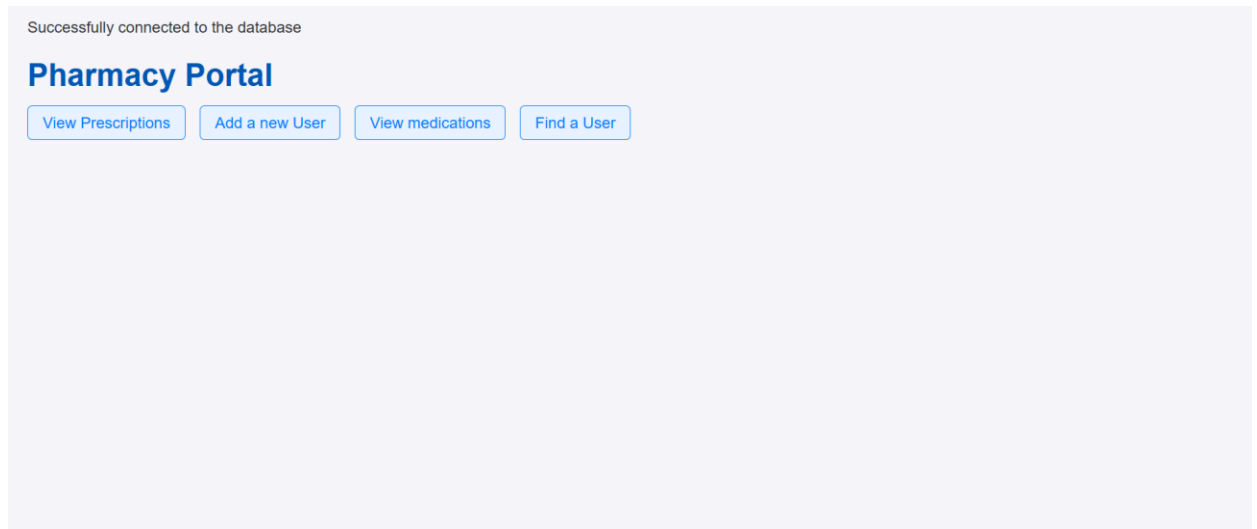
Password: (minimum length:10)

Contact information (email or phone number):

Are you a patient or a Pharmacist?

Save

After the user provides their information, they will be directed to the respective page depending whether they have logged in as a patient or pharmacist, where as the patient home page has less options then the pharmacist home page, as most of the options provided by the website are pharmacist exclusive. The login page would also then add the user's information into the user table within the database if the user didn't exist already, as well as make no changes if they did already exist.



After finishing the login page, I then started implimenting the user addition section of the project through creating a page called `addUser.php`, which is dedicated to adding or udating a user's information, which is displayed within the pharmacist page.

Please add a new user

To update an existing user, please use the same username to change information.

Username:

Contact information (email or phone number):

Are you a patient or a Pharmacist?

Save

[Back to Home](#)

Within the addUser page, I utilized a format similar to the login page's format but without the password column, as they both provide the requirements for the user table. Afterwards, I created the addUser function which calls the addOrUpdateUser stored procedure within the database, allowing the user to add the provided user information into the database after saving. I have also provided the option for the user to update any existing users within the database by utilizing the same username, which allows the user to edit the contact information and the user's type, as the addOrUpdateUser stored procedure allows this by detecting if there is a column with the same name within the database.

Add a new medication to be used for future prescriptions

Medication Name:

The recommended dosage:

The Manufacturer of the Medication:

Quantity Available:

Save

After finishing the addUser page, I then created another page called addMedication.php, which allows the pharmacist to add a new medication to the database, similar to the addUser page. This page provides the options to provide the medication's name, the recommended dosage of the medication the name of the manufacturer of the medication and the quantity that is available, by which after the user has provided the medication information through saving, it will utilize addMedication function, inserting the information into the database. Despite not being part of the medications table, I have provided the quantity available table in order to fill the inventory table, as both the medication table and inventory table required the medicationId, so I believed it was a perfect opportunity to seamlessly allow the user to provide the inventory of each medication, as well as provide the proper information for the next page.

Successfully connected to the database

All Medications

Medication Name	Dosage	Manufacturer	Quantity Available
Gabapentin	600 mg	Leonardo DiCaprio	300
Atorvastatin	80 mg	Johnny Depp	500
Tramadol	100mg	Bruce Willis	250

[Back to Home](#)

Immediately after finishing the addMedication page, I have provide a viewMedication page dedicated to having the user have the ability to see all of the medications available within the website. To display all of the medications, I utilized the viewPrescription starter code and modified it to utilize the medicationInventoryView through the medicationInventory function, which provided a query that displays the medication's name, the dosage, the manufacturer and the quantity available, which is within the inventory table as explained previously.

Successfully connected to the database

Please provide the ID of the user you wish to view.

userID:

Search

[Back to Home](#)

Successfully connected to the database

Here is the Selected User

User ID	Prescription ID	Username	Contact Information	userType	Dosage Instructions	Quantity
1	1	Maria Lopez	646-298-0592	patient	take orally twice a day	10

[Back to Home](#)

Next, I then started focusing on working on the user data retrieval section of the project, leading to me creating the `getUser` page. The `getUser` page involves providing the user the options to find a specific user of their choice via providing the user's Id, which after doing so will lead them to the `selectedUser` page, which will show the chosen user's information, alongside the prescription they are associated with. This page works through utilizing the `getUserDetails` functions that provides that user and prescription join table to receive the information, as well as the `getUser` function to display the table for the `selectedUser` page, as well as display the `getUser` page if the `userId` was not provided yet, similar to the view pages.

Here is the updated Sale list

Sale ID	Prescription ID	Sale Date	Quantity Sold	Sale Amount
1	1	2025-03-25 12:09:35	5	25.00
2	2	2025-04-16 04:15:20	10	50.00
3	3	2025-05-05 01:12:45	5	25.00
4	1	2025-05-05 23:32:50	200	1000.00
5	1	2025-05-05 23:44:54	200	1000.00

[Back to Home](#)

Successfully connected to the database

Please confirm the sale of the most recent prescription.

Prescription ID:

The quantity amount sold:

Save

[Back to Home](#)

Lastly, I have added one more page called the ProcessSale page, which involves the user to confirm the sale of a recent prescription, which after providing the prescription ID and the quantity amount sold through saving, leads the user to the updated list of the provided sales. How this page works is through the utilization of the processSale stored procedure, which inserts the provided information into the sales tables, as well as calculates the sale amount depending on the quantity sold. This stored procedure is then used via the processSale function, alongside the saleList function, which displays the saleList page, which shows all of the respective sales.

CHANGES:

Alongside adding new features to the website, I have also made some additional changes to existing pages, as I believed that they were missing some features in order to make the website feel more complete. For starters, I have changed the addPrescription page to include a refill count column, as this is due to the refill count being part of the prescriptions table, allowing the user the ability to provide the refill count. I have also changed the viewPrescriptions page to display the refill count as well to make sure and show that the refill count information was added. I have also added a trigger within the sql called AfterPrescriptionInsert, by which after the user creates a prescription, it would then update the inventory of the chosen medication through subtracting the quantity available with the quantity provided in the prescription. Alongside the trigger, I created a notification that would be displayed within the view medications page that alerts the user if a medications inventory is low or depleted. In addition, I wanted the website to begin with the login before providing access to the home page and its features. To achieve this, I began modifying the handle request function to have its default actions to be 'login,' which utilizes a login function that I have created to first display the login page before moving on to the home page. However, in order to make sure the website does not go directly back to the login page accidentally, I have made sure that every 'back to home' option in every page has the href="?"action=home" code so it would lead to the home page every time. Lastly, in order for the patient user to stay within the patient homepage and the same for the pharmacist user, I have created a session within the pharmacy Server page, which allowed me to create a \$_session called login type, which takes the information from the user type that the user provided within the login page and allows me to create 'back to home' leads that leads back to the correct home page, as if the user was a patient, they will be directed to the patient Home page via the handle Request function, alongside directing the pharmacist into the regular home page.

CONCLUSION:

Throughout this project, at first, I was a little afraid of working on a project this large due to my challenging time with time management and still being new to coding in php. However, as I kept working on the project, it has allowed me to properly learn the different functionalities that PHP and SQL

provides, and it has given me enough practice to know what needs to be done in each section. However, despite my skills in coding still needing some improvements and not being able to perfect this project due to time restrictions, I am still proud of what I was able to accomplish despite said time restrictions, as I believe this is a sign that I am improving and there is still more things to learn in the future.