PROJECT REPORT

On

"FindCare – An Appointment Booking Website"

Submitted in partial fulfillment of the requirement for the award of

Bachelor's in Computer Application

(BCA) 6th Semester (2019 – 2022)

Submitted to

L. N. Mishra Institute of Economic Development & Social Change

1, Jawaharlal Nehru Marg, Veerchand Patel Road Area, Patna, Bihar – 800028

Bihar Government Autonomous Institute Affiliated To

ARYABHATTA KNOWLEDGE UNIVERSITY, PATNA

Approved By AICTE & Recognized and Aided by The H.R.D. Govt. Of India

Under The Guidance Of

Vijay Sir
Professor
Department of Computer Application
LNMI (Patna)

Submitted By

Aman Sharma
BCA (6th Semester)
Roll no. 19636
Reg No. 19303333035
Session: 2019-2022

Declaration

I **Aman Sharma**, Roll No. **19636, BCA (6th Semester), Lalit Narayan Mishra Institute of Economic Development and Social Change, Patna.** Hereby declare that the project report entitled **"FindCare"** which is submitted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Computer Application** to **ARYABHATTA KNOWLEDGE UNIVERSITY, PATNA**

I also declare that this project is an authentic record of my original work carried out under the guidance of **Mr. Santosh Kumar Jha,** Asst. Professor, Department of Computer Application, LNMI, PATNA and has not been submitted to any other University or Institution for the award of any degree, or personal favor whatsoever. All the details and analysis provided in the report hold true to the best of my knowledge.

_____

(Signature of Student)
Aman Sharma

Acknowledgment

I take this opportunity to express my deepest gratitude and appreciation to all those people who made this project work easier with word of encouragement, motivation and helped me towards the successful completion of this project work.

First I would like to express my sincere gratitude to my Project Guide, **Prof. Vijay Kumar** (Department of Computer Application), Lalit Narayan Mishra Institute of Economic Development and Social Change, Patna for his insightful advice, motivating suggestions, invaluable guidance, help and lots of moral support in successful completion of our Project **FindCare** and also for his constant encouragement and advice throughout my BCA (Bachelor Of Computer Application) programme.

We take the opportunity to thank our team members (**Aman Sharma, Amardeep Anand, Neeraj Kumar, Nitin Shivam**) who contributed towards the successful completion of this project.

I would like to thank all other teaching staff for their valuable teaching and constant advice which made me to finish this program successfully.

Finally my deepest gratitude goes to my parents and friends who have given me much needed comfort, support, encouragement and inspiration for completing this project.

Dated: 15th of May 2022

With Gratitude,
Aman Sharma
Roll no.: 19636
BCA (6th Semester)
Reg. No.: 19303333035
Session: 2019-2022

_____
Signature

Abstract

Our project "**FindCare**" is an online platform to bridge the gap between patients and doctors by offering efficient, convenient and affordable way of booking appointments with the best doctors, consult them.

This idea is not new in urban areas since other firms such as **PRACTO** and **LYBRATE** are already active there, but they are not so active in small cities and rural areas. Since, about 70% of India resides in the villages a vast population are unable to avail these kinds of online facilities due to illiteracy, lack of awareness and communication gap.

Moreover, a huge bunch of people are unaware of good doctors and healthcare facilities. So, our main aim is to cover these rural areas and fix the health problems in a very holistic manner by increasing the accessibility of doctors and enhancing the ability of users to communicate with doctors anytime anywhere.

"**FindCare**" will act as a link between the people and the doctors with ease. People can book appointments through our website with just at a click. They would not have to waste their time and money on transportations, lodging and fooding in case those people must go outstations for taking appointments.

Content

S. No.	Topic	Page No.
1.	**Objective of the Project**	1-2
2.	**Introduction of Project**	3-7
3.	**System Analysis** - Principle of System Analysis - Software Requirements Specification (SRS) - Project Perspective - Project Category - Hardware & Software Requirements - Definition, Acronyms and Abbreviation - Gantt Chart - Pert Chart - ER Diagram - DFD Diagram	8-28
5.	**System Implementation** - Hardware Evaluation Factors - Software Evaluation Factors	29-30
6.	**System Design** - UI Design (Frontend, Email) - System Design - Data Modeling - Module Design - Procedural Design - Scheduling	31-62
7.	**Coding** - Frontend - Backend	63-123
8.	**Testing** - Testing Plan Used - Testing Objective	124-125
9.	**Maintenance & Future Scope of The Project**	126
10.	**Limitations of the Project**	127
10.	**Conclusion**	128
11.	**Bibliography/References**	129

1. *Objective of the Project*

FindCare is designed to manage all the areas of a hospital such as medical, administrative and the corresponding processing of services. It is an integrated website that handles different directions of clinic workflows. It manages the smooth healthcare performance along with administrative, medical, legal, and financial control. That is a cornerstone for the successful operation of the healthcare facility.

❖ *Purpose*

FindCare is a website designed to manage all the areas of a hospital such as medical, financial, administrative and the corresponding processing of services. The primary purpose of creating this website is to is to bring the availability of doctors, pharmacy, nurses, ambulances and every medical facility at one platform and save the time and money of the people by letting them book appointment with the best doctors from their home itself or the "**FindCare**" near their houses in their villages.

❖ *Project Scope*

Daily functions like patient registration, managing appointment and overall management of various functions can be easily performed with higher accuracy after using **FindCare**. The modules of this website are user-friendly and easy to access. This project has a large scope as it has many features which help in making it easy to use, understand and modify it.

❖ *Document Conventions*

The document is prepared using Microsoft Word 2016 and has used the font type 'Calibri'. The fixed font size that has been used to type this document is 16pt and for headings 26pt with 1.5 linespacing. It has used the bold property to set the headings of the document. All pages except the cover page are numbered, the numbers appear on the lower right-hand corner of the page. Every image and data table are numbered and referred to the in the main text.

❖ *Intended Audience*

The intended audience of this document would be the client and specific employees like Doctors and Medical Staffs, consultants and System Operators of different Hospitals and Clinics, and project team, supervisor with the objective to refer and analyze the information. The SRS document can be used in any case regarding the requirements of the project and the solutions that have been taken. The document would finally provide a clear idea about the website.

❖ *Additional Information*

FindCare can be used by entering respective username and password. It is accessible either by an administrator or doctor or patient. **FindCare** uses **SSL** that is Secure Sockets Layer and, in short, it's the standard technology for keeping an internet connection secure and safeguarding any sensitive data that is being sent between two systems, preventing criminals from reading and modifying any information transferred, including potential personal details.

❖ *References*

MediBuddy: https://www.medibuddy.in/
Practo: https://www.practo.com/
Lybrate: https://www.lybrate.com/

2. *Introduction of Project*

❖ *Introduction*

My project is **Medical Appointment Booking Website – FindCare** which includes registration of patients, storing their details into the system. My website has the facility to give a unique id for every patient and stores the details of every patient. The Medical Appointment Booking Website can be used by entering respective email and password. Itis accessible either by an **Admin, Patients and Doctors**. Only the respective person can add data in the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected and data processing is very fast, accurate and relevant.

❖ *Product Perspective*

FindCare is a self- contained system that manages activities of the hospital as slot booking, scheduling appointments, personnel management, and administrative issues.

❖ *Product Features and Modules*

Doctor Module:

- Add clinic details
- Add slot timing and clinic timing
- Check Booked Appointments
- Check Upcoming Appointment
- Check List of all the patients
- Search Details of a particular patients
- Approve or reject the appointment
- Edit or update profile
- Change password
- Check analytics of clinic
- Update Profile Photo

Patient Module

- Add patient's details
- Book appointments
- Cancel Appointments
- Search doctors using name
- Search doctors using symptoms
- Check booked/upcoming appointments
- Edit or update profile
- Change password
- Update profile photo

Admin Module

- Create patient profile
- Create doctor profile
- Ban a patient
- Ban a doctor
- Verify a doctor
- Change password
- Check analytics of website

❖ ***Need of FindCare***

The following website has been designed in **SvelteKit** along with **Tailwind CSS** for styling, **Python and Node.js** for the backend work like, day to day activities of booking the doctors, checking the availability of slots for that doctor, check the timing of clinic and consulting the correct doctor regarding any specific diseases. The complete set of rules and procedures of booking and use of the medical facilities has been provided in this project. The following points provide the detailed information of the website.

- **Efficiency**: The basic need of the project is efficiency. The project should be efficient so that whenever the booking is done through the website no error is reported and the patient is satisfied through the service provided by **FindCare.**

- **Security**: Security must be the first and foremost criteria of any website so that the customer does not hesitate to use the their personal details in any manner. Since illegal access may corrupt the database the website and admin panel so security and authenticity of the user and admin must also be verified.

- **Performance**: The present booking system of appointment is the in person presence of the patient and book his appointment with the doctor. This manual booking of the appointment is time consuming and hectic. To improve the performance of the system **FindCare** provide a friendly interface for booking the appointments and availability of all the medical facility.

- **Control**: The complete control of the booking and order of facilities is under the control of the admin who is the authorized person and can see all the appointments, booking with the doctors. The admin can remove any suspected user from the website and has full control of adding or removing the doctors who do wish to continue with the company.

❖ ***Software and Hardware Requirements***

Software Requirements

- **Web Server:** A web server is used to host a website. All the Html, JavaScript, CSS files, databases, media files that make up the entire website are stored on this server. The web server can be either runs on Windows or Linux Operating System.

- **Server Software:** When a user visits any website on a web browser, the web server knows the client is requesting some specific information. So it processes that request and serves the correct files or page to the client.

- **Web Tools**: A web authoring tool is used to create the front end of the website. They range from basic HTML text editors (like Notepad++) to more complex graphic authoring tools and CMS (content management system) with built-in frameworks and debugging tools.

- **Database System:** A database is an integral part of any website. The database is used to store the information about the products and services of

the website such as pricing, description, image, details, and sales. In addition to this, the customer details are also stored in the database like what they ordered, their payment details, shipping details, and contact information.

- **Domain Names:** Domain names link to a company or a brand. Successful e-commerce companies have recognizable domain names. If a business becomes successful online, it is important to protect the brand by doing multiple registrations of domain names such as .com, .net, .uk, .org, .co

In brief Software Requirements for *FindCare* is as follow:

- **FrontEnd:** SvelteKit, HTML5, TailwindCSS
- **BackEnd:** Python and Node.js
- **Database:** PostgreSQL
- **Server:** Nginx Based Server

Hardware Requirements

- **Processor:** A processor is the logic circuitry that responds to and processes the basic instructions that drive a computer. The four primary functions of a processor are fetch, decode, execute and write back. The basic element of the processor is ALU, Registers, cache Memory. The processor needed for this project was Core 13 1.4 Ghz processor**.**

- **RAM:** Random Access Memory is a physical hardware inside a computer that temporarily stores data, serving as the computers working memory. The RAM used here is 4GB.

- **Hard Disk:** The hard disk is the main and usually largest, data storage hardware device in computer The computer operating system, software titles and most other files are stored in the hard disk drive. The hard disk that was used here was 500GB.

- **Cache Memory:** Cache memory is the memory that can access more quickly than it can access regular RAM. The purpose of the cache memory is to store program instructions that are frequently re referenced by software during operation. Fast access to these instructions increase the overall speed of the software program. The cache memory that was used here was 128Kb or above.

- **Network Card:** A Network interface card (also known as a NIC, network card, or network interface controller) is an electronic device that connects a computer to a computer network, usually a LAN. It is considered a piece of computer hardware. The Network Card used here was based on 802.11ax module.

3. ***System Analysis***

❖ ***Principle of System Analysis***

- **Requirement Gathering**: All possible requirements of the projects are to be developed are captured in this phase. This Stage will involve analyzing and doing feasibility study. This stage allows them to understand well what the user wants

- **Design**: If the requirements / analysis phase is completed and a clear and well thought out plan for the software development has been put on the table, then the next step involves formulating the basic of the software. The design phase takes information from the requirements / analysis phase like the hardware and system requirements and therefore helps in defining overall system architecture.

- **Implementation**: With information from the design phase, the system is first developed in small programs called units or prototypes, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as unit testing. This phase is the actual making of the project it would be nearly impossible to do this phase without the two phases before this is due to the fact with these two phases you know how to make the project.

- **Testing**: After the programming part of the project, it is just as significant to ensure that tests should run at every phase and any problem at any stage should be eliminated.

In IT, **systems analysis** can include looking at end-user implementation of a software package or product; looking in-depth at source code to define the methodologies used in building software or taking feasibility studies and other types of research to support the use and production of a software product, among other things.

Systems Analysis professionals are often called upon to look critically at systems, and redesign or recommend changes as necessary. Inside and outside of the business world, systems analysts help to evaluate whether a system is viable

or efficient within the context of its overall architecture and help to uncover the options available to the employing business or other party.
Systems analysts are different than systems administrators, who maintain systems day to day, and their roles generally involve a top-level view of a system to determine its overall effectiveness according to its design.

❖ ***Feasibility Study***

A **Feasibility Study** is a study that includes the analysis of the software if it is cost effective from the economic view, if it can fulfil the requirement technically, and if it is adaptable in the required environment. It also condiments the groundwork and determine whether the project should be taken or not. Finally, the net result will be rough plane for proceeding with the project.

In the first stage of SDLC we have to make measure feasibility study for the proposed system.

There are basically five parts to any effective Feasibility Study:

1. The Project Scope - l define the project scope earlier. Users can search for doctors, get appointment and also send message to the doctor. User can search doctors which can make user to find specific doctor an easy task. This project has a large scope as it has the following features which help in making it easy to use, understand and modify it.

- Easily take doctor appointment.
- No Need to do Paper Work.
- To save the environment by using paper free work.
- To increase the accuracy and efficiency of the procedure.
- Management of Patient and doctor Data.

2. The Current Analysis - In Various hospital we have to take appointment by phone call or by directly from the hospital. It is time consuming and costly. Hospital staffs have to collect the information manually. It needs paper work and takes input by human and there is a lot of transaction which is not efficiently done. If any modifications or updating is required of any doctor/patient, it has to search and to be done it manually. So it is prone to error.

3. Requirements - User requires using the system simply a laptop, PC or mobile phone to access internet. Using internet it is very much easy to take appointment.

4. Evaluation - In evaluation examine cost effectiveness of the system. It is cost effective as patient need not to go to hospital for taking appointment previously. Time is also saved. One can take appointment from his home using internet.

5. Review - From the above analysis it may be conclude that the system would be useful, cost effective, easy to use and will solve the manual system faults.

❖ ***Cost and Benefit Analysis:***

Commonly referred to by such names as online scheduling software, online booking applications and online scheduler, an online scheduling system is a Web-based application that allows individuals to book their appointments and reservations online conveniently and securely through any Web-connected device, such as a computer, laptop, smartphone, or tablet.

They typically access the online scheduling system through a "Book Now" button found on a Web site or page, or from a URL provided to them by the medial, healthcare or wellness facility. Once a date and time are selected, the system will automatically confirm the booking and instantly record it within the system, without any staff action needed.

In addition to online scheduling, online scheduling systems also come equipped with other beneficial features like automated e-mail and text messages reminders, which the system sends out to patients and booked individuals on a specific date prior to their scheduled appointment; recording and record-keeping capabilities that make it quick and simple to access data associated with a specific appointment.

❖ ***Technical Feasibility***

A system is technically feasible if the organization has or can obtain the equipment to develop, install and operate the system.

The following questions point up those we must solve:

- Will the combination of hardware and software be able to supply performance?

- What is the impact on the end users of the proposed system after implemented?

Technical Criteria: Technical criteria study is concerned with the hardware requirement for the system.

For our system the minimum hardware requirements are:

Processor: Pentium III or more
RAM: 512 MB or more
Hard Disk: Minimum 2 GB
I/O Device: Keyboard and Mouse

For our system the software requirements are:

Operating system: Linux, Windows XP, Windows Vista or more
Server: Nginx web server (node.js server)
Application Software: Google Chrome, Mozilla Firefox.

❖ ***Operational Feasibility***

Operationally feasible project can be accomplished with the organization's available facilities. Operational feasibility is concerned with, how the user will accept the software.

If the software does not meet the user expectation, then the user might not use the software. It is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. It is a measure of how well a proposed system solves the problems. And it takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

The system I developed, I tried my best to make it in such a way, and users of all level can easily use the software.

❖ ***Software Development Life Cycle***

Computer software experts have adapted several approaches from older engineering fields, and for developing new ones. For instance, divide and conquer a widely known technique for handling more complex problems, is used in a few ways in software engineering.

The software engineering process, for example, is normally divided into phases. SDLC is a series of steps which provides a model for development and lifecycle management of software. It varies across industry and organization. But it has a standard. For this several system development life cycle models have been created. Some of them are waterfall, spiral, fountain, build and fix.

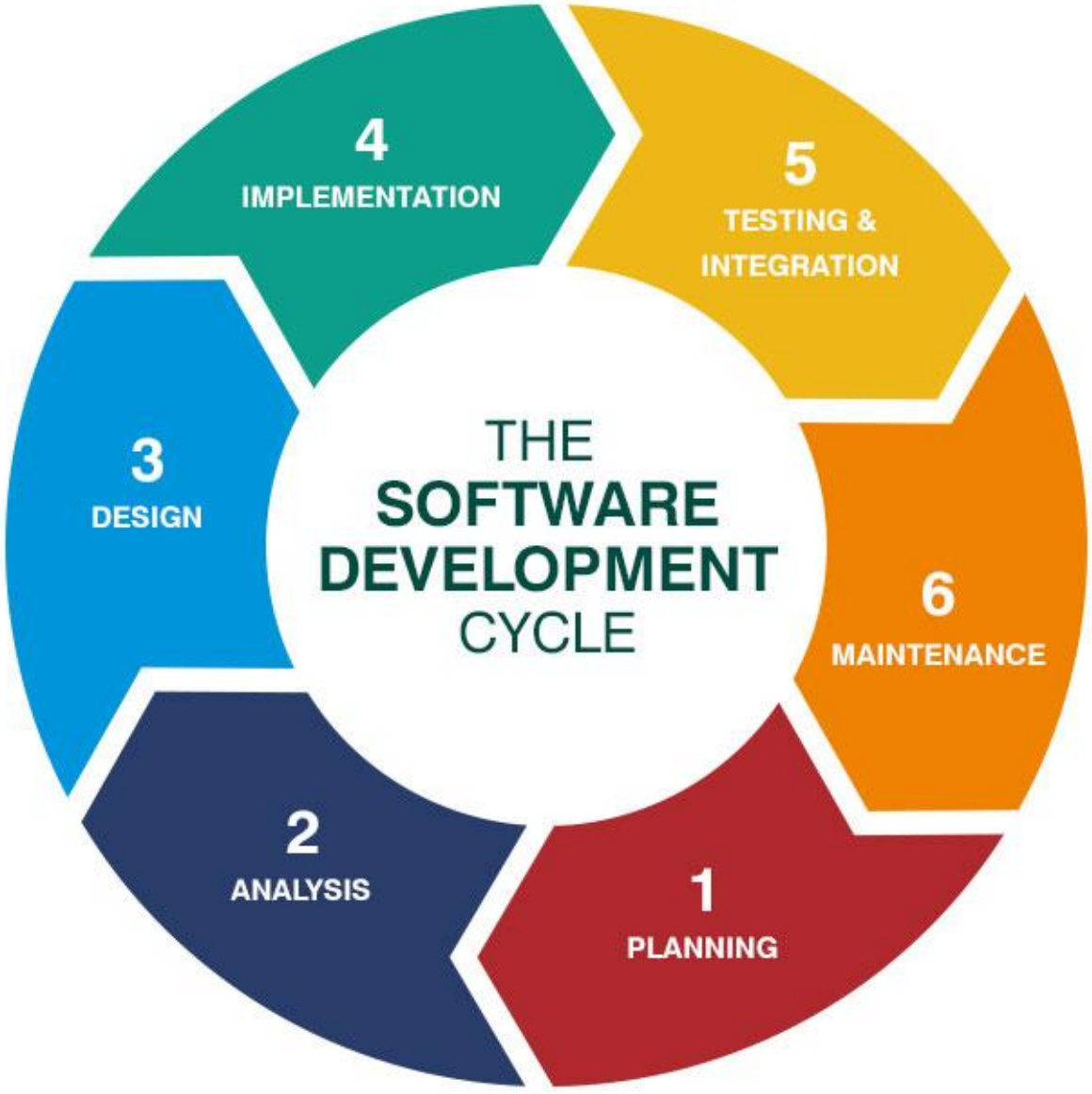


Figure 01: Software Development Life Cycle

Following are the different phases of software development life cycle:

- Feasibility Study
- Requirement Definition
- System Specification
- System Design
- Program, Design, Coding
- Implementation
- Testing
- Changing Request Definition

➢ Feasibility Study

In the first stage of SDLC I must make measure feasibility study for the proposed system. There are basically five parts to any effective Feasibility Study:

1. The Project Scope: l defines the project scope earlier. Users can search for doctors, get appointment, and send message to the doctor. User can search doctors which can make user to find specific doctor an easy task. This project has a large scope as it has the following features which help in making it easy to use, understand and modify it.

- Easily take doctor appointment.
- No Need to do Paperwork.
- To save the environment by using paper free work.
- To increase the accuracy and efficiency of the procedure.
- Management of Patient and doctor Data

2. The Current Analysis: In Various hospital we must take appointment by phone call or by directly from the hospital. It is time consuming and costly. Hospital staffs must collect the information manually. It needs paperwork and takes input by human and there is a lot of transaction which is not efficiently done. If any modifications or updating is required of any doctor/patient, it must search and to be done it manually. So, it is prone to error.
3. Requirements: User requires using the system simply a laptop, PC, or mobile phone to access internet. Using internet, it is very much easy to take appointment.

4. Evaluation: In evaluation examine cost effectiveness of the system. It is cost effective as patient need not to go to hospital for taking appointment previously. Time is also saved. One can take appointment from his home using internet.

5. Review: From the above analysis it may be conclude that the system would be useful, cost effective, easy to use and will solve the manual system faults.

➢ Requirement Definition

The purpose of the requirements phase is to define what a system should do and the constraints under which it must operate. This information is recorded in a requirements document.

A typical requirements document might include a product overview; a specification of the development, operating, and maintenance environment for the product; a high-level conceptual model of the system; a specification of the user interface; specification of functional requirements; specification of non-functional requirements; specification of interfaces to systems outside the system under development; specification of how errors will be handled; and a listing of possible changes and enhancements to the system.

➢ System Definition

In system specification I first focus is ERD (Entity Relationship Diagram), which shows entity with their attributes, relationship with entities involving one-one, one-many, many-one, or many-many.

Data flow diagram, use case diagram, activity diagram, sequence diagram is also designed to understand the system clearly and completely.

➢ System Design

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It helps in specifying hardware and system requirements and also helps in defining overall system architecture.

The system design specifications serve as input for the next phase of the model. It includes interface design, data design, and process design. In user interface design concerned with how users interact with the system, add information to the system.

And how the system will present information back to the users. In data design it concerned with how the data is represented and stored within the system. Finally,

Process Design is concerned with how data moves through the system. I show this in DFD.

➢ Program Design and Coding

The coding phase of the software life cycle is concerned with the development of code that will implement the design. This code is written is a formal language called a programming language.

Programming languages have evolved over time from sequences of ones and zeros directly interpretable by a computer, through symbolic machine code, assembly languages, and finally to higher level languages that are more understandable to humans.

Most coding today is done in one of the higher-level languages. When code is written in a higher-level language, it is translated into assembly code, and eventually machine code, by a compiler.

Many higher-level languages have been developed, and they can be categorized as functional languages, declarative languages, and imperative languages.

➢ Testing

Testing is the process of examining a software product to find errors. This is necessary not just for code but for all life-cycle products and all documents in support of the software such as user manuals.

The software testing process is often divided into phases. The first phase is unit testing of software developed by a single programmer. The second phase is integration testing where units are combined and tested as a group.

System testing is done on the entire system, usually with test cases developed from the system requirements. Acceptance testing of the system is done by its intended users. The basic unit of testing is the test case.

A test case consists of a test case type, which is the aspect of the system that the test case is supposed to exercise; test conditions, which consist of the input values for the test; the environmental state of the system to be used in the test; and the expected behaviors of the system given the inputs and environmental factors.

➢ Implementation

The implementation stage of any project is a true display of the defining moments that make a project a success or a failure. The implementation stage is defined as "the system or system modifications being installed and made operational in a production environment. The phase is initiated after the system has been tested and accepted by the user.

In SDLC model, this implementation part involves implementing the system to any types of administration site and to make process of client request and payment processing

➢ Changing Request Definition

If any advance or updated point comes after delivering the system, then at this stage explains the processing of development of the delivered system. This type of changeable maintenance is very important.

4. ***Software Requirements Specification (SRS)***

A **System Requirements Specification (SRS)** (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application. It includes a variety of elements (see below) that attempts to define the intended functionality required by the customer to satisfy their different users.

1. **Project Perspective**

Users can search for doctors, get appointment and also send message to the doctor. User can search doctors which can make user to find specific doctor an easy task. This project has a large scope as it has the following features which help in making it easy to use, understand and modify it.

- ✓ Easily take doctor appointment.
- ✓ No Need to do Paper Work.
- ✓ To save the environment by using paper free work.
- ✓ To increase the accuracy and efficiency of the procedure.
- ✓ Management of Patient and doctor Data.

2. **Project Category**

Domain: Web Application

A web application (or web app) is application software that runs on a web browser, unlike software programs that run locally and natively on the operating system (OS) of the device. Web applications are delivered on the World Wide Web to users with an active network connect.

3. **Hardware & Software Requirements**

Hardware Requirements (*Recommended*):

- Processor: Pentium III or more
- RAM: 512 MB or more
- Hard Disk: minimum 2 GB

Software Requirements *(Recommended):*

- Operating system**:** Linux, Windows XP, Windows Vista or more
- Server**:** Nginx web server.
- Application Software**:** Google Chrome, Mozilla Firefox.

Tools Used *(Recommended):*

- Code Editor - Visual Studio Code
- UI Designing & Prototyping Software - Figma, Adobe Photoshop.
- Web Hosting Domain – Okteto (okteto.com)
- API testing tool – Postman
- Version control and source code management tool – Github

Programming Languages and Frameworks Used (Front-end):

- HTML
- CSS
- JavaScript
- Tailwind CSS
- Svelte

Programming Languages and Frameworks Used (Back-end):

- Python
- PostgreSQL Database

Special Requirement *(Mandatory):*

- Internet Connection

GitHub: GitHub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, continuous integration, and wikis for every project. Headquartered in California, it has been a subsidiary of Microsoft since 2018.

HTML: HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages. HTML 5 is the fifth and current version of HTML. It has improved the markup available for documents and has introduced application programming interfaces (API) and Document Object Model (DOM).

CSS: Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of a web page but it provides powerful control over the presentation of an HTML document. CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes. Most commonly, CSS is combined with the markup languages HTML or XHTML. CSS is created and maintained through a group of people within the W3C called the CSS Working Group

Tailwind CSS: Tailwind CSS is basically a utility-first CSS framework for rapidly building custom user interfaces. It is a highly customizable, low-level CSS framework that gives you all of the building blocks you need to build bespoke designs without any annoying opinionated styles you have to fight to override.

JavaScript: JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities. JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser. It means that a web page need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content. For example, you might use JavaScript to check if the user has entered a valid email address in a form field. The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server. JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

Svelte: Svelte is a free and open-source front end compiler created by Rich Harris and maintained by the Svelte core team members. Svelte is not a monolithic JavaScript library imported by applications: instead, Svelte compiles HTML templates to specialized code that manipulates the DOM directly, which may reduce the size of transferred files and give better client performance; application code is also processed by the compiler, inserting calls to automatically recompute data and re-render UI elements when the data they depend on is modified. This also avoids the overhead associated with runtime intermediate representations, such as virtual DOM, unlike traditional frameworks (such as React and Vue) which carry out the bulk of their work at runtime, i.e in the browser. The compiler itself is written in TypeScript.

Python: Python is a popular general-purpose programming language that can be used for a wide variety of applications. It includes high-level data structures, dynamic typing, dynamic binding, and many more features that make it as useful for complex application development as it is for scripting or "glue code" that connects components together. It can also be extended to make system calls to almost all operating systems and to run code written in C or C++. Due to its ubiquity and ability to run on nearly every system architecture, Python is a universal language found in a variety of different applications.

PostgreSQL: PostgreSQL is an advanced, enterprise class open source relational database that supports both SQL (relational) and JSON (non-relational) querying. It is a highly stable database management system, backed by more than 20 years of community development which has contributed to its high levels of resilience, integrity, and correctness. PostgreSQL is used as the primary data store or data warehouse for many web, mobile, geospatial, and analytics applications.

4. **Product Functions:**

I. Patient can book appointment by searching symptoms or doctor’s name.
II. Doctor can accept or reject appointment of patients.
III. Providing an understandable and user interactive GUI

5. **Constraints:**

I. Data cannot be intruded by unauthorized persons.
II. Response should be high.
III. Parallel access should be allowed.
IV. Data integrity should be reliable.

6. Assumptions and Dependencies

I. Admin Id (1) is already created and available for use.
II. Roles and responsibilities are already established.

7. Specific Requirements:

- Functional Requirements
- Non-Functional Requirement

7.1 Functional Requirements

I. Functional requirements identified for this system are listed below:

a. Account creation and login of patient.
b. Search symptoms or doctor's name.
c. Booking of Appointment.

II. Online Appointment Booking System contains three phases.

III. The Online Appointment Booking System having a server which can be accessed through some registered systems for verifying doctors, adding users and also for changing the details of the users who were registered are stored in the database. All these information are managed by the administrator.

IV. Admin can also ban patients and doctors that are misusing the system.

V. Admin can also check analytics of website (like total no. of patients and doctors registered, total no. of appointments, etc.).

VI. Users can only login in the system only after verification of registered email.

VII. Users can update or edit their profile.

7.2 Non-Functional Requirements:

Non - Functional requirements identified for this system are listed below:

7.2.1 Usability Requirements:

Usability requirements identified for this system are listed below:

I. A logical interface is essential to make easy use of system, speeding up common tasks.

II. The product could be used by three categories of people: Patient, Doctor, Administrator.

7.2.2 Performance Requirements:

Performance requirements identified for this system are listed below:

I. Database can accommodate thousands of records.
II. At any instant the system should support use of multiple users at a time.
III. Retrieving of data should be reliable.

7.2.3 Reliability:

Some of the attributes identified for the reliability is listed below:

I. All data storage for user variables will be committed to the database at the time of entry.

7.2.4 Portability:

As it is a server-client model application, client can run on any device that can access internet (like mobile, tablet, computer, etc.) and support multiple operating system (like MacOS, Windows, Linux, etc.) and browser (like Chrome, Mozilla Firefox, etc.)

7.2.5 Security Techniques:

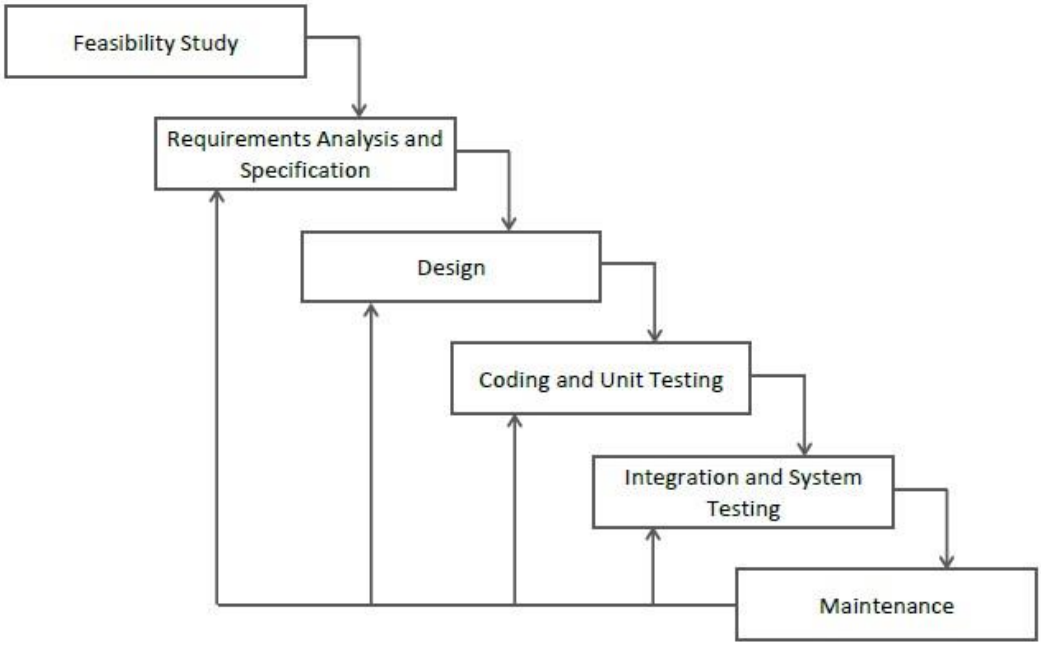
Some of the factors that are identified to protect the software from accidental or malicious access, use, modification, destruction, or disclosure are described below:

I. Every user should be licensed to use the system under any of the three categories provided i.e. Patient, Doctor and Administrator.

II. Communication needs to be restricted when the application is validating the user or license. (i.e., using https)

8. Software Development Life Cycle Model:

SDLC Model Used: Iterative Waterfall Model



Why Iterative Waterfall Model?

In a practical software development project, the classical waterfall model is hard to use. So, the Iterative waterfall model can be thought of as incorporating the necessary changes to the classical waterfall model to make it usable in practical software development projects. It is almost the same as the classical waterfall model except some changes are made to increase the efficiency of the software development.

The iterative waterfall model provides feedback paths from every phase to its preceding phases, when errors are detected at some later phase, these feedback paths allow correcting errors committed by programmers during some phase. The feedback paths allow the phase to be reworked in which errors are committed and these changes are reflected in the later phases.

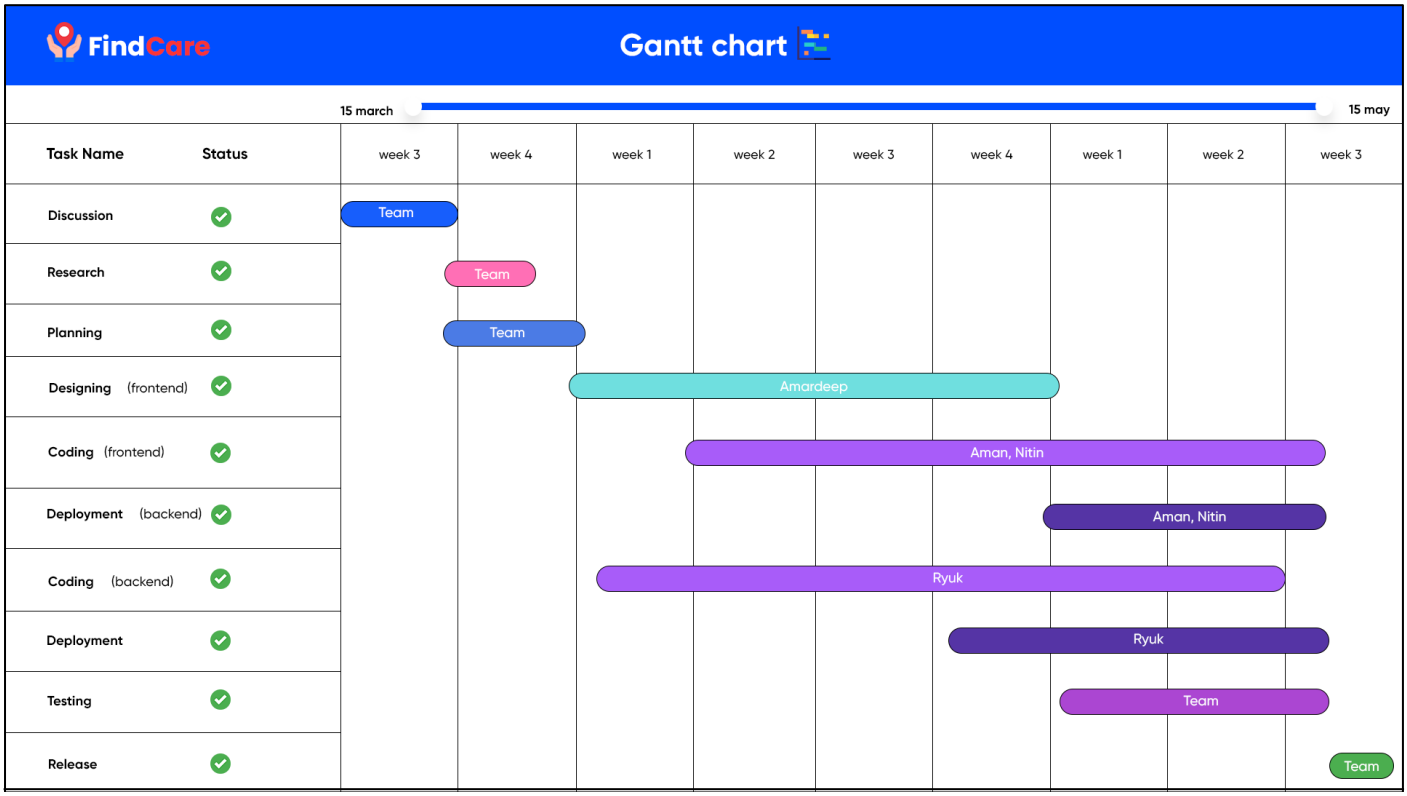
Additional Factors Why We Chose Iterative Waterfall Model:

- **Simple** – Iterative waterfall model is very simple to understand and use. That's why it is one of the most widely used software development models.
- **Cost-Effective** – It is highly cost-effective to change the plan or requirements in the model. Moreover, it is best suited for agile organizations.
- **Well-organized** – In this model, less time is consumed on documenting and the team can spend more time on development

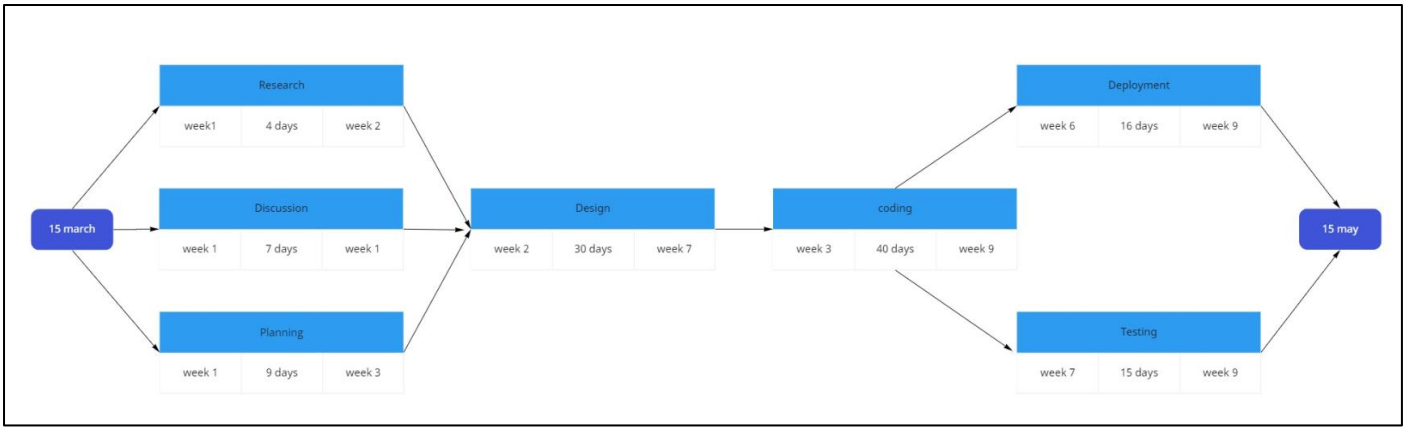
9. Definitions, Acronyms and Abbreviation

S. No.	Acronyms & Abbreviation	Definition
1.	**HTML**	Hyper Text Markup Language
2.	**CSS**	Cascading Style Sheets
3.	**JS**	JavaScript
4.	**API**	Application Programming Interface
5.	**GUI**	Graphical User Interface
6.	**JSON**	JavaScript Object Notation
7.	**URL**	Uniform Resource Locator
8.	**UX**	User Interface
9.	**WWW**	World Wide Web
10.	**PNG**	Portable Network Graphics
11.	**SVG**	Scalable Vector Graphics
12.	**JPEG**	Joint Photographic Experts Group
13.	**RAM**	Random Access Memory
14.	**CPU**	Central Processing Unit
15.	**HDD**	Hard Disk Drive
16.	**OS**	Operating System
17.	**HTTPS**	Hyper Text Transfer Protocol Secure
18.	**W3C**	World Wide Web Consortium
19.	**SQL**	Structured Query Language

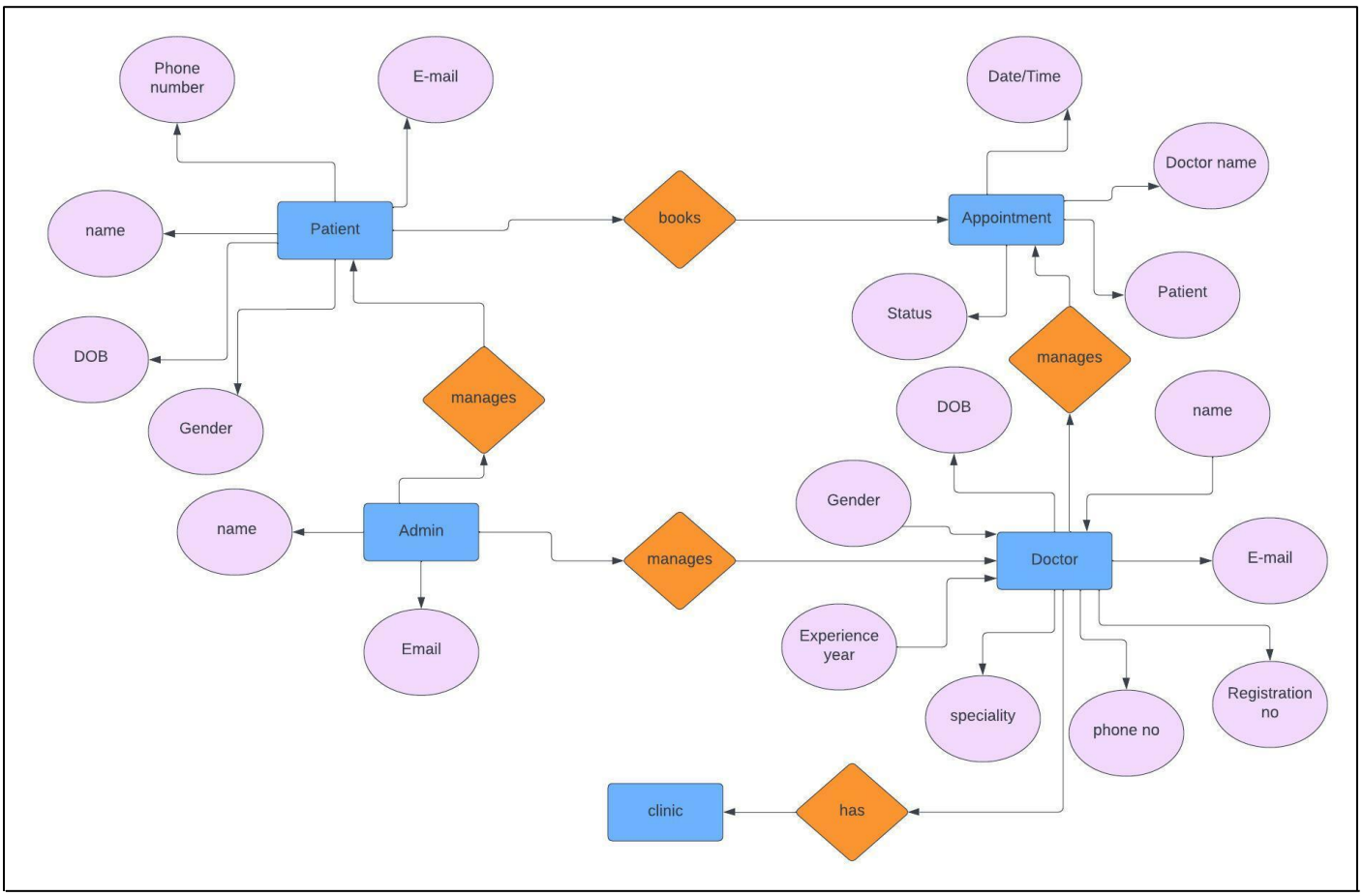
10. Gantt Chart



11. Pert Chart



12. ER Diagram

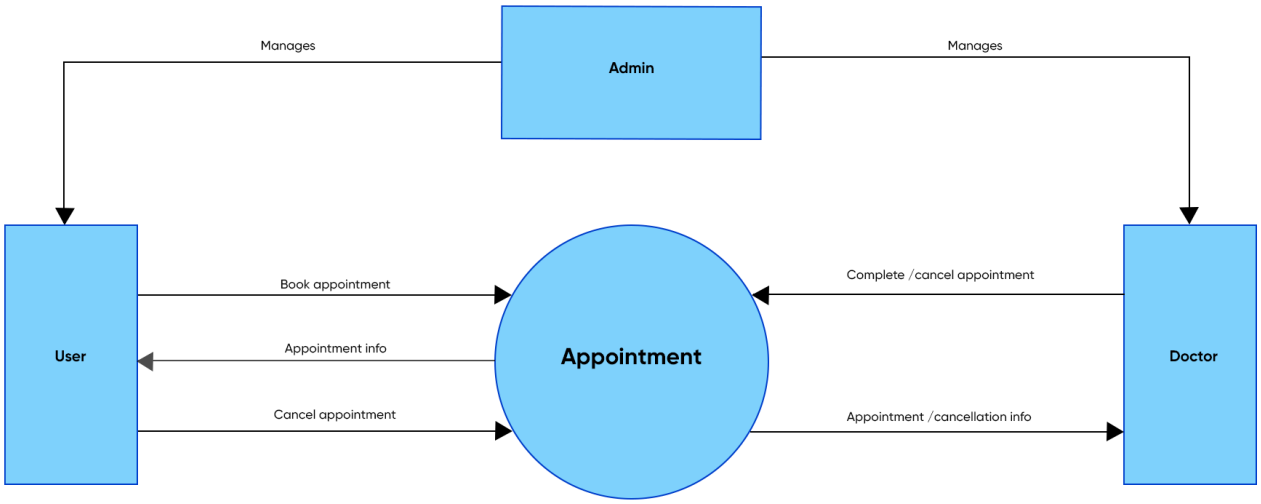


13. Data Flow Diagram

A Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. It's a structured analysis and design tool that can be used for flowcharting in place of or in association with information.

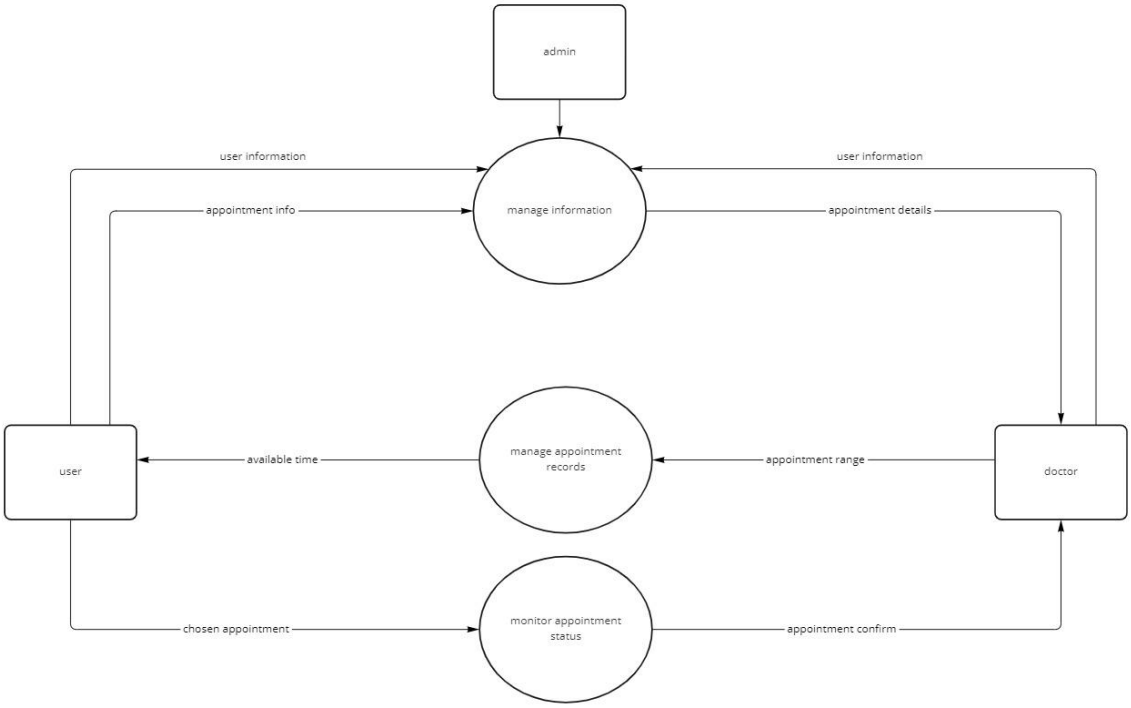
Oriented and process-oriented system flowcharts. Four basics symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage.

- **0 Level DFD**

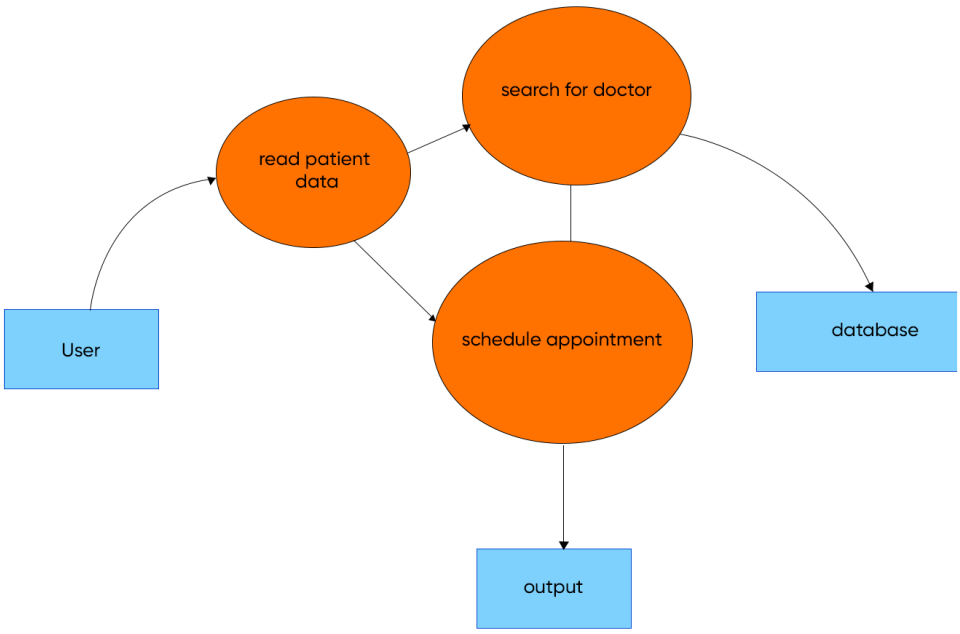


0 Level DFD

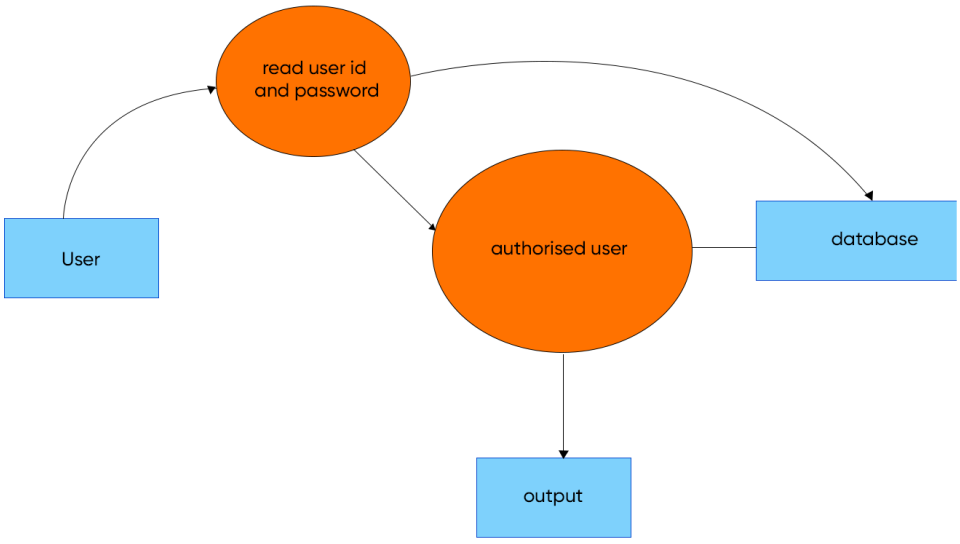
- **1 Level DFD**



1 Level DFD



DFD for booking appointment



DFD for user login

5. System Implementation

The process of creating the web application is difficult as compared to creating the project. Firstly, the website is implemented on a small scale to remove the bugs and errors after which the website is created on a large scale. But before this we must figure out both hardware and software requirement.

When we develop a project, a certain hardware and software is used by the system which should be such that it should be result driven and should satisfy all the needs which the project demands. The hardware should be such that the constraints of the clients should be considered without affecting the performance of the project.

❖ Hardware evaluation factors

While evaluating computer hardware, it should be investigated that that specific physical and performance characteristic should be acquired.

The following criteria is used in hardware evaluation factors:

- ✓ Performance
- ✓ Cost
- ✓ Reliability
- ✓ Availability
- ✓ Compatibility
- ✓ Modularity
- ✓ Technology
- ✓ Connectivity
- ✓ Environmental Requirements
- ✓ Support

❖ Software Evaluation Factors

Software's can be evaluated on number of factors. The most common of them include performance, cost, reliability, compatibility, technology, and support on which the software is judged.

In addition to these the software which require too much memory capacity is also slow, and not good to use. They are not a good selection for the end users and at times cause problem which could be quite irritating a hectic.

Before purchasing a software, it should be noted that the software should be evaluated on the following parameters:

- ✓ **Efficiency:** Whether the software is well written that does not use much of the capacity of the CPU time?
- ✓ **Flexibility:** Can it handle its processing assignments without major modification?
- ✓ **Security:** Does it provide control for errors, and does it authenticate the users?
- ✓ **Hardware:** Does the software works well with the given hardware?
- ✓ **Documentation:** Whether the software has been properly documented or whether there is any fault in the documenting part.?
- ✓ **Language:** Do the programmers write the code in language which you use or understand?
- ✓ **Performance and Economy:** The software is checked if the software is feasible and whether it is in the budget of the company?

6. System Design

❖ UI Implementation (Desktop)

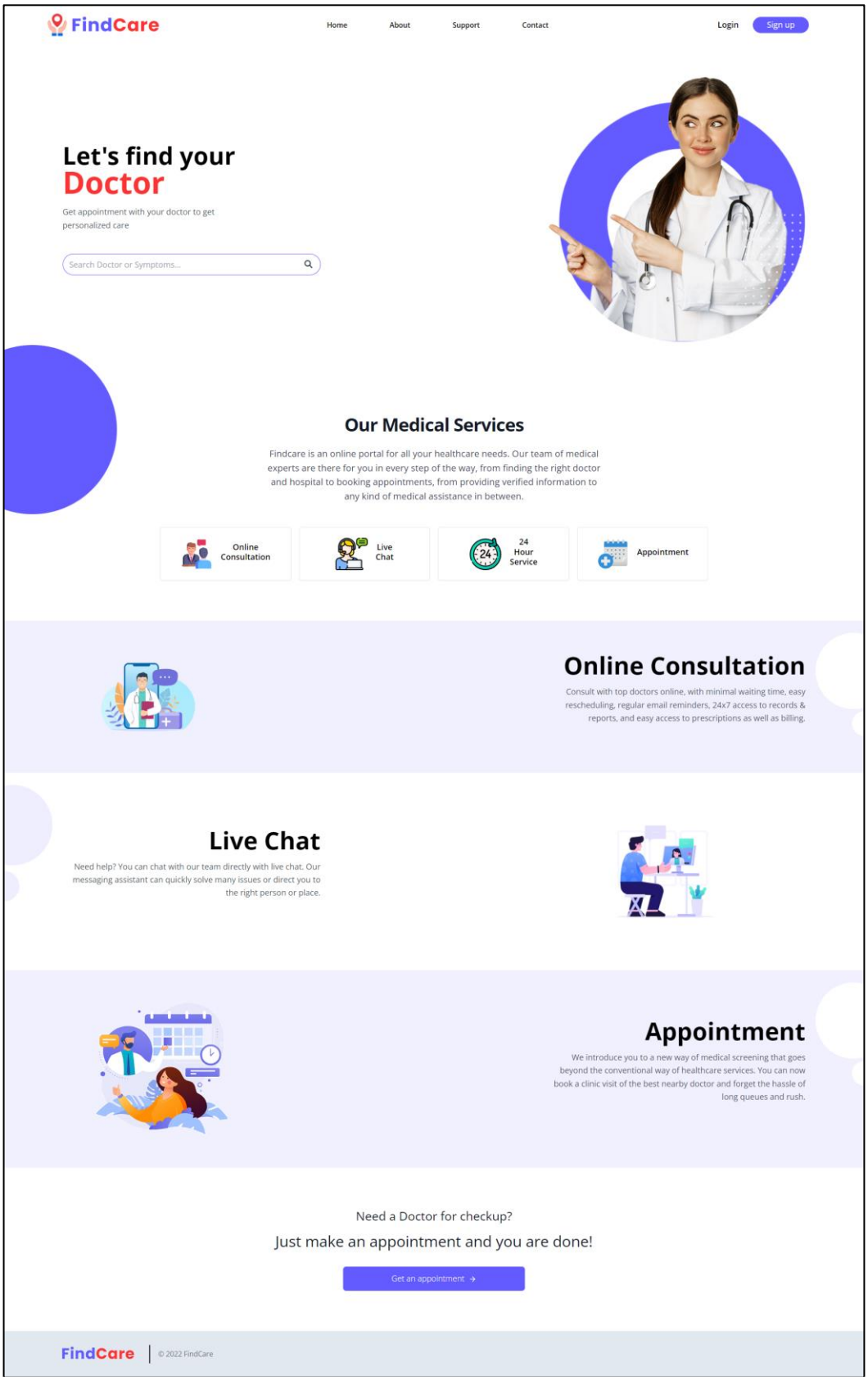


Fig. 1. Home Page

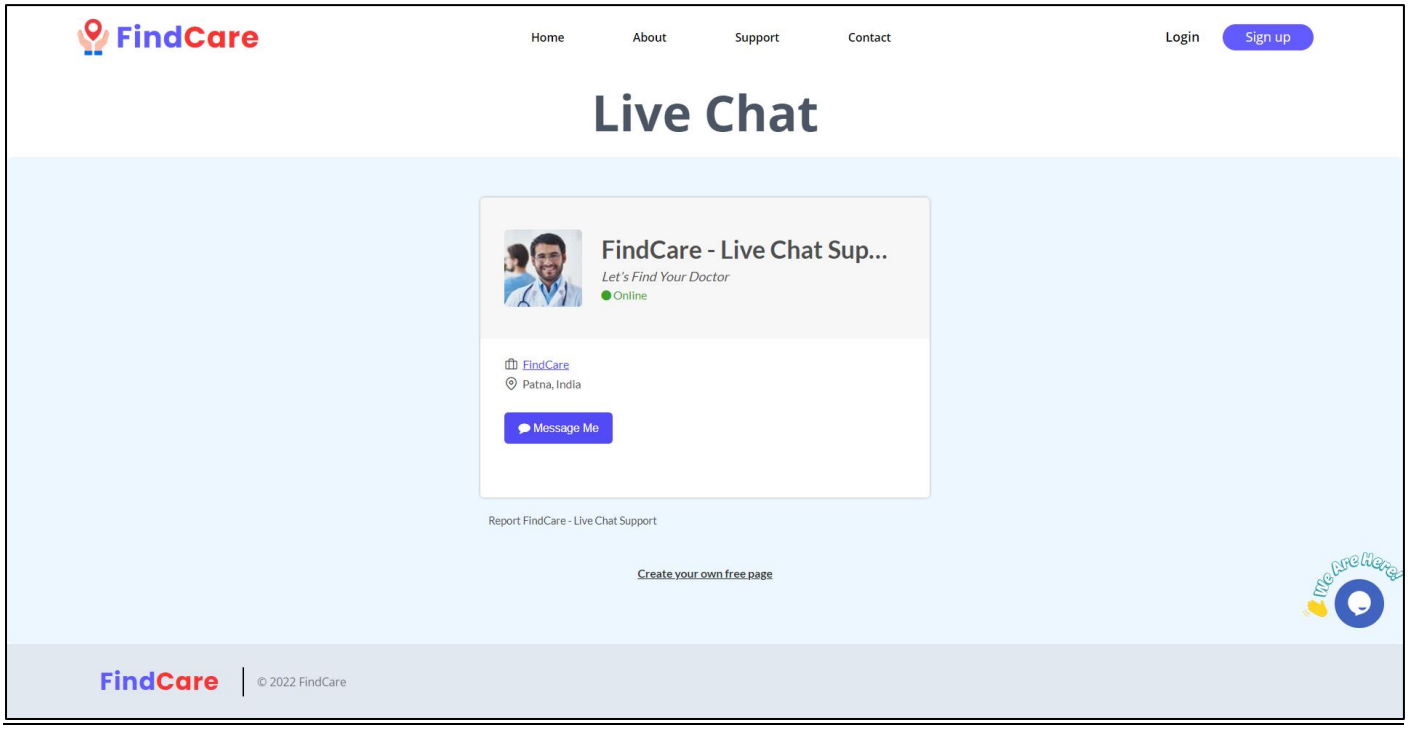


Fig. 2. Support Page

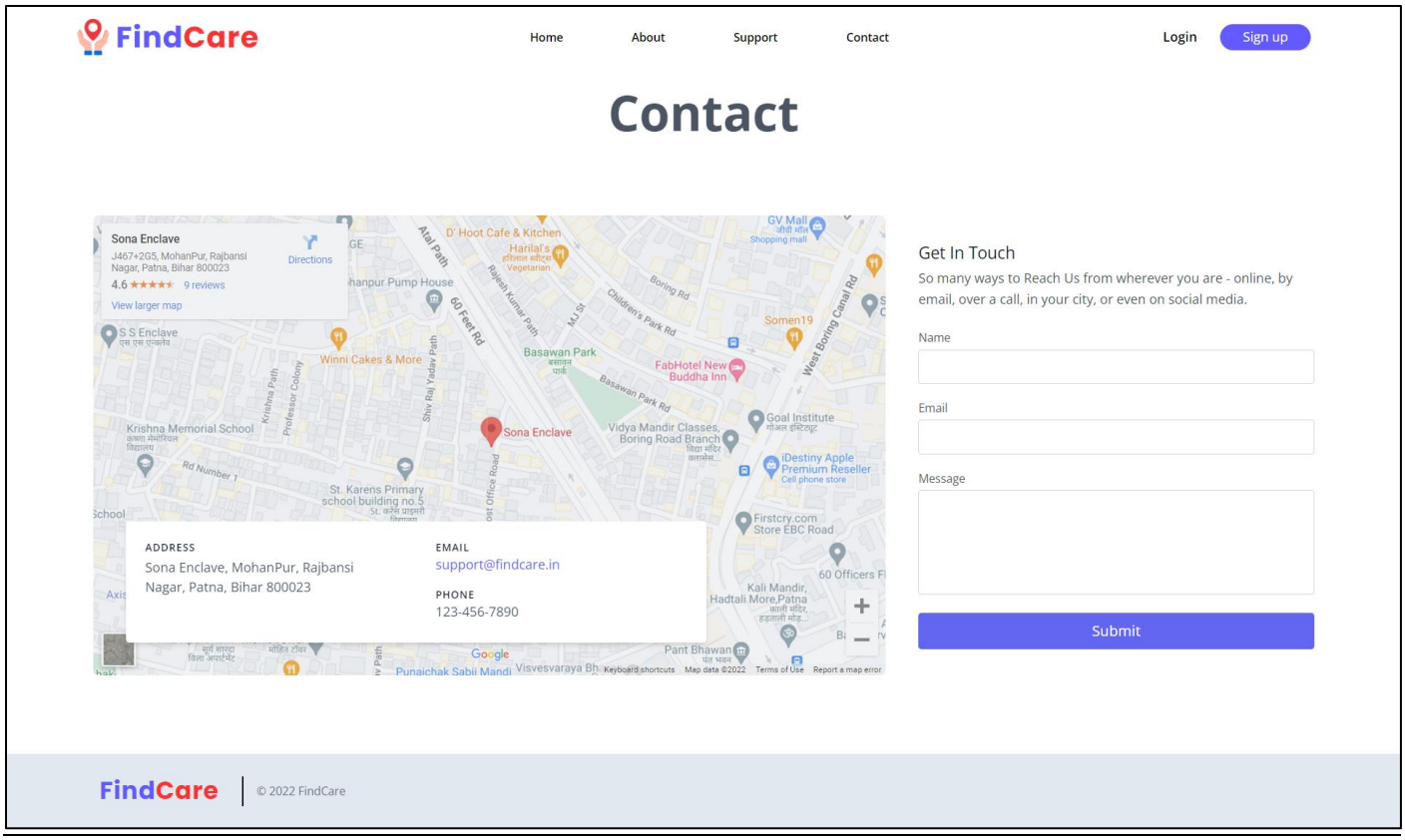


Fig. 3. Contact Page

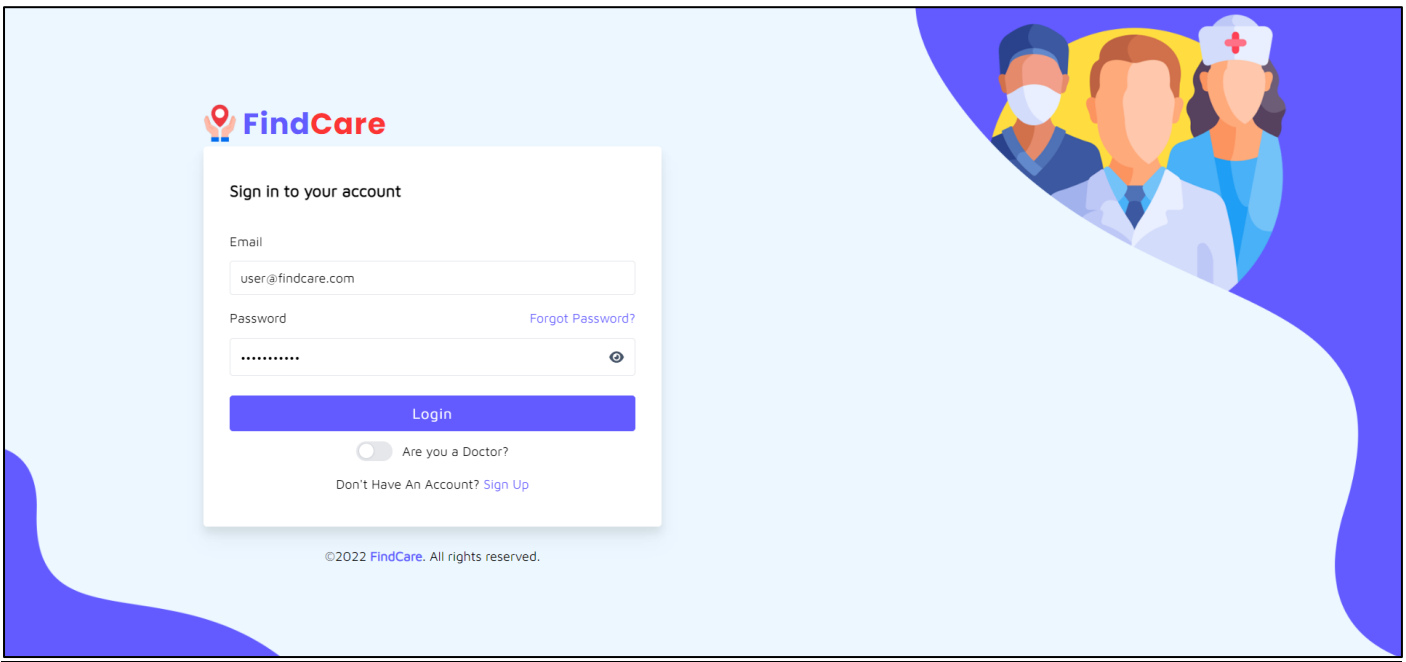


Fig. 4. User/Doctor Login Page

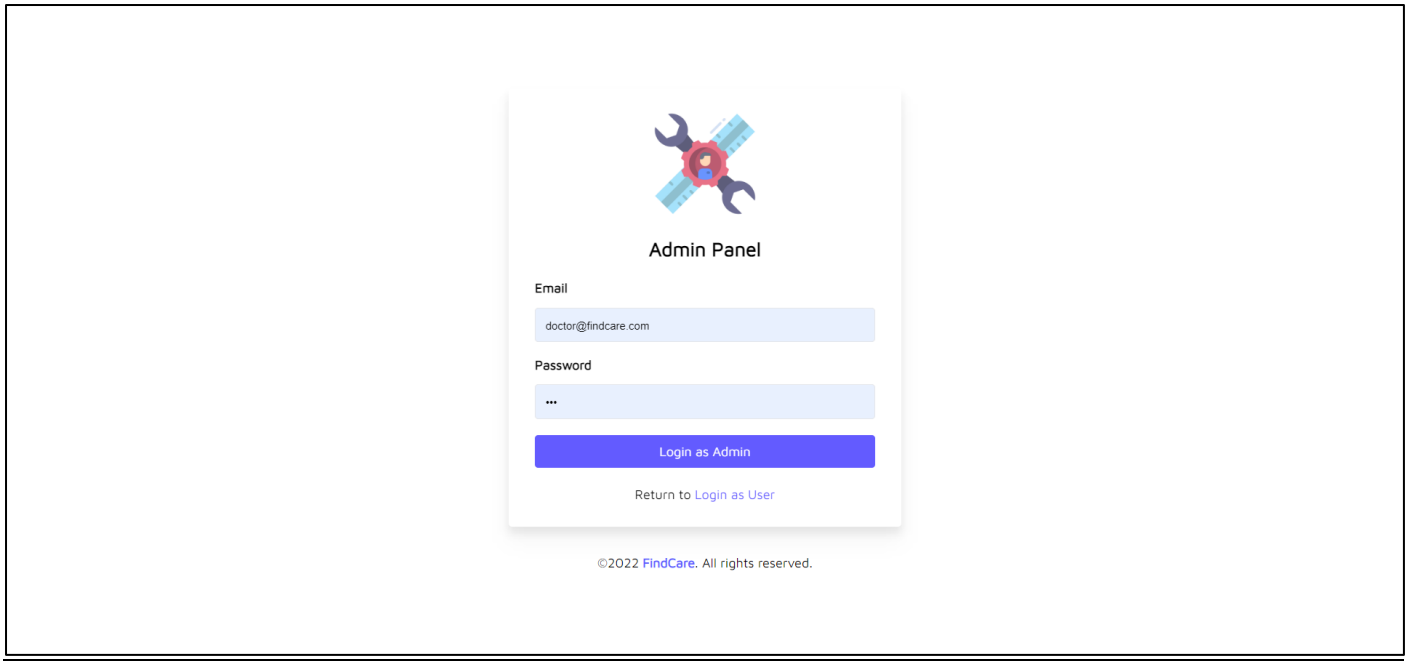


Fig. 5. Admin Login Page

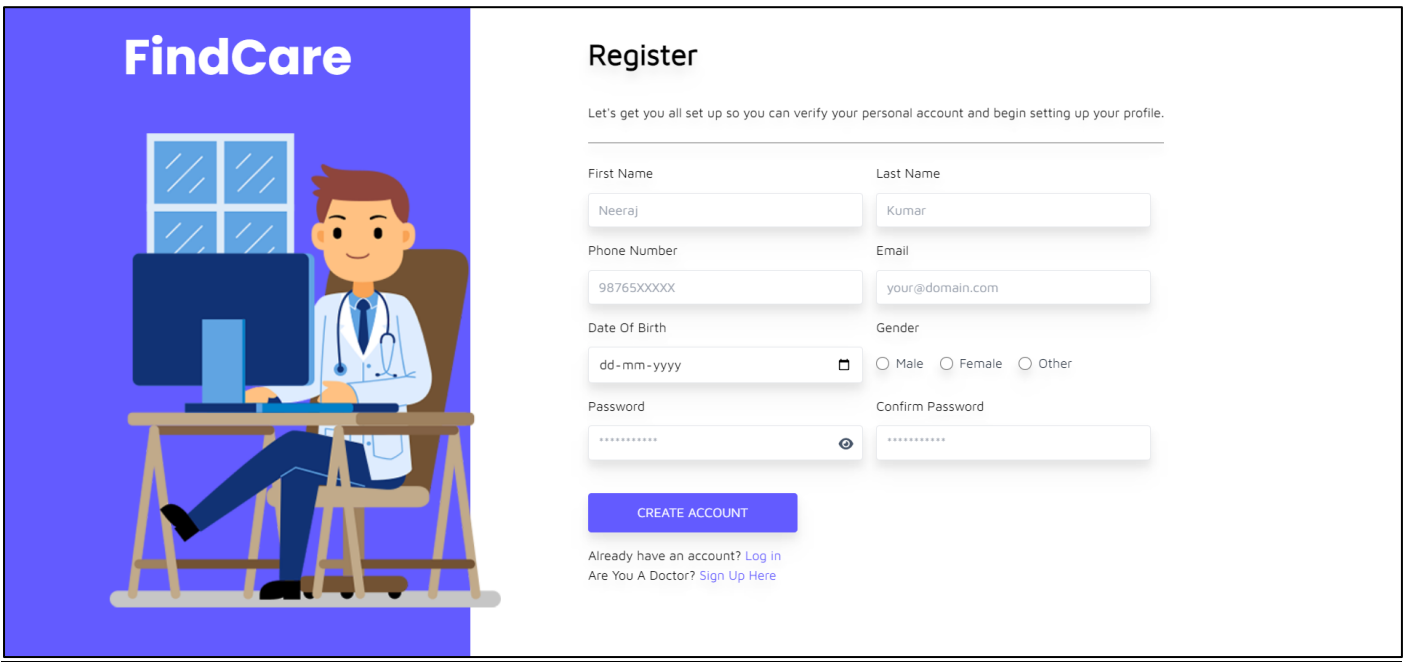


Fig. 6. User Registration Page

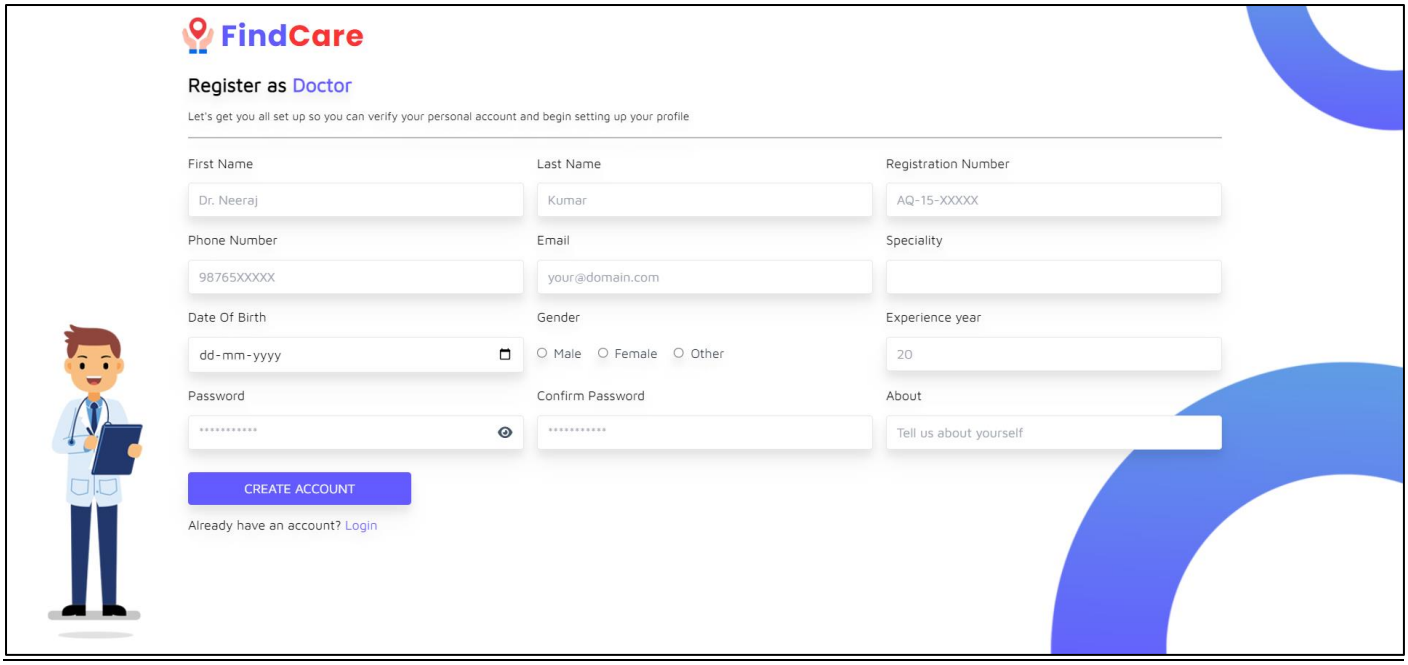


Fig. 7. Doctor Registration Page

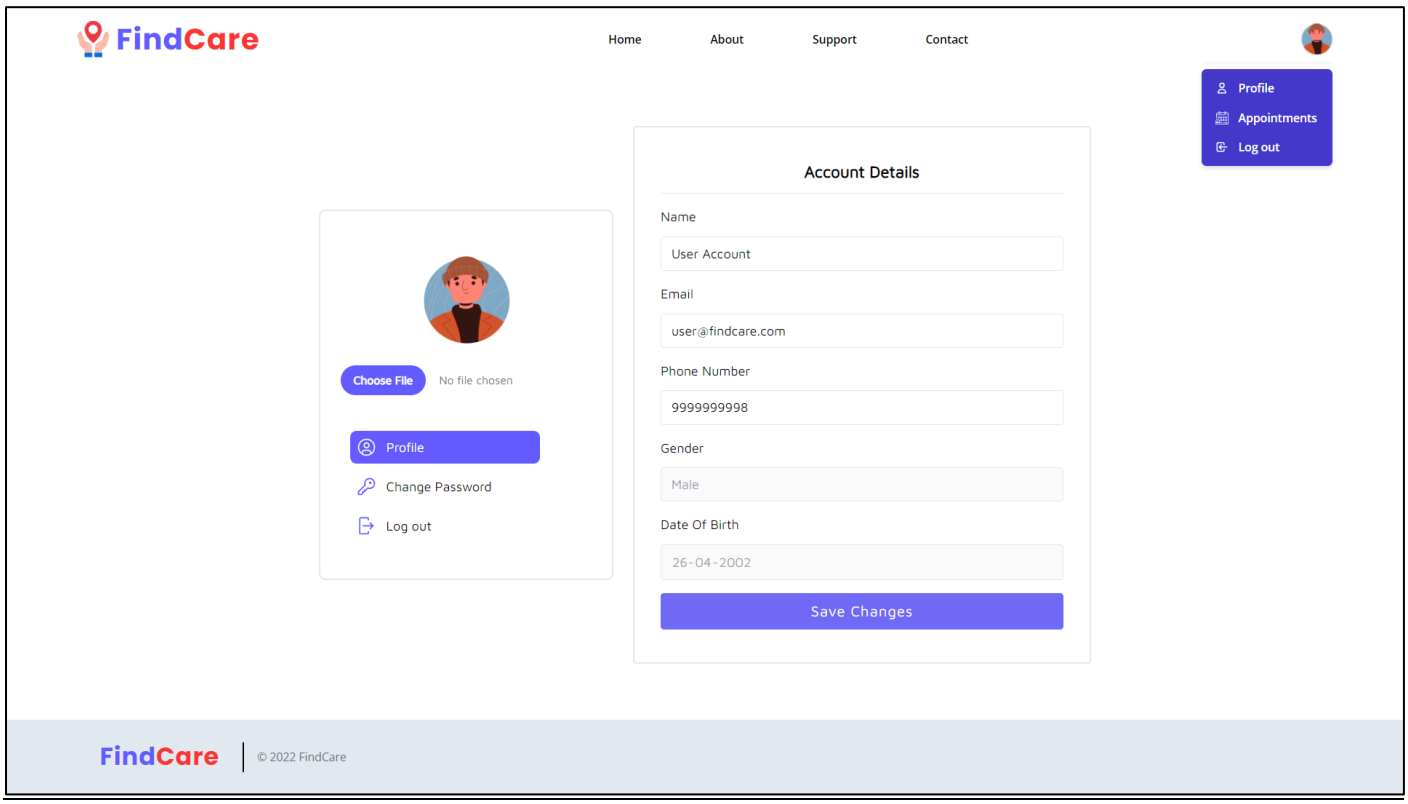


Fig. 8. User Profile Page

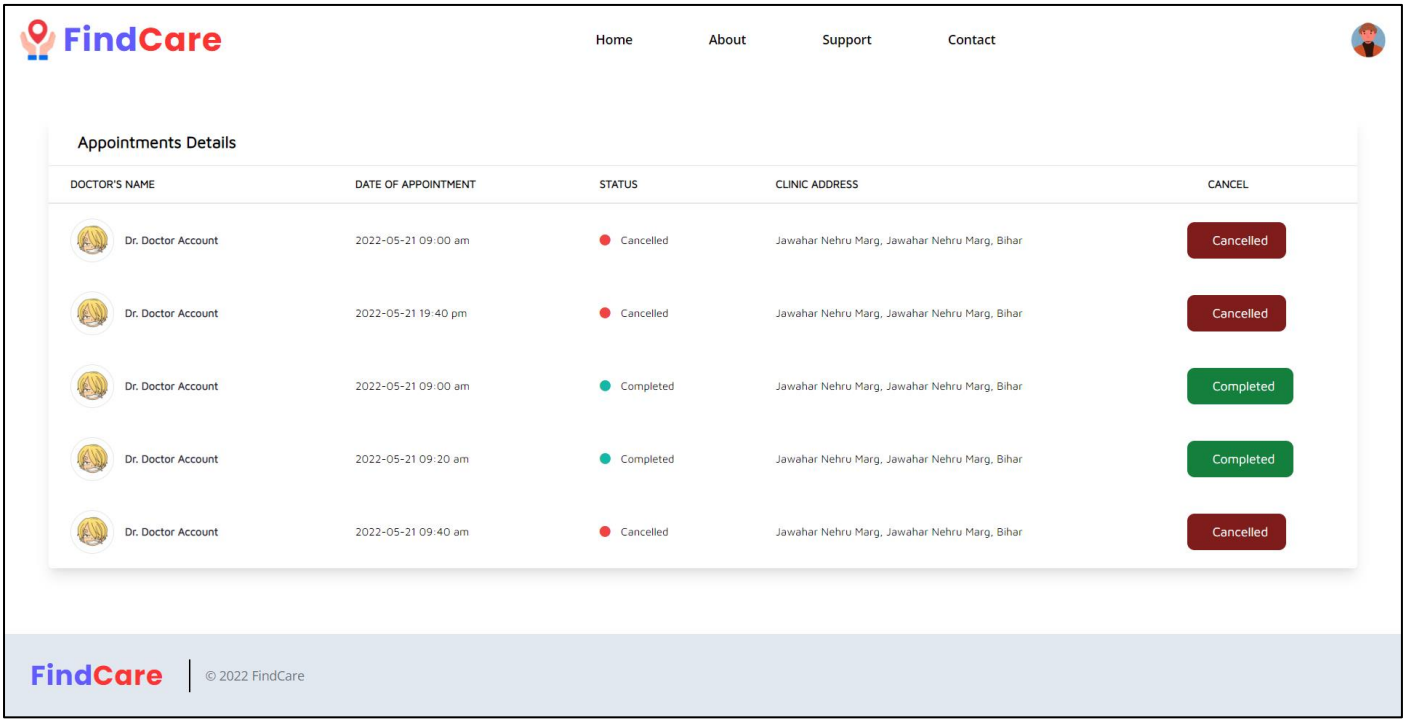


Fig. 9. User Appointment Page

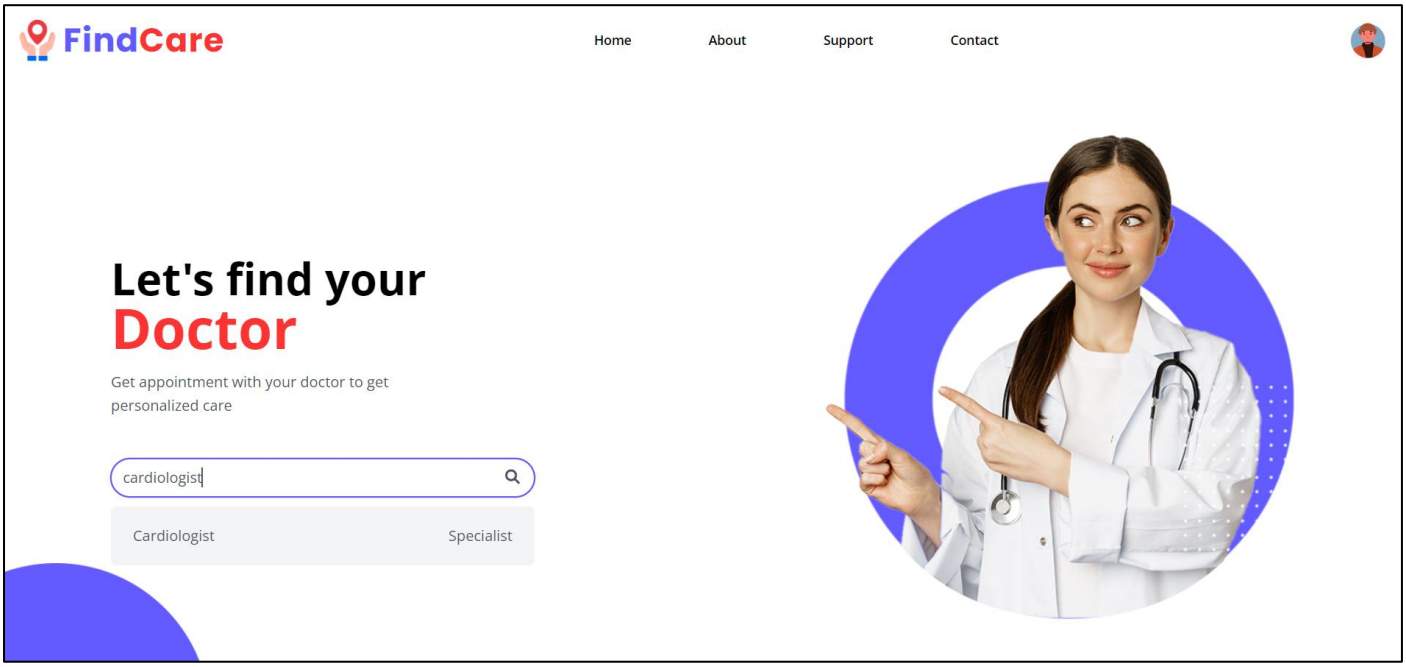


Fig. 10. Search Section

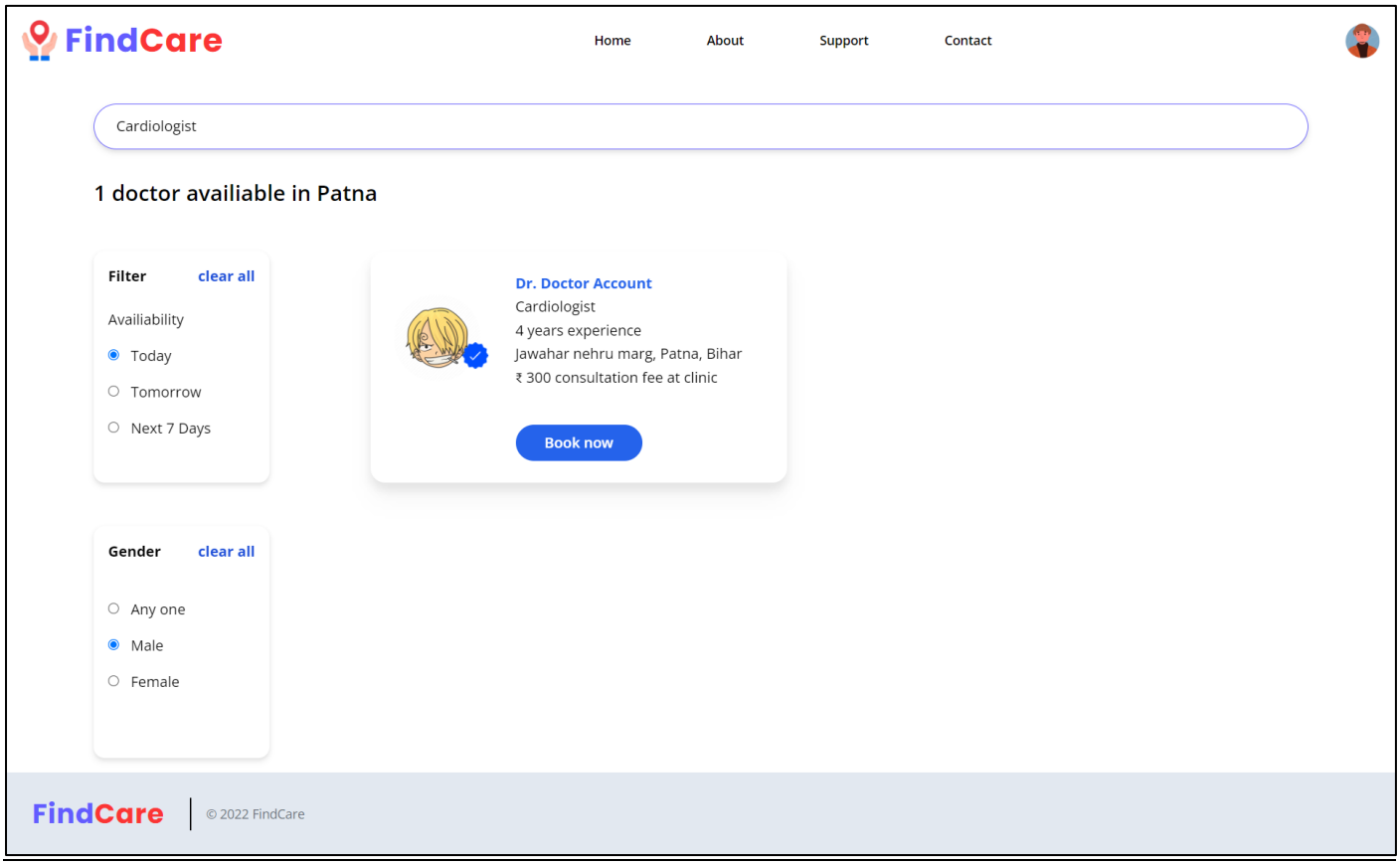


Fig. 11. Doctor Search Result

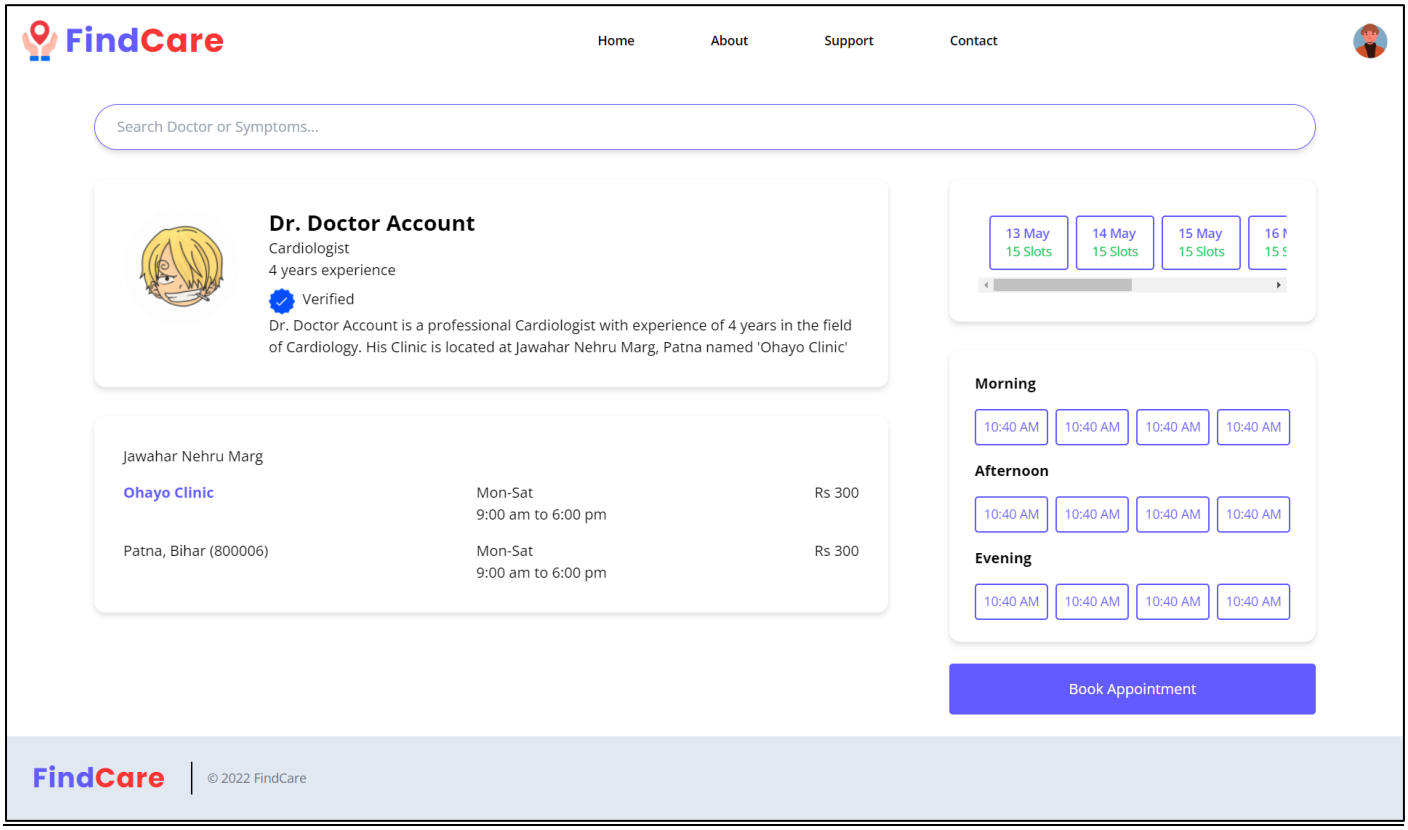


Fig. 12. Book Appointment Page

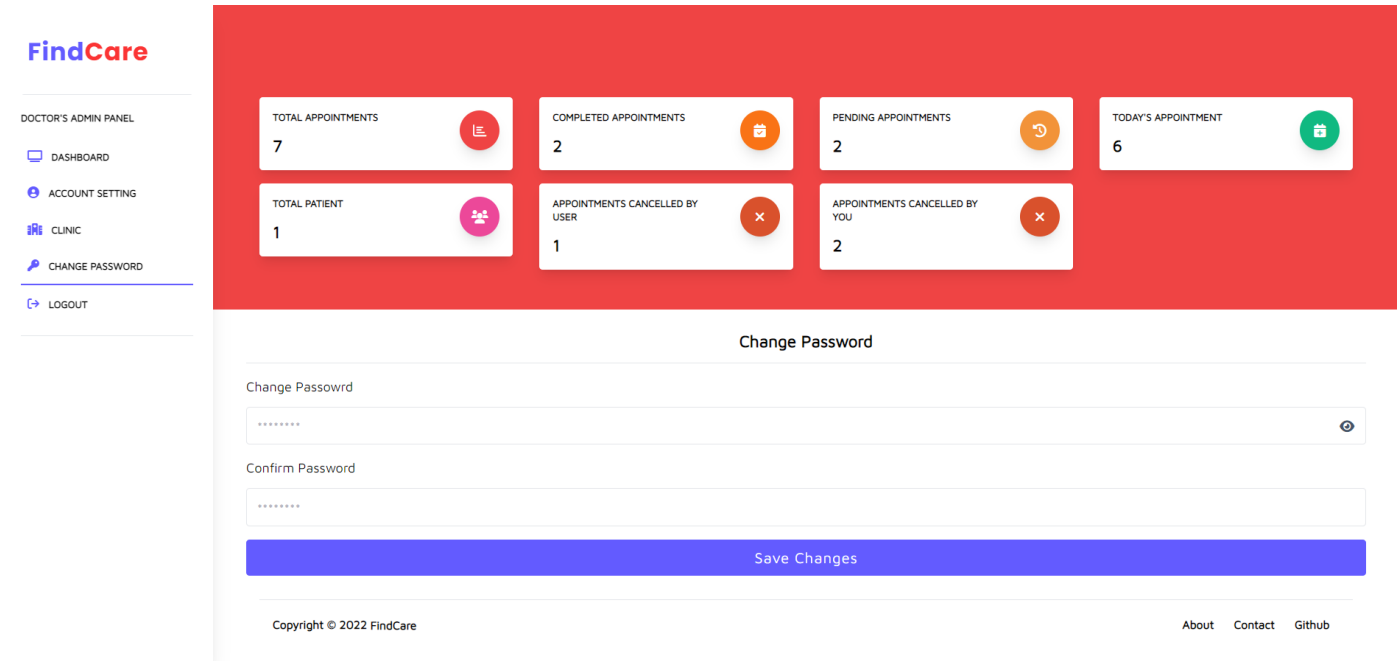


Fig. 13. Doctor Dashboard – Change Password

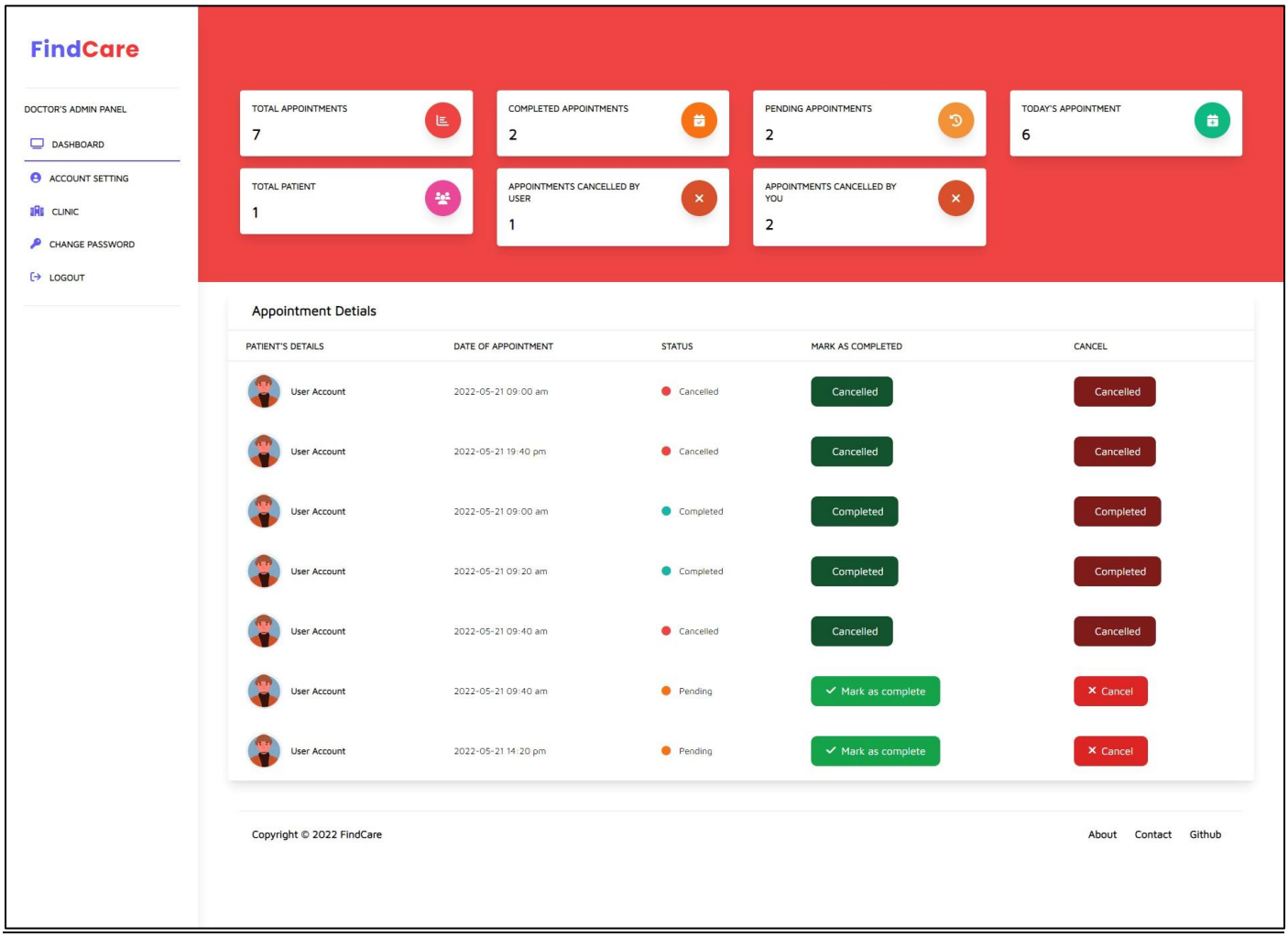


Fig. 13. Doctor Dashboard

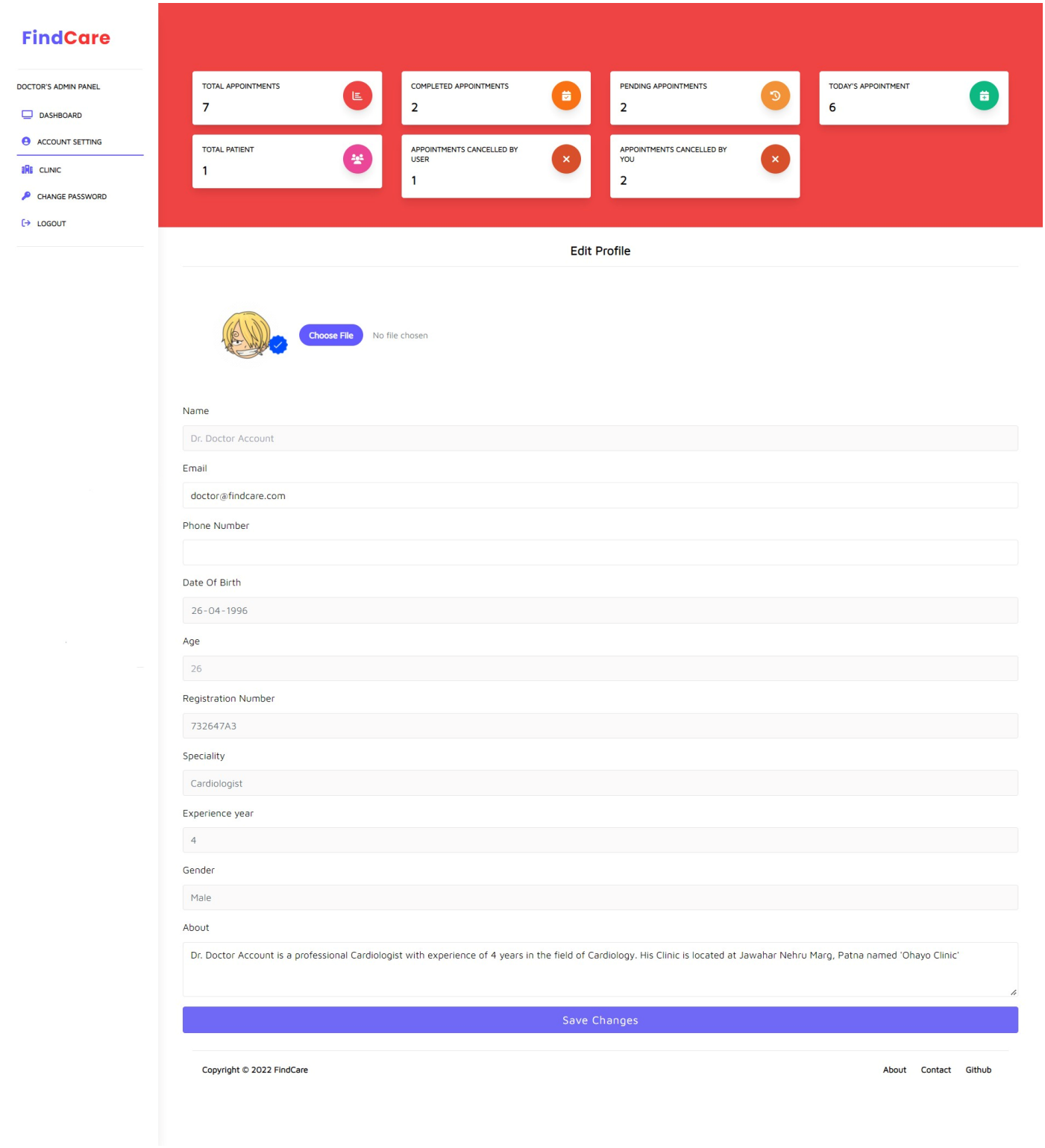


Fig. 14. Doctor Dashboard – Edit Profile

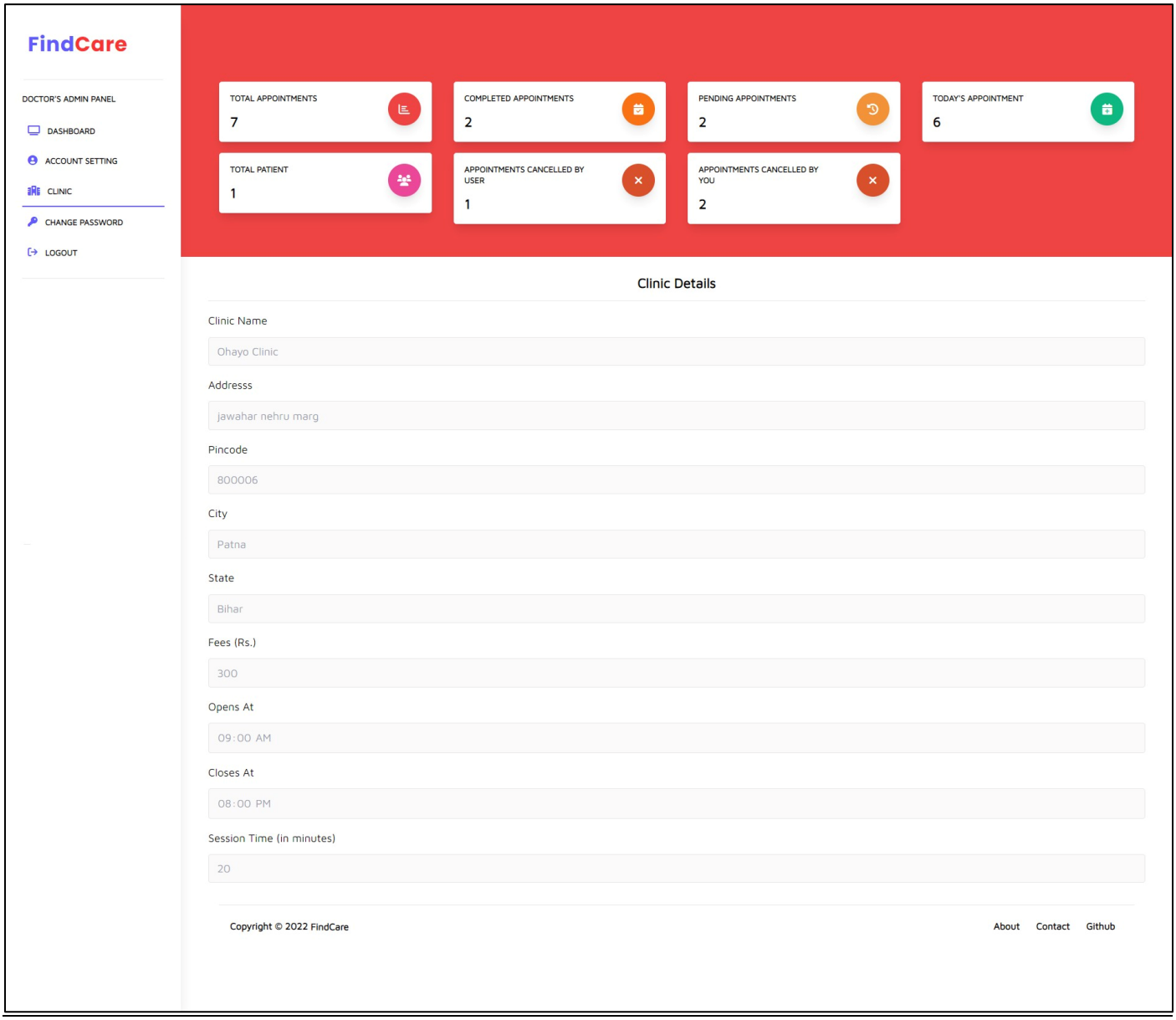


Fig. 15. Doctor Dashboard – Clinic Page

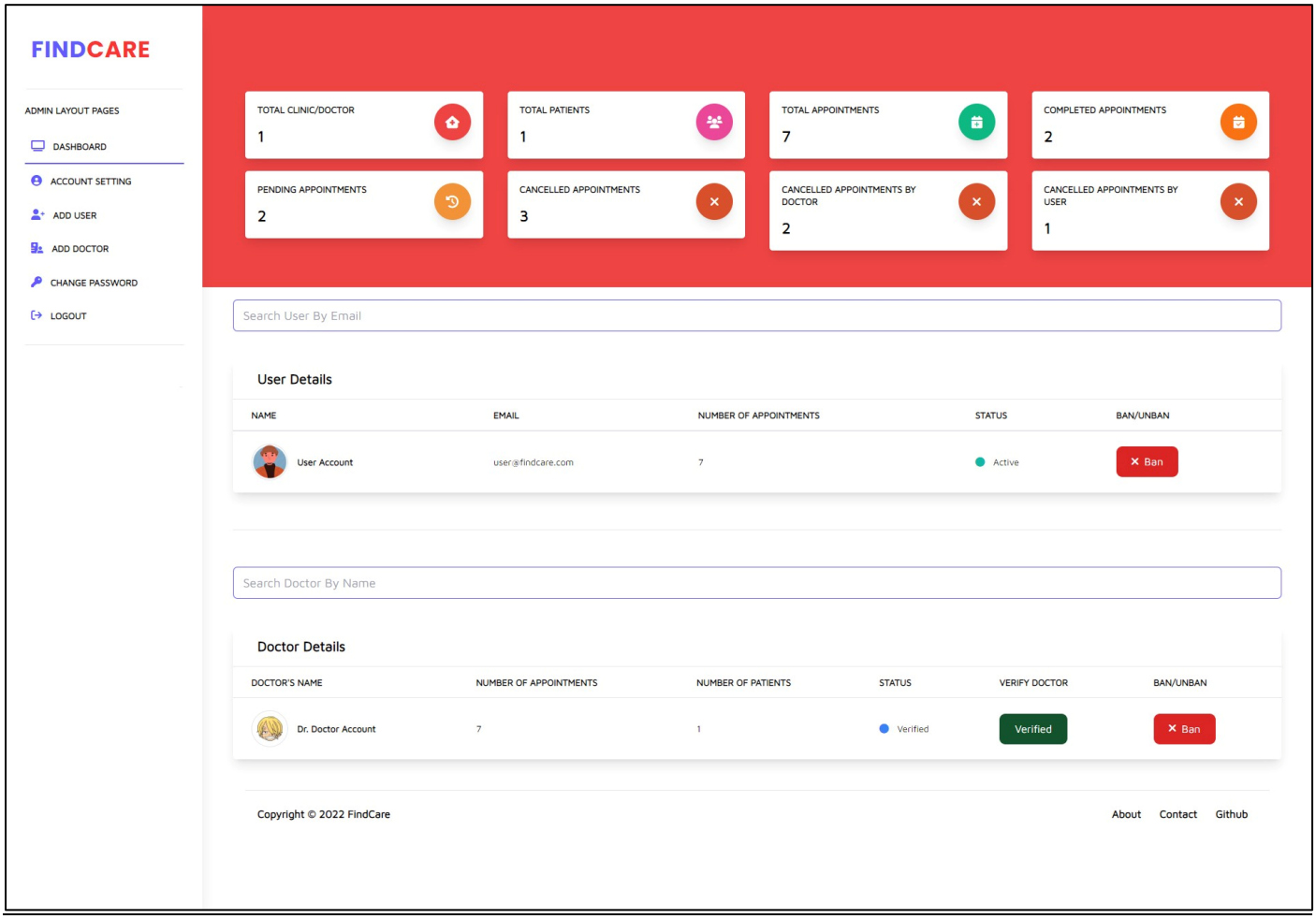


Fig. 16. Admin Dashboard

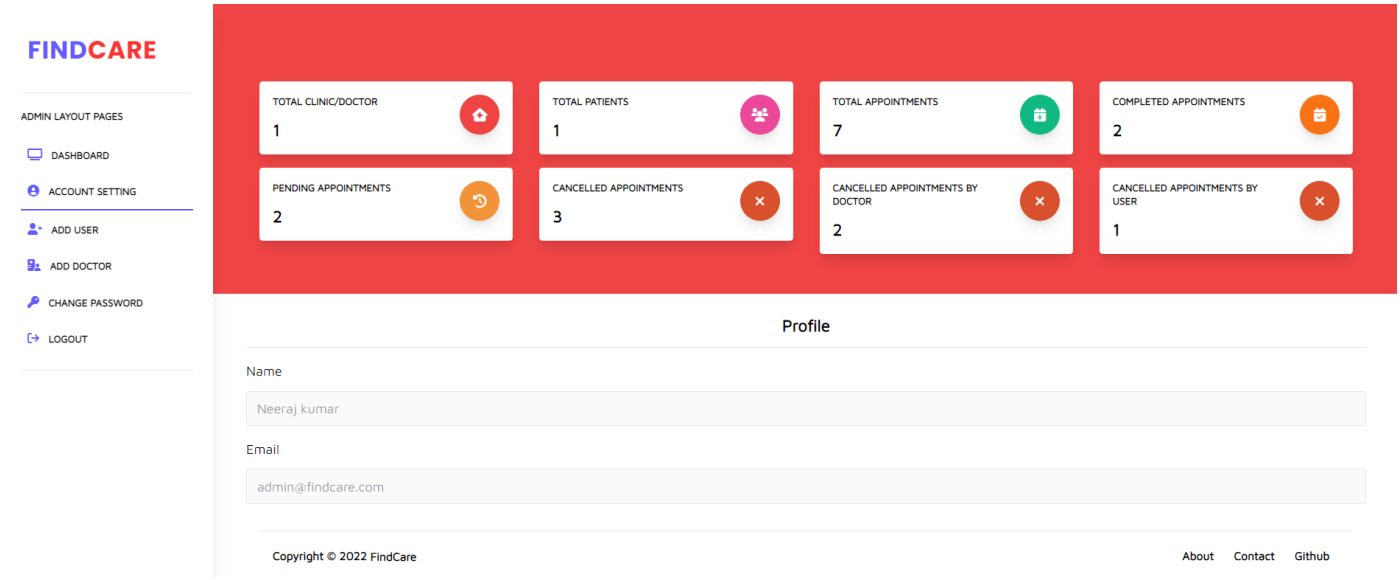


Fig. 17. Admin Dashboard – Profile

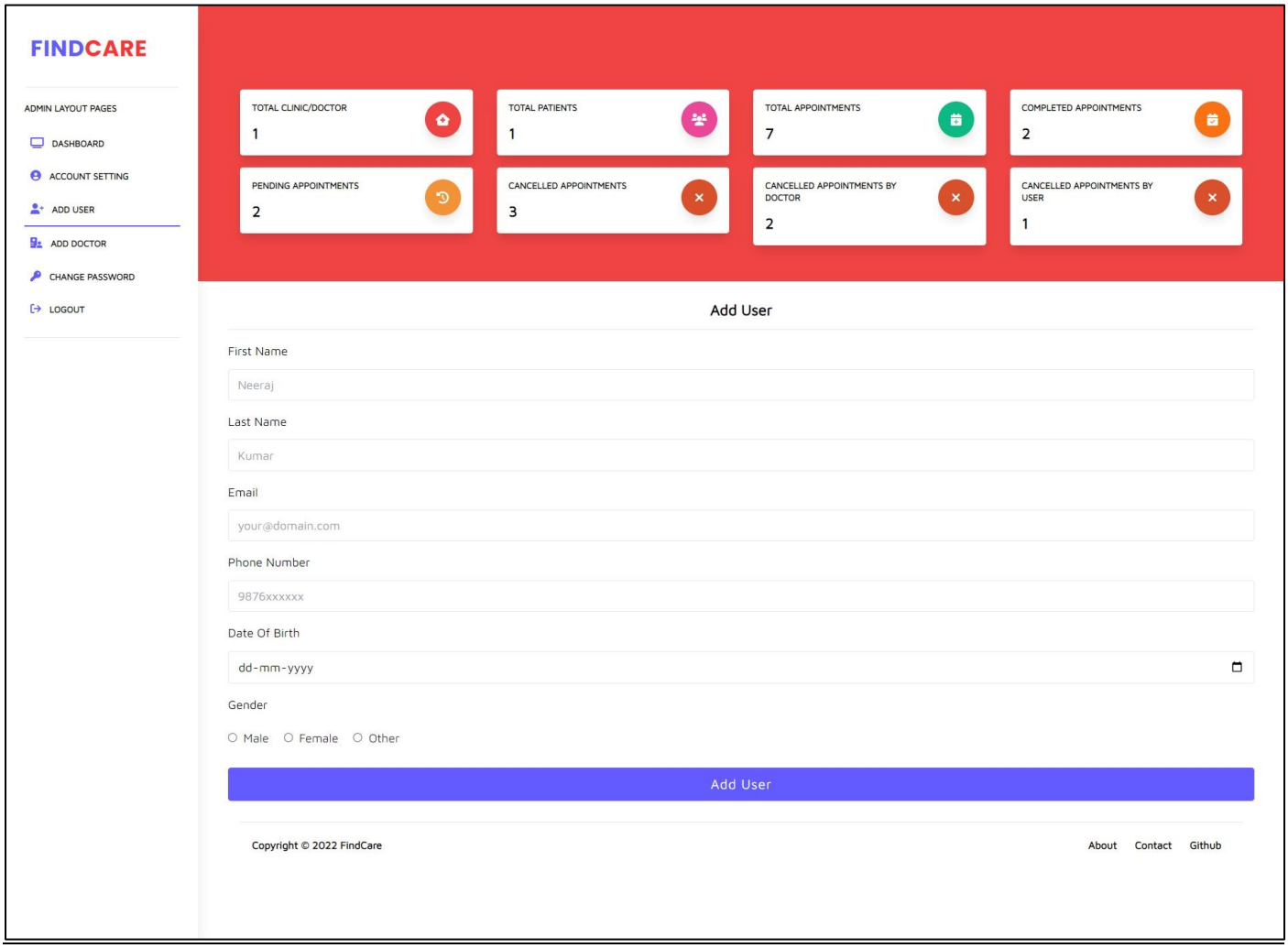


Fig. 17. Admin Dashboard – Add User

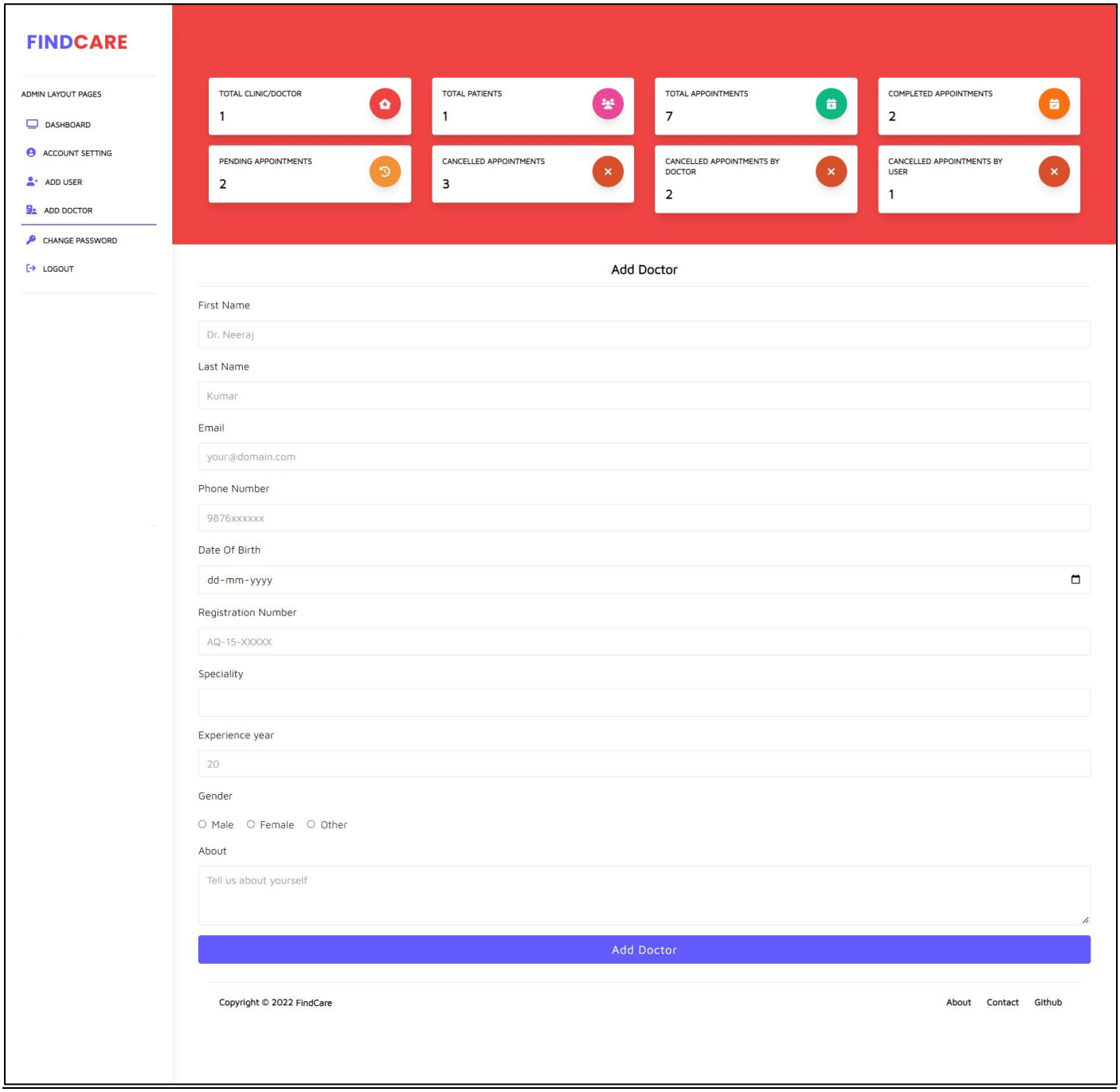


Fig. 17. Admin Dashboard – Add Doctor

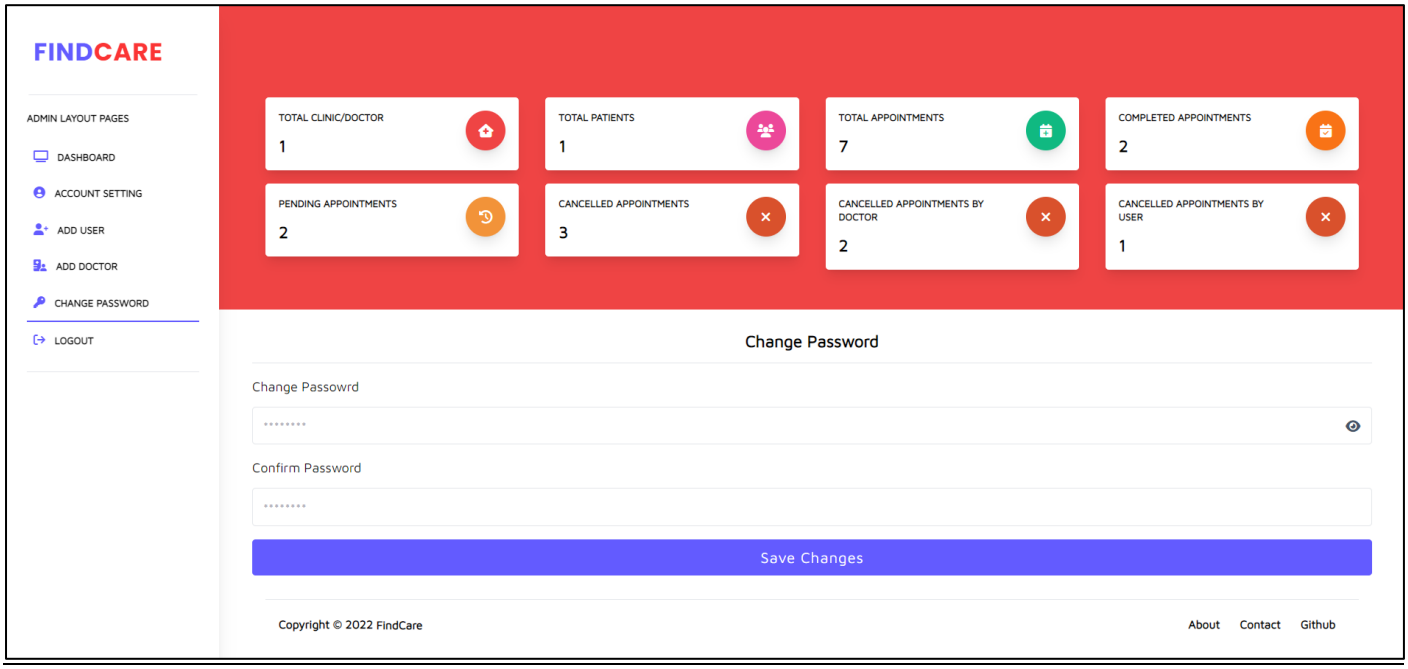


Fig. 18. Admin Dashboard – Change Password

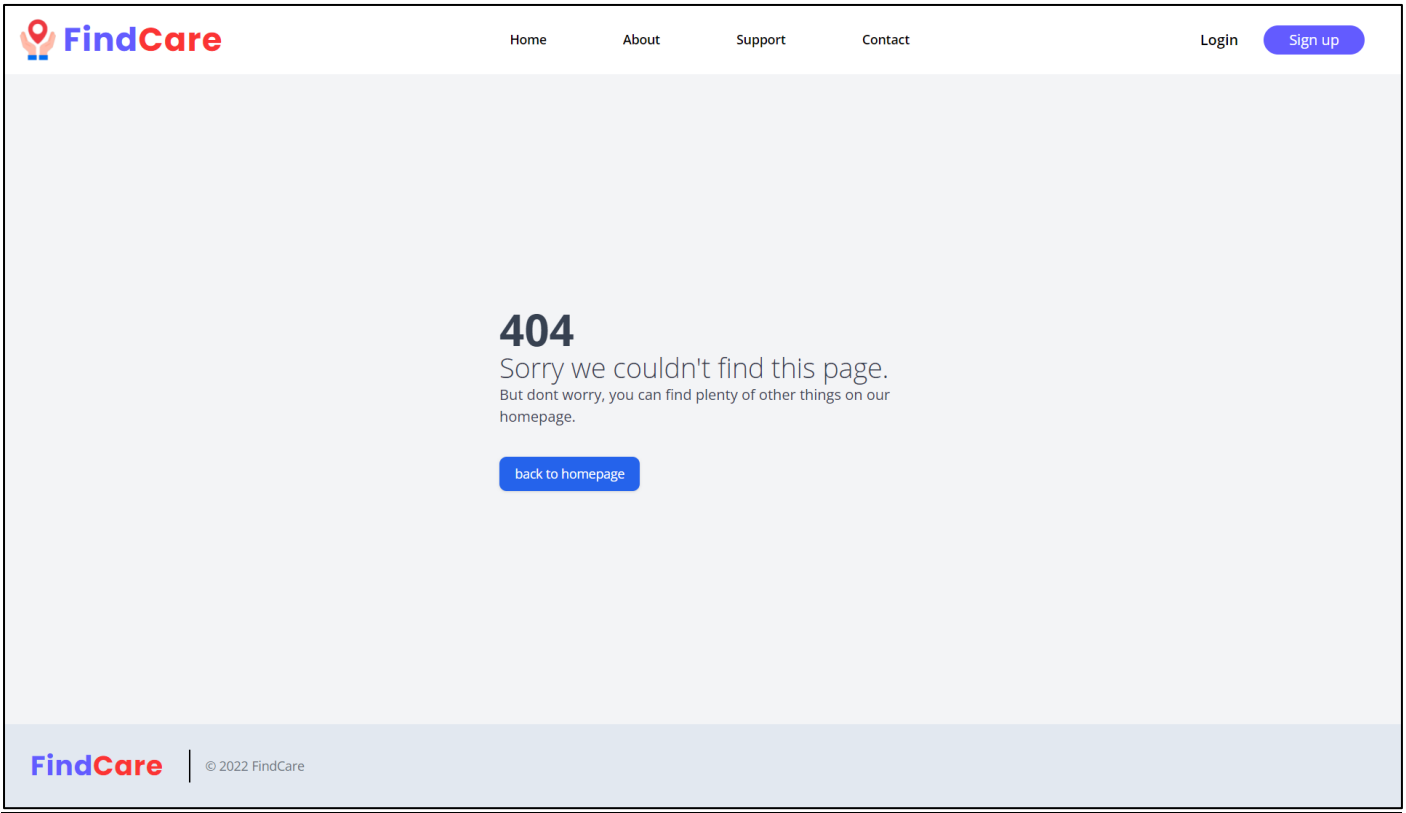


Fig. 19. Error 404 Page

❖ *UI Implementation (For Email)*

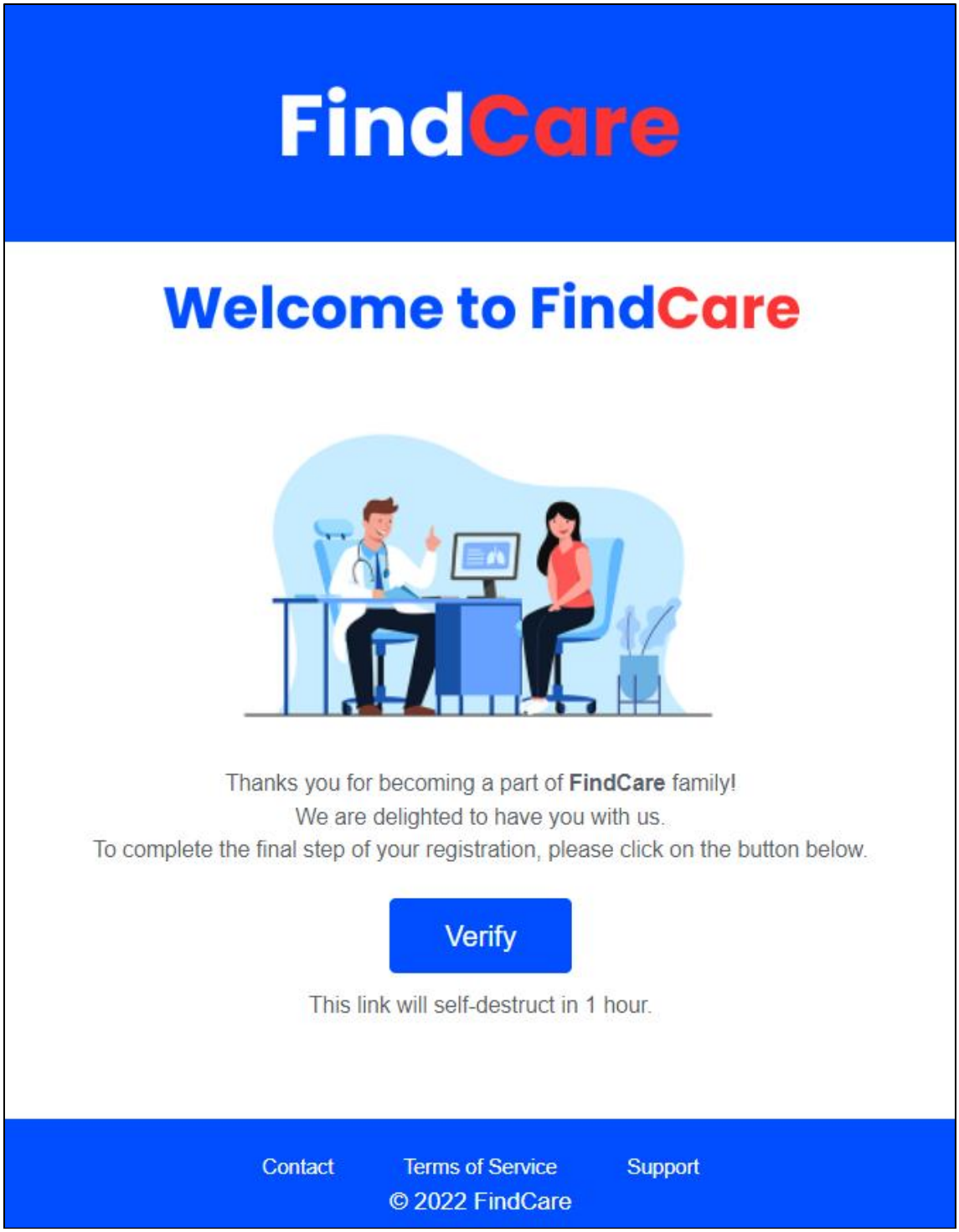


Fig. 20. Welcome E-Mail

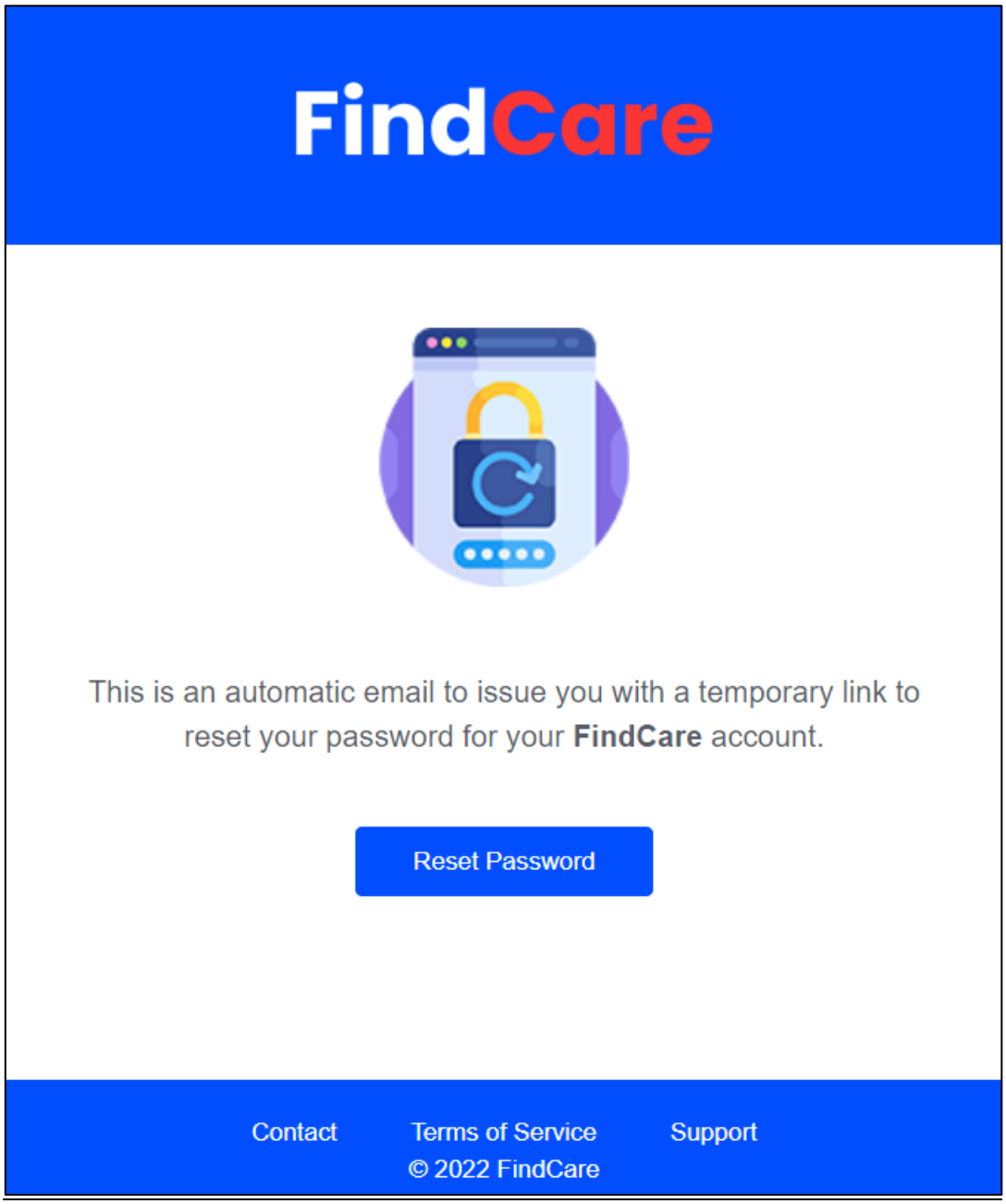


Fig. 21. Reset Password E-Mail

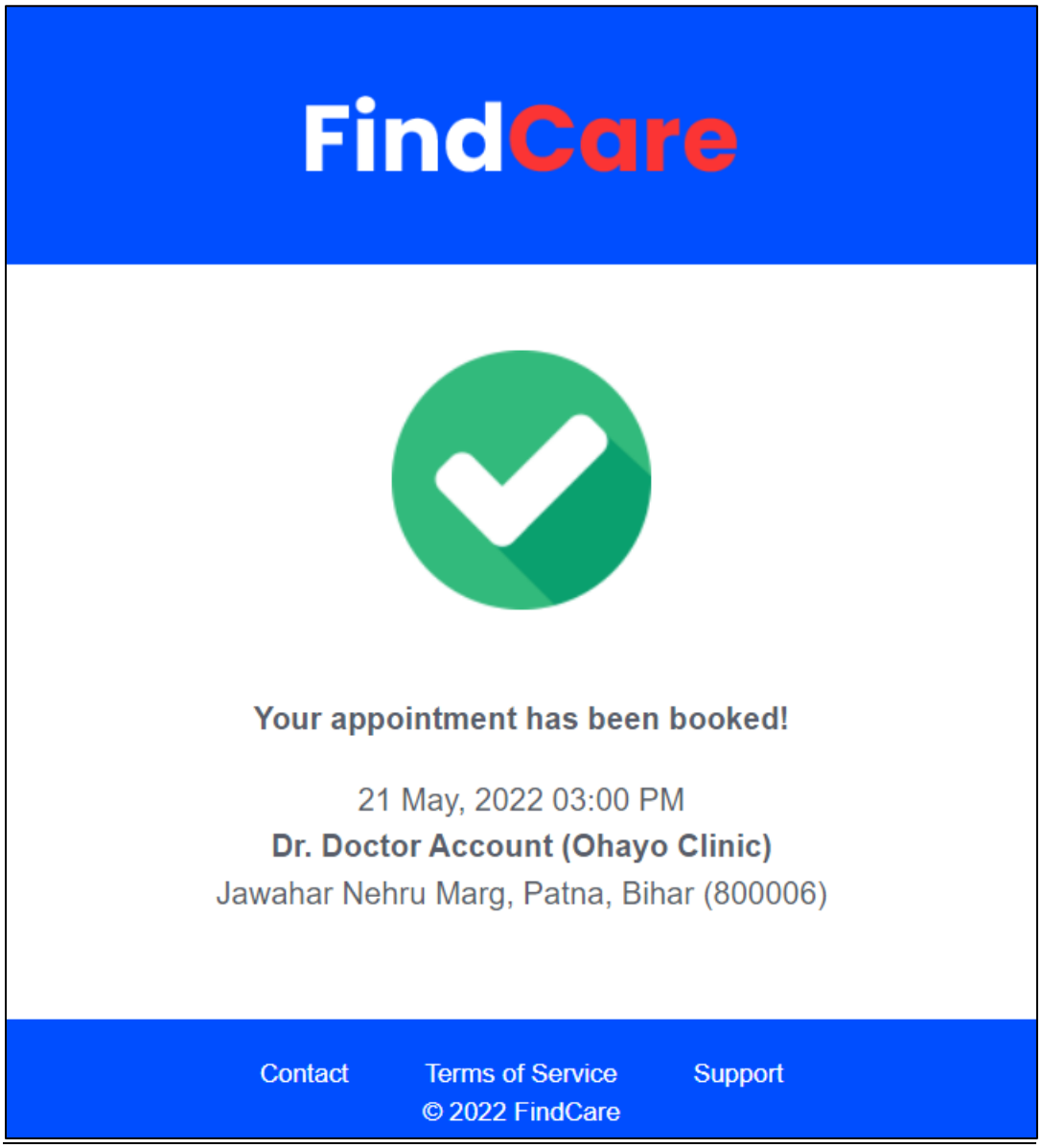


Fig. 22. Appointment Booked E-Mail

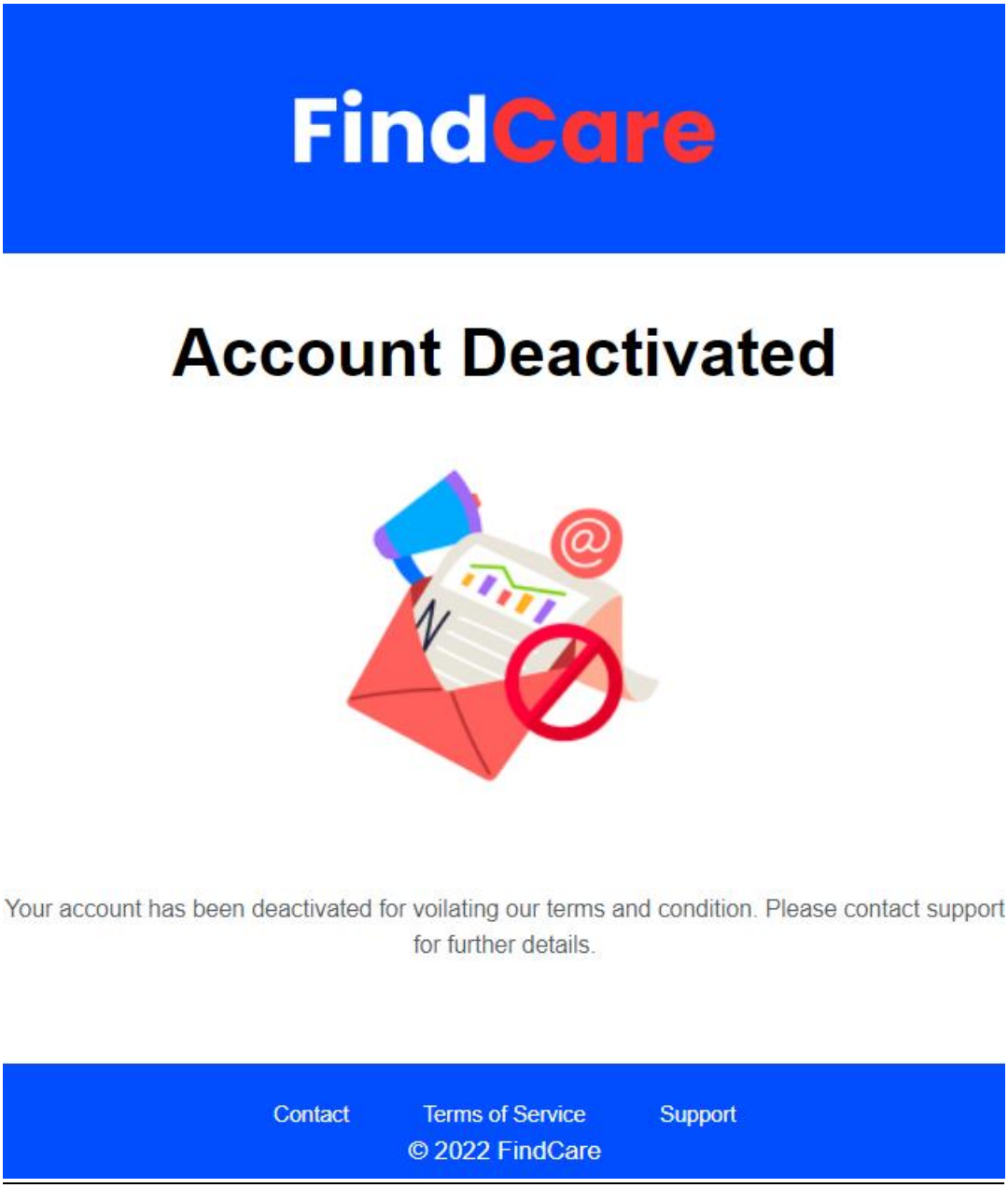


Fig. 23. Account Banned E-Mail

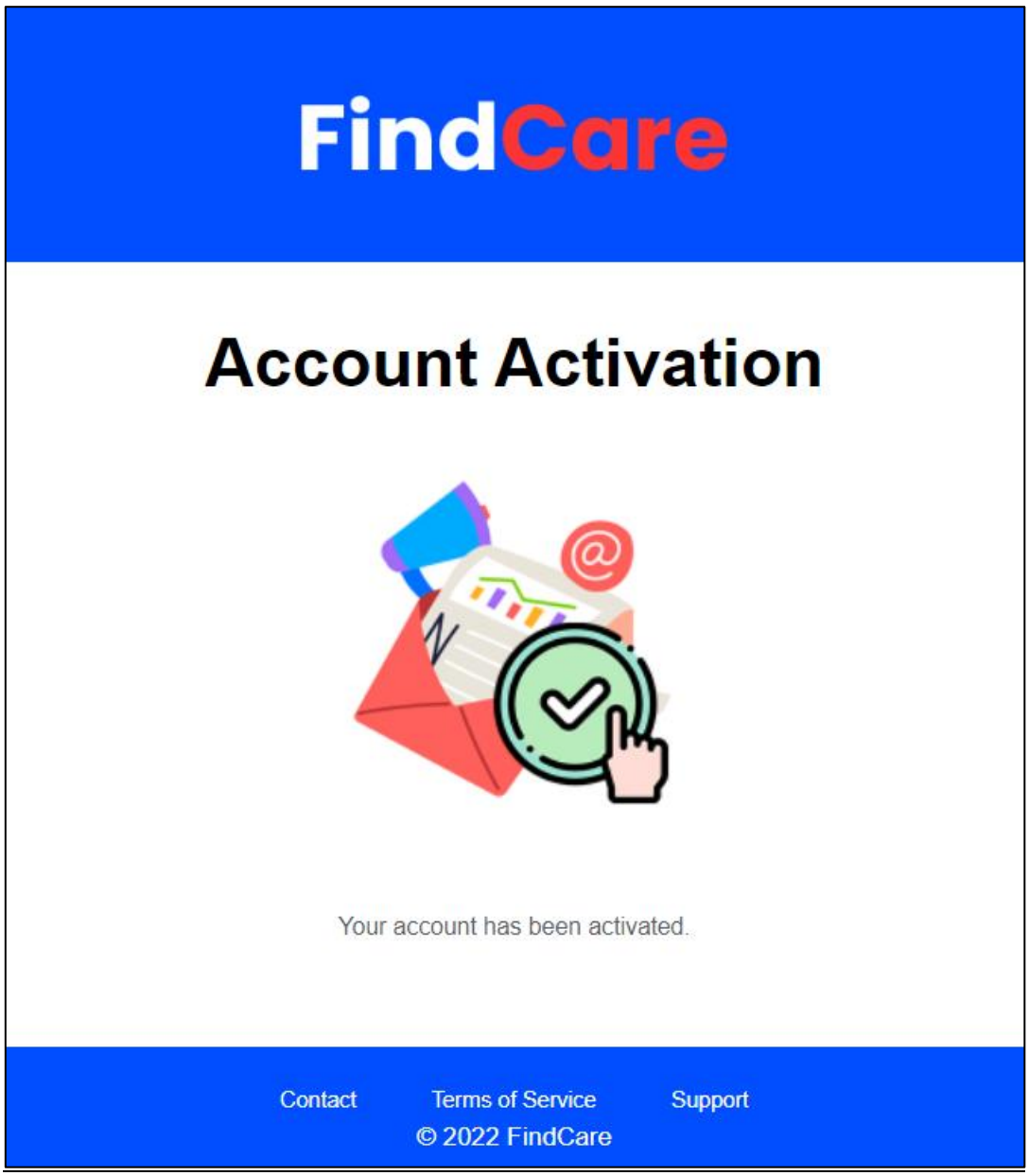


Fig. 24. Account Activated E-Mail

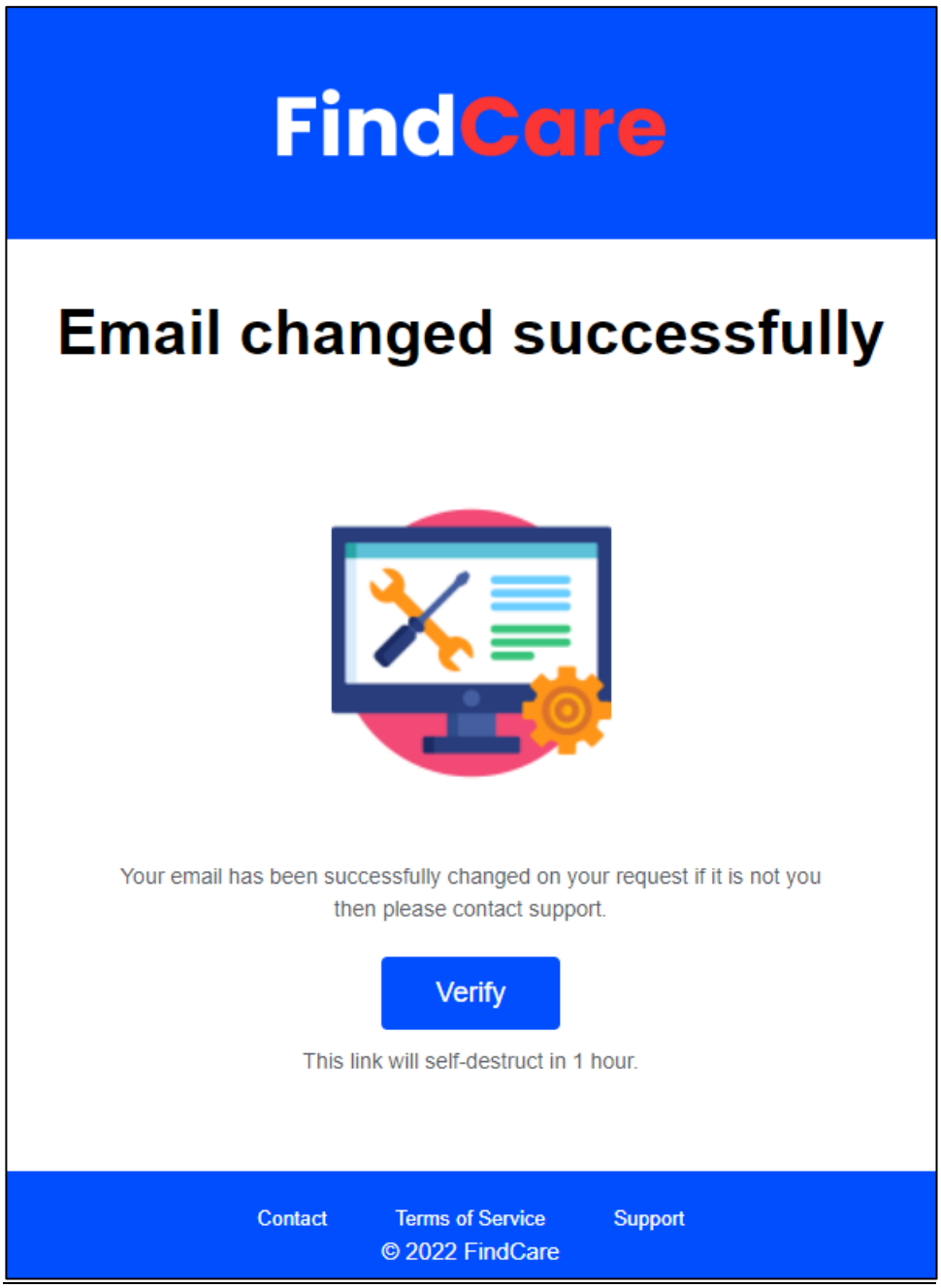


Fig.25. Email ID Change E-Mail

➢System Design

The proposed system **FindCare** aims to follow the two main properties of software architecture: Consistency and Completeness. The components of the **FindCare** would be consistent internally to avoid any contradiction once they are implemented together. System completeness would provide a practical test from a set of models to examine the balance of internal and external behavior. **FindCare** is consisting of three modules:

- Patients
- Doctors
- Administration

Patient's Features

To register new account and to login and logout from the system. Allows to view doctors list according to symptoms as well as doctor's name.

Users can book and cancel an appointment with doctor. Once an appointment is booked, system automatically sends Email to the Patient.

Doctor's Features

Doctors would be to view patient details and to update doctor's profile details. The system would allow doctors to view appointment records and to update appointment record. For example: status of appointment from "Pending" to "Completed" and doctor can approve or reject appointment.

The system would be designed that would allow clinical staffs to view patient details and to update doctor's details. Management staff would be allowed to view appointment records as well as to update appointment record.

Administration Features:

Administrative staff would be allowed to register new user in the system such as: Patient, Doctors. Admin can ban patients and doctors to avoid misuse of system. Only verified doctors can take appointment so admin verifies the details of doctor and approve them to get “Verified” status.

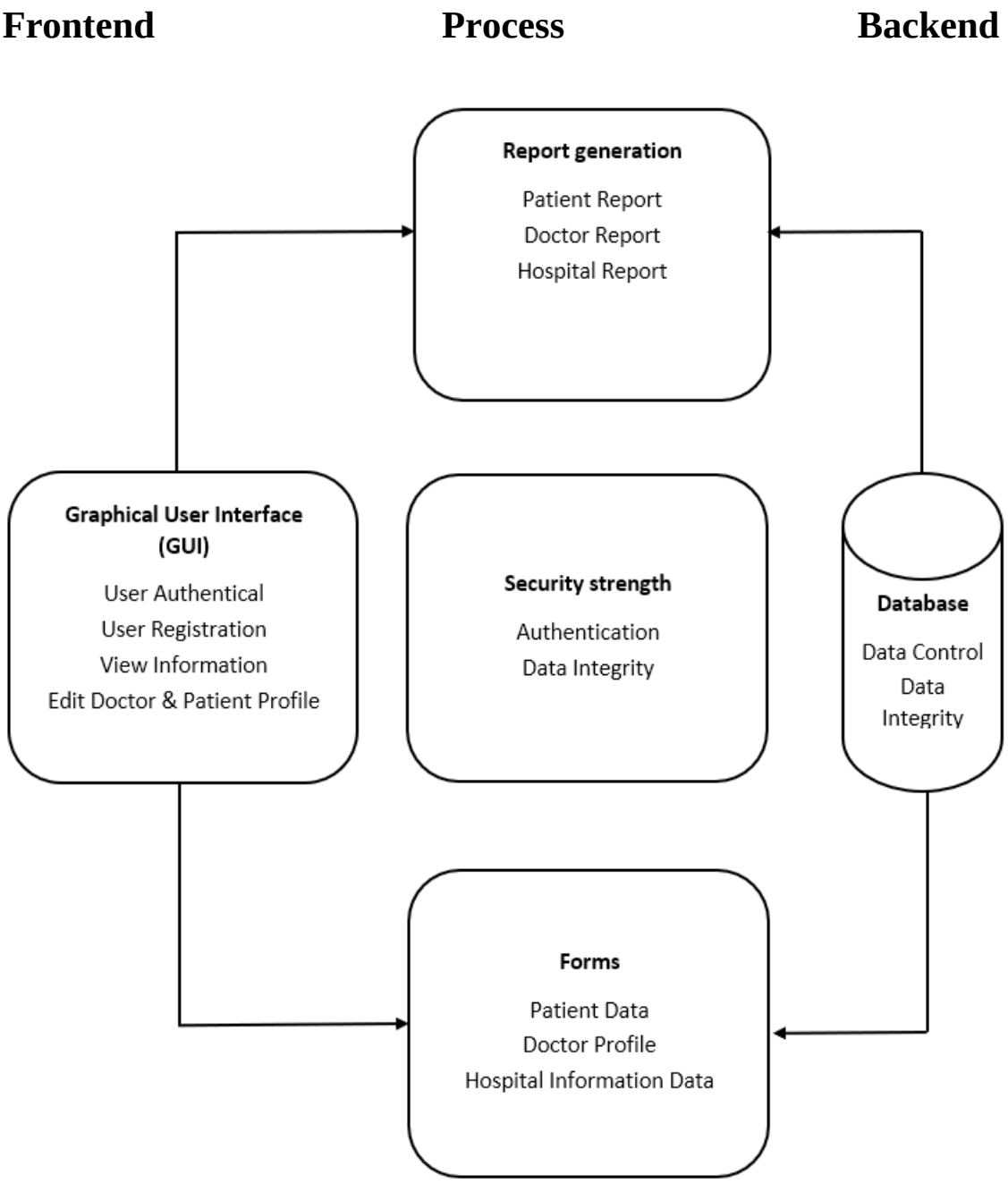


Fig. Simple Patient Scheduling Architecture

➢ Data Modeling

1. User

Fields	Data Type	Constraints
id	str	primary key, increment
name	varchar	not null
email	varchar	unique
phone	varchar	unique, not null
gender	varchar	not null
dob	date	not null
age	int	not null
password	varchar	not null
profile_image	varchar	not null
appointments	relationship (appointments)	not null
created_at	datetime	default: `now()`
updated_at	datetime	default: `now()`
is_active	bool	default: `false`
is_banned	bool	not null, default: false
when_banned	datetime	null

2. Doctor

Fields	Data Type	Constraints
id	str	primary key, increment
name	varchar	not null
email	varchar	unique
phone	varchar	unique, not null
password	varchar	not null
gender	varchar	not null
dob	date	not null
age	int	not null
profile_image	varchar	not null
about	varchar	not null
experience_years	varchar	not null
speciality	varchar	not null
registration_number	varchar	not null, unique
is_verified	bool	default: `false`
slug	str	not null, unique
is_active	bool	default: `false`
created_at	datetime	default: `now()`
updated_at	datetime	default: `now()`
is_banned	bool	not null, default:

		`false`
when_banned	datetime	null

3. Clinic

Fields	Data Type	Constraints
id	str	primary, increment
doctor_id	str	ref: - D.id
name	varchar	not null
fees	str	not null
session_time	str	default: `15`
opens_at	time	not null
closes_at	time	not null
slots	int	not null
is_open	bool	default: `false`
address	json	not null
created_at	time	default: `now()`
updated_at	time	default: `now()`

4. Appointment

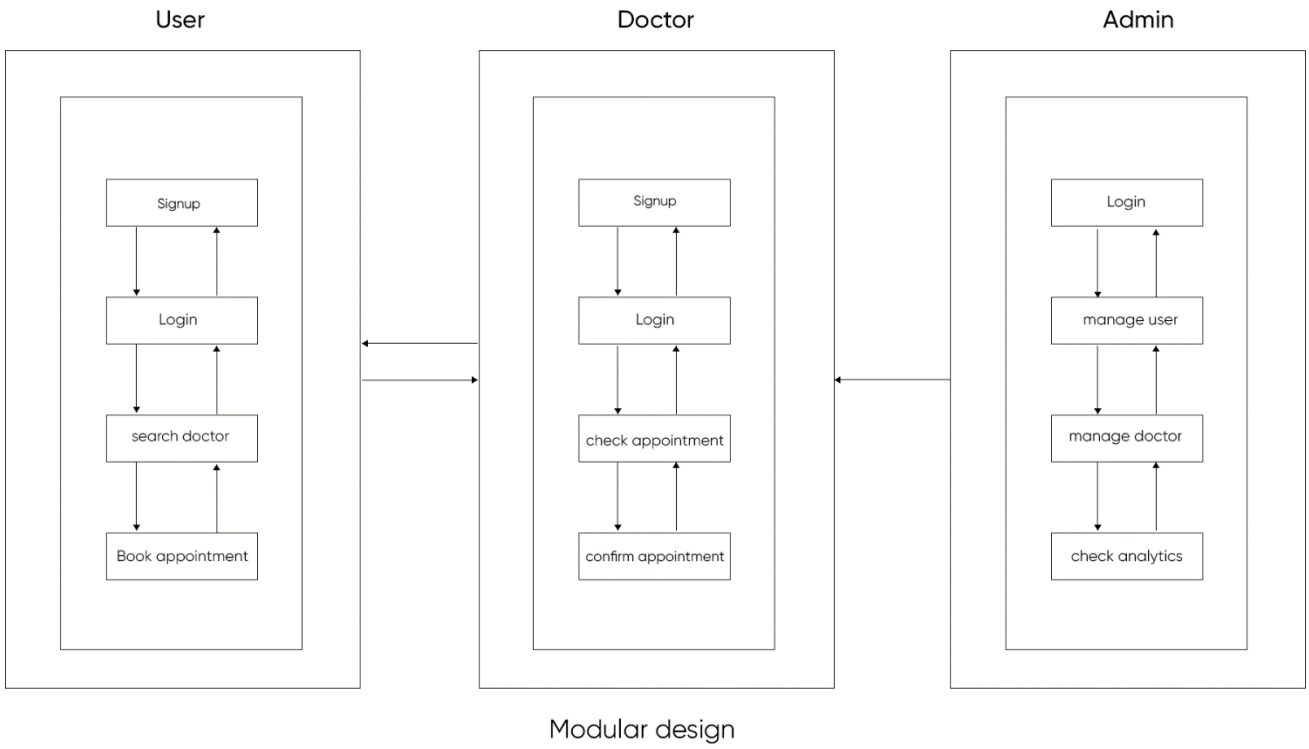
Fields	Data Type	Constraints
id	str	primary key, increment
user_id	str	not null, ref: - U.id
doctor_id	str	not null, ref: - U.id
cid	str	not null, ref: - U.id
clinic_id	str	not null
schedule	datetime	not null
fees_paid	bool	default: `false`
is_completed	bool	default: `false`
is_cancelled	str	default: null
when_cancelled	datetime	default: `now()`
clinic	relationship (clinic)	not null
created_at	int	not null

5. Admin

Fields	Data Type	Constraints
id	str	primary key, increment
name	varchar	not null
email	varchar	not null, unique

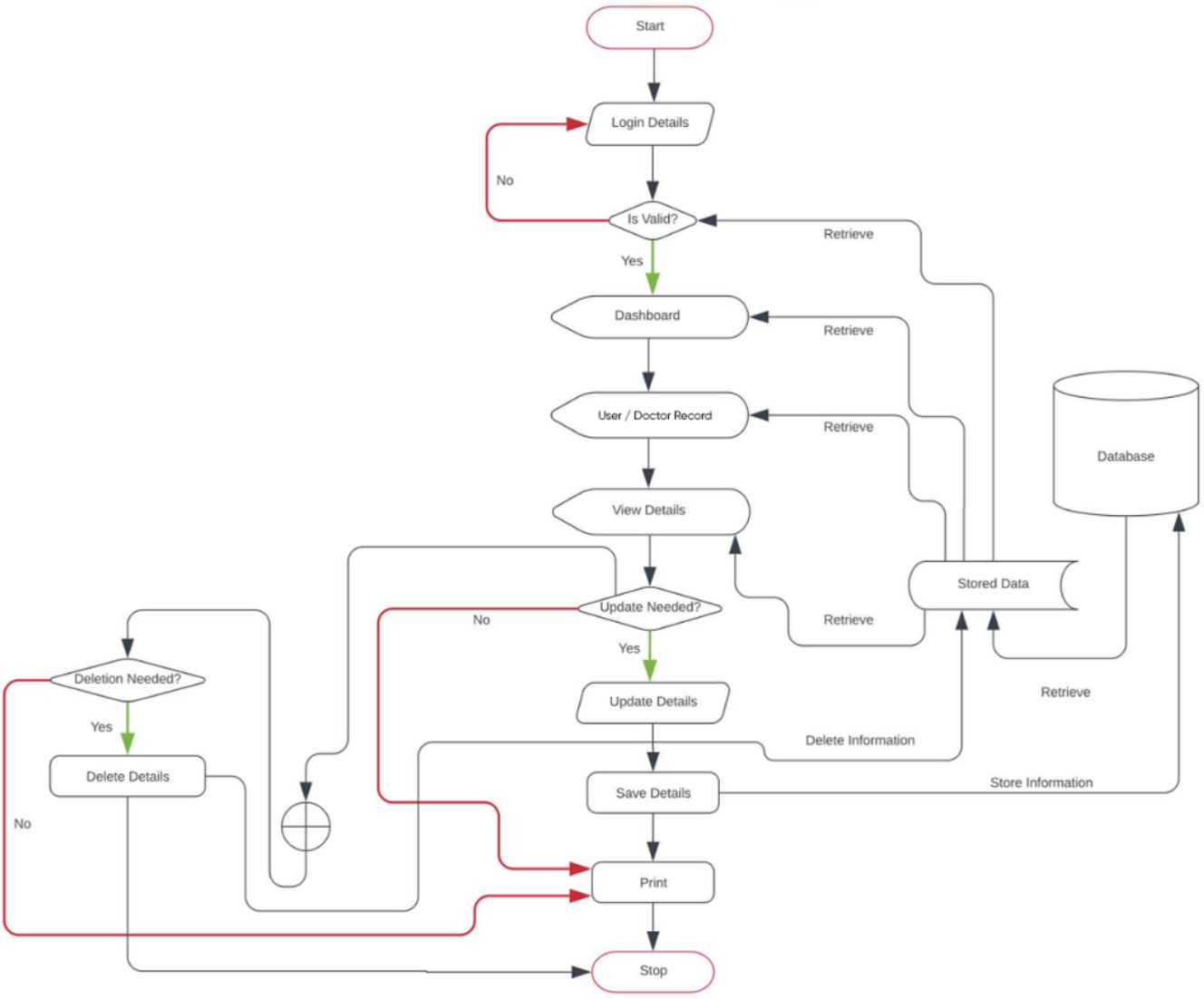
password	varchar	not null
is_super_admin	bool	default: `false
created_at	datetime	default: `now()`
updated_at	datetime	default: `now()`

➢Module Design

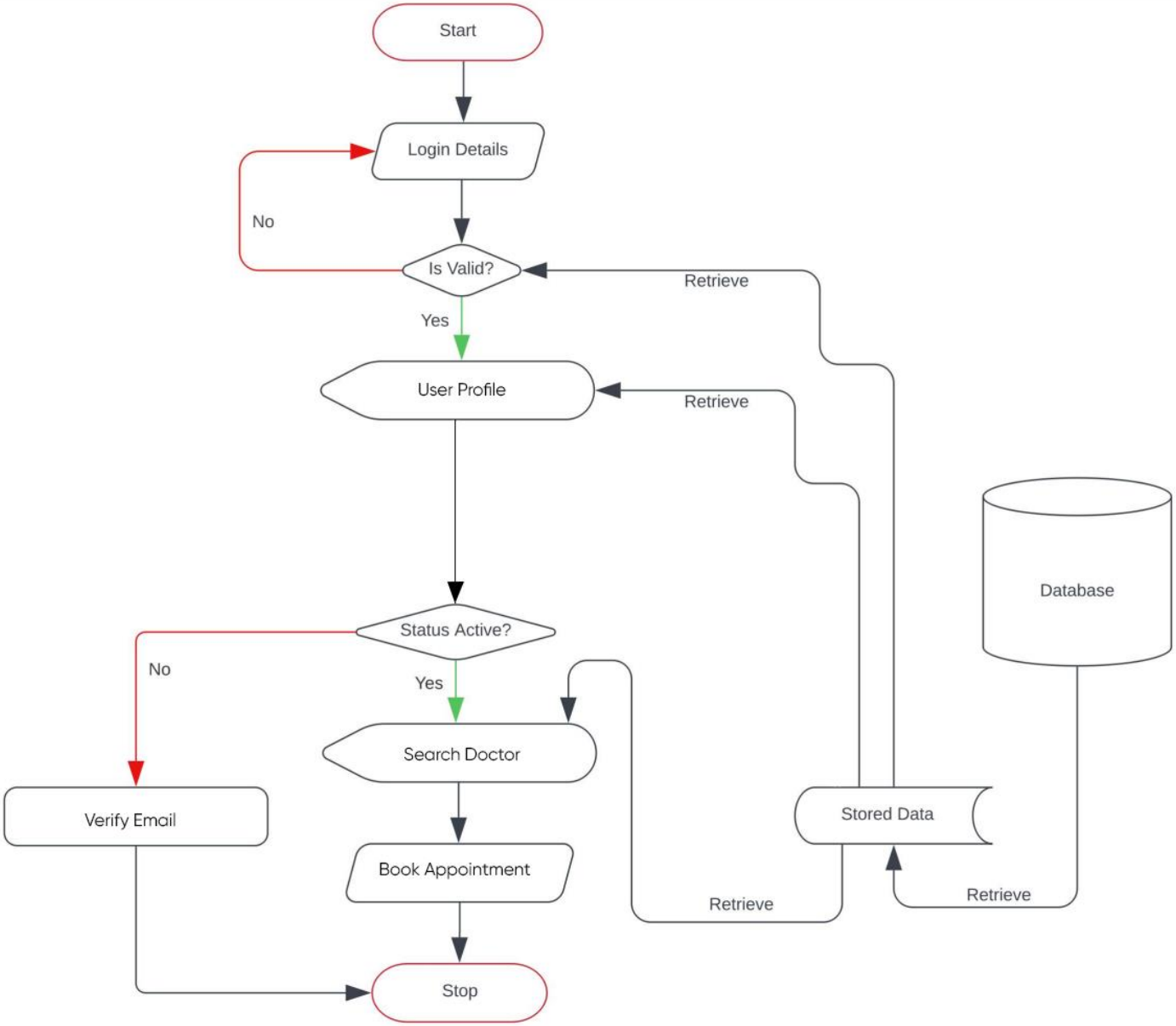


Modular design

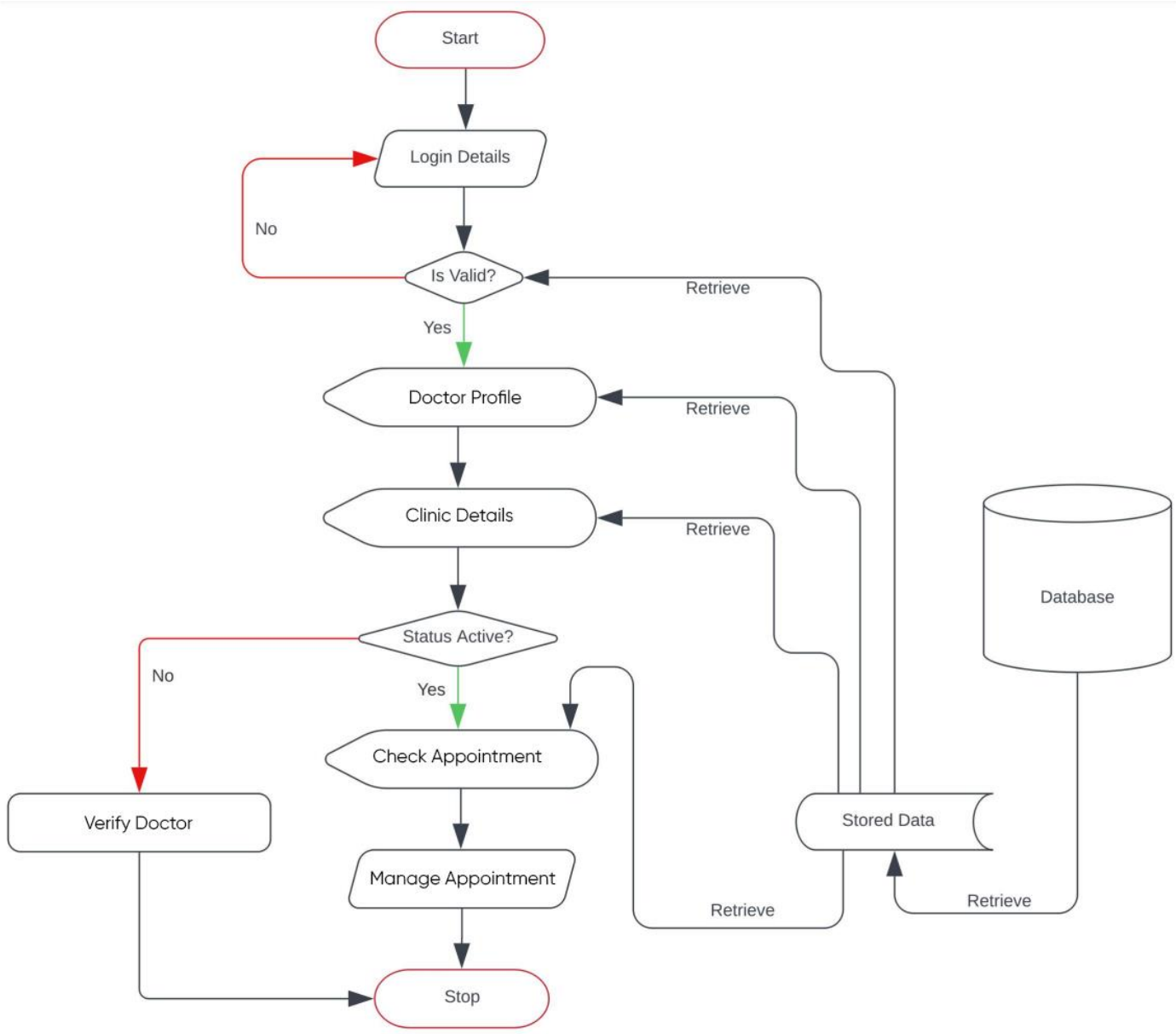
➢Procedural Design



Procedural design for admin



Procedural Design for user



Procedural Design for Doctor

➢ Scheduling

Project scheduling is a mechanism to communicate what tasks need to get done and which organizational resources will be allocated to complete those tasks in what timeframe. A project schedule is a document collecting all the work needed to deliver the project on time.

Program evaluation and review technique (PERT) and the critical path method (CPM) are two project scheduling methods that can be applied to software development. PERT Technique include following network:

1. **Activity duration estimates**

Activity durations are based on estimates made by human beings and are therefore error prone. In PERT, the technique requires three duration estimates for each individual activity, as follows:

- Optimistic time estimate (a): This is the shortest possible time in which the activity can be completed, and assumes that everything must go perfect

- Realistic time estimate (m): This is the most likely time in which the activity can be completed under normal circumstances

- Pessimistic time estimate (b). This is the longest possible time the activity might require and assumes a worst-case scenario.

2. **Activity average and variance**

Based on the three estimates, a weighted average and variance is calculated for each activity duration as a measure of the average duration and the corresponding variability, respectively. The weighted average is equal to (a + 4m + b) / 6 to express that the likeliness that the real activity duration lies close to the realistic estimate (m) is larger than the likeliness that it lies closer to the two extreme values a or b.

The standard deviation is equal to (b - a) / 6 and is based on the principles of a three-sigma interval that states that 99.73% (i.e. almost all) of the observations lie in that interval when the variable is normally distributed. Although it is assumed that the activity duration is beta distributed, the general principle is that the standard deviation assumes that almost all observations (i.e. real durations) will lie between the extreme values a and b.

3. **The central limit theorem**

The main idea of the central limit theorem (CLT) is that the average of a sample of observations drawn from a population with any distribution shape is approximately distributed as a normal distribution if certain conditions are met. More precisely, the central limit theorem states that given a distribution with a

mean μ and variance o^2, the sampling distribution of the mean approaches a normal distribution with a mean (u) and a variance o^2/n as n, the sample size, increases.

The amazing and counter intuitive thing about the central limit theorem is that no matter what the shape of the original distribution is, the sampling distribution of the mean approaches a normal distribution.In PERT, the project network of figure 2 represents the population, while the sample that is drawn from that population is equal to the average critical path (highlighted in red).

Consequently, the project duration follows a normal distribution with the following parameters: Average = E(D)= expected project duration (based on critical path) Variance = E(V) = expected variance (of the critical path)

4. The normal distribution

Using the characteristics of the normal distribution (N(15,7.33)) of the total project duration, basic statistical calculations can be applied to give answers to questions such as:

- What is the probability that the project will be finished before?
- What is the expected project deadline?

7. *Coding*

• Frontend

➢ Index Page (‘./’)

```
<script context="module">
  export async function load({ fetch }) {
    const res = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/search/speciality', {
      headers: {
        'Content-type': 'application/json'
      }
    })
    const data = await res.json()
    if (res.ok) {
      return {
        props: {
```

```
                specialityList: data
            }
        }
    }
    return {
        props: {
            specialityList: null
        }
    }
}
</script>

<script>
    import homepage_doctor from '$lib/assets/homepage/homepage-doctor.png'
    import appointment from '$lib/assets/homepage/appointment.png'
    import livechat from '$lib/assets/homepage/livechat.png'
    import live_chat_icon from '$lib/assets/homepage/live-chat.svg'
    import hour24_icon from '$lib/assets/homepage/24hour.svg'
    import appointment_icon from '$lib/assets/homepage/appointment.svg'
    import onlineConsultation_icon from '$lib/assets/homepage/onlineConsultation.svg'
    import medical_assistance from '$lib/assets/homepage/medical-assistance.png'
    import ellipse1 from '$lib/assets/homepage/Ellipse1.png'
    import ellipse2 from '$lib/assets/homepage/Ellipse2.png'
    import ellipse3 from '$lib/assets/homepage/Ellipse3.png'
    import Footer from '$lib/components/Footer.svelte'
    import Navbar from '$lib/components/Navbar.svelte'
    import SearchResult from '$lib/components/SearchResult.svelte'
    import { navigating } from '$app/stores'
    import Loading from '$lib/components/Loading.svelte'
    import { ENV } from '$lib/utils'
    export let specialityList

    let is_focus = false
    let speciality = ''
    let filteredList = []
    $: if (speciality && specialityList) {
        filteredList = specialityList.filter((data) => {
            const fullNameSpeciality = data.speciality.toLowerCase()
            const reversedSpeciality = fullNameSpeciality.split('').reverse().join('').toLowerCase()
            const trimmedSearchValue = speciality.toLowerCase()
            return (
                fullNameSpeciality.includes(trimmedSearchValue) ||
                reversedSpeciality.includes(trimmedSearchValue)
            )
        })
    } else {
        if (specialityList) {
            filteredList = [...specialityList]
        }
    }
</script>

<svelte:head>
```

```
    <title>FindCare</title>
</svelte:head>

{#if !$navigating}
    <div class="relative">
        <img src={ellipse1} alt="" class="lg:block absolute hidden w-56 top-[85vh] left-0" />

        <!-- Navbar -->

        <Navbar />

        <!-- hero section -->

        <section class="text-gray-600 lg:px-24">
            <div class="container mx-auto flex px-5 py-16 font-sans md:flex-row flex-col items-center">
                <div
                    class="lg:flex-grow md:w-1/2 lg:pr-24 md:pr-16 flex flex-col md:items-start md:text-left mb-16 md:mb-0
items-center text-center"
                >
                    <h1 id="hero" class="title-font lg:text-5xl text-4xl mb-4 font-bold text-black">
                        Let's find your
                        <br class="hidden lg:inline-block" /><span class="text-[#fb3434] lg:text-6xl text-5xl"
                            >Doctor</span
                        >
                    </h1>
                    <p class="mb-8 leading-relaxed">
                        Get appointment with your doctor to get <br />personalized care
                    </p>
                    <div class="flex justify-center relative lg:w-[30vw] w-[90vww]">
                        <input
                            type="text"
                            on:focus={() => (is_focus = true)}
                            bind:value={speciality}
                            on:mouseover={() => (is_focus = false)}
                            class="block border rounded-full py-2 pl-3 pr-10 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary  border-primary"
                            placeholder="Search Doctor or Symptoms..."
                            autocomplete="off"
                            id="search"
                        />
                        <span class="search-glass">
                            <i class="fa-solid fa-magnifying-glass" />
                        </span>
                        {#if is_focus && specialityList}
                            <div
                                class="w-full overflow-auto absolute top-16 flex flex-col space-y-2 bg-gray-100 p-3 rounded-md"
                            >
                                {#if filteredList.length !== 0}
                                    {#each filteredList as filter}
                                        <SearchResult name={filter.speciality} searchType="Specialist" />
                                    {/each}
                                {:else}
                                    <SearchResult name="No results Found" searchType="error" />
```

```
                {/if}
              </div>
            {/if}
          </div>
        </div>
        <div class="lg:max-w-lg lg:w-full md:w-1/2 w-5/6">
          <img class="object-cover object-center rounded" alt="hero" src={homepage_doctor} />
        </div>
      </div>
    </section>

    <!-- Features -->

    <div
      class="px-4 py-16 mx-auto sm:max-w-xl font-sans md:max-w-full lg:max-w-screen-xl md:px-24 lg:px-24 lg:py-20"
    >
      <div class="max-w-xl mb-10 md:mx-auto sm:text-center lg:max-w-2xl md:mb-12">
        <h2
          class="max-w-lg mb-6 font-sans text-3xl font-bold leading-none tracking-tight text-gray-900 sm:text-4xl
md:mx-auto"
        >
          <span class="relative">Our Medical Services</span>
        </h2>
        <p class="text-base text-gray-700 md:text-lg">
          Findcare is an online portal for all your healthcare needs. Our team of medical experts
          are there for you in every step of the way, from finding the right doctor and hospital to
          booking appointments, from providing verified information to any kind of medical
          assistance in between.
        </p>
      </div>
      <div class="upper-feature">
        <div class="grid gap-4 row-gap-5 sm:grid-cols-2 lg:grid-cols-4">
          <div class="flex flex-col justify-center items-center p-5 border rounded shadow-sm">
            <div>
              <div class="flex items-center justify-center w-16 h-16 ">
                <img src={onlineConsultation_icon} alt="" />
                <h6 class="mb-2 font-semibold pl-4 text-center leading-5">Online Consultation</h6>
              </div>
            </div>
          </div>
          <div class="flex flex-col justify-center items-center p-5 border rounded shadow-sm">
            <div>
              <div class="flex items-center justify-center w-16 h-16  ">
                <img src={live_chat_icon} alt="" />
                <h6 class="mb-2 font-semibold pl-4 text-center leading-5">Live Chat</h6>
              </div>
            </div>
          </div>
          <div class="flex flex-col justify-center items-center p-5 border rounded shadow-sm">
            <div>
              <div class="flex items-center justify-center w-16 h-16 ">
                <img src={hour24_icon} alt="" />
                <h6 class="mb-2 font-semibold pl-4 text-center leading-5">24 Hour Service</h6>
```

```
            </div>
          </div>
        </div>
        <div class="flex flex-col justify-center items-center p-5 border rounded shadow-sm">
          <div>
            <div class="flex items-center justify-center w-16 h-16 ">
              <img src={appointment_icon} alt="" />
              <h6 class="mb-2 font-semibold pl-4 text-center leading-5">Appointment</h6>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

  <!-- Online Consultation -->

  <section class="text-gray-600 font-sans relative lg:px-24 bg-[#f1f0ff]">
    <img src={ellipse2} alt="" class="lg:block absolute hidden w-24 top-6 right-0" />
    <div
      class="container mx-auto flex px-5 py-16 md:flex-row flex-col justify-between items-center"
    >
      <div class="lg:max-w-lg lg:w-full md:w-1/2 w-5/6 lg:pr-24 lg:pl-16 md:pr-16">
        <img
          class="object-cover object-center w-3/5 rounded"
          alt="hero"
          src={medical_assistance}
        />
      </div>
      <div class="lg:max-w-lg lg:w-full lg:text-right md:w-1/2 lg:pt-0 pt-5 w-5/6">
        <h1 class="title-font lg:text-5xl text-4xl mb-4 font-bold text-black">
          Online Consultation
        </h1>
        <p class="mb-8 leading-relaxed">
          Consult with top doctors online, with minimal waiting time, easy rescheduling,
          regular email reminders, 24x7 access to records & reports, and easy access to
          prescriptions as well as billing.
        </p>
      </div>
    </div>
  </section>

  <!-- Live Chat -->

  <section class="text-gray-600 font-sans relative lg:px-24">
    <img src={ellipse3} alt="" class="lg:block absolute hidden w-24 top-6 left-0" />
    <div
      class="container mx-auto flex px-5 py-16 md:flex-row flex-col justify-between items-center"
    >
      <div class="lg:max-w-lg lg:w-full lg:text-right md:w-1/2 lg:pt-0 pt-5 w-5/6">
        <h1 class="title-font lg:text-5xl text-4xl mb-4 font-bold text-black">Live Chat</h1>
        <p class="mb-8 leading-relaxed">
          Need help? You can chat with our team directly with live chat. Our messaging assistant
```

```
            can quickly solve many issues or direct you to the right person or place.
          </p>
        </div>
        <div class="lg:max-w-lg lg:w-full md:w-1/2 w-5/6 lg:pl-24 lg:pr-16 md:pl-16">
          <img class="object-cover object-center w-3/4 rounded" alt="hero" src={livechat} />
        </div>
      </div>
    </section>

    <!-- Appointment -->

    <section class="text-gray-600 font-sans relative lg:px-24 bg-[#f1f0ff]">
      <img src={ellipse2} alt="" class="lg:block absolute hidden w-24 top-6 right-0" />
      <div
        class="container mx-auto flex px-5 py-16 md:flex-row flex-col justify-between items-center"
      >
        <div class="lg:max-w-lg lg:w-full md:w-1/2 w-5/6 lg:pr-24 lg:pl-16 md:pr-16">
          <img class="object-cover object-center w-3/4 rounded" alt="hero" src={appointment} />
        </div>
        <div class="lg:max-w-lg lg:w-full lg:text-right md:w-1/2 lg:pt-0 pt-5 w-5/6">
          <h1 class="title-font lg:text-5xl text-4xl mb-4 font-bold text-black">Appointment</h1>
          <p class="mb-8 leading-relaxed">
            We introduce you to a new way of medical screening that goes beyond the conventional way
            of healthcare services. You can now book a clinic visit of the best nearby doctor and
            forget the hassle of long queues and rush.
          </p>
        </div>
      </div>
    </section>

    <!-- CTA -->

    <section class="">
      <div class="flex flex-col justify-center items-center px-5 py-16">
        <div>
          <h1
            class="flex-grow lg:pr-0 lg:text-2xl text-3xl font-medium title-font mt-4 text-gray-900"
          >
            Need a Doctor for checkup?
          </h1>
        </div>
        <div>
          <h1
            class="flex-grow lg:pr-0 lg:text-3xl text-xl  font-medium title-font my-4 text-gray-900"
          >
            Just make an appointment and you are done!
          </h1>
        </div>
        <div>
          <a href="#hero"
            ><button
              class="bg-primary block hover:bg-[#524af4] my-4 text-white w-full lg:px-24 px-14 rounded-md py-3
font-light"
```

```
                    >Get an appointment<svg
                      fill="none"
                      stroke="currentColor"
                      stroke-linecap="round"
                      stroke-linejoin="round"
                      stroke-width="2"
                      class="w-4 h-4 ml-2 inline-block"
                      viewBox="0 0 24 24"
                    >
                      <path d="M5 12h14M12 5l7 7-7 7" />
                    </svg></button
                  ></a
                >
              </div>
            </div>
        </section>

        <!-- Footer -->

        <Footer />
    </div>
{:else}
    <Loading />
{/if}

<style>
    .search-glass {
        position: absolute;
        right: 16px;
        transform: translate(0, -50%);
        top: 60%;
    }

    .scroll-style::-webkit-scrollbar-track {
        -webkit-box-shadow: inset 0 0 6px rgba(0, 0, 0, 0.3);
        border-radius: 10px;
        background-color: #f5f5f5;
    }

    .scroll-style::-webkit-scrollbar {
        width: 8px;
        background-color: #f5f5f5;
    }

    .scroll-style::-webkit-scrollbar-thumb {
        border-radius: 10px;
        -webkit-box-shadow: inset 0 0 6px rgba(0, 0, 0, 0.3);
        background-color: #8755f2;
    }
</style>
```

➢Login Page (‘./login’)

```
<script context="module">
    import { checkUserType } from '$lib/utils.js'
    export async function load({ session }) {
        if (session) {
            // This function will check for route access and redirect user to their appropriate routes accordingly
            return checkUserType(session)
        }
        return {}
    }
</script>

<script>
    import { session, navigating } from '$app/stores'
    import { goto } from '$app/navigation'
    import { post, capitalize } from '$lib/utils.js'
    import { notificationToast } from '$lib/NotificationToast'
    import jwt_decode from 'jwt-decode'
    import shape_png from '$lib/assets/login/shape.png'
    import medical_team_png from '$lib/assets/login/medical-team.png'
    import Loading from '$lib/components/Loading.svelte'
    import Logo from '$lib/components/Logo.svelte'
    let username = ''
    let password = ''
    let show = false
    let isDoctor = false

    let is_loading = false
    const handleInput = (event) => {
        password = event.target.value
    }
    async function handleLogin() {
        is_loading = true
        const response = await post(`api/v1/auth/login`, {
            username,
            password,
            isDoctor,
            isAdmin: false
        })

        is_loading = false
        if (response?.access_token) {
            const cookie = jwt_decode(response.access_token)
            $session = {
                session: response.access_token,
                //@ts-ignore
                status: cookie.status
            }
            //@ts-ignore
            if ($session.status == 'doctor') goto('/doctor')
            //@ts-ignore
```

```
        else if ($session.status == 'admin') goto('/admin/dashboard')
        else goto('/profile')
      } else {
        if (response?.detail[0]?.msg) {
          notificationToast(
            capitalize(response.detail[0].loc?.slice(1).join(', ')).replace(
              /username|Username/gm,
              'Email'
            ) +
              ' ' +
              response?.detail[0]?.msg,
            false,
            2000,
            'error'
          )
        } else {
          notificationToast(response?.detail, false, 2000, 'error')
        }
      }
    }
</script>

<svelte:head>
  <title>Findcare Login | Sign</title>
</svelte:head>

{#if !$navigating}
  <img src={shape_png} alt="shape.png" class="fixed hidden lg:block w-96 bottom-0 left-0" />

  <div
    class="w-screen h-screen flex flex-col justify-center items-center lg:grid lg:grid-cols-2 bg-[#ecf7ff]"
  >
    <div class="flex justify-center items-center lg:ml-48 font-maven">
      <div class="">
        <Logo />
        <form
          on:submit|preventDefault={handleLogin}
          class="flex flex-col justify-center items-start w-[90vw] lg:w-[35rem] bg-white rounded drop-shadow-xl mb-6 px-8 py-7"
        >
          <h2 class="font-bold my-3 mb-9 text-xl">Sign in to your account</h2>
          <div class="relative w-full mb-4">
            <label for="email" class="">Email</label>
            <input
              type="email"
              class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline focus:ring-1 focus:ring-primary"
              placeholder="your@domain.com"
              autocomplete="on"
              bind:value={username}
            />
          </div>
          <div class="w-full mb-3">
```

```
            <div class="flex justify-between">
              <label for="password">Password</label>
              <a href="reset" class="text-primary cursor-pointer hover:font-semibold font-medium"
                >Forgot Password?</a
              >
            </div>
            <div class="relative">
              <input
                type={show ? 'text' : 'password'}
                placeholder="*********"
                class="block border rounded py-2 pt-3 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
                id="password"
                autocomplete="off"
                on:input={handleInput}
              />
              <span>
                <i
                  class="fa {show ? 'fa-eye-slash' : 'fa-eye'} hover:cursor-pointer text-slate-600"
                  aria-hidden="true"
                  id="eye"
                  on:click|preventDefault={() => (show = !show)}
                />
              </span>
            </div>
          </div>
          <div class="flex flex-col justify-center items-center w-full my-3">
            {#if is_loading}
              <button
                disabled
                class="bg-[#7069f5] cursor-not-allowed tracking-wider text-lg w-full text-white mb-3 font-medium py-
2 rounded focus:outline-none focus:shadow-outline"
                ><i class="loading fa fa-spinner fa-spin relative right-2" />Login</button
              >
            {:else}
              <button
                class="bg-primary tracking-wider text-lg hover:bg-[#524af4] w-full text-white mb-3 font-medium py-2
rounded focus:outline-none focus:shadow-outline"
                >Login</button
              >
            {/if}

            <div class="">
              <label
                for="toggle-example"
                class="flex  flex-row  items-center cursor-pointer relative mb-4"
              >
                <input
                  type="checkbox"
                  id="toggle-example"
                  class="sr-only"
                  bind:checked={isDoctor}
                />
```

```
                    <div class="toggle-bg bg-gray-200 border-2 border-gray-200 h-6 w-11 rounded-full" />
                    <p class="ml-3">Are you a Doctor?</p>
                  </label>
                </div>
                <p>
                  Don't Have An Account? <a
                    href="./signup"
                    class="text-primary hover:font-semibold font-medium">Sign Up</a
                  >
                </p>
              </div>
            </form>

            <p class="text-center">
              &copy;2022 <a href="./" class="text-primary font-semibold">FindCare</a>. All rights
              reserved.
            </p>
          </div>
        </div>
        <div class="hidden w-full h-screen lg:flex items-end flex-col">
          <img src={medical_team_png} class="hidden lg:block max-h-full" alt="medical-team.png" />
        </div>
      </div>
    {:else}
      <Loading />
    {/if}

    <style>
      #eye {
        position: absolute;
        right: 14px;
        transform: translate(0, -50%);
        top: 50%;
      }
      .toggle-bg:after {
        content: '';
        @apply absolute top-0.5 left-0.5 bg-white border border-gray-300 rounded-full h-5 w-5 transition shadow-sm;
      }

      input:checked + .toggle-bg:after {
        transform: translateX(100%);
        @apply border-white;
      }

      input:checked + .toggle-bg {
        @apply bg-blue-600 border-blue-600;
      }
    </style>
```

➢ User Signup page (‘./signup’)

```
<script context="module">
```

```
    import { checkUserType } from '$lib/utils.js'

    export async function load({ session }) {
        if (session) {
            // This function will check for route access and redirect user to their appropriate routes accordingly
            return checkUserType(session)
        }
        return {}
    }
</script>

<script>
    import doctor_img from '$lib/assets/signup/doctor.png'
    import { ENV, status_code, titleCase, removeAlpha, removeSpecialCharacters } from '$lib/utils'
    import { notificationToast } from '$lib/NotificationToast'
    import { goto } from '$app/navigation'
    import { navigating } from '$app/stores'
    import Loading from '$lib/components/Loading.svelte'
    let show = false
    let firstName = ''
    let lastName = ''
    let email = ''
    let password = ''
    let confirmPassword = ''
    let phoneNumber = ''
    let gender = ''
    let dob = ''.split('-')

    const handleInput = (event) => {
        password = event.target.value
    }

    let is_loading = false

    const signUpUser = async () => {
        is_loading = true
        if (!phoneNumber.match(/^[6-9]\d{9}$/gm)) {
            notificationToast('Invalid Phone Number !', false, 2000, 'error')
            is_loading = false
            return
        }
        if (password !== confirmPassword) {
            notificationToast('Password do not match !', false, 2000, 'error')
            is_loading = false
            return
        }
        let name = titleCase(
            firstName.replace(/[^A-Za-z ]/g, '').trim() + ' ' + lastName.replace(/[^A-Za-z ]/g, '').trim()
        )
        const resp = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/user/', {
            method: 'post',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({
```

```
            name,
            email: email.trim(),
            phone: phoneNumber,
            gender,
            dob,
            password
          })
        })
        const data = await resp.json()
        is_loading = false
        if (resp.status === status_code.HTTP_201_CREATED) {
          const toastCallbackToLogin = () => goto('/login')
          if (data?.detail)
            notificationToast(data?.detail, false, 3000, 'success', toastCallbackToLogin)
        } else {
          if (data?.detail[0]?.msg) {
            notificationToast(
              data.detail[0].loc?.slice(1).join(', ') + ' ' + data?.detail[0]?.msg,
              false,
              3000,
              'error'
            )
          } else {
            notificationToast(data?.detail, false, 3000, 'error')
          }
        }
      }
    </script>

    <svelte:head>
      <title>Signup</title>
    </svelte:head>

    {#if !$navigating}
      <div class="w-full lg:h-screen h-full flex flex-row font-maven">
        <div class="bg-primary h-screen lg:w-1/3 flex flex-col w-0">
          <h1 class="text-[3.5vw] mt-4 text-center text-white font-poppins font-bold">FindCare</h1>
          <img src={doctor_img} alt="doctor.png" class="w-[28vw] relative left-32 top-[6vh]" />
          <!-- class="h-[60vh] w-[26vw] relative left-[12vw] top-[6vh]" -->
        </div>
        <form
          on:submit|preventDefault={signUpUser}
          class="lg:mt-5 lg:ml-32 lg:p-0 flex flex-col p-5 lg:w-auto w-full drop-shadow-xl"
        >
          <h1 class="text-4xl text-primary font-poppins font-bold mb-4 lg:hidden">FindCare</h1>
          <div class="border border-primary p-4 rounded lg:border-0 mb-8">
            <h1 class="lg:text-4xl text-3xl font-bold">Register</h1>
            <p class="lg:mt-[5vh] mt-[2vh] text-sm lg:text-base">
              Let's get you all set up so you can verify your personal account and begin setting up your
              profile.
            </p>
            <div class="py-6">
              <div class="w-full border-t-[2px] border-[#BABABA]" />
```

```
        </div>

        <div class="w-full lg:flex lg:justify-center">
          <div class="relative w-full mb-4">
            <label for="firstName">First Name</label>
            <input
              type="text"
              name="firstName"
              id="firstName"
              title="Enter Your First Name"
              placeholder="Neeraj"
              class="block border capitalize rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
              required
              on:keypress={removeSpecialCharacters}
              bind:value={firstName}
              autocomplete="off"
            />
          </div>
          <div class="relative w-full mb-4 lg:mx-4">
            <label for="lastName">Last Name</label>
            <input
              type="text"
              name="lastName"
              placeholder="Kumar"
              id="lastName"
              title="Enter Your Last Name"
              class="block border rounded capitalize py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
              required
              on:keypress={removeSpecialCharacters}
              bind:value={lastName}
              autocomplete="off"
            />
          </div>
        </div>
        <div class="w-full lg:flex lg:justify-center">
          <div class="relative w-full mb-4">
            <label for="phoneNumber">Phone Number</label>
            <input
              type="tel"
              placeholder="98765XXXXX"
              name="phoneNumber"
              id="phoneNumber"
              maxlength="10"
              minlength="10"
              on:keypress={removeAlpha}
              title="Enter Your Phone Number"
              class="block appearance-none border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-
outline focus:ring-1 focus:ring-primary"
              required
              bind:value={phoneNumber}
              autocomplete="off"
```

```
            />
          </div>
          <div class="relative w-full mb-4 lg:mx-4">
            <label for="email">Email</label>
            <input
              type="email"
              placeholder="your@domain.com"
              name="email"
              id="email"
              title="Enter Your Email"
              class="block border lowercase rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
              required
              autocomplete="off"
              bind:value={email}
            />
          </div>
        </div>
        <div class="w-full lg:flex lg:justify-center">
          <div class="relative w-full mb-4">
            <label for="dob">Date Of Birth</label>
            <input
              type="date"
              name="dob"
              id="dob"
              min="1950-01-01"
              max={`${new Date().getFullYear()}-01-01`}
              title="Enter Your Date Of Birth"
              class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline focus:ring-1
focus:ring-primary"
              required
              autocomplete="off"
              bind:value={dob}
            />
          </div>
          <div class="relative w-full mb-4 lg:mx-4">
            <label for="gender">Gender</label>
            <div class="flex items-center pt-2 lg:pt-5">
              <div class="flex items-center ">
                <input
                  id="male"
                  type="radio"
                  name="gender"
                  value="male"
                  class="h-4 w-4 mr-2"
                  required
                  bind:group={gender}
                  autocomplete="off"
                />
                <label class="form-check-label inline-block text-gray-800" for="male" value="male"
                  >Male</label
                >
              </div>
```

```
                <div class="flex items-center mx-5">
                  <input
                    id="female"
                    type="radio"
                    value="female"
                    name="gender"
                    class="h-4 w-4 mr-2"
                    required
                    bind:group={gender}
                    autocomplete="off"
                  />
                  <label
                    class="form-check-label inline-block text-gray-800"
                    for="female"
                    value="female">Female</label
                  >
                </div>
                <div class="flex items-center">
                  <input
                    id="other"
                    type="radio"
                    name="gender"
                    value="other"
                    class="h-4 w-4 mr-2"
                    autocomplete="off"
                    required
                    bind:group={gender}
                  />
                  <label class="form-check-label inline-block text-gray-800" for="other" value="other"
                    >Other</label
                  >
                </div>
              </div>
            </div>
          </div>
          <div class="w-full lg:flex lg:justify-center">
            <div class="relative w-full mb-4">
              <label for="password">Password</label>
              <div class="relative">
                <input
                  type={show ? 'text' : 'password'}
                  name="password"
                  placeholder="***********"
                  title="Enter Your Password"
                  id="password"
                  class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline focus:ring-
1 focus:ring-primary"
                  required
                  on:input={handleInput}
                  autocomplete="off"
                />
                <span>
                  <i
```

```
                        class="fa {show
                            ? 'fa-eye-slash'
                            : 'fa-eye'} hover:cursor-pointer text-slate-600 float-right relative bottom-7 right-3"
                        aria-hidden="true"
                        id="eye"
                        on:click|preventDefault={() => (show = !show)}
                      />
                    </span>
                  </div>
                </div>
                <div class="relative w-full mb-4 lg:mx-4">
                  <label for="confirmPassword">Confirm Password</label>
                  <input
                    type="password"
                    name="confirmPassword"
                    placeholder="**********"
                    title="Enter Your Password Again"
                    id="confirmPassword"
                    class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline focus:ring-1
focus:ring-primary"
                    required
                    autocomplete="off"
                    bind:value={confirmPassword}
                  />
                </div>
              </div>
              {#if is_loading}
                <button
                  disabled
                  class="mt-2 bg-[#7069f5] cursor-not-allowed lg:w-64 h-12 text-white rounded w-full lg:p-0 p-2 font-
medium"
                >
                  <i class="loading fa fa-spinner fa-spin relative right-2" />
                  CREATE ACCOUNT
                </button>
              {:else}
                <button
                  class="mt-2 bg-primary hover:bg-[#524af4] lg:w-64 h-12 text-white rounded w-full lg:p-0 p-2 font-medium"
                >
                  CREATE ACCOUNT
                </button>
              {/if}
              <p class="mt-4 text-center lg:text-left">
                Already have an account? <a
                  href="/login"
                  class="text-primary hover:font-semibold font-medium">Log in</a
                >
              </p>
              <p class="mb-5 text-center lg:text-left ">
                Are You A Doctor? <a
                  href="/register-doctor"
                  class="text-primary hover:font-semibold font-medium">Sign Up Here</a
                >
```

```
            </p>
          </div>
        </form>
    </div>
{:else}
    <Loading />
{/if}

<style>
    input::-webkit-outer-spin-button,
    input::-webkit-inner-spin-button {
        -webkit-appearance: none;
        margin: 0;
    }
</style>
```

➢Doctor Signup Page (‘./register-doctor’)

```
<script context="module">
    import { checkUserType } from '$lib/utils.js'
    export async function load({ session }) {
        if (session) {
            // This function will check for route access and redirect user to their appropriate routes accordingly
            return checkUserType(session)
        }
        return {}
    }
</script>

<script>
    import doctor_png from '$lib/assets/doctor/doctor.png'
    import lower_png from '$lib/assets/doctor/lower.png'
    import upper_png from '$lib/assets/doctor/upper.png'
    import { titleCase, status_code, ENV, removeAlpha, removeSpecialCharacters } from '$lib/utils'
    import { notificationToast } from '$lib/NotificationToast'
    import { goto } from '$app/navigation'
    import { navigating } from '$app/stores'
    import Loading from '$lib/components/Loading.svelte'
    import Logo from '$lib/components/Logo.svelte'

    let show = false
    let firstName = ''
    let lastName = ''
    let email = ''
    let password = ''
    let confirmPassword = ''
    let phoneNumber = ''
    let gender = ''
    let dob = ''.split('-')
    let experienceYears = ''
```

```
let speciality = ''
let about = ''
let registrationNumber = ''.toUpperCase()

const handleInput = (event) => {
    password = event.target.value
}

let is_loading = false
async function signUpDoctor() {
    is_loading = true
    // await new Promise((r) => setTimeout(r, 5000))
    if (!phoneNumber.match(/^[6-9]\d{9}$/gm)) {
        notificationToast('Invalid Phone Number !', false, 2000, 'error')
        is_loading = false
        return
    }
    if (password !== confirmPassword) {
        notificationToast('Password do not match !', false, 2000, 'error')
        is_loading = false
        return
    }
    let name = titleCase(
        firstName.replace(/[^A-Za-z .]/g, '').trim() +
            ' ' +
            lastName.replace(/[^A-Za-z ]/g, '').trim()
    )
    const resp = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/', {
        method: 'post',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({
            name,
            email,
            phone: phoneNumber,
            password,
            gender,
            dob,
            experience_year: experienceYears,
            speciality: titleCase(speciality.trim()),
            about: about.trim(),
            registration_number: registrationNumber.toLocaleUpperCase()
        })
    })
    const data = await resp.json()

    is_loading = false
    if (resp.status === status_code.HTTP_201_CREATED) {
        const toastCallbackToLogin = () => goto('/login')
        if (data?.detail)
            notificationToast(data?.detail, false, 3000, 'success', toastCallbackToLogin)
    } else {
        if (data?.detail[0]?.msg) {
            notificationToast(
```

```
                data.detail[0].loc?.slice(1).join(', ') + ' ' + data?.detail[0]?.msg,
                false,
                3000,
                'error'
              )
            } else {
              notificationToast(data?.detail, false, 3000, 'error')
            }
          }
        }
</script>

<svelte:head>
  <title>Findcare Register As Doctor</title>
</svelte:head>

{#if $navigating}
  <Loading />
{:else}
  <div class="w-full h-full">
    <img
      src={doctor_png}
      alt="doctor.png"
      class="fixed hidden lg:block bottom-0 ml-7 left-0 w-40"
    />
    <img src={upper_png} alt="upper.png" class="fixed lg:block right-0 top-0" />
    <img src={lower_png} alt="lower.png" class="fixed hidden lg:block right-0 bottom-0" />
    <img
      src={lower_png}
      alt="lower.png"
      class="fixed block lg:hidden left-0 bottom-0 scale-x-[-1]"
    />
    <div class="flex justify-center items-center">
      <div class="my-2">
        <Logo />
        <div class="flex justify-center items-center font-medium font-maven">
          <form
            on:submit|preventDefault={signUpDoctor}
            class="flex flex-col justify-center items-start w-[90vw] lg:w-[75vw] md:w-[70vw] bg-white lg:bg-
transparent  rounded drop-shadow-xl mb-8 px-2 pb-7 pt-4 lg:border-none border border-primary"
          >
            <div class=" w-full">
              <div class="font-bold mb-3 text-2xl">
                Register as <span class="text-primary font-extrabold">Doctor</span>
              </div>
              <div class="text-sm font-normal">
                Let's get you all set up so you can verify your personal account and begin setting
                up your profile
              </div>
              <div class="w-full border-t-[2px] my-4 border-[#BABABA]" />
            </div>
            <div class="">
              <div class="w-full block border-t-[2px] border-black" />
```

```
            </div>
            <div class="w-full lg:flex lg:justify-center">
              <div class="relative w-full mb-4">
                <label for="firstName">First Name</label>
                <input
                  type="text"
                  name="firstName"
                  id="firstName"
                  title="Enter Your First Name"
                  placeholder="Dr. Neeraj"
                  class="block border capitalize rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-
outline focus:ring-1 focus:ring-primary"
                  required
                  on:keypress={removeSpecialCharacters}
                  bind:value={firstName}
                  autocapitalize="on"
                  autocomplete="off"
                />
              </div>
              <div class="relative w-full mb-4 lg:mx-4">
                <label for="lastName">Last Name</label>
                <input
                  type="text"
                  name="lastName"
                  placeholder="Kumar"
                  id="lastName"
                  title="Enter Your Last Name"
                  class="block border capitalize rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-
outline focus:ring-1 focus:ring-primary"
                  required
                  on:keypress={removeSpecialCharacters}
                  bind:value={lastName}
                  autocomplete="off"
                />
              </div>
              <div class="relative w-full mb-4">
                <label for="registrationNumber">Registration Number</label>
                <input
                  type="text"
                  placeholder="AQ-15-XXXXX"
                  name="registrationNumber"
                  title="Enter Your Registration Number"
                  id="registrationNumber"
                  class="block uppercase appearance-none border rounded py-2 px-3 w-full mt-3 focus:outline-none
focus:shadow-outline focus:ring-1 focus:ring-primary"
                  required
                  bind:value={registrationNumber}
                  autocomplete="off"
                />
              </div>
            </div>
            <div class="w-full lg:flex lg:justify-center">
              <div class="relative w-full mb-4">
```

```
                    <label for="phoneNumber">Phone Number</label>
                    <input
                      type="tel"
                      placeholder="98765XXXXX"
                      name="phoneNumber"
                      id="phoneNumber"
                      maxlength="10"
                      minlength="10"
                      on:keypress={removeAlpha}
                      title="Enter Your Phone Number"
                      class="block appearance-none border rounded py-2 px-3 w-full mt-3 focus:outline-none
focus:shadow-outline focus:ring-1 focus:ring-primary"
                      required
                      bind:value={phoneNumber}
                      autocomplete="off"
                    />
                  </div>
                  <div class="relative w-full mb-4 lg:mx-4">
                    <label for="email">Email</label>
                    <input
                      type="email"
                      placeholder="your@domain.com"
                      name="email"
                      id="email"
                      title="Enter Your Email"
                      class="block border lowercase rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-
outline focus:ring-1 focus:ring-primary"
                      required
                      autocomplete="off"
                      bind:value={email}
                    />
                  </div>
                  <div class="relative w-full mb-4">
                    <label for="speciality">Speciality</label>
                    <input
                      type="text"
                      name="speciality"
                      id="speciality"
                      title="Enter Your Speciality"
                      class="block border capitalize rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-
outline focus:ring-1 focus:ring-primary"
                      required
                      autocomplete="off"
                      bind:value={speciality}
                    />
                  </div>
                </div>
                <div class="w-full lg:flex lg:justify-center">
                  <div class="relative w-full mb-4">
                    <label for="dob">Date Of Birth</label>
                    <input
                      type="date"
                      name="dob"
```

```
                        id="dob"
                        title="Enter Your Date Of Birth"
                        class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
                        required
                        bind:value={dob}
                      />
                    </div>
                    <div class="relative w-full mb-4 lg:mx-4">
                      <label for="gender">Gender</label>
                      <div class="flex items-center pt-2 lg:pt-5">
                        <div class="flex items-center ">
                          <input
                            id="male"
                            type="radio"
                            name="gender"
                            class="mr-2"
                            value="male"
                            bind:group={gender}
                            required
                            autocomplete="off"
                          />
                          <label
                            class="form-check-label inline-block text-gray-800"
                            for="male"
                            value="male">Male</label
                          >
                        </div>
                        <div class="flex items-center mx-5">
                          <input
                            id="female"
                            type="radio"
                            name="gender"
                            class="mr-2"
                            value="female"
                            bind:group={gender}
                            required
                            autocomplete="off"
                          />
                          <label
                            class="form-check-label inline-block text-gray-800"
                            for="female"
                            value="female">Female</label
                          >
                        </div>
                        <div class="flex items-center">
                          <input
                            id="other"
                            type="radio"
                            name="gender"
                            class="mr-2"
                            value="other"
                            bind:group={gender}
```

```
                        required
                        autocomplete="off"
                      />
                      <label
                        class="form-check-label inline-block text-gray-800"
                        for="other"
                        value="other">Other</label
                      >
                    </div>
                  </div>
                </div>
                <div class="relative w-full mb-4">
                  <label for="expYear">Experience year</label>
                  <input
                    type="number"
                    name="expYear"
                    id="expYear"
                    title="Enter Your Experience Year"
                    min="1"
                    max="70"
                    placeholder="20"
                    class="block appearance-none border rounded py-2 px-3 w-full mt-3 focus:outline-none
focus:shadow-outline focus:ring-1 focus:ring-primary"
                    required
                    autocomplete="off"
                    bind:value={experienceYears}
                  />
                </div>
              </div>
              <div class="w-full lg:flex lg:justify-center">
                <div class="relative w-full mb-4">
                  <label for="password">Password</label>
                  <div class="relative">
                    <input
                      type={show ? 'text' : 'password'}
                      name="password"
                      placeholder="**********"
                      title="Enter Your Password"
                      id="password"
                      class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
                      required
                      autocomplete="off"
                      on:input={handleInput}
                    />
                    <span>
                      <i
                        class="fa {show
                          ? 'fa-eye-slash'
                          : 'fa-eye'} hover:cursor-pointer text-slate-600"
                        aria-hidden="true"
                        id="eye"
                        on:click|preventDefault={() => (show = !show)}
```

```
                        />
                      </span>
                    </div>
                  </div>
                  <div class="relative w-full mb-4 lg:mx-4">
                    <label for="confirmPassword">Confirm Password</label>
                    <input
                      type="password"
                      name="confirmPassword"
                      placeholder="**********"
                      title="Enter Your Password Again"
                      id="confirmPassword"
                      class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
                      required
                      autocomplete="off"
                      bind:value={confirmPassword}
                    />
                  </div>
                  <div class="relative w-full mb-4">
                    <label for="about">About</label>
                    <input
                      type="text"
                      name="about"
                      placeholder="Tell us about yourself"
                      title="Enter About Yourself"
                      id="about"
                      class="block border text rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
                      required
                      autocomplete="off"
                      bind:value={about}
                    />
                  </div>
                </div>
                <div class="w-full">
                  <div class="w-full">
                    {#if is_loading}
                      <button
                        disabled
                        class="bg-[#7069f5] cursor-not-allowed lg:w-[16vw] text-white my-3 py-2 w-full rounded
focus:outline-none focus:shadow-outline font-medium"
                        ><i class="loading fa fa-spinner fa-spin relative right-2" />
                        CREATE ACCOUNT</button
                      >
                    {:else}
                      <button
                        class="bg-primary hover:bg-[#524af4] lg:w-[16vw] text-white my-3 py-2 w-full rounded
focus:outline-none focus:shadow-outline font-medium"
                        >CREATE ACCOUNT</button
                      >
                    {/if}
                  </div>
```

```
                    <div>
                      Already have an account? <a href="./login" class="text-primary hover:font-semibold"
                        >Login</a
                      >
                    </div>
                  </div>
                </form>
              </div>
            </div>
          </div>
        </div>
{/if}

<style>
  input::-webkit-outer-spin-button,
  input::-webkit-inner-spin-button {
    -webkit-appearance: none;
    margin: 0;
  }
  img {
    -webkit-user-drag: none;
    -khtml-user-drag: none;
    -moz-user-drag: none;
    -o-user-drag: none;
    /* user-drag: none; */
  }
  #eye {
    position: absolute;
    right: 14px;
    transform: translate(0, -50%);
    top: 50%;
  }
</style>
```

➢ Appointment Page (‘./profile/appointment’)

```
<script context="module">
  export async function load({ session, fetch }) {
    if (!session) {
      return {
        status: 302,
        redirect: '/login'
      }
    }
    if (session?.status === 'user') {
      const res = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/user/appointment/', {
        method: 'GET',
        headers: {
          'Content-type': 'application/json',
          Authorization: `Bearer ${session.session}`
        }
      })
```

```
        if (
            res.status === status_code.HTTP_422_UNPROCESSABLE_ENTITY ||
            res.status === status_code.HTTP_403_FORBIDDEN ||
            res.status === status_code.HTTP_401_UNAUTHORIZED
        ) {
            await fetch('api/v1/auth/logout')
            return {
                status: 302,
                redirect: '/login'
            }
        }
        const appointments = await res.json()
        return {
            props: {
                status: res.status,
                session,
                appointments
            }
        }
    } else {
        if (session?.status === 'admin') {
            return {
                status: 302,
                redirect: '/admin/dashboard'
            }
        } else {
            return {
                status: 302,
                redirect: '/doctor'
            }
        }
    }
}
</script>

<script>
    import { ENV, status_code } from '$lib/utils'
    import Footer from '$lib/components/Footer.svelte'
    import Navbar from '$lib/components/Navbar.svelte'
    import AppointmentTable from '$lib/components/admin/AppointmentTable.svelte'
    import { navigating } from '$app/stores'
    import Loading from '$lib/components/Loading.svelte'
    export let session, appointments, status
</script>

<Navbar />
{#if !$navigating}
    <div class="md:m-6 md:p-6 m-1 p-1 font-maven">
        <!-- <AppointmentTable {appointments} {session} /> -->
        {#if status === status_code.HTTP_404_NOT_FOUND}
            <h2>No appointment Available</h2>
        {:else}
            <AppointmentTable {appointments} {session} />
```

```
    {/if}
  </div>

  <Footer />
{:else}
  <Loading />
{/if}
```

➢ Doctor's Dashboard ('./doctor')

```
<script context="module">
  export async function load({ session, fetch }) {
    if (!session) {
      return {
        status: 302,
        redirect: '/login'
      }
    }
    if (session?.status === 'doctor') {
      // FETCH PATIENT DETAILS HERE
      const response = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/clinic/', {
        method: 'GET',
        headers: {
          'Content-type': 'application/json',
          Authorization: `Bearer ${session.session}`
        }
      })
      if (
        response.status === status_code.HTTP_422_UNPROCESSABLE_ENTITY ||
        response.status === status_code.HTTP_403_FORBIDDEN ||
        response.status === status_code.HTTP_401_UNAUTHORIZED
      ) {
        await fetch('api/v1/auth/logout')
        return {
          status: 302,
          redirect: '/login'
        }
      }
      let data = await response.json()
      let doctor_profile = null
      if (response.status == status_code.HTTP_404_NOT_FOUND) {
        data = null
        let response = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/', {
          method: 'GET',
          headers: {
            'Content-type': 'application/json',
            Authorization: `Bearer ${session.session}`
          }
        })
        if (
          response.status === status_code.HTTP_422_UNPROCESSABLE_ENTITY ||
```

```
                response.status === status_code.HTTP_403_FORBIDDEN ||
                response.status === status_code.HTTP_401_UNAUTHORIZED
            ) {
                await fetch('api/v1/auth/logout')
                return {
                    status: 302,
                    redirect: '/login'
                }
            }
            doctor_profile = await response.json()
        }
        return {
            props: {
                response: data,
                doctor_profile
            }
        }
    } else {
        if (session?.status === 'admin') {
            return {
                status: 302,
                redirect: '/admin/dashboard'
            }
        } else {
            return {
                status: 302,
                redirect: '/profile'
            }
        }
    }
}
</script>

<script>
    import Footer from '$lib/components/dashboard-footer.svelte'
    import Sidebar from '$lib/components/sidebar.svelte'
    import Navbar from '$lib/components/Navbar.svelte'
    import Header from '$lib/components/dashboard-stats.svelte'
    import DashboardFooter from '$lib/components/dashboard-footer.svelte'
    import Changepass from '$lib/components/Changepass.svelte'
    import { ENV, status_code } from '$lib/utils'
    import DashboardTable from '$lib/components/DashboardTable.svelte'
    import DoctorProfile from '$lib/components/Doctor-profile.svelte'
    import ClinicDetails from '$lib/components/ClinicDetails.svelte'
    import { navigating, session } from '$app/stores'
    import { goto } from '$app/navigation'
    import { user as userProfileStore, doctorDashBoardHeader } from '../../stores'
    import Loading from '$lib/components/Loading.svelte'

    function toggleCollapseShow(classes) {
        collapseShow = classes
    }
    let collapseShow = 'hidden'
```

```
    let show = false
    let selected = 'dashboard'
    export let response
    export let doctor_profile
    if (response)
        $userProfileStore = {
            profile_image: response?.doctor?.profile_image
        }
    else
        $userProfileStore = {
            profile_image: doctor_profile?.profile_image
        }
    let is_dashboard_loading = false
    async function doctorDashboardHandler() {
        is_dashboard_loading = true
        const resp = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/clinic/', {
            method: 'GET',
            headers: {
                'Content-type': 'application/json',
                //@ts-ignore
                Authorization: `Bearer ${$session.session}`
            }
        })
        if (resp.ok) {
            response = await resp.json()
        }
        is_dashboard_loading = false
        selected = 'dashboard'
    }
    $: if ($doctorDashBoardHeader) response = $doctorDashBoardHeader
</script>

<!-- Navbar -->
<svelte:head>
    <title>{doctor_profile ? doctor_profile?.name : response?.doctor?.name}</title>
</svelte:head>
{#if !$navigating}
    <div class="h-screen w-screen overflow-x-hidden font-maven">
        <!-- Sidebar -->

        <nav
            class="md:left-0 md:block md:fixed md:top-0 md:bottom-0 md:overflow-y-auto md:flex-row md:flex-nowrap
md:overflow-hidden shadow-xl bg-white flex flex-wrap items-center justify-between relative md:w-64 z-10 py-4 px-6"
        >
            <div
                class="md:flex-col md:items-stretch md:min-h-full md:flex-nowrap px-0 flex flex-wrap items-center justify-
between w-full mx-auto"
            >
                <!-- Toggler -->
                <button
                    class="cursor-pointer text-black opacity-50 md:hidden px-3 py-1 text-xl leading-none bg-transparent rounded
border border-solid border-transparent"
                    type="button"
```

```
            on:click={() => toggleCollapseShow('bg-white m-2 py-3 px-6')}
          >
            <i class="fas fa-bars" />
          </button>
          <!-- Brand -->
          <a
            class="md:block text-left md:pb-2 text-blueGray-600 mr-0 inline-block whitespace-nowrap text-sm font-bold p-4 px-0"
            href="/"
          >
            <div class="p-2 text-primary text-3xl font-bold tracking-wide font-poppins">
              <a href="/">Find<span class="text-[#fb3434]">Care</span></a>
            </div>
          </a>
          <!-- User -->
          <ul class="md:hidden items-center flex flex-wrap list-none">
            <li class="inline-block relative" />
            <li class="inline-block relative" />
          </ul>
          <!-- Collapse -->
          <div
            class="md:flex md:flex-col md:items-stretch md:opacity-100 md:relative md:mt-4 md:shadow-none shadow absolute top-0 left-0 right-0 z-40 overflow-y-auto overflow-x-hidden h-auto items-center flex-1 rounded {collapseShow}"
          >
            <!-- Collapse header -->
            <div
              class="md:min-w-full md:hidden block pb-4 mb-4 border-b border-solid border-blueGray-200"
            >
              <div class="flex flex-wrap">
                <div class="w-6/12">
                  <a
                    class="md:block text-left md:pb-2 text-blueGray-600 mr-0 inline-block whitespace-nowrap text-sm uppercase font-bold p-2 px-0"
                    href="/"
                  >
                    <div class=" text-primary text-3xl font-bold tracking-wide font-poppins">
                      <a href="/">Find<span class="text-[#fb3434]">Care</span></a>
                    </div>
                  </a>
                </div>
                <div class="w-6/12 flex justify-end">
                  <button
                    type="button"
                    class="cursor-pointer text-black opacity-50 md:hidden px-3 py-1 text-xl leading-none bg-transparent rounded border border-solid border-transparent"
                    on:click={() => toggleCollapseShow('hidden')}
                  >
                    <i class="fas fa-times" />
                  </button>
                </div>
              </div>
            </div>
            <!-- Divider -->
```

```
<hr class="mb-4 md:min-w-full" />
<!-- Heading -->
<h6
  class="md:min-w-full text-blueGray-500 text-xs uppercase font-bold block pt-1 pb-4 no-underline"
>
  Doctor's Admin Panel
</h6>
<!-- Navigation -->

<ul class="md:flex-col md:min-w-full flex flex-col list-none">
  <li class="items-center">
    <button
      class="text-xs uppercase py-3 font-bold block  "
      on:click={doctorDashboardHandler}
    >
      <i
        class="{is_dashboard_loading
          ? 'loading fa fa-spinner fa-spin ml-2 mr-2'
          : 'fas fa-tv ml-2 mr-2'} text-sm text-primary "
      />
      Dashboard
    </button>
    {#if selected == 'dashboard'}
      <hr class="border-[1px] border-primary bg-primary" />
    {/if}
  </li>

  <li class="items-center">
    <button
      class="text-blueGray-700 hover:text-blueGray-500 text-xs uppercase py-3 font-bold block"
      on:click={() => (selected = 'Account Setting')}
    >
      <i class="fas fa-user-circle text-primary ml-2 mr-2 text-sm" />
      Account Setting
    </button>
    {#if selected == 'Account Setting'}
      <hr class="border-[1px] border-primary bg-primary" />
    {/if}
  </li>
  <li class="items-center">
    <button
      class="text-xs uppercase py-3 font-bold block"
      on:click={() => (selected = 'clinic')}
    >
      <i class="fas fa-hospital ml-2 mr-2 text-primary text-sm " />
      Clinic
    </button>
    {#if selected == 'clinic'}
      <hr class="border-[1px] border-primary bg-primary" />
    {/if}
  </li>
  <li class="items-center">
    <button
```

```
                    class="text-xs uppercase py-3  font-bold block"
                    on:click={() => (selected = 'changepass')}
                  >
                    <i class="fas fa-key ml-2 mr-2 text-sm text-primary" />
                    Change Password
                  </button>
                  {#if selected == 'changepass'}
                    <hr class="border-[1px] border-primary bg-primary" />
                  {/if}
                </li>
                <li class="items-center">
                  <button
                    class="text-xs uppercase py-3 font-bold block"
                    on:click={() => {
                      $session = null
                      $userProfileStore = null
                      goto('/logout')
                    }}
                  >
                    <i class="fas fa-arrow-right-from-bracket ml-2 mr-2 text-sm text-primary" />
                    Logout
                  </button>
                </li>
              </ul>

              <!-- Divider -->
              <hr class="my-4 md:min-w-full" />
            </div>
          </div>
        </nav>

        <!-- Body -->
        <div class="relative md:ml-64 bg-blueGray-100">
          <Header {response} />

          <div class="px-4 md:px-10 mx-auto w-full m-24 mt-3">
            {#if selected == 'dashboard'}
              <DashboardTable {response} />
            {/if}
            {#if selected == 'Account Setting'}
              <DoctorProfile {response} {doctor_profile} />
            {/if}
            {#if selected == 'changepass'}
              <Changepass />
            {/if}
            {#if selected == 'clinic'}
              <ClinicDetails {response} />
            {/if}

            <Footer />
          </div>
        </div>
      </div>
```

```
  <!-- Footer -->
{:else}
  <Loading />
{/if}
```

➢ Admin Login Page (‘./admin’)

```
<script context="module">
  import { checkUserType } from '$lib/utils.js'
  export async function load({ session }) {
    if (session) {
      // This function will check for route access and redirect user to their appropriate routes accordingly
      return checkUserType(session)
    }
    return {}
  }
</script>

<script>
  import { notificationToast } from '$lib/NotificationToast'
  import { session, navigating } from '$app/stores'
  import { goto } from '$app/navigation'
  import { post, capitalize } from '$lib/utils.js'
  import jwt_decode from 'jwt-decode'
  import Loading from '$lib/components/Loading.svelte'
  let username = ''
  let password = ''

  let is_loading = false
  async function handleLogin() {
    is_loading = true
    const response = await post(`api/v1/auth/login`, {
      username,
      password,
      isDoctor: false,
      isAdmin: true
    })

    is_loading = false
    if (response?.access_token) {
      const cookie = jwt_decode(response.access_token)
      $session = {
        session: response.access_token,
        //@ts-ignore
        status: cookie.status
      }
      //@ts-ignore
      if ($session.status == 'doctor') goto('/doctor')
      //@ts-ignore
      else if ($session.status == 'admin') goto('/admin/dashboard')
```

```
        else goto('/profile')
      } else {
        if (response?.detail[0]?.msg) {
          notificationToast(
            capitalize(response.detail[0].loc?.slice(1).join(', ')).replace(
              /username|Username/gm,
              'Email'
            ) +
              ' ' +
              response?.detail[0]?.msg,
            false,
            2000,
            'error'
          )
        } else {
          notificationToast(response?.detail, false, 2000, 'error')
        }
      }
    }
</script>

<svelte:head>
  <title>Findcare Admin Login</title>
</svelte:head>

{#if !$navigating}
  <div class="w-screen h-screen flex flex-col justify-center items-center font-maven">
    <div
      class="flex flex-col justify-center items-center w-[90vw] lg:w-[30rem] bg-white rounded drop-shadow-xl mb-8 px-8 py-7"
    >
      <div id="header" class="flex flex-col justify-center items-center">
        <div>
          <img
            src="https://img.icons8.com/external-icongeek26-flat-icongeek26/344/external-admin-project-work-icongeek26-flat-icongeek26.png"
            alt="lock svg"
            class="w-32"
          />
          <!-- <Locked fill="#635bff" class="w-32 h-32" /> -->
        </div>
        <div>
          <h2 class="text-2xl font-semibold mt-6">Admin Panel</h2>
        </div>
      </div>
      <hr class="h-2" />
      <div id="body" class="flex flex-col w-full justify-center items-center">
        <form class="w-full mt-3 mb-2" on:submit|preventDefault={handleLogin}>
          <label for="email" class="font-bold">Email</label>
          <input
            bind:value={username}
            type="email"
            id="email"
```

```
                placeholder="your@domain.com"
                required
                class="block border rounded py-2 px-3 w-full mt-3 mb-4 focus:outline-none focus:shadow-outline
focus:ring-1 focus:ring-primary"
              />
              <label for="password" class="font-bold">Password</label>
              <input
                bind:value={password}
                type="password"
                id="password"
                placeholder="************"
                required
                class="block border rounded py-2 px-3 w-full mt-3 focus:outline-none focus:shadow-outline focus:ring-1
focus:ring-primary"
              />
              {#if is_loading}
                <button
                  disabled
                  class="bg-[#7069f5] cursor-not-allowed text-white mb-3 font-medium py-2 mt-5 w-full rounded
focus:outline-none focus:shadow-outline"
                  ><i class="loading fa fa-spinner fa-spin relative right-2" />Login as Admin</button
                >
              {:else}
                <button
                  class="bg-primary hover:bg-[#524af4] text-white mb-3 font-medium py-2 mt-5 w-full rounded
focus:outline-none focus:shadow-outline"
                  >Login as Admin</button
                >
              {/if}
            </form>
            <div>
              <p>
                Return to <a href="/login" class="text-primary hover:font-semibold font-medium"
                  >Login as User</a
                >
              </p>
            </div>
          </div>
        </div>
        <p class="text-center">
          &copy;2022 <a href="./" class="text-primary font-semibold">FindCare</a>. All rights reserved.
        </p>
      </div>
    {:else}
      <Loading />
    {/if}
```

➢ Components Used

Web Components is a suite of different technologies allowing us to create reusable custom elements — with their functionality encapsulated away from the rest of your code — and utilize them in your web apps.

We have created multiple components of codes which are being used on the websites multiple times to improve our productivity.

- **Navbar Component**

```
<script>
  import { goto } from '$app/navigation'
  import { session } from '$app/stores'
  import { user as userProfileStore } from '../../stores'
  import Login from 'carbon-icons-svelte/lib/Login.svelte'
  import Logo from '$lib/components/Logo.svelte'
  import jwt_decode from 'jwt-decode'
  let user = $session ? true : false
  let userType = null
  if ($session) {
    //@ts-ignore
    const cookie = jwt_decode($session.session)
    //@ts-ignore
    userType = cookie.status
  }

  let menu = true
  let profileMenu = true
  $: profile = $userProfileStore?.profile_image
    ? $userProfileStore?.profile_image
    : 'https://cdn-icons-png.flaticon.com/512/3135/3135715.png'
</script>

<nav class="bg-white border-black px-2 sm:px-4 py-2.5 rounded">
  <div class="container flex flex-wrap justify-between items-center mx-auto">
    <Logo />
    {#if !user}
      <div class="flex-col lg:flex hidden lg:flex-row mx-6 md:order-2 justify-center items-center">
        <a href="/login" class="font-semibold mr-7 hover:text-[#524af4]">Login</a>
        <a href="/signup"
          ><button
            class="bg-primary hover:bg-[#524af4] text-white rounded-full w-full px-7 py-1 font-medium"
            >Sign up</button
          ></a
        >
      </div>
      <div class="flex relative items-center md:order-2 lg:hidden">
        <a href="/login"><Login class="text-lg hover:color-[#524af4] w-6 h-6" /></a>
      </div>
    {/if}
```

```
{#if user}
  <div class="flex relative items-center md:order-2">
    <button
      type="button"
      class="flex  mr-3 text-sm rounded-full md:mr-0 focus:ring-2  focus:ring-sky-200"
      id="user-menu-button"
      aria-expanded="false"
      data-dropdown-toggle="dropdown"
      on:click|preventDefault={() => (profileMenu = !profileMenu)}
    >
      <span class="sr-only">Open user menu</span>
      <img class="w-10 h-10 rounded-full" src={profile} alt="" />
    </button>

    <div
      class="{profileMenu
        ? 'hidden'
        : ''} absolute right-0 top-8 z-50 my-4 w-40 text-base list-none text-black bg-white border border-gray-100
rounded divide-y  "
      id="dropdown"
      data-popper-placement="bottom"
    >
      <ul
        class="absolute right-0 w-40 p-2 mt-2 space-y-2 text-white bg-white border border-gray-100 rounded-md
shadow-md  dark:border-indigo-700 dark:text-white dark:bg-indigo-700"
        aria-label="submenu"
      >
        {#if userType === 'user'}
          <li class="flex">
            <a
              class="inline-flex items-center w-full px-2 py-1 text-sm font-semibold transition-colors duration-150
rounded-md hover:bg-gray-100 hover:text-gray-800 dark:hover:bg-gray-800 dark:hover:text-gray-200"
              href="/profile"
            >
              <svg
                class="w-4 h-4 mr-3"
                aria-hidden="true"
                fill="none"
                stroke-linecap="round"
                stroke-linejoin="round"
                stroke-width="2"
                viewBox="0 0 24 24"
                stroke="#FFFFFF"
              >
                <path d="M16 7a4 4 0 11-8 0 4 4 0 018 0zM12 14a7 7 0 00-7 7h14a7 7 0 00-7-7z" />
              </svg>

              <span>Profile</span>
            </a>
          </li>
        {/if}

        <li class="flex">
```

```
                <a
                  class="inline-flex items-center w-full px-2 py-1 text-sm font-semibold transition-colors duration-150
rounded-md hover:bg-gray-100 hover:text-gray-800 dark:hover:bg-gray-800 dark:hover:text-gray-200"
                  href="/"
                >
                  <svg
                    stroke-linecap="round"
                    stroke-linejoin="round"
                    stroke-width="2"
                    stroke="currentColor"
                    width="16"
                    height="16"
                    x="0"
                    y="0"
                    viewBox="0 0 512 512"
                    style="enable-background:new 0 0 512 512"
                    xml:space="preserve"
                    class="w-4 h-4 mr-3"
                    ><g>
                      <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                          <path
                            d="M462,44h-47.667V30c0-16.542-13.458-30-30-30s-30,13.458-30,30v14h-168V30c0-
16.542-13.458-30-30-30s-30,13.458-30,30    v14.37c-0.85-0.235-1.741-0.37-2.666-0.37H83.823c-27.57,0-50,22.43-
50,50v170.333c0,5.523,4.478,10,10,10s10-4.477,10-10V147    h360.51c5.522,0,10-4.477,10-10s-4.478-10-10-
10H53.823V94c0-16.542,13.458-30,30-30h39.844c0.925,0,1.816-0.136,2.666-0.37V76.5    c0,16.542,13.458,30,30,30s30-
13.458,30-30V64h168v12.5c0,16.542,13.458,30,30,30s30-13.458,30-
30V64H462    c16.542,0,30,13.458,30,30v379.994c0,9.928-8.077,18.006-18.006,18.006s-18.006-8.078-18.006-18.006v-
25.869    c0-5.523-4.478-10-10-10H329.667c-5.522,0-10,4.477-
10,10s4.478,10,10,10h106.321v15.869c0,6.511,1.648,12.643,4.545,18.006H38    c-9.925,0-18-8.075-18-18v-
15.875h223.825c5.522,0,10-4.477,10-10s-4.478-10-10-10h-99.916    c17.474-15.049,28.57-37.309,28.57-62.125c0-45.215-
36.785-82-82-82c-45.215,0-82,36.785-82,82    c0,24.816,11.096,47.076,28.57,62.125H10c-5.522,0-10,4.477-
10,10V474c0,20.953,17.047,38,38,38h435.994    C494.95,512,512,494.951,512,473.994V94C512,66.43,489.57,44,462,44z
M166.333,76.5c0,5.514-4.486,10-10,10s-10-4.486-10-10V30    c0-5.514,4.486-10,10-10s10,4.486,10,10V76.5z
M394.333,76.5c0,5.514-4.486,10-10,10c-5.514,0-10-4.486-10-10V30    c0-5.514,4.486-10,10-
10c5.514,0,10,4.486,10,10V76.5z M28.479,376c0-34.187,27.813-62,62-62s62,27.813,62,62s-27.813,62-
62,62    S28.479,410.187,28.479,376z"
                            fill="#FFFFFF"
                            data-original="#000000"
                            class=""
                          />
                        </g>
                      </g>
                      <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                          <path
                            d="M468.309,129.93c-1.859-1.86-4.439-2.93-7.069-2.93c-2.631,0-5.21,1.07-7.07,2.93c-
1.86,1.86-2.93,4.44-2.93,7.07    s1.069,5.21,2.93,7.07c1.861,1.86,4.439,2.93,7.07,2.93c2.63,0,5.21-1.07,7.069-2.93c1.86-
1.86,2.931-4.44,2.931-7.07    S470.17,131.79,468.309,129.93z"
                            fill="#FFFFFF"
                            data-original="#000000"
                            class=""
                          />
```

```
                    </g>
                  </g>
                  <g xmlns="http://www.w3.org/2000/svg">
                    <g>
                      <path
                        d="M298.649,441.05c-1.859-1.86-4.439-2.92-7.069-2.92s-5.21,1.06-7.07,2.92c-1.86,1.87-2.93,4.44-2.93,7.07 c0,2.64,1.069,5.21,2.93,7.08c1.86,1.86,4.44,2.92,7.07,2.92s5.21-1.06,7.069-2.92c1.86-1.87,2.931-4.45,2.931-7.08 C301.58,445.49,300.51,442.92,298.649,441.05z"
                        fill="#FFFFFF"
                        data-original="#000000"
                        class=""
                      />
                    </g>
                  </g>
                  <g xmlns="http://www.w3.org/2000/svg">
                    <g>
                      <path
                        d="M226.245,304c-20.953,0-38,17.047-38,38s17.047,38,38,38s38-17.047,38-38S247.198,304,226.245,304z M226.245,360 c-9.925,0-18-8.075-18-18s8.075-18,18-18s18,8.075,18,18S236.17,360,226.245,360z"
                        fill="#FFFFFF"
                        data-original="#000000"
                        class=""
                      />
                    </g>
                  </g>
                  <g xmlns="http://www.w3.org/2000/svg">
                    <g>
                      <path
                        d="M319.578,304c-20.953,0-38,17.047-38,38s17.047,38,38,38s38-17.047,38-38S340.531,304,319.578,304z M319.578,360 c-9.925,0-18-8.075-18-18s8.075-18,18-18s18,8.075,18,18S329.503,360,319.578,360z"
                        fill="#FFFFFF"
                        data-original="#000000"
                        class=""
                      />
                    </g>
                  </g>
                  <g xmlns="http://www.w3.org/2000/svg">
                    <g>
                      <path
                        d="M412.912,304c-20.953,0-38,17.047-38,38s17.047,38,38,38c20.953,0,38-17.047,38-38S433.865,304,412.912,304z M412.912,360 c-9.925,0-18-8.075-18-18s8.075-18,18-18s18,8.075,18,18S422.837,360,412.912,360z"
                        fill="#FFFFFF"
                        data-original="#000000"
                        class=""
                      />
                    </g>
                  </g>
                  <g xmlns="http://www.w3.org/2000/svg">
                    <g>
                      <path
```

```
                                d="M132.912,200c-20.953,0-38,17.047-38,38s17.047,38,38,38s38-17.047,38-
38S153.865,200,132.912,200z M132.912,256    c-9.925,0-18-8.075-18-18s8.075-18,18-
18c9.925,0,18,8.075,18,18S142.837,256,132.912,256z"
                                fill="#FFFFFF"
                                data-original="#000000"
                                class=""
                            />
                        </g>
                    </g>
                    <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                            <path
                                d="M319.578,200c-20.953,0-38,17.047-38,38s17.047,38,38,38s38-17.047,38-
38S340.531,200,319.578,200z M319.578,256    c-9.925,0-18-8.075-18-18s8.075-18,18-
18s18,8.075,18,18S329.503,256,319.578,256z"
                                fill="#FFFFFF"
                                data-original="#000000"
                                class=""
                            />
                        </g>
                    </g>
                    <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                            <path
                                d="M412.912,200c-20.953,0-38,17.047-38,38s17.047,38,38,38c20.953,0,38-17.047,38-
38S433.865,200,412.912,200z M412.912,256    c-9.925,0-18-8.075-18-18s8.075-18,18-
18s18,8.075,18,18S422.837,256,412.912,256z"
                                fill="#FFFFFF"
                                data-original="#000000"
                                class=""
                            />
                        </g>
                    </g>
                    <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                            <path
                                d="M226.245,200c-20.953,0-38,17.047-38,38s17.047,38,38,38s38-17.047,38-
38S247.198,200,226.245,200z M226.245,256    c-9.925,0-18-8.075-18-18s8.075-18,18-
18s18,8.075,18,18S236.17,256,226.245,256z"
                                fill="#FFFFFF"
                                data-original="#000000"
                                class=""
                            />
                        </g>
                    </g>
                    <g xmlns="http://www.w3.org/2000/svg">
                        <g>
                            <path
                                d="M126.104,351.629c-3.906-3.905-10.236-3.905-14.143,0l-32.566,32.567l-11.129-
11.129c-3.906-3.905-10.236-3.905-14.143,0    c-3.905,3.905-
3.905,10.237,0,14.143l18.201,18.199c1.876,1.875,4.419,2.929,7.071,2.929c2.652,0,5.195-1.054,7.071-2.929    l39.638-
39.638C130.009,361.866,130.009,355.534,126.104,351.629z"
                                fill="#FFFFFF"
```

```
                                data-original="#000000"
                                class=""
                              />
                            </g>
                          </g>
                        </g></svg
                      >

                      {#if userType === 'user'}
                        <a href="/profile/appointment"> <span>Appointments</span></a>
                      {:else if userType === 'doctor'}
                        <a href="/doctor"> <span>Dashboard</span></a>
                      {:else}
                        <a href="/admin/dashboard"> <span>Dashboard</span></a>
                      {/if}
                    </a>
                  </li>

                  <li class="flex">
                    <!-- svelte-ignore a11y-invalid-attribute -->
                    <a
                      href="#"
                      class="inline-flex items-center w-full px-2 py-1 text-sm font-semibold transition-colors duration-150
rounded-md hover:bg-gray-100 hover:text-gray-800 dark:hover:bg-gray-800 dark:hover:text-gray-200"
                      on:click={() => {
                        $session = null
                        $userProfileStore = null
                        goto('/logout')
                      }}
                    >
                      <svg
                        class="w-4 h-4 mr-3"
                        aria-hidden="true"
                        fill="none"
                        stroke-linecap="round"
                        stroke-linejoin="round"
                        stroke-width="2"
                        viewBox="0 0 24 24"
                        stroke="#FFFFFF"
                      >
                        <path
                          d="M11 16l-4-4m0 0l4-4m-4 4h14m-5 4v1a3 3 0 01-3 3H6a3 3 0 01-3-3V7a3 3 0 013-3h7a3 3 0
013 3v1"
                        />
                      </svg>
                      <span>Log out</span>
                    </a>
                  </li>
                </ul>
              </div>
            </div>
          {/if}
          <button
```

```
        data-collapse-toggle="mobile-menu-2"
        type="button"
        class="inline-flex items-center p-2 ml-1 text-sm text-black rounded-lg md:hidden hover:bg-gray-200 focus:outline-
none focus:ring-2 focus:ring-sky-200"
        aria-controls="mobile-menu-2"
        aria-expanded="false"
        on:click|preventDefault={() => (menu = !menu)}
      >
        <span class="sr-only">Open main menu</span>
        <svg
          class="w-6 h-6"
          fill="currentColor"
          viewBox="0 0 20 20"
          xmlns="http://www.w3.org/2000/svg"
          ><path
            fill-rule="evenodd"
            d="M3 5a1 1 0 011-1h12a1 1 0 110 2H4a1 1 0 01-1-1zM3 10a1 1 0 011-1h12a1 1 0 110 2H4a1 1 0 01-1-1zM3
15a1 1 0 011-1h12a1 1 0 110 2H4a1 1 0 01-1-1z"
            clip-rule="evenodd"
          /></svg
        >
        <svg
          class="hidden w-6 h-6"
          fill="currentColor"
          viewBox="0 0 20 20"
          xmlns="http://www.w3.org/2000/svg"
          ><path
            fill-rule="evenodd"
            d="M4.293 4.293a1 1 0 011.414 0L10 8.586l4.293-4.293a1 1 0 111.414 1.414L11.414 10l4.293 4.293a1 1 0
01-1.414 1.414L10 11.414l-4.293 4.293a1 1 0 01-1.414-1.414L8.586 10 4.293 5.707a1 1 0 010-1.414z"
            clip-rule="evenodd"
          /></svg
        >
      </button>
      <div
        class="{menu
          ? 'hidden'
          : ''} justify-between items-center w-full md:flex md:w-auto md:order-1"
        id="mobile-menu-2"
      >
        <ul class="flex flex-col mt-4 md:flex-row md:space-x-8 md:mt-0 md:text-sm md:font-semibold">
          <li>
            <a href="/" class="block px-3 py-2 pr-4 pl-3 mx-3 hover:text-[#fb3434]">Home</a>
          </li>
          <li>
            <a href="/about-us" class="block px-3 py-2 pr-4 pl-3 mx-3 hover:text-[#fb3434]">About</a>
          </li>
          <li>
            <a href="/support" class="block px-3 py-2 pr-4 pl-3 mx-3 hover:text-[#fb3434]">Support</a>
          </li>
          <li>
            <a href="/contact" class="block px-3 py-2 pr-4 pl-3 mx-3 hover:text-[#fb3434]">Contact</a>
          </li>
```

```
        </ul>
      </div>
    </div>
</nav>
```

- **Footer Component**

```
<footer class="text-gray-600 bg-slate-200 body-font">
    <div class="container px-5 py-5 mx-auto flex items-center sm:flex-row flex-col">
      <div class="p-2 mr-1 text-primary text-3xl font-bold tracking-wide font-poppins">
        <a href="/">Find<span class="text-[#fb3434]">Care</span></a>
        </div>
     <p class="text-sm text-gray-500 sm:ml-4 sm:pl-4 sm:border-l-2 sm:border-black sm:py-2 sm:mt-0 mt-4">© 2022
FindCare
     </p>

    </div>
  </footer>
```

- **Loading Component**

```
<script>
    import { Circle } from 'svelte-loading-spinners'
    import logo_png from '$lib/assets/logo.png'
</script>

<div class="min-h-screen flex flex-col items-center justify-center relative">
    <img src={logo_png} alt="logo" class="mb-4 w-32" />
    <Circle size="2" color="rgb(239, 68, 68)" unit="rem" duration="1.5s" />
</div>
```

- **Table Component**

```
<script>
    import { session } from '$app/stores'
    import { ENV, status_code } from '$lib/utils'
    import { notificationToast } from '$lib/NotificationToast'
    import TableDropdown from '$lib/components/TableDropdown.svelte'
    import { doctorDashBoardHeader } from '../../stores'
    export let patientImg
    export let appointment_id
    export let patientName
    export let dateOfAppointment
    export let status = 'completed'
    export let color = 'light'
    let is_loading_cancel = false
    let is_loading_complete = false
    async function completeAppointment() {
        is_loading_complete = true
```

```
    const res = await fetch(
      ENV.VITE_FINDCARE_API_BASE_URL +
        '/api/v1/doctor/clinic/appointment/completed?id=' +
        appointment_id,
      {
        headers: {
          'Content-type': 'application/json',
          //@ts-ignore
          Authorization: `Bearer ${$session.session}`
        }
      }
    )
    const resp = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/clinic/', {
      method: 'GET',
      headers: {
        'Content-type': 'application/json',
        //@ts-ignore
        Authorization: `Bearer ${$session.session}`
      }
    })
    if (resp.ok) {
      $doctorDashBoardHeader = await resp.json()
    }
    const data = await res.json()
    is_loading_complete = false
    if (res.status == status_code.HTTP_202_ACCEPTED) {
      notificationToast(data?.detail, false, 2000, 'success')
      status = 'completed'
    } else if (res.status == status_code.HTTP_409_CONFLICT)
      notificationToast(data?.detail, false, 2000, 'error')
    else notificationToast('Some unknown error occured', false, 2000, 'error')
  }
  async function cancelAppointment() {
    is_loading_cancel = true
    const res = await fetch(
      ENV.VITE_FINDCARE_API_BASE_URL +
        '/api/v1/doctor/clinic/appointment/cancel?id=' +
        appointment_id,
      {
        headers: {
          'Content-type': 'application/json',
          //@ts-ignore
          Authorization: `Bearer ${$session.session}`
        }
      }
    )
    const resp = await fetch(ENV.VITE_FINDCARE_API_BASE_URL + '/api/v1/doctor/clinic/', {
      method: 'GET',
      headers: {
        'Content-type': 'application/json',
        //@ts-ignore
        Authorization: `Bearer ${$session.session}`
      }
```

```
        })
        if (resp.ok) {
            $doctorDashBoardHeader = await resp.json()
        }
        const data = await res.json()
        is_loading_cancel = false
        if (res.status == status_code.HTTP_202_ACCEPTED) {
            notificationToast(data?.detail, false, 2000, 'error')
            status = 'cancelled'
        } else if (res.status == status_code.HTTP_409_CONFLICT)
            notificationToast(data?.detail, false, 2000, 'error')
        else notificationToast('Some unknown error occured', false, 2000, 'error')
    }
</script>

<tr>
    <th
        class="border-t-0 px-6 align-middle border-l-0 border-r-0 text-xs whitespace-nowrap p-4 text-left flex items-center"
    >
        <img src={patientImg} class="h-12 w-12 bg-white rounded-full border" alt="..." />
        <span class="ml-3 font-bold {color === 'light' ? 'btext-blueGray-600' : 'text-white'}">
            {patientName}
        </span>
    </th>
    <td class="border-t-0 px-6 align-middle border-l-0 border-r-0 text-xs whitespace-nowrap p-4">
        {dateOfAppointment}
    </td>
    <td
        class="border-t-0 px-6 align-middle border-l-0 border-r-0 text-xs whitespace-nowrap p-4 capitalize"
    >
        {#if status == 'pending'}
            <i class="fas fa-circle text-orange-500 mr-2" /> {status}
        {/if}
        {#if status == 'completed'}
            <i class="fas fa-circle text-teal-500 mr-2" /> {status}
        {/if}
        {#if status == 'cancelled'}
            <i class="fas fa-circle text-red-500 mr-2" /> {status}
        {/if}
    </td>
    <td class="border-t-0 px-6 align-middle border-l-0 border-r-0 text-xs whitespace-nowrap p-4">
        <div class="flex">
            <button
                disabled={status == 'completed' || status == 'cancelled' || is_loading_complete
                    ? true
                    : false}
                id={appointment_id}
                on:click|preventDefault={completeAppointment}
                type="button"
                class="text-white   focus:ring-2 focus:outline-none  font-medium rounded-lg text-sm px-5 py-2.5 text-center
inline-flex items-center mr-2  {status ==
                    'completed' || status == 'cancelled'
                    ? 'bg-green-900  cursor-not-allowed'
```

```
            : 'bg-green-700 hover:bg-green-800 focus:ring-green-300 dark:bg-green-600 dark:hover:bg-green-700
dark:focus:ring-green-800'}"
        >
          <i
            class="{is_loading_complete
              ? 'loading fa fa-spinner fa-spin'
              : status == 'completed' || status == 'cancelled'
              ? ''
              : 'fas fa-check '} mr-2"
          />
          {status == 'completed'
            ? 'Completed'
            : status == 'cancelled'
            ? 'Cancelled'
            : 'Mark as complete'}
        </button>
      </div>
    </td>
    <td class="border-t-0 px-6 align-middle border-l-0 border-r-0 text-xs whitespace-nowrap p-4">
      <div class="flex items-center">
        <button
          id={appointment_id}
          disabled={status == 'completed' || status == 'cancelled' || is_loading_cancel
            ? true
            : false}
          type="button"
          on:click|preventDefault={cancelAppointment}
          class="text-white  focus:ring-2 focus:outline-none  font-medium rounded-lg text-sm px-5 py-2.5 text-center
inline-flex items-center mr-2  {status ==
            'completed' || status == 'cancelled'
            ? 'bg-red-900 cursor-not-allowed'
            : 'bg-red-700 hover:bg-red-800 focus:ring-red-300 dark:bg-red-600 dark:hover:bg-red-700 dark:focus:ring-red-
800'}"
        >
          <i
            class="{is_loading_cancel
              ? 'loading fa fa-spinner fa-spin'
              : status == 'completed' || status == 'cancelled'
              ? ''
              : 'fas fa-xmark'} mr-2"
          />
          {status == 'cancelled' ? 'Cancelled' : status == 'completed' ? 'Completed' : 'Cancel'}
        </button>
      </div>
    </td>
</tr>
```

- **Card Component**

```
<script>
  // core components
  export let statSubtitle = "Total Appointment";
```

```
  export let statTitle = "350";

  export let statIconName = "far fa-chart-bar";
  // can be any of the background color utilities
  // from tailwindcss
  export let statIconColor = "bg-red-500";
</script>

<div
  class="relative flex flex-col min-w-0 break-words bg-white rounded mb-6 xl:mb-0 shadow-lg"
>
  <div class="flex-auto p-4">
    <div class="flex flex-wrap">
      <div class="relative w-full pr-4 max-w-full flex-grow flex-1">
        <h5 class="text-blueGray-400 mb-3 uppercase font-bold text-xs">
          {statSubtitle}
        </h5>
        <span class="font-semibold text-xl text-blueGray-700">
          {statTitle}
        </span>
      </div>
      <div class="relative w-auto pl-4 flex-initial">
        <div
          class="text-white p-3 text-center inline-flex items-center justify-center w-12 h-12 shadow-lg rounded-full
{statIconColor}"
        >
          <i class="{statIconName}"></i>
        </div>
      </div>
    </div>
  </div>
</div>
```

- **Sidebar Component**

```
<script context="module">
  export let selected = 'changepass'
</script>

<script>
  function toggleCollapseShow(classes) {
    collapseShow = classes
  }
  let collapseShow = 'hidden'

</script>

<nav
    class="md:left-0 md:block md:fixed md:top-0 md:bottom-0 md:overflow-y-auto md:flex-row md:flex-nowrap
md:overflow-hidden shadow-xl bg-white flex flex-wrap items-center justify-between relative md:w-64 z-10 py-4 px-6"
  >
    <div
```

```
        class="md:flex-col md:items-stretch md:min-h-full md:flex-nowrap px-0 flex flex-wrap items-center justify-between w-full mx-auto"
      >
        <!-- Toggler -->
        <button
          class="cursor-pointer text-black opacity-50 md:hidden px-3 py-1 text-xl leading-none bg-transparent rounded border border-solid border-transparent"
          type="button"
          on:click={() => toggleCollapseShow('bg-white m-2 py-3 px-6')}
        >
          <i class="fas fa-bars" />
        </button>
        <!-- Brand -->
        <a
          class="md:block text-left md:pb-2 text-blueGray-600 mr-0 inline-block whitespace-nowrap text-sm uppercase font-bold p-4 px-0"
          href="/"
        >
          <div class="p-2 text-primary text-3xl font-bold tracking-wide font-poppins">
            <a href="/">Find<span class="text-[#fb3434]">Care</span></a>
          </div>
        </a>
        <!-- User -->
        <ul class="md:hidden items-center flex flex-wrap list-none">
          <li class="inline-block relative" />
          <li class="inline-block relative" />
        </ul>
        <!-- Collapse -->
        <div
          class="md:flex md:flex-col md:items-stretch md:opacity-100 md:relative md:mt-4 md:shadow-none shadow absolute top-0 left-0 right-0 z-40 overflow-y-auto overflow-x-hidden h-auto items-center flex-1 rounded {collapseShow}"
        >
          <!-- Collapse header -->
          <div
            class="md:min-w-full md:hidden block pb-4 mb-4 border-b border-solid border-blueGray-200"
          >
            <div class="flex flex-wrap">
              <div class="w-6/12">
                <a
                  class="md:block text-left md:pb-2 text-blueGray-600 mr-0 inline-block whitespace-nowrap text-sm uppercase font-bold p-2 px-0"
                  href="/"
                >
                  <div class=" text-primary text-3xl font-bold tracking-wide font-poppins">
                    <a href="/">Find<span class="text-[#fb3434]">Care</span></a>
                  </div>
                </a>
              </div>
              <div class="w-6/12 flex justify-end">
                <button
                  type="button"
                  class="cursor-pointer text-black opacity-50 md:hidden px-3 py-1 text-xl leading-none bg-transparent rounded border border-solid border-transparent"
```

```
            on:click={() => toggleCollapseShow('hidden')}
          >
            <i class="fas fa-times" />
          </button>
        </div>
      </div>
    </div>
    <!-- Divider -->
    <hr class="mb-4 md:min-w-full" />
    <!-- Heading -->
    <h6
      class="md:min-w-full text-blueGray-500 text-xs uppercase font-bold block pt-1 pb-4 no-underline"
    >
      Admin Layout Pages
    </h6>
    <!-- Navigation -->

    <ul class="md:flex-col md:min-w-full flex flex-col list-none">
      <li class="items-center">
        <button
          class="text-xs uppercase py-3 font-bold block  "
          on:click={() => (selected = 'dashboard')}
        >
          <i class="fas fa-tv mr-2 text-sm text-blueGray-300 " />
          Dashboard
        </button>
      </li>

      <li class="items-center">
      <button
          class="text-blueGray-700 hover:text-blueGray-500 text-xs uppercase py-3 font-bold block"
          on:click={() => (selected = 'Account Setting')}
        >
          <i class="fas fa-user-circle text-blueGray-300 mr-2 text-sm" />
          Account Setting
        </button>
      </li>
      <li class="items-center">
        <button
          class="text-xs uppercase py-3 font-bold block"
          on:click={() => (selected = 'clinic')}
        >
          <i class="fas fa-hospital mr-2 text-sm " />
          Clinic
        </button>
      </li>
      <li class="items-center">
        <button
          class="text-xs uppercase py-3 font-bold block "
          on:click={() => (selected = 'changepass')}
        >
          <i class="fas fa-key mr-2 text-sm " />
          Change Password
```

```
                </button>
              </li>
              <li class="items-center">
                <a href="/logout" class="text-xs uppercase py-3 font-bold block ">
                  <i class="fas fa-arrow-right-from-bracket mr-2 text-sm " />
                  Logout
                </a>
              </li>
            </ul>

            <!-- Divider -->
            <hr class="my-4 md:min-w-full" />
          </div>
        </div>
      </nav>
```

- **Dashboard Table**

```
<script>
  // core components
  import TableStats from '$lib/components/TableStats.svelte'
  import { getFormattedDateDashBoard } from '$lib/utils'
  // can be one of light or dark
  export let color = 'light'
  export let response
  let patients = response?.patients
  function checkAppointSatus(appointment) {
    if (appointment.is_completed) {
      return 'completed'
    } else if (appointment.is_cancelled) {
      return 'cancelled'
    }
    return 'pending'
  }
</script>

{#if response && patients}
  <div
    class="relative flex flex-col min-w-0 break-words w-full mb-6 shadow-lg rounded {color ===
    'light'
      ? 'bg-white'
      : 'bg-red-800 text-white'}"
  >
    <div class="rounded-t mb-0 px-4 py-3 border-0">
      <div class="flex flex-wrap items-center">
        <div class="relative w-full px-4 max-w-full flex-grow flex-1">
          <h3
            class="font-semibold text-lg {color === 'light' ? 'text-blueGray-700' : 'text-white'}"
          >
            Appointment Detials
          </h3>
```

```
            </div>
          </div>
        </div>
        <div class="block w-full overflow-x-auto">
          <!-- Projects table -->
          <table class="items-center w-full bg-transparent border-collapse">
            <thead>
              <tr>
                <th
                  class="px-6 align-middle border border-solid py-3 text-xs uppercase border-l-0 border-r-0 whitespace-
nowrap font-semibold text-left {color ===
                  'light'
                    ? 'bg-blueGray-50 text-blueGray-500 border-blueGray-100'
                    : 'bg-red-700 text-red-200 border-red-600'}"
                >
                  Patient's Details
                </th>
                <th
                  class="px-6 align-middle border border-solid py-3 text-xs uppercase border-l-0 border-r-0 whitespace-
nowrap font-semibold text-left {color ===
                  'light'
                    ? 'bg-blueGray-50 text-blueGray-500 border-blueGray-100'
                    : 'bg-red-700 text-red-200 border-red-600'}"
                >
                  Date of Appointment
                </th>
                <th
                  class="px-6 align-middle border border-solid py-3 text-xs uppercase border-l-0 border-r-0 whitespace-
nowrap font-semibold text-left {color ===
                  'light'
                    ? 'bg-blueGray-50 text-blueGray-500 border-blueGray-100'
                    : 'bg-red-700 text-red-200 border-red-600'}"
                >
                  Status
                </th>
                <th
                  class="px-6 align-middle border border-solid py-3 text-xs  uppercase border-l-0 border-r-0 whitespace-
nowrap font-semibold text-left {color ===
                  'light'
                    ? 'bg-blueGray-50 text-blueGray-500 border-blueGray-100'
                    : 'bg-red-700 text-red-200 border-red-600'}"
                >
                  Mark as Completed
                </th>
                <th
                  class="px-6 align-middle border border-solid py-3 text-xs uppercase border-l-0 border-r-0 whitespace-
nowrap font-semibold text-left {color ===
                  'light'
                    ? 'bg-blueGray-50 text-blueGray-500 border-blueGray-100'
                    : 'bg-red-700 text-red-200 border-red-600'}"
                >
                  Cancel
                </th>
```

```
            </tr>
          </thead>
          <tbody>
            {#each patients as patient}
              {#each patient.appointments as appointment}
                <TableStats
                  appointment_id={appointment.id}
                  patientImg={patient.profile_image}
                  patientName={patient.name}
                  dateOfAppointment={getFormattedDateDashBoard(appointment.schedule)}
                  status={checkAppointSatus(appointment)}
                />
              {/each}
            {/each}
          </tbody>
        </table>
      </div>
    </div>
{:else}
    <div class="text-center text-lg p-16 font-semibold">Appointments Not Available</div>
{/if}
```

- **Dashboard Stats Component**

```
<script>
    import CardStats from '$lib/components/cardStats.svelte'
    export let response
</script>

<!-- Header -->
<div class="relative bg-red-500 md:pt-28 pb-12 pt-12">
    <div class="px-4 md:px-10 mx-auto w-full">
        <div>
            <!-- Card stats -->
            <div class="flex flex-wrap">
                <div class="w-full lg:w-6/12 xl:w-3/12 px-4">
                    <CardStats
                        statSubtitle="Total Appointments"
                        statTitle={response ? response.total_appointments.toLocaleString() : 0}
                        statIconName="far fa-chart-bar"
                        statIconColor="bg-red-500"
                    />
                </div>
                <div class="w-full lg:w-6/12 xl:w-3/12 px-4">
                    <CardStats
                        statSubtitle="Completed Appointments"
                        statTitle={response ? response.completed_appointments.toLocaleString() : 0}
                        statIconName="fa-solid fa-calendar-check"
                        statIconColor="bg-orange-500"
                    />
                </div>
                <div class="w-full lg:w-6/12 xl:w-3/12 px-4">
```

```
            <CardStats
              statSubtitle="Pending Appointments"
              statTitle={response ? response.pending_appointments.toLocaleString() : 0}
              statIconName="fa-solid fa-clock-rotate-left"
              statIconColor="bg-[#F29339]"
            />
          </div>
          <div class="w-full lg:w-6/12 xl:w-3/12 px-4">
            <CardStats
              statSubtitle="Today's Appointment"
              statTitle={response ? response.today_appointments.toLocaleString() : 0}
              statIconName="fas fa-calendar-plus"
              statIconColor="bg-emerald-500"
            />
          </div>
          <div class="w-full lg:w-6/12 xl:w-3/12 px-4 sm:mt-4">
            <CardStats
              statSubtitle="Total Patient"
              statTitle={response ? response.total_patients.toLocaleString() : 0}
              statIconName="fas fa-users"
              statIconColor="bg-pink-500"
            />
          </div>
          <div class="w-full lg:w-6/12 xl:w-3/12 px-4 sm:mt-4">
            <CardStats
              statSubtitle="Appointments Cancelled By User"
              statTitle={response ? response.cancelled_appointments_by_user.toLocaleString() : 0}
              statIconName="fa-solid fa-xmark"
              statIconColor="bg-[#D9512C]"
            />
          </div>
          <div class="w-full lg:w-6/12 xl:w-3/12 px-4 sm:mt-4">
            <CardStats
              statSubtitle="Appointments Cancelled By You"
              statTitle={response ? response.cancelled_appointments_by_doctor.toLocaleString() : 0}
              statIconName="fa-solid fa-xmark"
              statIconColor="bg-[#D9512C]"
            />
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

- **Backend**

➢ Main.py

```
from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware
```

```
from app.Config import settings
from app.routers import root
from app.models import user_model, clinic_model, doctor_model, appointment_model
from app.database import engine
from app.routers.v1 import auth, user_router, doctor_router, clinic_router, \
    appointment_router, search_doctor_clinics_router, admin_router, email_router,
temporary_router


user_model.Base.metadata.create_all(bind=engine)
clinic_model.Base.metadata.create_all(bind=engine)
doctor_model.Base.metadata.create_all(bind=engine)
appointment_model.Base.metadata.create_all(bind=engine)


app = FastAPI(
    title="FindCare-API",
    version="1.0",
)

origins = [
    "https://findcare-api-ryuk-me.cloud.okteto.net",
    "http://findcare-api-ryuk-me.cloud.okteto.net",
    "https://findcare-ryuk-me.cloud.okteto.net",
    "http://findcare-ryuk-me.cloud.okteto.net",
    '127.0.0.1',
    '0.0.0.0',
    'localhost',
    "http://localhost:3000",
    'http://127.0.0.1:8009', 'http://127.0.0.1:8009/*',
    "http://localhost:3000/*",
    "https://findcare-api-ryuk-me.cloud.okteto.net",
    "https://findcare-api-ryuk-me.cloud.okteto.net/*",
    "https://findcare-ryuk-me.cloud.okteto.net", "https://findcare-ryuk-
me.cloud.okteto.net/*"
]

app.add_middleware(
    CORSMiddleware,
    allow_origins=origins,
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

app.include_router(root.router)
app.include_router(user_router.router)
app.include_router(auth.router)
app.include_router(doctor_router.router)
app.include_router(clinic_router.router)
app.include_router(appointment_router.router)
app.include_router(search_doctor_clinics_router.router)
app.include_router(admin_router.router)
```

```
app.include_router(email_router.router)
app.include_router(temporary_router.router)
```

➢Database.py

```
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker
from app.Config import settings
from sqlalchemy_utils import database_exists, create_database

# SQLALCHEMY_DATABASE_URL = "postgresql://postgres:ryuk@localhost/fast-api-test"
SQLALCHEMY_DATABASE_URL =
f'postgresql://{settings.DATABASE_USERNAME}:{settings.DATABASE_PASSWORD}@{settings.DATABAS
E_HOSTNAME}:{settings.DATABASE_PORT}/{settings.DATABASE_NAME}'

engine = create_engine(SQLALCHEMY_DATABASE_URL)

if not database_exists(engine.url):
    create_database(engine.url)

SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative_base()
```

➢OAuth.py

```
from jose import JWTError, jwt
from datetime import datetime, timedelta
from fastapi import Depends
from sqlalchemy.orm import Session
from app.error_handlers import errors
from fastapi.security import OAuth2PasswordBearer
from app.models import user_model, doctor_model, admin_model
from app.scheams import token_schema
from app.Config import settings
from app import services as _services


oauth2_scheme_user = OAuth2PasswordBearer(
    tokenUrl=settings.BASE_API_V1 + '/auth/user', scheme_name="USER LOGIN")
oauth2_scheme_doctor = OAuth2PasswordBearer(
    tokenUrl=settings.BASE_API_V1 + '/auth/doctor', scheme_name="DOCTOR LOGIN")
oauth2_scheme_admin = OAuth2PasswordBearer(
    tokenUrl=settings.BASE_API_V1 + '/auth/admin', scheme_name="ADMIN LOGIN")


def create_access_token(data: dict, expires_delta: timedelta | None = None):
    to_encode = data.copy()
    expire = datetime.utcnow() + expires_delta
    to_encode.update({"exp": expire})
```

```
    encoded_jwt = jwt.encode(
        to_encode, settings.SECRET_KEY, algorithm=settings.ALGORITHM)
    return encoded_jwt


def verify_token(token: str, is_email_verification_token: bool = False,
is_reset_password_token: bool = False):
    try:
        payload = jwt.decode(token, settings.SECRET_KEY,
                             algorithms=[settings.ALGORITHM])
        id = payload.get("id")
        if id is None:
            if is_email_verification_token:
                raise errors.VERIFICATION_LINK_EXPIRED
            if is_reset_password_token:
                raise errors.PASSWORD_RESET_LINK_EXPIRED
            raise errors.TOKEN_CREDENTIALS_ERROR
        token_data = token_schema.TokenData(**payload)
    except JWTError:
        if is_email_verification_token:
            raise errors.VERIFICATION_LINK_EXPIRED
        raise errors.TOKEN_CREDENTIALS_ERROR
    return token_data


def get_current_user(token: str = Depends(oauth2_scheme_user), db: Session =
Depends(_services.get_db)):
    token = verify_token(token)
    user = db.query(user_model.User).filter(
        user_model.User.id == token.id).first()
    if not user:
        raise errors.TOKEN_CREDENTIALS_ERROR
    return user


def get_current_doctor(token: str = Depends(oauth2_scheme_doctor), db: Session =
Depends(_services.get_db)):
    token = verify_token(token)
    doctor = db.query(doctor_model.Doctor).filter(
        doctor_model.Doctor.id == token.id).first()
    if not doctor:
        raise errors.TOKEN_CREDENTIALS_ERROR
    return doctor


def get_current_admin(token: str = Depends(oauth2_scheme_admin), db: Session =
Depends(_services.get_db)):
    token = verify_token(token)
    admin = db.query(admin_model.Admin).filter(
        admin_model.Admin.id == token.id).first()
    if not admin:
        raise errors.TOKEN_CREDENTIALS_ERROR
    return admin
```

➢ Errors.py

```
from fastapi import HTTPException, status


# ********************************************************************************
#                                                                                #
#                               USER ERRORS                                      #
#                                                                                #
# ********************************************************************************


USER_NOT_FOUND = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail=f'User not found')

USER_ALREADY_EXIST = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="User already exist")

USER_ALREADY_BANNED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="User is already banned")

USER_ALREADY_UNBANNED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="User is not banned")

USER_IS_BANNED = HTTPException(
    status_code=status.HTTP_422_UNPROCESSABLE_ENTITY, detail="Your account has been
blocked. Please contact support for further details")

NO_USER_FOUND = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail=f'No user found')

# ********************************************************************************
#                                                                                #
#                               DOCTOR ERRORS                                    #
#                                                                                #
# ********************************************************************************


DOCTOR_ALREADY_EXIST = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Doctor already exist")

DOCTOR_NOT_FOUND = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail=f'Doctor not found')

NO_DOCTOR_FOUND_WITH_THIS_ID = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="No doctor found with this id")

DOCTOR_IS_ALREADY_VERIFIED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Doctor is already verified")

DOCTOR_WITH_THIS_REGISTRATION_NUM_ALREADY_EXIST = HTTPException(
```

```
    status_code=status.HTTP_409_CONFLICT, detail="Doctor with this registration number
already exist")

NOT_POSSIBLE_EXPERINCE_YEAR = HTTPException(
    status_code=status.HTTP_406_NOT_ACCEPTABLE, detail="Experince year not acceptable
please try lower value")

DOCTOR_IS_NOT_VERIFIED = HTTPException(
    status_code=status.HTTP_422_UNPROCESSABLE_ENTITY, detail="Doctor is not verified")

DOCTOR_IS_BANNED = HTTPException(
    status_code=status.HTTP_422_UNPROCESSABLE_ENTITY, detail="Doctor is banned")

DOCTOR_IS_ALREADY_BANNED = HTTPException(
    status_code=status.HTTP_422_UNPROCESSABLE_ENTITY, detail="Doctor is already banned")

DOCTOR_IS_ALREADY_UNBANNED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Doctor is not banned")

# ********************************************************************************
#                                                                                #
#                              ADMIN ERRORS                                      #
#                                                                                #
# ********************************************************************************


YOU_CANNOT_SET_PASSWORD_FOR_USER = HTTPException(
    status_code=status.HTTP_406_NOT_ACCEPTABLE, detail="You cannot set password for user")

YOU_CANNOT_SET_PASSWORD_FOR_DOCTOR = HTTPException(
    status_code=status.HTTP_406_NOT_ACCEPTABLE, detail="You cannot set password for
doctor")

PLEASE_CONTACT_ADMIN = HTTPException(
    status_code=status.HTTP_406_NOT_ACCEPTABLE, detail="Please contact Admin for support")


# ********************************************************************************
#                                                                                #
#                       FORBIDDEN ACTIONS ERRORS                                 #
#                                                                                #
# ********************************************************************************


INVALID_CREDENTIALS_ERROR = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail="Incorrect Email or Password.")


TOKEN_CREDENTIALS_ERROR = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail="Could not validate credentials",
headers={"WWW-Authenticate": "Bearer"})

FORBIDDEN_ACTION_ERROR = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail="Not authorised to perform this action")
```

```
PLEASE_LOGIN_FIRST = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail=f'Please login first')

NOT_A_SUPER_ADMIN = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail="You dont have enough permission to
perform this action")


# ********************************************************************************
#                                                                                #
#                              CLINIC ERRORS                                     #
#                                                                                #
# ********************************************************************************


ALREADY_EXIST_CLINIC = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Clinic already exist for this doctor")

CLINIC_NOT_FOUND = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="Clinic not found")

CLINIC_IS_NOT_SERVICEABLE = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="Clinic is not servicable currently")


# ********************************************************************************
#                                                                                #
#                         APPOINTMENT ERRORS                                     #
#                                                                                #
# ********************************************************************************


APPOINTNEMT_ALREADY_CANCELLED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Appointment is already cancelled")

APPOINTMENT_ALREADY_SKIPPED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Appointment is already skipped")

APPOINTNEMT_ALREADY_CANCELLED_BY_USER = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Appointment is already cancelled by the
user")

APPOINTMENT_ALREADY_CANCELLED_BY_DR = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Appointment is already cancelled by the
doctor")

APPOINTMENT_ALREADY_COMPLETED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Appointment is already completed")

APPOINTMENT_SKIPPED_CANCELLATION = HTTPException(
    status_code=status.HTTP_409_CONFLICT,
    detail="Appointment cancelled due to patient was not on time"
)
```

```
NO_APPOINTMENT_FOUND_ERROR = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="No appointment found")

NO_CANCELLATION_REASON = HTTPException(
    status_code=status.HTTP_422_UNPROCESSABLE_ENTITY, detail="Please enter a cancellation
reason")

TIME_SLOT_NOT_AVAILABLE = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Time slot not available")

NOTHING_CHANGED = HTTPException(
    status_code=status.HTTP_304_NOT_MODIFIED, detail="Not modified")

# ********************************************************************************
#                                                                                #
#                              MISC ERRORS                                       #
#                                                                                #
# ********************************************************************************


NOT_FOUND_ERROR = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="Not found")


PHONE_NUMBER_ALREADY_EXIST = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="An account with this phone number
already exist")

EMAIL_ALREADY_EXIST = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="An account with this email already
exist")

PASSWORD_CANNOT_BE_SAME = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Password cannot be same as the old one")

ACCOUNT_NOT_FOUND_WITH_THIS_EMAIL = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="No account found with this email")

NO_SPECIALITY_FOUND = HTTPException(
    status_code=status.HTTP_404_NOT_FOUND, detail="No speciality found")

# ********************************************************************************
#                                                                                #
#                         EMAIL TOKEN ERRORS                                     #
#                                                                                #
# ********************************************************************************


VERIFICATION_LINK_EXPIRED = HTTPException(
    status_code=status.HTTP_410_GONE, detail="Verification link has been expired")

PASSWORD_RESET_LINK_EXPIRED = HTTPException(
    status_code=status.HTTP_410_GONE, detail="Password reset link has been expired")
```

```
EMAIL_ALREADY_VERIFIED = HTTPException(
    status_code=status.HTTP_409_CONFLICT, detail="Email already verified")

PLEASE_VERIFY_YOUR_EMAIL = HTTPException(
    status_code=status.HTTP_403_FORBIDDEN, detail="Please verify your Email first")
```

8. *Testing*

The testing of website means measuring or accessing the website to determine the quality. Testing is a measuring instrument for web app quality with the unit of measurement being the number of defects found during testing.

➢ Testing Plan Used:

- Unit Testing
- Integration Testing
- System Testing
- Acceptance Testing

➢ Unit Testing

Unit testing comprises the set of tests performed usually by the programmers prior to the integration of the unit in to a large program. This is the lowest level of testing and is done by the programmer (who develops it) who can test it in great detail.

The function is done in isolation. This is where the most detailed investigation of internal working of the individual unit is carried out.

➢ Integration Testing

When two or more tested components are combined into a larger structure, the testing process should look for errors in two ways:

- In the interface between the components
- The functions, which can be performed by the new group

➢ System Testing

After integration testing is completed, the entire system is tested as whole. The functional specifications or requirements specification are used to derive the test case. At this level the system testing looks for errors in the end-to-end functional quality.

Attributes such as performance, reliability, volume, stress tolerance, usability, maintainability, security etc. Independent testers can carry out this testing.

➢ Acceptance Testing

After system testing was complete, the system was handed over to the training section. Acceptance testing mark the transaction from ownership by the develop to ownership by the users.

The acceptance test is different in nature to the development testing. Acceptance testing gave confidence to the user that the system is ready for operational use.

White Box Testing: This testing is also called as glass box testing. In this testing, by knowing the specified function that a product has been designed to perform test can be conducted that demonstrates each function is fully operation at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Black Box Testing: In this testing by knowing the internal operation of a product, tests can be conducted to ensure that "all gears mesh", that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

➢ Testing Objective

- No bugs found in the execution of tasks
- System states are visible
- All factors affecting the output are visible
- Functional simplicity
- Distinct output is generated for each input

➢ Remark

The whole system is tested thoroughly, passed all test cases and found no error during the testing period. The system is ready to go in the final execution phase.

9. *Maintenance & Future Scope Of The Project*

As this software is a web application so the maintenance **FindCare** is very easy. Due to it's clean and easy to use application, anyone can use this website without having any problem.

In case of any bug or problem is found during the usage of the website then anyone can send their feedback directly to the **Developers** of **FindCare** through the **Contact Page** of the website or you can find the contact detail of any **particular Developer** of this project then you can visit the **About Page** of **FindCare.** So that we can look into the matter and the required action to fix the problem.

The team would be delighted to hear from its user and provide all types of support related to the system, so that user can make himself familiar with the application.

In future, we are planning to add some extra features and expand our network to various cities so that this could be available to everyone.

One of the features we are planning to add is the **Diseases Detection feature,** in which you'll be able to know the name of disease by just entering your symptoms.

10. *Limitations of the Project*

- Limited Time Project.
- There is no Windows Based Application for it.
- This project can run only in **Web Browsers.**

11. *Conclusion*

Taking into account all the mentioned details, we can make the conclusion that the **FindCare** is the inevitable part of the lifecycle of the modern medical institution. It automates the appointment booking and enables smooth interactions of the users. Developing this appointment booking system is a great opportunity to create the distinct, efficient and fast delivering healthcare model.

Implementation of **FindCare** project helps to store all the kinds of records, provide coordination and user communication, implement policies, improve day-to-day operations, arrange the supply chain, manage financial and human resources, and market hospital services. This beneficial decision covers the needs of the patients, staff and hospital authorities and simplifies their interactions. It has become the usual approach to manage the hospital. Many clinics have already experienced its advantages and continue developing new hospital management system project modules.

12. ***Bibliography/References***

➢Websites

- **Reference**
  - Practo - https://www.practo.com/
  - Lybrate - https://www.lybrate.com/
  - Medibuddy - https://www.medibuddy.in/

- **Documentations**
  - Sveltekit - https://kit.svelte.dev/docs/
  - Python - https://docs.python.org/3/
  - PostgresSQL - https://www.postgresql.org/docs/
  - TailwindCSS - https://tailwindcss.com/docs/