

University of Alabama in Huntsville

LOUIS

---

Honors Capstone Projects and Theses

Honors College

---

4-19-2023

## Patient Information Management System (PIMS)

Patrick Burns

Jackson Collete

Follow this and additional works at: <https://louis.uah.edu/honors-capstones>

---

### Recommended Citation

Burns, Patrick and Collete, Jackson, "Patient Information Management System (PIMS)" (2023). *Honors Capstone Projects and Theses*. 783.

<https://louis.uah.edu/honors-capstones/783>

This Thesis is brought to you for free and open access by the Honors College at LOUIS. It has been accepted for inclusion in Honors Capstone Projects and Theses by an authorized administrator of LOUIS.

# Patient Information Management System (PIMS)

by

**Patrick Burns & Jackson Collette**

(Additional non-honors teammates: Brian Kemle, Madison Smith, & Bryson White)

An Honors Capstone

submitted in partial fulfillment of the requirements

for the Honors Diploma

to

The Honors College

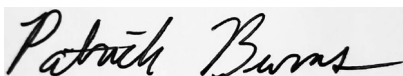
of

The University of Alabama in Huntsville

4/19/2023

Honors Capstone Director: Dr. Adam Colwell

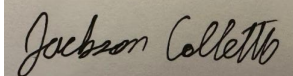
Part Time UAH Computer Science Professor



4/19/2023

Student (signature)

Date



4/19/2023

Student (signature)

Date



4/22/2023

Director (signature)

Date

Department Chair (signature)

Date

Honors College Dean (signature)

Date



Honors College  
Frank Franz Hall  
+1 (256) 824-6450 (voice)  
+1 (256) 824-7339 (fax)  
honors@uah.edu

### Honors Thesis Copyright Permission

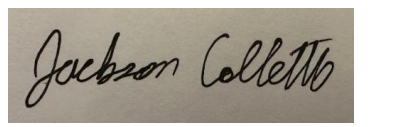
**This form must be signed by the student and submitted as a bound part of the thesis.**

In presenting this thesis in partial fulfillment of the requirements for Honors Diploma or Certificate from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department, Director of the Program, or the Dean of the Honors College. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

Patrick Burns  
Student Name (printed)

Jackson Collette  
Student Name (printed)

  
Student Signature

  
Student Signature

4/19/2023  
Date

## **Table of Contents**

Abstract	3
Introduction	4
Project Requirements	5
Architectural Design	7
GUI Design	9
Team Roles and Project Approach	15
Conclusion	17

## **Abstract**

Hospitals, and other medical institutions, have a need to store and easily access their patients' information. The Patient Information Management System (PIMS) accomplishes this goal. Due to the volume of patients, as well as the need to accurately and privately maintain patients' data, a system such as this is essential for any sort of medical institution. The Patient Information Management System strives to create an easy to use, secure, and reliable system for interfacing with patient data for any size medical facility. The code for this project can be found at the following GitHub repository: <https://github.com/patrickburns2557/PIMS-Project>

## **Introduction**

The objective of this project was to design a system that will allow a hospital to maintain current information on all patients, as well as those recently discharged. The system will allow this information to be readily available to all hospital staff that require it, as well as those who need to update any patient information as needed. To protect the privacy of patients, the users of the system will only have access to the information that they are allowed to see, depending on the type of hospital staff they are. The details of these user types, as well as the rest of the system are detailed in the following project requirements section. This project was completed by a team of five people across the course of a semester.

## **Project Requirements**

There were a number of features to be implemented in the patient management system project. The following is a detailed list of all of the features required to be implemented:

1. Users of the system must have a valid username and password to log into the system.
2. There shall be four types of user accounts: Doctor, Nurse, Office Staff, and Volunteer.
3. Doctors and Nurses shall have access to all available information about a patient.
4. Office Staff shall only have access to information required to identify a patient and information regarding insurance and billing.
5. Volunteers shall only have access to the name and location of a patient, the maximum number of visitors the patient may have, and the list of approved visitors for each patient.
6. A user may search for a patient by first name or last name. If the exact spelling of a name is not known, the user may enter a partial spelling, and any patients whose names contain the partial spelling will be displayed in a list. The user may then select the correct patient from the list.
7. Users may log on to the system from any computer connected to the hospital intranet.
8. The entire system shall have an appropriate graphical user interface.
9. Information on each patient shall consist of the following: First name, middle name, last name, mailing address, home phone number, mobile phone number, work phone number, names and numbers of two emergency contacts, date and time of admittance, reason for admission, name of family doctor, patient location (facility, floor, room number, and bed number), date and time of discharge, doctor's treatment notes, nurse's treatment notes, prescriptions administered while in the hospital (name, amount, and schedule), scheduled procedures, insurance carrier, insurance policy information (account number and group

number), billing information in the form of an itemized list of charges and amounts, amount paid, amount owed, and amount paid by insurance.

10. Doctors shall be able to enter additional notes on treatment, prescriptions, and scheduled procedures of each patient.
11. Nurses shall be able to view doctor's notes and enter treatment notes for nurses
12. Office Staff shall be able to view and update any information on a patient not related to medical treatment.
13. The system shall be capable of printing a variety of reports for a single patient or summary reports on all patients.
14. Users of the system shall be able to create new patients to add to the database.

The program was also subject to the following constraints:

1. Users must be able to run the program on a PC running Windows.
2. The program must be robust and assume that most users are not computer knowledgeable.
3. The system should be able to recover from errors, particularly input mistakes.

## **Architectural Design**

In order to begin the project, our team had to make several key decisions regarding the architectural design we would use for the project. Decisions had to be reached by the team for the primary programming language, how we would create the GUI, what type of database we would use to store the patient data, how we would test the code, and how we would structure the actual code. We decided to use Python for the primary language, MySQL for the database, CustomTkinter for the GUI, Pytest to create unit tests for the program, and a simple file structure that will be discussed later to organize the code. Additionally, the team used GitHub to store and share the code.

The system is primarily coded using Python. Python is an easy to use, easy to learn, and versatile programming language, which is why we picked it for this group assignment. Python allowed our group to focus on developing the actual application, without getting bogged down with syntactic and semantic issues when trying to use the language. Python also has support for a large number of libraries. We used many such libraries in the project, including the following: CustomTkinter, Pandas, Mysql-connector-python, and Pillow. CustomTkinter is used to create the GUI (see the “GUI Design” section). Pandas is used for data processing. For this project, it is specifically used to organize the data for all (or just one) of the patients in the system before pushing that data to a summary text file. Mysql-connector-python is used to link the Python code to the SQL database that stores all of the patients and each patient's information. Finally, Pillow is used for loading images into the program and working with CustomTkinter to display them to the GUI.

MySQL is used to handle the database aspect of the project. This is accomplished by running a MySQL server on a local machine. Note that if this system were implemented in a real

hospital, this server would be run on a remote machine. In order to connect to a remote machine, only one line of code needs to be modified to specify the IP address of the machine running the server. A SQL database containing the patient information is then loaded onto the MySQL server, which can then be accessed using the mysql-connector-python library functions. This database allows the program to interact with patient information that is not hard-coded/stored into the program itself.

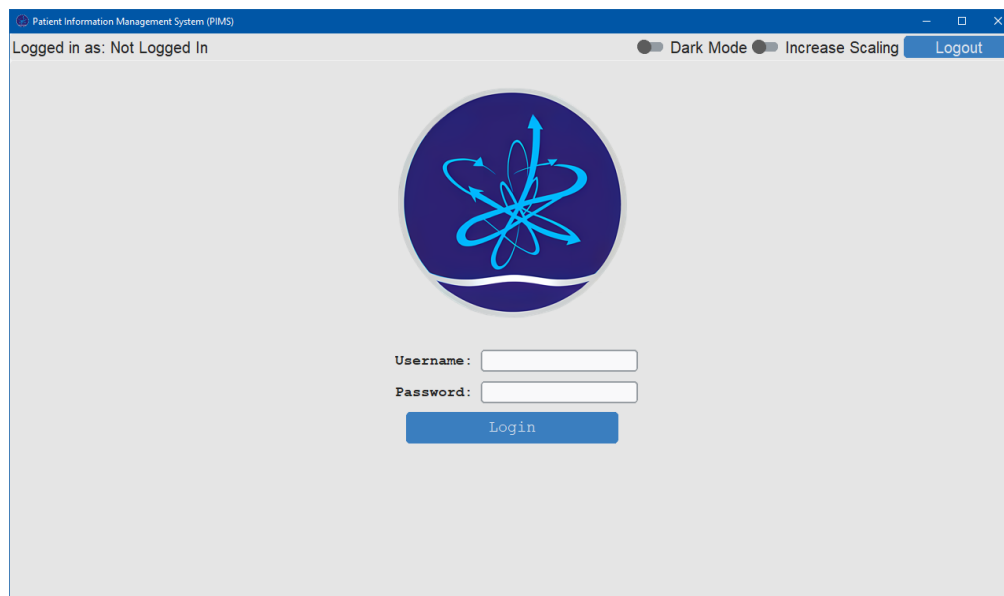
The testing framework used for the project is Pytest. Pytest allows for robust, easy to make unit tests for the project. We built unit tests for the front-end and back-end of the project, covering as much code as we could. These unit tests helped us catch errors and give us confidence that the users of our project will not run into any unexpected problems when using the software.

Our file structure is organized into three main divisions: data, GUI, and testing. As the name suggests, the data section handles all of the data for the program, including patient data loaded from the database, information about the current user, among other important information for the program. This section also includes the functions used to process and parse the data needed to display to the user. Similarly, the GUI section contains the files pertaining to the GUI. This part of the program handles displaying to the screen data from the users and providing users an easy to use interface with the program. The testing section contains the unit tests used to test the program; these tests are not needed for the program to be functional. The tests are run independently by the developer. The program also contains a driver file which just initialized the data section and the GUI section.

## GUI Design

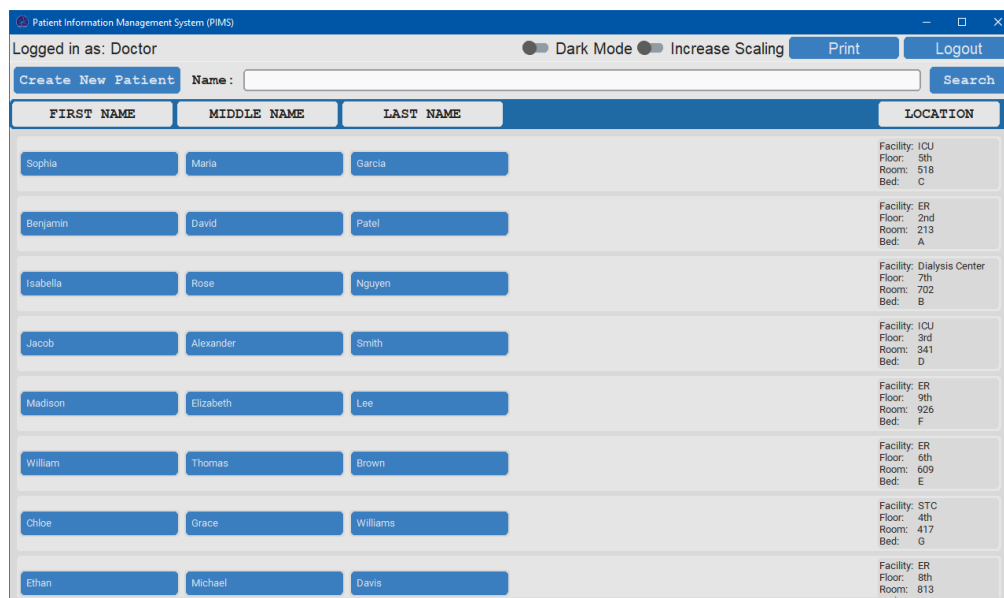
As previously mentioned, for the GUI of our project, we decided to use the CustomTkinter library. However, this wasn't the first pick. We were originally going to use TkInter for our frontend, but we found out early on that there were some shortcomings of it, such as the lack of an easy way to create an object that used a scrollbar and other various issues that seemed inherent to the library. CustomTkinter is a library built off of the original library that not only gives it a more modern look, but provides easy access to more complex GUI objects without the hassle of implementing them ourselves like would need to be done in regular TkInter.

In our frontend, we created several classes that extended the existing CTkFrame class in CustomTkinter that corresponded to the various views that the user will see while using the system. This allowed us to place these views anywhere we needed in our program and have them resize and populate their respective areas as needed.



*Figure 1: LoginView*

The first view that the user sees when running the program is the LoginView and the TopBar. The TopBar is shown at all times while the program is open, and it shows the type of user currently logged in, as well as has a button to allow the user to log out. It also allows the user to change options such as enabling dark mode and increasing the scaling to make all objects and text larger. The LoginView is what shows the text boxes for the user to enter their username and password. Once the user has typed their username and password, they can either click the login button, or simply press “Enter” on their keyboard to submit. If the user enters valid credentials, then the system will load the appropriate patient information from the MySQL database and the user will then be taken to the ListView screen.



FIRST NAME	MIDDLE NAME	LAST NAME	LOCATION
Sophia	Maria	Garcia	Facility: ICU Floor: 5th Room: 518 Bed: C
Benjamin	David	Patel	Facility: ER Floor: 2nd Room: 213 Bed: A
Isabella	Rose	Nguyen	Facility: Dialysis Center Floor: 7th Room: 702 Bed: B
Jacob	Alexander	Smith	Facility: ICU Floor: 3rd Room: 341 Bed: D
Madison	Elizabeth	Lee	Facility: ER Floor: 9th Room: 926 Bed: F
William	Thomas	Brown	Facility: ER Floor: 6th Room: 609 Bed: E
Chloe	Grace	Williams	Facility: STC Floor: 4th Room: 417 Bed: G
Ethan	Michael	Davis	Facility: ER Floor: 8th Room: 813

*Figure 2: ListView*

The ListView is what shows the list of all patients in the system to the user and allows them to select the patient they wish to access. Along with this, once signed in, a “Print” button is added to the TopBar. This allows the user to print a variety of reports depending on the view the user is currently looking at. For ListView, clicking the button will create and save a summary report to an external file on all patients that includes all information that the current user has

## Patient Information Management System

access to. At the top of the view, there's a search bar that allows the user to find a specific patient. Patients with first, middle, or last names that match the search term will be shown. If the user doesn't know how to spell their full name, they can enter a partial name, such as "mat", and then it will filter the list to show only patients whose names contain "mat" in them. To show all patients again, the user simply needs to clear the search bar and click "Search" again. If the user is logged in as a doctor, nurse, or office staff, a "Create New Patient" button will appear to the left of the search bar, otherwise it will be hidden. Clicking it will bring up the NewPatientView, which is discussed below. When the user clicks on the name of any of the patients in the list, they will then be taken to the PatientDetailedView for the corresponding patient.

The screenshot displays the Patient Information Management System (PIMS) interface, specifically the PatientDetailedView tabs for a patient named William Thomas Brown. The interface is divided into several sections:

- Top Bar:** Includes a search bar, a "Print" button, and a "Logout" button. The user is logged in as a Doctor.
- Navigation Tabs:** Back, Personal Information, Medical Information, and Billing Information.
- Personal Information:** Displays the patient's name (William Thomas Brown), address (6543 Cedar Rd, Buffalo, NY 14201), phone numbers (Mobile: (714)-984-4082, Home: (720)-725-1927, Work: (720)-182-9684), location (Facility: ER, Floor: 6th, Room: 609, Bed: E), emergency contacts (Blaine Brown, Blaze Rojas), and visitation information (Max simultaneous visitors: 3, List of approved visitors: Blaine Brown, Blaze Rojas, Nathan Kim).
- Medical Information:** Displays the patient's family doctor (Dr. Martin), prescriptions (Penicillin 100mg, Ceftriaxone 50mg), doctor notes (Patient is suffering from bacterial meningitis, needs surgical treatment to repair dural lacerations), nurse notes (Inflammations have decreased and vitals are stable), and scheduled procedures (Dural repair).
- Billing Information:** Displays the patient's insurance carrier (Kaiser Permanente), insurance policy info (Account Number: 3692581470, Group Number: KP006), list of charges (Penicillin \$120.42, Ceftriaxone \$230.67, Dural repair \$5670.20), amount paid (\$5021.29), amount paid by insurance (\$1000.00), and amount owed (\$0.00).

Figure 3: PatientDetailedView tabs

The PatientDetailedView is the screen where a user is able to see all of the information related to a patient in the database. If the user clicks the "Print" button in the TopBar from this

## Patient Information Management System

view, it will save and open an external file containing information only about the currently viewed patient. The PatientDetailedView is divided into 3 separate tabs: the Personal tab, the Medical tab, and the Billing tab. The user is able to click the corresponding button for each tab to reveal each tab's information. This view changes significantly depending on the type of user logged into the system. Doctors and Nurses will be able to view all information on a patient, so all three tabs will be available. Office Staff only have access to personal and billing information about a patient, so the Medical tab will be hidden for them. Office Staff will also see an additional button to the right of the Billing tab button that allows them to edit the current patient. Clicking this button will take the Office Staff to the EditPatientView. Volunteers will only be able to view the Personal tab, and the address and phone numbers on the personal tab will also be hidden to the volunteer. Lastly, there is a back button that can be clicked to bring the user back to the ListView discussed previously.

The screenshot displays the Patient Information Management System (PIMS) interface. At the top, a status bar indicates the user is logged in as 'Office Staff' and provides options for 'Dark Mode', 'Increase Scaling', 'Print', and 'Logout'. Below this, a navigation bar contains buttons for 'Back', 'Personal Information' (the active tab), 'Billing Information', and 'SavePatient'. The main content area is a form with three columns of input fields. The first column contains fields for First Name (Benjamin), Middle Name (David), Last Name (Patel), Emergency Contact 1 Name (Jaxon Patel), Emergency Phone Number 1 ((378)-103-439), Emergency Contact 2 Name (Omar Lucero), Emergency Phone Number 2 ((845)-198-034), Emergency Contact 3 Name, and Emergency Phone Number 3. The second column contains fields for Address Street (3579 Pine St), Address City (Athens), Address State (AL), Address Zip (35614), Max Simultaneous Visitors (3), Approved Visitor 1 (Jaxon Patel), Approved Visitor 2 (Omar Lucero), Approved Visitor 3 (Jada Thompson), and Approved Visitor 4. The third column contains fields for Facility (ER), Floor (2nd), Room (213), Bed (A), Mobile Phone ((238)-139-5932), Home Phone ((845)-643-3296), and Work Phone ((845)-248-1083).

First Name Entry	Address Street Entry	Facility Entry
Benjamin	3579 Pine St	ER
Middle Name Entry	Address City Entry	Floor Entry
David	Athens	2nd
Last Name Entry	Address State Entry	Room Entry
Patel	AL	213
Emergency Contact 1 Name Entry	Address Zip State Entry	Bed Entry
Jaxon Patel	35614	A
Emergency Phone Number 1 Entry	Max Simultaneous Visitors	Mobile Phone Entry
(378)-103-439	3	(238)-139-5932
Emergency Contact 2 Name Entry	Approved Visitor 1 Entry	Home Phone Entry
Omar Lucero	Jaxon Patel	(845)-643-3296
Emergency Phone Number 2 Entry	Approved Visitor 2 Entry	Work Phone Entry
(845)-198-034	Omar Lucero	(845)-248-1083
Emergency Contact 3 Name Entry	Approved Visitor 3 Entry	
	Jada Thompson	
Emergency Phone Number 3 Entry	Approved Visitor 4 Entry	

The screenshot displays the 'EditPatientView' interface within the 'Patient Information Management System (PIMS)'. The user is logged in as 'Office Staff'. The interface features a top navigation bar with 'Dark Mode' and 'Increase Scaling' toggles, and 'Print' and 'Logout' buttons. Below this is a tabbed interface with 'Back', 'Personal Information', 'Billing Information' (selected), and 'SavePatient' buttons. The 'Billing Information' tab contains two columns of input fields. The left column includes 'Insurance Carrier' (Aetna), 'Insurance Policy Account Number' (0987654321), and 'Insurance Policy Group Number' (AETNA002). The right column includes 'Add a Charge' (Benadryl), 'Add a Charge Amount' (525.6), 'Amount Paid' (525.6), 'Amount paid by Insurance' (0.0), and 'Amount Owed' (0.0).

*Figure 4: EditPatientView tabs*

The EditPatientView allows Office Staff to update the information on an existing patient. Similar to the DetailedPatientView for Office Staff, this view is also broken into Personal and Billing tabs that can be switched between. All of the existing information on the patient is shown in text boxes that can be edited by the user. Once the user has made the desired changes, clicking the “Save Patient” button will update all of the patient’s records accordingly and take the user back to the DetailedPatientView for that patient, with all of the edited values shown accordingly. If the user decides they wish to discard any changes made while in EditPatientView, they can simply click the “Back” button to be taken back to the DetailedPatientView without saving any changes.

As previously mentioned, Doctors, Nurses, and Office Staff will be able to see a “Create New Patient” button from the ListView. Clicking this button will bring the user to the NewPatientView. This view looks identical to the EditPatientView discussed previously, with the only difference being that none of the text boxes will already be filled out. As before, once the user has filled out all the information for the patient, they can click the “Save Patient” button.

## Patient Information Management System

This will add the patient to the database, and then take the user back to the ListView, where they will be able to see the new patient in the list.

## **Team Roles and Project Approach**

Each member of the team was assigned a role for the project. Patrick Burns and Jackson Collette, the authors of this report, alternated weeks as either the Team Lead or the Scrum Master since this was a requirement for the Honors Capstone project. For example, if Jackson was the Team Lead one week, then Patrick would be the Scrum Master for that week, and the next week those roles would flip. Patrick also handled the majority of the GUI development. The Team Lead for the project led meetings and made the key decisions for the overall project. The Scrum Master created and assigned individual tasks to the team members. Brian Kemle was the Technical Writer. The Technical Writer handled all the documentation and helped with the software development plan. Bryson White was the Quality Lead. The Quality Lead was responsible for ensuring the quality of the code as well as the quality of the overall project. Madison Smith was the Test Lead. The Test Lead was responsible for creating tests for the program. Note that as the project continued, Madison entered a development role, creating everything for the database and integrating it with the project, while Jackson handled some of the testing. Note that all team members were involved to varying degrees in developing the application in addition to their assigned roles.

The team met regularly three times a week: after class on Tuesday and Thursday, and on the weekends on either Saturday or Sunday night. The meetings after class typically lasted about 45 minutes, and the meetings on the weekends typically lasted about an hour. During the meetings, the team discussed updates on their progress as well as any issues they were having. Other team members were then able to help resolve those issues. Tasks were also assigned during the meetings, in addition to planning for the project.

## Patient Information Management System

The Scrum Master allocated tasks to individual members of the team. The group tried to break down everything that needed to be done for the project into tasks before we even started programming. Over the course of the project, as we discovered new tasks that needed to be completed, the Scrum Master would create new tasks on the project board throughout the semester as necessary. The Scrum Master assigned relevant tasks to each group member on a weekly basis, typically during the team's weekend meeting. The group used a GitHub project board to keep track of the tasks. A portion of the project board can be seen below. Tasks were organized under epics and user stories. Additionally, each task could be set as "Backlog," "In Progress," or "Done," as well as a designation for which team member was assigned the task.

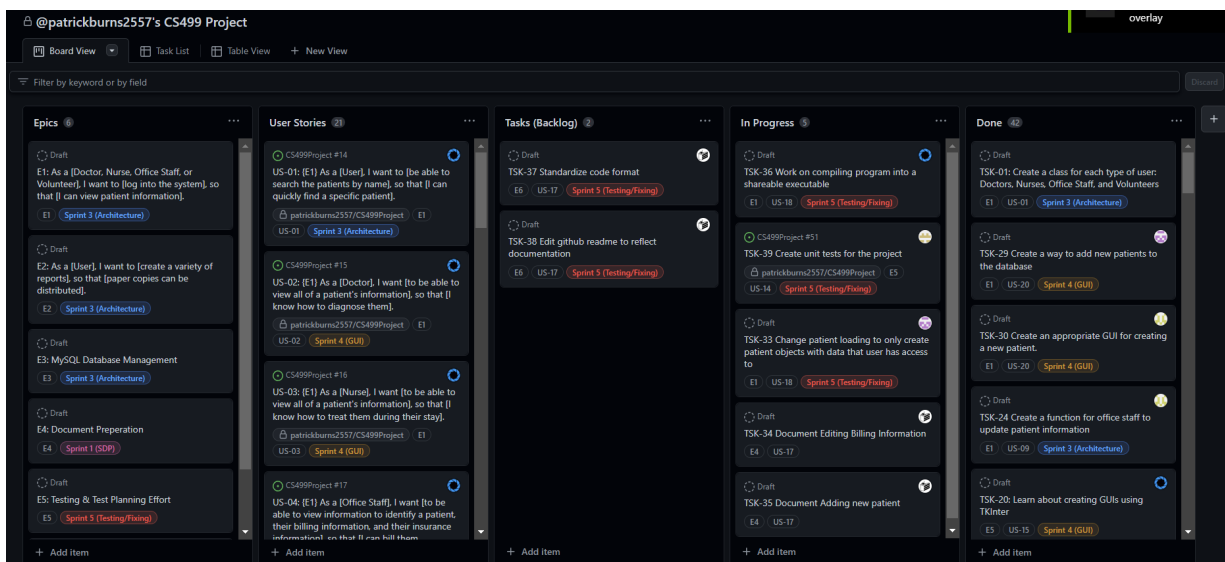


Figure 5: GitHub Project Board

## **Conclusion**

The Patient Information Management System set out to allow a hospital to view and manage data for each of their patients. The system is able to load all patient information from a MySQL database, whether it's on the local machine running the PIMS software, or on a remote machine on the same local network. Users are able to login to their accounts and view the appropriate information available to them. The system allows for the easy creation of new patients, as well as editing of existing patients. Lastly, the program allows users to print a variety of different reports to an external file concerning either a single patient, or all patients currently in the system. Overall, the system accomplishes all of the goals it set out to complete.