Gentle Haloes

[Palliative Care Services]

Mini Project Report

Submitted by

Maria Jacob

Reg. No.: AJC19MCA-I037

In Partial fulfillment for the Award of the Degree of

INTEGRATED MASTER OF COMPUTER APPLICATIONS (INMCA)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2023-2024

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, "**GENTLE HALOES"** is the bona fide work of **MARIA JACOB (Regno: AJC19MCA-I037)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Mr. Binumon Joseph	**Ms. Meera Rose Mathew**
Internal Guide	**Coordinator**

Rev. Fr. Dr. Rubin Thottupurathu Jose
Head of the Department

DECLARATION

I hereby declare that the project report **“GENTLE HALOES”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Integrated Master of Computer Applications (INMCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

Date: 31/10/2023	**MARIA JACOB**
KANJIRAPPALLY	**Reg: AJC10MCA-I037**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Mr. Binumon Joseph** for his inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

MARIA JACOB

ABSTRACT

The Panchayat Level Palliative Care Website project aimed to develop a comprehensive online platform to facilitate palliative care services at the panchayat level. Palliative care focuses on improving the quality of life for individuals with serious illnesses and their families. The website serves as a platform to provide information, support and resources related to palliative care to the residence of the panchayat.

This project aims to create a dynamic and interactive website that serves as a reliable source of information, promoting awareness about palliative care services, fosters communication and collaboration among users, and facilitates access to essential palliative care services. The website will provide general information on palliative care, symptom management, treatment options, and end-of-life care. It will include features such as user registration and authentication, an information repository, a healthcare provider directory, resources and support services, appointment booking, and notifications.

Admin Module:

- **Login**: Admin access.
- **Dashboard**: Overview of the website with key metrics, appointments, tasks, and recent activities.
- **Patients Management**: Create, edit, and delete user accounts, track user activities, manage registration, and view patient details.
- **Volunteer and Health Care Assistant Management**: Add/remove volunteers and health care assistants, track profiles, and manage volunteer hours and shifts.
- **Content Management**: Handle articles, guides, videos, and FAQs in the information repository and approve user-generated content.
- **Appointment Management**: View and manage scheduled appointments, including scheduling and service management.

User Module:

- **User Registration and Login**: Account creation and login.
- **Patient Profiles**: Profiles for registered patients.
- **Medical Record**: View and update patient details, including medical history, diagnosis, and treatment.
- **Appointment Management**: Schedule appointments, with reminders and rescheduling options.
- **Palliative Care Resources**: Access educational materials, resources, and links relevant to palliative care.

Volunteer Module:

- **Volunteer Registration**: Register Asha Workers as volunteers, with specific palliative care fields.
- **Volunteer Profiles**: Manage volunteer profiles, including personal details and skills.
- **Volunteer Shift Management**: Create and manage volunteer shifts, view assigned shifts, and receive notifications.
- **View Patients Details**: Access appointment scheduling, patients' list, and their conditions.

Health Care Assistant Module:

- **Health Care Assistant Registration**: Registration for health care assistants.
- **Health Care Assistant Profiles**: Manage profiles, including personal details and availability.
- **View Patients Details**: Access patients' list and their current conditions, and update medical history and medications.
- **View Appointments for the Hospital Visit**: View appointments related to hospital visits.

CONTENT

List of Abbreviation

IDE	Integrated Development Environment
HTML	Hyper Text Markup Language.
CSS	Cascading Style Sheet
SQLite	Structured Query Language Lite
UML	Unified Modelling Language
ML	Machine Learning
JS	JavaScript
AJAX	Asynchronous JavaScript and XML Environment
RDBMS	Relational Database Management System

CHAPTER 1

INTRODUCTION

1.1PROJECT OVERVIEW

The Gentle Haloes Palliative Care Service aims to create an all-encompassing online platform that facilitates palliative care services at the panchayat level. This project focuses on improving the quality of life for individuals with serious illnesses and their families. The website will serve as a reliable source of information, providing support and resources related to palliative care to the residents of the panchayat. The platform's primary goals are to raise awareness about palliative care services, promote communication and collaboration among users, and enhance access to essential palliative care services. It will offer comprehensive information on palliative care, including user registration and authentication, appointment booking, treatment options, resources and support services, notifications, an information repository, lab facilities and end-of-life care. The project comprises five modules such as Admin, Patient, Volunteer, Health Care Assistant and Guest User. Gentle Haloes Palliative Care Service project seeks to establish an interactive and informative platform that brings palliative care services to the panchayat level, fostering better support and care for individuals facing serious illnesses and their families.

1.2 PROJECT SPECIFICATION

The "Panchayat Level Palliative Care Website" project aims to develop a dynamic and interactive online platform to facilitate palliative care services at the panchayat level, focusing on improving the quality of life for individuals with serious illnesses and their families. The website will consist of five user modules: Admin, User, Volunteer, Health Care Assistant, and Guest User.

Key features include user registration and authentication, an information repository with categorized articles and resources, a healthcare provider directory with detailed profiles, appointment booking with reminders, and a notification system. Content management will be handled by the Admin module, ensuring up-to-date resources and healthcare provider listings. The project will also specify the technology stack, data storage solutions, UI design, testing procedures, deployment plans, budget estimates, legal and compliance considerations, and post-launch maintenance and support. This project specification will serve as a comprehensive guide for the development team to create a reliable and user-friendly platform for palliative care services at the panchayat level.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The system study of Palliative Care project serves as a critical examination of the existing palliative care landscape at the panchayat level. It is an exploration into the current state of affairs, aimed at identifying key challenges and deficiencies that individuals with serious illnesses and their families encounter. This study offers valuable insights into the limitations of the current system, which have hindered effective access to palliative care services and resources.

The existing system presents various hurdles, including fragmented service delivery, a lack of centralized information, communication gaps, and difficulties in scheduling appointments. These challenges underscore the pressing need for the development of the Gentle Haloes Palliative Care Service, an innovative online platform.

The goal is to create a system that not only addresses these issues but also significantly enhances the overall palliative care experience. The proposed platform will act as a centralized hub for comprehensive information, promoting better communication and collaboration among stakeholders, streamlining appointment booking, and providing ample support services. It will prioritize user security and efficient data management while engaging users effectively. By conducting this system study, we are taking a crucial step towards improving the quality of life for individuals facing serious illnesses and their families at the panchayat level.

2.2 EXISTING SYSTEM

The current palliative care system at the panchayat level lacks a centralized and comprehensive online platform. Palliative care services are often fragmented, making it challenging for individuals with serious illnesses and their families to access the necessary support and information.
Moreover, there is a notable absence of a centralized information hub, leaving residents of the panchayat with limited access to essential educational resources about palliative care services, symptom management, treatment options, and end-of-life care. Additionally, patients and caregivers face difficulties in scheduling appointments with healthcare providers offering palliative care services, potentially causing delays in receiving crucial care. Furthermore, the absence of automated systems for sending notifications and reminders about appointments, medication schedules, or important events related to palliative care contributes to the system's shortcomings. There may also be concerns about the security of patient information due to inadequate user authentication methods in the current system. Lastly, the inefficiencies in data management resulting from the absence of an information repository make it difficult to track patient progress and maintain accurate records. Overall, these challenges underscore the need for a comprehensive and integrated online platform like the proposed Gentle Haloes Palliative Care Service to address

these critical gaps and enhance palliative care at the panchayat level.

2.2.1 NATURAL SYSTEM STUDIED

In the present scenario, the palliative care system at the panchayat level exhibits a series of significant challenges. One of the primary issues is the lack of a centralized and comprehensive online platform. This fragmentation of palliative care services poses a formidable obstacle for individuals dealing with serious illnesses and their families, who often struggle to access vital support and information.

The current system faces significant communication and collaboration deficiencies. Stakeholders, including patients, healthcare providers, volunteers, and caregivers, often rely on ad-hoc or nonexistent communication channels. This lack of structured communication leads to inefficiencies in service delivery and support, hindering the overall effectiveness of palliative care services.

Appointment booking for healthcare services in palliative care is another noteworthy challenge. Patients and caregivers encounter obstacles when attempting to schedule appointments with healthcare providers specializing in palliative care. These difficulties can lead to delays in receiving critical care, further adding to the burdens faced by those in need. Access to essential resources and support services related to palliative care is limited, leaving patients and their families struggling to find the necessary assistance when they need it most. This shortage of resources can result in a less-than-optimal quality of life for individuals dealing with serious illnesses.

The current system lacks automated systems for sending notifications and reminders related to appointments, medication schedules, or other significant events tied to palliative care. This absence of automated reminders can lead to missed appointments and lapses in medication regimens, potentially compromising patient care. User authentication within the current system also raises concerns. Inadequate security measures may jeopardize the confidentiality of patient information, which is a critical aspect of palliative care.

The existing palliative care system at the panchayat level has several problems that make it less effective and accessible. These issues highlight the urgent requirement for creating the Gentle Haloes Palliative Care Service, an all-inclusive online platform, to tackle these problems and improve how palliative care services are provided to those who require them.

2.2.2 DESIGNED SYSTEM STUDIED

The Gentle Haloes Palliative Care Service is designed to address the shortcomings of the current palliative care system at the panchayat level. Our comprehensive online platform aims to provide a centralized source of information, making it easier for individuals with serious illnesses and their families to access essential resources and support.

Through improved communication and collaboration features, patients, healthcare providers, volunteers, and caregivers will be able to interact more effectively, enhancing the overall quality of care. Additionally, our platform streamlines appointment booking, ensuring timely access to palliative care services.

We're committed to overcoming resource scarcity by offering a wide range of support services and resources. Automated notifications and reminders will help patients stay on track with their care plans, and robust user authentication measures will protect patient information. Moreover, our platform includes an efficient data management system, ensuring accurate tracking of patient progress and records maintenance.

In summary, the Gentle Haloes Palliative Care Service is designed to transform palliative care at the panchayat level by providing a user-friendly and comprehensive online platform that addresses existing challenges, thereby improving accessibility, communication, and overall care for individuals facing serious illnesses and their families.

2.3 DRAWBACKS OF EXISTING SYSTEM

- **Lack of Centralization**: Lacks a central place for information, making it hard to find everything needed for palliative care.
- **Limited Information Access:** People in the area can't easily get info about palliative care.
- **Inefficient Communication:** People don't talk or work together well, making care less effective.
- **Appointment Scheduling Challenges:** Hard to book appointments with healthcare providers for palliative care.
- **Few Resources:** Not enough help and support services for those in need.
- **Lack of Automated Reminders:** No automated reminders for appointments or medication.
- **Security Concerns:** Worries about keeping patient information safe.
- **Limited User Engagement:** The existing system may not effectively engage users, hindering their ability to access and utilize palliative care services fully.

2.4 PROPOSED SYSTEM

The Gentle Haloes Palliative Care Service is a user-friendly online platform designed to improve palliative care at the panchayat level. It serves as a central hub for information on palliative care services, treatment options, and symptom management, making it easier for residents to access vital resources. The system enhances communication and collaboration among patients, healthcare providers, volunteers, and caregivers, streamlines appointment scheduling, and offers a wide range of support services to address existing resource scarcity. Automated notifications reduce missed appointments, robust user authentication safeguards data, and efficient data management tracks patient progress. With five distinct modules, including Admin, Patient, Volunteer, Health Care Assistant, and Guest User, the system engages users effectively. In essence, Gentle Haloes aims to transform palliative care, ensuring better access, communication, and overall support for individuals facing serious illnesses and their families at the panchayat level.

2.5 ADVANTAGES OF PROPOSED SYSTEM

- **Centralized Information:** Provides a centralized hub for palliative care information, making it easily accessible to patients and caregivers.
- **Comprehensive Resources:** Offers a wide range of educational resources on palliative care services, symptom management, treatment options, and end-of-life care.
- **Improved Communication:** Facilitates enhanced communication and collaboration among all stakeholders involved in palliative care, promoting better coordination of support.
- **Efficient Appointment Booking:** Streamlines appointment scheduling, ensuring patients and caregivers can access palliative care services promptly.
- **Ample Support Services:** Addresses the scarcity of resources and support services, providing assistance to individuals and families during challenging times.
- **Automated Reminders:** Sends automated notifications and reminders, reducing the risk of missed appointments and medication lapses.
- **Enhanced Security:** Incorporates robust user authentication methods, ensuring the confidentiality and security of patient information.
- **Efficient Data Management:** Enables accurate tracking of patient progress and maintains comprehensive and secure records.
- **Tailored User Modules:** Offers distinct modules for Admin, Patients, Volunteers, Health Care Assistants, and Guest Users, catering to specific needs and roles.
- **Improved User Engagement:** Engages users effectively, ensuring individuals can easily access and utilize palliative care services and resources.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. It helps the designer to identify prospective project focal areas and long-term outcomes. As part of a feasibility study, a project is evaluated for its viability in order to determine whether it will be successful. It is important to analyses all available options when determining whether a given framework is workable and beneficial for further research. The feasibility study's findings should be summarized in a report that suggests whether or not it is feasible to continue with the requirements engineering and system development processes.

3.1.1 Economical Feasibility

In order to assess a new project's value in terms of time and money commitment, it is essential to conduct an economic feasibility analysis. Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on. The suggested system, Gentle Haloes Palliative Care Service has undergone cost-benefit analysis and is proven to be both practical and costeffective given the project's presumptive cost.

The Palliative Care Service website is economically feasible because it offers valuable services that can potentially generate financial gains for the organization. The Palliative Care Service website is economically feasible because it offers various opportunities to generate income. As more people use the platform, there is a chance to receive donations and financial support from individuals, organizations, and sponsors who believe in the cause. Additionally, the website's appointment booking feature can generate revenue through appointment fees or service charges. By managing the costs efficiently and ensuring that the money coming in is more than the expenses, the website can be financially viable and successful. Overall, the website has the potential to create a positive economic impact, making it economically feasible and sustainable in the long run

3.1.2 Technical Feasibility

A Technical Feasibility Study is an essential assessment conducted to evaluate the practicality and viability of a proposed project from a technological standpoint. It aims to determine whether the project's implementation is technically feasible and can be executed using available technology, resources, and expertise. The study involves a comprehensive analysis of the project's technical requirements, including infrastructure, software, hardware, programming languages, and integration capabilities. Ultimately, the Technical Feasibility Study serves as a foundation for effective project planning and ensures that the proposed initiative can be realistically implemented using available technical resources and expertise. The Gentle Haloes Palliative Care Service Website is simple to use and doesn't need much instruction because it is self-explanatory. Even for first-time users, the website is simple to use. The technology is easily accessible, saving clients' money, with only the time spent online being wasted. Gentle Haloes is technically feasible because it has modern infrastructure and uses available technology that allows it to work well and be easily hosted. It is designed to handle more users and content as needed without slowing down. The website's modular design enables seamless integration of different modules for Admin, User, Volunteer and Health Care Assistant providing real-time data synchronization and a cohesive user experience. It is technically feasible due to secure payment gateways, responsive design, and efficient user management. Continuous improvement using feedback and analytics further enhances its viability as a successful Palliative Care Service Website.

3.1.3 Behavioral Feasibility

Behavioral feasibility refers to the assessment of whether a proposed project or system aligns with the organization's culture, existing processes, and the willingness of users to adopt and adapt to the changes brought about by the project. It evaluates the human and behavioral aspects of implementing the project and addresses potential challenges related to acceptance, resistance, and support from end-users.

Two crucial considerations have been made in order to guarantee the system's success:

(1) if users will have enough assistance, and

(2) whether the system will be harmful.

To make sure that the system will be useful after implementation, these issues were carefully examined. Additionally, to make sure the project is behaviorally feasible, all behavioral elements were considered throughout the feasibility assessment. The Palliative Care Service website is behaviorally feasible because it is user-friendly and easy to use. It offers helpful features like user registration, patient profiles, appointment scheduling, and resources for palliative care. The platform encourages smooth communication between healthcare professionals and patients. It also allows for lab testing scheduling and medicine booking, making it convenient for users. With its intuitive design, the website encourages active engagement, promoting palliative care services effectively at the panchayat level.

3.1.4 Feasibility Study Questionnaire

1. **Project Overview?**

   The Gentle Haloes Palliative Care Service aims to create an all-encompassing online platform that facilitates palliative care services at the panchayat level. This project focuses on improving the quality of life for individuals with serious illnesses and their families. The website will serve as a reliable source of information, providing support and resources related to palliative care to the residents of the panchayat.

   The platform's primary goals are to raise awareness about palliative care services, promote communication and collaboration among users, and enhance access to essential palliative care services. It will offer comprehensive information on palliative care, including user registration and authentication, appointment booking, treatment options, resources and support services, notifications, an information repository, lab facilities and end-of-life care. The project comprises five modules such as Admin, Patient, Volunteer and Health Care Assistant Gentle Haloes Palliative Care Service project seeks to establish an interactive and informative platform that brings palliative care services to the panchayat level, fostering better support and care for individuals facing serious illnesses and their families.

2. **To what extend the system is proposed for?**

   This project aims to provide compassionate and accessible palliative care services, streamlining the process of finding essential care and support. By providing easy online access to the website services, the website ensures patients have a convenient and satisfying resources and services. By offering online access to comprehensive resources and facilitating easy appointment booking, the website ensures individuals and their families have a comforting and supportive journey throughout their palliative care experience.

3. **Specify the Viewers/Public which is to be involved in the System?**

   Patients, Caregivers/Family Members, Healthcare Professionals, Volunteers/Asha Workers and General Public.

4. **List the Modules included in your System?**

   Admin, Patient, Volunteer/ Asha Workers, Health Care Assistant and Guest User

5. **Identify the users in your project?**

   Patient and Guest Users

6. **Who owns the system?**

Administrator

7. **System is related to which firm/industry/organization?**

   Healthcare Industry

8. **Details of person that you have contacted for data collection?**

   Zeenath (Asha Worker, Parathodu Panchayat, Ward 12)
   Tuby (Nurse, Primary Health Center)

9. **Questionnaire to collect details about the project? (min 10 questions, include descriptive answers, attach additional docs (e.g. Bill receipts, certificate models), if any?)**

1) **How is patient registration currently managed in the existing system?**

Answer: Patient registration is managed through a user registration form that collects essential details such as name, contact information, and address.

2) How are patient medical records and treatment plans stored and accessed?

Answer: Patient medical records and treatment plans are stored manually.

3) How are appointments scheduled and managed for patients, volunteers, and healthcare assistants in the manual system?

Answer: Appointments are scheduled by using a phone call or paper-based appointment book or log. Volunteers and healthcare assistants manually record them availability, and patients' appointments are noted alongside their preferred schedule.

4) How are resources and support services provided to users in the manual system?

Answer: Resources and support services are provided through printed educational materials and brochures that are handed out to patients and their families.

5) How is user activity monitored and tracked within the manual system?

Answer: User activity is not actively monitored in the manual system as there is no automated tracking. Healthcare professionals may manually record patient interactions and activities in their notes.

6) How are volunteers and healthcare assistants managed and scheduled in the current manual system?

Answer: Volunteer and healthcare assistant management is handled through physical sign-up sheets and manual communication. Coordinators manually assign tasks and shifts to volunteers and assistants.

7) How are donations managed and tracked in the manual system?

Answer: Donations are managed manually by recording donor information and donation amounts in a physical ledger or log. There may also be physical donation boxes at the facility. Donation amount is accepted through cash or credit card facility.

8) How is content, such as articles and educational materials, managed and approved in the manual system?

Answer: Content management is done manually, and the approval process may involve multiple stakeholders reviewing and verifying the information before it is used in patient education.

9) How are notifications and alerts sent to patients and staff in the manual system?

Answer: Notifications and alerts are sent manually through phone calls or physical letters. For example, patients may receive reminders about upcoming appointments via phone calls from healthcare staff.

10) How is user feedback and feedback on services collected and analysed in the manual system?

Answer: Feedback is collected through paper-based feedback forms that patients can fill out.

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- Intel core i3
RAM	- 8 . 0 0 G B
Hard disk	- 1 T B

3.2.2 Software Specification

Front End	-	HTML, CSS
Back End	-	Python Django
Database	-	SQLite
Client on PC	-	Windows 11
Technologies used	-	JS, HTML5, AJAX, J Query, Django, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 Python

Python is a high-level, versatile, and dynamically-typed programming language known for its simplicity and readability. Created by Guido van Rossum in 1991, Python has gained immense popularity within the software development community due to its ease of use, extensive standard library, and wide range of applications. Python emphasizes code readability and clean, easy-to-understand syntax, using indentation to define code blocks, which encourages consistent and human-readable code. It is a high-level language that abstracts many complex operations, handling memory management and low-level details, allowing developers to focus on solving problems. Python offers an extensive standard library that provides modules and packages for various tasks, reducing the need for third-party libraries. It is cross-platform compatible and open-source, fostering a large and active developer community that continually enhances the language. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Its ecosystem includes a vast repository of third-party libraries and frameworks, making it suitable for web development, data science, scientific computing, automation, game development, desktop applications, and IoT projects. In conclusion, Python is a versatile, open-source programming language that stands out for its simplicity, readability, and a wide range of applications, making it a preferred choice for developers across diverse domains.

3.3.2 Django

Django is a high-level, open-source web framework designed for the development of robust and scalable web applications. Conceived by Adrian Holovaty and Simon Willison in 2003, it has since matured into a powerful, adaptable tool widely embraced by the web development community. Django adheres to the Model-View-Controller (MVC) architectural pattern, implementing its own variant known as Model-View-Template (MVT). This framework excels in accelerating development, advocating for clean and reusable code, and adhering to the "Don't Repeat Yourself" (DRY) principle. Replete with a comprehensive set of features, including an Object-Relational Mapping (ORM) system, built-in authentication, and an admin interface, Django reduces the reliance on third-party libraries, effectively expediting development. Embracing modularity, Django encourages component-based application construction, ensuring complex systems are organized and maintainable. Security is a top priority, with built-in measures guarding against common web vulnerabilities, and comprehensive protection against threats like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Supporting multiple database backends, such as PostgreSQL, MySQL, SQLite, and Oracle, Django streamlines database operations via the ORM. Its scalability prowess is evidenced by its capacity to handle high-traffic websites, with provisions for load balancing, database sharding, and caching mechanisms. Backed by a thriving and engaged community, Django boasts an extensive ecosystem comprising reusable applications, packages, and plugins, simplifying extension of functionality and problem-solving. Django's versatility renders it apt for various web application development, including content management systems, e-commerce platforms, social networks, news portals, data analytics, and educational tools. In summation, Django is a potent, feature-rich web framework that expedites development, promotes clean code, and fortifies security. Characterized by a "batteries-included" philosophy, modularity, and a dynamic community, it remains the top choice for web developers seeking to construct resilient and scalable web applications spanning diverse domains, all while streamlining the development process and enabling innovation.

3.3.3 SQLite

SQLite is a lightweight and self-contained open-source relational database management system (RDBMS) developed by D. Richard Hipp in 2000. Renowned for its simplicity and efficiency, it operates as a serverless library, eliminating the need for a separate server process, and excels in embedded systems, mobile applications, and small to medium-scale projects. Offering zero configuration, ACID transaction support, and cross-platform compatibility, SQLite's self-contained nature simplifies database management, with the entire database stored in a single file. Its high-performance attributes, rich SQL capabilities, and adaptability have made it the database of choice for mobile apps, embedded systems, desktop software, web browsers, and prototyping purposes. In essence, SQLite is a versatile, efficient, and resource-friendly RDBMS, offering an ideal solution for various applications, from local data storage in mobile apps to serving as the data backbone for embedded systems and desktop software.

CHAPTER 4
SYSTEM DESIGN

4.1 INTRODUCTION

System design is a critical phase in the software development life cycle that plays a pivotal role in the successful creation of complex software systems. It is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements and objectives. Effective system design not only ensures that the software functions as intended but also addresses scalability, maintainability, and performance considerations. This phase is a bridge between the requirements specification and the actual implementation, where high-level concepts are transformed into detailed, practical blueprints. In this document, we will delve into the key principles, methodologies, and best practices for system design, emphasizing the importance of creating a robust and efficient architecture that aligns with the project's goals. It will encompass topics such as system architecture, data design, user interfaces, integration, security, and scalability, all of which are integral in producing software that meets both the immediate and long-term needs of stakeholders.

4.2 UML DIAGRAM

A standardized form of communication known as the Unified Modeling Language (UML) is employed for the purpose of conceptualizing, defining, designing, and illustrating software systems. The Object Management Group (OMG) was responsible for the development of UML, and the initial UML 1.0 specification was unveiled in January 1997. It is crucial to distinguish UML from programming languages like Java, C++, and COBOL. UML is a versatile visual modeling language utilized for software systems and a graphical language employed for software diagrams. UML has applications extending beyond software systems and can also represent non-software systems, such as manufacturing processes, although its primary use is in the representation of software systems.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 Use Case Diagram

A use case diagram is a visual representation of how clients and other external onscreen characters are connected to internal system components. Understanding, organizing, and coordinating a system's utilitarian requirements as viewed through the eyes of its users is the fundamental task of a use case diagram. Use case diagrams are typically created using the Unified Modelling Language (UML), a standard language for modeling real-world objects and systems.

Customer support, product obtaining, catalogue overhauling, and payment processing are as it were a few examples of use cases within the setting of item sales. Use cases can be used to achieve a variety of framework objectives, including setting fundamental prerequisites, confirming equipment plans, testing and investigating program, creating online offer assistance references, or performing client bolster obligations.

The system boundaries, actors, use cases, and their connections together make up a use case diagram. The system boundary establishes the system's boundaries in reference to its surroundings. Actors are often defined depending on the roles they play and reflect the people or systems that interact with the system. The precise activities or behaviors that actors carry out within or close to the system are known as use cases. Finally, the graphic shows the connections between actors and use cases as well as the use cases themselves.

Use case diagrams are graphical representations used to capture the functional requirements of a system. When drawing a use case diagram, it is important to follow these guidelines to ensure an efficient and effective diagram:

- Give use cases titles that are evocative of their functions and reflect those functions appropriately.
- Give actors proper names to make it easier to understand what part they play in the system.
- Verify that the diagram accurately depicts all links and dependencies.
- Since the major objective is to identify the essential criteria, refrain from listing every potential relationship.
- When required, take notes to help you remember key detail

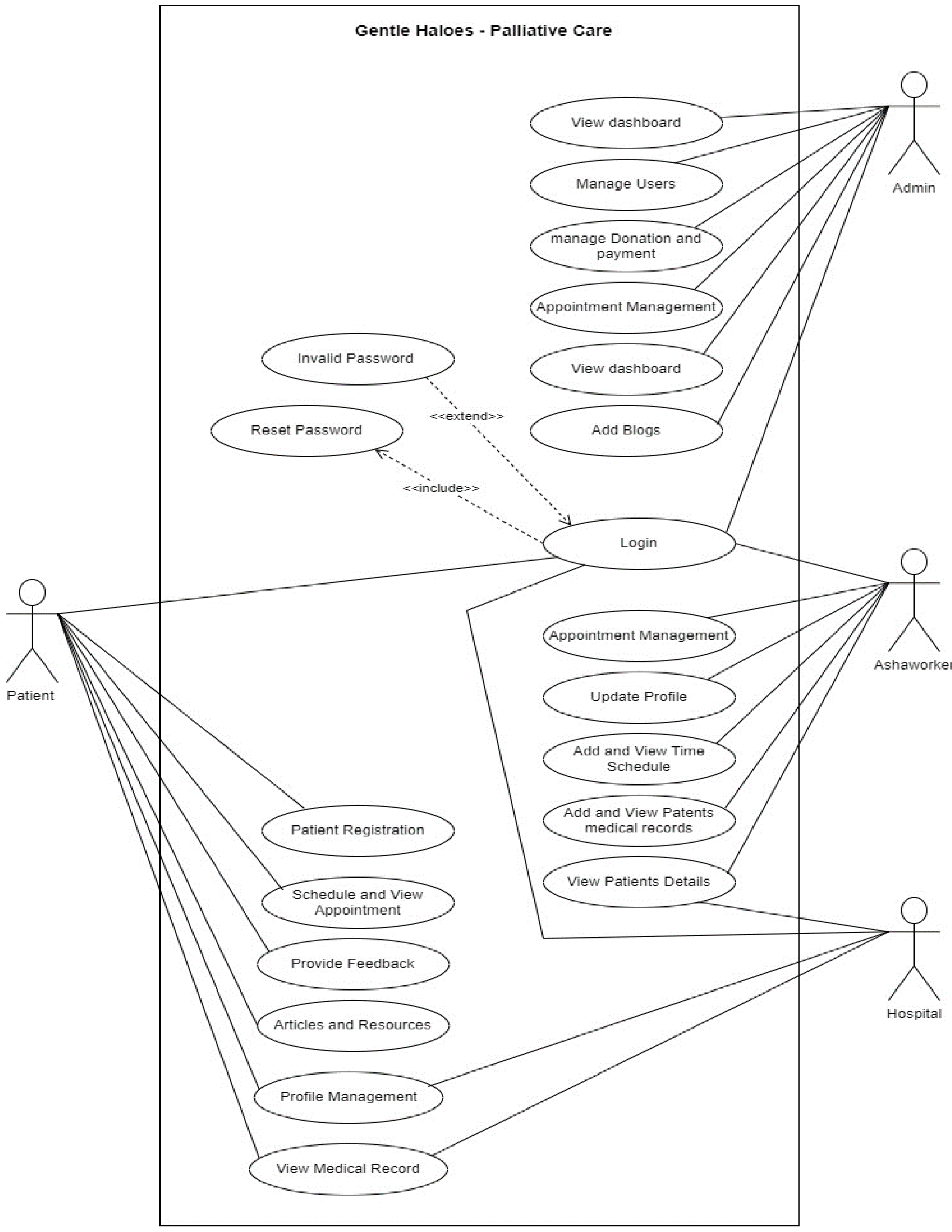


Fig 1: Use Case Diagram

4.2.2 Sequence Diagram

A sequence diagram, which is a type of interaction diagram, displays the interactions between different system components in chronological order. It shows how several things can communicate with one another through a string of messages. Event scenarios and event scenarios diagrams are other names for these pictures. In software engineering, sequence diagrams are frequently used to describe and comprehend the needs of both new and old systems. They support the visualization of object control relationships and the detection of systemic issues.

Notations for Sequence Diagrams -

i. Actors - In UML, an actor is a role that interacts with the system and its objects. Actors frequently exist outside of the system represented by the UML diagram. A variety of roles, such as those of external subjects or human users, can be played by actors. UML diagrams depict actors using a stick person notation. A sequence diagram may have multiple actors, depending on the circumstance being depicted.

ii. Lifelines - In a sequence diagram, a lifeline is a vertical dashed line that reflects an object's lifetime during the interaction. Each lifeline is labeled with the participant's name and serves to symbolize a specific participant in the series of events. The lifeline, which is represented as a vertical line running from the participant's activation point to its deactivation point, displays the participant's history of events.

iii. Messages – A crucial part of sequence diagrams, messages show how different objects or system components interact and communicate. Synchronous and asynchronous communications, create and delete messages, self-messages, reply messages, found messages, and lost messages are some of the categories they fall under. Guards are often employed to simulate constraints on message flow.

iv. Guards – In UML, guards are used to describe conditions and are used to limit the flow of messages when a specific condition is satisfied. This function is crucial for informing software developers of any restrictions or limitations related to a system or specific process. Sequence diagram applications include: • Modeling and visualizing the logic of intricate activities, processes, or procedures.

- Displaying UML use case diagrams in depth.
- Recognizing the precise operation of present or upcoming systems.
- Visualizing how messages and tasks move between objects or components in a system.
- Overall, sequence diagrams are useful for representing the flow of interactions between objects in a system, and can help both businesspeople and software engineers better understand and communicate system requirements and behavior.

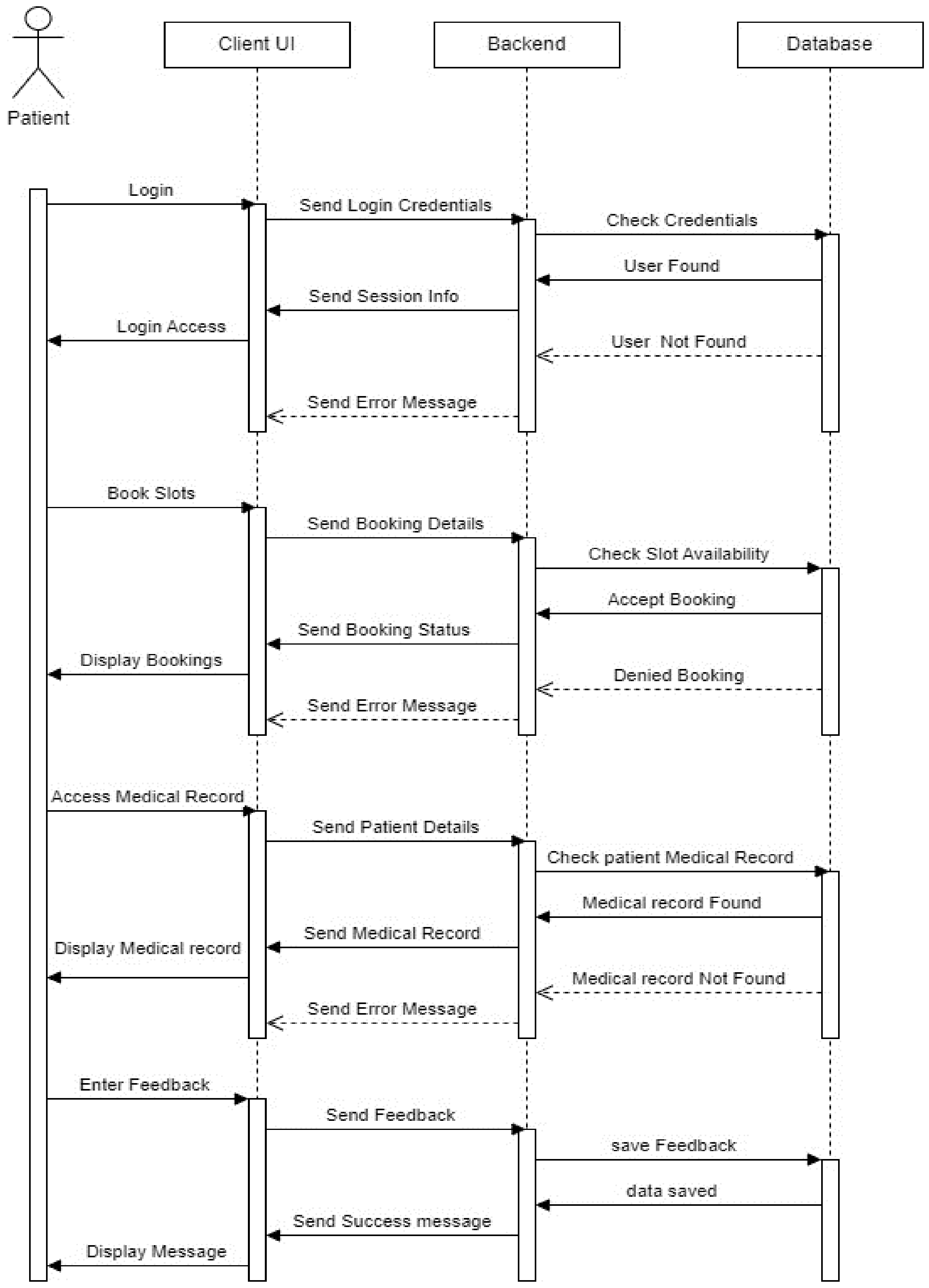


Fig 2: Sequence Diagram

4.2.3 State Chart Diagram

A state diagram, often constructed using the Unified Modeling Language (UML), serves as a visual representation illustrating the various states an object can be in and how it can transition between those states. This diagram is also known as a state machine diagram or state chart diagram.

A State Chart Diagram is a type of UML behavioral diagram used to depict the behavior of a system or object across time. It comprises several key components, such as:

- Initial State: This represents the starting point of the system or object, denoted by a solid black circle.
- State: This component describes the current condition of the system or object at a particular moment and is depicted as a rectangle with rounded corners.
- Transition: This element illustrates the shift of the system or object from one state to another and is represented by an arrow.
- Event and Action: An event acts as a trigger for a transition, and an action signifies the behavior or outcome of the transition.
- Signal: A message or trigger, brought about by an event, sent to a state, leading to a transition.
- Final State: The State Chart Diagram concludes with a Final State component, depicted as a solid black circle with a dot inside, indicating the completion of the system or object's behavior.

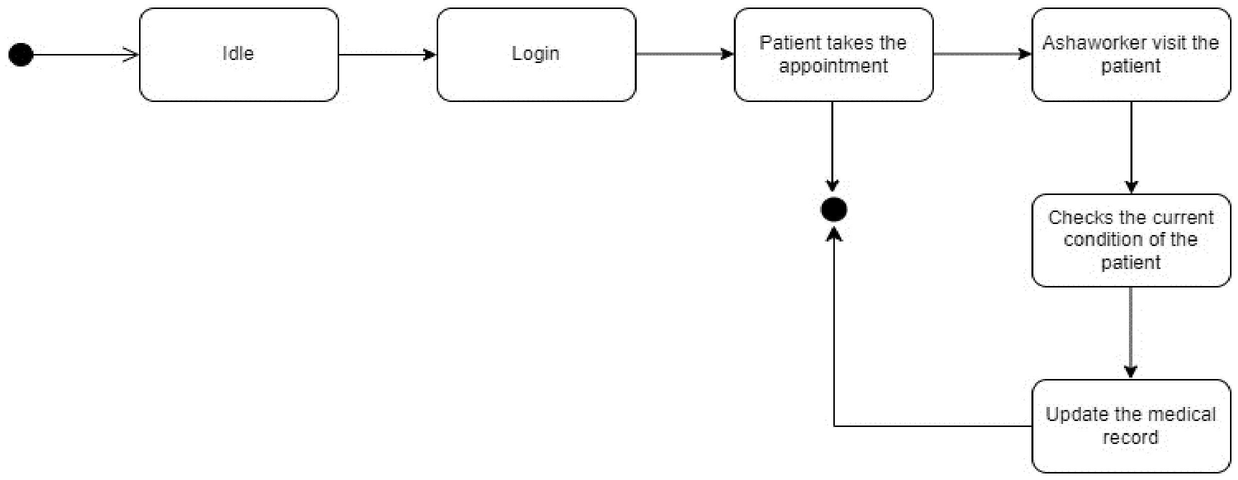

Idle
Login
Patient takes the appointment
Ashaworker visit the patient
Checks the current condition of the patient
Update the medical record

Fig 3: State Chart Diagram

4.2.4 Activity Diagram

An activity diagram is a visual representation of a workflow that illustrates the sequence of activities, showing how one activity leads to another. In this context, an activity refers to a system operation, and they are linked together in a controlled flow. The flow can take different forms, such as running in parallel, concurrently, or branching out. Activity diagrams employ various tools like branching and joining to manage all these types of flow control. These diagrams belong to the category of behavior diagrams, and they reveal the system's behavior. They display the control flow from the beginning to the end and depict the various decision points encountered during the activity's execution.

The essential components of an activity diagram are as follows:

- Initial node: This marks the starting point of the activity diagram, indicated by a black circle.
- Activity: Representing a task or action performed by the system or entity, it's symbolized by a rectangle with rounded corners.
- Control flow: Depicts the sequence of activities or actions carried out by the system or entity, represented by an arrow.
- Decision node: This signifies a decision or branching point in the activity flow, marked by a diamond shape.
- Merge node: It's used to combine multiple branches of the activity flow into a single flow, shown as a diamond shape with a plus sign inside.
- Fork node: Splits the activity flow into multiple parallel flows and is represented by a solid black circle with multiple arrows.
- Join node: Serves to reunite multiple parallel flows into a single flow, displayed as a solid black circle with multiple arrows pointing toward it.
- Final node: Marks the conclusion of the activity diagram, indicated by a black circle with a dot inside.
- Object flow: It demonstrates the flow of objects or data between activities through a dashed arrow.

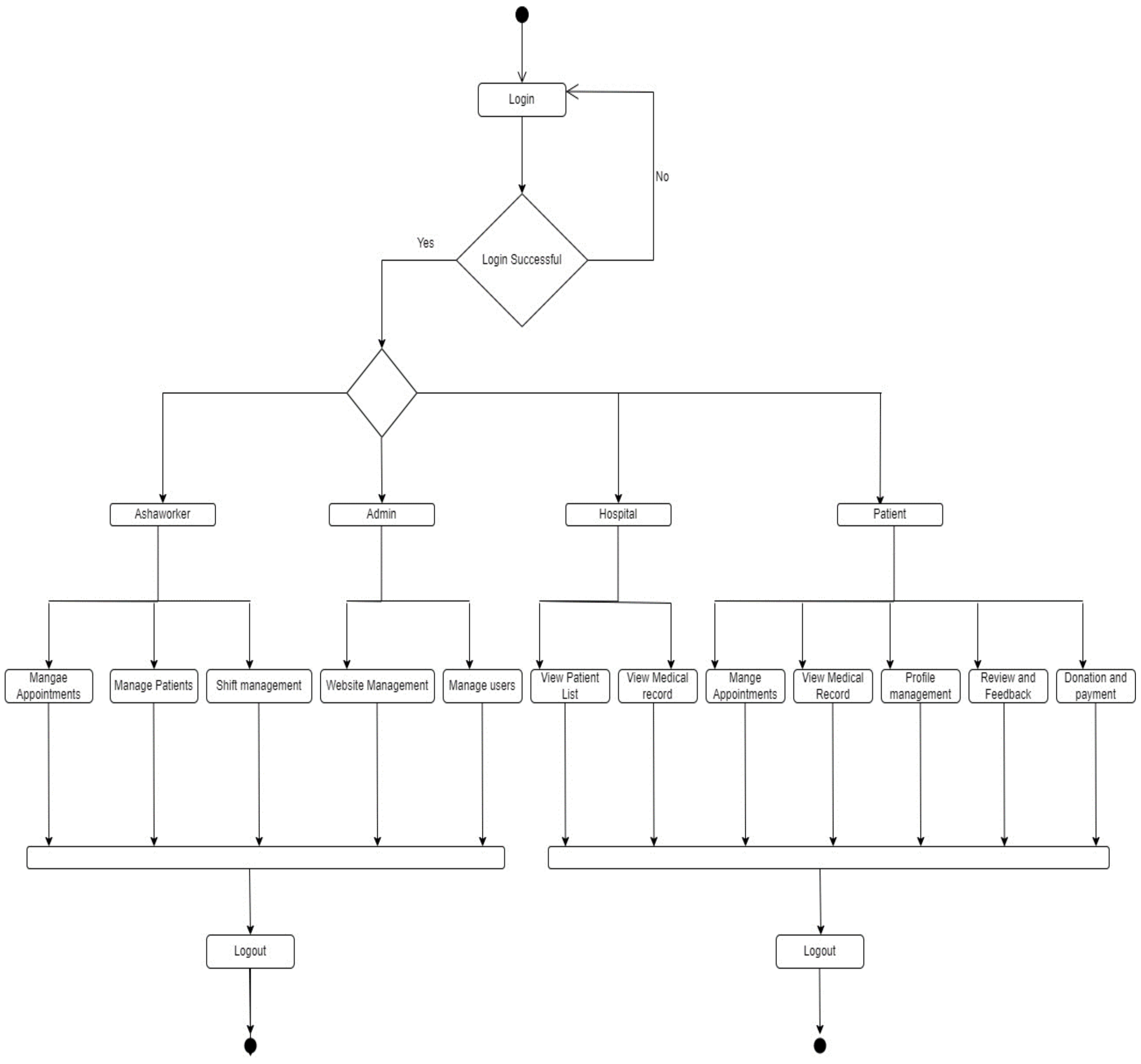


Fig 4: Activity Diagram

4.2.5 Class Diagram

The class diagram stands as a fundamental element in the realm of object-oriented modeling, serving as the primary tool for conceptually modeling an application's structure. Notably, class diagrams can also serve in detailed modeling that can later be translated into programming code and are useful for data modeling purposes.

Class diagrams are a pivotal facet of UML (Unified Modeling Language) that plays a role in representing classes, objects, interfaces, their attributes, and the relationships they share in a system. Some crucial elements of a class diagram include:

- Class: It serves as a blueprint or template for creating objects and is visually represented as a rectangle displaying the class name, its attributes, and methods.
- Interface: This signifies a collection of abstract methods defining a contract between a class and the external world. It's depicted as a circle with the interface name enclosed.
- Object: An object is an instance of a class with both state and behavior. It's visually represented as a rectangle, featuring the object's name.
- Association: This represents a relationship connecting two classes, indicating a connection or link. It's symbolized as a line and may include optional attributes such as directionality, multiplicity, and role names.
- Aggregation: This portrays a part-whole relationship where the whole (the aggregator) is made up of parts (the aggregates). It's depicted as a diamond shape on the aggregator side.
- Composition: A more rigid form of aggregation where parts cannot exist independently of the whole. It's represented as a solid diamond shape on the aggregator side.
- Inheritance: This defines a relationship between a superclass and its subclasses, signifying an "is-a" relationship. It's depicted as a line with an open arrowhead, pointing from the subclass to the superclass.
- Dependency: This showcases a relationship where changes in one class may impact another. It's presented as a dashed line with an arrowhead, pointing from the dependent class to the independent one.
- Multiplicity: This indicates the number of instances of a class that can be associated with another class and is expressed as a range of values near the association or aggregation line.

Class diagrams are vital in the design and modeling of object-oriented software systems, providing a visual representation of the system's structure, functionality, and the relationships among its components. They play a pivotal role in software development, maintenance, and enhancing communication among team members.

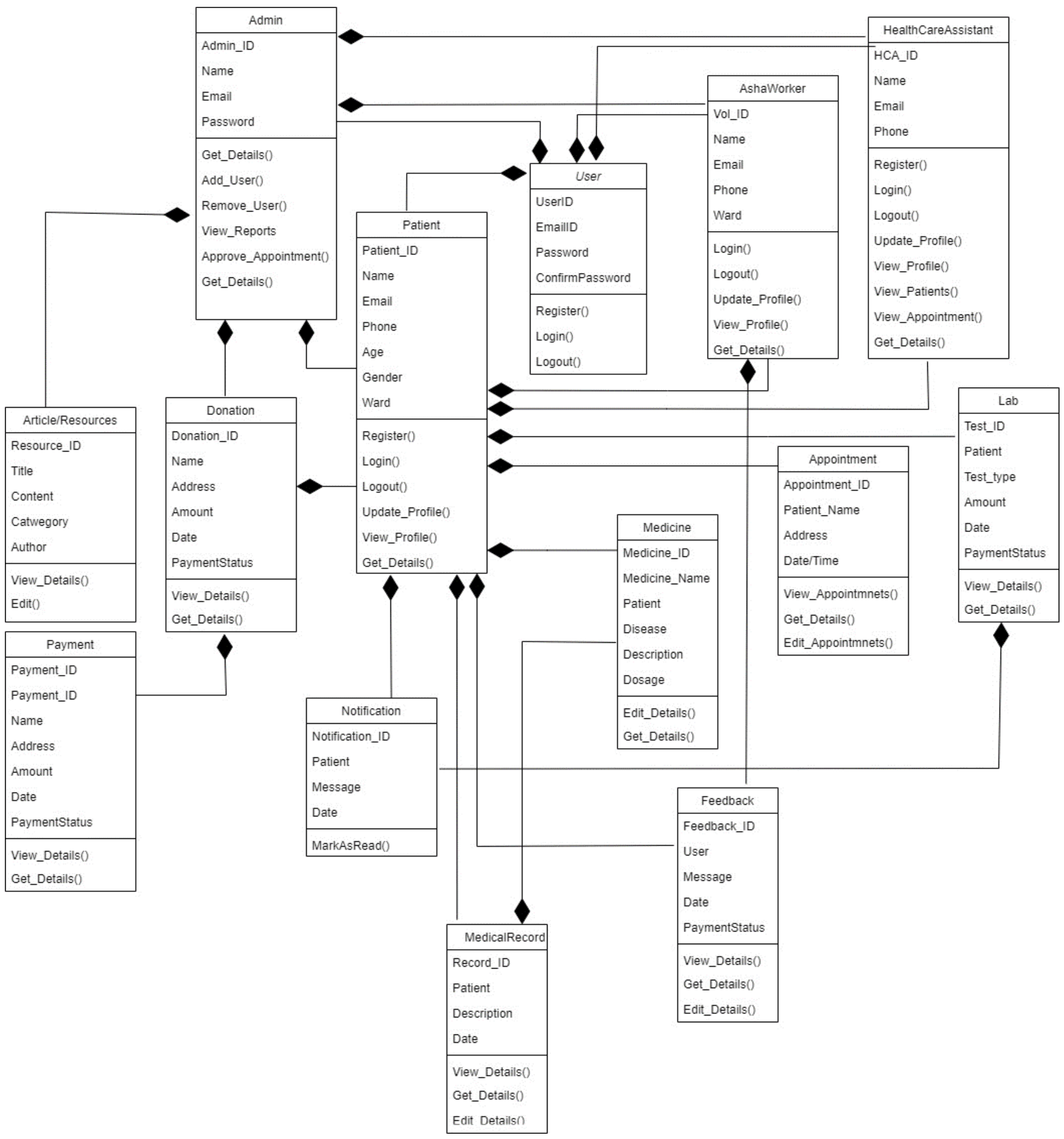


Fig 5: Class Diagram

4.2.6 Object Diagram

Class diagrams and object diagrams are closely intertwined in the context of object-oriented modeling. Object diagrams serve as instances of class diagrams, offering a snapshot of the system's state at a particular moment. Both types of diagrams employ similar concepts and notations to portray a system's structure. While class diagrams are used to model the system's overall structure, encompassing classes, attributes, and methods, object diagrams focus on depicting a group of objects and their connections at a specific point in time.

An object diagram falls under the category of structural UML diagrams and provides insight into instances of classes and their relationships. The primary components of an object diagram are as follows:

- Object: An object stands as an instance of a class, symbolizing a specific entity in the system. It's represented as a rectangle containing the object's name.
- Class: A class serves as the blueprint or template for creating objects, defining attributes and methods. It's depicted as a rectangle with compartments for the class name, attributes, and methods.
- Link: A link represents a relationship between two objects, signifying a connection or association. It's illustrated as a line connecting two objects, sometimes with optional labels.
- Attribute: An attribute represents a property or characteristic of an object, describing its state. It's indicated as a name-value pair within the object's rectangle.
- Value: A value denotes a particular instance or setting of an attribute and is positioned inside the attribute's name-value pair.
- Operation: An operation signifies a behavior or action that an object can perform, appearing as a method name within the class's rectangle.
- Multiplicity: Multiplicity indicates the number of instances of a class that can be linked to another class, represented as a range of values (e.g., 0..1, 1..*, etc.) near the link between objects.

Object diagrams are valuable for visualizing the relationships between objects and their attributes within a system. They aid in understanding the system's behavior at a specific moment and in pinpointing potential issues or inefficiencies in the system.

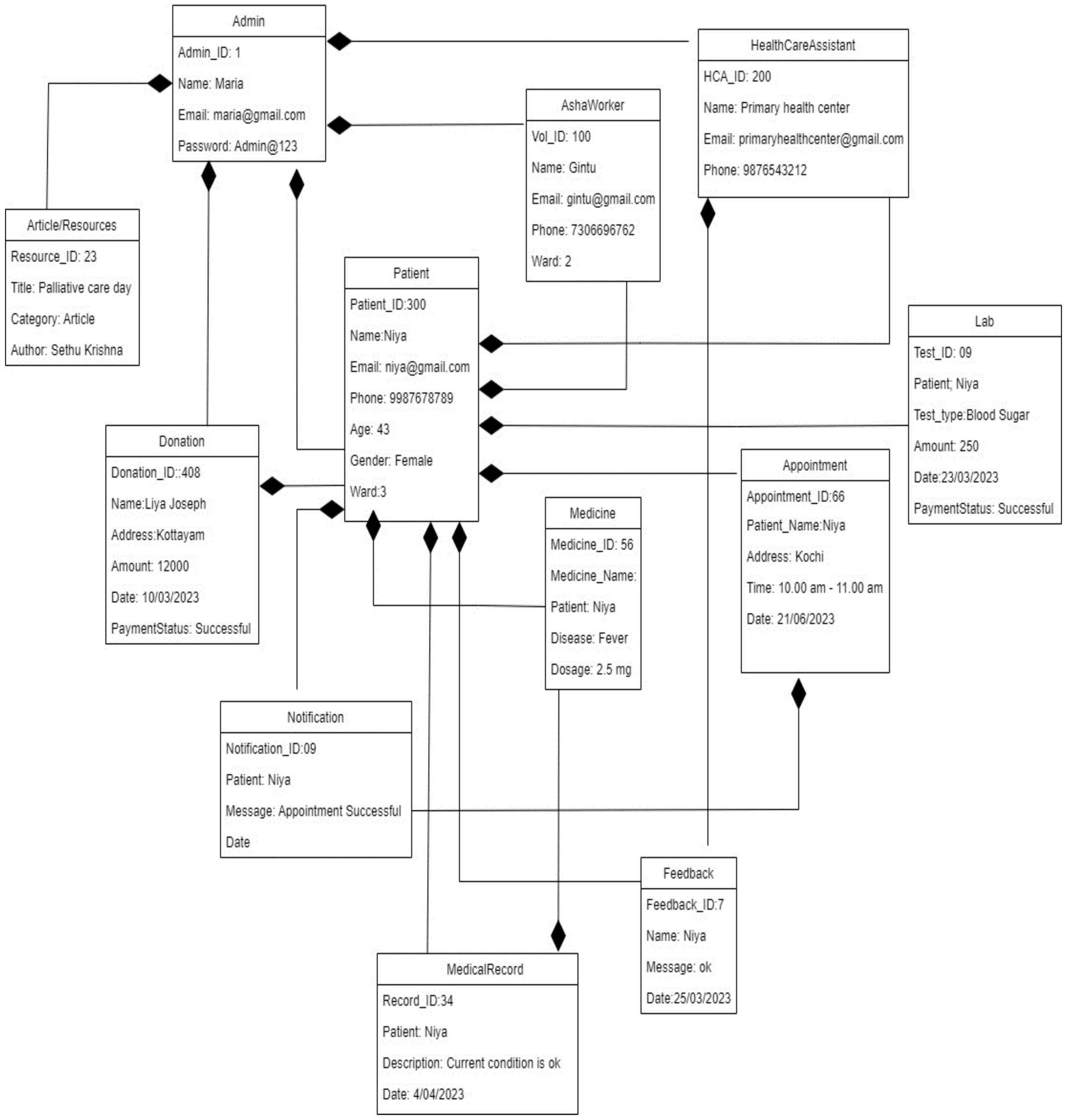


Fig 6: Object Diagram

4.2.7 Component Diagram

A UML component diagram serves to illustrate how various components connect to form more extensive components or software systems. This diagram is a valuable tool for representing the structure of intricate systems that consist of multiple components. By employing component diagrams, developers can readily visualize the internal arrangement of a software system and comprehend how different components collaborate to achieve specific tasks.

Key components within this diagram include:

- Component: These are self-contained units of functionality in a system that provide interfaces for interaction with other components. They are depicted as rectangles containing the component's name.
- Interface: This signifies an agreement between a component and its surroundings or other components, specifying a set of methods available for use by other components. It is visualized as a circle containing the interface name.
- Port: A point of interaction between a component and its environment or other components, represented as a small square on the component's boundary.
- Connector: A connection between two components that enables communication or data exchange. It appears as a line with optional embellishments and labels.
- Dependency: This is a relationship between two components where one relies on another for its implementation or functionality. It's portrayed as a dashed line with an arrowhead pointing from the dependent component to the independent one.
- Association: A connection between two components indicating a relationship or link. It's depicted as a line joining two components, possibly with direction, multiplicity, and role information.
- Provided/Required Interface: A provided interface is one offered by a component to other components, while a required interface is one a component depends on from others for proper functioning. These are illustrated as lollipops and half-circles, respectively.

Component diagrams prove invaluable in modeling the architecture of a software system and can help uncover potential issues and enhancements in the design. They also serve as a means to convey the system's structure and behavior to stakeholders, including developers and project managers.

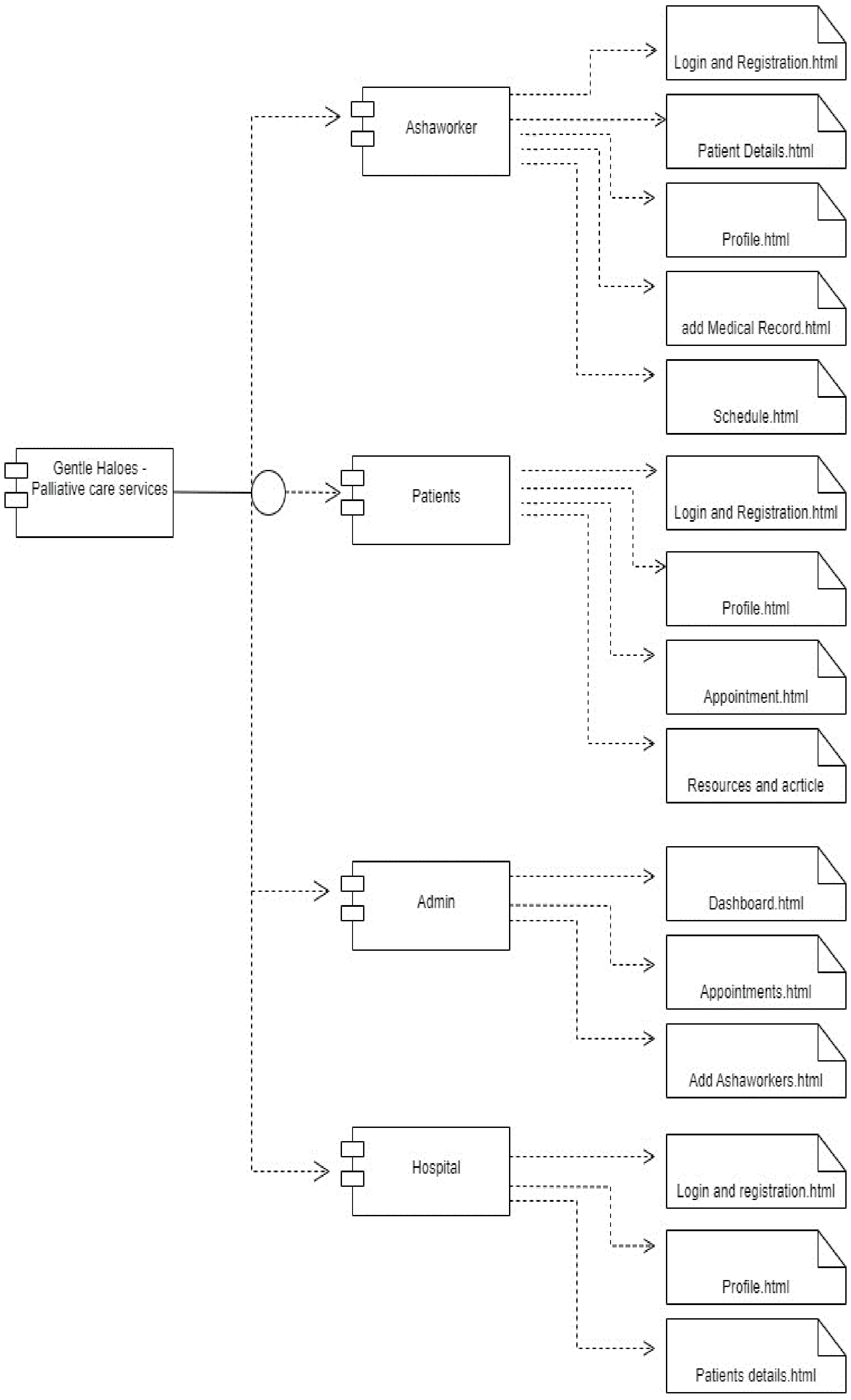


Fig 7: Component Diagram

4.2.8 Deployment Diagram

A deployment diagram, falling under the UML category, is dedicated to the physical hardware required to implement software. It offers a static view of a system's deployment, involving nodes and their interconnections. Essentially, the deployment diagram establishes a link between the software's architecture and the physical system setup, demonstrating how the software will run on these nodes. Communication paths are employed to depict the relationships between these nodes. In contrast to other UML diagram types, which concentrate on a system's logical elements, the deployment diagram accentuates the hardware structure.

The critical elements in a deployment diagram encompass:

- Node: These are actual or virtual machines on which components or artifacts are deployed. Nodes are depicted as boxes containing their names.
- Component: Components are software elements that execute specific functions or deliver specific services. They are portrayed as rectangles featuring the component's name.
- Artifact: Artifacts represent tangible pieces of data used or produced by a component, displayed as rectangles with the artifact's name.
- Deployment Specification: This specification defines how a component or artifact is deployed on a node, covering details like location, version, and configuration parameters.
- Association: Associations denote relationships between a node and a component or artifact, signifying a deployment dependency. These relationships are illustrated by lines connecting the components with options for direction, multiplicity, and role names.
- Communication Path: Communication paths exemplify the links between nodes, like network connections or communication channels, depicted by lines with optional labels and embellishments.

Deployment diagrams facilitate the visualization of a system's physical architecture and the detection of potential issues or bottlenecks in the deployment process. They also prove helpful in planning deployment strategies and optimizing the utilization of hardware resources.

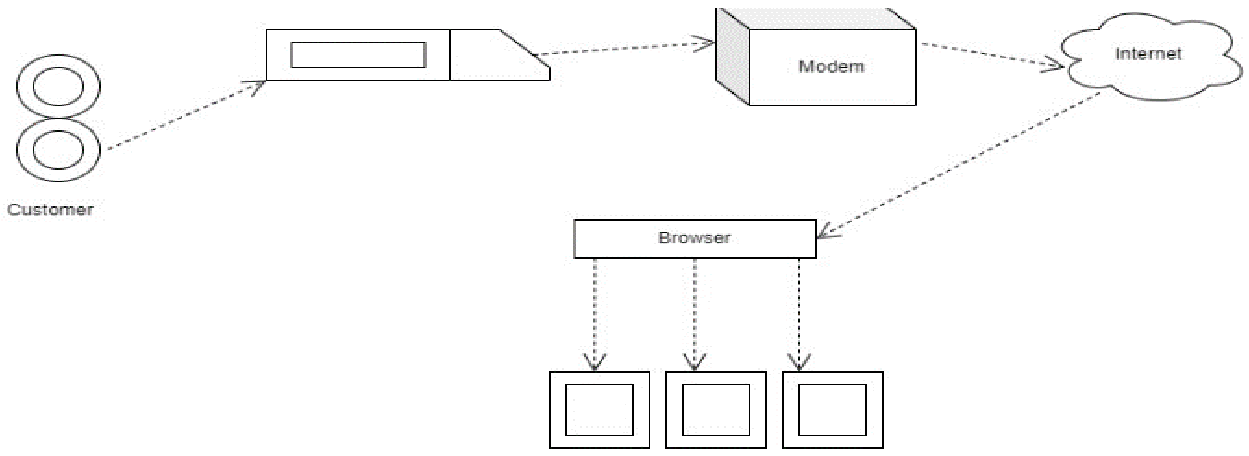
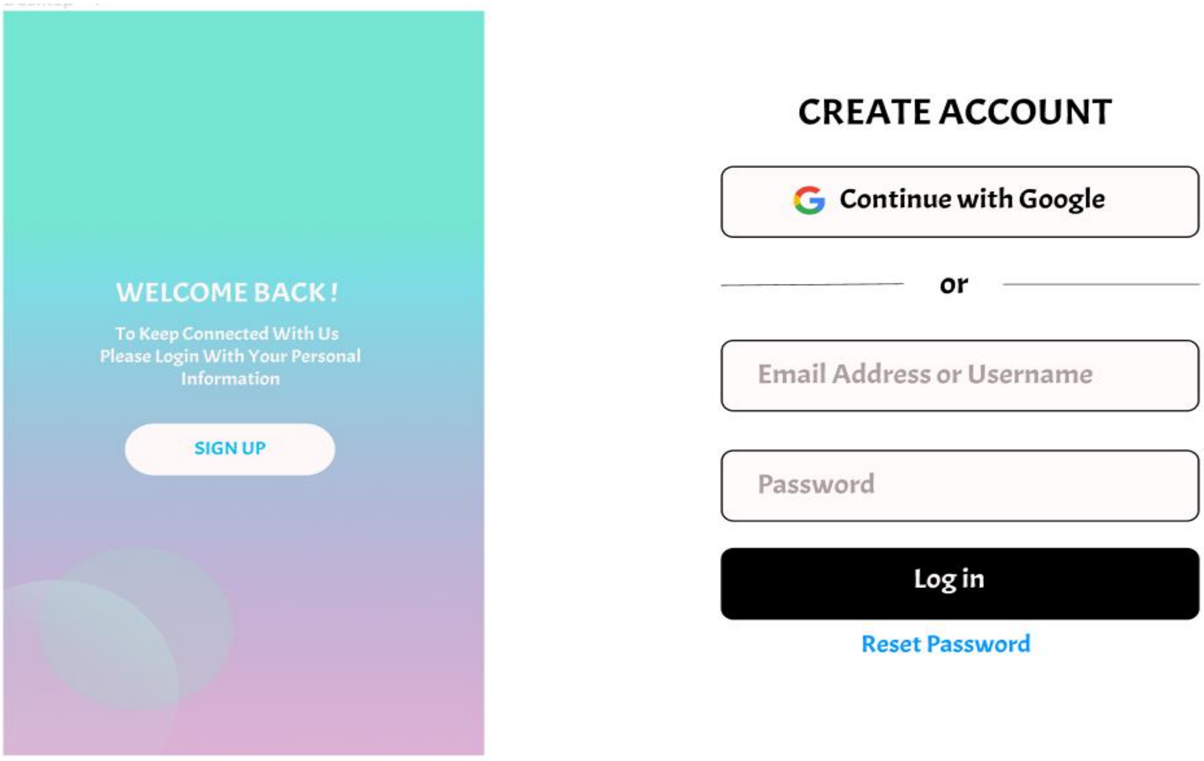


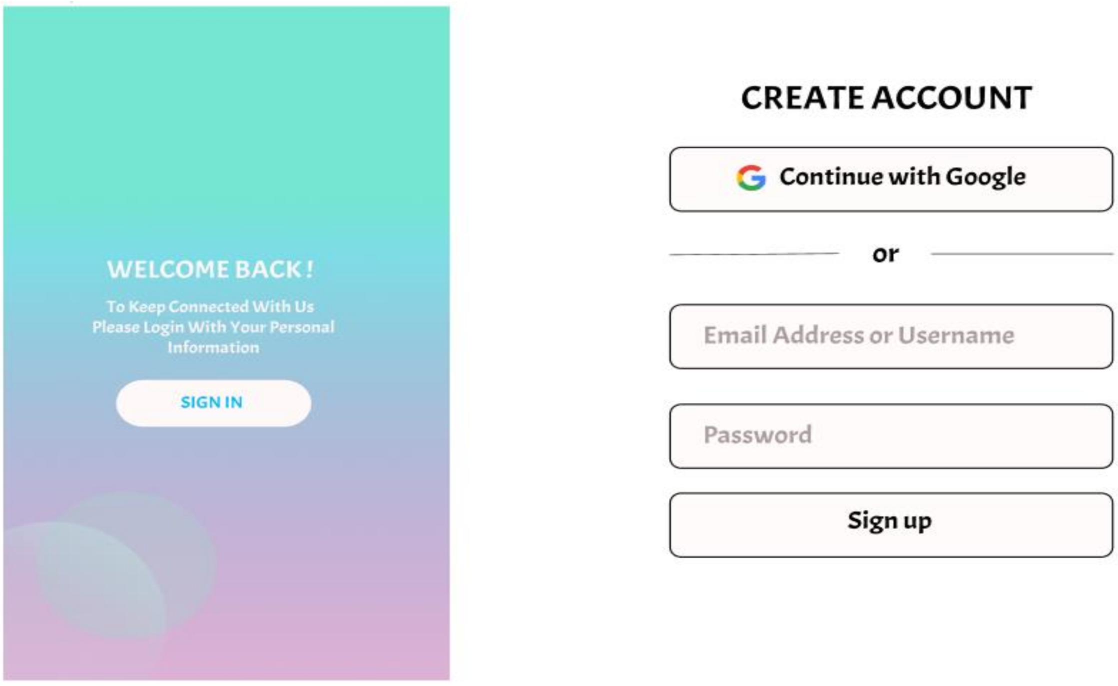
Fig 8: Deployment Diagram

4.3 USER INTERFACE DESIGN USING FIGMA

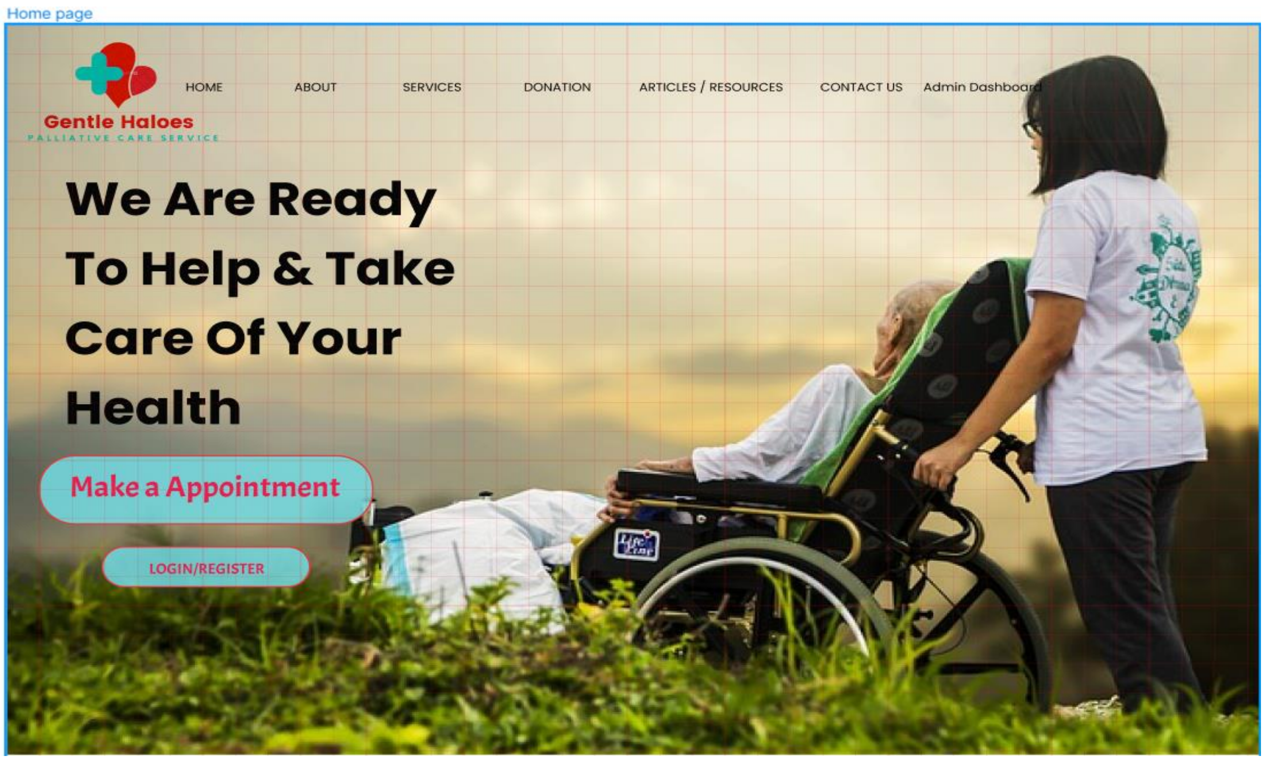
4.3.1 Form Name: Sign Up


WELCOME BACK !
To Keep Connected With Us Please Login With Your Personal Information
SIGN UP
CREATE ACCOUNT
Continue with Google
or
Email Address or Username
Password
Log in
Reset Password

4.3.2 Form Name: Sign In

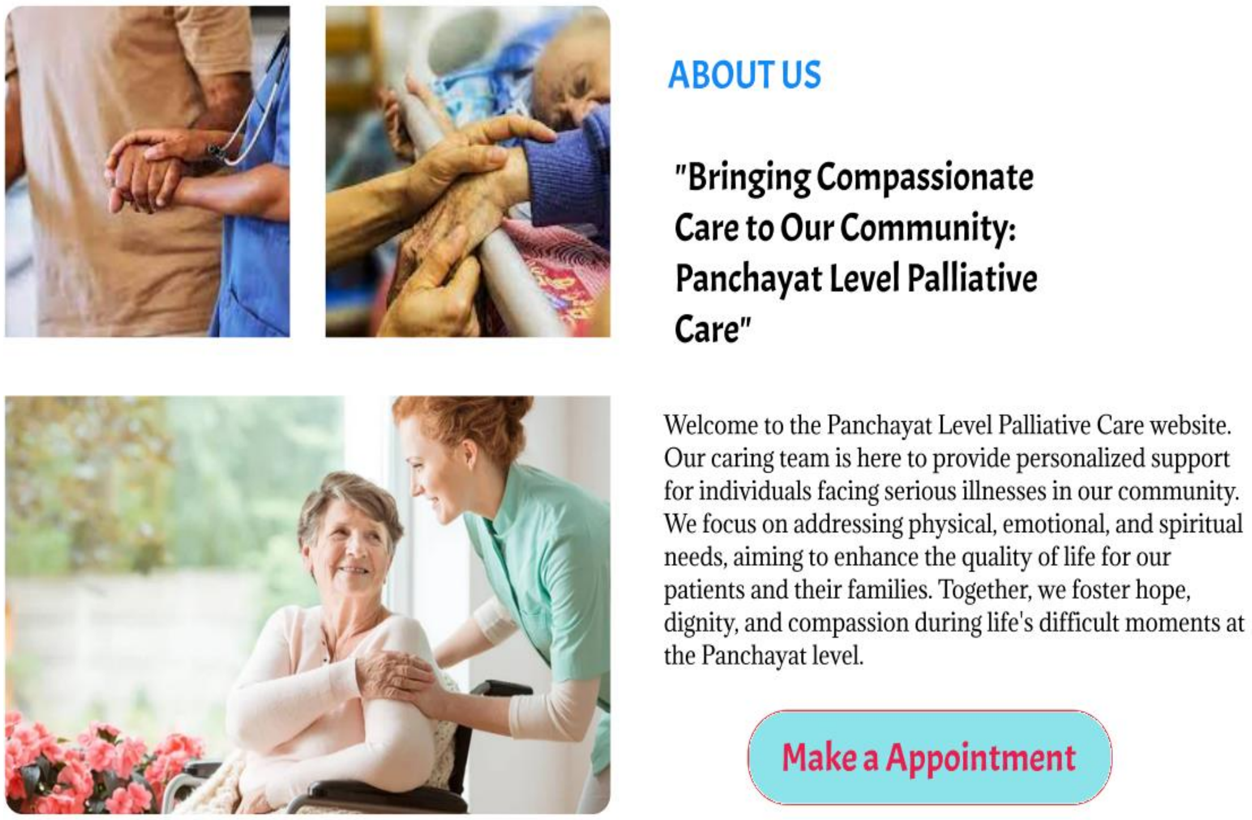

WELCOME BACK !
To Keep Connected With Us Please Login With Your Personal Information
SIGN IN
CREATE ACCOUNT
Continue with Google
or
Email Address or Username
Password
Sign up

4.3.3 Form Name: Home Page



We Are Ready To Help & Take Care Of Your Health

4.3.4 Form Name: About Us Page

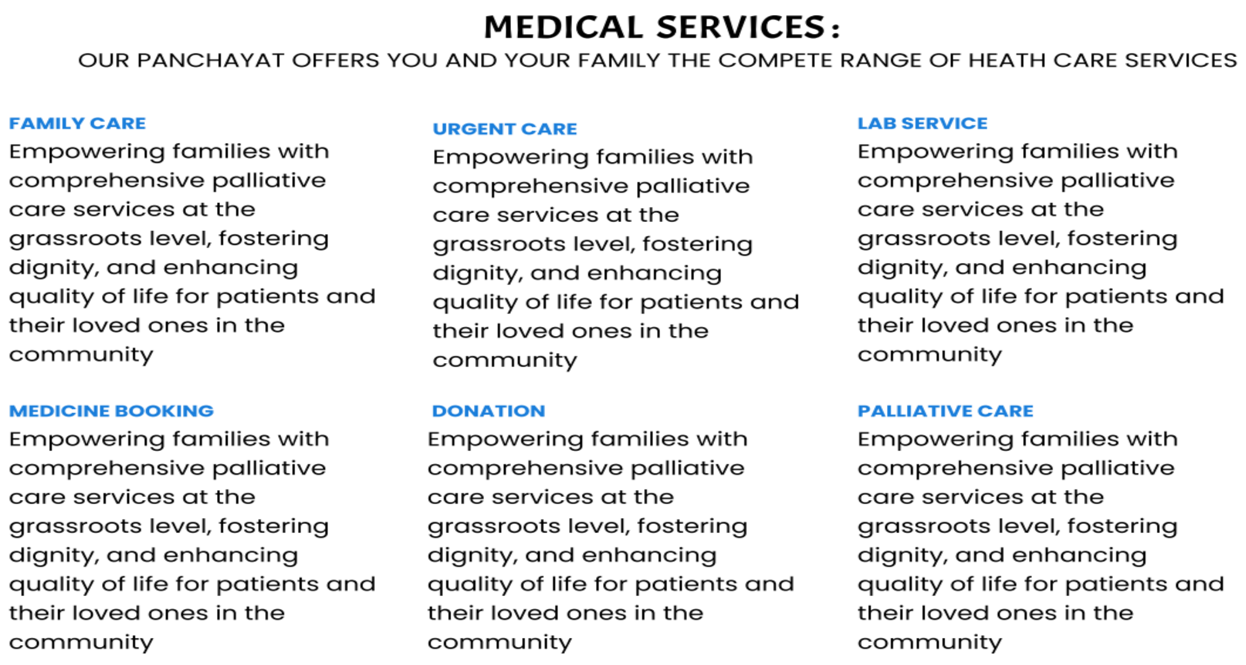


ABOUT US

"Bringing Compassionate Care to Our Community: Panchayat Level Palliative Care"

Welcome to the Panchayat Level Palliative Care website. Our caring team is here to provide personalized support for individuals facing serious illnesses in our community. We focus on addressing physical, emotional, and spiritual needs, aiming to enhance the quality of life for our patients and their families. Together, we foster hope, dignity, and compassion during life's difficult moments at the Panchayat level.

4.3.5 Form Name: Services Page



MEDICAL SERVICES:

OUR PANCHAYAT OFFERS YOU AND YOUR FAMILY THE COMPETE RANGE OF HEATH CARE SERVICES

FAMILY CARE
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

URGENT CARE
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

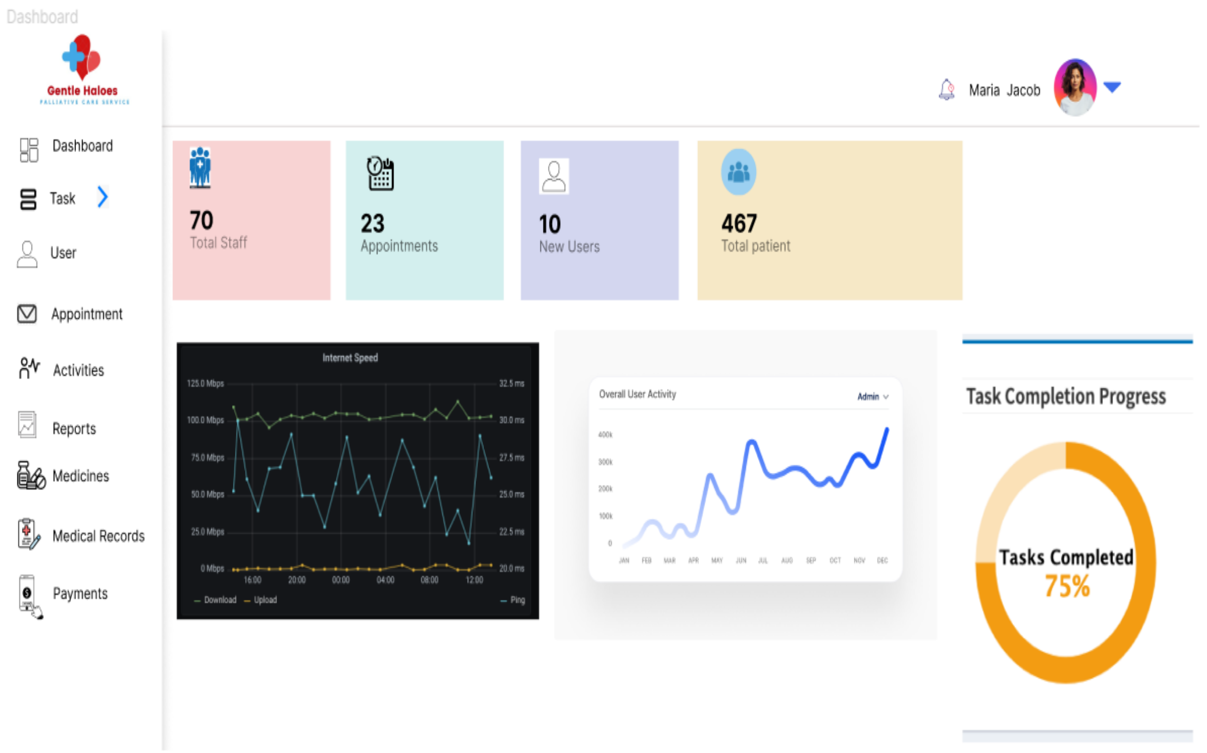
LAB SERVICE
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

MEDICINE BOOKING
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

DONATION
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

PALLIATIVE CARE
Empowering families with comprehensive palliative care services at the grassroots level, fostering dignity, and enhancing quality of life for patients and their loved ones in the community

4.3.6 Form Name: Admin Dashboard



70 Total Staff

23 Appointments

10 New Users

467 Total patient

Tasks Completed 75%

4.4 DATABASE DESIGN

A database is a well-structured repository of information, organized to facilitate easy accessibility, management, and updates. Safeguarding data is a fundamental priority in any database. The process of designing a database involves two key phases. In the initial phase, user requirements are gathered to create a database that aligns closely with those needs. This phase, termed information-level design, is carried out independently of any specific database management system (DBMS). In the second phase, the design is translated from an information-level design into a concrete DBMS design tailored for constructing the system. This stage is referred to as physical-level design and involves considering the specific characteristics of the chosen DBMS. Alongside system design, there is a parallel focus on database design, with two primary objectives: ensuring data integrity and achieving data independence.

4.4.1 Relational Database Management System (RDBMS)

A relational database management system (RDBMS) is a widely used type of database that organizes data into tables, simplifying the establishment of connections between various stored data sets. These tables can hold substantial amounts of data, ranging from hundreds to millions of rows, each referred to as records. In formal terms within the relational model, a row is known as a tuple, a column heading is called an attribute, and the table itself is referred to as a relation. A relational database encompasses multiple tables, each with a distinct name. Every row within a table represents a set of interrelated values.

Within a relational database, relationships are predefined between tables to maintain the integrity of both referential and entity relationships. A domain, labeled as 'D,' comprises atomic values, typically defined by choosing a data type from which the domain's data values are derived. Assigning a name to the domain helps in comprehending the content of its values. Each value within a relation is atomic and indivisible.

In a relational database, table relationships are established using keys, with primary keys and foreign keys being of paramount importance. These keys enable the establishment of entity integrity and referential integrity relationships. Entity integrity ensures that primary keys cannot have null values, while referential integrity guarantees that each distinct foreign key value must correspond to a matching primary key value within the same domain. Additionally, other types of keys, like super keys and candidate keys, can also be utilized.

4.4.2 Normalization

The most straightforward way to group data is to bring them together in a manner that allows for future changes with minimal impact on data structures. Data normalization is the formal process of reorganizing data structures in a way that reduces redundancy and promotes data integrity. This technique involves eliminating unnecessary fields and dividing large tables into smaller ones. It also helps prevent insertion, deletion, and update anomalies. In the realm of standard data modeling, two fundamental concepts are keys and relationships. A key is what uniquely identifies a row in a table. We have two distinct kinds of keys: the primary key, which is a component or set of components in a table used to distinguish records within that same table, and the foreign key, which is a column in a table used to uniquely identify records in other tables. All tables are normalized up to the third normal form.

Normalization, in the context of database design, is a process aimed at structuring data into appropriate tables and columns, making it user-friendly and eliminating data redundancy, which can strain computer resources. The primary steps in the normalization process encompass:

- Normalizing the data
- Selecting suitable names for tables and columns
- Choosing the right names for the data

4.4.3 Sanitization

Data sanitization refers to the process of eliminating any unlawful characters or values from data. In web applications, cleaning up user input is a common practice to safeguard against security vulnerabilities. In Python Django, involves the process of removing or filtering out any unauthorized or malicious characters or values from data. It's a critical practice in web applications built with Django to enhance security. Django provides various mechanisms and tools to sanitize and validate different forms of input, including user-generated content from forms, cookies, web services, server variables, and database queries. This practice ensures that all incoming data is clean and safe, reducing the risk of security vulnerabilities and protecting the application from harmful input.

Web applications routinely receive external input from diverse sources, including user submissions through forms, cookies, data from web services, server variables, and results from database queries. It's crucial to sanitize all external input thoroughly to ensure its safety and to preempt any potentially malicious code or content.

4.4.4 Indexing

An index is a database structure that enhances the speed of table operations. Indexes can be created on one or more columns to facilitate quick lookups and efficient ordering of records. Indexing in SQLite is a technique used to improve the retrieval speed of data from a database. It works by creating a data structure that organizes and sorts the data in the database, making it faster to search and access specific records. This index structure contains key-value pairs where the key represents a column or set of columns from a table, and the value points to the location of the corresponding data in the table. When you perform a query that involves the indexed column, SQLite can use the index to quickly locate the relevant data, significantly reducing the time and computational resources required for searching. This optimization is particularly valuable in large databases or when dealing with complex queries, as it accelerates data access and improves overall database performance.

4.5 TABLE DESIGN

1.Tbl_users_login

Primary key: **login_id**

Foreign key: **login_id** references table **Tbl_users_registration**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	login_id	int (10)	Primary Key	Login Id
2	email	varchar (20)	NOT NULL	Email
3	password	varchar (20)	NOT NULL	Password
4	role	int (10)	NOT NULL	Role of actors
5	status	int (10)	NOT NULL	Status-active/blocked

Tab 1: Tbl_users_login

2.Tbl_users_registration

Primary key: **user_id**

Foreign key: **login_id** references table **Tbl_users_ registration**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	user_id	int (10)	Primary Key	User Id
2	login_id	int (10)	Foreign Key	Login Id
3	email	varchar (20)	NULL	Email
4	password	varchar (20)	NOT NULL	Password

5	role	int (10)	NOT NULL	Role of actors
6	status	int (10)	NOT NULL	Status-active/blocked

Tab 2: Tbl_users_registration

3.Tbl_Add_Ashaworker

Primary key: **ashaworker _id**

Foreign key: **login_id** references table **Tbl_ Ashaworker**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	ashaworker_id	int (10)	Primary Key	Ashaworker Id
2	login_id	int (10)	Foreign Key	Login Id
3	Name	varchar (20)	NOT NULL	Name
4	Email	varchar (20)	NOT NULL	Email
5	Dob	Date	NOT NULL	Date of birth
6	Date_Join	Date	NOT NULL	Date of join
7	Gender	varchar (10)	NOT NULL	Gender
8	Address	varchar (20)	NOT NULL	Address
9	Taluk	varchar (20)	NOT NULL	Taluk
10	Panchayat	varchar (20)	NOT NULL	Panchayat
11	Ward	int (10)	NOT NULL	Ward number
12	Pin number	int (10)	NOT NULL	Pin number
13	Phone	int (10)	NOT NULL	Phone
14	education	varchar (20)	NOT NULL	Highest education
15	Institute name	varchar (20)	NOT NULL	Institute name
16	Year_pass	int (10)	NOT NULL	Year of pass out
17	role	int (10)	NOT NULL	Role of actors
18	status	int (10)	NOT NULL	Status-active/blocked

Tab 3: Tbl_Add_Ashaworker

4.Tbl_Ashaworker_Schedule

Primary key: **schedule _id**

Foreign key: **ashaworker_id** references table **Tbl_ Ashaworker**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	schedule_id	int (10)	Primary Key	Schedule Id
2	ashaworker _id	int (10)	Foreign Key	Ashaworker Id
3	available_days	varchar (20)	NOT NULL	Available days
4	preferred_date	Date	NOT NULL	Preferred date
5	preferred_time	DateTime	NOT NULL	Preferred time
6	role	int (10)	NOT NULL	Role of actors
7	status	int (10)	NOT NULL	Status-active/blocked

Tab 4: Tbl_Add_Ashaworker

5.Tbl_Image

Primary key: **img _id**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	img_id	int (10)	Primary Key	Image Id
2	Title	int (10)	NOT NULL	Title of the image
3	Descriprion	varchar (20)	NOT NULL	Description of the image
4	Is_enabled	int (10)	NOT NULL	Status-active/delete

Tab 5: Tbl_Image

6.Tbl_ Patient_Profile

Primary key: **patient_id**

Foreign key: **login_id** references table **Tbl_ users_login**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	patient_id	int (10)	Primary Key	Patient Id

2	login _id	int (10)	Foreign Key	Login Id
3	first_name	varchar (20)	NOT NULL	First name
4	last_name	Date	NOT NULL	Last name
5	birth_date	DateTime	NOT NULL	Birth date
6	gender	varchar (10)	NOT NULL	gender
7	house_name	varchar (20)	NOT NULL	House name
8	house_no	int (10)	NOT NULL	House number
9	address	varchar (20)	NOT NULL	Address
10	ward	int (10)	NOT NULL	Ward number
11	pin_code	int (10)	NOT NULL	Pin number
12	phone_number	int (12)	NOT NULL	phone
13	current_diagnosis	varchar (20)	NOT NULL	Current diagnosis
14	surgical_history	varchar (20)	NOT NULL	Surgical history
15	allergies	varchar (20)	NOT NULL	Allergies
16	height	int (10)	NOT NULL	Height
17	weight	int (10)	NOT NULL	Weight
18	bmi	int (10)	NOT NULL	Bmi
19	medication_names	varchar (20)	NOT NULL	Medication names
20	dosage	varchar (20)	NOT NULL	dosage
21	frequency	int (10)	NOT NULL	frequency
22	past_med_condition	varchar (20)	NOT NULL	Past medical conditions
23	role	int (10)	NOT NULL	Role of actors
24	status	int (10)	NOT NULL	Status-active/blocked

Tab 6: Tbl_ Patient_Profile

7.Tbl_Slots

Primary key: **slot _id**

Foreign key: **ashaworker_id** references table **Tbl_ Ashaworker**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	slot_id	int (10)	Primary Key	Slot Id
2	ashaworker _id	int (10)	Foreign Key	Ashaworker Id
3	Date	Date	NOT NULL	Date
4	Start_time	DateTime	NOT NULL	Start time
5	end_time	DateTime	NOT NULL	End time
6	role	int (10)	NOT NULL	Role of actors
7	status	int (10)	NOT NULL	Status-active/blocked

Tab 7: Tbl_Slots

8.Tbl_Appointment

Primary key: **appointment_id**

Foreign key: **patient_id** references table **Tbl_ Patient _Profile**

Foreign key: **ashaworker_id** references table **Tbl_ Ashaworker**

Foreign key: **slot_id** references table **Tbl_ Slots**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	appointment_id	int (10)	Primary Key	Appointment Id
2	patient_id	int (10)	Foreign Key	Patient Id
3	ashaworker_id	int (10)	Foreign Key	Ashaworker Id
4	slot_id	int (10)	Foreign Key	Slot Id
5	medical_conditions	varchar (20)	NOT NULL	Medical conditions
6	urgency	varchar (20)	NOT NULL	Urgency of appointment
7	medication_names	varchar (20)	NOT NULL	Medication names
8	symptoms	varchar (20)	NOT NULL	Symptoms of the patient

9	role	int (10)	NOT NULL	Role of actors
10	status	int (10)	NOT NULL	Status-active/blocked

Tab 8: Tbl_Slots

9.Tbl_MedicalRecord

Primary key: **record_id**

Foreign key: **patient_id** references table **Tbl_ Patient _Profile**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	record_id	int (10)	Primary Key	Medical record Id
2	patient_id	int (10)	Foreign Key	Patient Id
3	date	Date	NOT NULL	Date of medical record
4	doctor_notes	varchar (20)	NOT NULL	Doctors note
5	medications_needed	varchar (20)	NOT NULL	Medications needed
6	treatments	varchar (20)	NOT NULL	Treatments needed
7	current_conditions	varchar (20)	NOT NULL	Current conditions

Tab 9: Tbl_MedicalRecord

10.Tbl_home_visit

Primary key: **visit_id**

Foreign key: **patient_id** references table **Tbl_ Patient _Profile**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	visit_id	int (10)	Primary Key	Home visit Id
2	patient_id	int (10)	Foreign Key	Patient Id
3	date	Date	NOT NULL	Date of home visit

4	start_time	DateTime	NOT NULL	Starting time of visit
5	end_time	DateTime	NOT NULL	End time of visit

Tab 10: Tbl_home_visit

11.Tbl_Donation

Primary key: **donation_id**

Foreign key: **patient_id** references table **Tbl_ Patient _Profile**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	donation_id	int (10)	Primary Key	Donation Id
2	patient_id	int (10)	Foreign Key	Patient Id
3	amount	Date	NOT NULL	Amount donated
4	message	varchar (20)	NOT NULL	Message

Tab 11: Tbl_Donation

12.Tbl_Blog

Primary key: **donation_id**

No:	Fieldname	Datatype (Size)	Key Constraints	Description of the Field
1	blog_id	int (10)	Primary Key	Blog Id
2	author	varchar (20)	NOT NULL	Author of the blog
3	title	varchar (20)	NOT NULL	Title of the blog
4	category	varchar (20)	NOT NULL	Category of the blog
5	blog_date	Date	NOT NULL	Date of the blog created
6	description	varchar (20)	NOT NULL	Description of the blog

Tab 12: Tbl_Blog

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software testing is the process of systematically running a software program in a controlled manner to determine if it operates as intended, often employing methods of verification and validation. Validation ensures that a product complies with its specified requirements, while verification includes activities like reviews, analyses, inspections, and walkthroughs. Static analysis examines the software's source code to identify issues, while dynamic analysis assesses its behavior during runtime to collect data like execution traces, timing profiles, and test coverage details.

Testing consists of a well-planned sequence of activities that begin with testing individual components and progress to integrating the entire computer-based system. The main goals of testing are to discover errors and bugs in the software, confirm that the software performs according to its specifications, and validate that it meets performance criteria. Testing can be employed to assess correctness, implementation efficiency, and computational complexity.

A successful test is one that uncovers previously unidentified errors, and an effective test case has a high likelihood of finding such errors. Testing plays a crucial role in achieving the objectives of system testing and can utilize various techniques such as functional testing, performance testing, and security testing.

5.2 TEST PLAN

A test plan is a document that lays out the necessary steps for executing various testing methods, offering guidance on the activities involved in the testing process. Software developers are responsible for creating computer programs, documentation, and associated data structures. They must test each component of the program to ensure it aligns with its intended purpose. To address potential issues with self-evaluation, an independent test group (ITG) is often established.

Testing goals should be expressed in quantifiable terms, such as mean time to failure, the cost of identifying and fixing defects, the remaining defect density or frequency of occurrence, and test work-hours required for regression testing.

The various levels of testing encompass:

- Unit testing
- Integration testing
- Data validation testing
- Output testing

5.2.1 Unit Testing

Unit testing is a software testing approach that concentrates on validating individual components or modules within the software design. The main goal of unit testing is to assess the smallest building block of software design and ensure it functions as intended. Unit testing usually adopts a white-box perspective, allowing for the simultaneous testing of multiple components. During unit testing, the component-level design description is used as a reference to identify critical control paths and potential defects within the module's scope.

In the course of unit testing, the module's interface is scrutinized to guarantee that data is correctly processed as it enters and exits the software unit under examination. The local data structure is reviewed to ensure that data temporarily stored maintains its integrity throughout each step of an algorithm's execution. Boundary conditions are tested to ensure that all statements within a module have been executed at least once, and all error-handling paths are verified to ensure the software can handle errors effectively.

It is essential to evaluate data flow over a module interface before conducting any other testing, as without proper data input and output, all other tests become irrelevant. Another critical aspect of unit testing is the selective analysis of execution pathways to anticipate potential errors and verify that error-handling mechanisms are in place to redirect or halt processes when errors occur. Lastly, boundary testing is performed to ascertain that the software operates correctly at its operational limits.

5.2.2 Integration Testing

Integration testing is a methodical approach that involves constructing the program's framework while concurrently running tests to detect interface-related issues. The aim is to build a program structure based on the design that incorporates unit-tested components. Subsequently, the entire program undergoes testing. Correcting errors during integration testing can be quite challenging, mainly because of the program's overall size, which makes it hard to pinpoint the root causes of errors. Fixing one set of errors may give rise to new ones, creating a seemingly never-ending cycle.

Once unit testing has been completed for all modules within the system, they are integrated to uncover any inconsistencies in their interfaces. Any disparities in program structures are addressed, leading to the creation of a cohesive program structure

5.2.3 Validation Testing or System Testing

The last phase of the testing process involves assessing the entire software system in its entirety, encompassing all its forms, code, modules, and class modules. This phase is commonly known as system testing or black box testing. Black box testing primarily focuses on evaluating the software's functional requirements. Using this method, a software engineer can create input conditions that comprehensively test each program requirement. The primary types of errors that black box testing seeks to identify include incorrect or missing functions, interface issues, errors in data structure or external data access, performance problems, initialization glitches, and termination errors.

5.2.4 Output Testing or User Acceptance Testing

User acceptance testing is carried out to verify that the system aligns with the business requirements and user expectations. It's crucial to engage end users in the development process to guarantee that the software satisfies their needs. In the course of user acceptance testing, the input and output screen designs are examined using various sets of test data. Properly preparing this test data is essential for conducting a thorough evaluation of the system. Any errors discovered during testing are rectified and documented for future reference.

5.2.5 Automation Testing

Automation testing is a software testing method that employs specialized automated testing tools to run a set of test cases. Its primary objective is to confirm that the software or hardware functions precisely as intended, identifying defects, bugs, and other potential issues that may arise during the product development phase.

While certain types of testing, like manual functional or regression testing, can be carried out by humans, there are various advantages to automating the process. Automation testing can be conducted at any time, utilizing scripted sequences to assess the software. The results are documented and can be compared to previous test runs. Typically, automation developers write code in programming languages such as C#, JavaScript, or Ruby.

5.2.6 Selenium Testing

Selenium is an open-source automated testing framework employed for verifying web applications on various browsers and platforms. Selenium permits the development of test scripts in multiple programming languages, including Java, C#, and Python. Jason Huggins, an engineer at Thought Works, initially created Selenium in 2004 during his work on a web application that required frequent testing. He introduced a JavaScript program called "JavaScriptTestRunner" to automate browser actions and enhance the efficiency of testing. Over time, Selenium has continued to evolve, with contributions from a dedicated team.

In addition to Selenium, another widely-used tool for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It enables the creation of executable specifications in a human-readable format called Gherkin.

One of Cucumber's strengths is its capacity to bridge the communication gap between business stakeholders and technical teams. By using a common language, Cucumber fosters effective communication and collaboration during the testing process. It encourages a shared understanding of the requirements and helps ensure that the developed software aligns with the intended business objectives.

Cucumber can be integrated with Selenium to harness the combined benefits of both tools. Selenium is utilized for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This synergy enables the development of end-to-end tests that validate the behavior of web applications across various browsers and platforms, all presented in a business-readable and maintainable format.

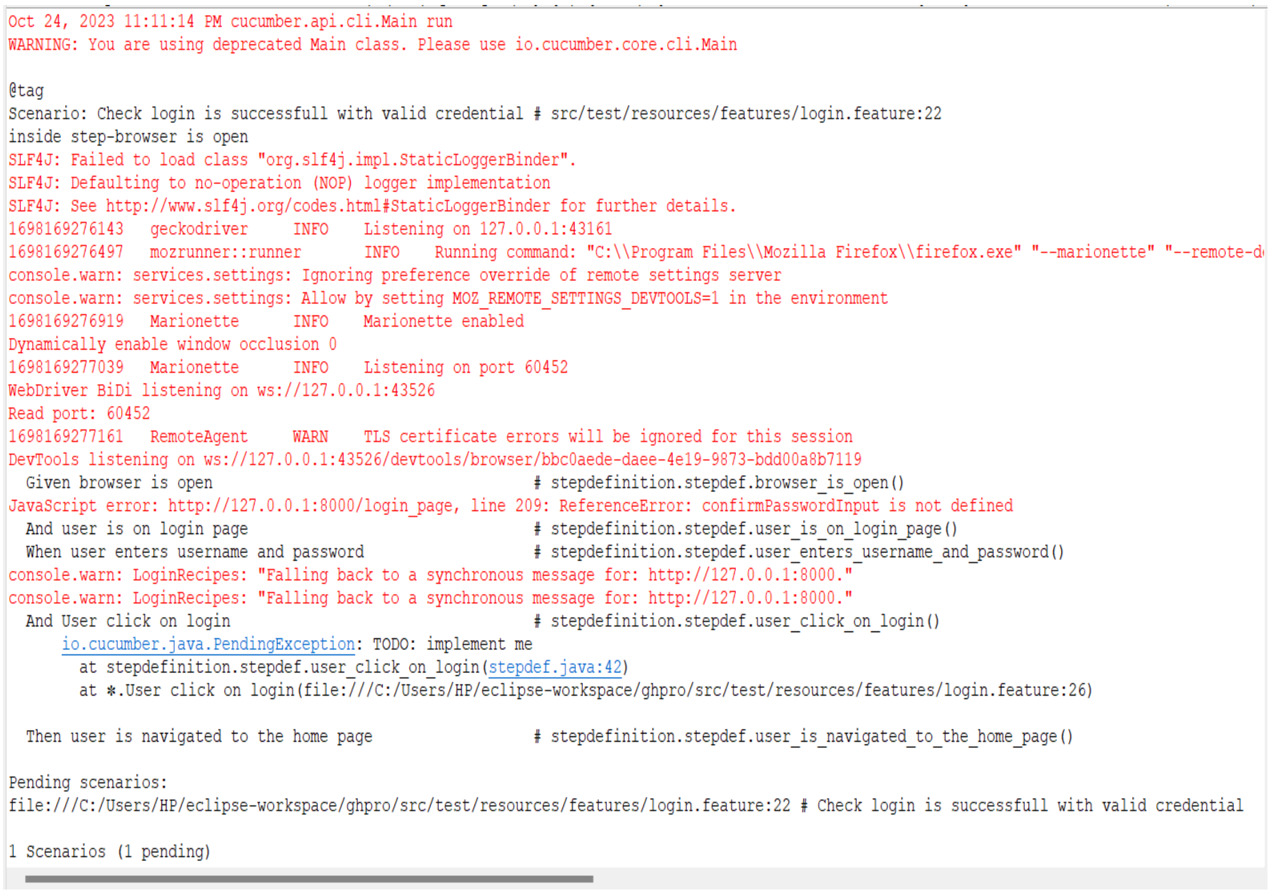
Example:

Test Case 1: Login

Code

```
package stepdefinition;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.junit.Assert;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
public class stepdef {
WebDriver driver=null;
@Given("browser is open")
public void browser_is_open() {
System.out.println("inside step-browser is open");
System.setProperty("webdriver.gecko.mariomette","D:\\cucumber_selenium\\newart
\\src\\test\\resources\\Drivers\\geckodriver.exe");
driver= new FirefoxDriver();
driver.manage().window().maximize();
}
@Given("user is on login page")
public void user_is_on_login_page() throws InterruptedException {
driver.navigate().to("http://127.0.0.1:8000/login_page");
Thread.sleep(2000);
}
@When("user enters username and password")
public void user_enters_username_and_password() {
driver.findElement(By.id("email")).sendKeys("ashna@gmail.com");
driver.findElement(By.id("password")).sendKeys("Ashna@123");
}
@When("User click on login")
public void user_click_on_login() {
driver.findElement(By.id("submit")).click();
throw new io.cucumber.java.PendingException();
}
@Then("user is navigated to the home page")
public void user_is_navigated_to_the_home_page() throws InterruptedException {
driver.findElement(By.id("logout-button-id")).isDisplayed();
Thread.sleep(4000);
driver.quit();
driver.close();
}
}
```

Screenshot



```
Oct 24, 2023 11:11:14 PM cucumber.api.cli.Main run
WARNING: You are using deprecated Main class. Please use io.cucumber.core.cli.Main

@tag
Scenario: Check login is successfull with valid credential # src/test/resources/features/login.feature:22
inside step-browser is open
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
1698169276143   geckodriver     INFO    Listening on 127.0.0.1:43161
1698169276497   mozrunner::runner       INFO    Running command: "C:\\Program Files\\Mozilla Firefox\\firefox.exe" "--marionette" "--remote-d
console.warn: services.settings: Ignoring preference override of remote settings server
console.warn: services.settings: Allow by setting MOZ_REMOTE_SETTINGS_DEVTOOLS=1 in the environment
1698169276919   Marionette      INFO    Marionette enabled
Dynamically enable window occlusion 0
1698169277039   Marionette      INFO    Listening on port 60452
WebDriver BiDi listening on ws://127.0.0.1:43526
Read port: 60452
1698169277161   RemoteAgent     WARN    TLS certificate errors will be ignored for this session
DevTools listening on ws://127.0.0.1:43526/devtools/browser/bbc0aede-daee-4e19-9873-bdd00a8b7119
  Given browser is open                                # stepdefinition.stepdef.browser_is_open()
JavaScript error: http://127.0.0.1:8000/login_page, line 209: ReferenceError: confirmPasswordInput is not defined
  And user is on login page                            # stepdefinition.stepdef.user_is_on_login_page()
  When user enters username and password               # stepdefinition.stepdef.user_enters_username_and_password()
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
  And User click on login                              # stepdefinition.stepdef.user_click_on_login()
      io.cucumber.java.PendingException: TODO: implement me
        at stepdefinition.stepdef.user_click_on_login(stepdef.java:42)
        at ✽.User click on login(file:///C:/Users/HP/eclipse-workspace/ghpro/src/test/resources/features/login.feature:26)

  Then user is navigated to the home page              # stepdefinition.stepdef.user_is_navigated_to_the_home_page()

Pending scenarios:
file:///C:/Users/HP/eclipse-workspace/ghpro/src/test/resources/features/login.feature:22 # Check login is successfull with valid credential

1 Scenarios (1 pending)
```

Test Report

Test Case 1

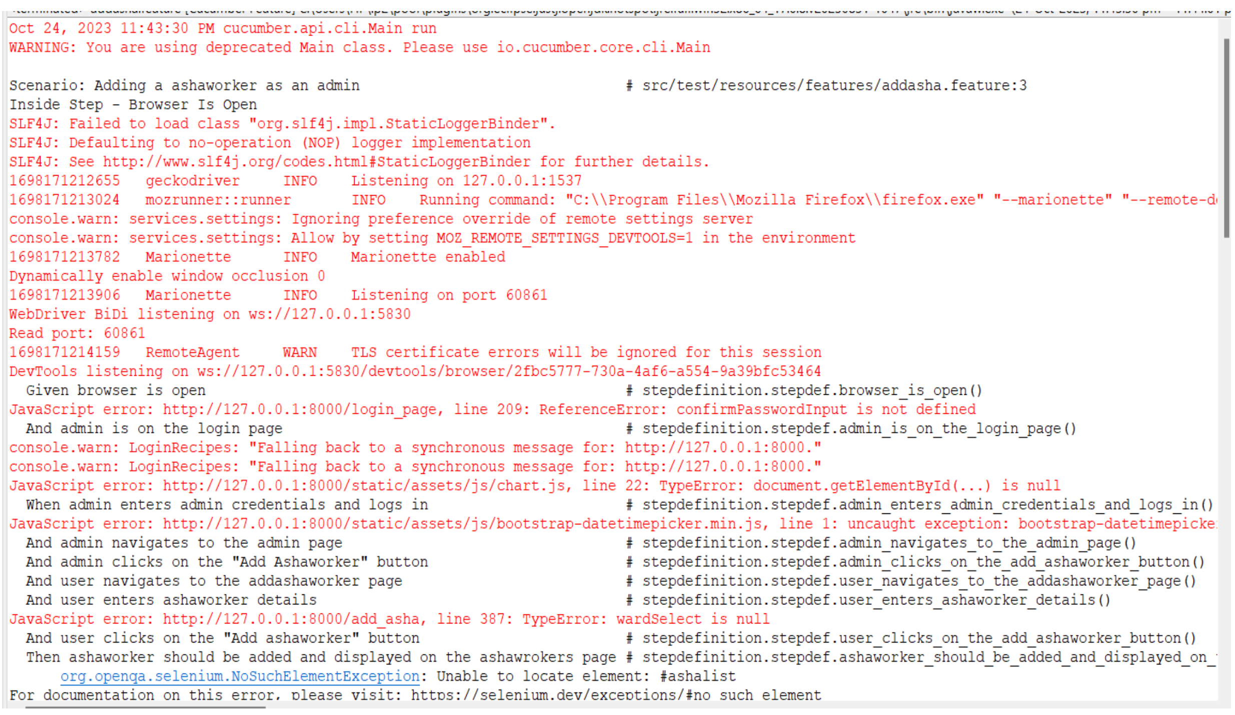
Project Name: Gentle Haloes					
Login Test Case					
Test Case ID: Test_1			**Test Designed By:** Maria Jacob		
Test Priority (Low/Medium/High): High			**Test Designed Date:** 25/09/2023		
Module Name: Login Screen			**Test Executed By:** Mr. Binumon Joseph		
Test Title: Login Page			**Test Execution Date:** 25/09/2023		
Description: Verify login with valid email and password					
Pre-Condition: User has valid email and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/ Fail)**
1	Navigation to Login Page		Dashboard should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: ashna@gmail.com	User should be able to Login	User Logged in and navigated to Index Page with records	Pass
	Provide Valid Password	Password: Ashna@123			
3	Click on Login button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database					

Test Case 2: Add Ashaworker

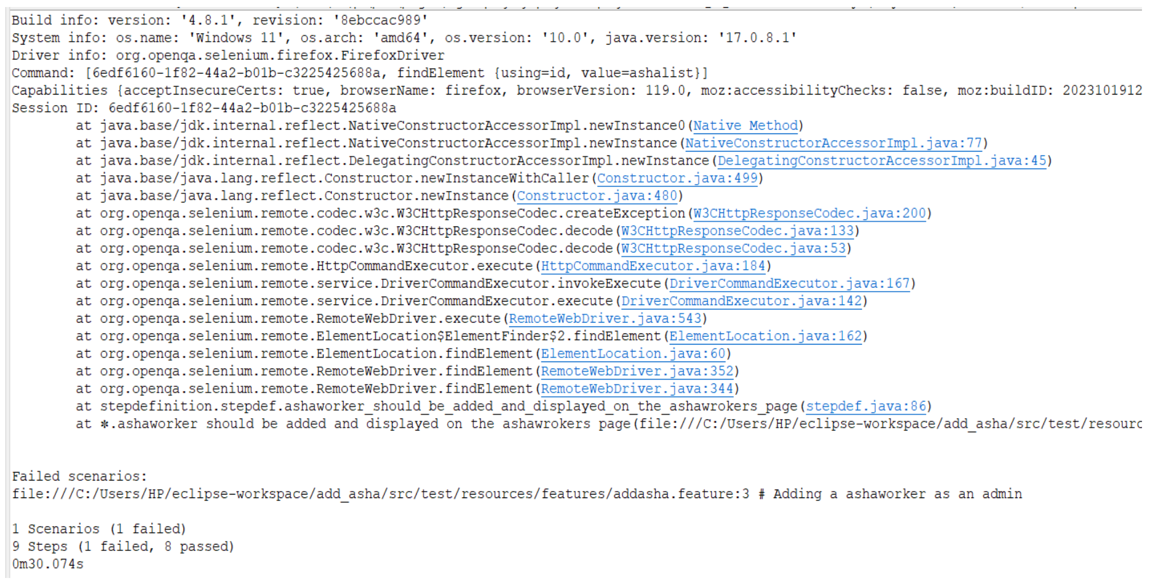
Code

```
package stepdefinition;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
//import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
public class stepdef {
WebDriver driver = null;
@Given("browser is open")
public void browser_is_open() {
System.out.println("Inside Step - Browser Is Open");
System.setProperty("webdriver.gecko.marionette",       "C:\\Users\\HP\\eclipse-
workspace\\add_asha\\src\\test\\resources\\Drivers\\geckodriver.exe");
driver = new FirefoxDriver();
driver.manage().window().maximize();
}
@And("admin is on the login page")
public void admin_is_on_the_login_page() throws Exception{
driver.navigate().to("http://127.0.0.1:8000/login_page");
Thread.sleep(3000);
}
@When("admin enters admin credentials and logs in")
public void admin_enters_admin_credentials_and_logs_in()throws Throwable {
driver.findElement(By.name("email")).sendKeys("admin123@gmail.com");
driver.findElement(By.name("password")).sendKeys("Admin@123");
driver.findElement(By.id("login")).click();
Thread.sleep(3000);
}
@And("admin navigates to the admin page")
public void admin_navigates_to_the_admin_page() throws Exception{
Thread.sleep(3000);
driver.findElement(By.id("ashaind")).click();
}
@And("admin clicks on the \"Add Ashaworker\" button")
public      void      admin_clicks_on_the_add_ashaworker_button()      throws
InterruptedException {
driver.findElement(By.id("addasha")).click();
Thread.sleep(3000);
}
@And("user navigates to the addashaworker page")
public      void      user_navigates_to_the_addashaworker_page()      throws
InterruptedException {
Thread.sleep(3000);
}
@And("user enters ashaworker details")
public void user_enters_ashaworker_details() throws InterruptedException{
// Enter category details
driver.findElement(By.name("name")).sendKeys("Niya Thomas");
driver.findElement(By.name("email")).sendKeys("niya@gmail.com");
driver.findElement(By.name("password")).sendKeys("Niya@123");
driver.findElement(By.name("dob")).sendKeys("14/10/2023");
driver.findElement(By.name("doj")).sendKeys("09/10/2023");
driver.findElement(By.name("gender")).sendKeys("Female");
driver.findElement(By.name("address")).sendKeys("Kply");
driver.findElement(By.name("taluk")).sendKeys("Kanjirappally");
driver.findElement(By.name("panchayat")).sendKeys("Parathodu");
driver.findElement(By.name("pin")).sendKeys("686518");
driver.findElement(By.name("phone")).sendKeys("9876545678");
// Add any other details related to category creation
}
@And("user clicks on the \"Add ashaworker\" button")
public void user_clicks_on_the_add_ashaworker_button() {
driver.findElement(By.id("createasha")).click();
}
@Then("ashaworker should be added and displayed on the ashawrokers page")public
void  ashaworker_should_be_added_and_displayed_on_the_ashawrokers_page()throws
InterruptedException {
driver.findElement(By.id("ashalist")).isDisplayed();
Thread.sleep(2000);
driver.close();
driver.quit();
}
}
```

Screenshot



```
Oct 24, 2023 11:43:30 PM cucumber.api.cli.Main run
WARNING: You are using deprecated Main class. Please use io.cucumber.core.cli.Main

Scenario: Adding a ashaworker as an admin                          # src/test/resources/features/addasha.feature:3
Inside Step - Browser Is Open
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
1698171212655   geckodriver     INFO    Listening on 127.0.0.1:1537
1698171213024   mozrunner::runner       INFO    Running command: "C:\\Program Files\\Mozilla Firefox\\firefox.exe" "--marionette" "--remote-d
console.warn: services.settings: Ignoring preference override of remote settings server
console.warn: services.settings: Allow by setting MOZ_REMOTE_SETTINGS_DEVTOOLS=1 in the environment
1698171213782   Marionette      INFO    Marionette enabled
Dynamically enable window occlusion 0
1698171213906   Marionette      INFO    Listening on port 60861
WebDriver BiDi listening on ws://127.0.0.1:5830
Read port: 60861
1698171214159   RemoteAgent     WARN    TLS certificate errors will be ignored for this session
DevTools listening on ws://127.0.0.1:5830/devtools/browser/2fbc5777-730a-4af6-a554-9a39bfc53464
  Given browser is open                                            # stepdefinition.stepdef.browser_is_open()
JavaScript error: http://127.0.0.1:8000/login_page, line 209: ReferenceError: confirmPasswordInput is not defined
  And admin is on the login page                                   # stepdefinition.stepdef.admin_is_on_the_login_page()
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
JavaScript error: http://127.0.0.1:8000/static/assets/js/chart.js, line 22: TypeError: document.getElementById(...) is null
  When admin enters admin credentials and logs in                  # stepdefinition.stepdef.admin_enters_admin_credentials_and_logs_in()
JavaScript error: http://127.0.0.1:8000/static/assets/js/bootstrap-datetimepicker.min.js, line 1: uncaught exception: bootstrap-datetimepicke
  And admin navigates to the admin page                            # stepdefinition.stepdef.admin_navigates_to_the_admin_page()
  And admin clicks on the "Add Ashaworker" button                  # stepdefinition.stepdef.admin_clicks_on_the_add_ashaworker_button()
  And user navigates to the addashaworker page                     # stepdefinition.stepdef.user_navigates_to_the_addashaworker_page()
  And user enters ashaworker details                               # stepdefinition.stepdef.user_enters_ashaworker_details()
JavaScript error: http://127.0.0.1:8000/add_asha, line 387: TypeError: wardSelect is null
  And user clicks on the "Add ashaworker" button                   # stepdefinition.stepdef.user_clicks_on_the_add_ashaworker_button()
  Then ashaworker should be added and displayed on the ashawrokers page # stepdefinition.stepdef.ashaworker_should_be_added_and_displayed_on_
      org.openqa.selenium.NoSuchElementException: Unable to locate element: #ashalist
For documentation on this error, please visit: https://selenium.dev/exceptions/#no_such_element
```



```
Build info: version: '4.8.1', revision: '8ebccac989'
System info: os.name: 'Windows 11', os.arch: 'amd64', os.version: '10.0', java.version: '17.0.8.1'
Driver info: org.openqa.selenium.firefox.FirefoxDriver
Command: [6edf6160-1f82-44a2-b01b-c3225425688a, findElement {using=id, value=ashalist}]
Capabilities {acceptInsecureCerts: true, browserName: firefox, browserVersion: 119.0, moz:accessibilityChecks: false, moz:buildID: 2023101912
Session ID: 6edf6160-1f82-44a2-b01b-c3225425688a
	at java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
	at java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:77)
	at java.base/jdk.internal.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
	at java.base/java.lang.reflect.Constructor.newInstanceWithCaller(Constructor.java:499)
	at java.base/java.lang.reflect.Constructor.newInstance(Constructor.java:480)
	at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.createException(W3CHttpResponseCodec.java:200)
	at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.decode(W3CHttpResponseCodec.java:133)
	at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.decode(W3CHttpResponseCodec.java:53)
	at org.openqa.selenium.remote.HttpCommandExecutor.execute(HttpCommandExecutor.java:184)
	at org.openqa.selenium.remote.service.DriverCommandExecutor.invokeExecute(DriverCommandExecutor.java:167)
	at org.openqa.selenium.remote.service.DriverCommandExecutor.execute(DriverCommandExecutor.java:142)
	at org.openqa.selenium.remote.RemoteWebDriver.execute(RemoteWebDriver.java:543)
	at org.openqa.selenium.remote.ElementLocation$ElementFinder$2.findElement(ElementLocation.java:162)
	at org.openqa.selenium.remote.ElementLocation.findElement(ElementLocation.java:60)
	at org.openqa.selenium.remote.RemoteWebDriver.findElement(RemoteWebDriver.java:352)
	at org.openqa.selenium.remote.RemoteWebDriver.findElement(RemoteWebDriver.java:344)
	at stepdefinition.stepdef.ashaworker_should_be_added_and_displayed_on_the_ashawrokers_page(stepdef.java:86)
	at ✽.ashaworker should be added and displayed on the ashawrokers page(file:///C:/Users/HP/eclipse-workspace/add_asha/src/test/resourc


Failed scenarios:
file:///C:/Users/HP/eclipse-workspace/add_asha/src/test/resources/features/addasha.feature:3 # Adding a ashaworker as an admin

1 Scenarios (1 failed)
9 Steps (1 failed, 8 passed)
0m30.074s
```

Test report

Test Case 2

Project Name: Gentle Haloes	
Add Ashaworker Test Case	
Test Case ID: Test_2	**Test Designed By:** Maria Jacob
Test Priority (Low/Medium/High): High	**Test Designed Date:** 25/09/2023
Module Name: Add ashaworker Screen	**Test Executed By:** Mr. Binumon Joseph
Test Title: Add Ashaworker Page	**Test Execution Date:** 25/09/2023
Description: Verify login with valid email and password then navigates to the Ashaworkers page and add ashaworker	
Pre-Condition: Admin has valid email and password	

Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Dashboard should be displayed	Login page displayed	Pass
	Provide Valid	Email:		User Logged in	Pass

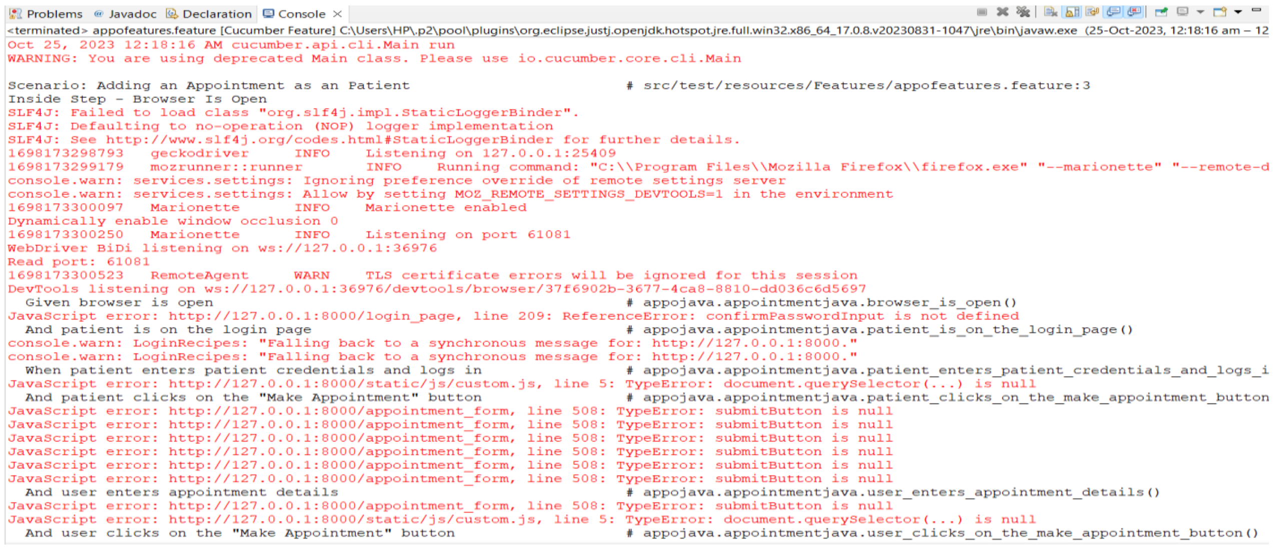
2	Email	admin@gmail.com	User should be able to Login	and navigated to Index Page with records	
	Provide Valid Password	Password: Admin@123			
3	Click on Login button				
4	Click on add Ashaworker		Add ashaworker page should be displayed	Admin navigated to add Ashaworker page	pass
5	Provide Asha workers Details	Name: Niya Email: niya@gmail.com			
6	Click on Create Ashaworker Button		Admin should be able to add new ashaworker	Admin created new ashaworker	Pass
7	Navigation to ashaworker display page		Admin should be able to view ashaworkers list	Admin navigated to ashaworker page and view new ashaworkers added	Pass
Post-Condition: New ashaworker is successfully added by admin					

Test Case 3: Appointment

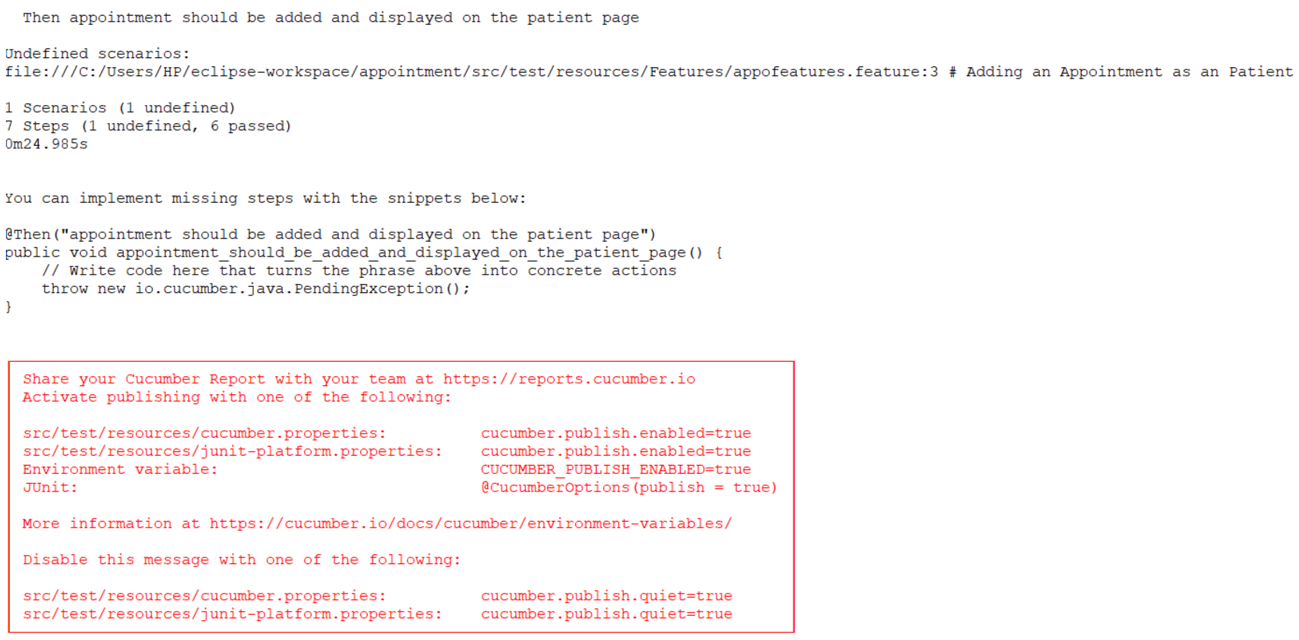
Code

```
package appojava;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
public class appointmentjava {
WebDriver driver = null;
@Given("browser is open")
public void browser_is_open() {
System.out.println("Inside Step - Browser Is Open");
System.setProperty("webdriver.gecko.marionette",
"C:\\Users\\HP\\eclipseworkspace\\appointment\\src\\test\\resources\\
Drivers\\geckodriver.ee");
driver = new FirefoxDriver();
driver.manage().window().maximize();
}
@And("patient is on the login page")
public void patient_is_on_the_login_page() throws Exception{
driver.navigate().to("http://127.0.0.1:8000/login_page");
Thread.sleep(3000);
}
@When("patient enters patient credentials and logs in")
public   void   patient_enters_patient_credentials_and_logs_in()throws
Throwable {
driver.findElement(By.name("email")).sendKeys("mariajacob2024@mca.ajc
e.in");
driver.findElement(By.name("password")).sendKeys("Mariya@123");
driver.findElement(By.id("login")).click();
Thread.sleep(3000);
} Exception{
//
//        Thread.sleep(3000);
//        driver.findElement(By.id("makeapp")).click();
//
//
/}
@And("patient clicks on the \"Make Appointment\" button")
public  void  patient_clicks_on_the_make_appointment_button()  throws
InterruptedException {
driver.findElement(By.id("makeapp")).click();
Thread.sleep(3000);
}
@And("user navigates to the addappointment page")
public   void   user_navigates_to_the_addappointment_page()   throws
InterruptedException {
Thread.sleep(3000);
}
@And("user enters appointment details")
public      void      user_enters_appointment_details()      throws
InterruptedException{
// Enter category details
driver.findElement(By.name("urgency")).sendKeys("Non-urgent   medical
check-up");
driver.findElement(By.name("medical_conditions")).sendKeys("ok");
driver.findElement(By.name("symptoms")).sendKeys("nill");
driver.findElement(By.name("ashaworker")).sendKeys("jiminnn:Ward   1-
VENGATHANAM");
driver.findElement(By.name("date")).sendKeys("2023-10-04");
driver.findElement(By.name("time")).sendKeys("01:24 AM-02:24 AM");
// Add any other details related to category creation
}
@And("user clicks on the \"Make Appointment\" button")
public void user_clicks_on_the_make_appointment_button() {
driver.findElement(By.id("schedule")).click();
}
@Then("ashaworker should be added and displayed on the ashawrokers
page")
public                                                           void
ashaworker_should_be_added_and_displayed_on_the_ashawrokers_page()thr
ows InterruptedException {
driver.findElement(By.id("viewbtn")).isDisplayed();
Thread.sleep(2000);
driver.close();
driver.quit();
}}
```

Screenshot



```
<terminated> appofeatures.feature [Cucumber Feature] C:\Users\HP\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (25-Oct-2023, 12:18:16 am – 12
Oct 25, 2023 12:18:16 AM cucumber.api.cli.Main run
WARNING: You are using deprecated Main class. Please use io.cucumber.core.cli.Main

Scenario: Adding an Appointment as an Patient                       # src/test/resources/Features/appofeatures.feature:3
Inside Step - Browser Is Open
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
1698173298793   geckodriver     INFO    Listening on 127.0.0.1:25409
1698173299179   mozrunner::runner       INFO    Running command: "C:\\Program Files\\Mozilla Firefox\\firefox.exe" "--marionette" "--remote-d
console.warn: services.settings: Ignoring preference override of remote settings server
console.warn: services.settings: Allow by setting MOZ_REMOTE_SETTINGS_DEVTOOLS=1 in the environment
1698173300097   Marionette      INFO    Marionette enabled
Dynamically enable window occlusion 0
1698173300250   Marionette      INFO    Listening on port 61081
WebDriver BiDi listening on ws://127.0.0.1:36976
Read port: 61081
1698173300523   RemoteAgent     WARN    TLS certificate errors will be ignored for this session
DevTools listening on ws://127.0.0.1:36976/devtools/browser/37f6902b-3677-4ca8-8810-dd036c6d5697
  Given browser is open                                             # appojava.appointmentjava.browser_is_open()
JavaScript error: http://127.0.0.1:8000/login_page, line 209: ReferenceError: confirmPasswordInput is not defined
  And patient is on the login page                                  # appojava.appointmentjava.patient_is_on_the_login_page()
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
  When patient enters patient credentials and logs in               # appojava.appointmentjava.patient_enters_patient_credentials_and_logs_i
JavaScript error: http://127.0.0.1:8000/static/js/custom.js, line 5: TypeError: document.querySelector(...) is null
  And patient clicks on the "Make Appointment" button               # appojava.appointmentjava.patient_clicks_on_the_make_appointment_button
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
  And user enters appointment details                               # appojava.appointmentjava.user_enters_appointment_details()
JavaScript error: http://127.0.0.1:8000/appointment_form, line 508: TypeError: submitButton is null
JavaScript error: http://127.0.0.1:8000/static/js/custom.js, line 5: TypeError: document.querySelector(...) is null
  And user clicks on the "Make Appointment" button                  # appojava.appointmentjava.user_clicks_on_the_make_appointment_button()
```



```
  Then appointment should be added and displayed on the patient page

Undefined scenarios:
file:///C:/Users/HP/eclipse-workspace/appointment/src/test/resources/Features/appofeatures.feature:3 # Adding an Appointment as an Patient

1 Scenarios (1 undefined)
7 Steps (1 undefined, 6 passed)
0m24.985s


You can implement missing steps with the snippets below:

@Then("appointment should be added and displayed on the patient page")
public void appointment_should_be_added_and_displayed_on_the_patient_page() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}


 Share your Cucumber Report with your team at https://reports.cucumber.io
 Activate publishing with one of the following:

 src/test/resources/cucumber.properties:          cucumber.publish.enabled=true
 src/test/resources/junit-platform.properties:    cucumber.publish.enabled=true
 Environment variable:                            CUCUMBER_PUBLISH_ENABLED=true
 JUnit:                                           @CucumberOptions(publish = true)

 More information at https://cucumber.io/docs/cucumber/environment-variables/

 Disable this message with one of the following:

 src/test/resources/cucumber.properties:          cucumber.publish.quiet=true
 src/test/resources/junit-platform.properties:    cucumber.publish.quiet=true
```

Test report

Test Case 3	
Project Name: Gentle Haloes	
Appointment Test Case	
Test Case ID: Test_3	**Test Designed By:** Maria Jacob
Test Priority (Low/Medium/High): High	**Test Designed Date:** 25/09/2023
Module Name: Schedule appointment Screen	**Test Executed By:** Mr. Binumon Joseph

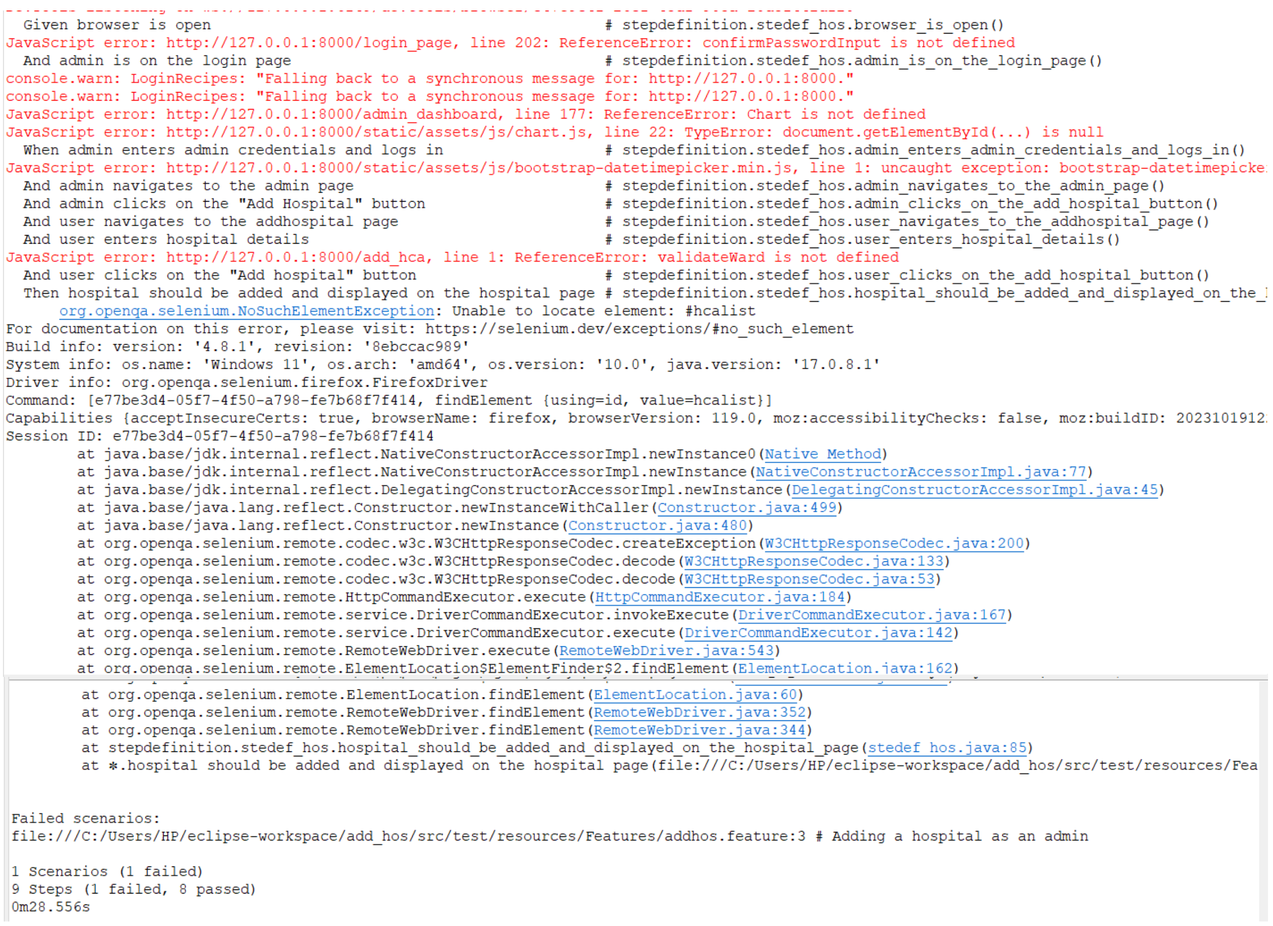
Test Title: Schedule Appointment Page			**Test Execution Date:** 25/09/2023		
Description: Verify login with valid email and password then navigates to the Appointment page and schedule appointment					
Pre-Condition: Patient has valid email and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Dashboard should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: anna@gmail.com	User should be able to Login	User Logged in and navigated to Index Page with records	Pass
	Provide Valid Password	Password: Anna@123			
3	Click on Login button				
4	Click on Schedule Appointment		Appointment page should be displayed	Patient navigated to schedule appointment page	pass
5	Provide Appointment Details	Name: Anna Date:25/-9/2023 Time: 10.00 am			
6	Click on Schedule Appointment Button		Patient should be able to schedule appointment	Patient schedules new appointment	Pass
7	Navigation to appointment display page		Patient should be able to view appointment list	Patient navigated to appointment page and view new appointments schedules	Pass
Post-Condition: New appointment is successfully scheduled by patient					

Test Case 4: Add Hospital

Code

```
package stepdefinition;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
public class stedef_hos {
WebDriver driver = null;
@Given("browser is open")
public void browser_is_open() {
System.out.println("Inside Step - Browser Is Open");
System.setProperty("webdriver.gecko.marionette", "C:\\Users\\HP\\eclipse-
workspace\\add_hos\\src\\test\\resources\\Drivers\\geckodriver.exe");
driver = new FirefoxDriver();
driver.manage().window().maximize();
}
@And("admin is on the login page")
public void admin_is_on_the_login_page() throws Exception{
driver.navigate().to("http://127.0.0.1:8000/login_page");
Thread.sleep(3000);
}
@When("admin enters admin credentials and logs in")
public void admin_enters_admin_credentials_and_logs_in()throws Throwable {
driver.findElement(By.name("email")).sendKeys("admin123@gmail.com");
driver.findElement(By.name("password")).sendKeys("Admin@123");
driver.findElement(By.id("login")).click();
Thread.sleep(3000);
}
@And("admin navigates to the admin page")
public void admin_navigates_to_the_admin_page() throws Exception{

Thread.sleep(3000);
driver.findElement(By.id("hosind")).click();

}
@And("admin clicks on the \"Add Hospital\" button")
public void admin_clicks_on_the_add_hospital_button() throws
InterruptedException {
driver.findElement(By.id("addhos")).click();
Thread.sleep(3000);
}
@And("user navigates to the addhospital page")
public void user_navigates_to_the_addhospital_page() throws
InterruptedException {
Thread.sleep(3000);

}
@And("user enters hospital details")
public void user_enters_hospital_details() throws InterruptedException{
// Enter category details
driver.findElement(By.name("hcaname")).sendKeys("Primary Health Center");
driver.findElement(By.name("email")).sendKeys("primaryhealthcenter@gmail.com")
;
driver.findElement(By.name("password")).sendKeys("Primary@123");
driver.findElement(By.name("year_hca")).sendKeys("14/10/2022");
driver.findElement(By.name("date_of_join")).sendKeys("09/10/2023");
driver.findElement(By.name("address")).sendKeys("Kply");
driver.findElement(By.name("taluk")).sendKeys("Kanjirappally");
driver.findElement(By.name("panchayat")).sendKeys("Parathodu");

driver.findElement(By.name("postal")).sendKeys("686519");
driver.findElement(By.name("phone")).sendKeys("9877657898");
// Add any other details related to category creation
}
@And("user clicks on the \"Add hospital\" button")
public void user_clicks_on_the_add_hospital_button() {
driver.findElement(By.id("createhos")).click();
}
@Then("hospital should be added and displayed on the hospital page")
public void
hospital_should_be_added_and_displayed_on_the_hospital_page()throws
InterruptedException {
driver.findElement(By.id("hcalist")).isDisplayed();

Thread.sleep(2000);

driver.close();

driver.quit();
}
}
```

Screenshot



```
  Given browser is open                                    # stepdefinition.stedef_hos.browser_is_open()
JavaScript error: http://127.0.0.1:8000/login_page, line 202: ReferenceError: confirmPasswordInput is not defined
  And admin is on the login page                           # stepdefinition.stedef_hos.admin_is_on_the_login_page()
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
console.warn: LoginRecipes: "Falling back to a synchronous message for: http://127.0.0.1:8000."
JavaScript error: http://127.0.0.1:8000/admin_dashboard, line 177: ReferenceError: Chart is not defined
JavaScript error: http://127.0.0.1:8000/static/assets/js/chart.js, line 22: TypeError: document.getElementById(...) is null
  When admin enters admin credentials and logs in          # stepdefinition.stedef_hos.admin_enters_admin_credentials_and_logs_in()
JavaScript error: http://127.0.0.1:8000/static/assets/js/bootstrap-datetimepicker.min.js, line 1: uncaught exception: bootstrap-datetimepicke
  And admin navigates to the admin page                    # stepdefinition.stedef_hos.admin_navigates_to_the_admin_page()
  And admin clicks on the "Add Hospital" button            # stepdefinition.stedef_hos.admin_clicks_on_the_add_hospital_button()
  And user navigates to the addhospital page               # stepdefinition.stedef_hos.user_navigates_to_the_addhospital_page()
  And user enters hospital details                         # stepdefinition.stedef_hos.user_enters_hospital_details()
JavaScript error: http://127.0.0.1:8000/add_hca, line 1: ReferenceError: validateWard is not defined
  And user clicks on the "Add hospital" button             # stepdefinition.stedef_hos.user_clicks_on_the_add_hospital_button()
  Then hospital should be added and displayed on the hospital page # stepdefinition.stedef_hos.hospital_should_be_added_and_displayed_on_the_
      org.openqa.selenium.NoSuchElementException: Unable to locate element: #hcalist
For documentation on this error, please visit: https://selenium.dev/exceptions/#no_such_element
Build info: version: '4.8.1', revision: '8ebccac989'
System info: os.name: 'Windows 11', os.arch: 'amd64', os.version: '10.0', java.version: '17.0.8.1'
Driver info: org.openqa.selenium.firefox.FirefoxDriver
Command: [e77be3d4-05f7-4f50-a798-fe7b68f7f414, findElement {using=id, value=hcalist}]
Capabilities {acceptInsecureCerts: true, browserName: firefox, browserVersion: 119.0, moz:accessibilityChecks: false, moz:buildID: 2023101912
Session ID: e77be3d4-05f7-4f50-a798-fe7b68f7f414
        at java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
        at java.base/jdk.internal.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:77)
        at java.base/jdk.internal.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
        at java.base/java.lang.reflect.Constructor.newInstanceWithCaller(Constructor.java:499)
        at java.base/java.lang.reflect.Constructor.newInstance(Constructor.java:480)
        at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.createException(W3CHttpResponseCodec.java:200)
        at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.decode(W3CHttpResponseCodec.java:133)
        at org.openqa.selenium.remote.codec.w3c.W3CHttpResponseCodec.decode(W3CHttpResponseCodec.java:53)
        at org.openqa.selenium.remote.HttpCommandExecutor.execute(HttpCommandExecutor.java:184)
        at org.openqa.selenium.remote.service.DriverCommandExecutor.invokeExecute(DriverCommandExecutor.java:167)
        at org.openqa.selenium.remote.service.DriverCommandExecutor.execute(DriverCommandExecutor.java:142)
        at org.openqa.selenium.remote.RemoteWebDriver.execute(RemoteWebDriver.java:543)
        at org.openqa.selenium.remote.ElementLocation$ElementFinder$2.findElement(ElementLocation.java:162)
        at org.openqa.selenium.remote.ElementLocation.findElement(ElementLocation.java:60)
        at org.openqa.selenium.remote.RemoteWebDriver.findElement(RemoteWebDriver.java:352)
        at org.openqa.selenium.remote.RemoteWebDriver.findElement(RemoteWebDriver.java:344)
        at stepdefinition.stedef_hos.hospital_should_be_added_and_displayed_on_the_hospital_page(stedef_hos.java:85)
        at *.hospital should be added and displayed on the hospital page(file:///C:/Users/HP/eclipse-workspace/add_hos/src/test/resources/Fea


Failed scenarios:
file:///C:/Users/HP/eclipse-workspace/add_hos/src/test/resources/Features/addhos.feature:3 # Adding a hospital as an admin

1 Scenarios (1 failed)
9 Steps (1 failed, 8 passed)
0m28.556s
```

Test report

Test Case 4

Project Name: Gentle Haloes					
Add Hospital Test Case					
Test Case ID: Test_4			**Test Designed By:** Maria Jacob		
Test Priority (Low/Medium/High): High			**Test Designed Date:** 25/09/2023		
Module Name: Add hospital Screen			**Test Executed By:** Mr. Binumon Joseph		
Test Title: Add Hospital Page			**Test Execution Date:** 25/09/2023		
Description: Verify login with valid email and password then navigates to the hospital page and add hospital					
Pre-Condition: Admin has valid email and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Dashboard should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: admin@gmail.com	User should be able to Login	User Logged in and navigated to Index Page with records	Pass
	Provide Valid Password	Password: Admin@123			
3	Click on Login button				
4	Click on add Hospital		Add hospital page should be displayed	Admin navigated to add Hospital page	pass
5	Provide Hospital Details	Name: Primary Health Center Email:			

		primaryhealthcenter@gmail.com			
6	Click on Create Hospital Button		Admin should be able to add new hospital	Admin created new hospital	Pass
7	Navigation to Hospital display page		Admin should be able to view hospital list	Admin navigated to hospital page and view new hospital added	Pass
Post-Condition: New hospital is successfully added by admin					

CHAPTER 6
IMPLEMENTATION

6.1 INTRODUCTION

The implementation phase of a project is where the design is turned into a functional system. It's a critical stage in ensuring the success of the new system, as it involves gaining user confidence in its effective and accurate operation. During this phase, user training and documentation are essential priorities. Conversion might occur simultaneously with user training or at a later stage. Implementation is the process of transforming a newly designed system into an operational one.

In this stage, the user department carries the primary workload, goes through significant changes, and exerts the most substantial influence on the existing system. Poorly planned or uncontrolled implementation can lead to confusion and disorder. Whether the new system is entirely new, replaces an existing manual or automated system, or modifies an existing system, a well-executed implementation is crucial to meet the organization's requirements. System implementation encompasses all the activities necessary to transition from the old to the new system. It can only happen after thorough testing confirms that it works as per the specifications. System personnel assess the system's feasibility. Implementation demands extensive effort in three primary areas: education and training, system testing, and changeover. The implementation phase involves meticulous planning, examining system constraints, and designing methods for achieving the changeover.

6.2 IMPLEMENTATION PROCEDURES

Software implementation involves the installation of software in its designated environment to confirm that it meets its intended purpose and functions as anticipated. In certain organizations, the software development project may be initiated by individuals who won't be the end users of the software. Initially, there might be some uncertainty about the software, but it's crucial to prevent any resistance from emerging. This can be accomplished by:

- Making sure that active users understand the advantages of the new system, thereby increasing their trust in the software.
- Offering appropriate guidance to users to ensure they feel at ease when using the application.

Before using the system, it's important for users to be informed that the server program needs to be operational on the server. Without the server object being up and running, the intended processes will not be able to proceed.

6.2.1 User Training

The purpose of user training is to equip individuals with the knowledge and skills necessary for testing and transitioning to the new system. To realize the anticipated benefits and objectives of a computer-based system, it's crucial for those involved to feel confident in their roles within the new system. As the system grows in complexity, the significance of training becomes even more pronounced. Through user training, individuals learn how to input data, address error messages, interact with the database, access routines for generating reports, and carry out other essential functions.

6.2.2 Training on the Application Software

After providing the fundamental training for computer literacy, it is crucial to deliver training on the new application software to the users. This training should encompass the fundamental principles guiding the usage of the new system, such as the sequence of screens, screen layout, the available on-screen assistance, potential data entry errors, and the corresponding validation procedures for each input, as well as methods for rectifying entered data. Additionally, the training should incorporate user-specific or group-specific information that is essential for effective utilization of the entire system or specific components. It's worth noting that this training may vary among different user groups and hierarchical levels

6.2.3 System Maintenance

The maintenance phase holds significant importance within the software development cycle, as it marks the period when the software is actively used and fulfills its intended functions. Effective maintenance is vital to ensure that the system remains operational, dependable, and adaptable to changes in its operating environment. Maintenance activities encompass more than just identifying and rectifying errors or defects; they may entail software updates, functional adjustments, and performance enhancements, among other tasks. Essentially, software maintenance is a continuous and ongoing process that demands constant monitoring, assessment, and enhancement of the system to align with evolving user needs and demands

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

Gentle Haloes Palliative Care Website" project represents a significant step forward in the provision of palliative care services at the panchayat level. By creating a dynamic and interactive online platform, the project aims to improve the quality of life for individuals with serious illnesses and their families within the panchayat. This comprehensive website serves as a valuable resource, fostering awareness, communication, and collaboration among users while facilitating access to essential palliative care services.

The project incorporates various modules to cater to the needs of different user groups. The Admin Module ensures efficient management of the platform, with features such as patient and volunteer management, content control, and appointment scheduling. The User Module enables registered users to access their patient profiles, medical records, and vital resources, making it a central hub for palliative care information. The Volunteer and Health Care Assistant Modules provide specific tools for managing volunteer shifts, patients' conditions, and appointments, streamlining the delivery of care.

By combining these modules, the project creates a comprehensive ecosystem that empowers healthcare providers, volunteers, and patients alike. This holistic approach ensures that palliative care services are not only accessible but also efficiently managed, with a focus on improving the quality of life for patients and their families. The website's educational resources and user-friendly interface enhance awareness and understanding of palliative care, ultimately contributing to a better quality of life for those in need.

In essence, the "Gentle Haloes Palliative Care Website" project strives to bridge the gap in palliative care services, offering a user-friendly, informative, and collaborative platform. It's a testament to the power of technology in improving the lives of individuals facing serious illnesses and their families, and it underscores the importance of community-based initiatives in healthcare. The successful implementation of this project promises to bring comfort, support, and improved quality of life to many within the panchayat, further highlighting the importance of palliative care in our society

7.2 FUTURE SCOPE

The "Panchayat Level Palliative Care Website" project holds immense potential for the future. As it stands, the project offers a robust foundation for the delivery of palliative care services at the panchayat level, significantly improving the quality of life for individuals facing serious illnesses and their families within the panchayat.

Looking ahead, the project could expand to cover additional regions or countries, with localization and customization to suit specific healthcare needs. Developing a mobile application alongside the website would enhance accessibility, providing features like telemedicine consultations and prescription management. Integration with Electronic Health Records (EHR) systems can further streamline patient care, while the inclusion of machine learning and AI can optimize appointment scheduling and symptom management.

Building a community aspect into the platform, along with support groups and forums, would offer emotional support and valuable resources. Research opportunities, fundraising, and partnerships can ensure sustained growth and development. Education and training, addressing legal and ethical considerations, and public awareness campaigns are essential elements for the project's future success.

With continuous innovation and a commitment to improving the quality of life for patients, the "Panchayat Level Palliative Care Website" project can make a lasting impact on palliative care services.

CHAPTER 8
BIBLIOGRAPHY

REFERENCES:

- https://www.painandpalliativecarethrissur.org/
- https://getpalliativecare.org/
- https://www.palliativecare.in/

WEBSITES:

- https://chat.openai.com/
- www.w3schools.com
- https://docs.djangoproject.com/en/4.2/topics/install/

CHAPTER 9
APPENDIX

9.1 Sample Code

1. Login

- **HTML Page**

```
{% load static %}
{% load socialaccount %}
<!DOCTYPE html>
<html>
<head>
<title>LOGIN</title>
<!-- Add Bootstrap CSS link -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
<style>
h1{
margin-top: 10px;
text-align: center;
color: rgb(0, 0, 0);
font-size: 30px;
color:blue;
}
.container {
padding: 10px;
border-radius: 10px;
position: absolute;
top: 10%;
left: 10%;
transform: translate(-10%, -10%);
}
.image-container {
background-size: cover;
min-height: 100vh;
display: flex;
align-items: center;
justify-content: center;
}
.form-container {
background-color: rgba(231, 171, 171, 0.845);
padding-right: 50px;
padding-left: 50px;
border-radius: 30px;
max-width: 550px;
padding-bottom: 30px;
}
.image-container img {
max-width: 100%;
height: auto;
display: block;
}
.btn-primary {
width: 100%;
height: 50px;
background-color: blue;
border-color: black;
color:white;
font-family: 'Times New Roman', Times, serif;
}
.btn-primary:hover {
background-color: white;
border-color: darkgreen;
color:black;
}
.btn-primary1 {
width: 100%;
height: 50px;
background-color: black;
border-color: black;
color: black;
font-family: 'Times New Roman', Times, serif;
font-size: 20px;
position: relative;
padding-left: 40px; /* Adjust as needed */
background-image: url('{% static 'images/google.png' %}');
background-size: 24px; /* Adjust the size of the Google image */
background-repeat: no-repeat;
background-position: 10px center; /* Adjust the position of the Google image */
}
.btn-primary1:hover {
background-color:white;
border-color: darkgreen;
color:white;
}
.form-control {
height: 50px; /* Set the desired height */
width: 100%; /* Set the width to span the container */
}
.form-group label {
font-size:20px;
font-family: 'Times New Roman', Times, serif;
}
pre {
font-size: 18px; /* Set the desired font size */
color: purple; /* Set the desired text color */
/* Add any other styling you want for the <pre> tag */
}
pre a {
font-size: 18px; /* Set the desired font size */
color: red; /* Set the desired text color */
/* Add any other styling you want for the link inside <pre> */
}
.line{
position: relative;
height: 1px;
width: 100%;
margin: 36px 0;
background-color: #d4d4d4;
}
.line::before{
content: 'Or';
position: absolute;
top: 50%;
left: 50%;
transform: translate(-50%, -50%);
background-color: #FFF;
color: #8b8b8b;
padding: 0 15px;
}
</style>
</head>
<body>
<div class="container-fluid">
```

```
<div class="row">
<div class="col-md-6 image-container">
<img src="https://media.istockphoto.com/id/1184108166/vector/nursing-home-hospital-hospice-
concept.jpg?s=612x612&w=0&k=20&c=5BMqSuGN2oUVLEeA9J-v8HC8n74U9uoImMQAHlZ3BFI=">
</div>
<div class="col-md-6">
<div class="container form-container mt-5">
<h1>Login Form</h1>
{% for messages in messages %}
<h3 style="color: red">{{ messages }}</h3>
{% endfor %}
<form action="#" method="post" id="registrationForm" autocomplete="off">
{% csrf_token %}
<div class="form-group">
<label for="email">Email:</label>
<input type="text" class="form-control" id="email" name="email" required>
<small id="emailError" class="form-text text-danger"></small>
</div>
<div class="form-group">
<label for="password">Password:</label>
<input type="password" class="form-control" id="password" name="password" required>
<small id="passwordError" class="form-text text-danger"></small>
</div>
<br><button type="submit" id="login" class="btn btn-primary" id="submit">LOGIN</button>
<pre>Don't have an account? <a href="{% url 'register' %}" >Signup</a></pre>
</form>
<div>
<form action="{% url 'password_reset' %}" method="post">
{% csrf_token %}
<div class="form-group">
<a href="{% url 'password_reset' %}" class="forgot-password">Forgot Password?</a>
</div>
</form>
<div class="line"></div>
</button>
<button type="submit" class="btn btn-primary1">
<a href="{% provider_login_url 'google' %}?next=/">
<span class="google-button-content">
Sign in with Google
</span>
</button>
</div>
</div>
</div>
</div>
<!-- Add Bootstrap JS and jQuery scripts -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.3/dist/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
<script>
const registrationForm = document.getElementById('registrationForm');
const emailInput = document.getElementById('email');
{% comment %} const phoneNumberInput = document.getElementById('phoneNumber'); {% endcomment %}
const passwordInput = document.getElementById('password');
function validateEmail() {
const emailValue = emailInput.value;
const emailError = document.getElementById('emailError');
if (!/^\S+@\S+(\.\S+)?$/.test(emailValue)) {
emailError.textContent = 'Enter a valid email address with an optional period (".").';
return false;
} else {
```

```
emailError.textContent = '';
return true;
}
}
function validatePassword() {
const passwordValue = passwordInput.value;
const passwordError = document.getElementById('passwordError');
if (!/(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?=.*\W).{8,}/.test(passwordValue)) {
passwordError.textContent = 'Password should contain at least 8 characters, one uppercase letter, one lowercase letter, one number, and one special character.';
return false;
} else {
passwordError.textContent = '';
return true;
}
}
emailInput.addEventListener('keyup', validateEmail);
passwordInput.addEventListener('keyup', validatePassword);
confirmPasswordInput.addEventListener('keyup', validateConfirmPassword);
registrationForm.addEventListener('submit', function(event) {
// const isUsernameValid = validateUsername();
const isEmailValid = validateEmail();
// const isPhoneNumberValid = validatePhoneNumber();
const isPasswordValid = validatePassword();
const isConfirmPasswordValid = validateConfirmPassword();

if (!isEmailValid ||  !isPasswordValid || !isConfirmPasswordValid) {
event.preventDefault();
}
});
</script>
</body></html>
```

2. Add Ashaworker

• HTML Page

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<!-- add-doctor24:06-->
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=0">
<link rel="shortcut icon" type="image/x-icon" href="{% static 'assets/img/favicon.ico'%}">
<title>GentleHaloes Admin Dashboard</title>
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/bootstrap.min.css'%}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/font-awesome.min.css'%}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/select2.min.css'%}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/bootstrap-datetimepicker.min.css'%}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/style.css'%}">
<script src="{% static 'assets/js/html5shiv.min.js' %}"></script>
<script src="{% static 'assets/js/respond.min.js' %}"></script>
</head>
<body>
<div class="main-wrapper">
<div class="header">
<div class="header-left">
<a href="{% url 'admin_dashboard' %}" class="logo">
<img src="{% static 'images/ghbc1.png' %}" width="100" height="55" alt=""><span>GentleHaloes</span>
```

```
</a>
</div>
<a id="toggle_btn" href="javascript:void(0);"><i class="fa fa-bars"></i></a>
<a id="mobile_btn" class="mobile_btn float-left" href="#sidebar"><i class="fa fa-bars"></i></a>
<ul class="nav user-menu float-right">
<li class="nav-item dropdown has-arrow">
<a href="#" class="dropdown-toggle nav-link user-link" data-toggle="dropdown">
<i class="fa fa-user" aria-hidden="true"></i>
<span>{{ user.email }}</span>
</a>
{% if user.is_authenticated %}
<div class="dropdown-menu">
<a class="dropdown-item" href="{% url 'loggout' %}">Logout</a>
</div>
{% else %}
<div class="dropdown-menu">
<a class="dropdown-item" href="{% url 'login_page' %}">Login</a>
</div>
{% endif %}
</li>
</ul>
<div class="dropdown mobile-user-menu float-right">
<a href="#" class="nav-link dropdown-toggle" data-toggle="dropdown" aria-expanded="false"><i class="fa fa-
ellipsis-v"></i></a>
<div class="dropdown-menu dropdown-menu-right">
<a class="dropdown-item" href="profile.html">My Profile</a>
<a class="dropdown-item" href="edit-profile.html">Edit Profile</a>
<a class="dropdown-item" href="settings.html">Settings</a>
<a class="dropdown-item" href="login.html">Logout</a>
</div>
</div>
</div>
<div class="sidebar" id="sidebar">
<div class="sidebar-inner slimscroll">
<div id="sidebar-menu" class="sidebar-menu">
<ul>
<li class="menu-title">Main</li>
<li class="active">
<a href="{% url 'admin_dashboard' %}"><i class="fa fa-dashboard"></i> <span>Dashboard</span></a>
</li>
<li >
<a href="{% url 'ad_ashaworker' %}"><i class="fa fa-user"></i> <span>Asha Workers List </span></a>  </li>
<li>
<a href="{% url 'patients' %}"><i class="fa fa-wheelchair"></i> <span>Patients</span></a>
</li>
<li class="submenu">
<a href="#"><i class="fa fa-user"></i> <span>Appointments </span> <span class="menu-arrow"></span></a>
<ul style="display: none;">
                        <li><a href="{% url 'appointments' %}">All Appointments</a></li>
<li><a href="{% url 'current_appointment' %}">Current Appointments</a></li>
<li><a href="{% url 'future_appointment' %}">Future Appointments</a></li>
<li><a href="{% url 'past_appointment' %}">Past Appointments</a></li>
</ul>
</li>
<li>
<a href="{% url 'schedule' %}"><i class="fa fa-calendar-check-o"></i> <span>Asha Workers Schedule</span></a>
</li>

{% comment %} <li>
<a href="{% url 'adgallery' %}"><i class="fa fa-calendar-check-o"></i> <span>Gallery</span></a>
</li>  {% endcomment %}
```

```
<li class="submenu">
<a href="#"><i class="fa fa-user"></i> <span> Gallery </span> <span class="menu-arrow"></span></a>
<ul style="display: none;">

<li><a href="{% url 'adgallery' %}">Add Image</a></li>
<li><a href="{% url 'dis_gallery' %}">Edit Image</a></li>

</ul>
</li>
<li class="submenu">
<a href="#"><i class="fa fa-commenting-o"></i> <span> Blog</span> <span class="menu-arrow"></span></a>
<ul style="display: none;">
<li><a href="{% url 'blog' %}">Blog View</a></li>
<li><a href="{% url 'add-blog' %}">Add Blog</a></li>
</ul>
</li>
</ul>
</div>
</div>
</div>
<div class="page-wrapper">
<div class="content">
<div class="row">
<div class="col-lg-8 offset-lg-2">
<h4 class="page-title">Add Asha Workers</h4>
</div>
</div>
<div class="row">
<div class="col-lg-8 offset-lg-2">
{% comment %} <form method="POST" action="" autocomplete="off" enctype="multipart/form-data"> {%
endcomment %}
<form method="POST" action="" autocomplete="off" enctype="multipart/form-data" onsubmit="return
validateWard()">
{% csrf_token %}
<div class="row">
{% comment %} <div class="col-sm-6">
<div class="form-group">
<label>Name <span class="text-danger">*</span></label>
<input class="form-control" id="name" name="name" type="text">
</div>
</div> {% endcomment %}
<div class="col-sm-6">
<div class="form-group">
<label>Name <span class="text-danger">*</span></label>
<input class="form-control" id="name" name="name" type="text">
<span class="error" id="name-error"></span> <!-- Display validation error message here -->
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label>Email <span class="text-danger">*</span></label>
<input class="form-control" id="email" name="email" type="email">
<span class="text-danger" id="email-error"></span>
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label>Password</label>
<input class="form-control" id="pass" name="password" type="password">
<span class="text-danger" id="password-error"></span>
</div>
```

```
</div>

<div class="col-sm-6">
<div class="form-group">
<label>Date of Birth</label>
<div class="cal-icon">
<input type="text" id="dob" name="dob" class="form-control datetimepicker">
</div>
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label>Date of Join</label>
<div class="cal-icon">
<input type="text" id="doj" name="doj" class="form-control datetimepicker">
</div>
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label for="gender">Gender:</label>
<select class="form-control" id="gender" name="gender">
<option value="" disabled selected>Select Gender</option>
<option value="male">Male</option>
<option value="female">Female</option>
</select>
</div>
</div>

<div class="col-sm-12">
<div class="row">
<div class="col-sm-12">
<div class="form-group">
<label>Address</label>
<input type="text" name="address" id="address" class="form-control ">
</div>
</div>
<div class="col-sm-6 col-md-6 col-lg-3">
<div class="form-group">
<label>Taluk</label>
{% comment %} <select class="form-control select" name="taluk" id="taluk">
<option></option>
<option>Kanjirappally</option>
</select> {% endcomment %}
<input class="form-control" value="Kanjirappally" id="taluk" name="taluk" type="text" readonly >
</div>
</div>
<div class="col-sm-6 col-md-6 col-lg-3">
<div class="form-group">
<label>Panchayat</label>
{% comment %} <select class="form-control select" name="panchayat" id="panchayat">
<option></option>
<option>Parathodu</option>
</select> {% endcomment %}
<input class="form-control" value="Parathodu" id="panchayat" name="panchayat" type="text" readonly>
</div></div>
<div class="col-sm-6 col-md-6 col-lg-3">
<div class="form-group">
<label>Ward</label>
<select class="form-control select" name="ward" id="ward" size="100">
<option>Ward 1-VENGATHANAM</option>
```

```
<option>Ward 2-PALAPRA</option>
<option>Ward 3-VELICHIYANI</option>
<option>Ward 4-CHOTTY</option>
<option>Ward 5-MANGAPPARA</option>
<option>Ward 6-VADAKKEMALA</option>
<option>Ward 7-PARATHODU</option>
<option>Ward 8-NADUKANI</option>
<option>Ward 9-EDAKKUNNAM</option>
<option>Ward 10-KOORAMTHOOKKU</option>
<option>Ward 11-KOOVAPPALLY</option>
<option>Ward 12-KULAPPURAM</option>
<option>Ward 13-PALAMPRA</option>
<option>Ward 14-MUKKALY</option>
<option>Ward 15-PODIMATTAM</option>
<option>Ward 16-ANAKKALLU</option>
<option>Ward 17-PULKUNNU</option>
<option>Ward 18-PAZHUMALA</option>
<option>Ward 19-CHIRABHAGAM</option>
</select>
<div id="ward-validation-message"></div>
</div>
</div>
<div class="col-sm-6 col-md-6 col-lg-3">
<div class="form-group">
<label>Postal Code</label>
<input type="number" class="form-control" name="pin" id="pin">
<span class="text-danger" id="pin-error"></span>
</div>
</div>
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label>Phone </label>
<input class="form-control" type="number" name="phone" id="phone">
<span class="text-danger" id="phone-error"></span>
</div>
</div>
<div class="col-sm-6">
<div class="form-group">
<label>Profile Photo</label>
<input type="file" class="form-control" name="profile_photo">
</div>
</div>
</div>
<div>
<div class="m-t-20 text-center">
<button  class="btn btn-primary submit-btn" id="createasha">Create Asha Worker</button>
</div>
</div>
</form>
</div>
</div>
</div>
</div>
</div>
<div class="sidebar-overlay" data-reff=""></div>
<script src="{% static 'assets/js/jquery-3.2.1.min.js' %}"></script>
<script src="{% static 'assets/js/popper.min.js' %}"></script>
<script src="{% static 'assets/js/bootstrap.min.js' %}"></script>
<script src="{% static 'assets/js/jquery.slimscroll.js' %}"></script>
```

```
<script src="{% static 'assets/js/select2.min.js' %}"></script>
<script src="{% static 'assets/js/moment.min.js' %}"></script>
<script src="{% static 'assets/js/bootstrap-datetimepicker.min.js' %}"></script>
<script src="{% static 'assets/js/app.js' %}"></script>
<style>
.header {
background-color: #00c6a9;
left: 0;
position: fixed;
right: 0;
top: 0;
z-index: 1039;
height: 50px;
box-shadow: 0 1px 1px rgba(0, 0, 0, 0.1);
}
</style>
<script>
$(document).ready(function () {
// Add keyup event listeners to input fields
$("#name").keyup(function () {
validateName();
});

// Other input fields' keyup event listeners here

// Function to validate Name
function validateName() {
var name = $("#name").val();
var nameError = $("#name-error");

// Regular expression to allow only alphabetic characters
var alphabeticRegex = /^[A-Za-z ]+$/;

if (name.length === 0) {
nameError.text("Name is required.");
} else if (!alphabeticRegex.test(name)) {
nameError.text("Name should contain only alphabetic characters.");
} else {
nameError.text(""); // Clear the error message
}
}
// Other validation functions for input fields here
});
$(document).ready(function () {
// Add keyup event listener to the email input field
$("#email").keyup(function () {
validateEmail();
});
// Function to validate Email
function validateEmail() {
var email = $("#email").val();
var emailError = $("#email-error");
// Regular expression for basic email validation
var emailRegex = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;
if (email.length === 0) {
emailError.text("Email is required.");
} else if (!emailRegex.test(email)) {
emailError.text("Invalid email address.");
} else {
emailError.text(""); // Clear the error message
}
```

```
}
});
$(document).ready(function () {
// Add keyup event listener to the password input field
$("#pass").keyup(function () {
validatePassword();
});
// Function to validate Password
function validatePassword() {
var password = $("#pass").val();
var passwordError = $("#password-error");
// Regular expressions for password validation
var capitalRegex = /[A-Z]/;
var specialCharRegex = /[!@#$%^&*()_+{}\[\]:;<>,.?~\\-]/;
var numberRegex = /[0-9]/;
// Check minimum and maximum length
if (password.length < 6 || password.length > 20) {
passwordError.text("Password must be between 6 and 10 characters.");
} else if (!capitalRegex.test(password)) {
passwordError.text("Password must contain at least one capital letter.");
} else if (!specialCharRegex.test(password)) {
passwordError.text("Password must contain at least one special character.");
} else if (!numberRegex.test(password)) {
passwordError.text("Password must contain at least one number.");
} else {
passwordError.text(""); // Clear the error message
}
}
});

$(document).ready(function () {
// Add keyup event listener to the postal code input field
$("#pin").keyup(function () {
validatePostalCode();
});
// Function to validate Postal Code
function validatePostalCode() {
var postalCode = $("#pin").val();
var pinError = $("#pin-error");
// Regular expression to match exactly 6 digits
var regex = /^\d{6}$/;
if (!regex.test(postalCode)) {
pinError.text("Postal code must contain exactly 6 digits.");
} else {
pinError.text(""); // Clear the error message
}
}
});
$(document).ready(function () {
// Add keyup event listener to the phone number input field
$("#phone").keyup(function () {
validatePhoneNumber();
});
// Function to validate Phone Number
function validatePhoneNumber() {
var phoneNumber = $("#phone").val();
var phoneError = $("#phone-error");
// Regular expression to match exactly 10 digits
var regex = /^\d{10}$/;
if (!regex.test(phoneNumber)) {
phoneError.text("Phone number must contain exactly 10 digits.");
} else {
phoneError.text(""); // Clear the error message
}
}
});
</script>
<script>
function validateWard() {
const wardSelect = document.getElementById("ward");
const selectedWard = wardSelect.value
fetch(`/check-ward-exists/?ward=${selectedWard}`)
.then(response => response.json())
.then(data => {
if (data.exists) {
// Ward already assigned, show a popup alert
        alert("This ward is already assigned to another Ashaworker. Please select a different  ward.");
} else {
// Ward is available, continue with form submission
document.querySelector("form").submit();
}   })
.catch(error => {
console.error("Error checking ward existence:", error);
});
// Prevent the form from submitting immediately
return false;
}
</script>
</body>
<!-- add-doctor24:06-->
</html>
```

3. Medical Record

- **HTML Page**

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=0">
<link rel="shortcut icon" type="image/x-icon" href="assets/img/favicon.ico">
<title>GentleHaloes Admin Dashboard</title>
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/bootstrap.min.css' %}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/font-awesome.min.css' %}">
<link rel="stylesheet" type="text/css" href="{% static 'assets/css/style.css' %}">
<!--[if lt IE 9]>
<script src="assets/js/html5shiv.min.js"></script>
<script src="assets/js/respond.min.js"></script>
<![endif]-->
</head>
<body>
<div class="main-wrapper">
<div class="header">
<div class="header-left">
<a href="{% url 'index' %}" class="logo">
<img src="{% static 'images/ghbc1.png' %}" width="100" height="55" alt=""><span>GentleHaloes</span>
</a>
</div>
```

```
<a id="toggle_btn" href="javascript:void(0);"><i class="fa fa-bars"></i></a>
<a id="mobile_btn" class="mobile_btn float-left" href="#sidebar"><i class="fa fa-bars"></i></a>
<ul class="nav user-menu float-right">

<li class="nav-item dropdown has-arrow">
<a href="#" class="dropdown-toggle nav-link user-link" data-toggle="dropdown">
<i class="fa fa-user" aria-hidden="true"></i>
<span>{{ user.email }}</span>
</a>
{% if user.is_authenticated %}
<div class="dropdown-menu">
{% comment %} <a class="dropdown-item" href="{% url 'edit_asha_profile' %}">Profile</a> {% endcomment %}
<a class="dropdown-item" href="{% url 'pro_ashaworker' %}">Profile</a>
<a class="dropdown-item" href="{% url 'loggout' %}">Logout</a>
</div>
{% else %}
<div class="dropdown-menu">
<a class="dropdown-item" href="{% url 'login_page' %}">Login</a>
</div>
{% endif %}
</li>
</div>
<div class="sidebar" id="sidebar">
<div class="sidebar-inner slimscroll">
<div id="sidebar-menu" class="sidebar-menu">
<ul>
<li class="menu-title">Main</li>
<li class="active">
<a href="{% url 'asha_index' %}"><i class="fa fa-dashboard"></i> <span>Dashboard</span></a>
</li>
<li>
<a href="{% url 'asha_approved_appo' %}"><i class="fa fa-calendar"></i> <span>Appointments</span></a>
</li>
<li>
<a href="{% url 'asha_timeslots'%}"><i class="fa fa-calendar"></i> <span>Schedules</span></a>
</li>
<li>
<a href="{% url 'asha_timeslots_shows'%}"><i class="fa fa-calendar"></i> <span>View Schedules</span></a>
</li>
{% comment %} <li>
<a href="{% url 'patient_users' %}"><i class="fa fa-wheelchair"></i> <span>Patients List</span></a>
</li> {% endcomment %}
<li class="submenu">
<a href="#"><i class="fa fa-user"></i> <span>Patients </span> <span class="menu-arrow"></span></a>
<ul style="display: none;">
<li><a href="{% url 'patient_users' %}">All Patients List</a></li>
<li><a href="{% url 'patients_by_ward' %}"> Patients By Ward</a></li>
<li><a href="{% url 'add_view_rec' %}"> Medical Record</a></li>
</ul>
</li>
</ul>
</div>
</div>
</div>
<section class="contact_section layout_padding-bottom">
<div class="container">
<br><br><br>
<div class="heading_container">
<h2>
Medical Record
</h2>
```

```
</div>
<div class="row">
<div class="col-md-7">
<div class="form_container">
<form method="POST" action="{% url 'medical_record' patient.id %}">
{% csrf_token %}
<div>
<input type="email" id="email" name="email" class="text-input" value="{{ patient.email }}" placeholder="Email"
required >
</div>
<div>
<input type="text" id="first_name" name="first_name" class="text-input" value="{{ patient.first_name }}"
placeholder="First name" required >
</div>
<div>
<input type="text" id="last_name" name="last_name" class="text-input" value="{{ patient.last_name }}"
placeholder="Last name" required >
</div>
<div>
<input type="date" id="date" name="date" class="text-input" placeholder="Date" required>
</div>
<div>
<textarea id="doctor_notes" name="doctor_notes" rows="4" class="text-input" placeholder="Doctor's Notes"
required></textarea>
</div>
<div>
<textarea id="medications_needed" class="text-input" name="medications_needed" rows="4"
placeholder="Medications Needed" required></textarea>
</div><div>
<textarea id="treatments" class="text-input" name="treatments" rows="4" placeholder="Treatments"
required></textarea>
</div>
<div>
<textarea id="current_conditions" class="text-input" name="current_conditions" rows="4" placeholder="Current
Conditions" required></textarea>
</div>
<div class="btn_box">
<button type="submit">Save</button>
</div>
</form>
</div>
</div>
</div>
</div>
</section>
<div class="sidebar-overlay" data-reff=""></div>
<script src="{% static 'assets/js/jquery-3.2.1.min.js' %}"></script>
<script src="{% static 'assets/js/popper.min.js' %}"></script>
<script src="{% static 'assets/js/bootstrap.min.js' %}"></script>
<script src="{% static 'assets/js/jquery.slimscroll.js' %}"></script>
<script src="{% static 'assets/js/Chart.bundle.js' %}"></script>
<script src="{% static 'assets/js/chart.js' %}"></script>
<script src="{% static 'assets/js/app.js' %}"></script>
<style>
/* Style the text input fields */
.text-input {
width: 100%;
padding: 10px;
margin-bottom: 10px;
border: 1px solid #00c6a9;
border-radius: 5px;
font-size: 16px;
}
/* Style the placeholder text for all text input fields */
.text-input::placeholder {
color: #00c6a9;
}
/* Style the placeholder text specifically for date input */
.text-input[type="date"]::placeholder {
color: #00c6a9; /* Set the placeholder color to #00c6a9 for date input */
}
/* Style for the button */
.btn_box button {
background-color: #00c6a9; /* Blue background color */
color: #fff; /* White text color */
padding: 10px 20px; /* Padding for height and width */
border-color:black; /* Remove border */
cursor: pointer; /* Add a pointer cursor on hover */
border-radius: 5px; /* Rounded corners */
height:50px;
width:200px;
margin-left:200px;
}
/* Hover effect */
.btn_box button:hover {
background-color: black; /* Darker blue on hover */
}
h2{
color:black;
text-decoration:bold;
}
.container{
margin-left:700px;
}</style>
</body>
</html>
```

4. Appointment

- **HTML Page**

```
{% load static%}
<!DOCTYPE html>
<html>
<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />
<title>Mico</title>
<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="{% static 'css/bootstrap.css' %}" />
<!-- fonts style -->
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;700;900&display=swap"
rel="stylesheet">
<!--owl slider stylesheet -->
```

```
<link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.carousel.min.css" />
<!-- font awesome style -->
<link href="{% static 'css/font-awesome.min.css' %}" rel="stylesheet" />
<!-- nice select -->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/jquery-nice-select/1.1.0/css/nice-select.min.css"
integrity="sha256-mLBIhmBvigTFWPSCtvdu6a76T+3Xyt+K571hupeFLg4=" crossorigin="anonymous" />
<!-- datepicker -->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-datepicker/1.3.0/css/datepicker.css">
<!-- Custom styles for this template -->
<link href="{% static 'css/style.css' %}" rel="stylesheet" />
<!-- responsive style -->
<link href="{% static 'css/responsive.css' %}" rel="stylesheet" />
</head>
<body class="sub_page">
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="header_top">
<div class="container">
<div class="contact_nav">
<a href="">
<i class="fa fa-phone" aria-hidden="true"></i>
<span>
Call : +01 123455678990
</span>
</a>
<a href="">
<i class="fa fa-envelope" aria-hidden="true"></i>
<span>
Email : gentlehaloes@gmail.com
</span>
</a>
<a href="">
<i class="fa fa-map-marker" aria-hidden="true"></i>
<span>
Location
</span>
</a>
</div>
</div>
</div>
<div class="header_bottom">
<div class="container-fluid">
<nav class="navbar navbar-expand-lg custom_nav-container ">
<a class="navbar-brand" href="{% url 'index' %}">
<img src="{% static 'images/ghbc1.png' %}" alt="" style="width:120px; height: 80px;">
</a>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
<span class=""> </span>
</button>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<div class="d-flex mr-auto flex-column flex-lg-row align-items-center">
<ul class="navbar-nav">
<li class="nav-item active">
<a class="nav-link" href="{% url 'index' %}">Home <span class="sr-only">(current)</span></a>
</li>
<li class="nav-item">
<a class="nav-link" href="{% url 'about' %}"> About</a>
```

```
</li>
<li class="nav-item dropdown"> <!-- Add 'dropdown' class to create a dropdown item -->
<a class="nav-link dropdown-toggle" href="#" id="galleryDropdown" role="button" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
Blog
</a>
<div class="dropdown-menu" aria-labelledby="galleryDropdown">
<!-- Add your dropdown menu items here -->
<a class="dropdown-item" href="{% url 'resources' %}">Resource</a>
<a class="dropdown-item" href="{% url 'gallery' %}">Gallery</a>
<!-- Add more items as needed -->
</div>
</li>
<li class="nav-item">
<a class="nav-link" href="{% url 'testimonial' %}">Review</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{% url 'contact' %}">Contact</a>
</li>
</ul>
</div>
<div class="quote_btn-container">
<div class="quote_btn-container">
{% if user.is_authenticated %}
<div class="dropdown">
<button class="dropdown-toggle" type="button" id="loggedInDropdown" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
<i class="fa fa-user" aria-hidden="true"></i>
<span>{{ user.email }}</span>
</button>
<div class="dropdown-menu" aria-labelledby="loggedInDropdown">
<a class="dropdown-item" href="{% url 'print_patient_profile' %}">
<i class="fa fa-sign-out" aria-hidden="true"></i>
<span>Profile</span>
</a>
<a class="dropdown-item" href="{% url 'appointment_form' %}">
<i class="fa fa-sign-out" aria-hidden="true"></i>
<span>Make Appointments</span>
</a>
<a class="dropdown-item" href="{% url 'appointment_view' %}">
<i class="fa fa-sign-out" aria-hidden="true"></i>
<span>View Appointments</span>
</a>
<a class="dropdown-item" href="{% url 'loggout' %}">
<i class="fa fa-sign-out" aria-hidden="true"></i>
<span>Logout</span>
</a>
</div>
</div>
{% else %}
{% comment %} <div class="dropdown">
<button class="dropdown-toggle" type="button" id="loginDropdown" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Login</span>
</button>
<div class="dropdown-menu" aria-labelledby="loginDropdown">
<a class="dropdown-item" href="{% url 'login_page' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Patient Login</span>
</a>
```

```
<a class="dropdown-item" href="{% url 'login_page' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>AshaWorker Login</span>
</a>
<a class="dropdown-item" href="{% url 'login_page' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Hospital Login</span>
</a>
</div>
</div> {% endcomment %}
{% endif %}
</div>

<div class="quote_btn-container">
<div class="quote_btn-container">
{% if user.is_authenticated %}
{% comment %} <div class="dropdown">
<button class="dropdown-toggle" type="button" id="loggedInDropdown" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Logged in as {{ user.username }}</span>
</button> {% endcomment %}
<div class="dropdown-menu" aria-labelledby="loggedInDropdown">
<a class="dropdown-item" href="{% url 'loggout' %}">
<i class="fa fa-sign-out" aria-hidden="true"></i>
<span>Logout</span>
</a>
</div>
</div>
{% else %}
<div class="dropdown">
<button class="dropdown-toggle" type="button" id="loginDropdown" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Signup</span>
</button>
<div class="dropdown-menu" aria-labelledby="loginDropdown">
<a class="dropdown-item" href="{% url 'register' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Patient Login</span>
</a>
<a class="dropdown-item" href="{% url 'login_page' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>AshaWorker Login</span>
</a>
<a class="dropdown-item" href="{% url 'login_page' %}">
<i class="fa fa-user" aria-hidden="true"></i>
<span>Hospital Login</span>
</a>
</div>
</div>
{% endif %}
</div>
</div>
</div>
</nav>
</div>
</div>
</header>
<!-- end header section -->
</div>
```

```
<section class="book_section layout_padding">
<div class="container">
<div class="row">
<div class="col">
<form method="POST" action="" autocomplete="off" enctype="multipart/form-data">
{% csrf_token %}
<h4 id="bookid">BOOK <span>APPOINTMENT</span></h4>

<!-- General Information Section -->
<div class="form-section">
<h5>General Information</h5>
<br>
<div class="form-row">
<!-- Patient Information -->
<div class="form-group col-lg-4">
<label for="inputPatientName">First Name</label>
<input type="text" name="first_name" value="{{ patientprofile.first_name }}" class="form-control"
id="inputPatientName" placeholder="Full Name" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputPatientName">Last Name</label>
<input type="text" name="last_name" value="{{ patientprofile.last_name }}" class="form-control"
id="inputPatientName" placeholder="Full Name" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputEmail">Email</label>
<input type="email" name="email" value="{{ patientprofile.email }}" class="form-control" id="inputEmail"
placeholder="Email Address" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputAddress">Address</label>
<input type="text" class="form-control" value="{{ patientprofile.address }}" name="address" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputPhoneNumber">Phone Number</label>
<input type="tel" name="phone_number" value="{{ patientprofile.phone_number }}" class="form-control"
id="inputPhoneNumber" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputMedications">Medications Currently Taking</label>
<input type="text" class="form-control" value="{{ patientprofile.medication_names }}" name="medication_names"
readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputEmail">Gender</label>
<input type="text" name="gender" value="{{ patientprofile.gender }}" class="form-control" id="inputEmail"
placeholder="Email Address" readonly>
</div>
<div class="form-group col-lg-4">
<label for="inputAddress">Ward</label>
<input type="number" class="form-control" value="{{ patientprofile.ward }}" name="ward" readonly>
</div>
<!-- Add more fields for general information here -->
</div>
</div>
<br>   <br>
<!-- Fill Data Section -->
<div class="form-section">
<h5>Fill Your Informations</h5>
<br>
<div class="form-row">
```

```
<div class="form-group col-lg-4">
<label for="ashaworker" >Select Your Ashaworker:</label>
<select class="form-control" id="ashaworker" name="ashaworker" required>
<option value="" disabled selected>-Select Your Ashaworker-</option>
{% for ashaworker in ashaworkers %}
<option value="{{ ashaworker.id }}">{{ ashaworker.Name }} : {{ ashaworker.ward }}</option>
{% comment %} <option value="{{ ashaworker.id }}" {% if selected_asha == ashaworker.id %} selected {% endif
%}>{{ ashaworker.Name }} : {{ ashaworker.ward }}</option> {% endcomment %}
{% endfor %}
</select>
</div>
<div class="form-group col-lg-4">
<label for="date" >Select a Date:</label>
<select class="form-control" id="date" name="date" required>
<option value="">Select a Date</option>
{% for date in date_options %}
<option value="{{ date.date }}">{{ date.date }}</option>
{% endfor %}
</select>
</div>
<div class="form-group col-lg-4">
<label for="time" >Select a Time Slot:</label>
<select class="form-control" id="time" name="time" required>
<option value="" disabled selected>Select a Time Slot</option>
{% for option in time_options %}
<option value="{{ option.id }}">{{ option.text }}</option>
{% endfor %}
</select>
</div>
<!-- Appointment Status/Urgency Selection -->
<div class="form-group col-lg-4">
<label for="urgency" >Appointment Status/Urgency</label>
<select class="form-control" name="urgency" id="urgency" required>
<option value="" disabled selected>-Select Appointment Status/Urgency-</option>
<option value="Routine check-up">Routine check-up</option>
<option value="Non-urgent medical check-up">Non-urgent medical check-up</option>
<option value="Urgent medical check-up">Urgent medical check-up</option>
</select>
<span id="urgencyError" class="text-danger"></span>
</div>
<div class="form-group col-lg-4">
<label for="inputMedicalConditions" >Existing Medical Conditions</label>
<textarea class="form-control" name="medical_conditions" id="inputMedicalConditions" rows="3"
placeholder="Enter existing medical conditions"></textarea>
<div id="medicalConditionsError" style="color: red;"></div>
</div>
<div class="form-group col-lg-4">
<label for="inputSymptoms" >Symptoms/Reason for Visit</label>
<textarea class="form-control" name="symptoms" id="inputSymptoms" rows="3" placeholder="Describe symptoms
or reason for the visit"></textarea>
<div id="symptomsError" style="color: red;"></div>
</div>
<div class="btn-box">
<button type="submit" class="btn" class="form-group col-lg-6" id="schedule">Schedule Now</button>
</div>&ensp;&ensp;&ensp;&ensp;
<div class="btn-box">
{% comment %} <a href="{% url 'appointment_view' %}">View Appointments</a> {% endcomment %}
<button  class="btn"  ="form-group col-lg-6" onclick="window.location.href = '{% url 'appointment_view'
%}'">View Appointments</button>
</div>
<!-- Add more fields for filling data here -->
```

```
</div>
</div>
</div>
</div>
</form>
</div>
</div>
</div>
</section>
<section class="info_section ">
<div class="container">
<div class="info_top">
<div class="info_logo">
<a href="">
<img src="{% static 'images/ghbc1.png' %}" alt="" style="width:140px; height:90px;">
</a>
</div>
<div class="info_form">
<form action="">
<input type="email" placeholder="Your email">
<button>
Subscribe
</button>
</form>
</div>
</div>
<div class="info_bottom layout_padding2">
<div class="row info_main_row">
<div class="col-md-6 col-lg-3">
<h5>
Address
</h5>
<div class="info_contact">
<a href="">
<i class="fa fa-map-marker" aria-hidden="true"></i>
<span>
Location
</span>
</a>
<a href="">
<i class="fa fa-phone" aria-hidden="true"></i>
<span>
Call +01 1234567890
</span>
</a>
<a href="">
<i class="fa fa-envelope"></i>
<span>
gentlehaloes@gmail.com
</span>
</a>
</div>
<div class="social_box">
<a href="">
<i class="fa fa-facebook" aria-hidden="true"></i>
</a>
<a href="">
<i class="fa fa-twitter" aria-hidden="true"></i>
</a>
<a href="">
<i class="fa fa-linkedin" aria-hidden="true"></i>
```

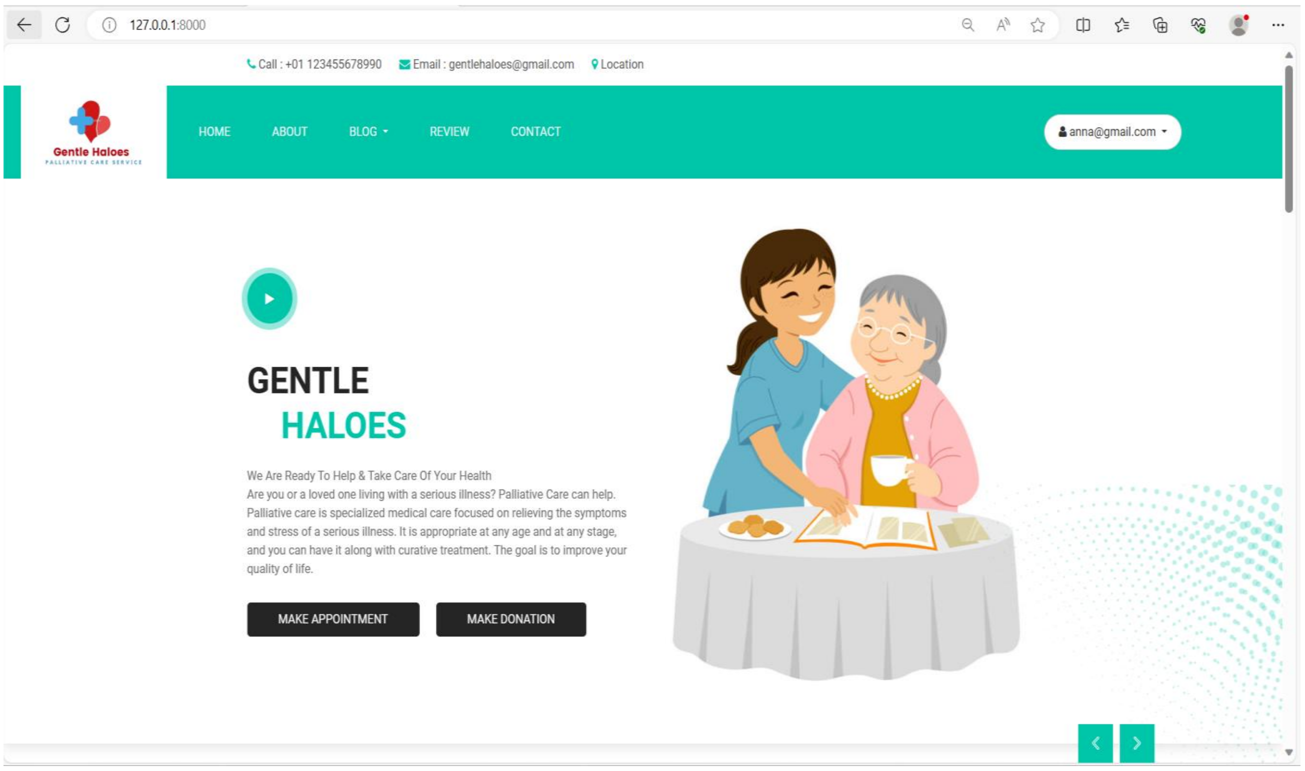
```
</a>
<a href="">
<i class="fa fa-instagram" aria-hidden="true"></i>
</a>
</div>
</div>
<div class="col-md-6 col-lg-3">
<div class="info_links">
<h5>
Useful link
</h5>
<div class="info_links_menu">
<a class="active" href="{% url 'index' %}">
Home
</a>
<a href="{% url 'about' %}">
About
</a>
<a href="{% url 'treatment' %}">
Services
</a>
<a href="{% url 'gallery' %}">
Gallery
</a>
<a href="{% url 'resources' %}">
Resource
</a>
<a href="{% url 'testimonial' %}">
Review
</a>
<a href="{% url 'contact' %}">
Contact
</a>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</section>
<!-- jQery -->
<script src="{% static 'js/jquery-3.4.1.min.js' %}"></script>
<!-- bootstrap js -->
<script src="{% static 'js/bootstrap.js' %}"></script>
<!-- nice select -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-nice-select/1.1.0/js/jquery.nice-select.min.js"
integrity="sha256-Zr3vByTlMGQhvMfgkQ5BtWRSKBGa2QlspKYJnkjZTmo="
crossorigin="anonymous"></script>
<!-- owl slider -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/owl.carousel.min.js"></script>
<!-- datepicker -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-datepicker/1.3.0/js/bootstrap-datepicker.js"></script>
<!-- custom js -->
<script src="{% static 'js/custom.js' %}"></script>
<script>
// Get the textarea elements
const medicalConditionsTextarea = document.getElementById('inputMedicalConditions');
const symptomsTextarea = document.getElementById('inputSymptoms');
// Get the error message elements
const medicalConditionsError = document.getElementById('medicalConditionsError');
const symptomsError = document.getElementById('symptomsError');
// Get the submit button element
const submitButton = document.getElementById('submitButton');
// Function to validate a textarea field
function validateTextarea(textarea, errorElement) {
// Get the value of the textarea
const value = textarea.value.trim();
// Regular expression to match only alphabetic characters, spaces, and commas
const regex = /^[a-zA-Z, ]+$/;

if (value === '') {
// Empty field, show an error message
errorElement.textContent = 'This field cannot be empty.';
submitButton.disabled = true; // Disable the submit button
} else if (!regex.test(value)) {
// Field contains invalid characters, show an error message
errorElement.textContent = 'This field can only contain alphabetic characters, spaces, and commas.';
submitButton.disabled = true; // Disable the submit button
} else {
// Field is valid, clear the error message
errorElement.textContent = '';
submitButton.disabled = false; // Enable the submit button
}
}
// Add keyup event listeners to both textarea fields
medicalConditionsTextarea.addEventListener('keyup', function() {
validateTextarea(medicalConditionsTextarea, medicalConditionsError);
});
symptomsTextarea.addEventListener('keyup', function() {
validateTextarea(symptomsTextarea, symptomsError);
});
// Add a submit event listener to the form
document.querySelector('form').addEventListener('submit', function(event) {
validateTextarea(medicalConditionsTextarea, medicalConditionsError);
validateTextarea(symptomsTextarea, symptomsError);

// Check if there are validation errors in either fieldif (medicalConditionsError.textContent !== '' ||
symptomsError.textContent !== '') {
// Prevent the form from submitting if there are errors
event.preventDefault();
}
});
</script>
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
var selectedDatesAndTimes = {};
$(document).ready(function () {
var dateSelect = $("#date");
var timeSelect = $("#time");
$("#ashaworker").change(function () {
var ashaworker_id = $(this).val();
// Send an AJAX request to get available dates for the selected Ashaworker
$.ajax({
url: "{% url 'get_dates_for_ashaworker' %}",
data: { 'ashaworker_id': ashaworker_id },
dataType: 'json',
success: function (data) {
dateSelect.empty().append('<option value="" selected>Select a Date</option');
timeSelect.empty().append('<option value="" selected>Select a Time Slot</option');
$.each(data.date_options, function (index, value) {
dateSelect.append($('<option>', {
```

```
value: value,
text: value
}));
});
},
error: function (data) {
console.log(data.error_message);
}
});
});
$("#date").change(function () {
var ashaworker_id = $("#ashaworker").val();
var selected_date = $(this).val();
// Send an AJAX request to get available time slots for the selected Ashaworker and date
$.ajax({
url: "{% url 'get_timeslots_for_date' %}",
data: { 'ashaworker_id': ashaworker_id, 'selected_date': selected_date },
dataType: 'json',
success: function (data) {
timeSelect.empty().append('<option value="" selected>Select a Time Slot</option');
$.each(data.time_options, function (index, value) {
// Check if the time is already selected by another patient
if (!selectedDatesAndTimes[ashaworker_id] || !selectedDatesAndTimes[ashaworker_id][selected_date] ||
selectedDatesAndTimes[ashaworker_id][selected_date] !== value.id.toString()) {
timeSelect.append($('<option>', {
value: value.id,
text: value.text
}));
}
});
},
error: function (data) {
console.log(data.error_message);
}
});
});
});
</script>
{% comment %} {% if error_flag %}
<script>
alert("{{ error_message }}");
</script>
{% endif %} {% endcomment %}
<style>
.quote_btn-container .dropdown-toggle {
background-color: white;
color: #333; /* You can change this color to your preference */
border-radius: 50px; /* Adjust the value for more or less roundness */
padding: 10px 20px; /* Adjust padding as needed */
border: 1px solid #ccc; /* Add a border for better visibility */
}
/* Style the dropdown menu items */
.dropdown-menu a.dropdown-item {
color: #333; /* You can change this color to your preference */
}
.dropdown-menu a.dropdown-item:hover {
background-color: #f7f7f7; /* Change color on hover if desired */
}
.quote_btn-container {
margin-bottom:10px;
margin-top:10px;
```

```
margin-right:15px;
margin-left:10px;
}
.dropdown-menu a.dropdown-item {
color: #333;
}
.dropdown-menu a.dropdown-item:hover {
background-color: #f7f7f7;
}
.book_section h4::before {
position: absolute;
left: 100px;
bottom: 0;
width: 100px;
height: 2px;
background-color: #00c6a9;}
h5 {
font-weight: bold; /* Make it bold */
text-align: center; /* Center align the text */
font-size: 24px; /* Customize the font size to your preference */
}
</style>
</body>
</html>
```

9.1 Screen Shots

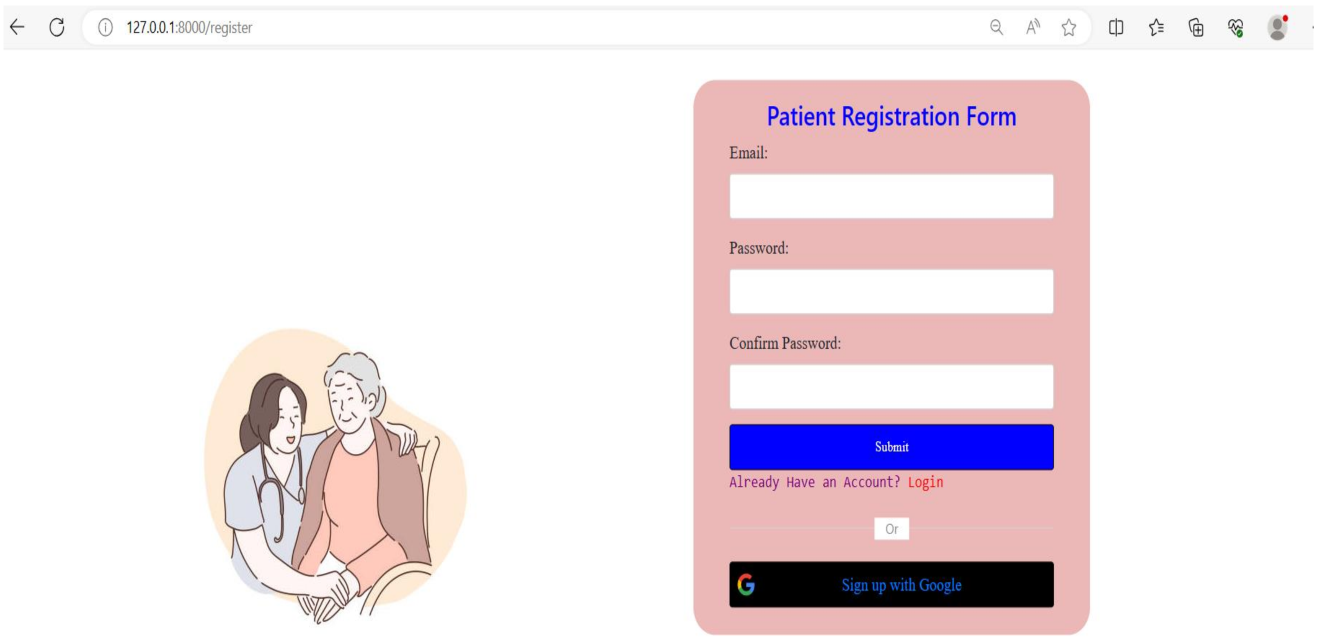
1. Index page



GENTLE
HALOES

We Are Ready To Help & Take Care Of Your Health
Are you or a loved one living with a serious illness? Palliative Care can help. Palliative care is specialized medical care focused on relieving the symptoms and stress of a serious illness. It is appropriate at any age and at any stage, and you can have it along with curative treatment. The goal is to improve your quality of life.

2. Patient Sign up


127.0.0.1:8000/register

Patient Registration Form

Email:

Password:

Confirm Password:

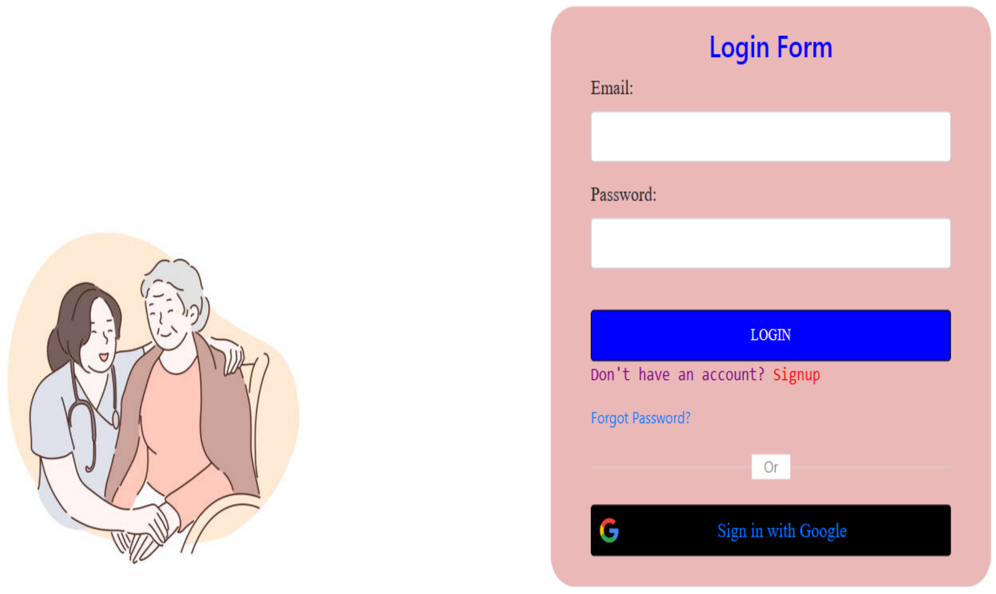
Submit

Already Have an Account? Login

Or

Sign up with Google

3. Login


Login Form

Email:

Password:

LOGIN

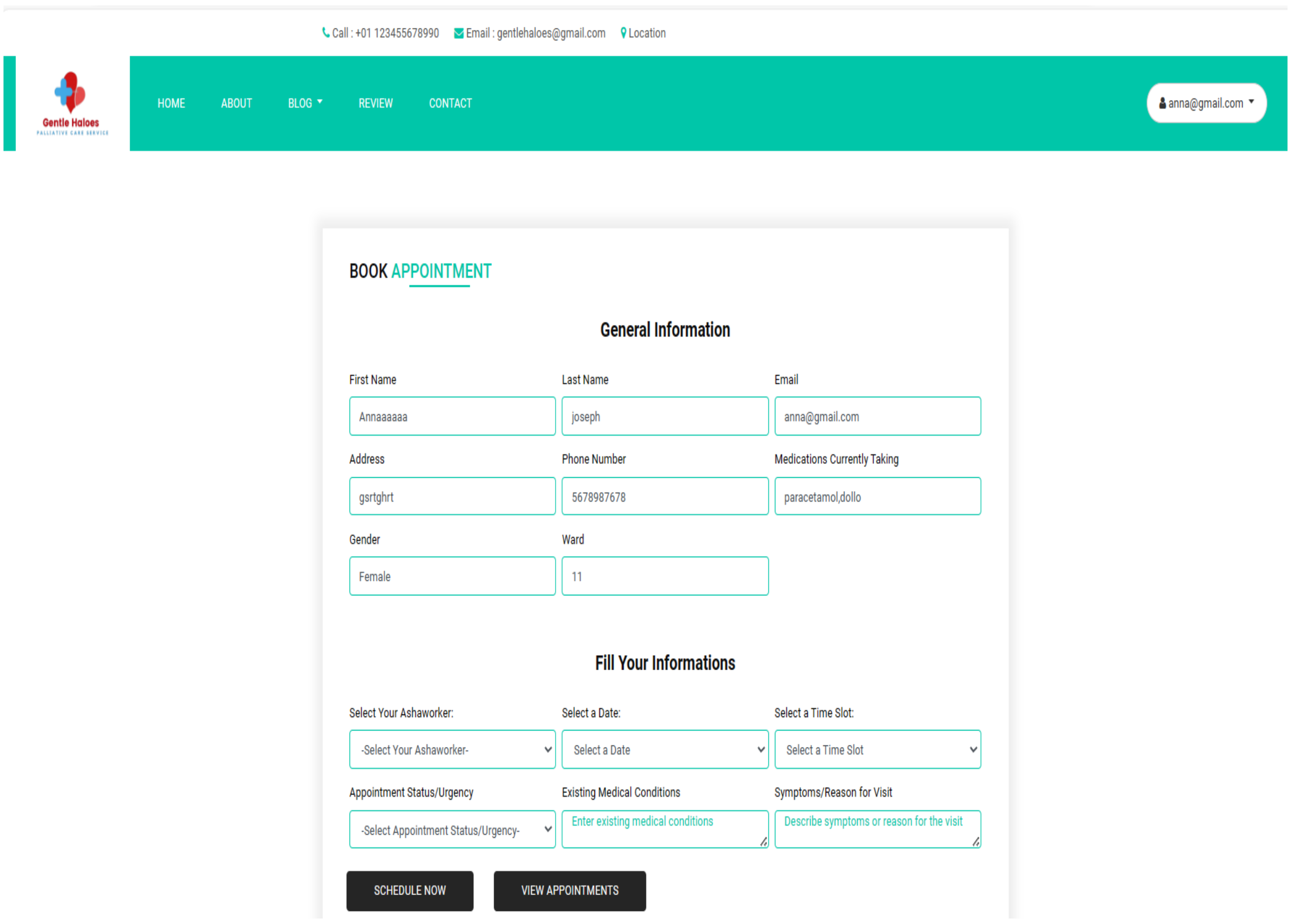
Don't have an account? Signup

Forgot Password?

Or

Sign in with Google

4. Appointment


Call : +01 123455678990 Email : gentlehaloes@gmail.com Location

HOME ABOUT BLOG REVIEW CONTACT anna@gmail.com

BOOK APPOINTMENT

General Information

First Name: Annaaaaaa	Last Name: joseph	Email: anna@gmail.com
Address: gsrtghrt	Phone Number: 5678987678	Medications Currently Taking: paracetamol,dollo
Gender: Female	Ward: 11	

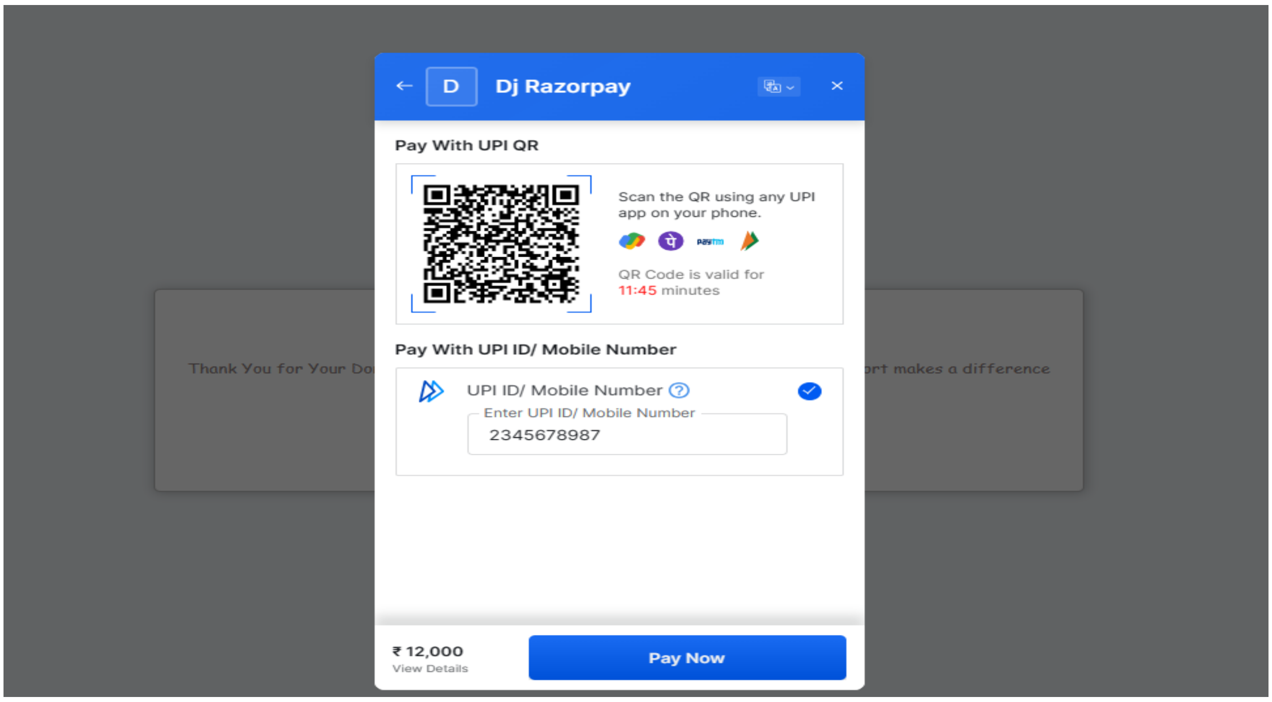
Fill Your Informations

Select Your Ashaworker: -Select Your Ashaworker-	Select a Date: Select a Date	Select a Time Slot: Select a Time Slot
Appointment Status/Urgency: -Select Appointment Status/Urgency-	Existing Medical Conditions: Enter existing medical conditions	Symptoms/Reason for Visit: Describe symptoms or reason for the visit

SCHEDULE NOW VIEW APPOINTMENTS

Screen short 14: Appointment

5. Payment


Dj Razorpay

Pay With UPI QR

Scan the QR using any UPI app on your phone.

QR Code is valid for 11:45 minutes

Pay With UPI ID/ Mobile Number

UPI ID/ Mobile Number

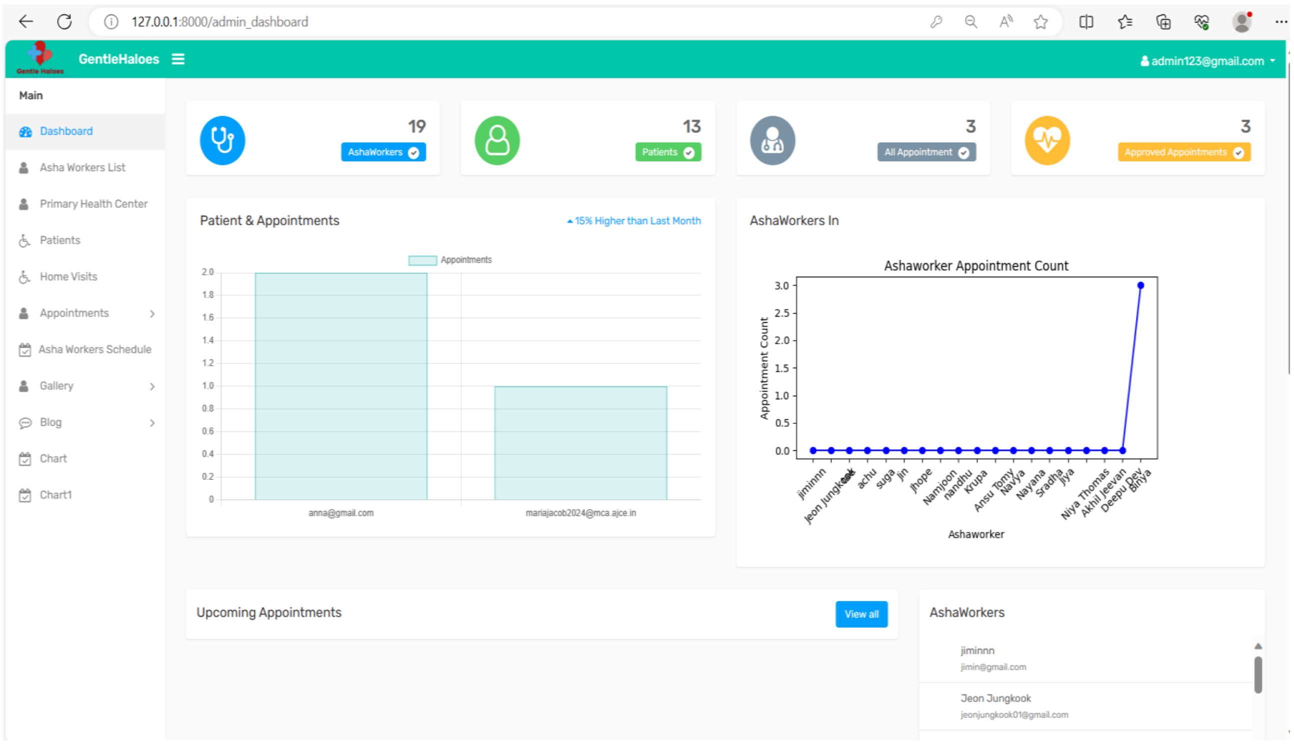
Enter UPI ID/ Mobile Number: 2345678987

₹ 12,000 View Details

Pay Now

Thank You for Your Do... ...rt makes a difference

6. Admin – Dashboard



19 AshaWorkers

13 Patients

3 All Appointment

3 Approved Appointments

Patient & Appointments ▲15% Higher than Last Month

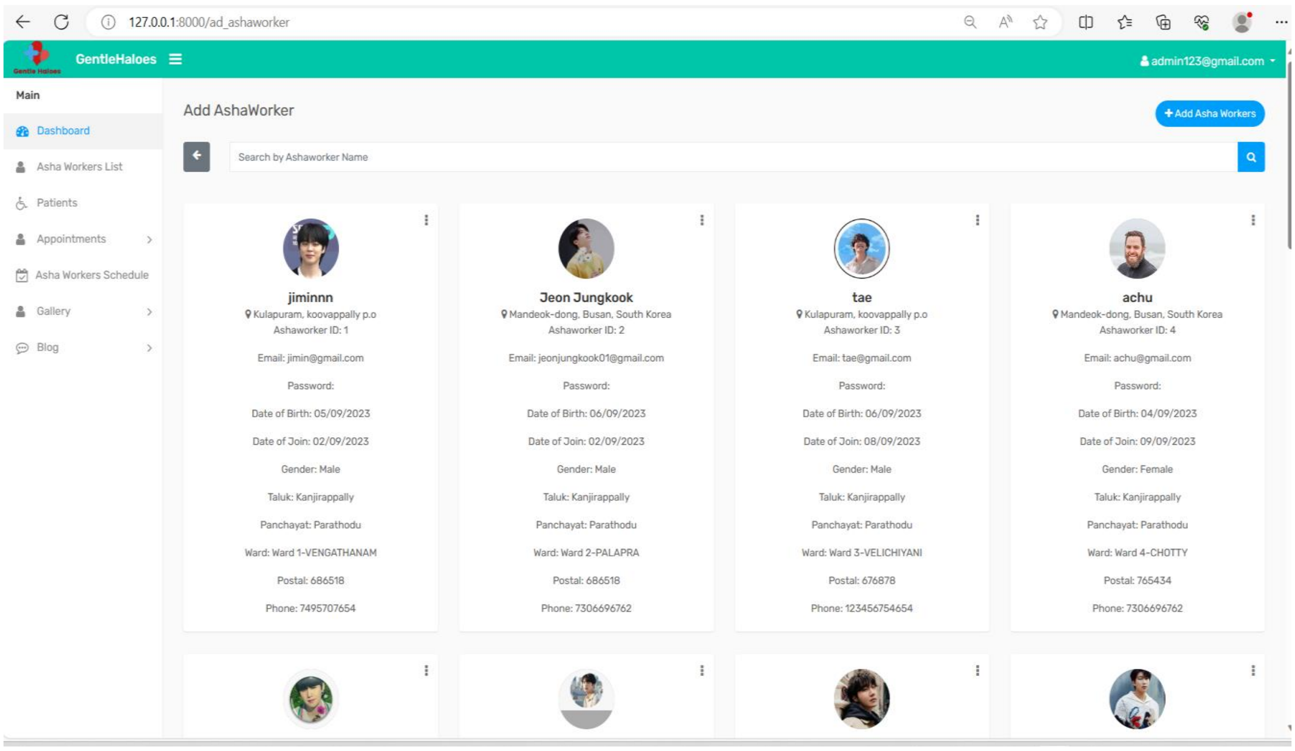
Upcoming Appointments

AshaWorkers

jiminnn
jimin@gmail.com

Jeon Jungkook
jeonjungkook01@gmail.com

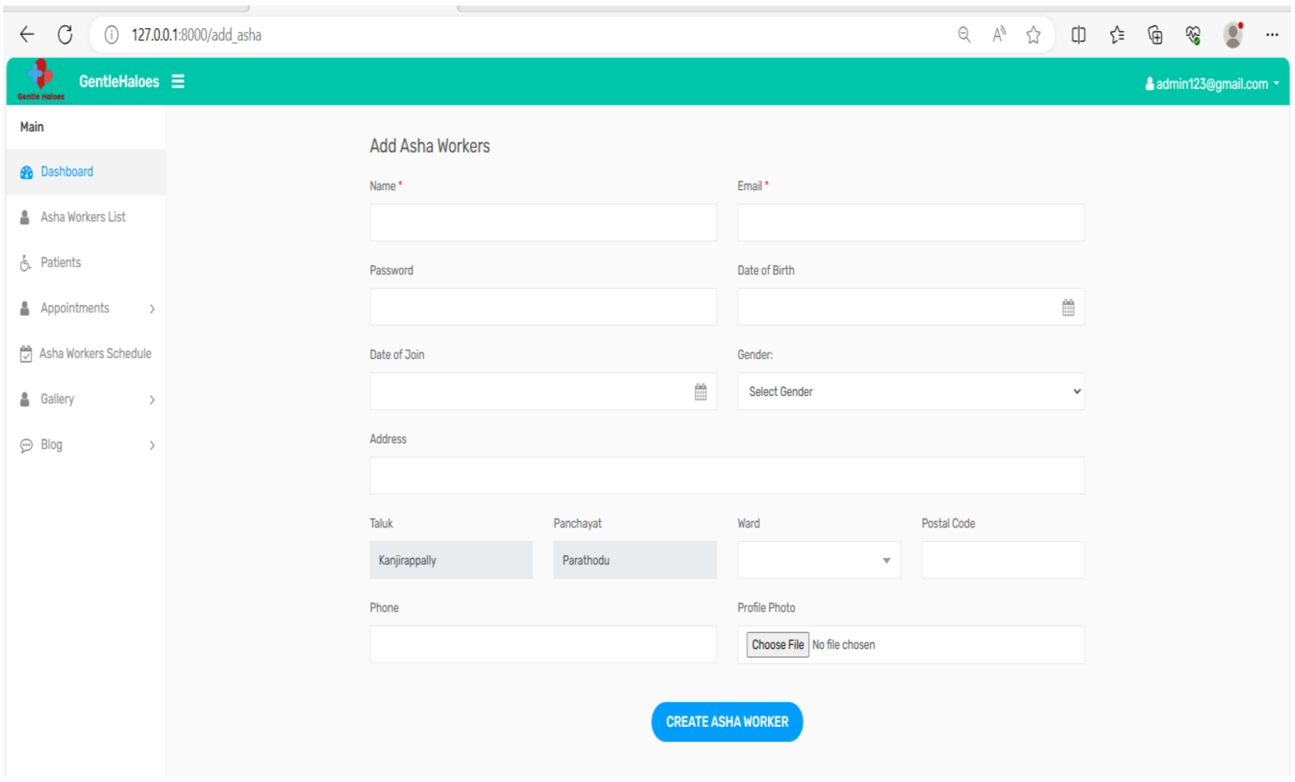
7. Admin – Ashaworker



Add AshaWorker

jiminnn	Jeon Jungkook	tae	achu
Kulapuram, koovappally p.o	Mandeok-dong, Busan, South Korea	Kulapuram, koovappally p.o	Mandeok-dong, Busan, South Korea
Ashaworker ID: 1	Ashaworker ID: 2	Ashaworker ID: 3	Ashaworker ID: 4
Email: jimin@gmail.com	Email: jeonjungkook01@gmail.com	Email: tae@gmail.com	Email: achu@gmail.com
Password:	Password:	Password:	Password:
Date of Birth: 05/09/2023	Date of Birth: 06/09/2023	Date of Birth: 06/09/2023	Date of Birth: 04/09/2023
Date of Join: 02/09/2023	Date of Join: 02/09/2023	Date of Join: 08/09/2023	Date of Join: 09/09/2023
Gender: Male	Gender: Male	Gender: Male	Gender: Female
Taluk: Kanjirappally	Taluk: Kanjirappally	Taluk: Kanjirappally	Taluk: Kanjirappally
Panchayat: Parathodu	Panchayat: Parathodu	Panchayat: Parathodu	Panchayat: Parathodu
Ward: Ward 1-VENGATHANAM	Ward: Ward 2-PALAPRA	Ward: Ward 3-VELICHIYANI	Ward: Ward 4-CHOTTY
Postal: 686518	Postal: 686518	Postal: 676878	Postal: 765434
Phone: 7495707654	Phone: 7306696762	Phone: 123456754654	Phone: 7306696762

8. Admin – Add Ashaworker



Add Asha Workers

Name *

Email *

Password

Date of Birth

Date of Join

Gender: Select Gender

Address

Taluk: Kanjirappally

Panchayat: Parathodu

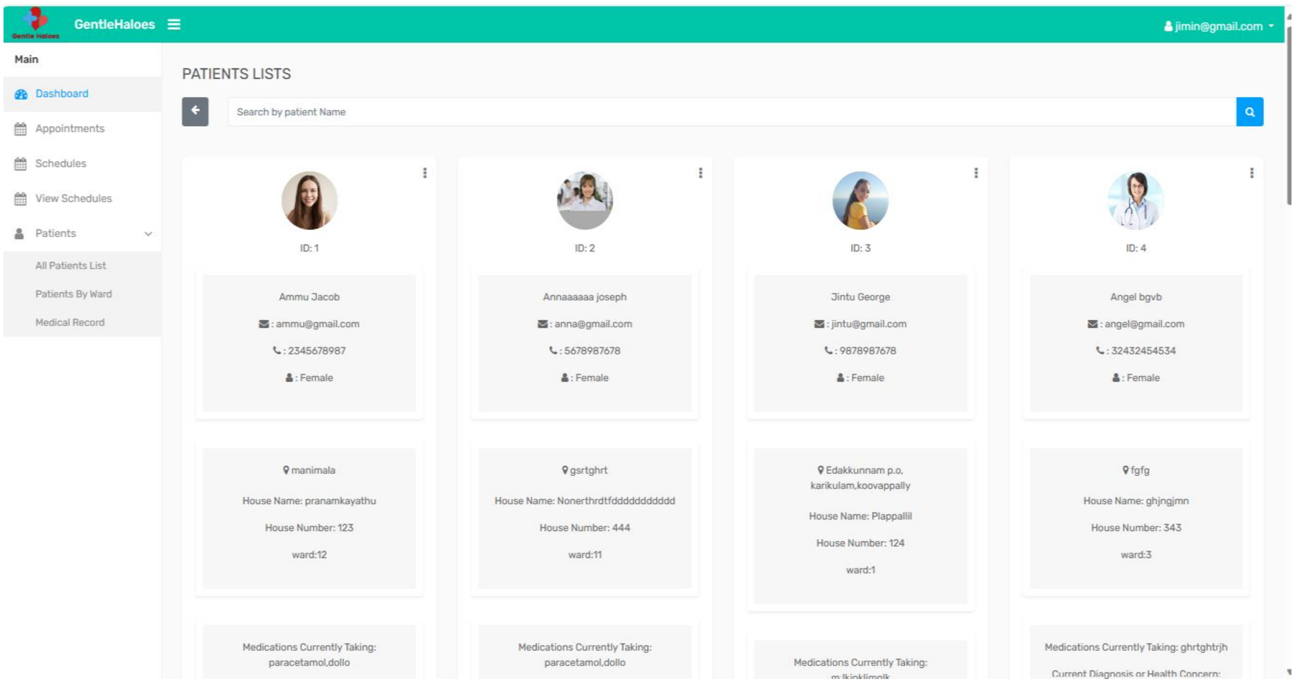
Ward

Postal Code

Phone

Profile Photo: Choose File No file chosen

9. Ashaworker - Patients List



PATIENTS LISTS

ID: 1

Ammu Jacob

: ammu@gmail.com

: 2345678987

: Female

manimala

House Name: pranamkayathu

House Number: 123

ward:12

Medications Currently Taking: paracetamol,dollo

ID: 2

Annaaaaaa joseph

: anna@gmail.com

: 5678987678

: Female

gsrtghrt

House Name: Nonerthrdtfdddddddddd

House Number: 444

ward:11

Medications Currently Taking: paracetamol,dollo

ID: 3

Jintu George

: jintu@gmail.com

: 9878987678

: Female

Edakkunnam p.o, karikulam,koovappally

House Name: Plappallil

House Number: 124

ward:1

Medications Currently Taking:

ID: 4

Angel bgvb

: angel@gmail.com

: 32432454534

: Female

fgfg

House Name: ghjngjmn

House Number: 343

ward:3

Medications Currently Taking: ghrtghtrjh

Current Diagnosis or Health Concern: