Aaron Campuzano

CIS 344

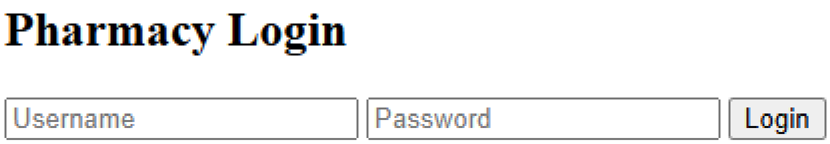
5/5/2025

The Pharmacy database

Well, the whole point of this program was to make a secure way for patients and pharmacist. I began my work on wednesday of last week but really got into the work on friday. When I woke up today, Xampp decided to get corrupted and there went my files, I managed to recover everything else aside from the two aforementioned pages and their data as well. With that explained, My code will deal with the adding of users, the viewing of inventory, the viewing of prescriptions, and the adding of such things. I managed to get all the pages I need to work with some hiccups here and there. The website I designed will allow a neat splitting of the information needed for each user. One user type, the Pharmacist will be able to access the Pharmacist portal which allows the addition of users, addition of medications, additions of prescriptions, the viewing of inventory, and the viewing of all prescriptions. The other user type doesn't have that privilege since they only need to know what their prescriptions are since they are just patients who need to know what they came for.

Let us go to the very base of it all, the database. I recreated it using Mysql. The tables I made are for the users, the medications, the inventory of said medications, the prescriptions for said mediations, and the sales of the medications. The primary key of the User table is the userid so it can be used as shorthand for when the pharmacist needs to quickly enter a prescription for a user and to need to type less. Its unique key is the Username since it would make life much simpler since it forces the user to create a new username so patients cant be confused with another and get their prescriptions mixed up. The userType is what separates the patient user versus the pharmacist and makes sure each of these users get only the information they need out from this website. The table Medictions is basically where the prescriptions, inventory, and sales all pull from since its where the different medications that the pharmacy may stock. It only has a primary key of medicationId which is used everywhere else as a foreign key to pull its id so people can quickly input a medication for a prescription or to check inventory. The others follow suit. The two processes that I made were the AddorUpdateUser and ProcessSale, the first deals with adding a new user or updating said users information within the database. It takes 4 different inputs which would come from the php side of the website. It checks if the param_userid exists for the user, then it would update the old users information with the new information given, if not it would insert a new user. In the former event, the user name, type, and contact info would be changed to new information that was input, in the latters case it will make a new user entirely. It would end its work by saying either “User updated successfully” or “User added successfully”. ProcessSale basically automatically does the work of both an inventory manager and a cashier. If the medicine is in stock it updates the inventory, records the sale and updates the refill count, if there isn't any in stock, it just says “out of stock. It takes 2 Input, the prescription id and the quantity sold, it then checks the inventory and looks at what's in stock then if there is enough in stock, the sale goes through, if not the sale doesnt go through. If the sale goes through, the inventory reduces the stock by subtracting the amount requested from stock, records the sale, then updates refill count +1. I did a trigger for it as well, AfterPrescriptionInsert. This is where every new prescription gets watched then immediately educes the stock. After INSERT is made to the prescription

Now lets talk about the php side, The PHP is split for the two users, that of the pharmacist and the patient, they need two different portals because a patient only needs to know their prescription. The first thing that the user needs to do is to log in. The page to the user would look like this.



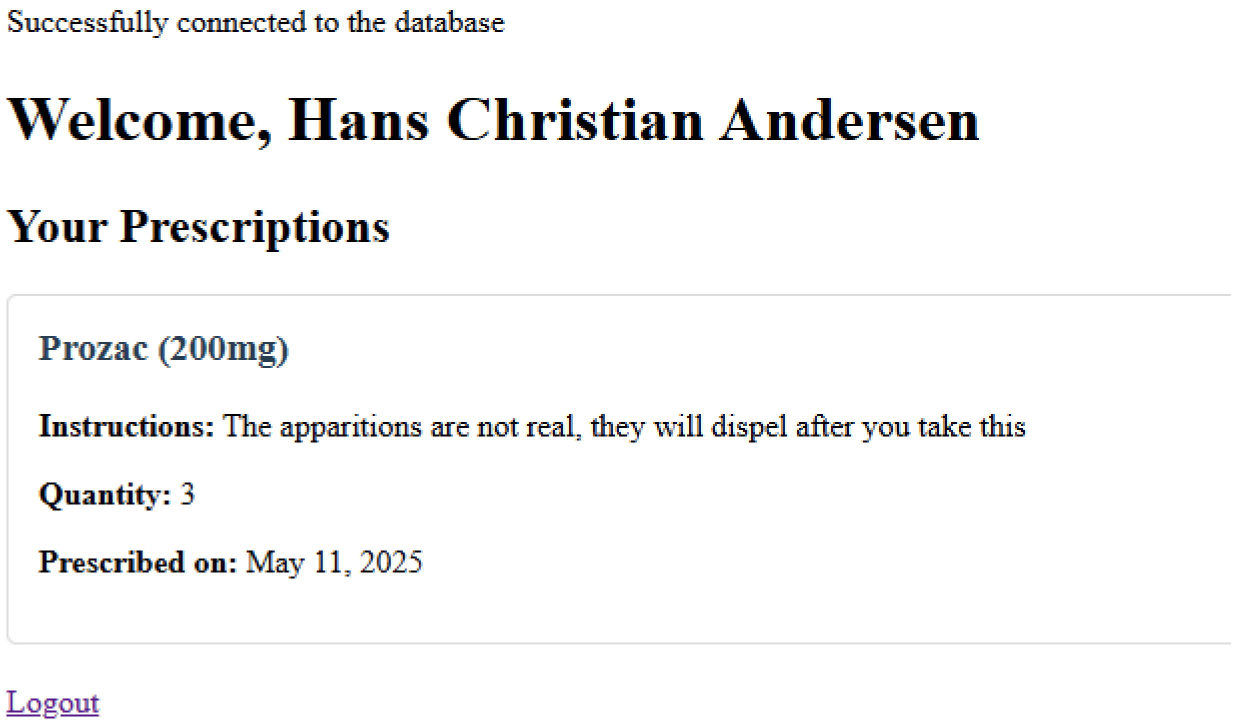
Simple and to the point. The backbone of this page starts at the database.php file of this. It uses the public function of verifyUser. The first action that is made is the creation of function that is called as such that takes two inputs, username and password. The next action taken is the creation of the the SQL query to obtain the user details that are the userid, the name, the type, and the password it is ended with a ? so an SQL injection cannot be made. The next line then binds the input given from the form, specifically $username and “s” which means the inputs a string. The next two line that comes after that runs the query and they get the results from the resulting query. What comes next is checking if the user exists in the user table, it is done by seeing if the resulting query had exactly 1 row which means it succeed if it is found then the next action would take the result would be turned into a associative array. Thats the verification of the Username done, the next action would be the password verification. This is done with an if statement that looks at the inputted password and compares it to the hashed password in the database. This ends off with 2 possibilities, either the username and password exist and the user is sent to either the pharmacy server or the patient portal, or the user does not exist/ the password is incorrect and or doesn't exist and the user is kept from accessing the rest of the server.

Aaron Campuzano

CIS 344

5/5/2025

The next part of the website then splits into two paths for the two users, the first we shall talk about is the patient portal. This is how it will appear to the user



Successfully connected to the database

Welcome, Hans Christian Andersen

Your Prescriptions

Prozac (200mg)

Instructions: The apparitions are not real, they will dispel after you take this

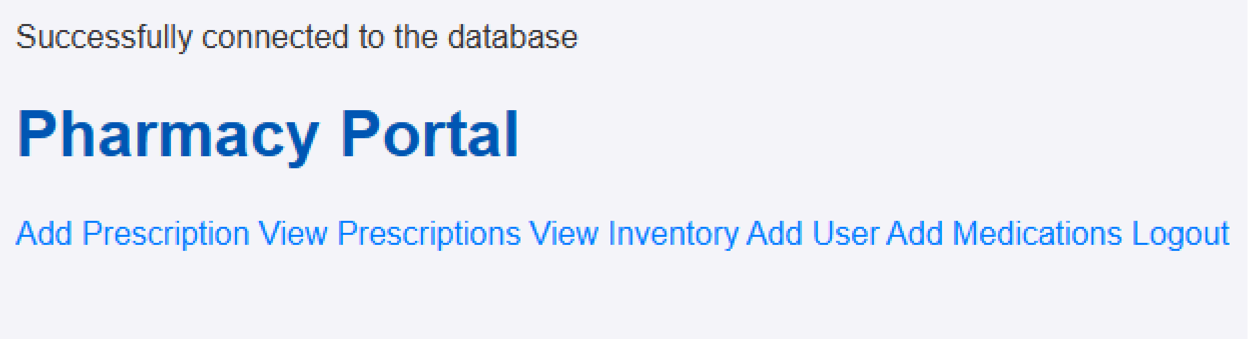
Quantity: 3

Prescribed on: May 11, 2025

Logout

This appears when the session is made and the user is a patient, if the page is attempted to be used without a session made that is either not logged in or isn't a patient, it would be redirected to the login page. The Pharmacy database is then loaded in to then fetches the userdetails that are pulled by the public function in the database page. A query is made for the userid, username, contactInfo, and the usertype which returns false if nothing there's. The next action then gets all the prescription of the user by joining the prescription and medication tables. The patient portal then loops through each prescription and displays the medication name, the dosage, instructions and the prescription date that are echoed from userdetails function. The only button on this page is a logout button that kills the session and kicks the user back to login page.

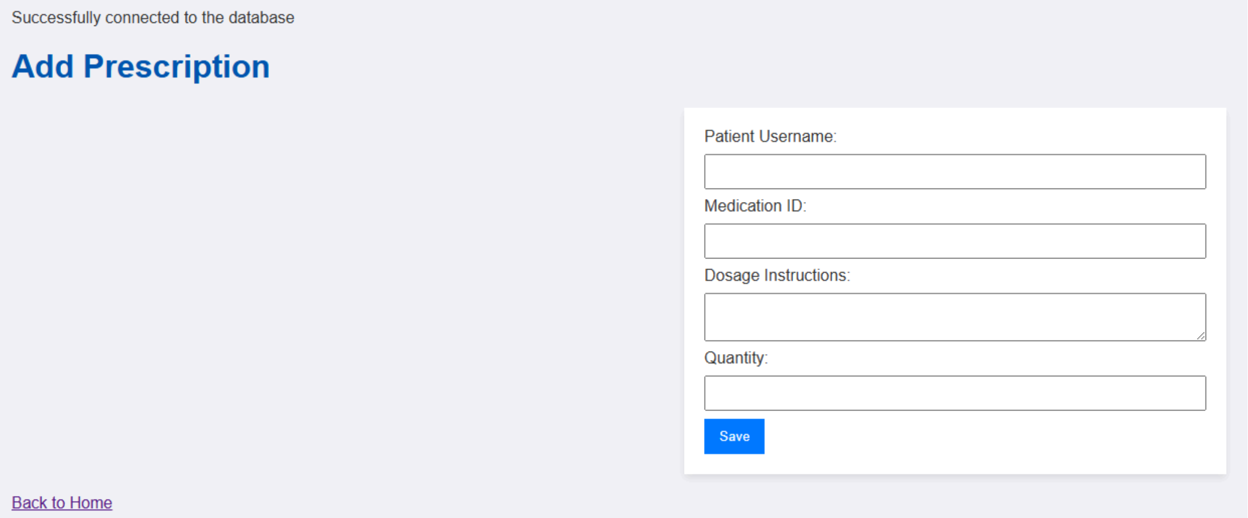
The second and more extensive portal that the user may use is that of the pharmacist portal. It has many uses, and pages. This is what will greet you when you login as a pharmacist



Successfully connected to the database

Pharmacy Portal

You can add and view prescriptions, view the inventory, add users, add medications, and logout. Lets go left to right, starting off with add prescription. This is the page in question.

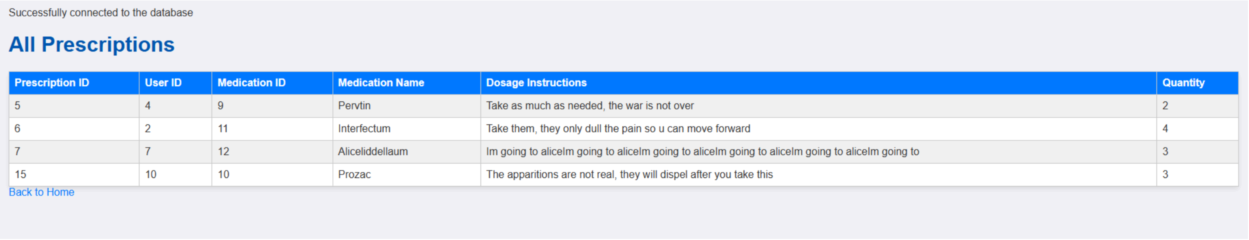


Successfully connected to the database

Add Prescription

This is a form that allows the user to make new prescriptions. How this is done is by the use of the information thats fed into the form that are then submitted to the php script private function addPrescription(). This triggers the addPrescription() that is from PharmacyServer.php and checks if the username is empty, med id is not zero, if the dosage is empty, or the quantity is more than zero to give an error message, otherwise it calls the add prescription function from the database page and sends the prescription added message. The function thats then adds the prescription into the prescription table by first looking if the patient exists for the username who is a patient, by looking in the user table. If the user is both in the system and is a patient, then the data is inserted into the prescription table using the information given and the date which is done automatically.

The next section I will talk about is that of all prescriptions. This is how it looks.



Successfully connected to the database

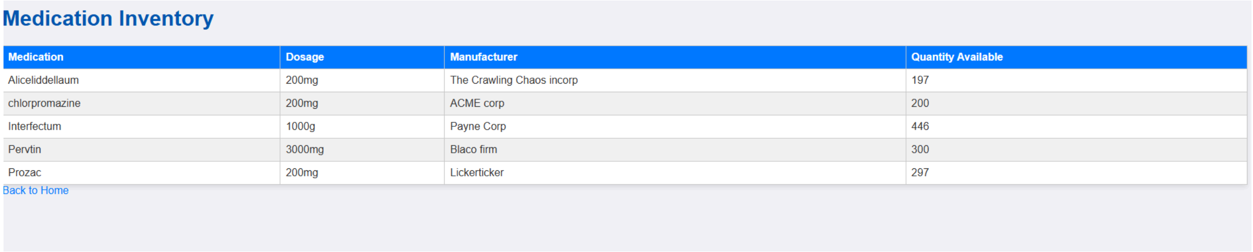
All Prescriptions

Prescription ID	User ID	Medication ID	Medication Name	Dosage Instructions	Quantity
5	4	9	Pervtin	Take as much as needed, the war is not over	2
6	2	11	Interfectum	Take them, they only dull the pain so u can move forward	4
7	7	12	Aliceliddellaum	Im going to alicelm going to alicelm going to alicelm going to alicelm going to alicelm going to	3
15	10	10	Prozac	The apparitions are not real, they will dispel after you take this	3

Back to Home

How this is made is by the getALLprescription public function, It pulls everything from the prescription table. This is then called by the private function that exists in the server file which calls the function that is then used by the view prescription that gives the form of the table and loops the rows until all data is filled.

The next button over brings you to the inventory page. This is where the inventory of the pharmacy is displayed for all to see.



Medication Inventory

Medication	Dosage	Manufacturer	Quantity Available
Aliceliddellaum	200mg	The Crawling Chaos incorp	197
chlorpromazine	200mg	ACME corp	200
Interfectum	1000g	Payne Corp	446
Pervtin	3000mg	Blaco firm	300
Prozac	200mg	Lickerticker	297

Back to Home

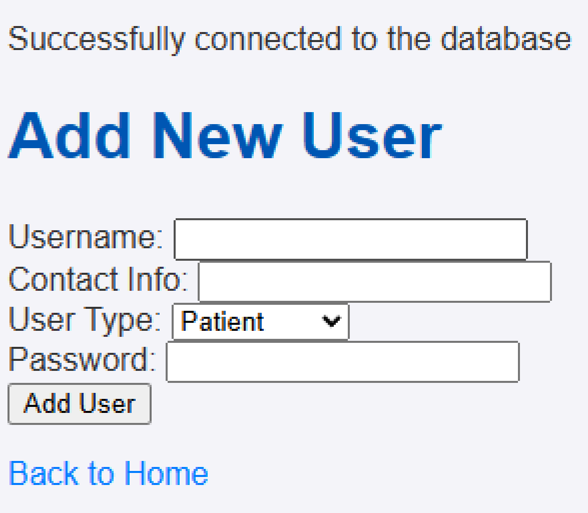
Aaron Campuzano

CIS 344

5/5/2025

This is achieved by using the view I made in MySQL, how this is achieved, in the database.php file, the query is made for the MedicationInventoryView and if it goes through, it pull up the views data, else an error is displayed. Then server file then passes the information over to the template view inventory file which takes each row of the view and displays it and loops said action until there is no more data left.

The next action is one of the most important since it what allows who has access to this entire website. It is the action of adding new users to the database.



Successfully connected to the database

Add New User

Username:

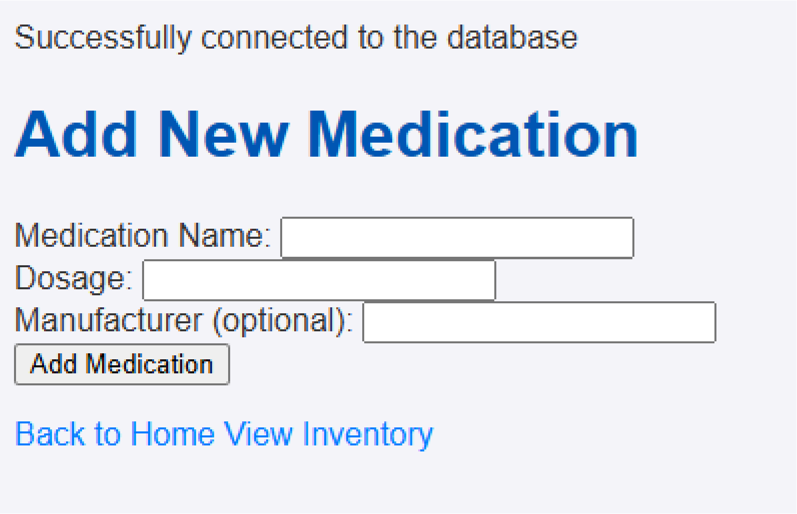
Contact Info:

User Type: Patient

Password:

This is the form that you will use, It is set up like the other that allows you to use the form to record some data that is then turned into 4 variables that are held onto at the server level then the database function then takes the data that was stored and make a query to insert the user into the user table.

The final action that you can take is adding a new medication. The form is similar to the page previously mentioned.



Successfully connected to the database

Add New Medication

Medication Name:

Dosage:

Manufacturer (optional):

It does the same song and dance of recording the input as variables that are then used by addMedication() public function. This is where the two differ, not only does this insert the data into the typical medication table, but inserts a corresponding inventory record to go with it. The other difference is that there is a button to directly view the inventory to make sure that you have inserted the data.

The challenges I have faced with making this website were quite discouraging. During the day before the submission time, my Xampp had corrupted and the files with it got taken down with it, I had to reinstall the application, thankful most of my files were backed up, I say most since i had to redo my patient portal page and my login page. Another thing I had an issue with was understanding how to implement and make a login page in the first place, but then i then looked at the exercise of lesson 16, the secret page. It detailed how to make such a thing work, as in using($_POST['name'] == 'user') && ($_POST['password'] == 'pass')) line really opened my eyes on how to make it work. What I did was use this example and modified it to work with the sessions I needed to implement. Which i already explained above how i did it.

Aaron Campuzano

CIS 344

5/5/2025

Aside from that, all the other issues were minor syntax errors that I ended up fixing along the way. It was a cycle of making a new line, testing, then looking at the code if there was an issue that needed to be fixed. Most of the time it was my fault for not ending a line, or forgetting either a />, ?>,), or a }. The easiest part was getting the database made since its the part of the program that is the most static.