ASSIGNMENT 1

SOFTWARE ENGINEERING FOR APOLLO HOSPITAL MANAGEMENT SYSTEM

NAME: LES PAUL RANALAN

STUDENT NO.: 2230727

MODULE NO.: SWE5203

MODULE NAME: SOFTWARE ENGINEERING

MARKING TUTOR: DR. REHNA KALAM

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

At Apollo Hospital, various day-to-day processes include Patient Registration, Appointment Booking, Check-in Procedures, Billing, Prescription Management, Diagnostic Tests, and Appointment Cancellation.

To enhance operational efficiency and brand value, Apollo aims to strategically implement technology, replacing the current paper-based system for patient records, appointments, and billing. Our task is to thoroughly investigate the existing system, its interfaces, and the computerized accounting system in place.

2. PROJECT PLANNING

Identification of tasks should be the very first step when it comes to project planning. Below are the Gantt chart and work breakdown structure (WBS) to further visualize what are the things that are needed to be done.

A. GANTT CHART

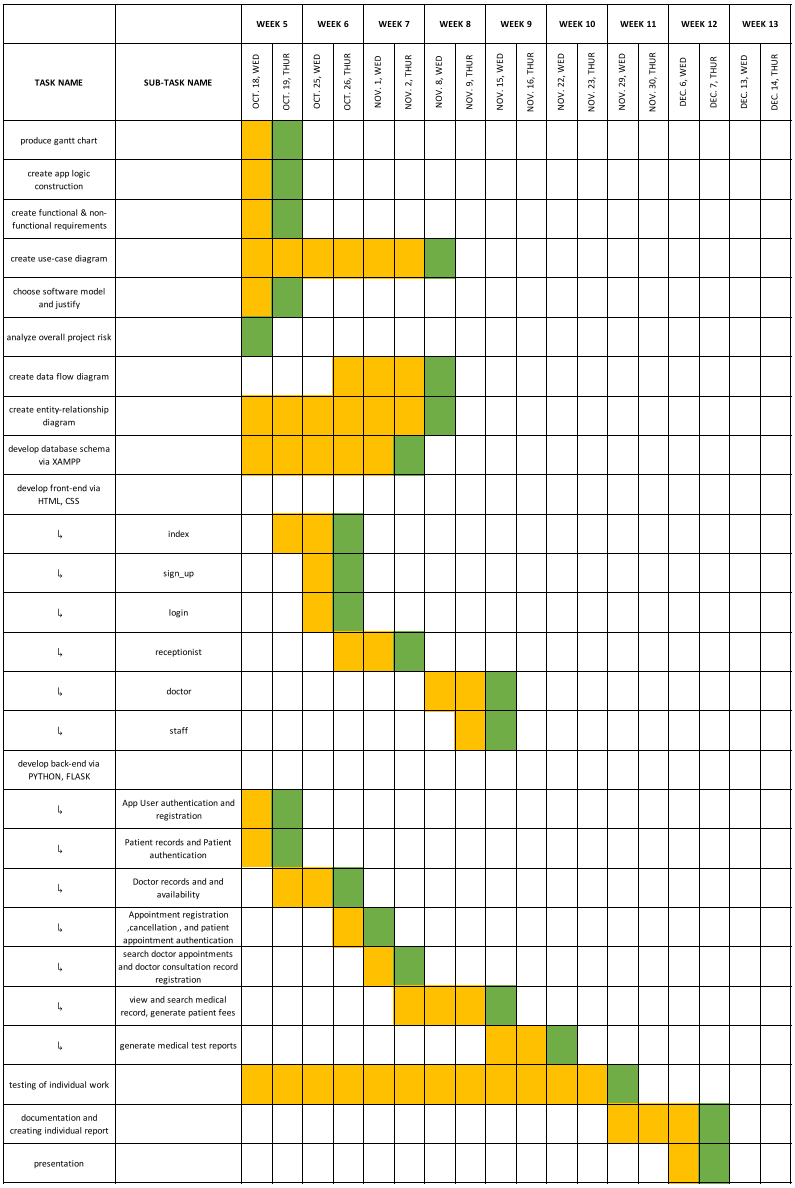


Figure 1: Gantt chart of overall project

A Gantt chart was employed to outline essential project tasks and create a timeline. Tasks are assigned to specific weeks and dates, aiding project management, especially for deadlines. The chart includes task names on the left, sub-tasks with arrows, and color coding: orange for in-progress tasks, green for completed tasks, and red for canceled tasks.

B. WORK BREAKDOWN STRUCTURE (WBS)

Below is the work breakdown structure for this project, which visualizes all tasks and sub-tasks that are needed to be completed. Note that the WBS is a summary of the entire workflow of the project. In comparison with Gantt chart, it shows more tasks in detail than the WBS.

Figure 2: Work breakdown structure (WBS)

C. SDLC PROCESS MODEL

SDLC is an acronym for “Software Development Life Cycle”. It is a structured and systematic process used for software development. It aims to produce software that meets customer expectations. In an SDLC, the typical phases or stages are:

- **Planning** – defining the objectives and resources required in a project
- **Feasibility Study** – evaluating technical and financial feasibility of the project
- **System Design** – creating detailed blueprint of system architecture of the project
- **Implementation** – writing code based on design specifications
- **Testing** – conducting various tests to identify and fix bugs
- **Deployment** – making the software available for users to use
- **Maintenance** – updating the software to fix bugs or add features

Figure 3: SDLC model

In this project, two SDLC process models were used: waterfall model and iterative model. Furthermore, all other models are defined with their own example diagrams.

I. WATERFALL MODEL

Waterfall model is a step-by-step approach wherein tasks are completed one after another, like a waterfall flowing downwards. Phases of waterfall model includes planning, design, testing.

PROJECT USAGE:

Waterfall model was used for this project. Although this model is mainly used for smaller and light projects, it would still outweigh the disadvantages of the waterfall model wherein the specified requirements are very clear and concise, as stated in the scenario.

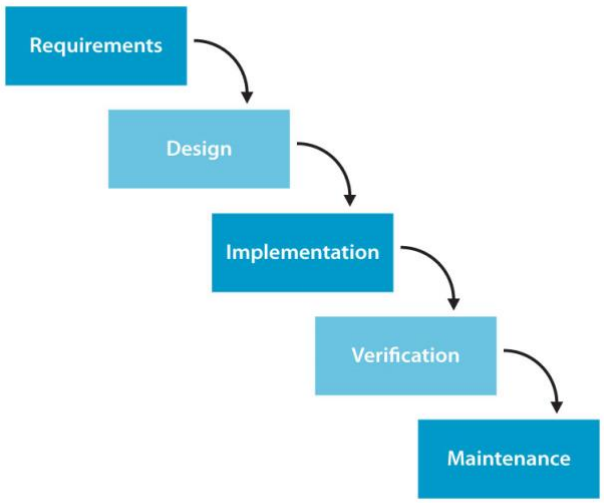


Figure 4: Waterfall model

ADVANTAGES:

- Simplicity and clarity
- Structure approach
- Best for small projects
- Documentation

DISADVANTAGES:

- Inflexibility
- Risk and uncertainty

II. ITERATIVE MODEL

The iterative model divides the project into small cycles involving planning, designing, implementing, and testing different software modules.

PROJECT USAGE:

The project utilized the iterative model, cycling through versions for testing, optimizations, and system enhancements until all software bugs were resolved. Iteration incorporated feedback, contributing to the project's continuous improvement.

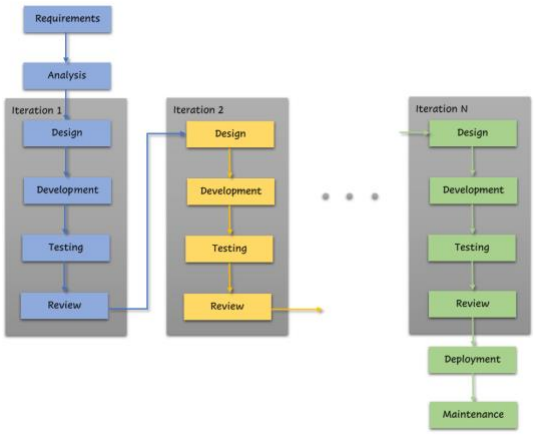


Figure 5: Iterative model

ADVANTAGES:

- Flexibility & adaptability
- Early delivery of core features
- Regular feedback
- Easier to manage changes

DISADVANTAGES:

- Higher resource requirements
- Extended development timeframes

III. AGILE MODEL

Agile is a flexible, customer-centric approach emphasizing iterative and incremental development. It delivers small functional software pieces quickly, adapting to changing requirements.

ADVANTAGES:

- Flexibility and adaptability
- Improved quality
- Collaborative approach
- Transparent progress tracking

DISADVANTAGES:

- Complexity in management
- Potential for scope creep
- Higher resource requirements
- Extended development timeframes

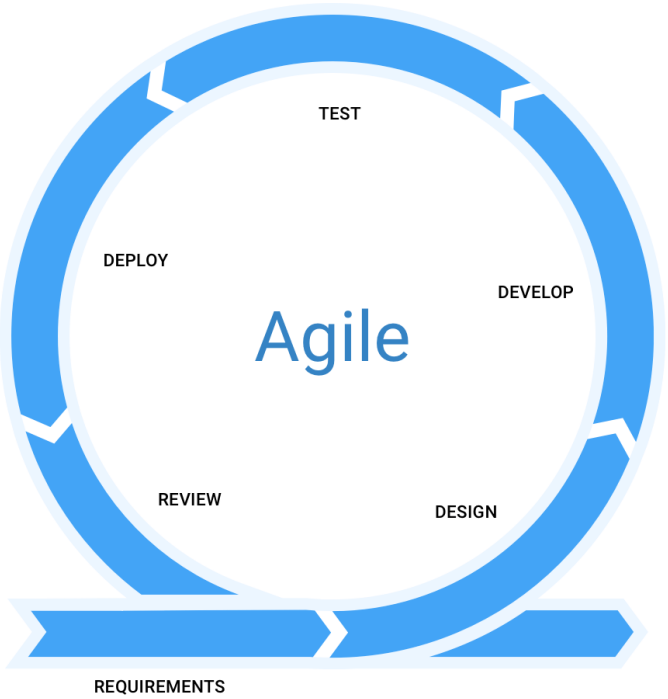


Figure 6: Agile model

IV. PROTOTYPING MODEL

Prototyping involves creating a simplified system version for feedback and requirement refinement. It's an iterative process where a prototype is built, evaluated, and modified based on user feedback.

ADVANTAGES:

- Clearer requirements
- Reduced risk of miscommunication
- Early detection of issues
- Increased flexibility

DISADVANTAGES:

- Time consuming
- Potential for scope creep
- Not suitable for ALL projects
- May not fully reflect on final system

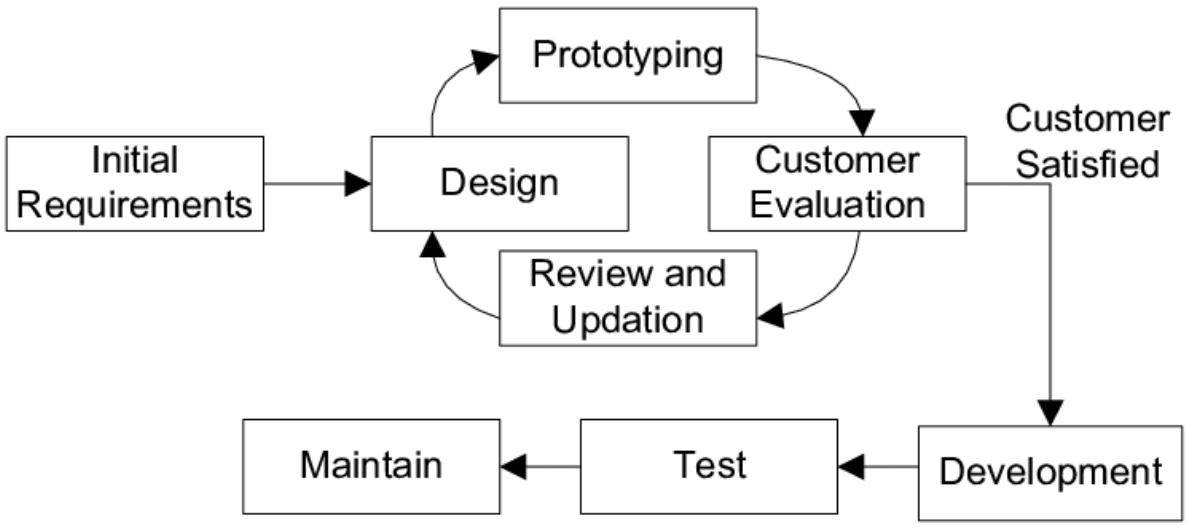


Figure 7: Prototyping model

D. RISK ANALYSIS TABLE

SOFTWARE RISK:

Software risk involves potential challenges in various aspects of software development, requiring identification, assessment, and mitigation for project success. Examples include technical compatibility issues, operational scalability challenges, business strategy changes, and external factors such as economic shifts.

INDEX TABLE:

Below is the index table which was used to rate the general likelihood and severity of harm against both technical and project-based risks.

		SEVERITY			
		Negibile - 1	Managable - 2	Critical - 3	Catastrophic - 4
PROBABILITY	Frequent - 5	5	10	18	20
	Likely - 4	4	8	12	16
	Occasional - 3	3	6	9	12
	Unlikely - 2	2	4	6	8
	Rare - 1	1	2	3	4

Figure 8: Index table

Likelihood - how often it can happen.

Severity of harm - how much it impacts/damages the overall project

TECHNICAL RISK ANALYSIS:

Below is the table for the technical risks. Presented in the table are description/name of the risks, along with its index rating which was referred above in the index table. Technical risks are mainly focused on software/code implementation in the project

Description	Index Rating
Unable to render HTML file	12
Unlabe to render CSS file	12
Authentication issue	6
Registration issue	6
No database connection	6
Unlabe to display data	2
Unable to manipulate data (add, insert, etc.)	10
Unable to create medical test files	10
Dependencies not installed	16
Loss of data	12

Figure 9: Technical risk analysis table

PROJECT-BASED RISK ANALYSIS:

Below is the table for the project-based risks. Presented in the table are description/name of the risks, along with its index rating which was referred above in the index table. Project-based risks are mainly focused on reports, diagrams, and documentation.

Description	Index Rating
Technical diagrams	8
No testing	8
No data gathering	2
Improper time management/procastination	20

Figure 10: Project-based risk analysis table

3. PROBLEM ANALYSIS

A. PREVIOUS SYSTEM

I. BREAKDOWN OF PREVIOUS SYSTEM

- **Patient Registration**
  - New patients provide personal details to the receptionist
  - Details are manually registered on a printed form, creating a physical patient file.

- **Appointment Booking**
  - Patients contact the receptionist for booking
  - Receptionists provide available time slots
  - Appointment details are maintained in a hardcopy file

- **Check-in Procedures**
  - Patients use a special terminal in the waiting area
  - Provide phone number and appointment number for check-in

- **Billing**
  - Receptionists generate a bill for specifying fees
  - Patients pay fees to the accounting department for a printed receipt

- **Prescription Management**
  - Doctors record outcomes and prescription in the patient's file

- **Diagnostic Test**
    - Patients visit specialized departments for x-ray or blood test

- **Appointment Cancellation**
    - Patients cancel appointments by contacting a receptionist
    - Receptionists update the hardcopy file with changes

II. PROBLEMS OF PREVIOUS SYSTEM

- **Data Risk**
  - Manual data entry increases the risk of inaccuracies, errors or duplication in patient records, billing, and prescription management
  - Having a lack of database or central system, records are at risk of damage or loss, which is critical for medical historical data

- **Limited Accessibility**
  - Printed records/files may limit the accessibility of patient information. This can lead to difficulty retrieving or updating records
  - Manual appointment scheduling may lead to difficulty in retrieving accurate availability of doctor and appointment slots
  - Manual appointment cancellation may lead to difficulty in updating the appointment record
  - Having the lack of database or central system, generation of medical reports may be hindered, leading to inaccuracies

- **Poor Patient Experience**
  - Having manual processes for patient registration or appointment scheduling can lead to longer wait times, which can contribute to negative experience for the patient

B. OUR SOLUTION

I. BREAKDOWN OF OUR SOLUTION

- **Patient Registration**
  - New patients provide personal details to the receptionist
  - Patient details are entered by receptionist onto the system via laptop or computer
  - Patient details are stored in patient records via centralized database
  - Patient details can be updated via patient records
  - Patient details can be deleted via patient records

- **Appointment Booking**
  - Patients contact the receptionist for booking
  - Receptionists provide available time slots through viewing doctor records in centralized database
  - Appointment details are maintained in appointment records via centralized database

- **Check-in Procedures**
  - Receptionist confirms identity of patient via searching in patient records in centralized database

- **Billing**
  - Receptionist views medical records via centralized database and generates bills for specifying fees
  - When generating bill, it prints out a PDF document.

- **Prescription Management**
  - Doctors record outcomes and prescription in the patient's medical records in centralized database

- **Diagnostic Test**
  - Patients visit specialized departments for x-ray or blood test
  - Staff views medical records of patients, in centralized database, and can generate specific patient's test report
  - Generating the test report, it will print out a PDF file

- **Appointment Cancellation**
  - Patients cancel appointments by contacting a receptionist
  - Receptionists update the appointment details in appointment record

II. ADVANTAGES OF OUR SOLUTION

- **Centralized Database**
  - This ensures that there is a single source of data that is being stored and being retrieved. This reduces the likelihood of data inconsistencies or errors.
  - All users can efficiently and quickly retrieve records from the database.
  - All users can collaborate more effectively since all data is from one source

- **Improved Data Security**
  - This allows for better control over access permissions and security protocols. This is important especially when dealing with patient's personal details.

- **Scalability**
  - This allows for better management of the amounts of data that is being stored. This is important since the hospital already has a lot of patients and can even grow further in quantity.

4. REQUIREMENT ANALYSIS & SPECIFICATION

A. REQUIREMENT DETERMINATION

Identify stakeholders (medical staff, administrators, patients) for requirement determination. The following outlines functional and non-functional requirements, providing insights into their significance.

B. REQUIREMENTS DEFINITION

I. FUNCTIONAL REQUIREMENTS

- **Reception**
  - Registration of patients
  - Viewing, deleting, modifying, and searching for patients based on their names
  - Verifying patients via their names
  - Registering appointments, ensuring validity of scheduled time
  - Viewing and cancelling of appointments
  - Accessing a list of doctors along with their availability
  - Searching for patient's appointments
  - Generating bills for specific patients by searching through medical records

- **Doctor**
  - Explore records of doctor’s appointments
  - Register medical records for patients, with prescription, findings, and tests

- **Staff**
  - Access all medical records of patients
  - Search records of medical information for patients
  - Create test reports for x-ray and blood test

II. NON-FUNCTIONAL REQUIREMENTS

- **Performance**
  - The system should feature decent performance, with significant number of users without the system becoming unresponsive

- **Scalability**
  - The system should be able to scale effectively to accommodate even more user data

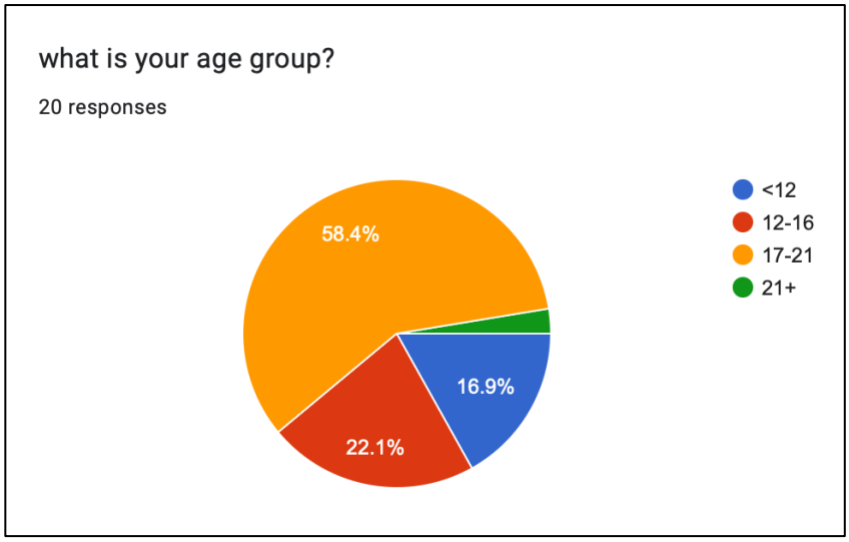
- **Reliability**
  - Should be up and always running, ensuring usage to users. Avoid sudden crashing

- **Compatibility**
  - Should be able to run on different operating systems and web browsers (Chrome, Edge, Safari, etc.)

C. REQUIREMENTS GATHERING

I. SURVEY FORM

We gather data from medical professionals for diverse perspectives. Input from experienced system management programmers enhances our understanding of system creation and its interactions in various scenarios, facilitating impactful adjustments.



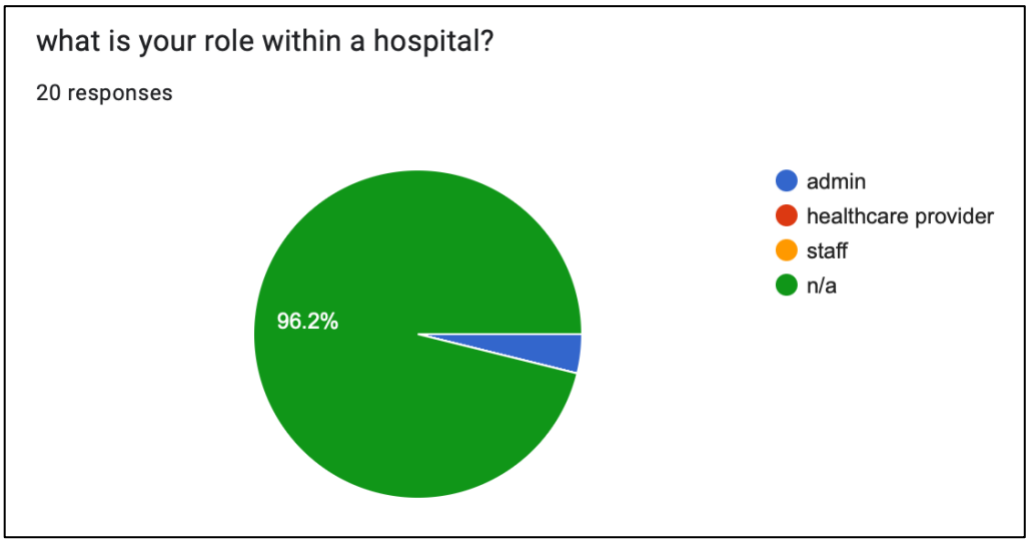


Figure 11: Pie graphs of survey results, general questions

General questions were asked to further specify the demographic groups of people who participated in the survey.

Figure 12: Text output of survey results, scenario questions

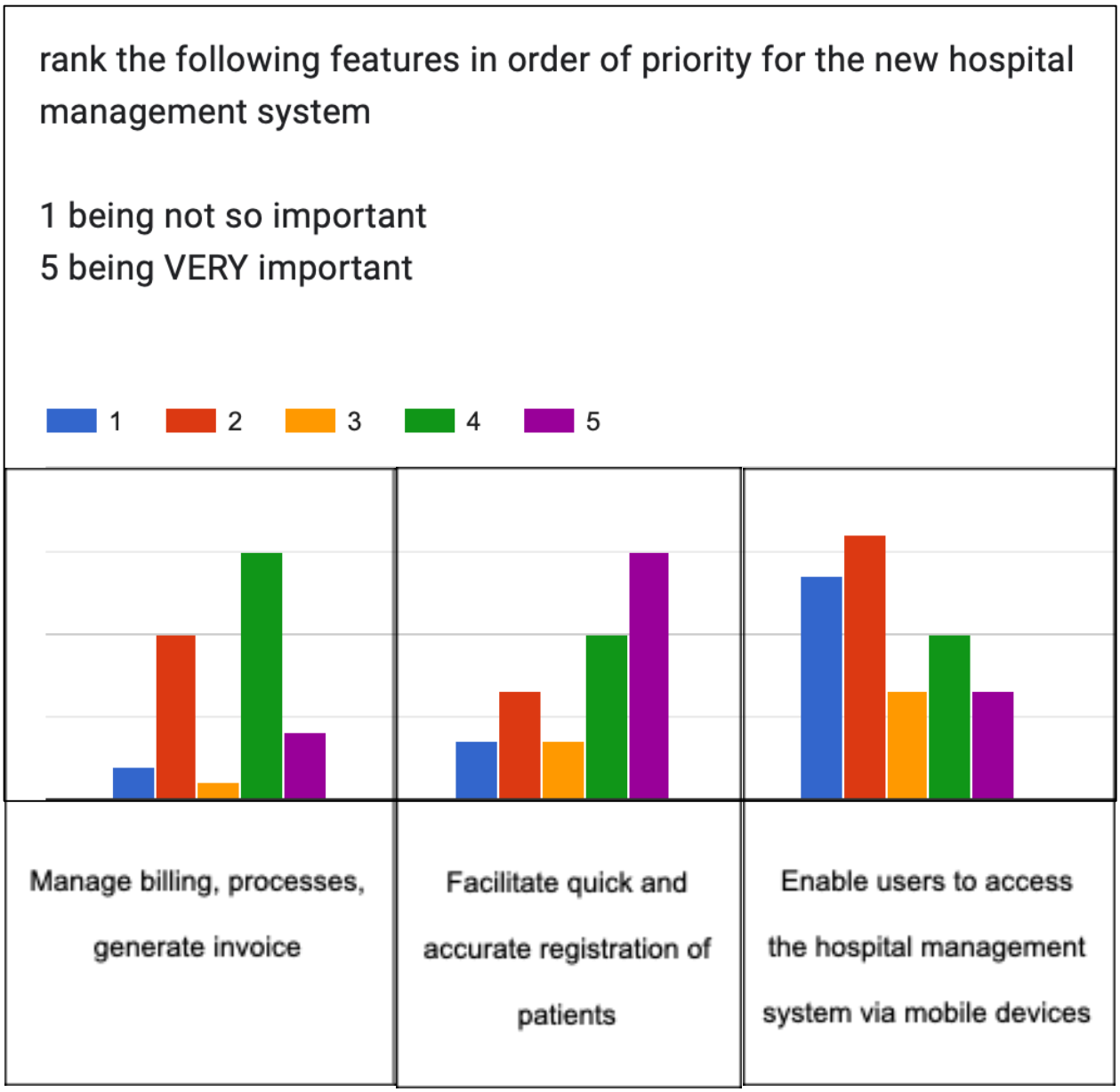
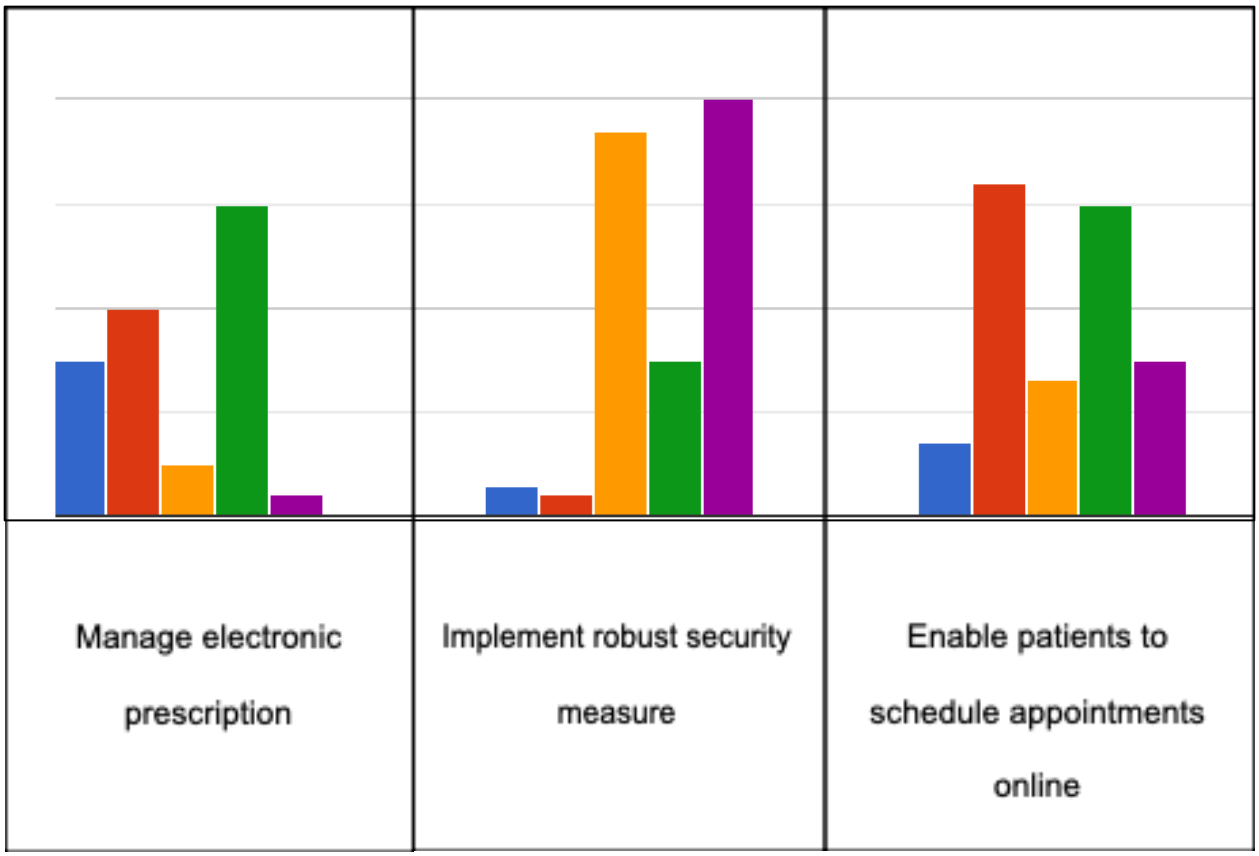


Figure 13: Bar graph of survey results, scenario questions (1/2)



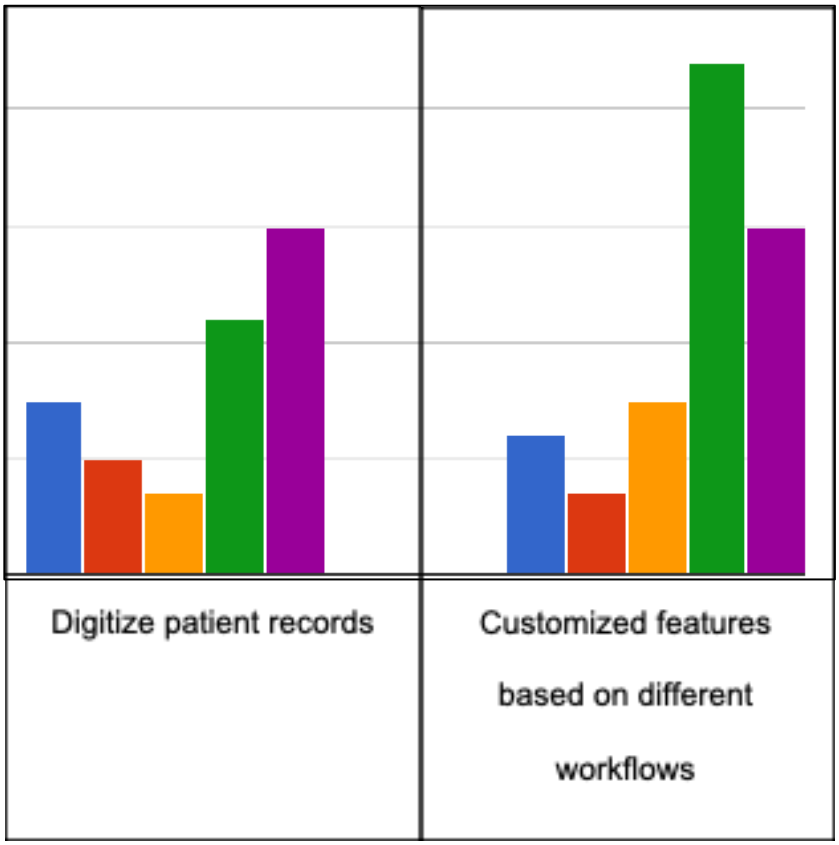


Figure 14: Bar graph of survey results, scenario questions (2/2)

Survey results show that most respondents are aged 17-21 and are not affiliated with a hospital role (civilian). They provided feedback on current system inefficiencies and suggested improvements. Additionally, responders prioritized features through bar graphs. Features such as:

- **Facilitating quick and accurate registration of patient**
- **Implement robust security measure**
- **Digitalize patient records**
- **Manage billing, processes,**
- **Generate invoice, manage electronic prescription**
- **Customized features based on different workflows**

II. INTERVIEW

Below are the questions for the interview that we gave to both individuals who are experienced in system management programming and in the medical field.

QUESTION: *"Common hardware for data management in hospitals?"*

RESPONSE: *"Hospitals use servers, storage devices, and network infrastructure. Servers store and process patient data, storage devices organize information, and network infrastructure ensures seamless communication between hospital departments and systems."*

QUESTION: *"How often do issues disrupt the existing system's workflow?"*

RESPONSE: *"Periodic issues like system crashes and software glitches occurred, impacting workflow. Emphasizes the need for a robust information management system for uninterrupted healthcare services." "What potential challenges might be anticipated in relation to the proposed system?"*

QUESTION: *"Insights into data retrieval: old vs new system?"*

RESPONSE: *"Previous data retrieval was cumbersome. The new system streamlines with improved interfaces, real-time updates, and validation checks, ensuring accurate and accessible information for healthcare professionals."*

QUESTION: *"How significant is the color scheme for the user interface, and what style is suitable?"*

RESPONSE: *"Color scheme is crucial for GUI development, impacting user experience. A soothing palette, like blues and greens, is preferred for healthcare environments, reducing eye strain and fostering a positive working atmosphere."*

QUESTION: *"Are printers heavily used for medical test reports?"*

RESPONSE: *"Yes, printers play a crucial role in generating hard copies of test reports and medical records, essential for documentation, reference, and sharing information with patients and healthcare providers."*

D. FEASIBILITY STUDY

I. TECHNICAL FEASIBILITY

The proposed system employs HTML, CSS, and JavaScript for the responsive frontend. Python Flask manages data processing and server-side logic, while XAMPP serves as the SQL database for storage and retrieval. Essential Python libraries/modules, including "Reportlabs" for reports, "mysql-connector" for database connections, and "Flask Compress" for performance optimization, make the project technically feasible.

II. ECONOMIC FEASIBILITY

The proposed system incurs no expenses in development and implementation, using open-source software and libraries. Operational costs may include up-time based on electrical usage, maintenance, system upgrades, and daily hardware use, making it economically feasible.

III. LEGAL FEASIBILITY

The proposed system ensures compliance with data protection laws. Access control through login authentication restricts information to authorized users. Centralized database storage enhances security, with room for additional features like encryption and security patches. The system is legally feasible.

IV. ORGANIZATIONAL FEASIBILITY

Upon implementing the proposed system, staff from the previous system must acquire knowledge of new technological solutions through online webinars or training programs. This ensures the necessary skills for using the system. A user manual is essential, making it organizationally feasible.

V. SCHEDULE FEASIBILITY

The system's planning, development, and implementation phases follow a project timeline, scheduling tasks efficiently. This ensures timely completion of deliverables and objectives, making the project feasible in terms of scheduling.

VI. COST-BENEFIT ANALYSIS

Below is the Cost-benefit analysis chart which covers all the cost of different components for Apollo Hospital to transition to our proposed system. Note that these costs may change over time and is currently only an estimation.

S/N	DESCRIPTION	UNIT PRICE (AED)	QUANTITY	TOTAL PRICE (AED)
01	Database server	AED3,000.00	1	AED3,000.00
02	4TB storage drives for database	AED800.00	2	AED1,600.00
03	Workstation computers	AED500.00	30	AED15,000.00
04	Mobile tablets	AED150.00	15	AED2,250.00
05	Fax machines	AED250.00	10	AED2,500.00
06	Security systems (firewalls, anti-virus software)	AED30.00	50	AED1,500.00
07	Windows 10 license	AED50.00	30	AED2,500.00
08	Microsoft Office license	AED100.00	30	AED300.00
09	Training programs	AED5,000.00	NA	AED5,000.00
10	Software development	AED0.00	NA	AED0.00
11	Operational adjustments	AED5,000.00	NA	AED5,000.00
12	Website application hosting	AED600.00 **PER YEAR**	NA	AED600.00
				AED39,250.00

Figure 15: Cost-benefit analysis chart

The proposed system prioritizes a robust database server and workstation setup for receptionists and doctors, including mobile tablets for flexible access. Training programs enhance personnel readiness, and operational adjustments cover transition costs. Web hosting incurs a yearly fee. Development costs remain at 0 AED, utilizing free and open-source programs. The total cost is approximately 40,000 AED, ensuring cost savings and operational efficiency for Apollo Hospital, ultimately enhancing patient care and staff satisfaction.

5. SYSTEM DESIGN

A. DATA FLOW DIAGRAM

- *LEVEL 0*

Below is the level 0 of the data flow diagram which showcases the basic processes of how the proposed system operates.

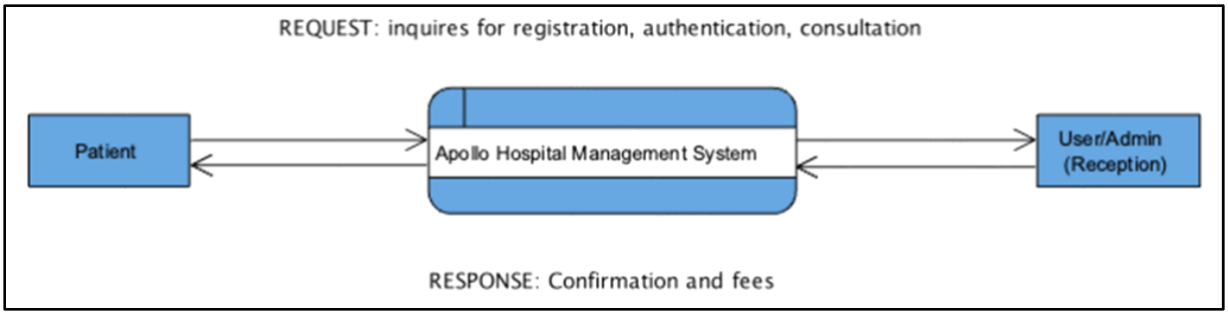


Figure 16: Level 0, data flow diagram of Apollo Hospital

Patient inquires for registration, authentication, and consultation. This process goes through Apollo Hospital, and then to the receptionists. Which then sends back confirmations and bills back to the patient.

- *LEVEL 1*

Below is the level 0 of the data flow diagram which showcases more processes of how the proposed system operates.

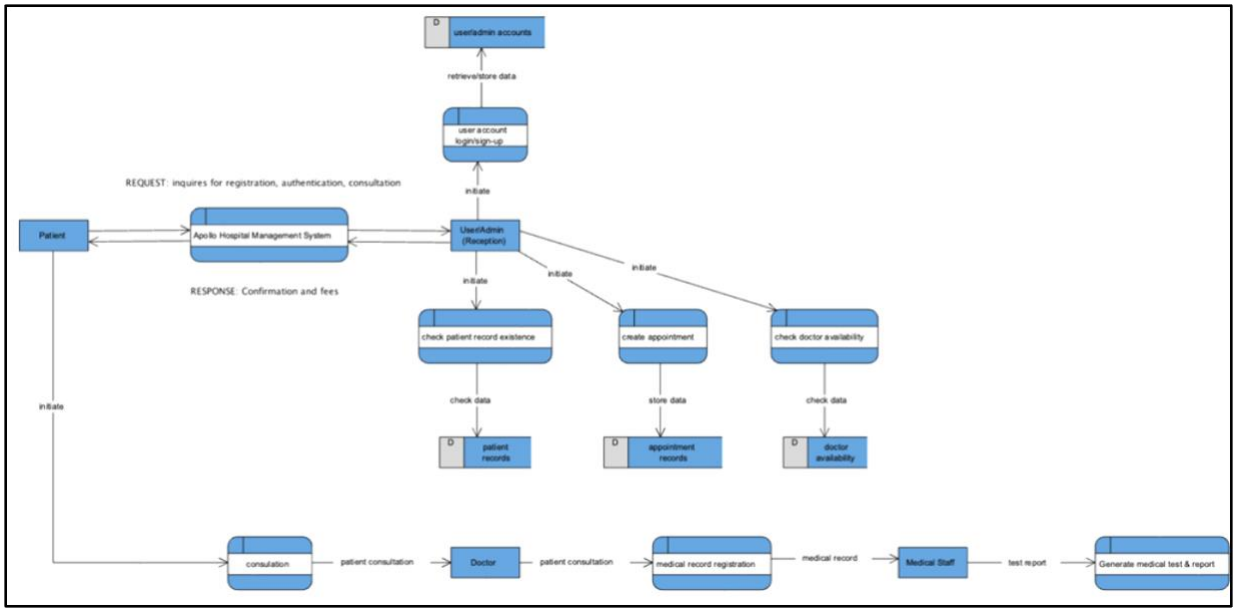


Figure 17: Level 1, data flow diagram of Apollo Hospital

Patient registration, authentication, and consultation remain unchanged. Receptionists now have access to an accounts database containing information for doctors, staff, and receptionists. They can check patient records, create appointments, and verify doctor availability. After the patient consults with the doctor, a medical record is created, sent to medical staff for report generation, including medical and test reports.

- *LEVEL 2*

Below is the level 0 of the data flow diagram which showcases more and detailed processes of how the proposed system operates.

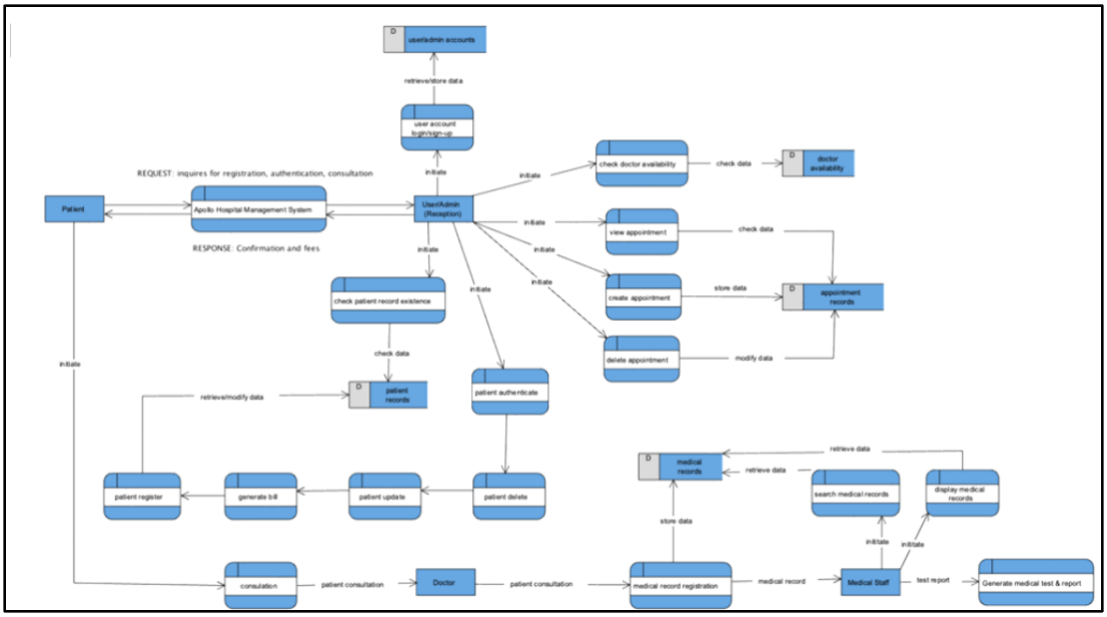


Figure 18: Level 2, data flow diagram of Apollo Hospital

Patient registration, authentication, and consultation processes remain consistent from the previous system. Receptionists have added functions like appointment deletion and viewing, and patient detail retrieval and insertion. Medical staff now include processes like searching and displaying medical records, accessing a medical record database. Despite these additions, the overall process flow between entities remains unchanged. Level 2 provides a detailed diagram illustrating data interactions between entities.

B. USE-CASE DIAGRAM

The use-case diagram illustrates how entities interact in the proposed system, with stick figures representing entities and circles indicating general processes. Notably, the Receptionist has the most processes, interacting with patient records, appointment records, and doctor availability records.

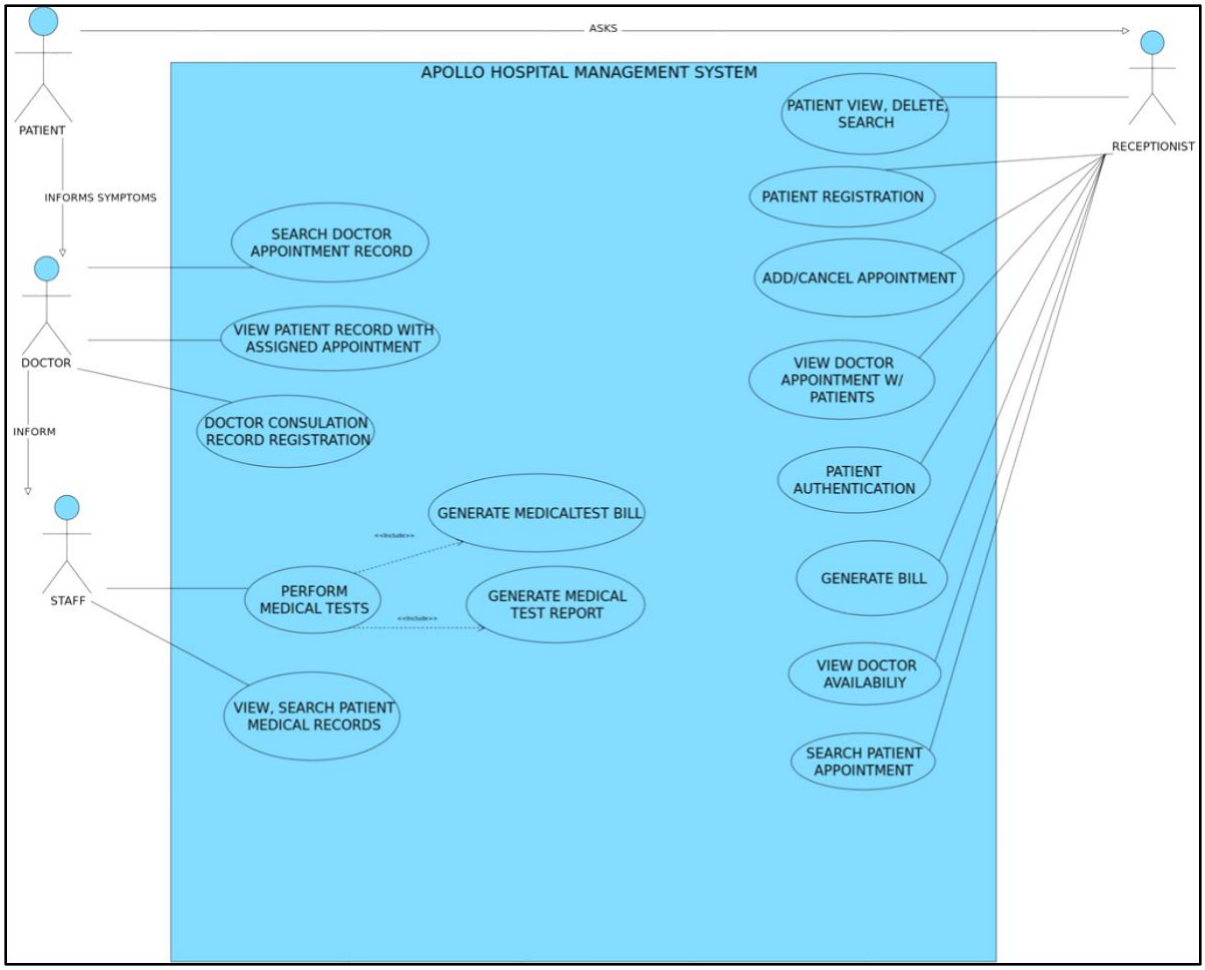


Figure 19: Use-case diagram of proposed system

This, along with the previously shown data flow diagram, summarizes the day-to-day workflow of all entities and processes. It aids in the project's design and construction from the ground up.

C. ENTITY RELATIONSHIP DIAGRAM

The ER diagram is crucial for hospital database design. It outlines entities (boxed) with attributes (circled), illustrating relationships (diamond) like Appointment linking patient and doctor. Unique IDs for each entity prevent record duplication and ensure efficient data management, avoiding anomalies.

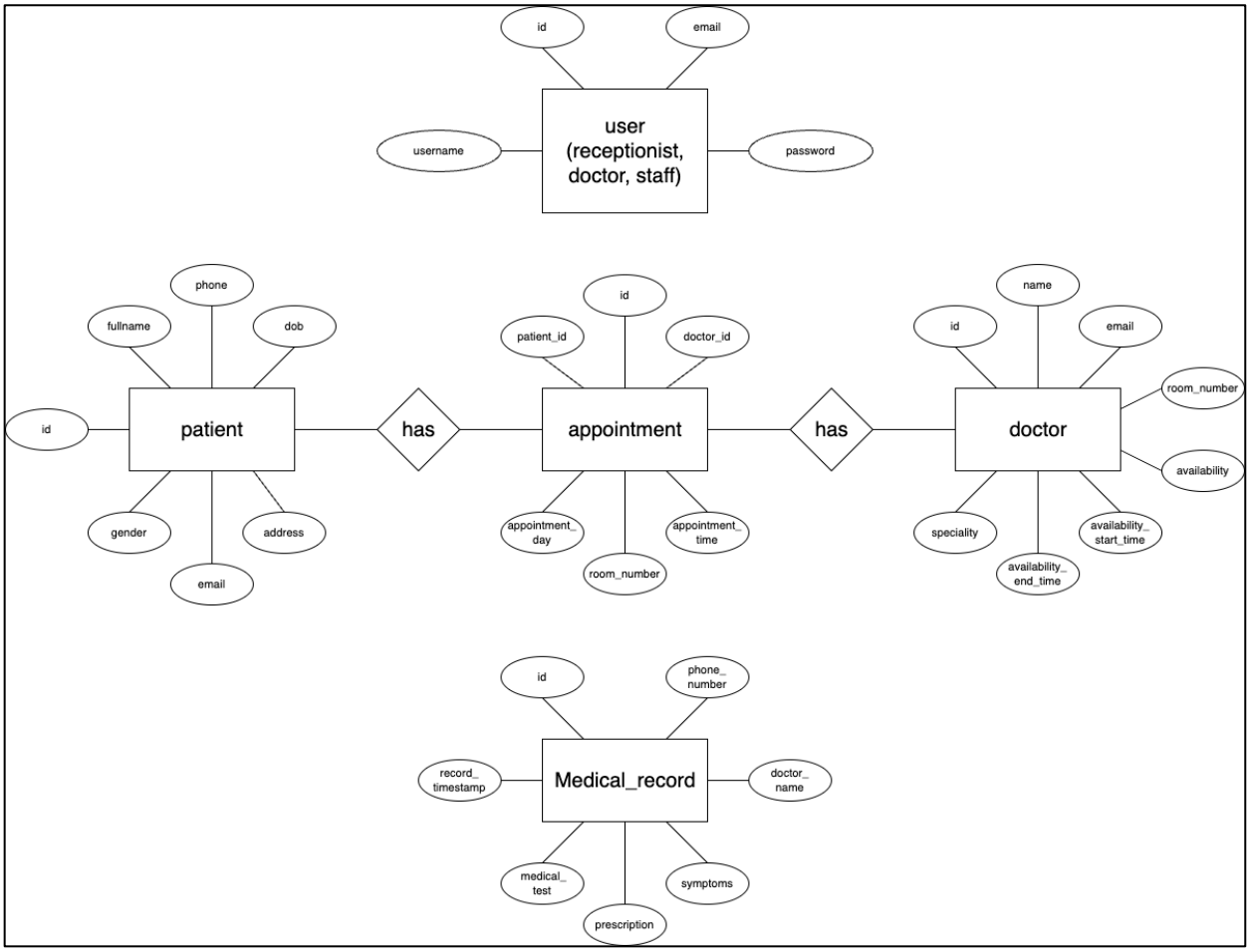


Figure 20: Entity-relationship diagram of proposed system

6. GUI DESIGN

Figma, a cloud-based UI design tool, created the graphical user interface for the proposed system. These designs, currently a generalized prototype, may undergo changes during the implementation phase.

A. LOGIN PAGE

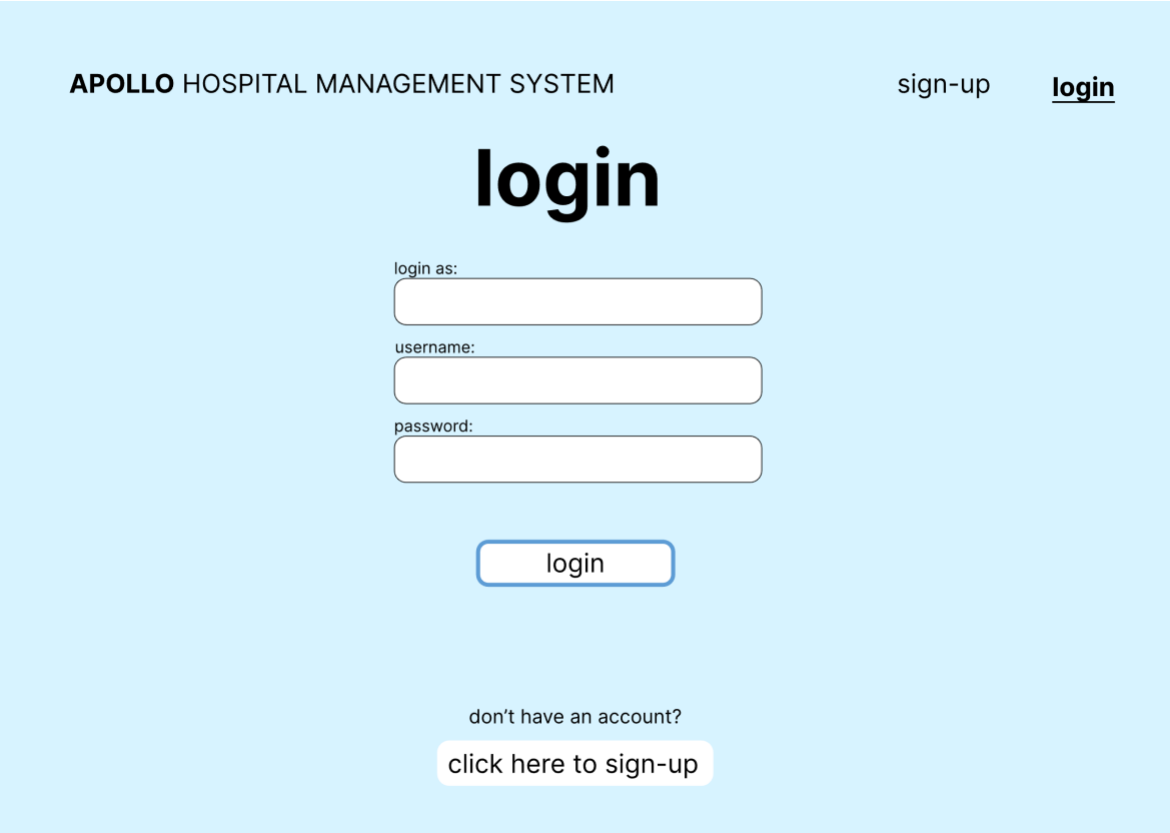


Figure 21: Prototype design of login page

The login page prototype has a minimal, flat aesthetic with a light and dark blue color scheme. The webpage includes a navigation bar with the hospital name and navigation buttons. The login section has fields for user type, username, and password, with a login button below. A signup option is available at the bottom for new users.

B. REGISTRATION PAGE

Figure 22: Prototype design of sign-up page

The signup page prototype maintains consistency with the login page design. It features a navigation bar and includes four fields for user type, username, email address, and password. A signup button is located just below the fields, and at the bottom, there's an option for users who are already registered to access their accounts.

7. IMPLEMENTATION

A. BACKEND IMPLEMENTATION

When implementing the backend of the hospital management system; Python Flask was used to implement the general backend of the inner workings of the website application, whilst XAMPP was used to implement the database side, in which data is stored.

I. XAMPP DATABASE

Below is the general database structure of the hospital management system. It showcases the attributes that are present in each table. With appointments having foreign keys connecting to both doctors and patient table.

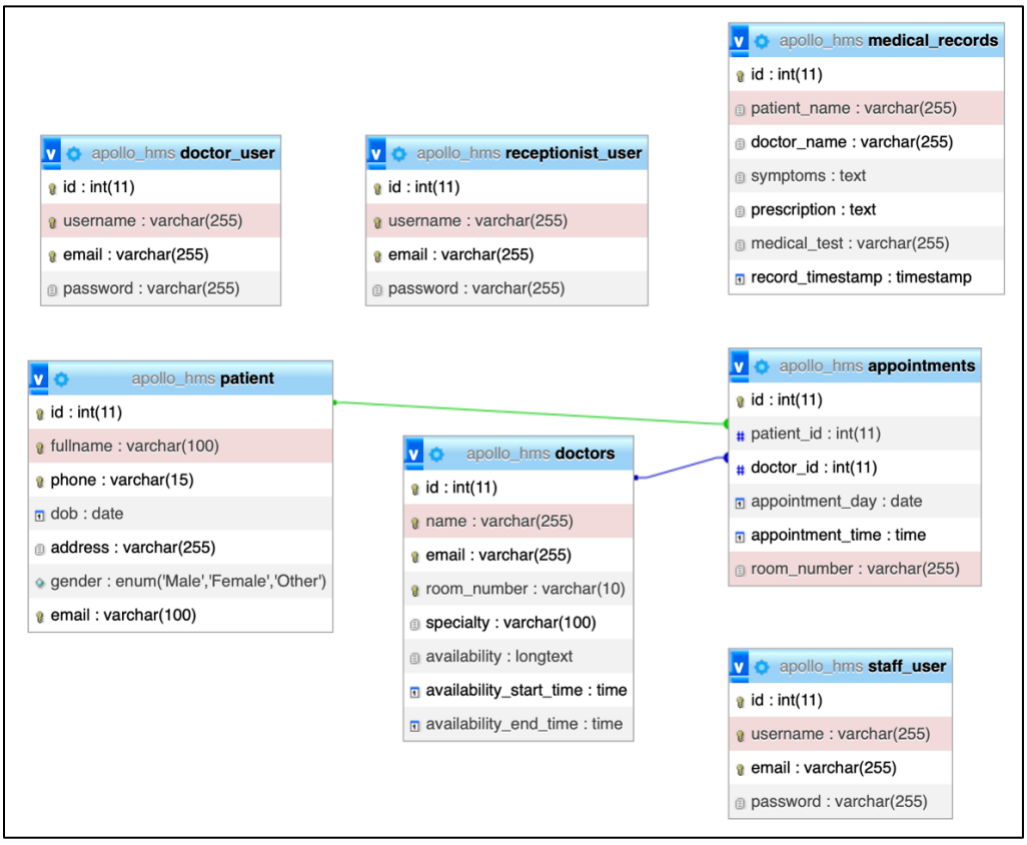


Figure 23: Database structure of proposed system

II. PYTHON FLASK

Python Flask was used to create the hospital management system's backend, enabling it to interact with the XAMPP database, such interactions are data retrieval, data entry, data manipulation, etc.

B. FRONTEND IMPLEMENTATION

The Figma prototype design was implemented using HTML and CSS to build the frontend of the proposed system. Only the frontend output will be discussed due to the length of the codes. Screenshot previews and brief explanations of each webpage's functions will be provided. Note that some webpages share identical CSS files.

I. HTML & CSS

○ *INDEX PAGE*

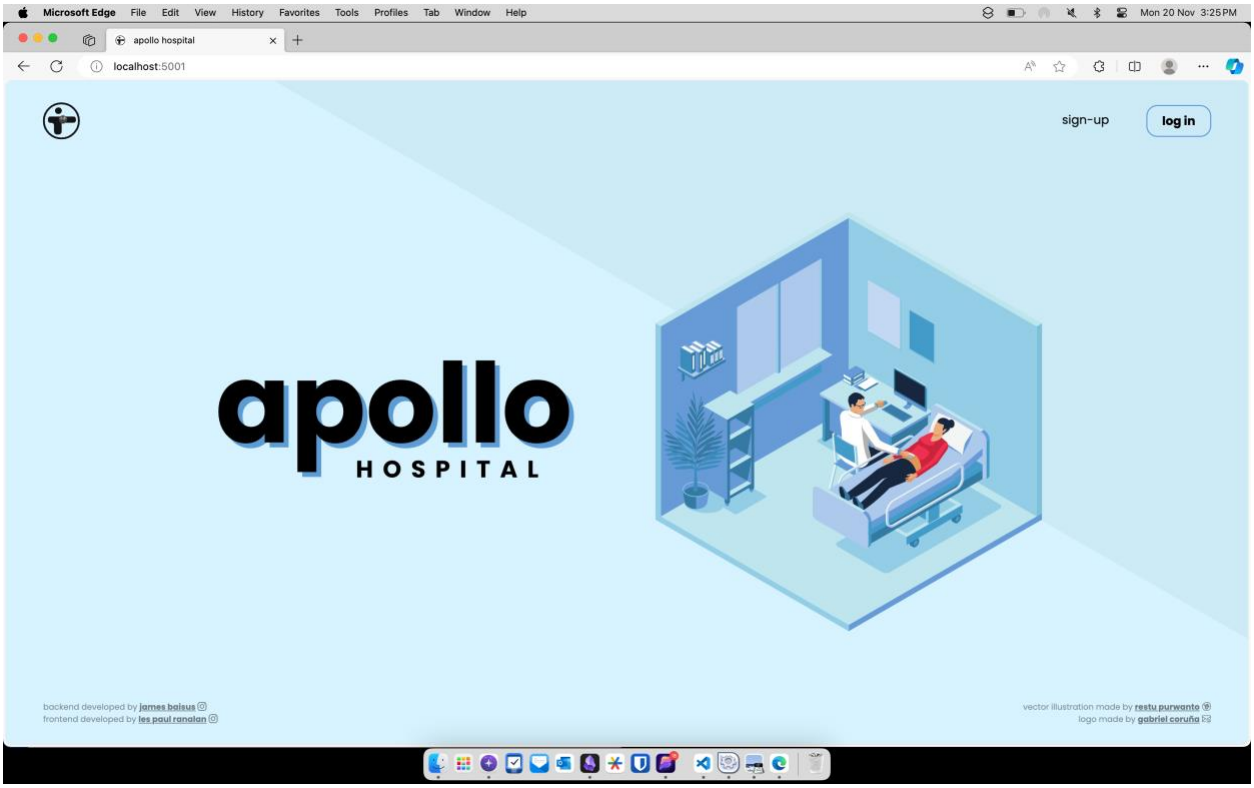


Figure 24: Index page

The index webpage features an animated display of vector art, the company name, and the navigation bar upon entry. At the page's bottom, attributions to the work are shown. Users can navigate to the login and sign-up pages using the navigation bar.

- *LOGIN & SIGN-UP PAGE*

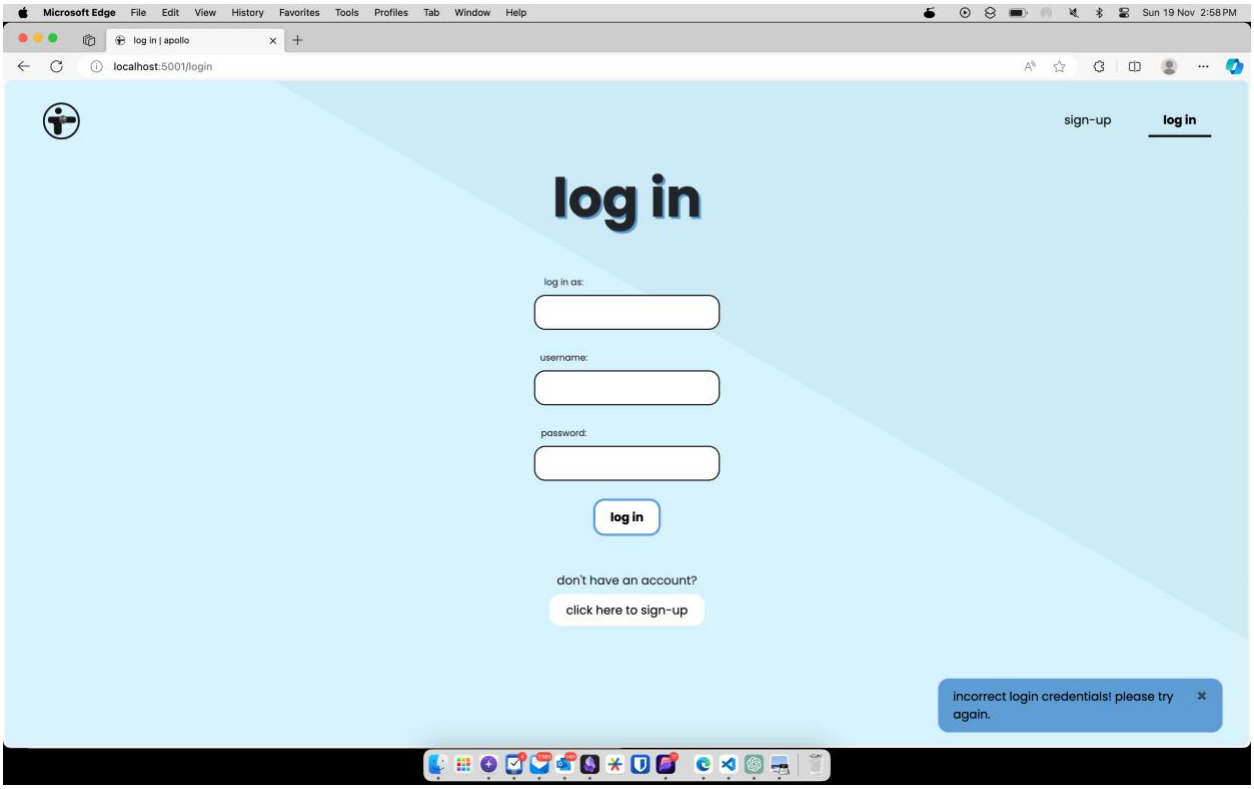


Figure 25: Login page

The login page allows users to input user type (receptionist, doctor, staff), username, and password. Successful login redirects users to their respective views; otherwise, an "incorrect login credentials" banner is displayed.

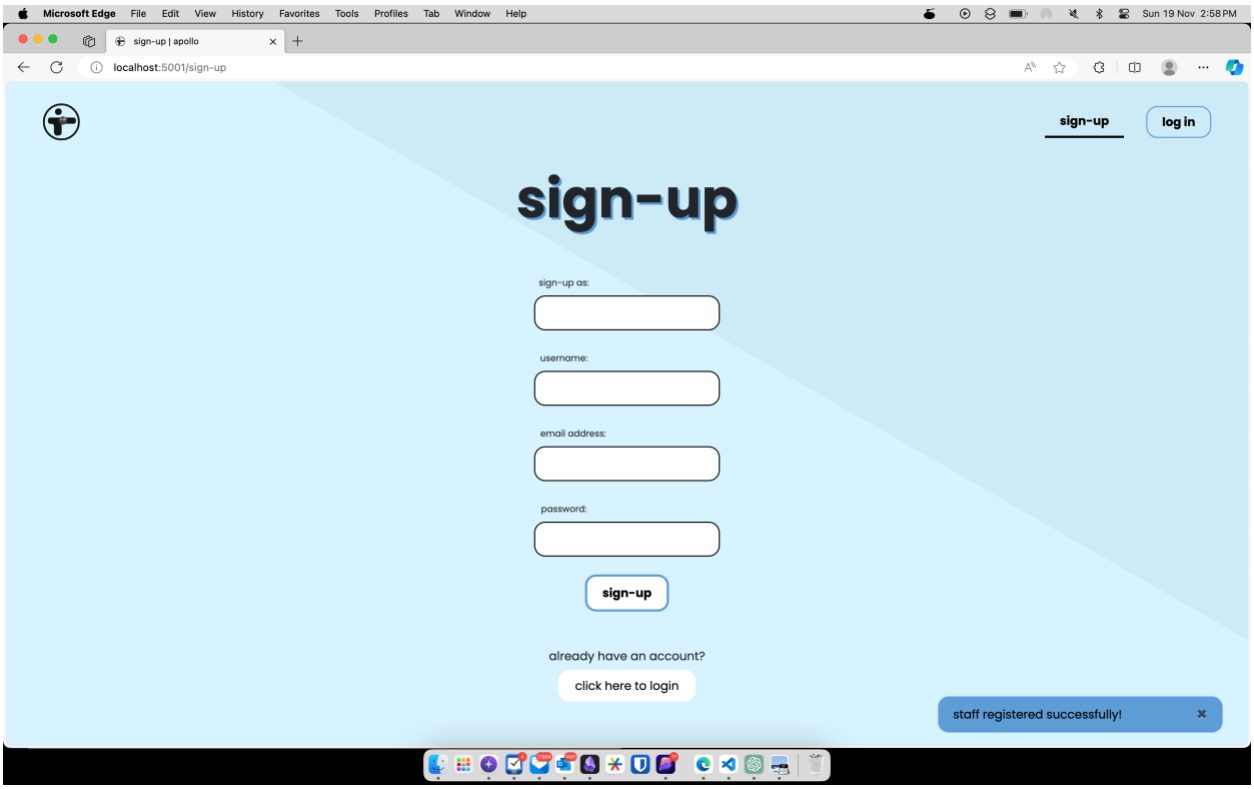


Figure 26: Sign-up page

The signup page, like the login page, requires users to input user type, username, password, and email address. Upon successful signup, a banner saying, "user registered successfully" is displayed; otherwise, it shows "invalid credentials."

○ *RECEPTIONIST VIEW*

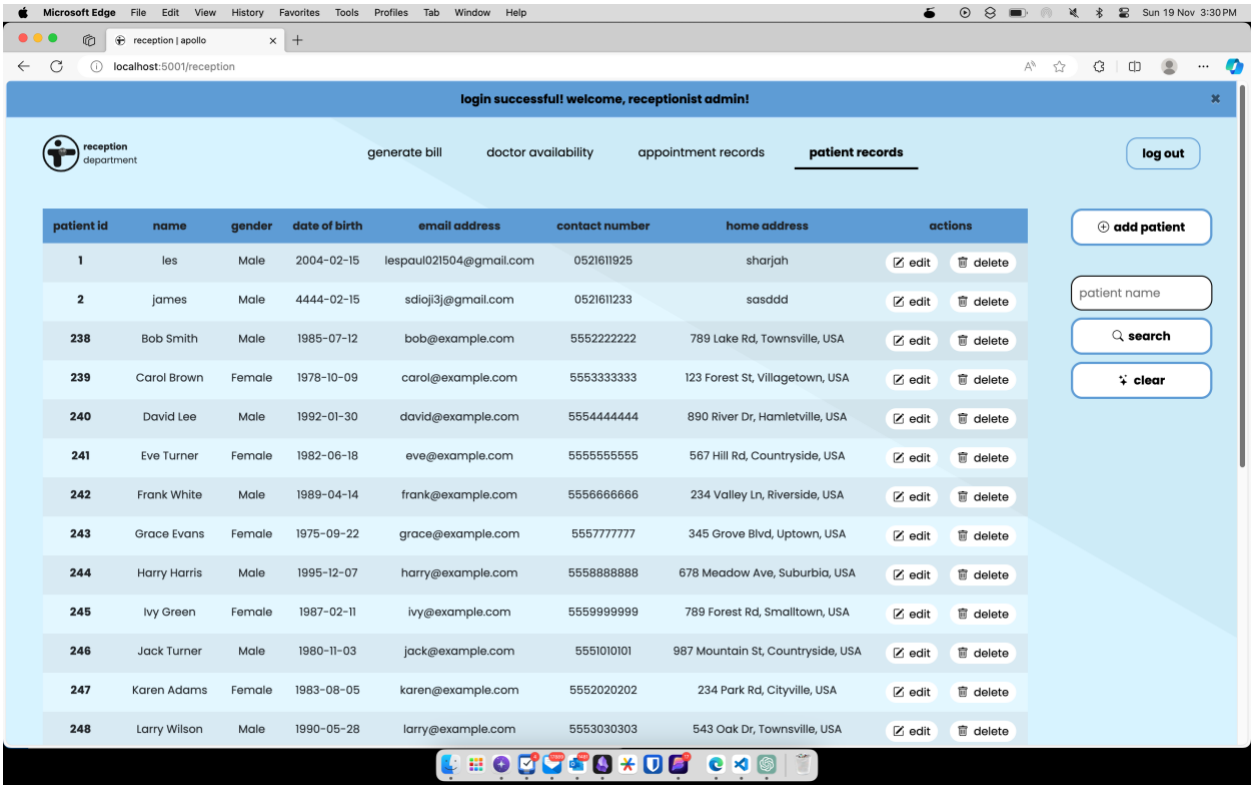


Figure 27: Receptionist view, patient records page

Upon successful login as a receptionist, a banner confirms the login and welcomes the receptionist. The navigation bar allows access to various webpages like home, generate bill, doctor availability, and more. Currently, the patient records page is selected. The page features a table with patient details, along with options for adding, searching, editing, and deleting records. Successful actions prompt a banner confirmation. This page is used to create new patient files when requested.

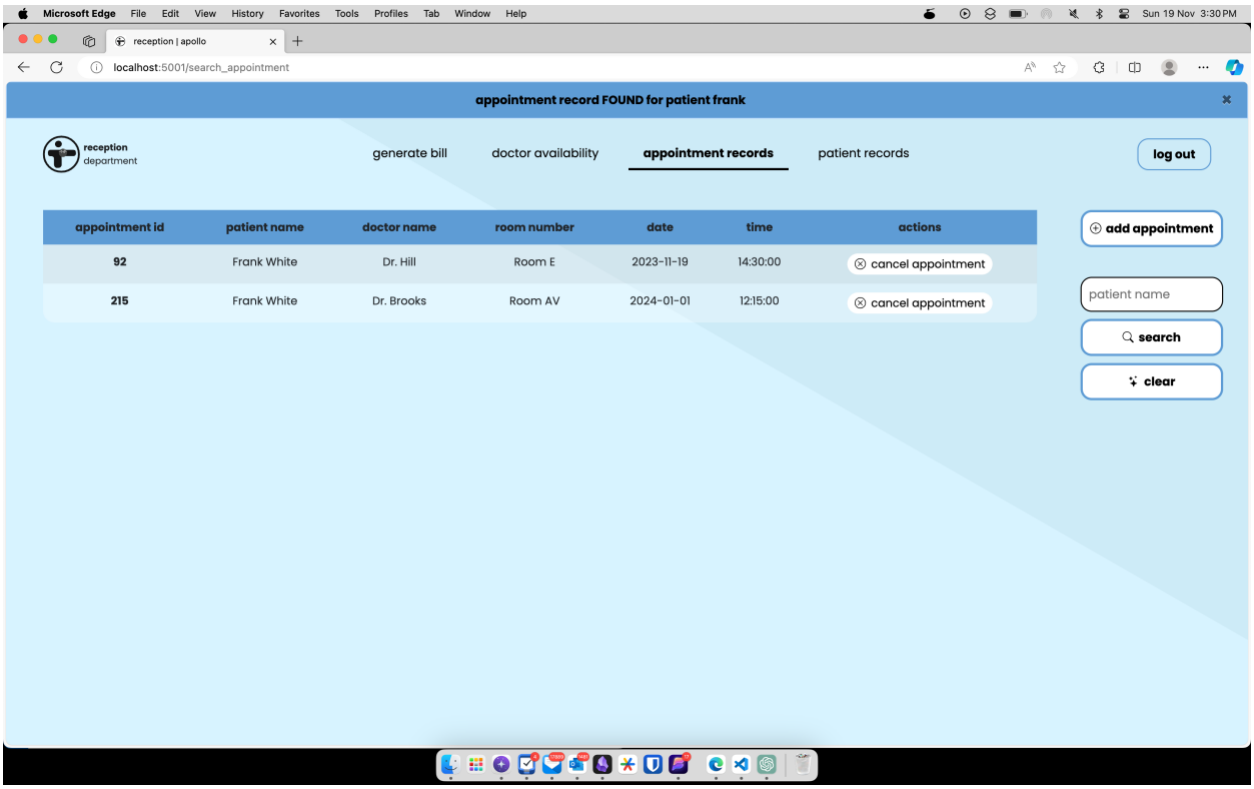


Figure 28: Receptionist view, appointment records page

The appointment records page for receptionists displays a table with crucial details. Operations include adding appointments, searching patient names, and clearing. Record actions involve canceling appointments, equivalent to removing records. Successful actions prompt banners like "appointment record added" or "appointment record found." This is the page for creating appointments when requested.

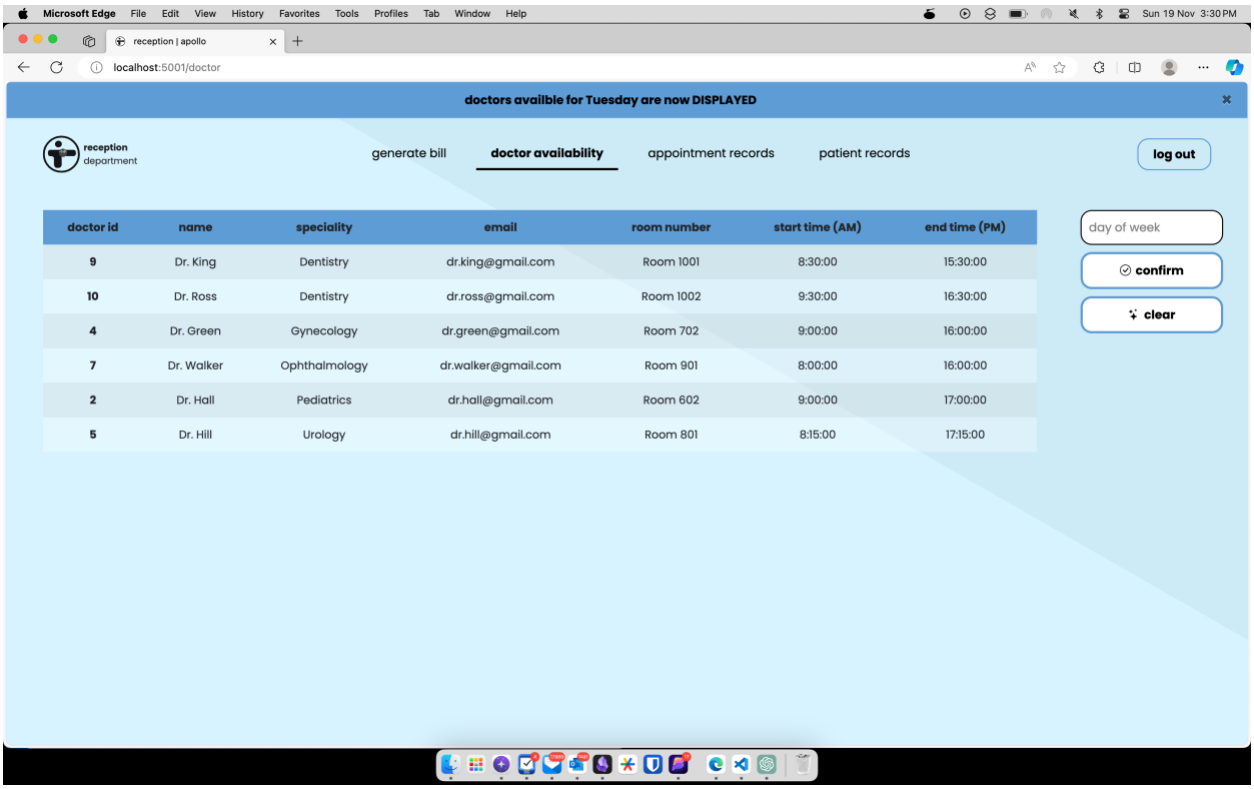


Figure 29: Receptionist view, doctor availability page

The doctor availability page for receptionists displays a table with essential details, crucial for determining doctor availability for appointment creation. Operations include a search function and clear, with no record actions. Selecting a day updates the table, accompanied by a banner saying, "doctors available for this day are now displayed." This page streamlines the process of checking doctors' availability for easier appointment scheduling.

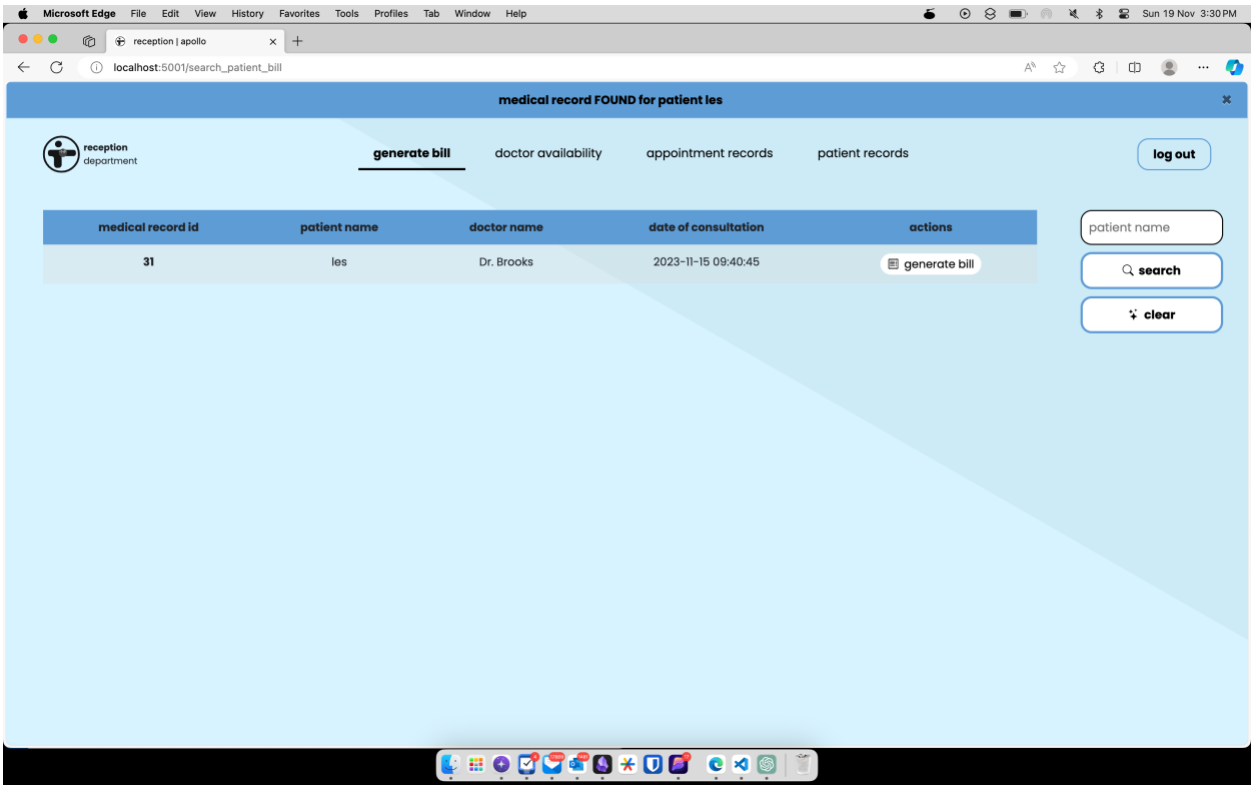


Figure 30: Receptionist view, generate bill page

The generate bill page for receptionists displays a table with medical record details. Operations include search and clear, with the specific record action being "generate bill." Successful operations prompt a banner saying, "medical record found," otherwise, it prints "medical record not found." This is the page for generating patient bills, showcasing all medical fees.

○ *DOCTOR VIEW*

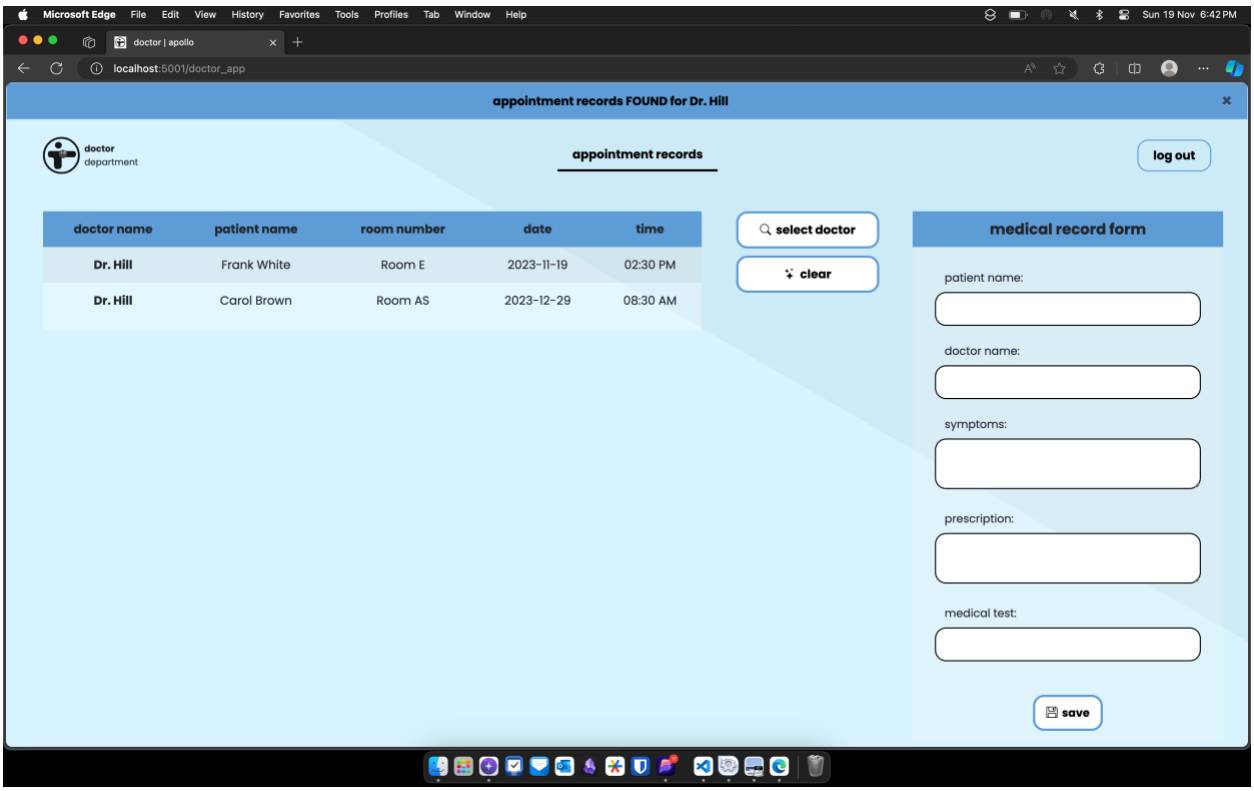


Figure 31: Doctor view, appointment records page

The doctor's appointment records page displays a medium-sized table with crucial details. No record actions are present, but there are operation buttons like "select doctor" and "clear" in the middle. On the right, a medical record form allows quick access to the left-side table for entering patient details. Banners indicate the success or failure of operations.

This page is exclusively for doctors, where they filter appointments to show only their own. It provides a quick overview of all their appointments. After consultations, doctors enter medical details like symptoms, prescriptions, and tests.

- *STAFF VIEW*

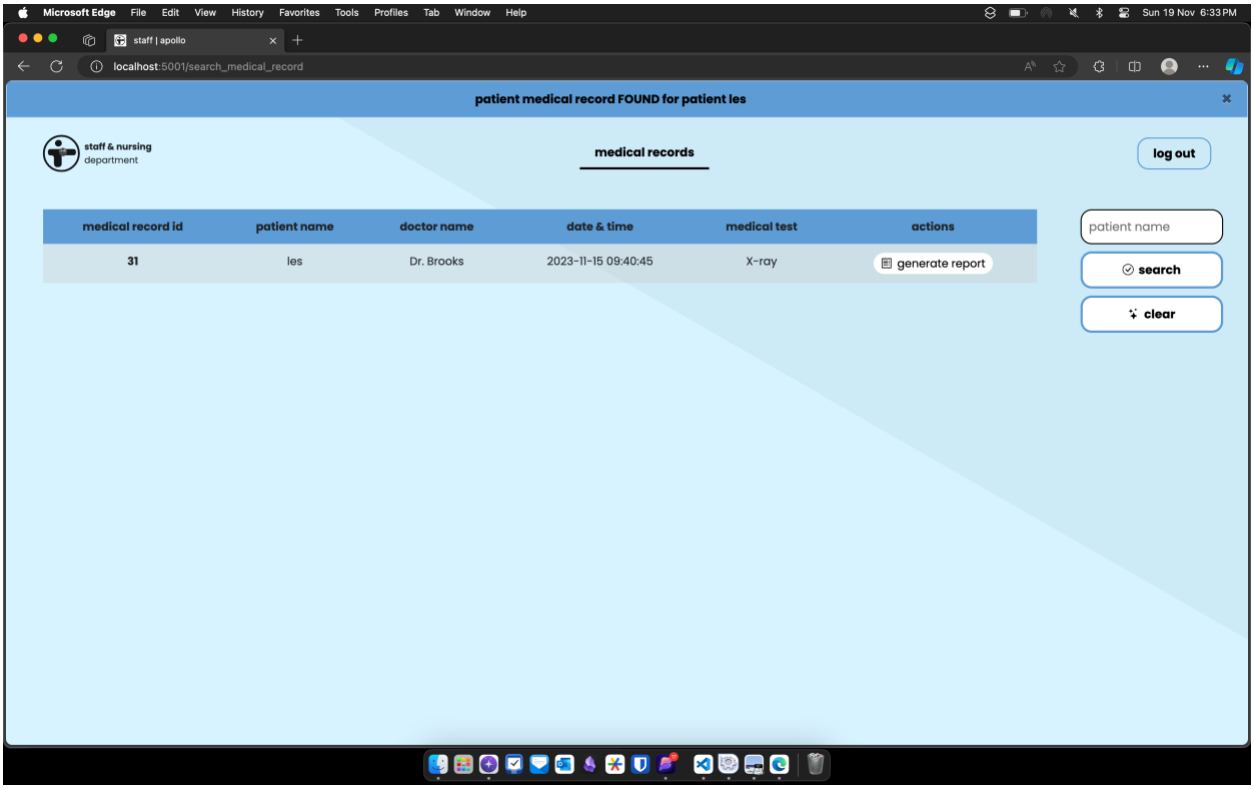


Figure 32: Staff view, medical records page

The medical records page for staff displays a large table with patient medical details. Operation buttons on the right include search and clear, with the specific record action being "generate report." Banners indicate the success or failure of operations.

This page is used by staff to generate results for patients' medical tests.

II. JAVASCRIPT

JavaScript is minimally utilized in the system, specifically on the index page. The code ensures elements load before executing the entrance animation, preventing simultaneous loading. It also restricts interaction with the navigation bar during the ongoing animation, allowing interaction only after completion.

This code is complemented by the external module "Flask Compress," which compresses webpage elements for faster loading times.

III. BOOTSTRAP

The proposed system extensively relies on Bootstrap, an external library facilitating grid manipulation and auto-adjusting divisions for frontend development. The navigation bar exemplifies Bootstrap's basic use, where predefined class names like "navbar" automatically apply styling without the need for additional CSS definitions.

8. TESTING

A. DEFINITION OF TESTING

Testing is essential to verify if the proposed system meets specified requirements. Individual units or modules undergo testing to ensure accurate functionality and identify any system errors or bugs.

B. TEST CASES

Here are detailed test cases outlining scenarios to validate system functionalities, including expected outcomes, steps for recreation, and test status (pass or failure).

I. LOGIN & SIGN-UP AUTHENTICATION

TestID	:	auth01
ModuleNo	:	auth-signup
TestName	:	signing-up as user (receptionist, staff, doctor)
Test Scenario/Steps	:	1. launch project
		2. go to sign-up page
		3. input user information
		4. click sign-up button
Test Input	:	user information
Expected Result	:	user directed to home page with alert saying "user registrered successfully"
Actual Result	:	user directed to home page with alert saying "user registrered successfully"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 33: Test auth01

TestID	:	auth02
ModuleNo	:	auth-signup
TestName	:	signing-up with invalid/incorrect details or duplicate account
Test Scenario/Steps	:	1. launch project
		2. go to sign-up page
		3. input user information
		4. click sign-up button
Test Input	:	invalid user information
Expected Result	:	show alert saying "credentials already taken or invalid! please try again"
Actual Result	:	show alert saying "credentials already taken or invalid! please try again"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 34: Test auth02

TestID	:	auth03
ModuleNo	:	auth-login
TestName	:	logging in with incorrect/invalid details
Test Scenario/Steps	:	1. launch project
		2. go to login page
		3. input user information
		4. click login button
Test Input	:	incorrect/invalid credentials
Expected Result	:	show alert saying "incorrect login credentials. please try again"
Actual Result	:	show alert saying "incorrect login credentials. please try again"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 35: Test auth03

TestID	:	auth04
ModuleNo	:	auth-login
TestName	:	logging in as user (receptionist, staff, doctor)
Test Scenario/Steps	:	1. launch project
		2. go to login page
		3. input credentials
		4. click log in button
Test Input	:	user credentials
Expected Result	:	user directed to specific user page with alert saying "login successful! Welcome, user!"
Actual Result	:	user directed to specific user page with alert saying "login successful! Welcome, user!"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 36: Test auth04

II. PATIENT RECORDS

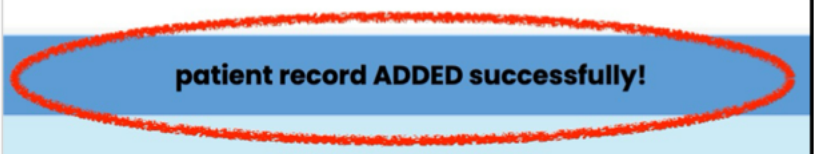
TestID	:	p_record01
ModuleNo	:	p_record-actions
TestName	:	adding patient record
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click add record button
		4. enter patient details
		5. click save button
Test Input	:	patient details
Expected Result	:	patient record is added and show alert saying, "patient record added successfully"
Actual Result	:	patient record is added and show alert saying, "patient record added successfully"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 37: Test p_record01

TestID	:	p_record02
ModuleNo	:	p_record-actions
TestName	:	adding duplicate entry, error handling
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click add record
		4. input patient details
		5. click save button
Test Input	:	patient details that already exist
Expected Result	:	show alert saying "invalid pateint: duplicate entry"
Actual Result	:	show alert saying "invalid pateint: duplicate entry"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 38: Test p_record02

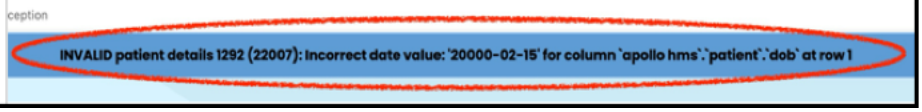
TestID	:	p_record03
ModuleNo	:	p_record-actions
TestName	:	entering invalid details, error handling
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click "add reocrd" button
		4. enter patient details
		5. click save
Test Input	:	invalid birth date like 02/15/200000
Expected Result	:	show alert saying, "invalid details, incorrect date value"
Actual Result	:	show alert saying, "invalid details, incorrect date value"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 39: Test p_record03

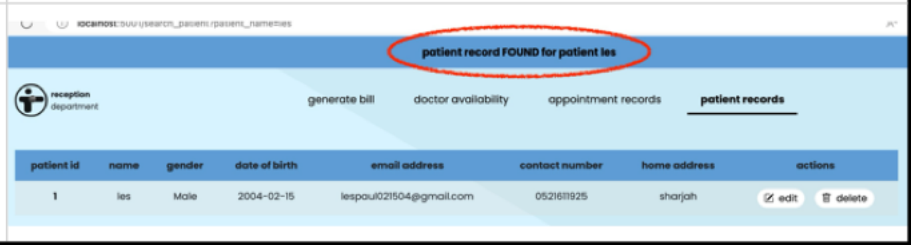
TestID	:	p_record04
ModuleNo	:	p_record-actions
TestName	:	searching existing entry
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. enter patient name in search field
		4. click search button
Test Input	:	les
Expected Result	:	searched patient record is displayed and show alert saying "patient record found"
Actual Result	:	searched patient record is displayed and show alert saying "patient record found"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 40: Test p_record04

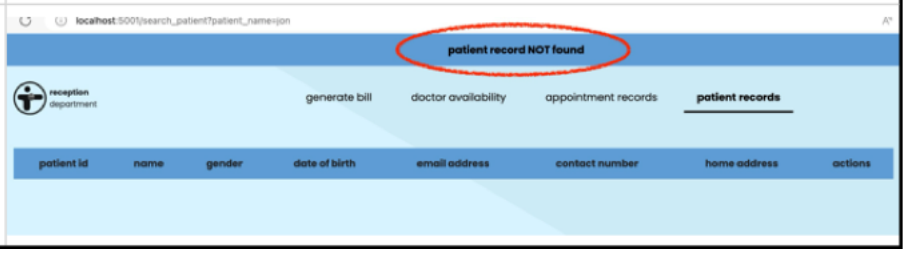
TestID	:	p_record05
ModuleNo	:	p_record-actions
TestName	:	searching non-existent entry
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. enter patient name in search field
		4. click search button
Test Input	:	jon
Expected Result	:	nothng is displayed and show alert saying, "patient record not found"
Actual Result	:	nothng is displayed and show alert saying, "patient record not found"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 41: Test p_record05

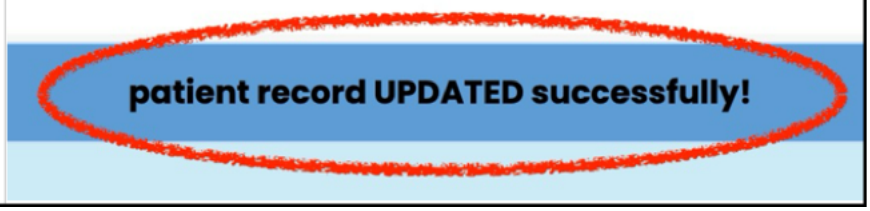
TestID	:	p_record06
ModuleNo	:	p_record-actions
TestName	:	editing entry
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click edit button of specific entry
		4. enter modified patient details
		5. click update
Test Input	:	updated name: bill wilkins
Expected Result	:	patient record is updated and show alert saying, "patient record updated successfully"
Actual Result	:	patient record is updated and show alert saying, "patient record updated successfully"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 42: Test p_record06

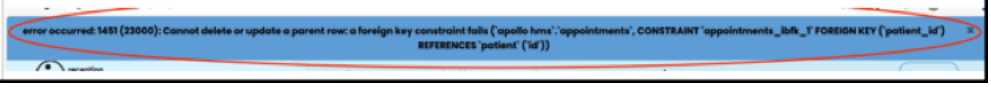
TestID	:	p_record07
ModuleNo	:	p_record-actions
TestName	:	deleting entry with existing appointment, error handling
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click delete button of specific entry
		4. click yes when warning is prompted
Test Input	:	deleting patient bill wilkins entry
Expected Result	:	show alert saying, "error: unable to delete, existing appointment record"
Actual Result	:	show alert saying, "error: unable to delete, existing appointment record"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 43: Test p_record07

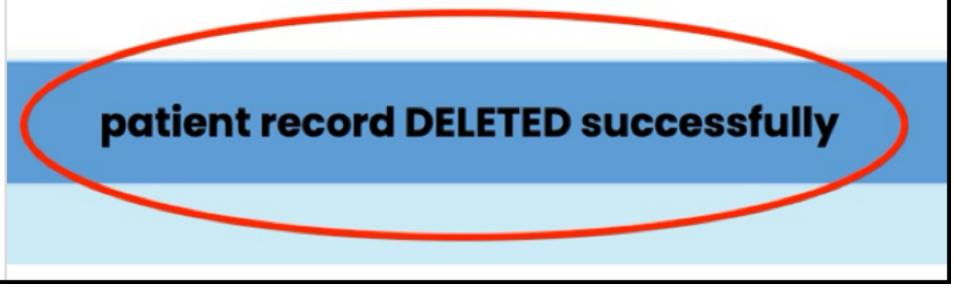
TestID	:	p_record08
ModuleNo	:	p_record-actions
TestName	:	deleting entry with non-existent appointment
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click delete button of specific entry
		4. click yes when warning is prompted
Test Input	:	deleting patient bill wilkins entry
Expected Result	:	delete the specified patient record and show alert saying, "patient record deleted successfully"
Actual Result	:	delete the specified patient record and show alert saying, "patient record deleted successfully"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 44: Test p_record08

III. APPOINTMENT RECORDS

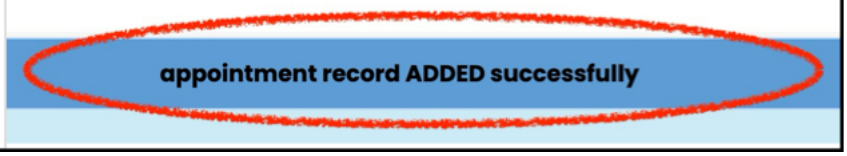
TestID	:	a_record01
ModuleNo	:	a_record-actions
TestName	:	adding appointment
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click appointment records
		4. click add appointment button
		5. enter appointment details
		6. click save
Test Input	:	appointment for carol brown
Expected Result	:	appointment record is added and show alert saying, "appointment record added successfully"
Actual Result	:	appointment record is added and show alert saying, "appointment record added successfully"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 45: Test a_record01

TestID	:	a_record02
ModuleNo	:	a_record-actions
TestName	:	adding appointment with invalid details, error handling
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click appointment records
		4. click add appointment button
		5. enter appointment details
		6. click save
Test Input	:	invalid birth date like 02/12/200000
Expected Result	:	show alert saying, "error: invalid value for date"
Actual Result	:	show alert saying, "error: invalid value for date"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 46: Test a_record02

TestID	:	a_record03
ModuleNo	:	a_record-actions
TestName	:	adding duplicate appointment, error handling
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click appointment records
		4. click add appointment button
		5. enter appointment details
		6. click save
Test Input	:	duplicate appointment for frank white and dr. hill
Expected Result	:	show alert saying, "appointment slot already taken"
Actual Result	:	show alert saying, "appointment slot already taken"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 47: Test a_record03

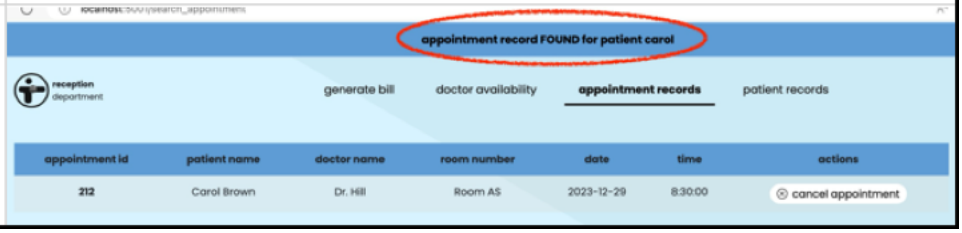
TestID	:	a_record04
ModuleNo	:	a_record-actions
TestName	:	searching existing appointment
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click appointment records
		4. enter name of patient in search fied
		5. click search
Test Input	:	carol brown
Expected Result	:	searched appointment record is displayed and show alert saying "appointment record found"
Actual Result	:	searched appointment record is displayed and show alert saying "appointment record found"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 48: Test a_record04

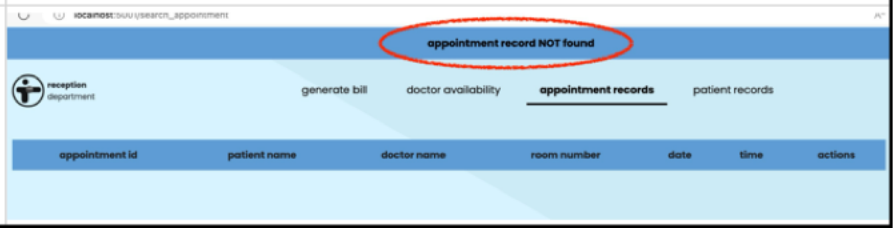
TestID	:	a_record05
ModuleNo	:	a_record-actions
TestName	:	searching non-existent appointment
Test Scenario/Steps	:	1. launch project
		2. login as receptionist
		3. click appointment records
		4. enter name of patient in search fied
		5. click search
Test Input	:	jon
Expected Result	:	displays nothing and show alert saying "appointment record not found"
Actual Result	:	displays nothing and show alert saying "appointment record not found"
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 49: Test a_record05

IV. DOCTOR AVAILABILITY

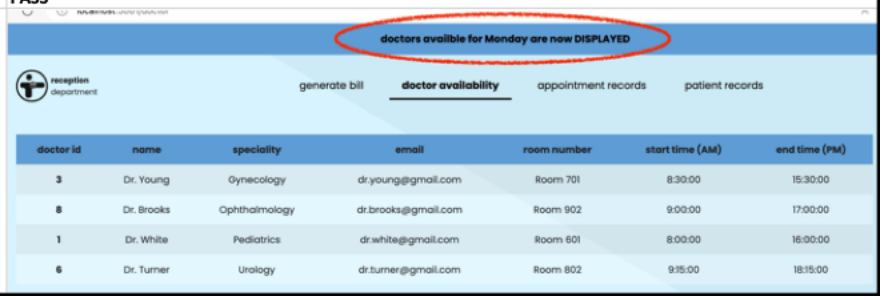
TestID	:	Doc01
ModuleNo	:	Doctor Availability
TestName	:	Filtering doctor availabilities
Test Scenario/Steps	:	1. Select day to view doctor availablity
		2. display doctors that are available in a curtain day
		3. Notification pop up for success status
Test Input	:	Sunday as the day
Expected Result	:	Doctors available on sunday
Actual Result	:	List of doctros availability on sunday
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 50: Test Doc01

V. GENERATE MEDICAL BILL

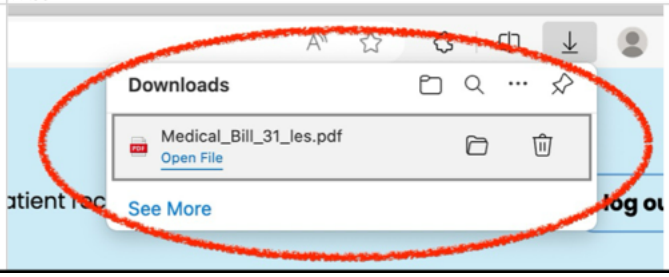
TestID	:	GenerateBill01
ModuleNo	:	Generate Bill
TestName	:	Generate bill for a specific patient
Test Scenario/Steps	:	1. Search the patient medical record
		2. click to generate patient bill
		3. download file should be available for printing
Test Input	:	generate bill for Harry Hilton
Expected Result	:	PDF file containing the bill template
Actual Result	:	PDF file for patient Harry
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 51: Test GenerateBill01

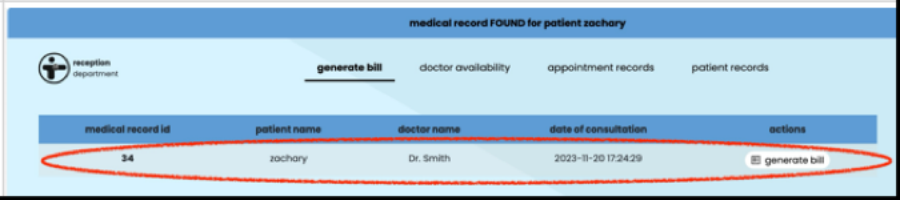
TestID	:	Search001
ModuleNo	:	Search patient bill
TestName	:	Search patient to acces their medical bill
Test Scenario/Steps	:	1. use the search bar provided
		2. input a patient name
		3. display the patient consultation with the doctor record
Test Input	:	searching Patientt zachary
Expected Result	:	Patient zachary biill
Actual Result	:	Patient zachary with button to genereate bill
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 52: Test Search001

VI. GENERATE MEDICAL REPORT

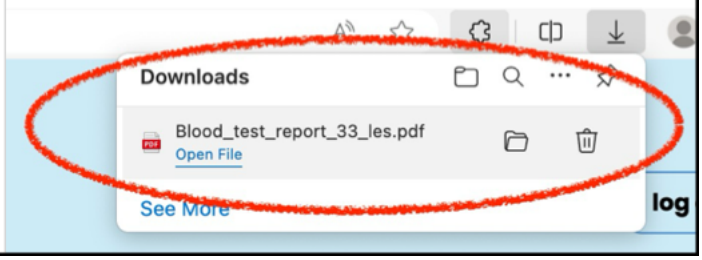
TestID	:	Staff01
ModuleNo	:	Generate Medical report
TestName	:	searching and generating medical report
Test Scenario/Steps	:	1. search patient consultation with the doctor record
		2. click on the patient record to pull the modal preview
		3. download the PDF Medical test template
Test Input	:	Generating medical report for Harry Hilton
Expected Result	:	PDF file medical report for the patient
Actual Result	:	PDF File
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 53: Test Staff01

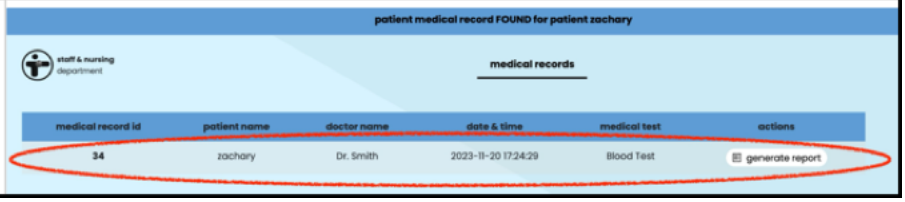
TestID	:	Serch002
ModuleNo	:	search medical template
TestName	:	Search patient medical test record
Test Scenario/Steps	:	1. use the search bar provided
		2. input a patient name
		3. display the patient consultation with the doctor with medical report record
Test Input	:	searching Patient zachary
Expected Result	:	Patientt zachary medical test report
Actual Result	:	Patientt zachary with buttion to generate medical report template
Pass/Fail	:	**PASS**
Screenshot	:	

Figure 54: Test Search002

C. TEST TABLE

Here is the comprehensive test table summarizing all test cases, including details such as test name, input, results, and status (pass or fail).

Test Case ID	Test Name	Test Input	Expected Result	Actual Result	Pass/Fail
auth01	signing-up as user (receptionist, staff, doctor)	user information	user directed to home page with alert saying "user registrered successfully"	user directed to home page with alert saying "user registrered successfully"	PASS
auth02	signing-up with invalid/incorrect details or duplicate account	invalid user information	show alert saying "credentials already taken or invalid! please try again"	show alert saying "credentials already taken or invalid! please try again"	PASS
auth03	logging in with incorrect/invalid details	incorrect/invalid credentials	show alert saying "incorrect login credentials. please try again"	show alert saying "incorrect login credentials. please try again"	PASS
auth04	logging in as user (receptionist, staff, doctor)	user credentials	user directed to specific user page with alert saying "login successful! Welcome, user!"	user directed to specific user page with alert saying "login successful! Welcome, user!"	PASS
p_record01	adding patient details	patient details	patient record is added and show alert saying, "patient record added successfully"	patient record is added and show alert saying, "patient record added successfully"	PASS
p_record02	adding duplicate entry, error handling	patient details that already exists	show alert saying "invalid pateint: duplicate entry"	show alert saying "invalid pateint: duplicate entry"	PASS
p_record03	entering invalid details, error handling	invalid birth date (i.e.: 02/15/20000)	show alert saying, "invalid details, incorrect date value"	show alert saying, "invalid details, incorrect date value"	PASS
p_record04	searching existing entry	"les"	searched patient record is displayed and show alert saying "patient record found"	searched patient record is displayed and show alert saying "patient record found"	PASS
p_record05	searching non-existent entry	"jon"	nothng is displayed and show alert saying, "patient record not found"	nothng is displayed and show alert saying, "patient record not found"	PASS
p_record06	editing entry	updated name: "bill wilkins"	patient record is updated and show alert saying, "patient record updated successfully"	patient record is updated and show alert saying, "patient record updated successfully"	PASS
p_record07	deleting entry with existing appointment, error handling	deleting patient bill wilkins entry	show alert saying, "error: unable to delete, existing appointment record"	show alert saying, "error: unable to delete, existing appointment record"	PASS
p_record08	deleting entry with non-existent appointment	deleting patient bill wilkins entry	delete the specified patient record and show alert saying, "patient record deleted successfully"	delete the specified patient record and show alert saying, "patient record deleted successfully"	PASS
a_record01	adding appointments	appointment for "carol brown"	appointment record is added and show alert saying, "appointment record added successfully"	appointment record is added and show alert saying, "appointment record added successfully"	PASS
a_record02	adding appointment with invalid details, error handling	invalid birth date (i.e.: 02/15/20000)	show alert saying, "error: invalid value for date"	show alert saying, "error: invalid value for date"	PASS
a_record03	adding duplicate entry, error handling	duplicate appointment for "frank white" and "dr. hill"	show alert saying, "appointment slot already taken"	show alert saying, "appointment slot already taken"	PASS
a_record04	searching existing appointment	"carol brown"	searched appointment record is displayed and show alert saying "appointment record found"	searched appointment record is displayed and show alert saying "appointment record found"	PASS
a_record05	searching non-existent appointment	"jon"	displays nothing and show alert saying "appointment record not found"	displays nothing and show alert saying "appointment record not found"	PASS
Doc01	filtering doctor availabilities	"Sunday" as day	Doctors available on sunday	List of doctros availability on sunday	PASS
GenerateBill01	generate bill for specific patient	generate bill for "harry hilton"	PDF file containing the bill template	PDF file for patient Harry	PASS
Search001	Search patient to access their bill	searching patient "zachary"	Patient zachary biill	Patient zachary with button to genereate bill	PASS
Staff01	searching and generating medical report	generateing medical report for "harry hilton"	PDF file medical report for the patient	PDF file	PASS
Search002	search patient medical test record	searching patient "zachary"	Patient zachary medical test report	Patientt zachary with buttion to generate medical report template	PASS

Figure 55: Test table

9. PROJECT MANAGEMENT TECHNIQUES

Project management involves essential methodologies and tools to plan, execute, monitor, and control projects effectively. Various techniques exist for this purpose, and below are some of the most common ones.

A. PROJECT PLANNING

It entails defining project scope, objectives, timelines, and required resources for successful execution. This phase establishes the foundation for the project life cycle, providing a roadmap guiding teams through crucial stages.

Figure 56: Project planning cycle

B. GANTTCHART

It's a widely used visual tool in project management for scheduling and tracking tasks. This chart provides a clear and intuitive understanding of project scheduling, task dependencies, and overall progress, enabling efficient resource allocation.

Note: the Gantt chart for the project has can be found in the previous chapters.

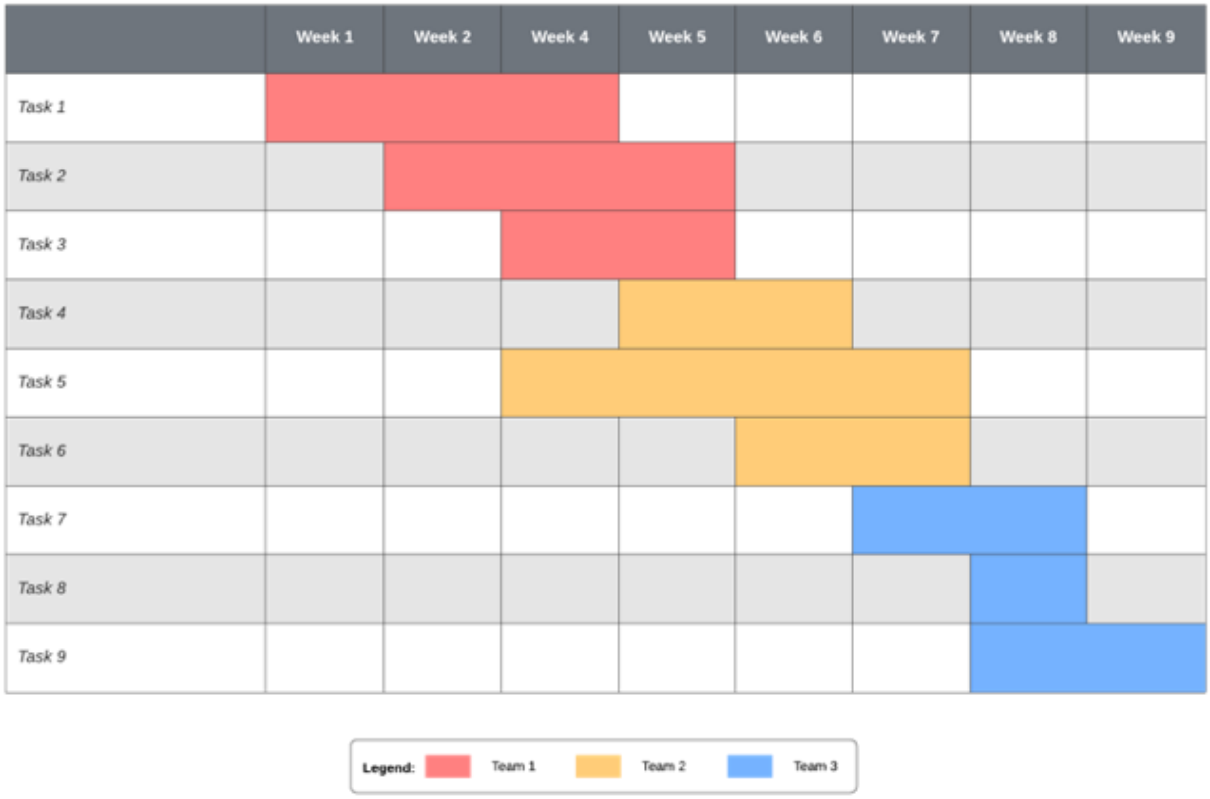


Figure 57: Gantt chart template

C. NETWORK DIAGRAM

It visualizes relationships and dependencies among project tasks, showing the project's workflow and interconnected activities. Identifying the critical path and understanding task dependencies helps teams optimize resource allocation and manage project objectives more effectively.

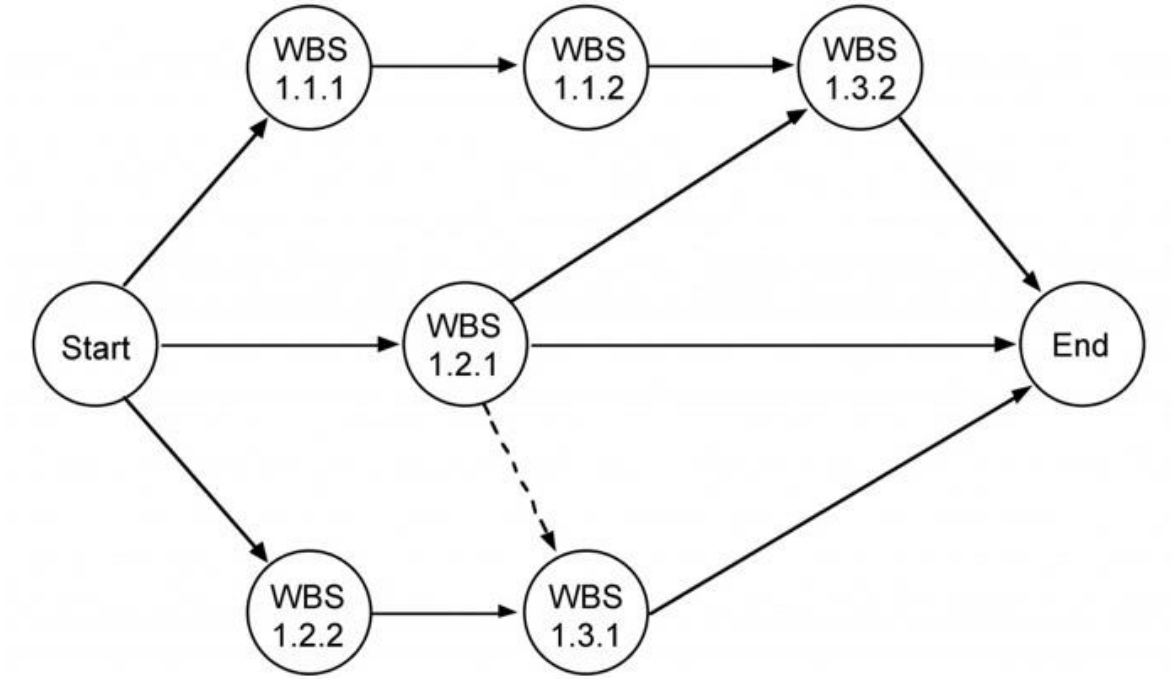


Figure 58: Example of a network diagram

D. PERT CHART

It's known as PERT, standing for "Program Evaluation and Review Technique." This chart visualizes task durations and dependencies in a project using arrows and circles to illustrate the flow of work. PERT helps teams track project timelines, ensuring a smooth progression of tasks.

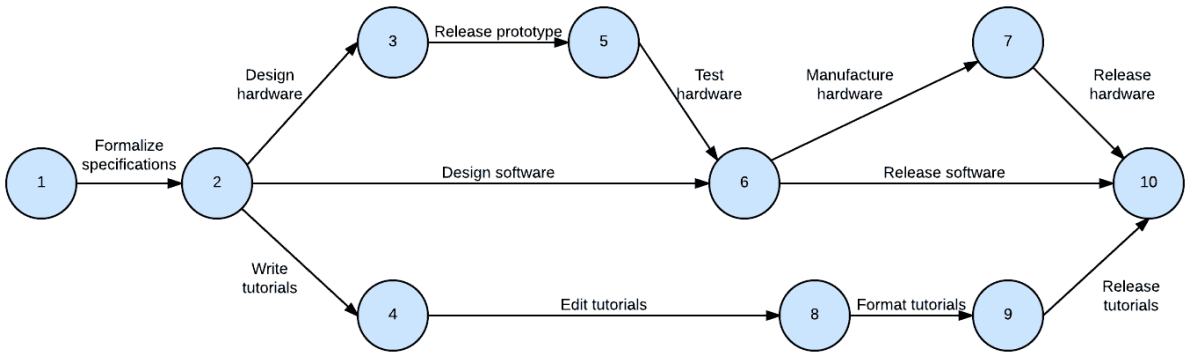


Figure 59: Example of a PERT chart

E. RASIC

It's called RASIC, short for "Responsible, Accountable, Supportive, Informed, and Consulted." This tool clarifies roles and responsibilities within a project team, ensuring every member understands their contribution. RASIC enhances accountability, minimizes confusion, and promotes efficient collaboration.

	Roles involved in managing projects					
Processes	Role 1	Role 2	Role 3	Role 4	...	Role n
Process 1	I	R	I	-		-
Process 2	A	R	S	S		C
Process 3	A	R	S	I		I
Process 4	I	S	R	S		S
...						
Process 5	A	S	R	S		I

Figure 60: RASIC chart template

10. PROJECT QUALITY ASSURANCE

A. QUALITY MANAGEMENT

It ensures the project meets set standards and expectations by planning for and continuously checking quality. This guide helps teams maintain high standards throughout the project's duration.

B. QUALITY ASSURANCE

It ensures the project uses the right processes for the desired quality, involving systematic activities to prevent problems and keep the project on the right track. This approach ensures the project moves smoothly toward its goals.

C. QUALITY CONTROL

It's where teams check project deliverables to ensure they meet required standards by inspecting and testing the work for problems. This phase ensures everything aligns with quality standards and serves as a final check before the project's presentation.

D. SOFTWARE STANDARDS

In project quality assurance, adherence to software standards is vital. These standards, like ISO, IEEE, IEC, and SPICE, provide guidelines for reliable and maintainable software, aligning with industry best practices. Following these standards ensures the software meets high-quality benchmarks.

11. CONFIGURATION MANAGEMENT

It's a crucial aspect of project development, systematically managing changes to project features and functions. This involves identifying, controlling, and documenting configuration items to ensure well-organized components are only altered when necessary. This structured approach handles modifications, tracks versions, and maintains project reliability.

12. MAINTENANCE

In project management, maintenance is crucial post-development, focusing on preserving, updating, and enhancing the project for continued effectiveness.

For the proposed Apollo Hospital Management System (AHMS), it's a flexible and adaptable system open to changes. Planned system updates include:

- **Accessibility to mobile devices**
- **Improved security with user sessions**
- **Doctors accessing medical records via patient records**
- **Addition of patient user type**
- **Enhanced web responsiveness on various operating systems**
- **Additional options for types of medical tests**
- **Bug fixes**

13. REFERENCES

- Abdelhadi Dyouri (2020). How To Make a Web Application Using Flask in Python 3. [online] Available at: https://www.digitalocean.com/community/tutorials/how-to-make-a-web-application-using-flask-in-python-3.
- GeeksforGeeks. (2018). Software Engineering Software Maintenance. [online] Available at: https://www.geeksforgeeks.org/software-engineering-software-maintenance/.
- GeeksforGeeks. (2020a). Functional vs Non Functional Requirements. [online] Available at: https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/.
- GeeksforGeeks. (2020b). Performing DataBase Operations in XAMPP. [online] Available at: https://www.geeksforgeeks.org/performing-database-operations-in-xampp/.
- GeeksforGeeks. (2020c). Software Development Life Cycle SDLC. [online] Available at: https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/
- GeeksforGeeks. (2020d). Types of Feasibility Study in Software Project Development. [online] Available at: https://www.geeksforgeeks.org/types-of-feasibility-study-in-software-project-development/.
- Guru99. (2023a). How to Write Test Cases in Software Testing with Examples. [online] Available at: https://www.guru99.com/test-case.html.

- Guru99. (2023b). Software Configuration Management in Software Engineering. [online] Available at: https://www.guru99.com/software-configuration-management-tutorial.html.
- Investopedia. (2023). Feasibility Study. [online] Available at: https://www.investopedia.com/terms/f/feasibility-study.asp.
- Jardine, J. (2022). Quality Assurance, Quality Control, and Quality Management Systems: Clarifying Confusion. [online] MasterControl. Available at: https://www.mastercontrol.com/gxp-lifeline/quality-assurance-quality-control-quality-management/.
- Javascript.info. (2023). The Modern JavaScript Tutorial. [online] Available at: https://javascript.info/.
- Learn | Hevo. (2022). Create an XAMPP MySQL Database in 6 Easy Steps. [online] Available at: https://hevodata.com/learn/xampp-mysql/.
- Lucidchart. (2023). What is a Data Flow Diagram. [online] Available at: https://www.lucidchart.com/pages/data-flow-diagram.
- Palletsprojects.com. (2023). Welcome to Flask — Flask Documentation (3.0.x). [online] Available at: https://flask.palletsprojects.com/en/3.0.x/.
- Project Central. (2023). 6 Project Management Techniques Every Project Manager Should Know. [online] Available at: https://www.projectcentral.com/blog/project-management-techniques/.
- SE 555 Software Requirements & Specification Requirements Analysis. (n.d.). Available at: https://www.se.rit.edu/~se555/Requirements%20Analysis.pdf.

- Smartsheet. (2022). The Essential Guide to Project Risk Analysis. [online] Available at: https://www.smartsheet.com/content/project-risk-analysis.
- Team Asana (2022). 6 Steps to Requirements Gathering for Project Success [2023] • Asana. [online] Asana. Available at: https://asana.com/resources/requirements-gathering.
- Teamgantt.com. (2021). How to Use a Gantt Chart Effectively | TeamGantt. [online] Available at: https://www.teamgantt.com/what-is-a-gantt-chart.
- Tomasz Bąk and SoftKraft (2016). Software Development Standards: ISO compliance and Agile. [online] Softkraft.co. Available at: https://www.softkraft.co/software-development-standards/.
- UX Design Institute. (2022). Getting started with Figma: An introduction for UI designers. [online] Available at: https://www.uxdesigninstitute.com/blog/figma-introduction-for-ui-designers/.
- W3schools.com. (2014). Bootstrap Get Started. [online] Available at: https://www.w3schools.com/bootstrap/bootstrap_get_started.asp.
- W3schools.com. (2023). HTML Styles CSS. [online] Available at: https://www.w3schools.com/html/html_css.asp.