SOFE 3700U: Data Management Systems

Final Report

Supervised By: Dr. Khalid A. Hafeez

Project Title

Clinic Registration

Group Members and IDs

Group 10

Team Leader: Yash Patel - 100746810

Ammar Salmawy - 100756573

Poojah Karunakaran - 100618754

Raza Naqvi - 100754516

Kalapan Kannathasan - 100759041

Date: November 29, 2021

Table of Contents

Introduction

Digitizing documents and providing/receiving some services online have come a long way in serving millions of people around the globe in an easier and more efficient way. Before the information age, Looking for a document in a library archive could take up hours or days, whilst now it can be one click away.

In health care facilities, clinic staff must keep track of their patients' information, and doctors need this information to stay updated and follow up with their patients. Moreover, patients can also benefit from technology by having a more convenient way to receive health services, such as through the use of mobile applications.

The system we decided to design for our final data management project was a Clinic Registration system. The Clinic Registration system is designed to help users register for an account with the clinic, and also get prescriptions from doctors. There are two main components to the system, those being the doctor component and the patient component. For patients, after they finish registering/logging in, they can view their current active prescriptions, and book appointments with the doctors. For doctors, after they register/log in, they can view appointment requests from patients, and also prescribe and view prescribed medication. The system uses myPHP and mySQL for the database portion, while the front end portion or the GUI is designed using HTML and CSS.

Background and Motivation

People usually visit clinics when they need treatment or for general checkups. The process of seeing a doctor at a clinic is usually done in two ways; either through booking an appointment with the doctor by calling the receptionist, or by going to walk-in clinics without appointments. Clinic staff and doctors, on the other hand, need to schedule appointments for the patients, collect their information and keep track of their medical records.

The process of booking an appointment has many drawbacks. Firstly, it can be time-consuming sometimes due to long hold times, and it also has to be done during working hours. Second, the average wait time for a walk-in clinic is around 1 hour, and patients need to go through the same process for prescription refills as well. Lastly, doctors have to notify their patients when rescheduling is needed, which can take several calls if the patient is unreachable.

During the pandemic, hospitals and healthcare facilities became crowded, the pressure on health care workers increased dramatically, and people who needed treatment tried to avoid visiting clinics even when they needed medical aid, because of the long waiting times and the fear of getting the virus.

The system we came up with provides a solution for clinics to reduce the burden on receptionists, eliminate waiting time for patients, and make the way doctors and patients contact each other both easier and more convenient for both parties.

The system, unlike other available software, provides a doctors portal connected to the database so doctors can access and make changes remotely. It allows patients to ask for refills, and doctors to view prescriptions with the date they were prescribed so that there can be a better evaluation of a patient's progress, and so the refills can be authorized accordingly.

System Design

The premise for the website was created through brainstorming ideas that would allow a lot of concepts taught in the Data Management course to be put to practical use. Eventually, we concluded through the process of elimination that the best idea would be to create a pharmaceutical website that would use different queries and languages. It was decided that the data would be acquired from the users, who would be the doctors and patients. The respective user will have to create accounts and fill out forms with restricted inputs, which will be sent to the database. It was also decided that an admin should be included to oversee the database information, and that said information would only be accessed by users who knew how to access the admin page. To properly represent how the website will function, a flowchart was created, which can be viewed by referencing the appendix figure 1.

To understand how the database was going to be constructed, an ER diagram was made, which can be seen by viewing the appendix figure 7. The ER diagram walks through all the connections each variable has and how it links to one another. The diagram shows what type of relation each variable has such as one to one or one to many. The diagram also shows if variables are a weak entity set or unique, just to name a few.

To further facilitate understanding for the purpose of each user, a use case diagram was created, which can be viewed by referencing the appendix figure 2-4. The purpose of the doctor is to view all patients within the system, view the prescriptions they have prescribed, all prescriptions within the system, view appointments booked with the specific doctor, view all appointments, and ability to log out. The purpose of the patient is to view all doctors, fill in the prescription they have been given, view the prescriptions they have entered to the website, book appointments with doctors, view appointments that have been booked, and the ability to log out. Finally, the purpose of the admin is to create an XML document that can be viewed in tabular format, and view database views also in a tabular format.

After creating the previous diagrams we moved to writing the code which consists of the languages HTML, CSS, Javascript, PHP, SQL, and XML. HTML, CSS, and Javascript were used in the frontend to create a user interface that will assist users during their time on the website. PHP, SQL, XML, and Javascript were used to store data within the database and to use the information through the backend of the website. Below is a breakdown of the full logistics the pages use.

Upon starting the website, users are welcomed with an animated welcome page. This page has a navigation bar that lets users navigate through the website easily. The navigation bar consists of four links. These links take the user to an about us page, login/signup page, terms and conditions page and finally a home link which rerouters you back to the welcome page. The login page will check the entered credentials and verify if the credentials exist in the database. If the credentials exist, the page will confirm if the user is a doctor or patient and direct the user to the corresponding page. If the credentials do not exist in the database, the user is told the username/password does not exist. The signup page will prompt the user to select if they are a doctor or patient and direct them to the corresponding signup page. If the user signs up as a doctor, they are directed to the doctor's main page which has links such as view patients, prescription given, all prescriptions, appointments and all appointments. Each of these links will direct the doctor to proper pages. The doctor also has the option to log out of the account which takes you back to the welcome page. If the user chose to sign up as a patient, they will be directed to the patient main page. This page has links such as current appointments, new appointments, current prescriptions, new prescriptions, and view doctors. Each of these links will direct the patient to proper pages. The patient also has the option to log out of the account which takes you back to the welcome page. If you are the admin of the site, you can enter the required admin link directly into the address bar. This link will take you to the admin main page. The admin main page has links such as create patient XML, use patient XML, and all ten views. Each of these links will direct the admin to the proper pages (see appendix fig.1)

Observations and Analysis

In terms of this project's business case model, we were fortunate enough to create a service that we feel is essential, especially when it comes to today's time. Because of the devastating impact that covid-19 had on our healthcare and social systems, it became a necessity to have a process that allowed for online accommodation when seeking medical aid. By creating a website that revolved around the booking of appointments, as well as the ordering of prescribed medication, we successfully managed to cater to the issue at hand. With this service, social distancing guidelines that have been implemented by the government will be easy to follow, and healthcare services (such as those offered by hospitals and clinics) will not be affected in the process.

The website database utilized queries as questions or data requests made to the system, in order to acquire and retrieve key information for the data tables. Due to how omnipresent they were in the code itself, all the functions of the project ran smoothly and without any notable issues. There were two main types of queries which were used in the code, those being the "insert" and "read" queries, with each of them having several steps in their process. The insert query initially connects to the database, then uses the "POST" command to get data from forms. Finally, the data that is retrieved from the forms is then sent on to the tables. The read query on the other hand has two steps, where first the email that is passed through the URL is used, and then you read from the table using that email, or data obtained

from the email.

The website itself is polished and user-friendly, with a simple interface that allows people of all ages to utilize its services. In terms of cost-effectiveness, it is minimal in its expenditure and upkeep, thus making it an ideal business model to implement for whosoever decides to use it. It is also unobtrusive in the amount of space it requires, while also holding much potential for expansion on its scope and functions. For example, a potential implementation that could be made to this project would be a remote database. This particular website was designed for one use at a time, whereas remote databases would allow for several users to access the server at the same time. This will be useful within the healthcare industry if the same patient is visiting a different branch of a system, or if an employee is working in a different branch of the same healthcare facility. Information will be readily available for transfer within branches, which will additionally provide aid as backup servers when there is a server down at one location.

Furthermore, with future updates made to the database and website, it will be possible to implement several different features and processes to the finished product. The goal in doing so would be to improve the utility and functions of the service being provided by our webpage. For example, the database can be overhauled so that both doctors and patients can reset their passwords without needing the help of an admin. Additionally, doctors will be able to cancel both appointments and prescriptions, appointments will be able to be booked in accordance with the doctor's availability in their work schedule, and there will be some form of direct communication set up between the patients and doctors over the web. All of these updates will allow the website to diversify the number of tasks it can handle over the internet, thus allowing us to accomplish our goals when it came to the creation of this project.

Conclusion

In this project, a database system for a clinic registration system, where the information of both doctors and patients is set up using PHP, SQL, HTML, and CSS were used for the design aspect. There is also a third user, the admin, who overlooks and mandates the database server, and has access to those fields.

There were a lot of challenges that were encountered throughout the final project, such as the implementation of XML, learning about its usage, sending queries through the URL, and having the database system check for already existing or registered email accounts. However, we were able to overcome these challenges and satisfy the requirements for the final project.

Some of the future work improvements that our group talked about were replacing email accounts with ID numbers for employees, password entry and remote databases. In a healthcare facility, it would seem inappropriate for employees to sign in with personal accounts, especially when there is sensitive information involved. However, providing emails within a specific domain representing the clinic, or rather, in this case, we favoured ID numbers as it was a more secure way into logging in as an employee. Secondly, as mentioned, this would be a

database server handling extremely sensitive information about the patients and doctors. We were open to several options that would make entry into the system more secure. Some of the topics that were discussed were hashing passwords, giving password restrictions while providing a password strength test and a two-way password authentication. Thirdly, as mentioned in the observations, a remote database would be rather beneficial if the clinic were to expand to other locations. This would allow users to access their information at other locations remotely or at the same time.

In conclusion, this final project gave us insight into the processes and construction of a database system, which caters to a specific purpose. This experience was also accompanied by the development of our skills when it came to the writing of several different reports throughout the course of the semester. To be able to build a database for a real-life scenario, in this case, a clinic registration system, we were able to learn about the potential back-end processes of how a healthcare system facilitates its own information using databases. We believe that the time that we spent on this endeavour was tangibly valuable, and that we can utilize what we've gained from this project to better assemble and compile more complicated websites and databases in the future.

References

[1] yash-patel268, “System_Flowchart_Diagram,” *GitHub*, Nov. 25, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/System_Flowchart_Diagram.png (accessed Nov. 25, 2021).

[2] yash-patel268, “Doctor_Usecase_Diagram,” *GitHub*, Nov. 25, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/Doctor_Usecase_Diagram.png (accessed Nov. 25, 2021).

[3] yash-patel268, “Patient_Usecase_Diagram,” *GitHub*, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/Patient_Usecase_Diagram.png (accessed Nov. 25, 2021).

[4] yash-patel268, “Admin_Usecase_Diagram,” *GitHub*, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/Admin_Usecase_Diagram.png (accessed Nov. 25, 2021).

[5] yash-patel268, “Phase 1_ Project Proposal,” *GitHub*, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/Phase%201_%20Project%20Proposal%20.pdf (accessed Nov. 25, 2021).

[6] yash-patel268, “Phase 2_ Clinic Regstration (updated),” *GitHub*, 2021. https://github.com/yash-patel268/Data-Management-Project/blob/master/Formal%20Report/Phase%202_%20Clinic%20Regstration%20(updated).pdf (accessed Nov. 25, 2021).

[7] Y. Patel, “Data-Management-Project,” *GitHub*, Nov. 29, 2021. https://github.com/yash-patel268/Data-Management-Project (accessed Nov. 30, 2021).

Appendices

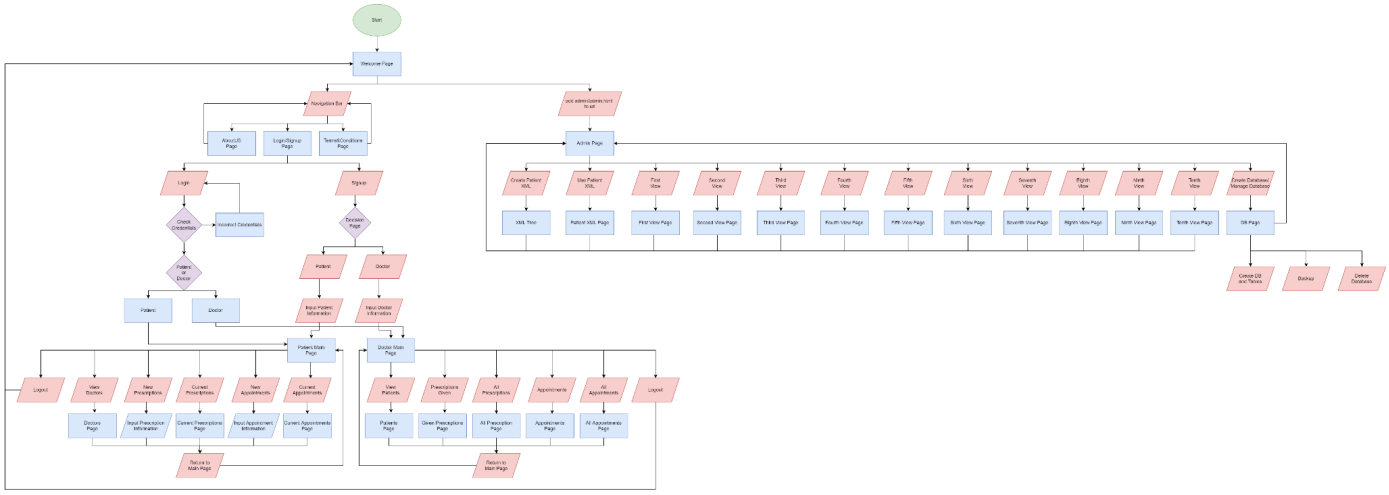

Figure 1: Flowchart of the website, Source: Adapted from [1]

Figure 2: Doctor Use Case, Source: Adapted from [2]

Figure 3: Patient Use Case, Source: Adapted from [3]

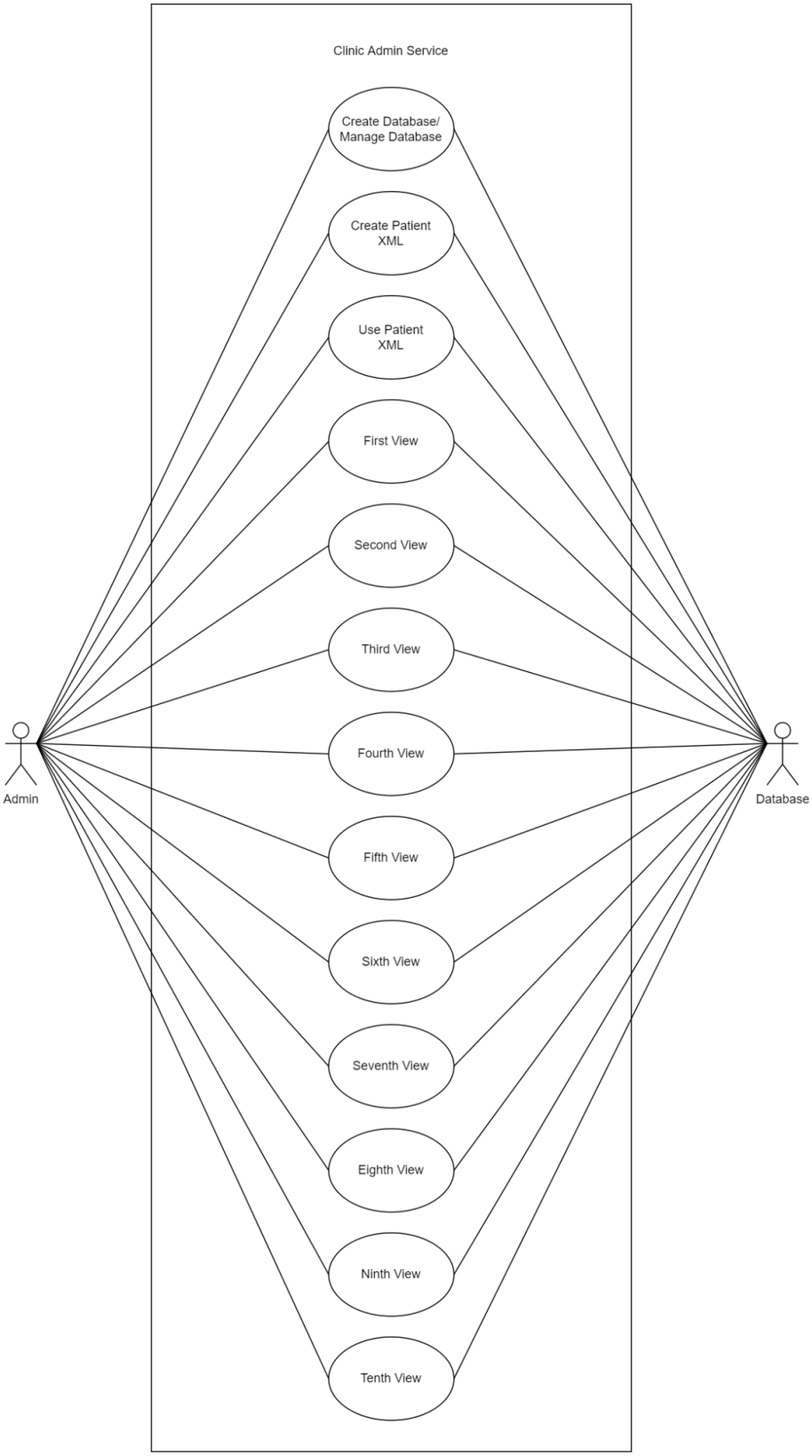

Figure 4: Admin Use Case, Source: Adapted from [4]

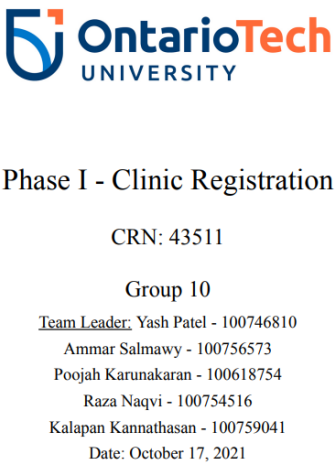

Figure 5: Project Phase 1, Source: Adapted from [5]

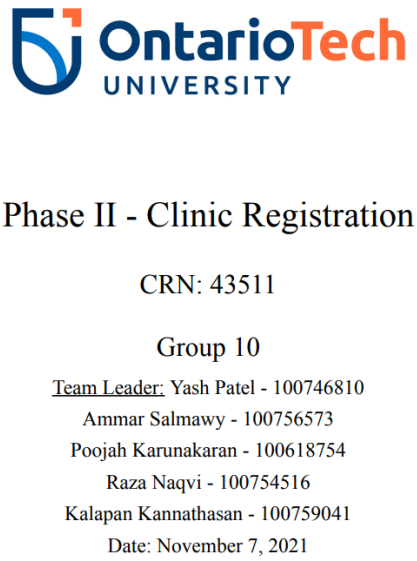

Figure 6: Project Phase 2, Source: Adapted from [6]

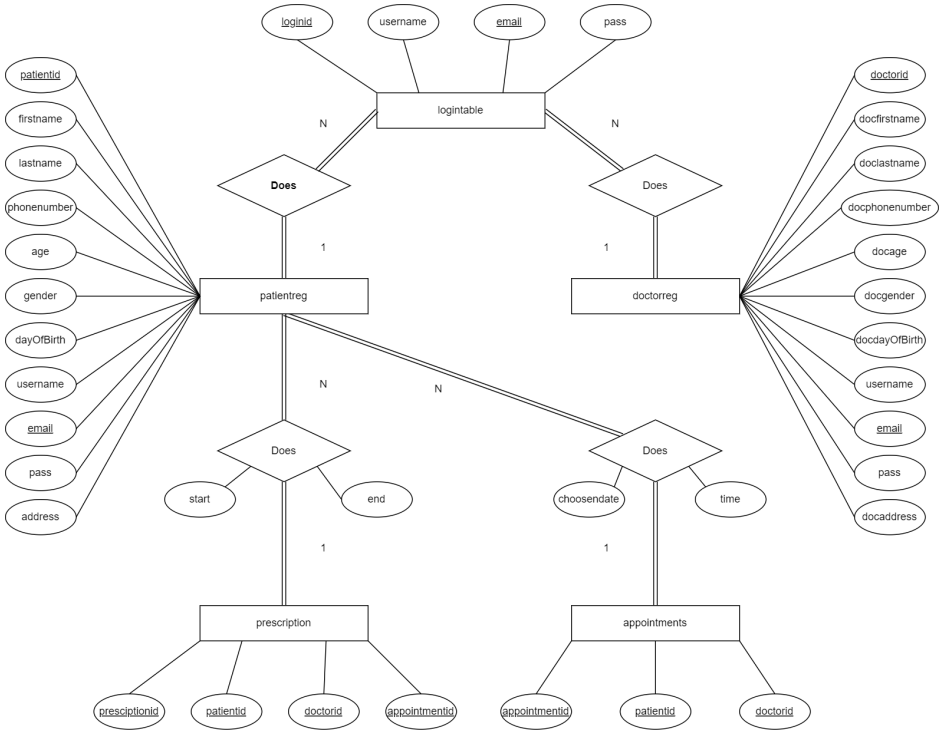

Figure 7: Project Phase 2, Source: Adapted from [6]

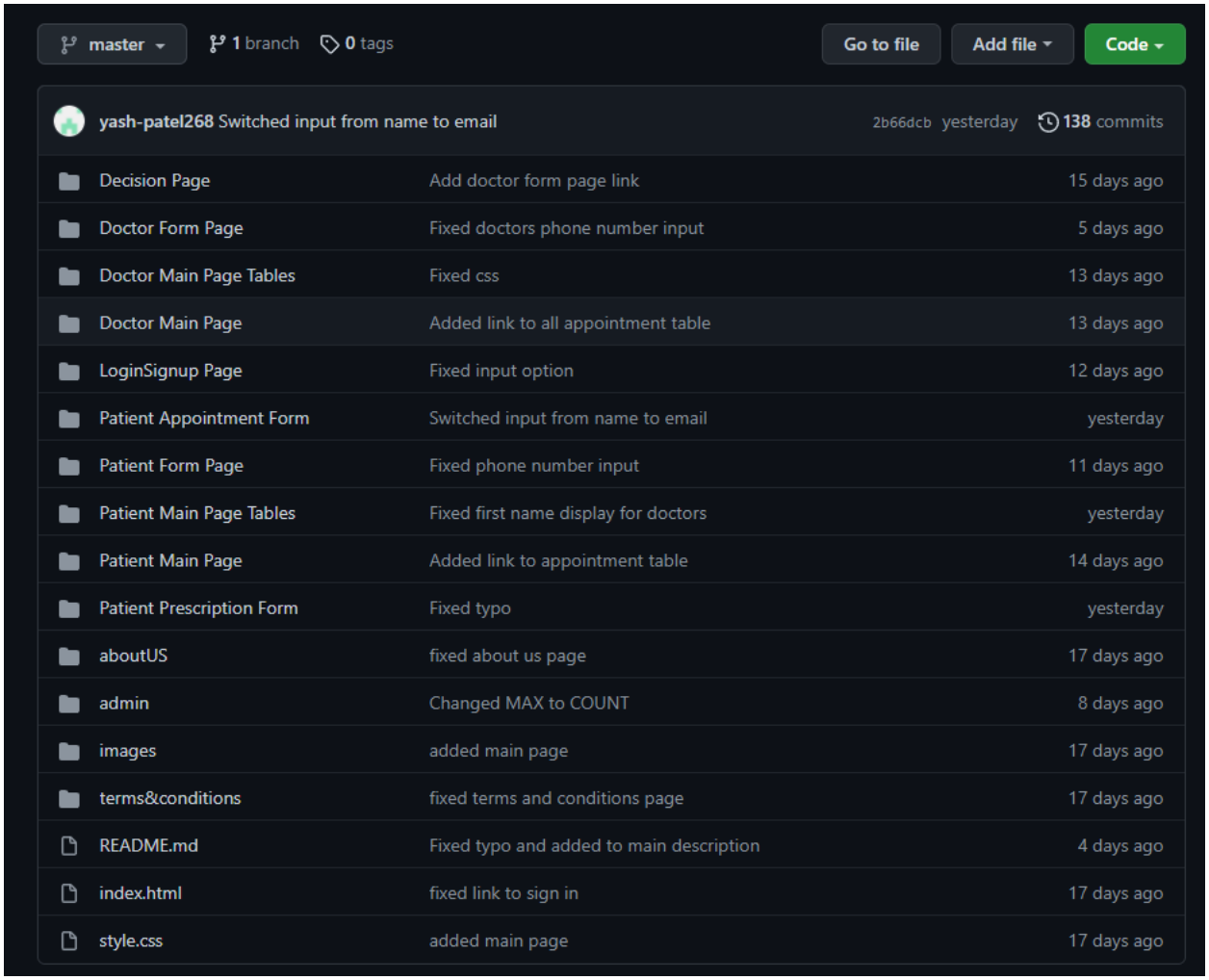

Figure 8: Github repo, Source: Adapted from [7]

To run the project, first, navigate to Yash Patels GitHub. It is located at the following link. From there, click the green code button and choose how you would like to download the project. For simplicity's sake, click download zip. After downloading the zip folder, extract it. After extracting the file, make sure you use wamp or any other service to host the database. Since we used wamp to host the database, we will walk you through how to download and get it ready. For downloading wamp, first head over to the wamp server website. It is located at this link. Once there, download the latest version of wamp server. After downloading, follow the pop-ups and install it onto your computer. This website walks you through the installation if you are lost. After installing the wamp server, navigate to the www folder inside wamp and paste in the unzipped project file. This www folder is located within the wamp folder which should be under your pc root folder. After pasting the file in, click the "index.html" file located in the "Data-Management-Project-master" folder. After clicking the index file, the file will open up on the browser. Now you can use the navigation bar to navigate through the website.

Contribution Matrix

Discord was used with the communication and organization of the project. All links and files were sent through discord. Due to privacy issues, screenshots will not be included in the report.

Name + Student ID	**Percentage of Participation for Project**
Yash Patel 100746810	22%
Ammar Salmawy - 100756573	19%
Poojah Karunakaran - 100618754	19%
Raza Naqvi - 100754516	19%
Kalapan Kannathasan - 100759041	21%
Total	**100%**

Task + Date	**Who worked on it**
Group Members' Names: September 18th to October 1st	- Everyone worked on this portion of the project - Everyone gave their name + id - Kalapan emailed TA
Mini Report: October 1st to October 7th	- Everyone worked on this portion of the project - Poojah came up with the "**Project Idea"** - Poojah and Kalapan worked on **"Motivation"** - Yash, Raza, and Ammar worked on **"Mandatory DB System"** - Yash worked on the **"Implementation"**
Phase 1: October 1st to October 17th	- Everyone worked on this portion of the project - Work carried over from mini report - Yash updated **"Database Tables"** - Kalapan worked on **"Problem Statement"** - Yash worked on **"Related Work"** - Yash worked on **"Plan"**
Phase 2: October 17 to November 7th	- Not everyone worked on this portion of the project, but group members tried to communicate - Yash and Kalapan worked on the majority of this part of the project - Yash emailed TA, Heyam, about the issue with views as they needed to be changed due to them not working in the code

	- Yash **resubmitted**
Time slot booking: November 16th	- Everyone worked on this portion of the project - We booked on the day Heyam emailed about booking - All group members came to an agreement that we would book on Friday from 12:00 to 12:15 with Taghreed.
Phase 3 Project Presentation: November 7th to November 26th	- Everyone worked on this portion of the project - Yash worked on **"Intro and Demo"** - Ammar worked on **"Project aim, System Requirements, and Database Tables"** - Kalapan worked on **"System Features, and System Flowchart"** - Raza worked on **"Queries"** - Poojah worked on **"Conclusion"**
Phase 3 Project Code: November 7th to November 29th	- Not everyone worked on this portion, however, there was communication between group members, and they did provide assistance where they could - Kalapan and Yash did **"Front-end"** - Yash, with some help from Kalapan and Ammar, did **"Back-end"** - Yash made and updated the **"Github"**
Phase 3 Project Report: November 7th to November 29th	- Everyone worked on this portion of the project - Ammar worked on **"Introduction and Background and Motivation"** - Kalapan worked on **"System Design"** - Raza worked on **"Observation and Analysis"** - Poojah worked on **"Conclusion"**