

Design a Model of Application for eHealth

Ornella Rezgar Byea Sher

Bachelor's thesis
2023



Tomas Bata University in Zlín
Faculty of Applied Informatics

Tomas Bata University in Zlín
Faculty of Applied Informatics
Department of Informatics and Artificial Intelligence

Academic year: 2022/2023

ASSIGNMENT OF BACHELOR THESIS

(project, art work, art performance)

Name and surname: **Ornella Rzgar Byea Sher**
Personal number: **A19910**
Study programme: **B0613A140021 Software Engineering**
Type of Study: **Full-time**
Work topic: **Návrh modelu aplikace pro elektronické zdravotnictví**
Work topic in English: **Design a Model of Application for eHealth**

Theses guidelines

1. Prepare the literature review of the thesis topic.
2. Briefly describe the technologies that will be used.
3. Describe and analyse the requirements for the solution.
4. Create the HTML prototype of the proposed application.
5. Describe the security options for web applications.

Form processing of bachelor thesis: **printed/electronic**

Recommended resources:

1. ARLOW, Jim; NEUSTADT, Ila. UML 2 and the unified process: practical object-oriented analysis and design. Pearson Education, 2005. JOHNSON, Glenn. Programming in HTML5 with JavaScript and CSS3: training guide. Redmond, Wash.: Microsoft, 2013. ISBN 978-0735674387.
2. JOHNSON, Glenn. Programming in HTML5 with JavaScript and CSS3: training guide. Redmond, Wash.: Microsoft, 2013. ISBN 978-0735674387.
3. UNHELKAR, Bhuvan. Software engineering with uml Auerbach Publications, 2017.
4. LETT, Jacob. Bootstrap 4 Quick Start: A Beginners Guide to Building Responsive Layouts with Bootstrap 4 Bootstrap Creative, 2018.
5. AKOBUS, Benjamin. Mastering Bootstrap 4: Master the latest version of Bootstrap 4 to build highly customized responsive web apps Packt Publishing Ltd, 2018.
6. BEN-GAN, Itzik; DAVIDSON, Louis; VARGA, Stacia. MCSA SQL Server 2016 Database Development Exam Ref 2-pack: Exam Refs 70-761 and 70-762 Microsoft Press, 2017.

Supervisors of bachelor thesis: **doc. Ing. Petr Šilhavý, Ph.D.**
Department of Computer and Communication Systems

Date of assignment of bachelor thesis: **October 5, 2022**

Submission deadline of bachelor thesis: **May 12, 2023**

doc. Ing. Jiří Vojtěšek, Ph.D. m.p.
Dean



prof. Mgr. Roman Jašek, Ph.D., DBA m.p.
Head of Department

In Zlín October 10, 2022

I hereby declare that:

- I understand that by submitting my Bachelor's Thesis, I agree to the publication of my work according to Law No. 111/1998, Coll., On Universities and on changes and amendments to other acts (e.g. the Universities Act), as amended by subsequent legislation, without regard to the results of the defence of the thesis.
- I understand that my Bachelor's Thesis will be stored electronically in the university information system and be made available for on-site inspection, and that a copy of the Bachelor's Thesis will be stored in the Reference Library of the Faculty of Applied Informatics, Tomas Bata University in Zlín, and that a copy shall be deposited with my Supervisor.
- I am aware of the fact that my Bachelor's Thesis is fully covered by Act No. 121/2000 Coll. On Copyright, and Rights Related to Copyright, as amended by some other laws (e.g. the Copyright Act), as amended by subsequent legislation; and especially, by §35, Para. 3.
- I understand that, according to §60, Para. 1 of the Copyright Act, TBU in Zlín has the right to conclude licensing agreements relating to the use of scholastic work within the full extent of §12, Para. 4, of the Copyright Act.
- I understand that, according to §60, Para. 2, and Para. 3, of the Copyright Act, I may use my work - Bachelor's Thesis, or grant a license for its use, only if permitted by the licensing agreement concluded between myself and Tomas Bata University in Zlín with a view to the fact that Tomas Bata University in Zlín must be compensated for any reasonable contribution to covering such expenses/costs as invested by them in the creation of the thesis (up until the full actual amount) shall also be a subject of this licensing agreement.
- I understand that, should the elaboration of the Bachelor's Thesis include the use of software provided by Tomas Bata University in Zlín or other such entities strictly for study and research purposes (i.e. only for non-commercial use), the results of my Bachelor's Thesis cannot be used for commercial purposes.
- I understand that, if the output of my Bachelor's Thesis is any software product(s), this/these shall equally be considered as part of the thesis, as well as any source codes, or files from which the project is composed. Not submitting any part of this/these component(s) may be a reason for the non-defence of my thesis.

I herewith declare that:

- I have worked on my thesis alone and duly cited any literature I have used. In the case of the publication of the results of my thesis, I shall be listed as co-author.
- That the submitted version of the thesis and its electronic version uploaded to IS/STAG are both identical.

In Zlín; dated:

Student's Signature

ABSTRAKT

Tato práce si klade za cíl navrhnout webovou stránku pro eHealth, přičemž zkoumá klíčové technologie použité k jejímu vytvoření. Cíle projektu zahrnují plánování, shromažďování požadavků, prototypování, zohledňování bezpečnosti a základní nástroje a technologie potřebné pro návrh webové stránky eHealth. Výsledkem má být esteticky přitažlivá a intuitivní webová stránka, která zajišťuje optimální výkon, s důrazem na bezpečnost uživatelských dat. Práce se také zabývá návrhem uživatelsky přívětivých webových stránek pro eHealth a potvrzuje, že intuitivní navigace je klíčová pro spokojenost a angažovanost uživatelů.

Klíčová slova: Návrh, Prototyp, UML

ABSTRACT

This thesis aims to design a successful eHealth website while exploring the key technologies involved in its construction. The project objectives encompass planning, requirements gathering, prototyping, security considerations, and the fundamental tools and technologies required for designing the eHealth website. The ultimate goal is to create an aesthetically pleasing and intuitive website that delivers optimal performance, with special emphasis on ensuring the security of user data. The research also addresses the challenge of designing user-friendly eHealth websites, recognizing that intuitive navigation is crucial for user satisfaction and engagement.

Keywords: Design, Prototype, UML

ACKNOWLEDGMENT

I would like to express my deepest gratitude and appreciation to all those who have supported and contributed to the completion of this thesis.

First and foremost, I am indebted to my supervisor, Assoc.Prof. Petr Šilhavý, Assoc.Prof. Radek Šilhavý for their invaluable guidance, unwavering support, and continuous encouragement throughout the research process. Their expertise, insightful feedback, and dedication to my academic growth have been instrumental in shaping the direction and quality of this work.

My heartfelt appreciation goes to my family for their unwavering love, encouragement, and unwavering belief in my abilities. Their constant support, understanding, and sacrifices have been a constant source of inspiration and motivation.

I am also indebted to my friends and colleagues who have provided support, intellectual discussions, and assistance throughout this journey. Their encouragement, insightful conversations, and willingness to lend a helping hand have been invaluable.

I would like to extend a special thank you to Dr.Tomas Vogeltanz for his exceptional assistance and support. His expertise, technical knowledge, and willingness to go above and beyond have significantly contributed to the success of this research. His guidance and mentorship have been invaluable, and I am grateful for his unwavering belief in my abilities.

Thank you all for your contributions, both big and small, that have shaped this thesis into what it is today.

CONTENTS

INTRODUCTION	10
I THEORY.....	11
1 APPLICATIONS AND TECHNOLOGY IN HEALTH CARE INDUSTRY	12
1.1 ELECTRONIC HEALTH RECORD (EHR).....	12
1.2 TELEMEDICINE	12
1.3 MOBILE HEALTH(MHEALTH).....	13
1.4 ELECTRONIC HEALTH INFORMATION EXCHANGE (HIE).....	13
1.5 EVALUATION OF FUNCTIONALITIES IN CURRENT eHEALTH DEMO APPLICATIONS.....	13
1.5.1 DocTry	13
1.5.2 Emantal Hospital System	15
2 HEALTH SOLUTIONS AND THEIR LIMITATIONS	17
2.1 EHR LIMITATIONS	17
2.1.1 TELEMEDICINE LIMITATIONS	17
2.1.2 MHealth LIMITATIONS.....	17
2.1.3 HIE LIMITATIONS	17
3 BRIEFLY DESCRIBE THE TECHNOLOGIES BEING USED.....	18
3.1 HTML.....	18
3.1.1 Advantages Of HTML	18
3.1.2 DISADVANTAGES OF HTML.....	19
3.2 CSS.....	19
3.2.1 ADVANTAGES OF CSS	19
3.2.2 DISADVANTAGES OF CSS.....	19
3.3 JAVASCRIPT	20
3.3.1 ADVANTAGES OF JAVASCRIPT.....	20
3.3.2 DISADVANTAGES OF JAVASCRIPT	20
3.4 BOOTSTRAP 5.....	20
3.4.1 ADVANTAGES OF BOOTSTRAP	21
3.4.2 DISADVANTAGES OF BOOTSTRAP.....	21
3.5 UML(UNIFIED MODELING LANGUAGE)	21
3.5.1 Functional requirements.....	22
3.5.2 Non-functional requirements	22
3.5.3 Use Case Diagram.....	22
3.5.4 Class Model.....	22
3.5.5 Sequence Diagram	23
3.5.6 Wireframe	23
3.6 FIGMA	23

II	ANALYSIS.....	24
4	DESIGN OF THE EHEALTH APPLICATION.....	25
4.1	FUNCTIONAL REQUIREMENTS	25
4.2	NON-FUNCTIONAL REQUIREMENTS:	27
4.3	USE CASE DIAGRAMS	28
4.3.1	UCO1-01 Register.....	28
4.3.2	UCO2-02 Log in	29
4.3.3	UCO3-03 Schedule an appointment.....	30
4.3.4	UCO4-04 View Appointment	31
4.3.5	UCO5-05 Update Appointment	32
4.3.6	UCO6-06 Cancel appointment.....	32
4.3.7	UCO7-07: View patient record	33
4.3.8	UCO8-08: Create Prescription	33
4.3.9	UCO9-09: View Prescription.....	34
4.3.10	UCO10-10: Update Prescription	34
4.3.11	UCO11-11 Add Patient	35
4.3.12	UCO12-12 Delete Patient	36
4.3.13	UCO13-13 Update Patient	36
4.4	CLASS MODEL DIAGRAM	37
4.5	CREATE ACCOUNT SEQUENCE DIAGRAM	38
4.5.1	Login Sequence Diagram	39
4.5.2	Create Appointment Management Diagram	40
4.5.3	View Appointment Sequence Diagram.....	42
4.5.4	Update Appointment Sequence Diagram.....	43
4.5.5	Cancel Appointment Sequence Diagram	45
4.5.6	Medical Record Sequence Diagram.....	46
4.5.7	Create Prescription Sequence Diagram.....	48
4.5.8	View Prescription Sequence Diagram	49
4.5.9	Update Prescription Sequence Diagram.....	50
4.5.10	Add Patient Sequence Diagram	52
4.5.11	Delete Patient Sequence Diagram.....	53
4.5.12	Update Patient Sequence Diagram.....	54
4.5.13	UML Wireframe.....	55
5	HTML PROTOTYPE.....	59
5.1	THE FIGMA WIREFRAME	59
5.1.1	Low Fidelity Wireframe: Home Page	60
5.1.2	Low Fidelity Wireframe: Service section	61
5.1.3	Low Fidelity Wireframe: Footer Section	62
5.1.4	Low Fidelity Wireframe: Login pages	62
5.1.5	Low Fidelity Wireframe: Sign Up	63
5.1.6	Low Fidelity Wireframe: Hospitals page	64
5.1.7	Wireframe 6 : Doctors page	65
5.1.8	Low FidelityWireframe: Medicines Page	66
5.1.9	High Fidelity Wireframes: Hero Section	67
5.1.10	High in Fidelity wireframes: Service Section	68

5.1.11	High-Fidelity Wireframe: Footer Section	69
5.1.12	High-Fidelity Wireframe: Doctors Login	70
5.1.13	High-Fidelity wireframes: Doctors Page	71
5.1.14	High Fidelity Wireframe: Hospitals Page	72
5.1.15	High Fidelity Wireframe: Medicines Page	73
6	IMPLEMENTING THE DESIGN INTO CODE.....	74
6.1	HOME PAGE	75
6.2	DOCTOR LOGIN	76
6.3	PATIENT LOGIN	77
6.4	USER-SPECIFIC DASHBOARD: DOCTOR	78
6.5	SCHEDULE AN APPOINTMENT: DOCTORS DASHBOARD.....	81
6.6	ADD CONSULTATIONS	84
6.7	USER-SPECIFIC DASHBOARD PATIENT	85
6.8	PATIENT DASHBOARD APPOINTMENTS.....	86
6.9	THE CSS CODE	88
6.9.1	Nav Login Button Style	90
7	DESCRIBE THE SECURITY OPTIONS FOR WEB APPLICATION.....	91
7.1	HTML INJECTION	92
7.1.1	How To Protect Website Against HTML Injection	93
7.2	JAVASCRIPT INJECTION	93
7.2.1	How To Protect Website Against JavaScript Injection.....	94
7.3	SQL INJECTION	95
7.3.1	How To Protect A Website Against SQL Injection	95
	CONCLUSION	96
	BIBLIOGRAPHY	97
	LIST OF ABBREVIATIONS	100
	LIST OF FIGURES	101
	APPENDICES.....	103

INTRODUCTION

There is a discernible rise in the use of technology for online information seeking in the current digital era. Websites must provide high-quality content and services while maintaining an attractive and user-friendly design as more consumers rely on technology as their primary means of information gathering and service access.

The goal of this thesis is to establish and design a successful eHealth website while investigating the major technologies used in its construction. Project objectives will include planning, requirements collecting, prototyping, security, and the fundamental technologies and tools needed for designing the eHealth website. The objective is to develop a website that is not only aesthetically pleasing but also intuitive to use and operates well. Additionally, the security of the eHealth website will receive additional attention.

To improve health and healthcare services, e-health, or eHealth, refers to the use of electronic technology including computers, the internet, and mobile devices. This field includes a wide range of tasks and offerings, including gathering and analyzing health data, managing patient files, offering online consultation and treatment services, and keeping track of patients' well-being.

To improve user experience, the research will address the problem of designing eHealth websites. Users are more likely to leave a website if it is hard to navigate. Therefore, it is essential to guarantee that the website is simple to use and understand. Many eHealth websites fail because they are unable to provide user-friendly navigation.

I. THEORY

1 APPLICATIONS AND TECHNOLOGY IN HEALTH CARE INDUSTRY

This Section's main goal is to thoroughly analyze the numerous applications and technology used in the healthcare industry. It seeks to pinpoint and examine the common problems that websites in this field run into, as well as the accompanying restrictions. The assessment will also focus on the features and functionalities of healthcare websites' designs, with a particular focus on those areas.

The use of information and communication technology in the healthcare industry is referred to as "eHealth". It includes a wide range of technology and applications, such as health information exchange (HIE)", telemedicine,"electronic health records (EHRs)", and "mobile health (mHealth)". These technologies could be combined to improve patient care, boost health outcomes, and cut costs for the healthcare system[1].

1.1 Electronic Health Record (EHR)

"Electronic Health Records (EHRs)" serve as digital alternatives to traditional paper-based medical records, offering digitized archives that contain essential patient data, including medication history, past medical conditions, diagnostic imaging data (such as X-rays), and laboratory test results. "(EHRs)" play a pivotal role in supporting various healthcare operations and facilitating efficient storage and retrieval of medical information. They enable healthcare professionals to make evidence-based decisions, manage the quality of care, and facilitate outcome reporting. The utilization of "(EHRs)" in healthcare settings provides numerous benefits, including enhanced data accessibility, improved care coordination, and the potential for more efficient and effective healthcare delivery[2].

1.2 Telemedicine

A method of delivering healthcare called telemedicine allows for the provision of medical services remotely, overcoming geographic distances with the aid of telecommunications technologies. It includes a variety of healthcare tasks like consultations, remote monitoring, and the use of email and mobile health apps. Patients who are unable to physically visit their healthcare provider due to non-emergency circumstances or geographic limitations can use telemedicine as a substitute, especially when the patient and the healthcare provider are in different cities[3].

1.3 Mobile health(mHealth)

The use of mobile phones, tablets, and other wireless devices to access and share health-related information is known as "mobile health (mHealth)". It includes a wide variety of applications, such as diet, fitness, and health apps, which offer helpful tools for managing weight, planning diets, and engaging in online workouts. Additionally, mHealth has shown to be particularly useful in pandemics since it permits remote consultations with medical professionals and makes it easier to disseminate public health updates via mobile applications[4].

1.4 Electronic Health Information Exchange (HIE)

Involves the safe electronic communication of patient data between medical staff members such as nurses, doctors, and pharmacists. This procedure successfully reduces medical errors and gets rid of data duplication. The goal of "HIE" is to improve coordination and continuity of care by the sharing of vital patient data, such as medical history, prescribed medications, and test results[5].

1.5 Evaluation of Functionalities In Current eHealth Demo Applications

In this section, The focus lies on existing eHealth demo websites, highlighting their functionality as well as identifying their associated disadvantages. These demo websites serve as platforms designed to showcase and simulate various aspects of electronic health (eHealth) systems, providing users with a glimpse into the potential functionalities and features of such systems. However, it is important to note that these demo websites may not fully represent the comprehensive capabilities and complexities of real-world.

1.5.1 DocTry

Is an eHealth demo website that encompasses a range of functionalities aimed at showcasing the potential of the platform. It offers a comprehensive set of features and tools designed to simulate real-world healthcare scenarios. Users can interact with the website to experience various aspects of eHealth systems and explore their capabilities. The purpose of DocTry is to provide an educational and informative environment where individuals can familiarize themselves with eHealth technologies and their applications in a controlled and simulated setting. By offering a diverse range of functionalities, DocTry aims to demonstrate the

potential benefits and practical applications of eHealth solutions in improving healthcare delivery and patient outcomes.

1.5.1.1 Doctry Functionalities

1. **Patient Registration:** Users are provided with the opportunity to register on the platform by completing the requisite input fields necessary for the login process.
2. **Doctor Search:** Users can search for doctors, enabling them to explore comprehensive profiles of healthcare professionals. Users have the option to search for doctors based on diverse criteria, including medical specialty, geographical location, and availability, users can rate the doctors and book appointments with them.
3. **Appointment Scheduling:** Users can schedule an appointment with doctors in a convenient and organized manner. This functionality empowers users to secure dedicated time slots for their medical consultations, promoting efficient healthcare delivery and patient satisfaction.
4. **Mobile-friendly interface:** The website is mobile-friendly interface, which is thoughtfully designed to adapt and provide optimal user experience across a range of devices, including smartphones and tablets.

1.5.1.2 Disadvantages Of Doctry

1. **Patient Registration:** The website exhibits suboptimal performance during the user registration process, characterized by slow loading times and a lack of seamless page transitions.
2. **Doctor Search:** The doctry demo website presents limitations in its doctor search functionality, whereby after obtaining a list of doctors, users encounter an issue when attempting to schedule appointments. The system prompts users to log in as the respective user, but even after successful login, the "Book Now" feature fails to function as intended, impeding the appointment scheduling process.
3. **User Interface Design:** The doctry demo website exhibits deficiencies in its user interface design, resulting in a suboptimal user experience. The lack of clarity and confusion in the website's design can potentially lead users to become disoriented and have difficulty navigating through its various components.

4. **Button Functionality:** The doctry demo website exhibits a significant issue where a substantial portion of the buttons fail to function as intended. Users encounter limitations and frustrations when attempting to interact with these non-responsive buttons, resulting in an impaired user experience.

1.5.2 Emantal Hospital System

The emantal hospital system demo website serves as a simulated platform that showcases the functionalities and features of a hospital system. It provides users with a controlled and simulated environment where they can explore and interact with the various components and capabilities of the system.

1.5.2.1 Functionalities Of Emantal Hospital System

1. **User registration:** The emantal hospital demo website alllows users to create an account on the emantal hospital system demo website by providing their personal information. This feature typically includes fields for entering details such as name, email address, password, contact information, and any other required information.
2. **Doctors Profile:** The emantal hospital system demo website offers comprehensive doctor profiles, allowing users to access detailed information about healthcare providers. These profiles serve as a valuable resource for users to explore and gain insights into the backgrounds, qualifications, and areas of expertise of the doctors. Users can access information such as the doctor's name, specialization, qualifications, professional experience, and affiliations.
3. **Services:** The emantal hospital system demo website offers a diverse range of services to cater to the needs of patients, doctors, and the overall management of the hospital. These services encompass various aspects of hospital operations and include features such as human resources management, department management, bed management, and patient management.

1.5.2.2 Disadvantages Of Emantal Hospital System

1. **User Registration:** In the emantal hospital system, during the evaluation of the registration process, it was observed that the system encounters an error or becomes unresponsive when users attempt to register by inputting their information into the registration form. This issue results in the inability of users to successfully complete the registration process, causing inconvenience and hindering their access to the system's functionalities.
2. **Doctors Profile:** In the doctors page of the emantal hospital system, it was noted that certain essential features are missing. Specifically, there is a lack of a "Book Appointment" button or any relevant information pertaining to the doctors available on the platform. As a result, users may face difficulties in scheduling appointments with doctors or obtaining necessary details about their qualifications and specialties. This limitation can hinder the user experience and may lead to frustration and inconvenience for individuals seeking medical services through the website.
3. **User-Interface Design:** The user-interface design of the emantal hospital system demonstrates several weaknesses. The layout and organization of elements appear to be inconsistent and disjointed, making it challenging for users to navigate the website seamlessly. The lack of a cohesive visual hierarchy and proper use of color, typography, and spacing further contribute to the suboptimal user experience. Additionally, the placement and visibility of important features and functionalities are not prioritized effectively, resulting in user confusion and frustration. Enhancements to the user-interface design are necessary to improve usability, clarity, and overall user satisfaction.

2 HEALTH SOLUTIONS AND THEIR LIMITATIONS

To improve healthcare services and patient care, a variety of eHealth solutions have been put into place. These methods include telemedicine, "mobile health (mHealth)", "electronic health records (EHR)", and "health information exchange (HIE)". Although these technologies have greatly improved healthcare, they do have certain drawbacks.

2.1 EHR LIMITATIONS

The significant financial consequences of "Electronic Health Records (EHRs)" are one of the main limitations connected with them. Incomplete patient information, repetitive duties, and a higher chance of medical errors are all potential outcomes of the existence of many systems and processes used to store patients' data[5]. [6] "EHRs" also frequently display usability issues, with user-friendliness issues affecting healthcare workers' interactions and system navigation[5]. The uneven adoption of "EHRs" across healthcare providers, which impedes the seamless sharing of patient information between various healthcare institutions, exacerbates these restrictions[5].

2.1.1 TELEMEDICINE LIMITATIONS

The different insurance company coverage guidelines that can make telemedicine less accessible for some patients are an inherent constraint of the practice[7]. Telemedicine can also present obstacles in the path of delivering timely healthcare. The availability of telemedicine services may encounter difficulties, particularly during emergency situations, due to limitations such as insufficient internet bandwidth and other technical constraints. These challenges can impede the prompt provision of healthcare services, thereby potentially affecting patient outcomes and the overall effectiveness of care delivery[7].

2.1.2 MHealth LIMITATIONS

Concerns about the veracity of the information provided are raised by the growth of unreliable mobile health (mHealth) applications, potentially putting users in danger[8].

2.1.3 HIE LIMITATIONS

As patients must be fully informed and provide their agreement before their sensitive information is shared, preserving privacy and confidentiality, the security component of Health Information Exchange (HIE) has constraints[9].

3 BRIEFLY DESCRIBE THE TECHNOLOGIES BEING USED

This section gives a general overview of the technologies used in the creation of the website, discusses their benefits and drawbacks, and explains why they were chosen.

3.1 HTML

Hypertext Markup Language, or HTML, is the most common markup language used to create online pages. It includes several benefits that support its broad use and significance in the field of web development[10].

3.1.1 Advantages Of HTML

1. **Ease of Learning and Use:** HTML has a user-friendly language structure that makes it approachable to beginners and makes it easier for them to quickly comprehend its syntax than with more complex programming languages[11].
2. **Universal Compatibility:** HTML has full support in all web browsers, ensuring fluid rendering and accessibility on a variety of devices with internet connectivity[11].
3. **Cost-Effectiveness:** HTML requires no financial commitment because it can be created for free using simple text editors like Notepad or TextEdit[11].
4. **Interoperability:** HTML works in harmony with CSS and JavaScript, complementing programming languages, to produce complex, dynamic, and interactive web content[11].
5. **Versatility:** HTML offers a great deal of versatility by enabling the direct integration of different media formats, including photos, videos, and audio files, into web pages[11].
6. **Compatibility with search engine optimization (SEO):** HTML demonstrates inherently SEO-friendly characteristics since the proper application of HTML components improves website exposure and ranking in search engine results[11].
7. **Maintainability:** By allowing direct updates to the HTML codebase and providing an instantaneous reflection of changes upon page loading, HTML facilitates the maintenance and updating of web pages[11].
8. **Accessibility:** HTML incorporates characteristics that encourage the accessibility of web content, accommodating a range of users, including those who use assistive technology[11].

9. Online games, video material, and other interactive and dynamic content may all be created using HTML5, the newest version of HTML, without the need for additional plugins[11].

In conclusion, HTML is a fundamental building block of web development, including a variety of benefits for both users and developers, cementing its crucial position within the digital environment.

3.1.2 DISADVANTAGES OF HTML

1. HTML is a primarily static language, therefore adding dynamic features necessitates the use of a backend programming language like PHP, JavaScript, or Python[12].
2. When it comes to webpage design, HTML alone is insufficient; CSS must be used in conjunction with HTML to obtain the required visual appeal and layout. However, it's crucial to recognize that HTML has some security restrictions[13].

3.2 CSS

Cascading Style Sheets, or CSS, is a styling language that works with HTML, XML, and other markup languages. Its main purpose is to give websites a polished and uniform look by exercising control over many visual features, such as color schemes, background components, layout designs, and a variety of other design elements. Websites can appear comprehensive and visually appealing with the help of CSS[14].

3.2.1 ADVANTAGES OF CSS

1. CSS displays versatility by offering a wide range of design options, including responsive layouts that automatically adjust to various screen sizes and devices[15].
2. The concision of its code further demonstrates its inherent efficiency, which leads to increased speed and ideal performance[15].
3. Notably, the separation of CSS and HTML code allows for quicker design element updates, enabling developers to quickly make specific design changes[15].

3.2.2 DISADVANTAGES OF CSS

- When using CSS, cross-browser compatibility is essential since the same code that works flawlessly on one browser could not provide the expected results on another.

To maintain consistent performance and appearance across many platforms, web developers must carefully test their programs across multiple browsers [16]

- The availability of security is limited[16].
- Once the changes have been made, it is crucial to confirm their compatibility with different browsers. All supported web browsers should be equally impacted by the changes[16].

3.3 JAVASCRIPT

A dynamic scripting language used in the client-side browser environment is called JavaScript. It enables developers to include interactive features in their websites by smoothly integrating with CSS and HTML. These features include several interactive components, such as animated visuals, pop-up windows, and drop-down menus. JavaScript gives programmers the ability to create dynamic, responsive online content that will captivate website users and improve user experiences[17]

3.3.1 ADVANTAGES OF JAVASCRIPT

- JavaScript is used in a variety of industries, including the creation of desktop software, mobile apps, and websites [18].
- It does not require continual server access because it runs client-side[18].
- There are numerous educational resources available for JavaScript, which is extensively used, to assist newbies in becoming adept with the language[18].

3.3.2 DISADVANTAGES OF JAVASCRIPT

- Because JavaScript runs on the client side, it is vulnerable to unwanted access by other users. Furthermore, the language lacks full debugging features[19].
- Furthermore, like CSS and HTML, JavaScript may exhibit differences in functionality across different web browsers[19].

3.4 BOOTSTRAP 5

A front-end web development framework made up of HTML, CSS, and JavaScript, Bootstrap is publicly available and open source. It simplifies the development of mobile-first and responsive web applications[20].

3.4.1 ADVANTAGES OF BOOTSTRAP

1. Given the prevalence of mobile device access to digital information, Bootstrap is an excellent choice for designing responsive websites. Its major goal is to guarantee that web pages are designed to adapt and perform well across a wide range of screen sizes and devices, including mobile phones. Using Bootstrap, web developers can construct user-friendly and visually appealing websites that provide a seamless surfing experience on mobile devices[21].
2. For developers to use in their projects, Bootstrap provides a wide range of pre-designed components, including dropdowns, navigation bars, forms, and progress bars [21].
3. This eliminates the need for developers to design these components from scratch, allowing them to save time and effort during the development process. Bootstrap is built with widely used front-end technologies including JavaScript and CSS, which are widely used in the design of user interfaces and web applications[21].
4. The use of these languages provides interoperability with existing front-end frameworks and reinforces Bootstrap's efficacy in constructing responsive and dynamic websites[21].

3.4.2 DISADVANTAGES OF BOOTSTRAP

1. Bootstrap provides a wealth of freely available templates, allowing developers to quickly design websites without requiring in-depth understanding of website creation [21].
2. By employing Bootstrap's comprehensive framework, developers gain access to a wide range of tools and resources required for website building, which can create hurdles for users with less coding experience[21].
3. Furthermore, Bootstrap templates have a constant appearance, making it difficult for developers to achieve diverse designs even with adjustments[21].

3.5 UML(UNIFIED MODELING LANGUAGE)

The Unified Modeling Language (UML) is a graphical notation used in system design to help software engineers, business analysts, and system architects depict the functionality of their systems. UML is a collection of diagrams that each serve a specific role in system

modeling. The functional requirements model, use case models, class models, sequence diagram models, activity diagrams, and wireframe diagrams are some examples of these diagrams. Each UML diagram depicts a specific component of how the website will interact and function[22]

3.5.1 Functional requirements

Functional requirements encompass two fundamental aspects: function and behavior. The function of a system refers to its intended operation, such as executing specific tasks or performing certain actions. On the other hand, the behavior of a system pertains to the manner in which it operates, including the detailed specifications and actions involved in carrying out its functions[23]

Example:

1. Function: Register
2. Behaviour: The system shall allow the user to register by entering their name and password.

3.5.2 Non-functional requirements

Also known as quality characteristics, encompass the overall properties and attributes of a system. These requirements go beyond the system's specific functionalities and focus on its general qualities and characteristics[24].

3.5.3 Use Case Diagram

Represents the interactions between actors and a system graphically. It depicts how multiple actors, such as users or other systems, interact with the system under consideration. Use case diagrams are employed to describe the functional requirements and intended behavior of the system in terms of the actions of the actors and the system's responses. Use case diagrams enhance understanding of a system's operation and interactions with other entities by graphically representing these interactions[25].

3.5.4 Class Model

is a fundamental component of a system that outlines its structure by describing the static aspects of its objects. It provides a comprehensive representation of each class's objects, including their attributes and operations, and establishes the relationships between these

objects within the system. In essence, the class model serves to define the blueprint for the system's objects and their interactions, facilitating a clear understanding of the system's structure and behavior[26].

3.5.5 Sequence Diagram

Is a specific type of interaction diagram that illustrates the flow of messages exchanged between objects within a system. It provides a visual representation of the chronological order in which these messages are sent and received, depicting the interactions and communication among objects over time. By showcasing the dynamic behavior of the system, the sequence diagram allows for a clear understanding of the sequence of events and the collaboration between objects during the execution of a particular scenario or functionality[27]

3.5.6 Wireframe

Is a simplified representation or sketch of the system that is to be developed. It is designed to be clear, easy to understand, and provides just enough information to convey the basic structure and layout of a screen or interface. Rather than focusing on intricate details, a wireframe serves as a blueprint for the final screen design that will be created later in the development process. By presenting the wireframe to the customer, it allows for visual communication of the system's scenario and facilitates obtaining their approval and feedback on the requirements[28]

3.6 Figma

Figma is a robust design tool that empowers users to create diverse visual assets, ranging from websites and applications to logos[29].

II. ANALYSIS

4 DESIGN OF THE EHEALTH APPLICATION

In the early stages of my project, UML diagrams were created to better understand the complexities of proposed website's behavior and responsiveness. This project included the development of UML diagram types, including functional requirements, non-functional requirements, use case diagrams, class models, and sequence diagrams. The main features relevant to the website's operation and structural composition using these UML diagrams.

4.1 Functional Requirements

In Figure 1, 2, and 3, functional requirements for the healthcare system are shown.

Figure 1 and 2 show functional requirements which are associated with doctors.

Functional Requirements	Description
FRQ1:Doctors Management	The system should allow new doctors to register, providing necessary details such as name, specialization, and contact information.
FRQ2:Doctor Login	Registered doctors should be able to log in to the system using their unique credentials
FRQ3: Profile Management:	Doctors should be able to update their profiles, including their qualifications, specializations, consultation hours, and contact information.
FRQ4:Patient Record Access:	Doctors should be able to access patient records to review medical histories, previous diagnoses, test results, and other relevant information.
FRQ5:Appointment Scheduling	Doctors should be able to view, accept, or reschedule appointments booked by patients.

Figure 1: Doctor's Management Functional Requirements 1/2

Functional Requirements	Description
FRQ6:Prescription Management	Doctors should be able to create, update, and view prescriptions for their patients.
FRQ7:Consultation Notes	The system should enable doctors to write and save consultation notes during or after an appointment.
FRQ8:Availability Status	Doctors should be able to set their availability status (available, busy, on leave, etc.) to manage patient expectation.
FRQ9:Notification Center	The system should allow users to see the notifications they have.

Figure 2: Doctor's Management Functional Requirements 2/2

Functional Requirements	Description
FRQ1:Patient Registration	The system allow patients to register providing, the information of the patients such as name, email, phone number, age and gender
FRQ2:Patient Login	The system should allow patient to allow into their account by providing, email and password
FRQ3:Appointment Scheduling	Patient should be able to schedule, view, update and cancel appointment
FRQ4:Patient Record Access	The system should allow patients to access and view their medical record
FRQ5:Chat Management	The system should allow patients to send messages to their health providers

Figure 3: Patients Functional Requirements

4.2 Non-Functional Requirements:

In this section there is five non-functional requirements for the eHealth system, which encompass crucial aspects of Usability, Accessibility, Performance, Robustness and Availability.

Non-Functional Requirements	Description
NFRQ1:Usability	The website should exhibit a user-centric design approach, emphasizing usability and intuitive navigation, to ensure an optimal user experience
NFRQ2:Accessibility	The website should be online accessible, ensuring continuous availability via web browsers when connected to the internet
NFRQ3:Performance	The website should exhibit optimal responsiveness characterized by swift and efficient processing of user requests and delivering prompt responses
NFRQ4:Robustness	The website should demonstrate robust scalability, allowing it to effectively accommodate varying user loads and maintain stable performance without experiencing system failures or crashes
NFRQ5:Availability	The website should exhibit uninterrupted availability, ensuring continuous accessibility for users

Figure 4: Non-Functional Requirement

4.3 USE CASE DIAGRAMS

In Figure 5, the use case diagram for the proposed eHealth system is shown. Three actors exist in the system: Patient, Doctor, and User. User is a general actor which is used just for simplification.

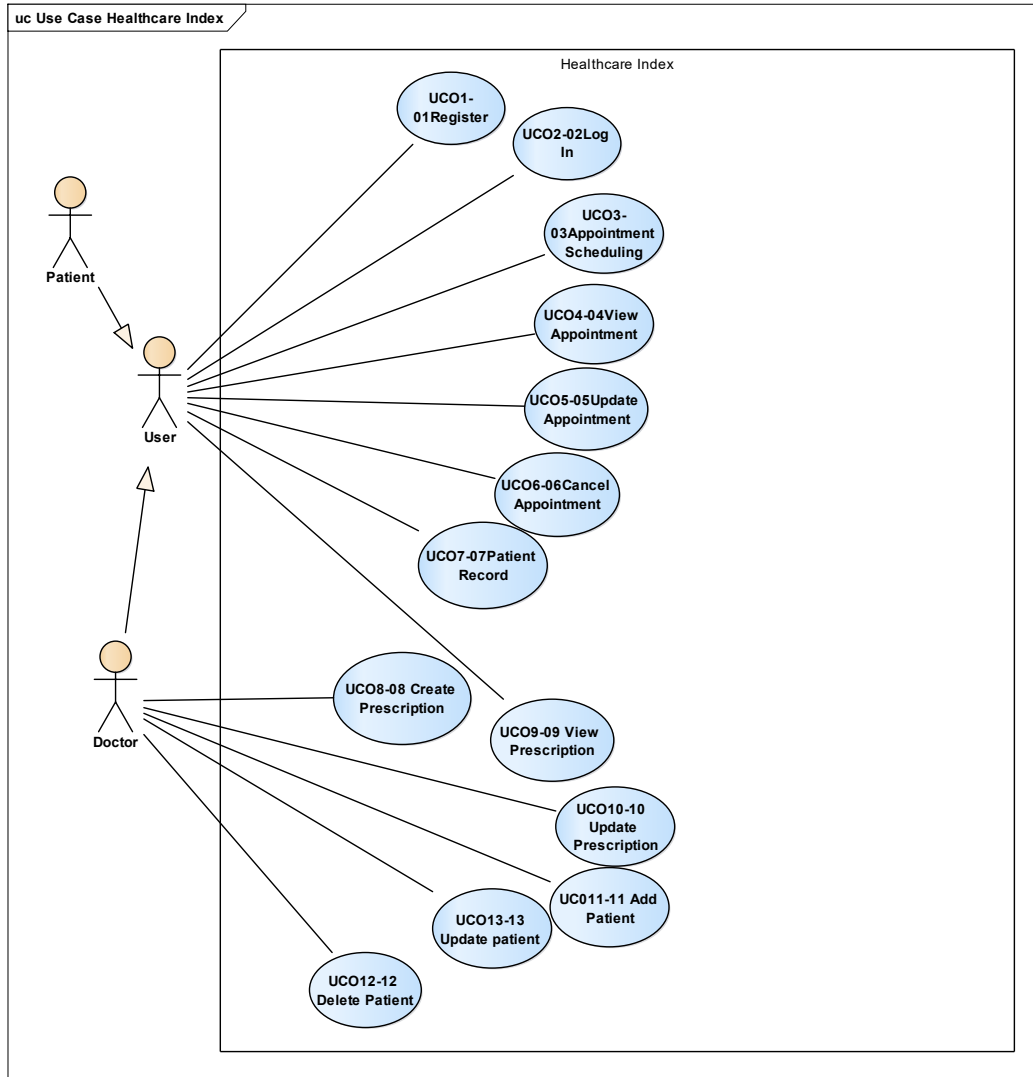


Figure 5: Use Case Diagram

4.3.1 UCO1-01 Register

Scenario: User registers into the system.

Flow:

1. The user clicks on the login button.
2. The user locates and clicks on the "Create Account" button.

3. The system presents the registration form to the user.
4. The user enters their personal information, including name, email, password, phone number, age, and gender.
5. The user submits the registration form.

Alternative Scenario: User encounters registration errors.

The alternative scenario starts in step 4

1. After the user submits the registration form, the system validates the entered information.
2. If any required fields are left blank or contain invalid data, the system displays an error message indicating the specific errors.
3. The user reviews the error message and corrects the invalid or missing information.
4. The user resubmits the registration form.
5. If the system detects any further errors, the process returns to step 2, prompting the user to make necessary corrections.
6. Once all entered information is valid, the system successfully processes the registration request and creates a new user account.

Post-Condition: The system registers the user successfully after resolving any registration errors, allowing the user to log in and access the system's features and functionalities.

4.3.2 UCO2-02 Log in

Scenario: User Logs in to the Website

Pre-Condition: The user must have a registered account to log in.

The Flow:

1. The user clicks on the login button.
2. The system presents the login form to the user.
3. The user enters their login credentials, including the email and password associated with their account.
4. The user submits the login form.

5. The system verifies the entered email and password combination.
6. If the authentication is successful, the system grants access to the user by logging them into the website.
7. The user gains access to the website's features and functionalities.

Alternative Scenario: User Encounters Login Authentication Error

The alternative scenario starts at step 3

1. After the user submits the login form, the system verifies the entered email and password combination.
2. If the entered credentials are incorrect or do not match any existing user accounts, the system displays an error message indicating the authentication failure.
3. The user reviews the error message and rechecks the entered email and password.
4. The user makes the necessary corrections to the email and/or password.
5. The user resubmits the login form.
6. If the system still detects authentication errors, the process returns to step 2, prompting the user to reenter the correct email and password.
7. Once the correct credentials are entered, the system successfully authenticates the user and allows access to the website's features and functionalities.

Post-Condition: The user is logged in successfully after resolving any login authentication errors, granting access to the website's resources and functionalities.

4.3.3 UCO3-03 Schedule an appointment

Use Case Scenario: User Schedules an Appointment

Description: The user, who does not have a scheduled appointment, initiates the process of scheduling a new appointment.

Preconditions:

- The user is logged into their account.
- The user does not have any existing appointments scheduled.

Main Flow:

1. The user accesses the application or website and logs into their account.
2. The user navigates to the appointment scheduling section.
3. The user locates and clicks on the "Schedule Appointment" button.
4. The system presents a form or interface for scheduling an appointment.
5. The user selects the desired doctor from a list of available doctors.
6. The user enters the preferred date and time for the appointment into the appropriate fields.
7. The user submits the appointment scheduling form.
8. The user's appointment is now scheduled and can be viewed in their appointment list.

Post-Conditions: The user successfully schedules a new appointment, and the appointment is reflected in their appointment list within the system.

4.3.4 UCO4-04 View Appointment

Use Case Scenario: User Views Appointment

Description: The user wants to view the details of a previously scheduled appointment.

Precondition:

- The user is logged into their account.
- The user has at least one appointment scheduled.

Main Flow:

1. The user accesses the application or website and logs into their account.
2. The user navigates to the appointment section or dashboard.
3. The user locates and selects the specific appointment they want to view.
4. The system retrieves and displays the details of the selected appointment.
5. The appointment details include information such as the date, time, doctor, and any additional notes or instructions associated with the appointment.
6. The user reviews the displayed appointment details.
7. If desired, the user can take note of any important information or make necessary preparations based on the appointment details.

Postconditions: The user has successfully viewed the details of the selected appointment. They can refer to the displayed information for reference or take any necessary actions based on the appointment details.

4.3.5 UCO5-05 Update Appointment

Scenario: The user wishes to change the specifics of their scheduled appointment.

Pre-Condition: The user is unable to attend the booked appointment and wants to update due to an emergency.

1. The user logs into their account.
2. The user navigates through the appointment schedule section.
3. The system will provide the user with the appointment that have been scheduled.
4. The user clicks on the appointment they wish to update.
5. The system will provide the user with the date and time that needs to be updated.
6. The user updates the date and time.
7. The system will update the appointment.

Post-Condition: The appointment successfully updated.

4.3.6 UCO6-06 Cancel appointment.

Use Case Scenario: User Cancels Scheduled Appointment

Description: The user needs to cancel their previously scheduled appointment due to an emergency or other unforeseen circumstances.

Precondition:

- The user is logged into their account.
- The user has a previously scheduled appointment.

Main Flow:

1. The user accesses the application or website and logs into their account.
2. The user navigates to the appointment schedule section.
3. The system retrieves and displays the list of appointments that have been scheduled by the user.

4. The user selects the specific appointment they wish to cancel.
5. The system presents the details of the selected appointment, including the date and time.
6. The user confirms their decision to cancel the appointment.
7. The system processes the cancellation request and removes the appointment from the schedule.
8. The system confirms the successful cancellation and displays a confirmation message to the user.

Postcondition:

- The appointment is successfully cancelled, and it is removed from the user's scheduled appointments. The user is no longer scheduled for the cancelled appointment.

4.3.7 UCO7-07: View patient record

Scenario: the user wants to view the patient record

Pre-Condition: the user needs to access and view patient records.

The Flow:

1. The user logs into their account
2. The user will click to view the specific patient.
3. The user will be able to see the basic information of the patient.
4. The user will click on the date that the patient had before.
5. The system will provide the medical record of the patient.
6. The user is able to view the patient's medical record such as, medical history, diagnoses and test results.

Post-Condition: the user is now able to view patient records.

4.3.8 UCO8-08: Create Prescription

Scenario: The user wants to prescribe medication for patient

Pre-Condition: The user is logged in into their account.

The Flow:

1. The user selects the patient that wants to prescribe a medication for

2. The system will show the medical record of the patient.
3. The user will click on the consultation button.
4. The user will specify the usage and dosage, frequency, and duration.
5. The system updates the patient's medical record.

Post-Condition: the prescription details are recorded in patients' medical record.

4.3.9 UCO9-09: View Prescription

Scenario: The user wants to view the prescription

Pre-Condition: The user is logged in into their account.

The Flow:

1. 1. The user logs into their account.
2. The user navigates to the patient's medical record or prescription section.
3. The system retrieves and displays the patient's medical record or prescription information.
4. The user selects the specific prescription they want to view.
5. The system presents detailed information of the selected prescription, including the prescribed medication, dosage, frequency, and duration.
6. The user reviews the prescription details.
8. The user exits the prescription view section.

Post-Condition: The user has successfully viewed the prescription details for the selected patient. They can review the prescribed medication and related information as needed.

4.3.10 UCO10-10: Update Prescription

Scenario: The user wants to update a prescription for a patient.

Pre-Condition: The user is logged into their account and has access to the patient's medical records.

The Flow:

1. The user selects the patient for whom they want to update the prescription.
2. The system retrieves and displays the patient's medical record or prescription section.

3. The user locates the specific prescription they wish to update.
4. The user reviews the existing prescription details, including medication, dosage, frequency, and duration.
5. The user identifies the changes they want to make to the prescription.
6. The user updates the prescription information, modifying the medication, dosage, frequency, or duration as required.
7. The system validates the updated prescription details for accuracy and completeness.
8. The system saves the updated prescription in the patient's medical record.
9. The system notifies the user that the prescription has been successfully updated.
10. The user reviews the updated prescription information to ensure it reflects the desired changes.

Post-Condition: The prescription for the patient has been successfully updated with the revised medication, dosage, frequency, or duration. The updated prescription details are now recorded in the patient's medical record.

4.3.11 UCO11-11 Add Patient

Scenario: The user wants to add new patient into the website.

Pre-Condition: The user is logged into their account and has the necessary permissions to add new patients.

The Flow:

1. The user navigates to the patient management section of the website.
2. The user clicks on the add a new patient button.
3. The system presents a form for the user to enter the patient's information.
4. The user fills in the required fields, including patient (1. Name 2. Phone Number 3. Age 4. Gender), and any relevant medical details.
5. The user optionally adds additional information or notes about the patient.
6. The user verifies the entered information for accuracy.
7. The user submits the form to add the new patient.

Post-Condition: The new patient's information is successfully added to the website, allowing the user to manage and access their details for future interactions and appointments.

4.3.12 UCO12-12 Delete Patient

Scenario: The User wants to delete a patient from the patients list.

Pre-Condition: The user is logged into their account and has the necessary permissions to delete patients.

The Flow:

1. The user navigates to the patient management section of the website.
2. The user accesses the list of patients.
3. The user selects the patient they want to delete.
4. The User will click on the delete button.
5. The system removes the selected patient from the patients list and associated data.
9. The user can no longer find the deleted patient in the patients list or access their profile.

Post-Condition: The selected patient is successfully deleted from the patients list, removing their information from the website and preventing further access to their profile and associated data.

4.3.13 UCO13-13 Update Patient

Scenario: The User wants to update the patient's information.

Pre-Condition: The user is logged into their account and has the necessary permissions to update patient information.

The Flow:

1. The user navigates to the patient management section of the website.
2. The user accesses the list of patients.
3. The user selects the patient whose information needs to be updated.
4. The system presents the patient's current information for review.
5. The user modifies the necessary fields, such as name, contact information, or age and Gender.

Post-Condition: The patient's information is successfully updated in the system, reflecting any changes made by the user.

4.4 Class Model Diagram

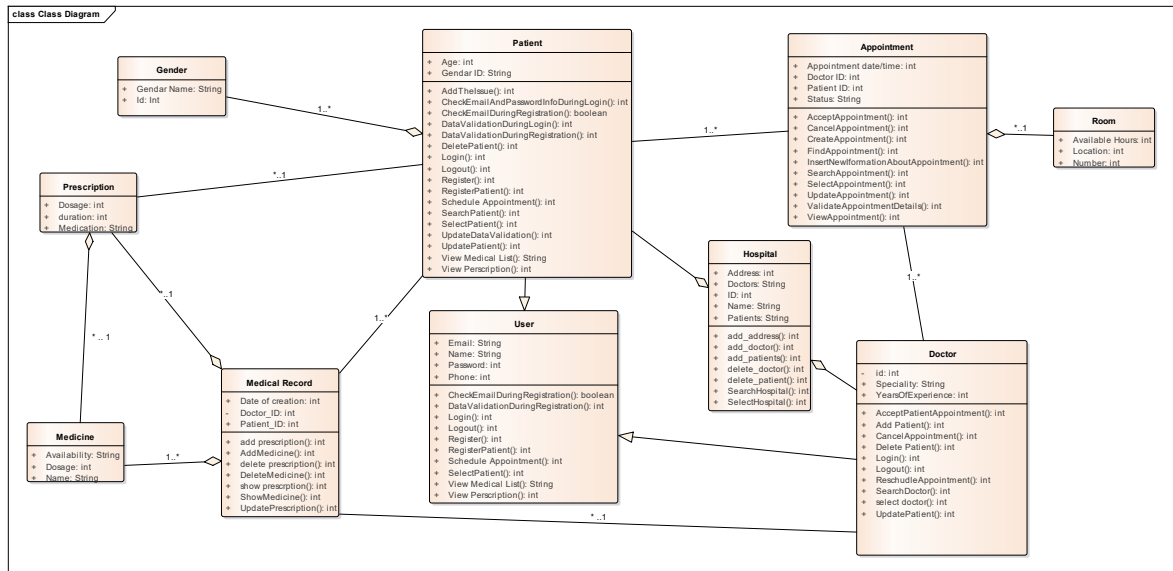


Figure 6: Class Model

In the Figure 6 class model diagrams show the relationship between classes. It Contains attributes, operations, and methods of each class.

In the class diagram, the one-to-one relationship between the Patient class and the Gender class indicates that each patient has many-to-one gender.

The aggregation relationship implies a "has-a" relationship, meaning that a patient has a gender. It represents a whole-part relationship, where the Patient class is the whole and the Gender class is a part of it.

This aggregation allows the Patient class to access and utilize the attributes and behaviors defined in the Gender class, such as retrieving and setting the gender of a patient.

Overall, the aggregation relationship between the Patient and Gender classes in the class diagram indicates that a patient has a gender attribute, which is represented by the aggregation arrow pointing from the Patient class to the Gender class.

The Patient class is shown as being generalized to the User class.

By generalizing the Patient class to the User class, it implies that a patient is a type of user. The User class defines common attributes and behaviors that are shared by all users, while the Patient class adds additional attributes and behaviors specific to patients.

In the class diagram, the Medicine class is shown to have a one-to-many aggregation relationship with the Medical Record class.

The aggregation relationship between the Medicine and Medical Record classes indicates that a medical record can have multiple medicines associated with it. This relationship is represented by a line with a diamond-shaped arrowhead pointing from the Medical Record class to the Medicine class.

The Doctor class is shown to be generalized to the User class.

In this case, the User class is the superclass, and the Doctor class is a subclass of User. This relationship signifies that a doctor is a type of User, inheriting the attributes and behaviors defined in the User class.

4.5 Create Account Sequence Diagram

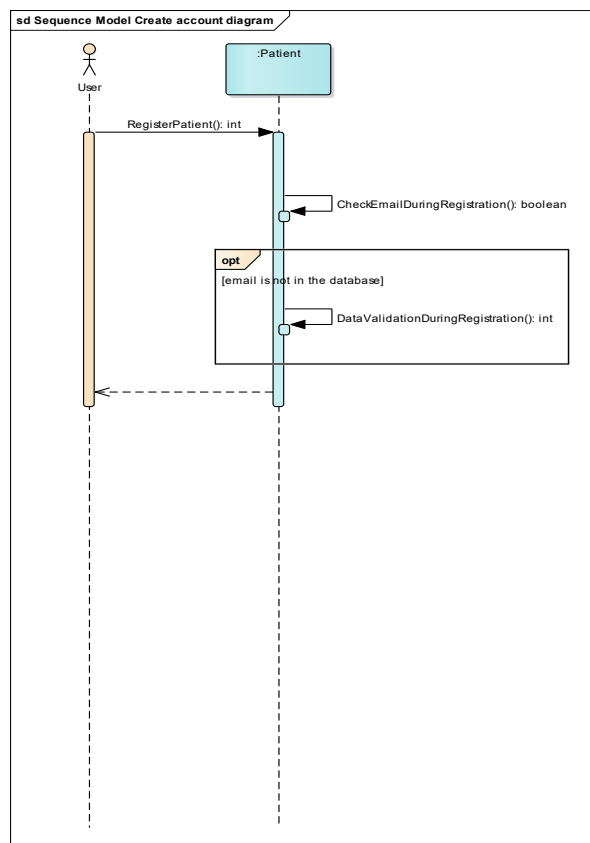


Figure 7: Create Account Sequence Diagram

In Figure 7, the sequence diagram depicts the process of creating an account. It involves one actor, which is the User, and the lifeline representing the Patient class from the class model.

The sequence starts when the User sends a message to the lifeline of the Patient class, specifically the "Register Patient" message. This message indicates the user's intent to register as a patient.

The lifeline representing the Patient class performs certain actions in response to the message. It checks the validity of the email provided during the registration process, ensuring it meets the required format and is not already in use. Additionally, the lifeline performs data validation, verifying that all the required information is provided and meets the specified criteria.

Once the email validation and data validation processes are completed, a message is sent back to the User, indicating the outcome of the registration attempt. This message may contain information such as a success or failure status, an error message in case of invalid data, or a confirmation of successful registration.

The lifeline of the Patient class acts as an intermediary between the User and the registration process, handling the necessary checks and validations. This sequence diagram provides a visual representation of the interactions and flow of messages between the User and the Patient class during the account creation process.

4.5.1 Login Sequence Diagram

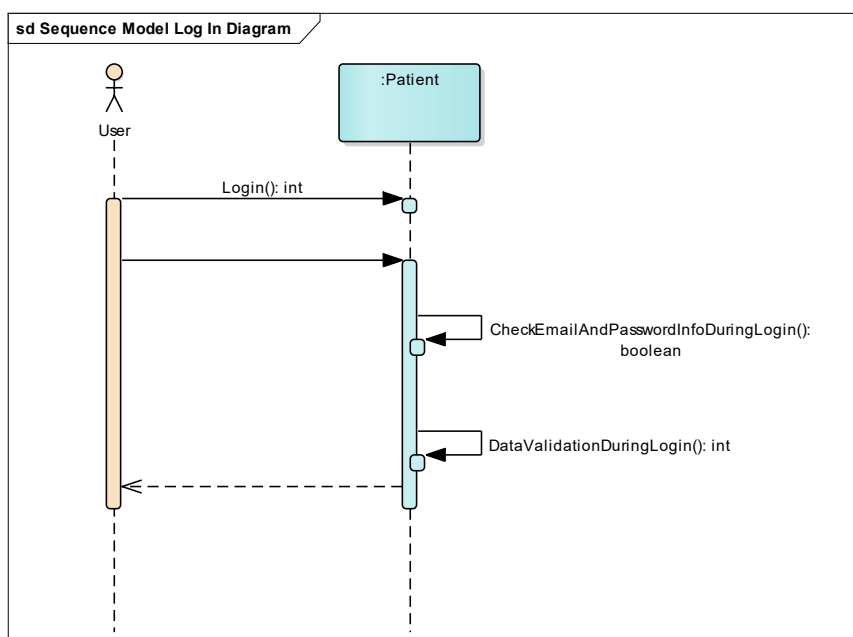


Figure 8: Login Sequence Diagram

In Figure 8, the presented sequence diagram illustrates the login process within the context of the healthcare system. The diagram consists of one actor, which is the user, and one lifeline representing the patient.

The sequence begins with the user initiating the login process by sending a message to the patient lifeline, indicating their intention to log in. The patient lifeline, being responsible for managing patient-related actions, performs a validation check on the entered email and password.

If the email and password provided by the user are correct, the patient lifeline allows the user to proceed with the login process. This means granting access to the system and providing the user with the necessary privileges and functionalities. However, if the email and password are found to be incorrect during the validation check, an error message is generated, indicating that the login credentials are invalid.

4.5.2 Create Appointment Management Diagram

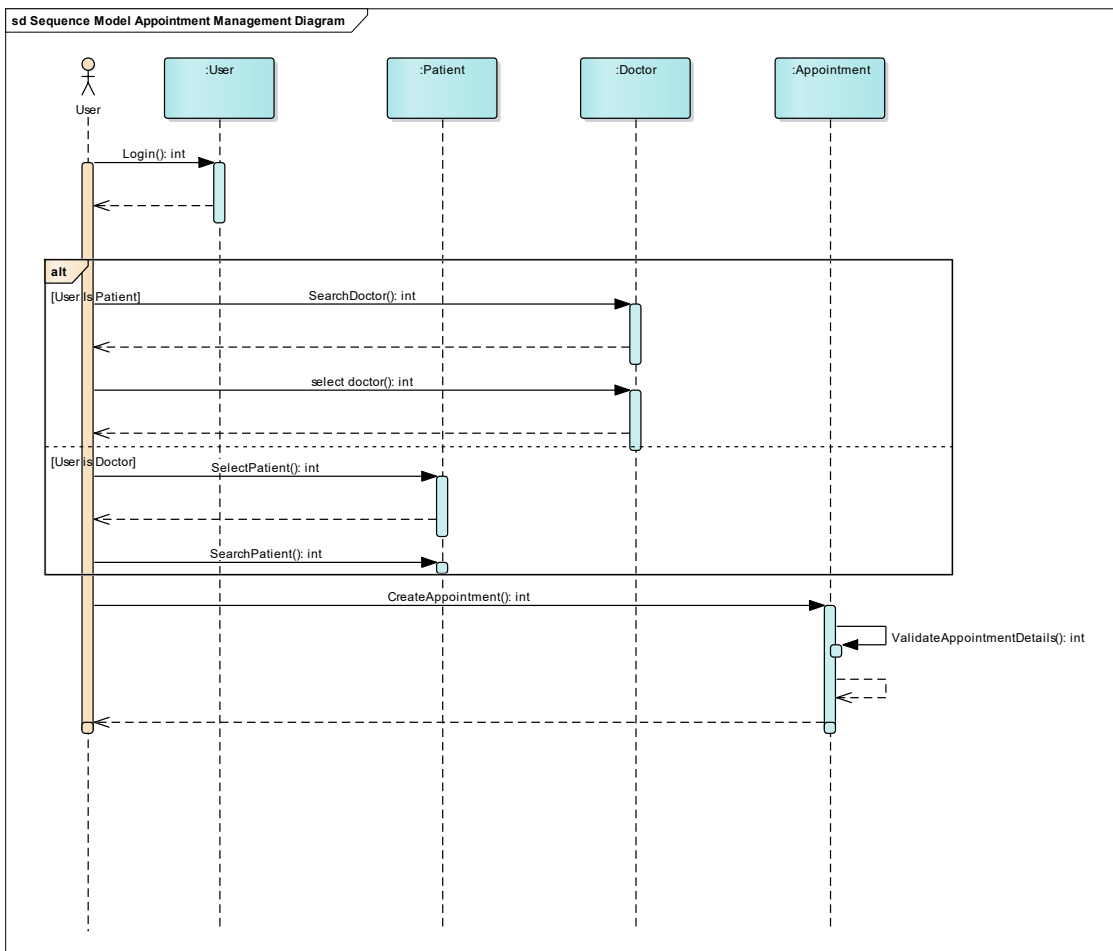


Figure 9: Create Appointment Sequence Diagram

This sequence diagram illustrates the appointment management process. It involves one actor, which is the User, and five lifelines representing different components.

The sequence begins with the User actor sending a message to the lifeline representing the User class in the class model. This message represents the user's intention to log in. The User lifeline processes the login request and sends a message back to the User actor indicating that the login has been successful.

Note that there are two users, in the search doctor and select doctor the user is patient.

In the search patient and select patient the user is doctor.

Next, the User actor sends a message to the Doctor lifeline, representing the doctor component, requesting a search for available doctors. The Doctor lifeline performs the search operation and returns the message to the User actor, providing the list of doctors.

The User actor then sends another message to the Doctor lifeline, specifying the selection of a particular doctor. The Doctor lifeline acknowledges the selection and returns the message to the User actor.

Subsequently, the User actor sends a message to the Patient lifeline, representing the patient component, to select patients. The Patient lifeline acknowledges the request and returns the message to the User actor, providing the list of patients.

The User actor proceeds to send another message to the Patient lifeline, requesting a search for specific patients. The Patient lifeline performs the search operation and returns the message to the User actor, providing the search results.

Finally, the User actor sends a message to the Appointment lifeline, representing the appointment component, with the intention of creating a new appointment. The Appointment lifeline validates the appointment details and returns the message to the Appointments lifeline. Finally, the Appointment lifeline sends the message back to the User actor, completing the sequence of interactions.

This sequence diagram outlines the series of message exchanges between the User actor and various lifelines representing different components involved in the appointment management process. It illustrates the steps of selecting a hospital, doctor, patient, and creating an appointment, with the appropriate lifelines validating and returning messages at each stage.

4.5.3 View Appointment Sequence Diagram

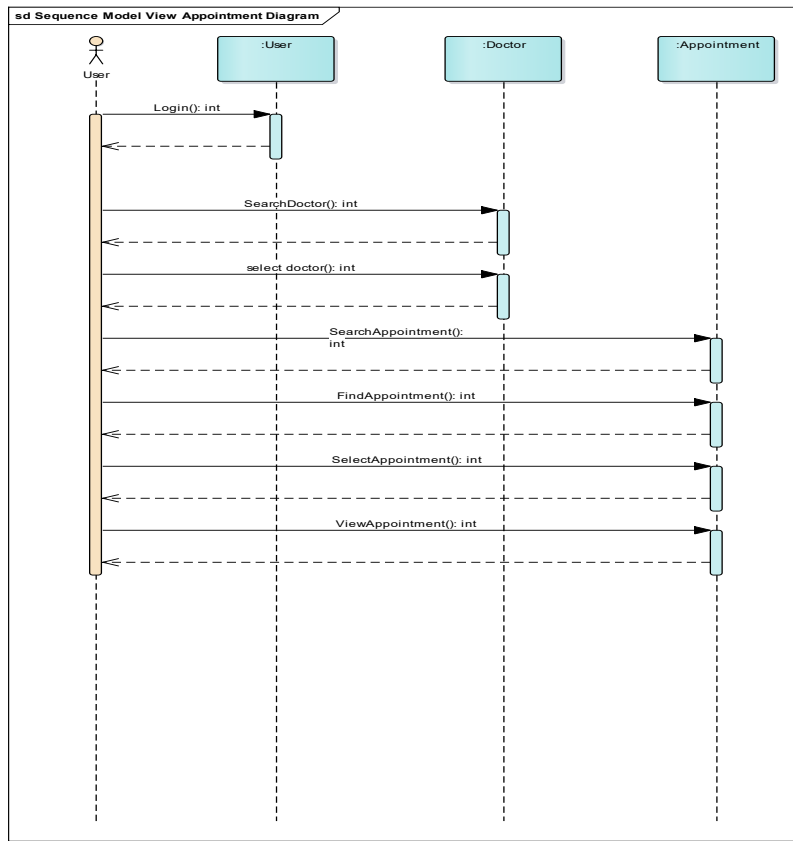


Figure 10: View Appointment Sequence Diagram

In the In Figure 10, the sequence diagram depicts the process of viewing an appointment in the system. The diagram involves one actor, which is the user, and several lifelines representing different system components.

The sequence begins with the user sending a login request to the user lifeline. Upon processing the request, the user lifeline responds with a message confirming the successful login. Subsequently, the user interacts with the doctor lifeline, initiating a search for doctors. The doctor lifeline processes the request and returns a message to the user, presenting the search results for doctors.

Following this, the user selects a specific doctor by sending a message to the doctor lifeline. The doctor lifeline acknowledges the selection and responds with a confirmation message. The user then proceeds to interact with the appointment lifeline, requesting to search for appointments. The appointment lifeline processes the request and provides the user with a return message.

Next, the user specifies the desired appointment and sends a corresponding message to the appointment lifeline. The appointment lifeline processes the request and returns a message to the user, supplying the details of the requested appointment. Finally, the user sends a message to the appointment lifeline, requesting to view the appointment. The appointment lifeline processes the request and responds with a message, displaying the relevant appointment details.

This sequence diagram illustrates the step-by-step interactions between the user and the different system components involved in viewing an appointment.

4.5.4 Update Appointment Sequence Diagram

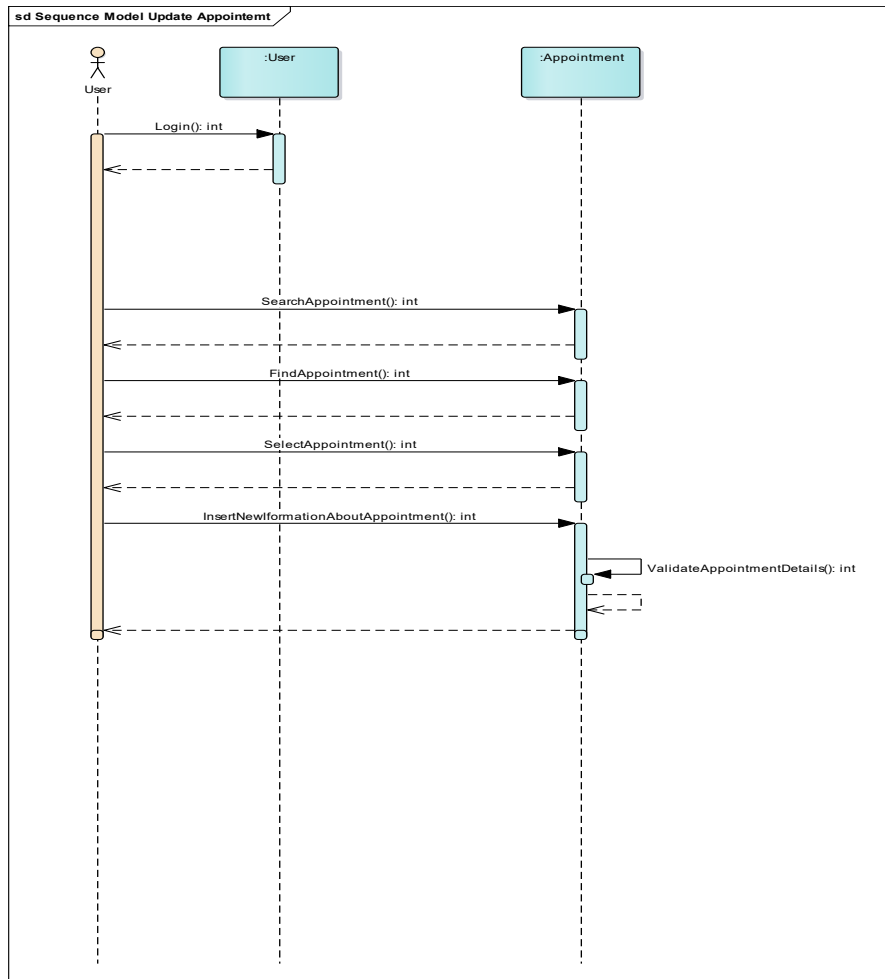


Figure 11: Update Appointment Sequence Diagram

In Figure 11, the sequence diagram depicts the process of updating an appointment in the system. The diagram involves one actor, which is the user, and two lifelines representing different system components.

The sequence starts with the user sending a login request to the user lifeline. Upon processing the request, the user lifeline responds with a message confirming the successful login. Subsequently, the user interacts with the appointment lifeline, initiating a search for appointments. The appointment lifeline processes the request and returns a message to the user, presenting the search results for appointments.

Following this, the user selects a specific appointment by sending a message to the appointment lifeline. The appointment lifeline acknowledges the selection and responds with a confirmation message. The user then proceeds to interact with the appointment lifeline again, this time requesting to insert new information about the appointment. The appointment lifeline validates the appointment details and returns a message to itself for further processing.

Finally, the appointment lifeline sends a message back to the user actor, indicating the successful update of the appointment information. This sequence diagram illustrates the step-by-step interactions between the user and the appointment lifeline for updating an appointment in the system.

4.5.5 Cancel Appointment Sequence Diagram

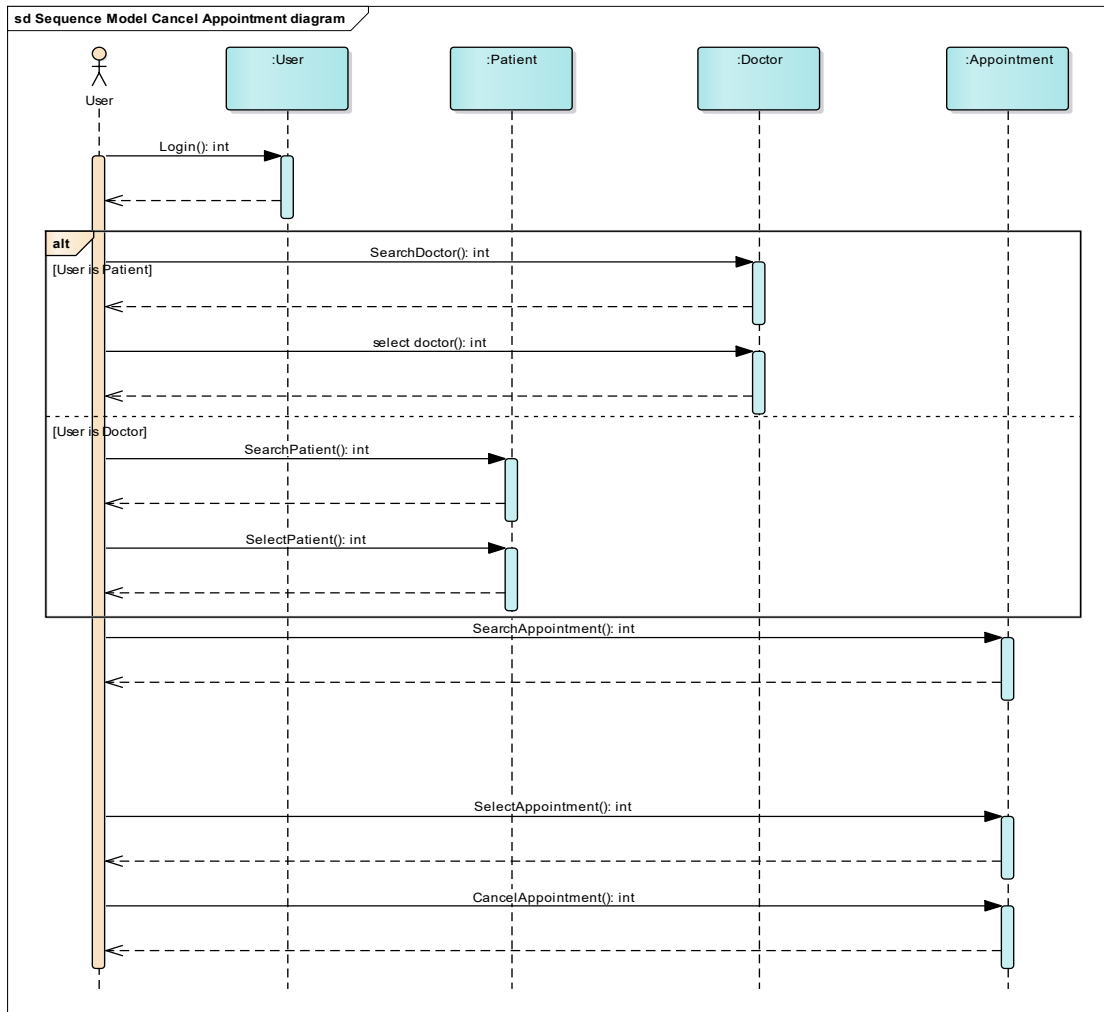


Figure 12: Cancel Appointment Sequence Diagram

In Figure 12, the sequence diagram illustrates the interaction between the user actor (patient) and various lifelines in the system. The sequence begins with the user actor sending a login request to the user lifeline, which validates the login credentials and responds with a message confirming the successful login.

Next, the user actor interacts with the patient lifeline, initiating a search for doctors. The patient lifeline processes the request and returns a message to the user lifeline, presenting the search results for doctors. The user actor then sends a message to the doctor lifeline, selecting a specific doctor. The doctor lifeline acknowledges the selection and responds with a confirmation message, which is returned to the user lifeline.

At this point, the actor in focus becomes the doctor. The user actor (doctor) sends a message to the patient lifeline, initiating a search for patients. The patient lifeline processes the request and returns a message to the user lifeline, presenting the search results for patients. The user

actor then sends a message to the patient lifeline, selecting a specific patient. The patient lifeline acknowledges the selection and responds with a confirmation message, which is returned to the user lifeline.

The user actor (patient) now interacts with the appointment lifeline, initiating a search for appointments. The appointment lifeline processes the request and returns a message to the user lifeline, presenting the search results for appointments. The user actor then sends a message to the appointment lifeline, selecting a specific appointment. The appointment lifeline acknowledges the selection and responds with a confirmation message, which is returned to the user lifeline.

Finally, the user lifeline sends a message to the appointment lifeline, requesting to cancel the selected appointment. The appointment lifeline processes the request and returns a message to the user lifeline, confirming the cancellation of the appointment.

4.5.6 Medical Record Sequence Diagram

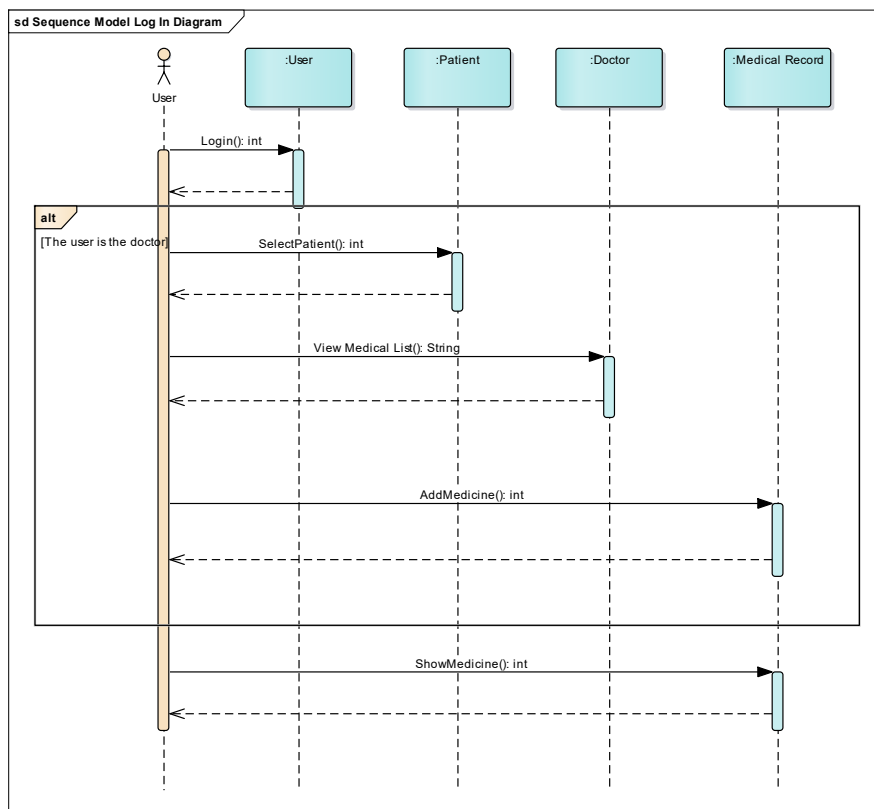


Figure 13: Medical Record Sequence Diagram

In Figure 13, the sequence diagram illustrates the medical record process. It involves one actor, the User, and four lifelines representing different components.

The User actor initiates the process by sending a login message to the User lifeline, representing the user component. The User lifeline verifies the login credentials and returns a message to the User actor, indicating the success or failure of the login.

Next, the User actor sends a message to the Patient lifeline, requesting the selection of a patient. The Patient lifeline processes the request and returns the message to the User actor, providing the selected patient information.

Subsequently, the User actor sends a message to the Doctor lifeline, intending to view the medical list. The Doctor lifeline retrieves the medical list and returns the message to the User actor, presenting the requested information.

Following that, the User actor sends a message to the Medical Record lifeline, specifically targeting the viewing of medicines. The Medical Record lifeline retrieves the medicine data and returns the message to the User actor, providing the relevant details.

Lastly, the User actor sends a message to the Medical Record lifeline, requesting the display of medicines. The Medical Record lifeline processes the request and returns the message to the User actor, displaying the requested medicine.

4.5.7 Create Prescription Sequence Diagram

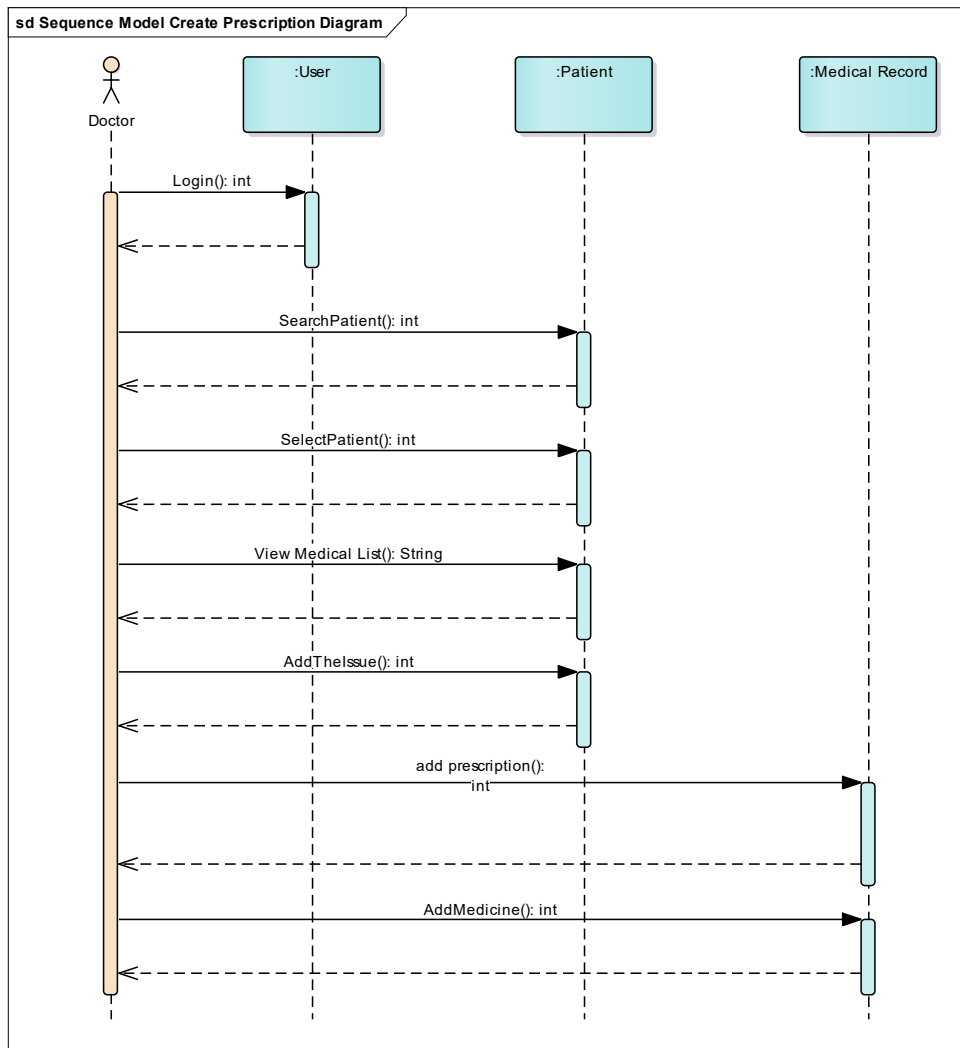


Figure 14: Create Prescription Sequence Diagram

In Figure 14, the sequence diagram depicts the process of creating a prescription by the doctor actor. The sequence begins with the doctor actor sending a login request to the user lifeline. The user lifeline validates the login credentials and responds with a message confirming the successful login.

Next, the doctor actor interacts with the patient lifeline, initiating a search for patients. The patient lifeline processes the request and returns a message to the doctor actor, presenting the search results for patients. The doctor actor then sends a message to the patient lifeline, selecting a specific patient. The patient lifeline acknowledges the selection and responds with a confirmation message, which is returned to the doctor actor.

The doctor actor proceeds by sending a message to the patient lifeline, requesting to view the patient's medical list. The patient lifeline retrieves the medical list and returns it to the

doctor actor. The doctor actor then sends a message to the patient lifeline, adding the medical issue to the patient's record. The patient lifeline acknowledges the addition and responds with a confirmation message, which is returned to the doctor actor.

Next, the doctor actor sends a message to the medical record lifeline, specifying the issue to be added. The medical record lifeline processes the request and returns a message back to the doctor actor, confirming the addition of the prescription. Finally, the doctor actor sends a message to the medical record lifeline, adding the prescribed medicine. The medical record lifeline acknowledges the addition and responds with a confirmation message, which is returned to the doctor actor.

4.5.8 View Prescription Sequence Diagram

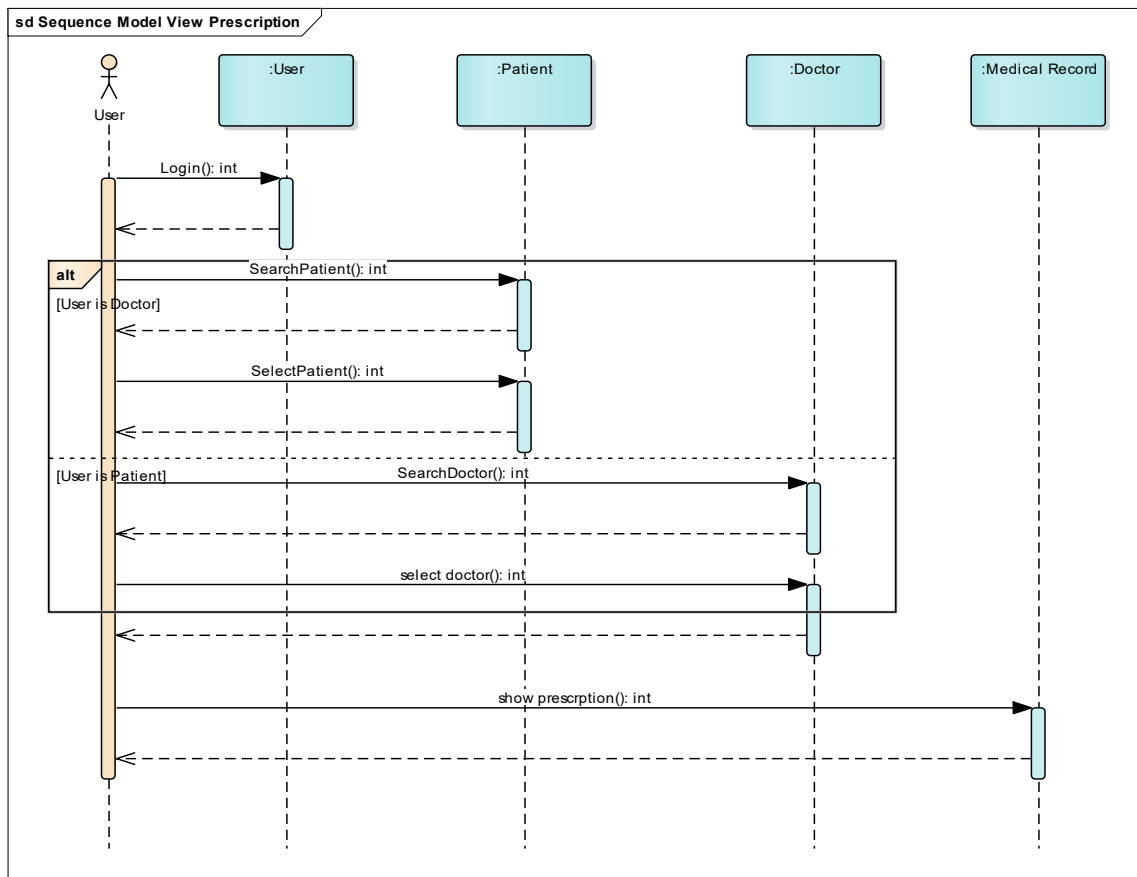


Figure 15: View Prescription Sequence Diagram

In Figure 15, the sequence diagram illustrates the interaction between the user (doctor) and the lifelines in the system. The sequence begins with the user (doctor) sending a login request to the user lifeline. The user lifeline verifies the login credentials and responds by returning a message confirming the successful login.

Next, the user (doctor) interacts with the patient lifeline, initiating a search for patients. The patient lifeline processes the request and returns a message to the user (doctor), presenting the search results for patients. The user (doctor) then sends a message to the patient lifeline, further specifying the search for a particular patient. The patient lifeline acknowledges the search request and responds with the relevant information, which is returned to the user (doctor).

At this point, the user becomes a patient and sends a message to the doctor lifeline, indicating the selection of a specific doctor. The doctor lifeline processes the request and returns a message to the user (patient) actor, confirming the selection of the doctor.

The user then sends a message to the medical record lifeline, requesting to view the prescription. The medical record lifeline retrieves the prescription information and returns it as a message to the user actor.

4.5.9 Update Prescription Sequence Diagram

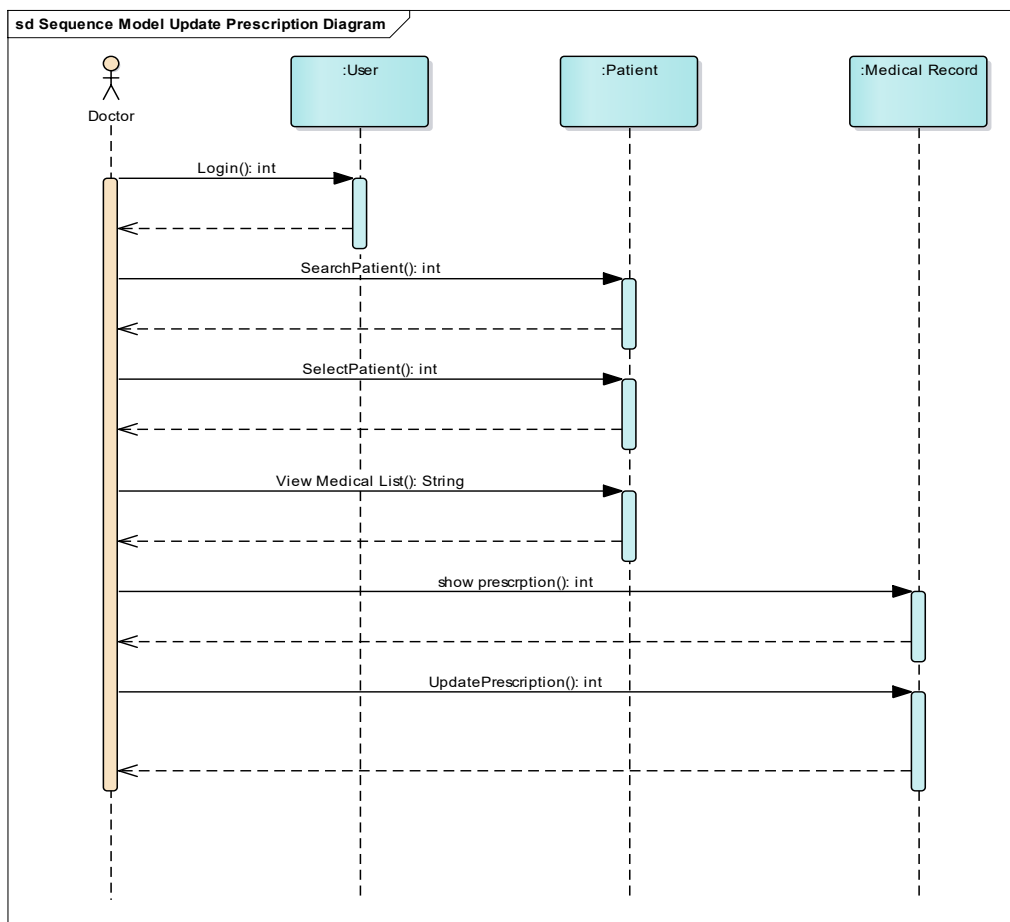


Figure 16: Update Prescription Sequence Diagram

In Figure 16, the sequence diagram depicts the process of updating a prescription. The actor, a doctor, initiates the sequence by sending a login message to the user lifeline. The user lifeline verifies the login and responds by returning a message to the doctor actor, confirming the successful login.

Next, the doctor actor interacts with the patient lifeline, sending a message to search for a patient. The patient lifeline processes the request and returns a message to the doctor actor, presenting the search results for patients. The doctor actor then selects a specific patient by sending a message to the patient lifeline. The patient lifeline acknowledges the selection and responds with the relevant information, which is returned to the doctor actor.

The doctor actor proceeds to send a message to the patient lifeline, requesting to view the medical list. The patient lifeline retrieves the medical list and returns it as a message to the doctor actor. The doctor actor then sends a message to the medical record lifeline, specifically requesting to show the prescription. The medical record lifeline retrieves the prescription information and returns it to the doctor actor as a message.

Finally, the doctor actor sends a message to the medical record lifeline, indicating the intention to update the prescription. The medical record lifeline processes the update request and returns a message to the doctor actor, confirming the successful prescription update.

4.5.10 Add Patient Sequence Diagram

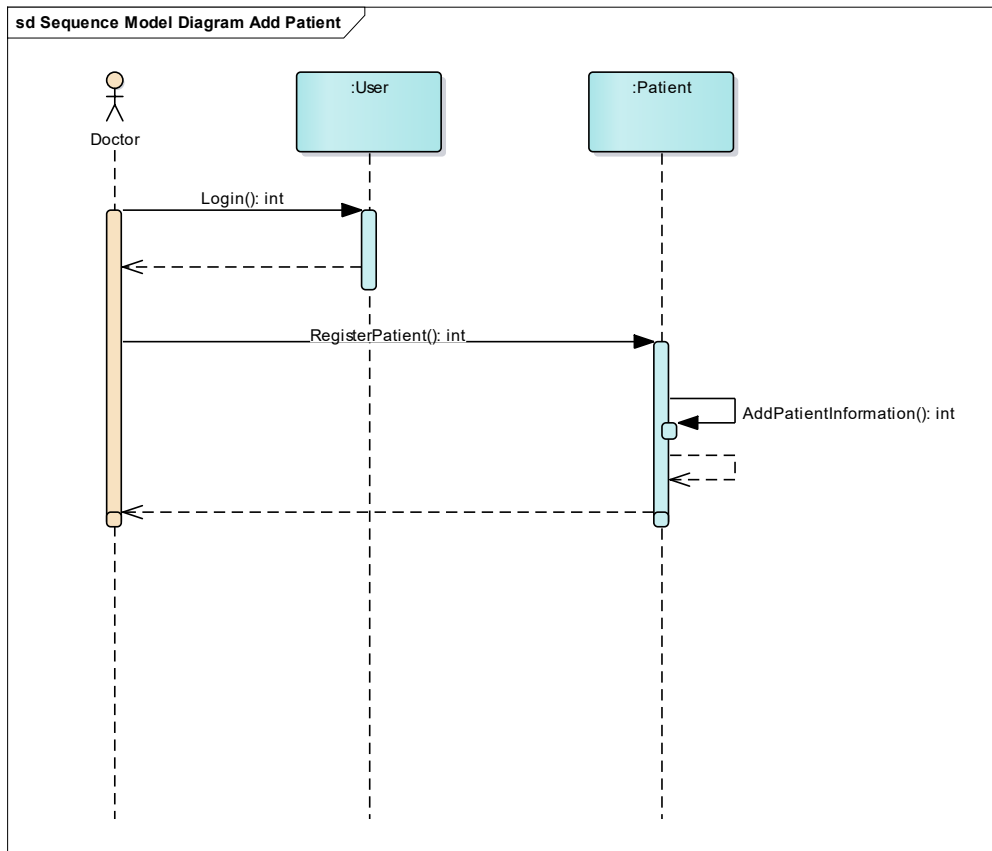


Figure 17: Add Patient Sequence Diagram

In Figure 17, the sequence diagram depicts the interactions between the Doctor actor and three lifelines representing different components: User, Patient.

The Doctor actor initiates the process by sending a login message to the User lifeline, representing the user component. The User lifeline validates the login and returns a message to the Doctor actor, confirming the success or failure of the login.

Next, the Doctor actor sends a message to the Patient lifeline, requesting the registration of a new patient. The Patient lifeline processes the request and adds the patient information then returns the message to the patient lifeline, the patient lifeline returns the message to the Doctor actor, indicating the successful registration.

4.5.11 Delete Patient Sequence Diagram

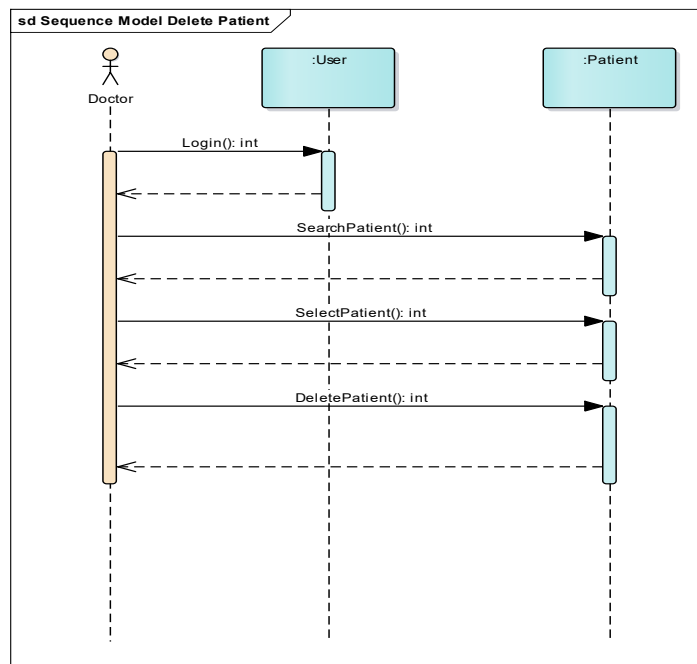


Figure 18: Delete Patient Sequence Diagram

In Figure 18, the sequence diagram illustrates the process of deleting a patient. The actor, a doctor, initiates the sequence by sending a login message to the user lifeline. The user lifeline verifies the login and responds by returning a message to the doctor actor, confirming the successful login.

Next, the doctor actor interacts with the patient lifeline, sending a message to search for a patient. The patient lifeline processes the request and returns a message to the doctor actor, presenting the search results for patients. The doctor actor then selects a specific patient by sending a message to the patient lifeline. The patient lifeline acknowledges the selection and responds with the relevant information, which is returned to the doctor actor.

The doctor actor proceeds to send a message to the patient lifeline, requesting to delete the patient. The patient lifeline processes the deletion request and returns a message to the doctor actor, confirming the successful deletion.

4.5.12 Update Patient Sequence Diagram

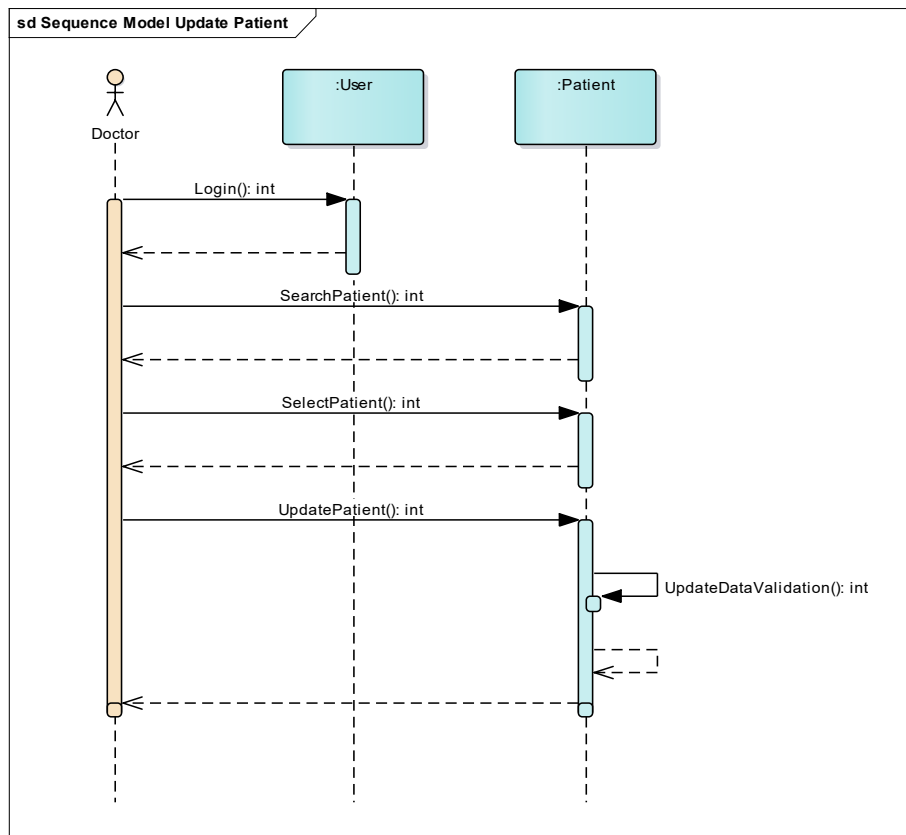


Figure 19: Update Patient Sequence Diagram

In Figure 19, the sequence diagram represents the process of updating patient information. The actor, a doctor, begins by sending a login message to the user lifeline. The user lifeline validates the login and responds by returning a message to the doctor actor, confirming the successful login.

Next, the doctor actor interacts with the patient lifeline, sending a message to search for a patient. The patient lifeline processes the request and returns a message to the doctor actor, presenting the search results for patients. The doctor actor then selects a specific patient by sending a message to the patient lifeline. The patient lifeline acknowledges the selection and responds with the relevant patient information, which is returned to the doctor actor.

The doctor actor proceeds to send a message to the patient lifeline, requesting to update the patient's data. The patient lifeline performs data validation to ensure the updated information is valid. Once the validation is completed, the patient lifeline returns a message to the doctor actor, confirming the successful update.

4.5.13 UML Wireframe

The hero wireframe consists of a navigation bar positioned at the top of the page. The navigation bar contains a logo, representing the brand or website, and a menu, which provides options for navigating through different sections or pages of the site.

In the center of the hero wireframe, there is a picture enclosed within a rounded shape. This picture serves as a focal point and can be an image representing the main theme or purpose of the website.

Adjacent to the picture, there is a descriptive section that provides information, or a brief introduction related to the content of the website. This section may include text or other elements to enhance the visual presentation.

Additionally, there are two buttons located within the hero wireframe.

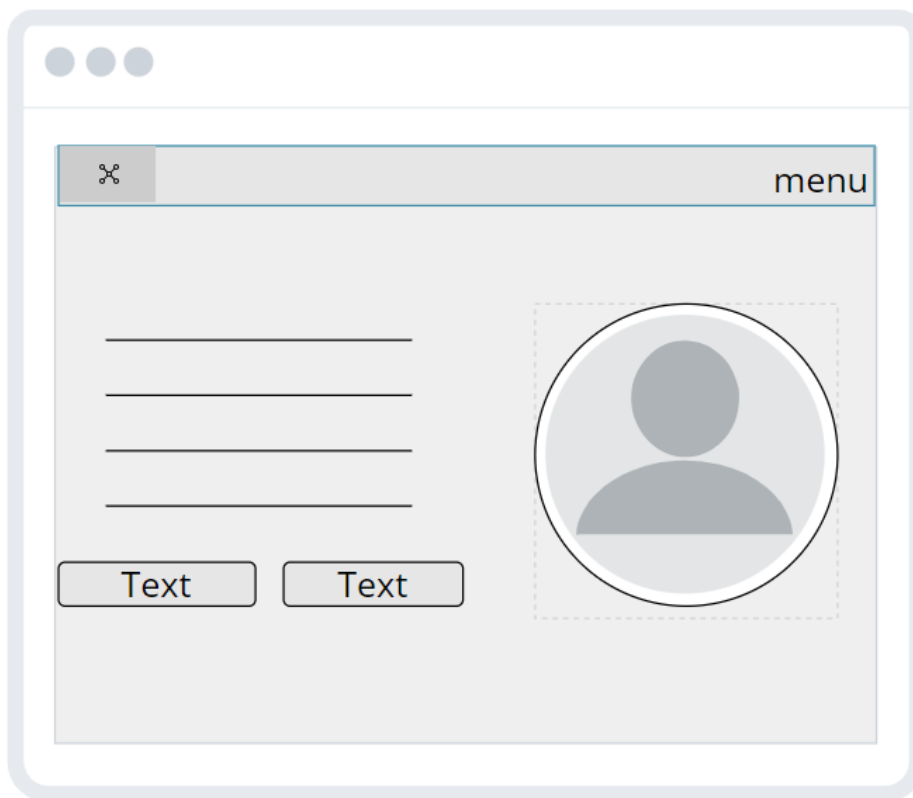


Figure 20: Hero Wireframe

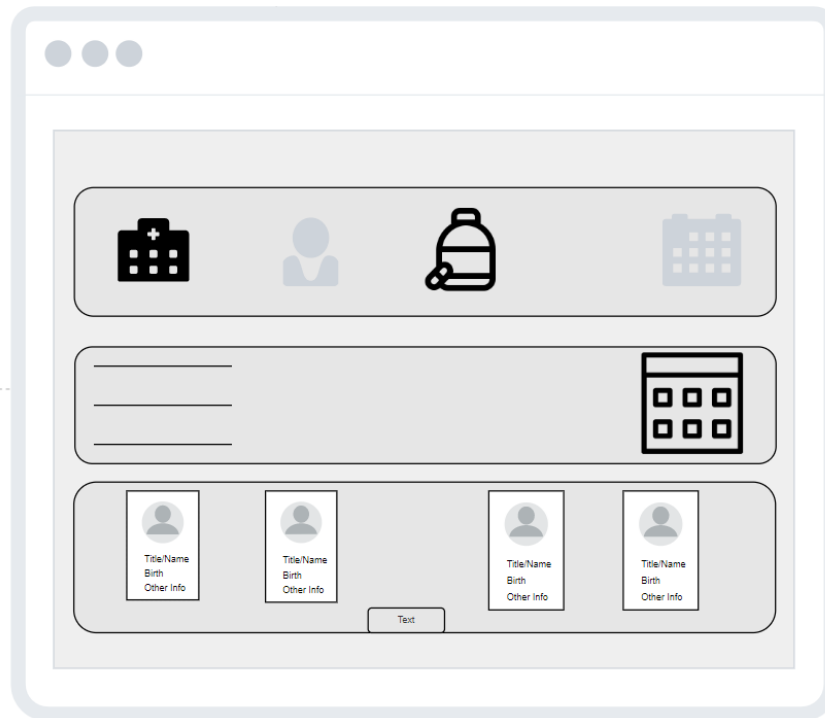


Figure 21: Service Section Wireframe

The service section of the wireframe features four logos, representing different services or categories offered by the website or business. These logos serve as visual cues to indicate the variety of services available.

Below the logos, there is a picture of a man standing with a calendar. This image visually represents the concept of appointment scheduling. It can be a symbolic representation or a real-life depiction of a person using a calendar to schedule appointments.

Adjacent to the picture, there is a description that provides information about the appointment scheduling service.

Below the description, there is a grid layout displaying information about doctors or medical professionals. This grid may include their names, specialties, or other relevant details. Each grid item can be clickable or interactive, allowing users to access more information about a specific doctor.

Finally, there is a button associated with the grid.

The doctor's wireframe showcases a navigation bar with a logo and menu options. This navigation bar provides users with easy access to different sections or functionalities of the website related to doctors.

In the center of the wireframe, there is a grid layout that displays doctors' profiles. Each grid item represents an individual doctor and typically includes relevant information such as their name, specialty, and profile picture. Users can browse through the grid to explore different doctors and their respective profiles.

The grid layout allows for a visually organized presentation of doctors' information, making it easier for users to compare and choose the desired doctor based on their preferences or requirements.

Overall, the doctor's wireframe emphasizes the importance of providing users with a clear and accessible interface to explore and choose from a selection of doctors. The grid layout helps in presenting the doctors' profiles in a visually appealing and user-friendly manner.

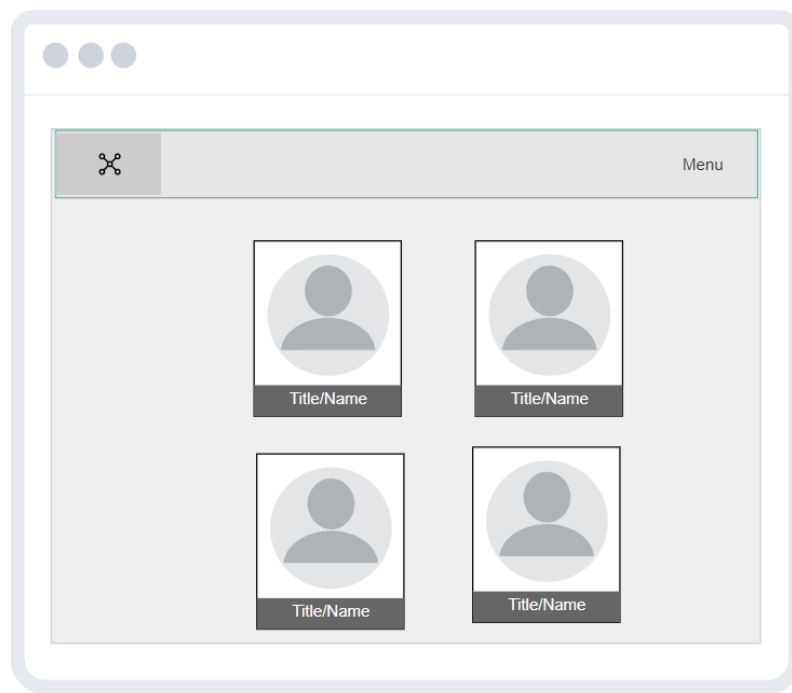


Figure 22: Doctor's Page Wireframe

The doctors page, after login, features a navigation bar with a logo and menu options, providing doctors with easy access to different sections or functionalities of the website.

In the middle of the page, there is a section dedicated to displaying patient appointments. This section presents a list or a grid layout showcasing the appointments scheduled by patients with the particular doctor. Each appointment entry typically includes relevant information such as the patient's name, appointment date and time, and any additional details or notes.

By having a centralized section for patient appointments, doctors can conveniently view and manage their scheduled appointments. This helps doctors stay organized and ensures they have easy access to the necessary information for each appointment.

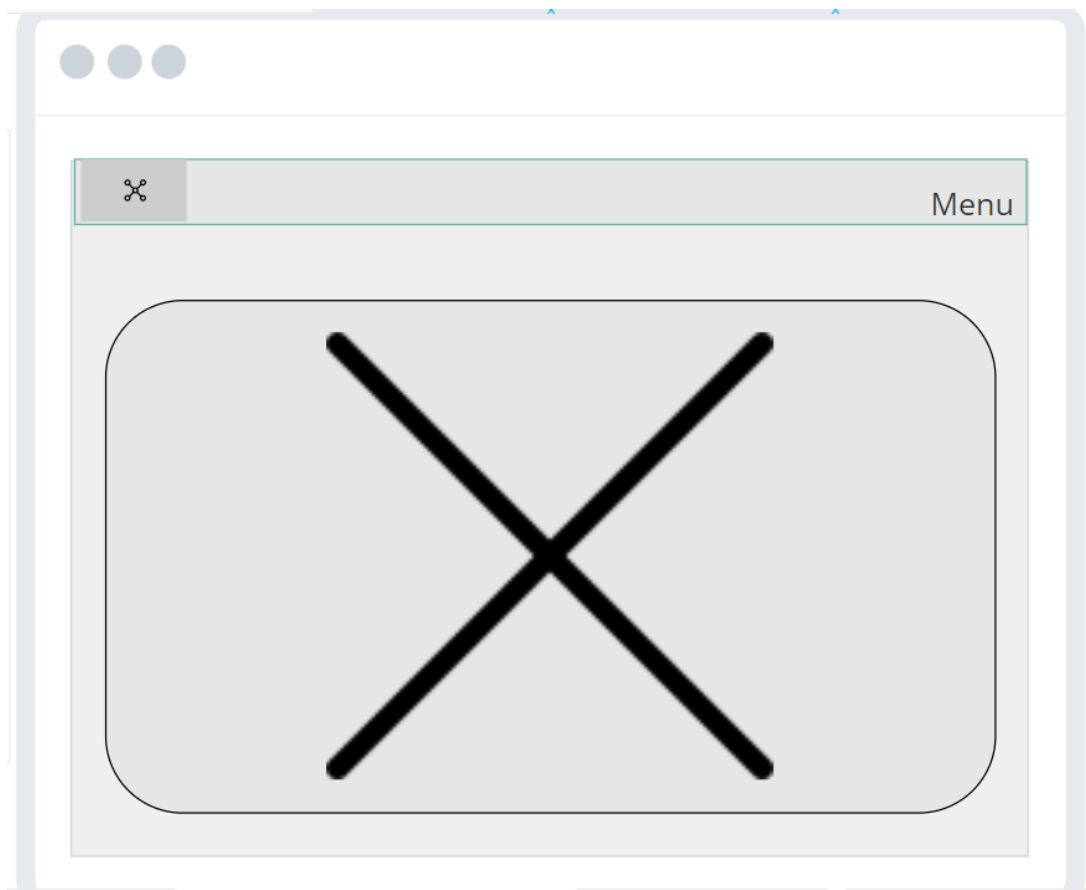


Figure 23: Doctors Page After Log In

5 HTML PROTOTYPE

For creating the HTML prototype, the process involved designing the website pages using UML diagrams. These diagrams helped in visualizing the overall structure and relationships between different components of the website.

Next, Figma wireframes were created for each page based on the UML diagrams. These wireframes served as a blueprint for the visual design and layout of the website. They provided a detailed representation of the user interface, including the placement of elements, navigation, and overall visual hierarchy.

After finalizing the wireframes, the design was implemented into code using HTML, CSS, JS, and Bootstrap 5 framework. HTML was used to structure the content, CSS was used to style the elements and define the visual appearance, while JavaScript was employed for adding interactivity and dynamic behavior to the website. Bootstrap 5 was utilized to facilitate responsive design and enhance the overall user experience.

By following this process, a functional HTML prototype was created, which closely resembled the final design of the website. This prototype served as a tangible representation of the website's layout, functionality, and user interface.

5.1 The Figma Wireframe

After creating UML diagrams, the next step involved creating wireframes using Figma. Figma is a powerful online design and prototyping tool that allows users to create visually appealing and interactive designs for various purposes, such as websites and mobile applications.

In the case of the website, eight wireframes were created . These wireframes served as a visual representation of the layout, structure, and content of each page.

These Figma wireframes served as a foundation for the visual design and layout of the website, providing a clear understanding of how each page should be structured and how elements should be arranged.

1. Home Page
2. Doctors page
3. Hospitals Page

4. Medicines Page
5. Login pages for patient consist of three parts (log in, create account, Reset Password)
6. Login Pages for doctors consist of three parts (log in, create account, reset password)
7. Appointment page.

For each one of these pages, two types of wireframes were created:

1. Low Fidelity wireframes
1. High fidelity wireframes

5.1.1 Low Fidelity Wireframe: Home Page

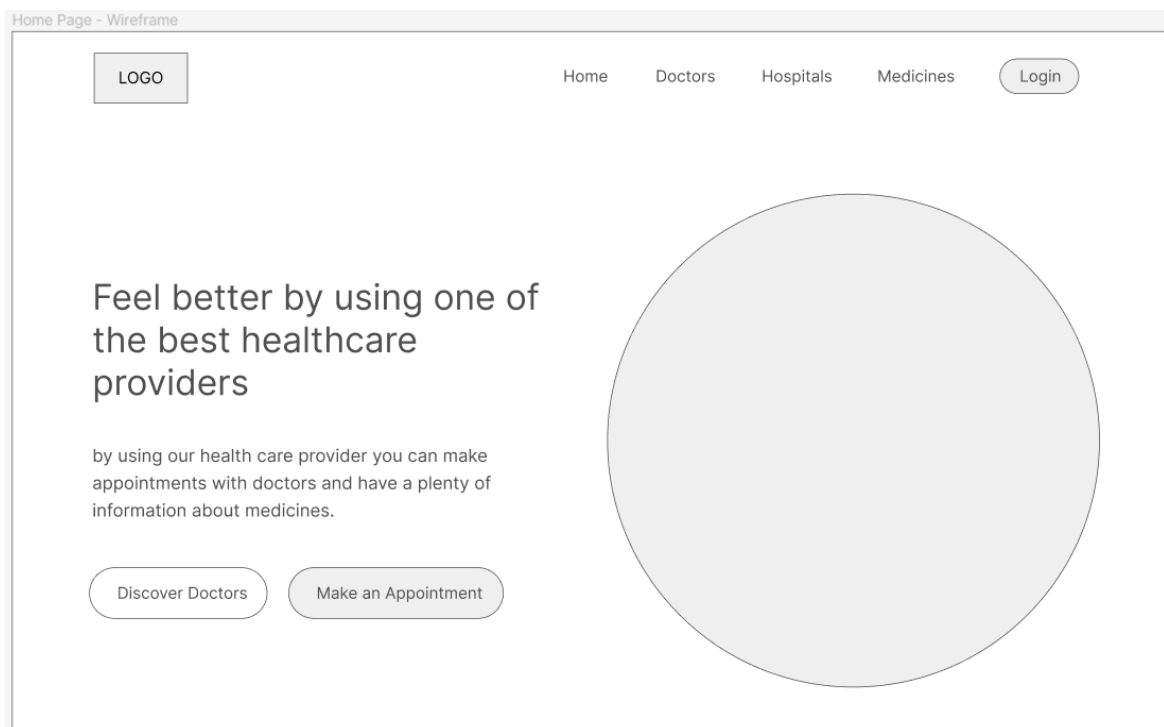


Figure 24: Lo-Fi Hero Section Wireframe

5.1.2 Low Fidelity Wireframe: Service section

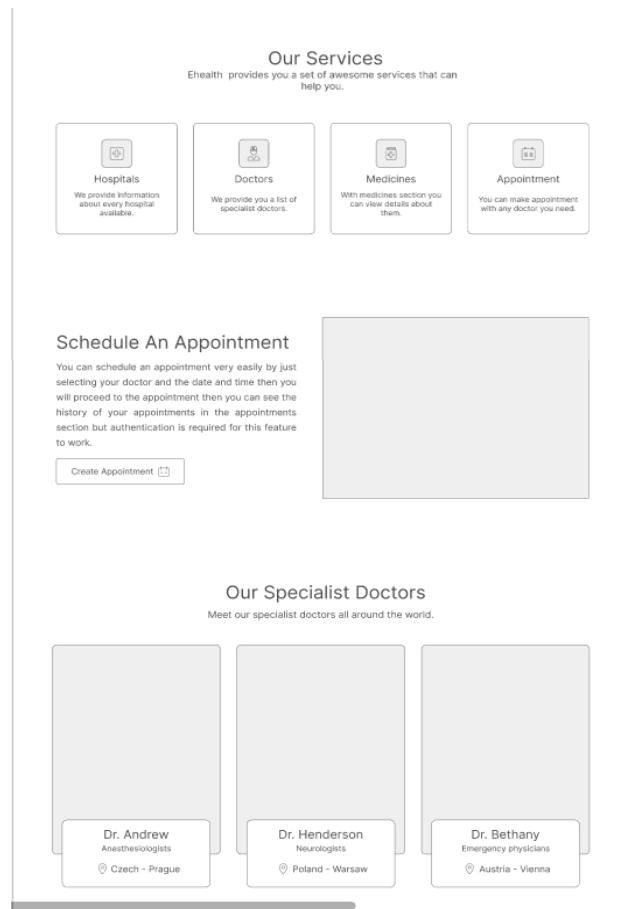


Figure 25: Lo-Fi Service Section Wireframe

In the low fidelity wireframe in the service section of the home page, a grid layout is utilized to organize the content. Within each grid, a smaller grid is incorporated to accommodate a logo representing the specific service, accompanied by a description text positioned below it. This arrangement allows for a clear and structured presentation of the services offered. Additionally, another grid is employed for the scheduling appointment feature, which includes a smaller grid containing a button for creating appointments. This button incorporates a small logo and is accompanied by a descriptive text positioned above it, providing users with a clear understanding of its purpose and functionality.

Furthermore, in the wireframe's specialty doctors section, three additional grids are allocated to display information about specialized doctors. Each grid consists of the doctor's name, a description of their expertise, and an accompanying map logo.

5.1.3 Low Fidelity Wireframe: Footer Section



Figure 26: Lo-Fi Footer Section Wireframe

In the low fidelity wireframe of the footer section, a small grid layout is dedicated to displaying logos. These logos typically represent the various social media platforms associated with the website. Adjacent to the grid, there are text elements serving as links. These text links are typically used to provide additional information or navigate to relevant pages within the website. This layout allows for a visually organized presentation of logos and text links in the footer section.

5.1.4 Low Fidelity Wireframe: Login pages

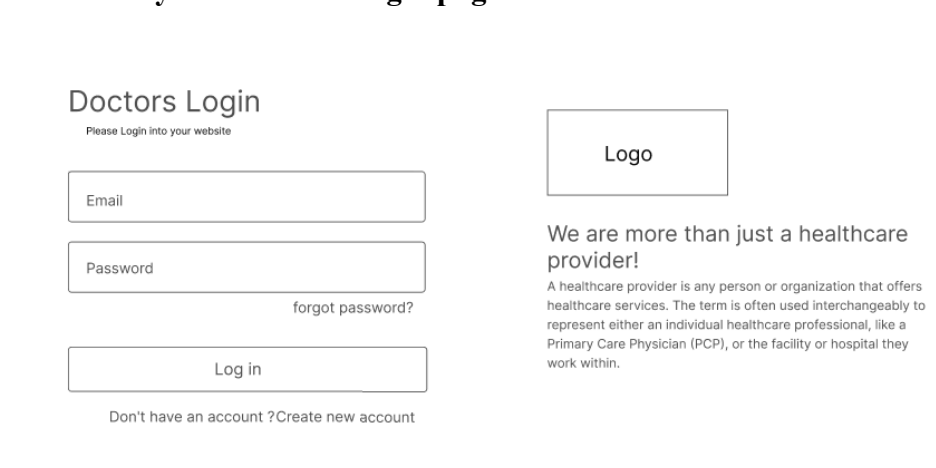


Figure 27: Lo-Fi Login Page Wireframe

The low fidelity wireframe for the Doctors login page exhibits a descriptive section that encompasses two input fields for email and password entry. These fields serve as the means for doctors to provide their login credentials. Additionally, a login button is positioned adjacent to the input fields, enabling doctors to initiate the login process. Next to this section, a logo is prominently displayed, representing the brand or identity associated with the

website. Beneath the logo, a descriptive text further enhances the visual presentation and provides contextual information to the doctors. This layout facilitates a clear and intuitive interface for doctors to access the login functionality and engage with the website's features.

5.1.5 Low Fidelity Wireframe: Sign Up

Figure 28: Lo-Fi Sign Up Wireframe

The low fidelity wireframe for the Create Account page showcases a layout consisting of six input fields, allowing users to input their personal information. These input fields serve as containers for users to provide details such as their name, email, password, address, phone number, and any other relevant information required for creating an account. Accompanying these input fields is a prominent "Create Account" button, enabling users to proceed with the account creation process.

It is worth noting that while the overall design and structure of the three pages (Login, Create Account, and Forgot Password) are consistent, the specific input fields, links, and buttons may vary depending on the page's purpose. This uniformity in design elements promotes a cohesive and intuitive user experience throughout the website, allowing users to navigate and interact with the different functionalities seamlessly.

5.1.6 Low Fidelity Wireframe: Hospitals page

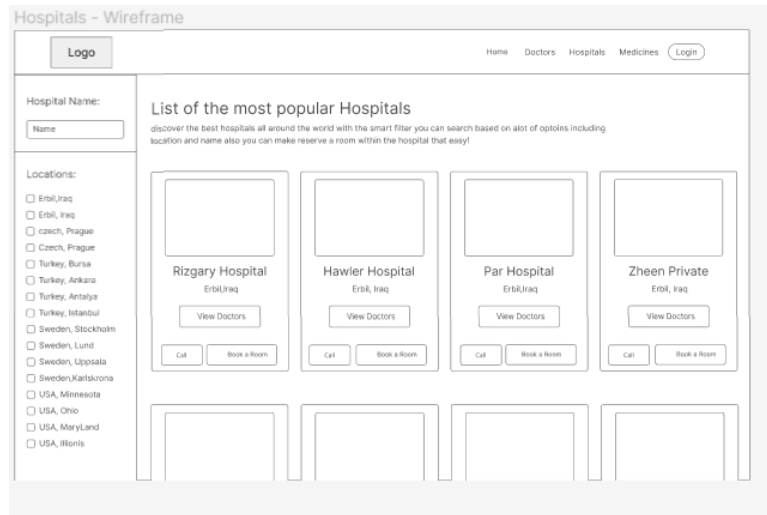


Figure 29: Lo-Fi Hospitals Page Wireframe

The low fidelity wireframe for the Hospitals page exhibits a structured layout that encompasses various components. At the top, a navigation bar is present, housing the logo and links to other pages, facilitating easy access and navigation within the website. Adjacent to the navigation bar, a search box is situated, enabling users to search for specific hospitals or related information.

Beneath the search box, a section dedicated to displaying hospital locations is incorporated. It may include a list of locations accompanied by checkmarks, signifying the availability or presence of hospitals in those areas.

The central portion of the wireframe consists of a grid-like structure showcasing hospital names and their respective locations. This grid layout aids in presenting the information in an organized manner. Each entry in the grid may be accompanied by three buttons, namely "View Doctors," "Book a Room," and "Call." These buttons allow users to access additional details, such as doctors associated with the hospital, room booking options, and contact information for further inquiries.

A descriptive text may be positioned above the grid, providing an overview or pertinent information related to hospitals, their services, or any other relevant details.

Overall, this low fidelity wireframe effectively combines elements such as navigation, search functionality, location display, grid layout, buttons, and descriptive text to create a user-friendly and informative Hospitals page.

5.1.7 Wireframe 6 : Doctors page

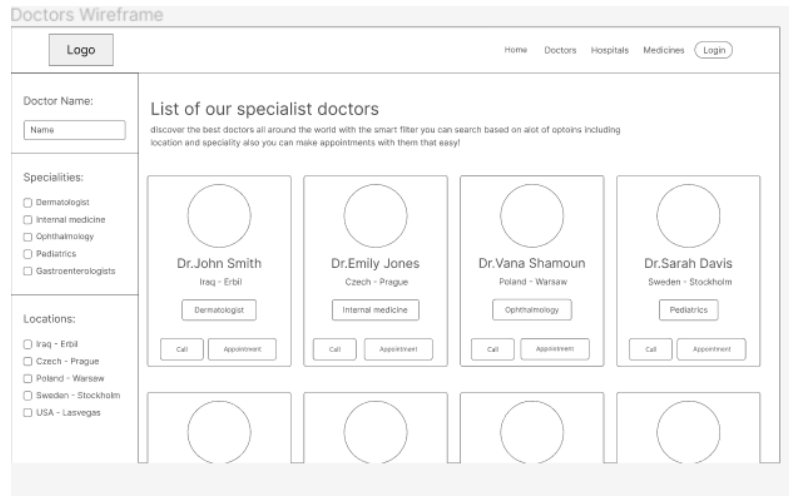


Figure 30: Lo-Fi Doctor's Page Wireframe

The low fidelity wireframe for the Doctors page shares a similar structure to the Hospitals page, maintaining consistency in the overall design. The primary components, such as the navigation bar and search box, remain unchanged.

Below the search box, the wireframe includes a section for displaying individual doctors' profiles. Notably, the pictures of the doctors are presented in a rounded format, adding a visually appealing touch to the design. Beneath each doctor's picture, their respective location is mentioned, providing users with information regarding their practice or affiliation.

Additionally, the wireframe introduces a new element specific to the Doctors page - the inclusion of the doctors' specialties. This information is positioned below the location details, enabling users to quickly identify the area of expertise for each doctor.

While maintaining the grid-based structure, the modifications in the doctor's page wireframe, including the rounded pictures and the addition of the specialties, aim to enhance the user experience and facilitate easy identification and selection of doctors based on their visual representation and professional expertise.

Overall, this low-fidelity wireframe effectively extends the existing design principles from the Hospitals page to the Doctor's page, incorporating appropriate adjustments to accommodate the unique characteristics of doctor profiles, ultimately enhancing the usability and presentation of information.

5.1.8 Low Fidelity Wireframe: Medicines Page

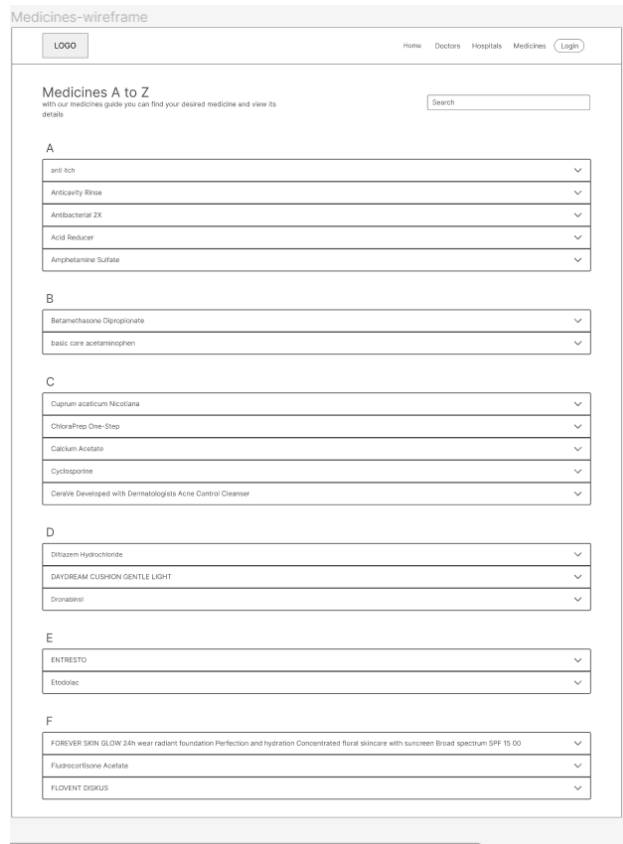


Figure 31: Lo-Fi Medicines Page Wireframe

The low fidelity wireframe for the Medicines page features a structured layout that incorporates essential components for efficient navigation and information retrieval. At the top of the page, there is a prominently placed search box, allowing users to input specific medicine names or keywords to initiate a search process.

Adjacent to the search box, the wireframe includes a drop-down menu. This drop-down menu is designed to provide users with access to detailed descriptions or additional information about specific medicines. By selecting a medicine from the drop-down menu, users can access a comprehensive description, including details such as usage instructions, dosage recommendations, potential side effects, and other relevant information.

The combination of the search box and the drop-down menu empowers users to efficiently explore and retrieve information about different medicines of interest. The search functionality enables targeted searches based on specific medicine names, while the drop-down menu enhances discoverability by providing users with a comprehensive list of available medicines and their associated descriptions. Through this low fidelity

wireframe, the Medicines page aims to offer users a user-friendly interface that facilitates seamless navigation and access to detailed medicine information.

5.1.9 High Fidelity Wireframes: Hero Section

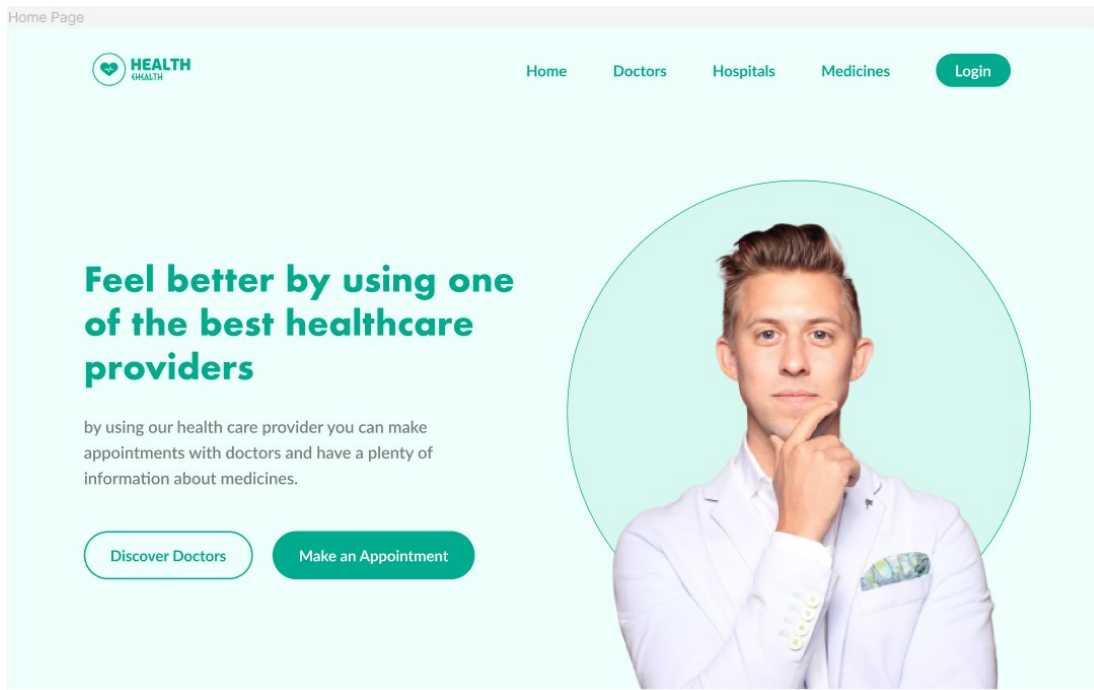


Figure 32: Hi-Fi Hero Section Wireframe

In the high fidelity wireframe, the hero section takes on a prominent role as the visually captivating centerpiece of the website. It provides users with a glimpse of the final visual representation of the website once it is fully developed.

The navigation bar, positioned at the top of the page, incorporates the eHealth logo, serving as a recognizable and representative symbol of the platform. Adjacent to the logo, the menu is displayed, enabling users to access various sections of the website. The text within the navigation bar is presented in a medium dark shade of green-cyan, which adds visual appeal and enhances readability.

To create a visually appealing and intuitive design, the login button is designed with rounded edges, contributing to a cohesive and harmonious overall aesthetic. This rounded shape is also applied to the image within the hero section, adding a consistent visual element throughout the design.

In terms of the hero section's background, a very pale, predominantly white shade with hints of cyan is employed. This color choice contributes to a clean and modern look while allowing the focal points of the hero section, such as the rounded image and accompanying text, to stand out prominently.

By utilizing these design choices and elements in the high fidelity wireframe, the aim is to create an engaging and visually appealing user interface that aligns with the overall brand identity and enhances the user experience on the eHealth website.

5.1.10 High in Fidelity wireframes: Service Section

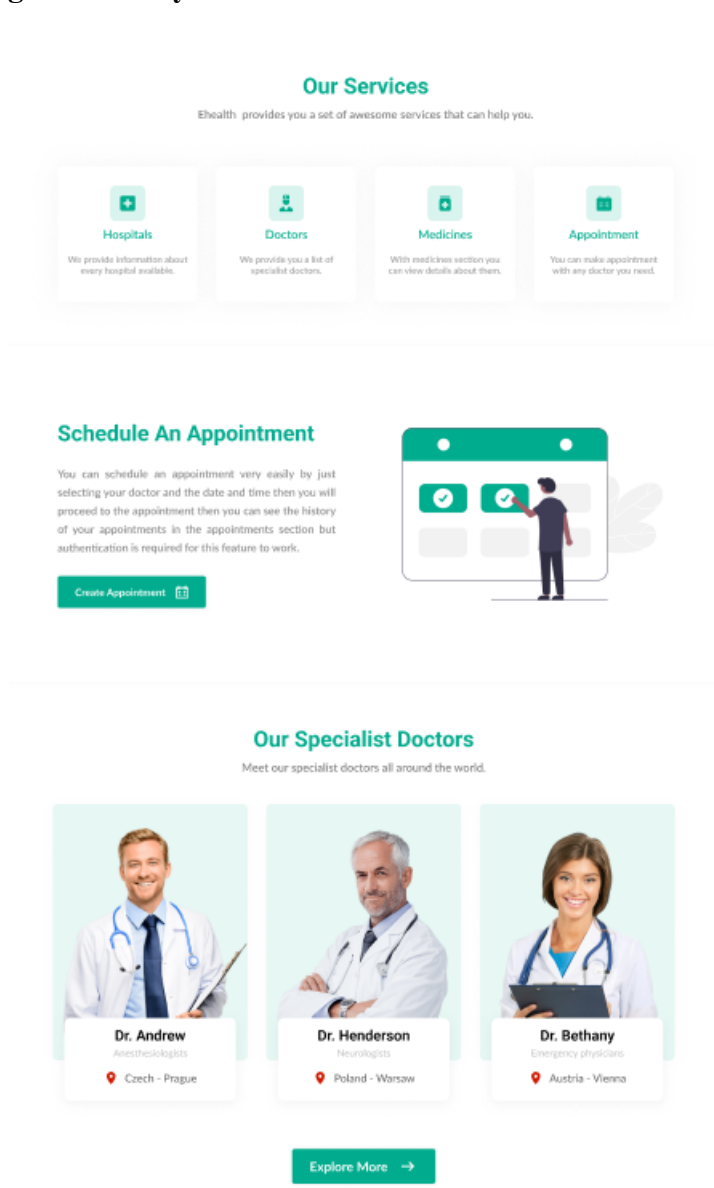


Figure 33: Hi-Fi Service Section Wireframe

In the high fidelity wireframe of the service section, a visually appealing and informative layout is presented. The section is divided into rectangular grids, each representing a specific aspect of the services provided.

The first grid highlights the different service categories, namely hospitals, doctors, medicines, and appointments. Each category is represented by a small logo placed within the grid. The chosen colors for this section are a dark shade of green-cyan, which imparts a sense of professionalism and recognition within the healthcare domain. The overall background color is a very pale cyan, which complements the visual elements and creates a cohesive design.

The second grid focuses on scheduling appointments. It includes relevant information and a concise description of the appointment scheduling process. Additionally, a single button is provided to enable users to easily access the appointment scheduling feature.

The third grid is dedicated to showcasing doctors. It features three rectangular-shaped grids, each containing an image of a doctor, along with their name, description, and a map. This design element helps users quickly identify and connect with doctors of interest. The inclusion of these visual and informational components enhances the user experience and facilitates informed decision-making.

By employing a combination of distinct grid layouts, visually appealing images, and carefully chosen color schemes, the high-fidelity wireframe of the service section aims to create an engaging and intuitive user interface for the eHealth website.

5.1.11 High-Fidelity Wireframe: Footer Section



Figure 34: Hi-Fi Footer Section Wireframe

In the high fidelity wireframe of the footer section, meticulous attention has been given to its design elements. The background color is predominantly a very pale cyan, which creates a subtle and clean aesthetic.

At the top of the footer section, a logo representing the eHealth platform is prominently displayed. Below the logo, a concise text description is presented in shades of gray, adding visual contrast and readability. This choice of colors enhances the overall legibility and ensures that the information is easily comprehensible to users.

Additionally, the footer section includes logos of various social media platforms, symbolizing the presence of eHealth on these platforms. These logos are strategically placed, allowing users to navigate and connect with eHealth through their preferred social media channels.

The combination of a predominantly pale cyan background, a distinct eHealth logo, and a well-crafted text description, along with the inclusion of social media logos, creates a cohesive and visually appealing footer section. This design approach aims to provide users with a seamless and intuitive experience while navigating the eHealth website.

5.1.12 High-Fidelity Wireframe: Doctors Login

Doctors Login
Please Login into your website

Email

Password

[forgot password?](#)

Log in

[Don't have an account ?Create new account](#)

HEALTH
eHEALTH

We are more than just a healthcare provider!

A healthcare provider is any person or organization that offers healthcare services. The term is often used interchangeably to represent either an individual healthcare professional, like a Primary Care Physician (PCP), or the facility or hospital they work within.

Figure 35: Hi-Fi Doctors Login Wireframe

In the high-fidelity wireframe of the doctor's login page, deliberate attention has been given to the visual presentation and design elements. The textual components have been enhanced through the use of colors, creating a visually appealing and engaging interface. Two rectangular-shaped input boxes are prominently displayed, allowing doctors to enter their credentials securely. Adjacent to the input boxes, a login button is visually highlighted through a distinct color, indicating the action of logging in.

Furthermore, the page incorporates the eHealth logo, serving as a visual identifier for the platform. Below the logo, a text description provides additional context and information related to the login process. This combination of elements in the doctor's login page aims to provide a clear and visually appealing interface, facilitating a seamless and intuitive login experience for doctors accessing the eHealth platform.

5.1.13 High-Fidelity wireframes: Doctors Page

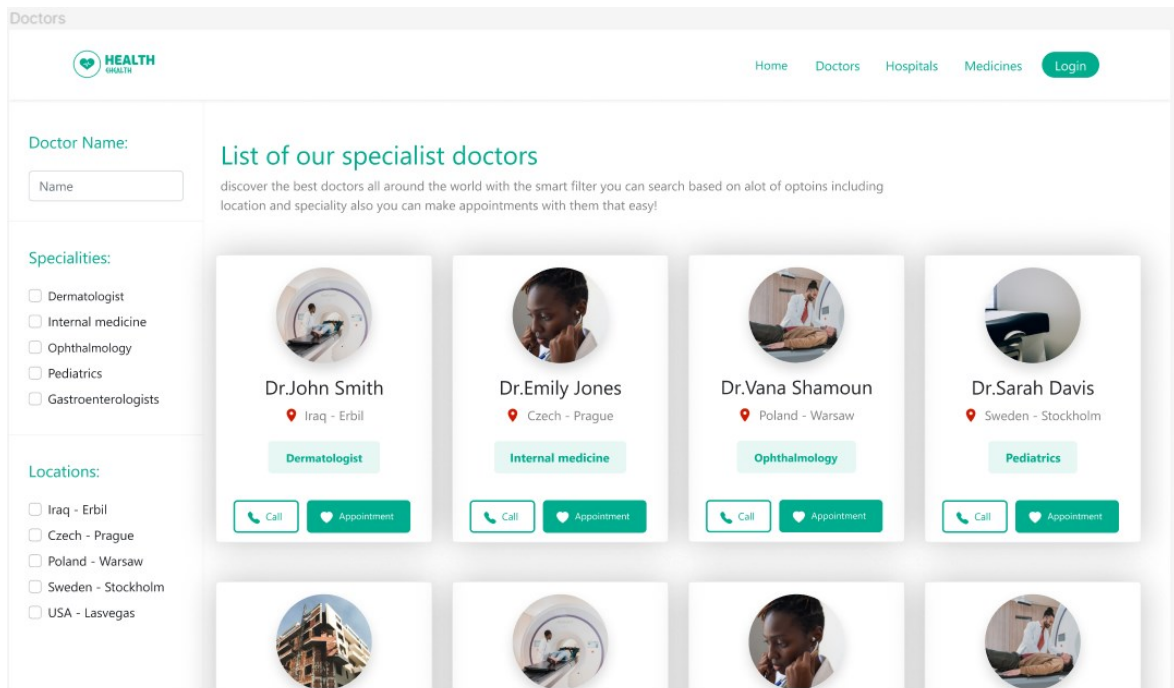


Figure: 36 Hi-Fi Doctors Page Wireframe

In the high-fidelity wireframe of the doctors page, meticulous attention has been given to its visual design and layout. The navigation bar, located at the top of the page, features a logo representing the platform and a horizontally arranged menu. The use of color is purposeful, with the names and login button visually standing out through a medium dark shade of green-cyan, creating a visually appealing contrast.

In the main section of the page, grids have been incorporated to organize and present the doctors' information. Each grid contains a rounded photo of the respective doctor, alongside their specialization. Additionally, two buttons are provided within each grid, offering users convenient options to interact further with the doctor's profile.

To facilitate easy searching and filtering, a side bar is included, housing a search box along with check boxes. This allows users to refine their search criteria and find specific doctors based on their preferences.

The combination of a well-designed navigation bar, visually striking color scheme, organized grids with rounded photos and specialization, and the inclusion of search functionality and check boxes in the side bar collectively contribute to an aesthetically pleasing and user-friendly doctors page. This thoughtful design approach aims to enhance the user experience and facilitate seamless navigation and interaction within the eHealth platform.

5.1.14 High Fidelity Wireframe: Hospitals Page

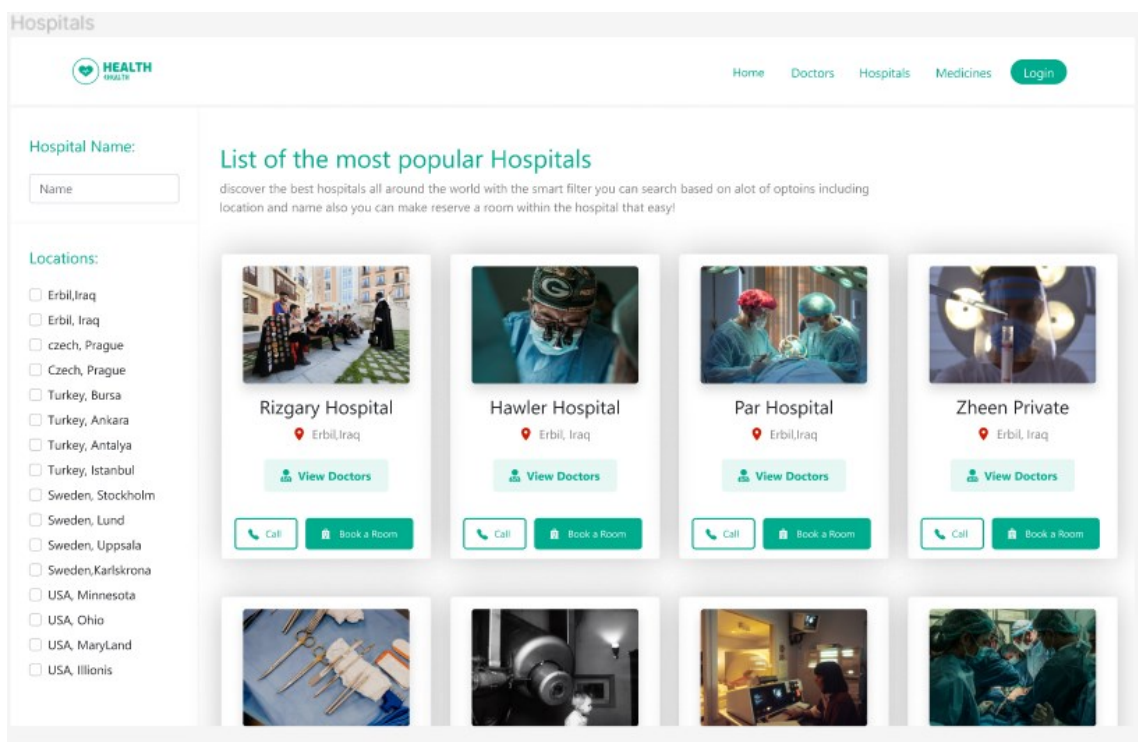


Figure 37: Hi-Fi Hospitals Page Wireframe

The high fidelity wireframe for the hospital page shares a similar design concept with the doctors page. The layout and visual elements are consistent between these pages, providing a unified user experience. Grids or sections are used to organize the content, allowing for easy comprehension and navigation.

Within these grids, rectangular photos are displayed to represent hospitals. Accompanying each representation are the names of the hospitals. A color scheme comprising shades of green-cyan is employed to enhance the visual appeal and maintain a cohesive design aesthetic.

By maintaining design consistency across the hospital and doctors pages, the high fidelity wireframe ensures a seamless and intuitive user interface. Users will find familiarity in

interacting with the website, as they encounter similar elements and functionalities throughout these pages.

5.1.15 High Fidelity Wireframe: Medicines Page

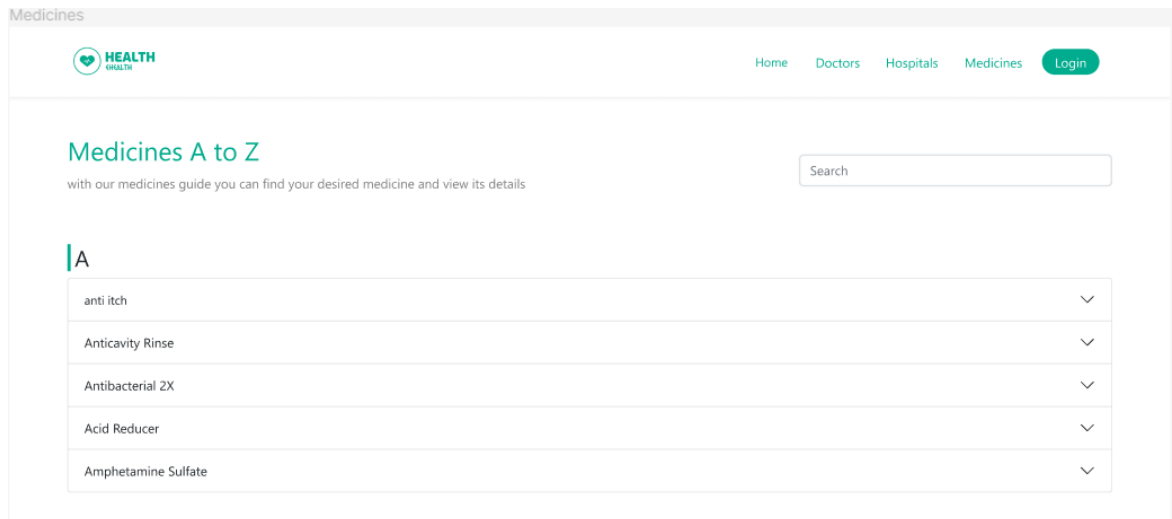


Figure 38: Hi-Fi Medicines Page Wireframe

In the high fidelity wireframe of the medicines page, the design follows a consistent layout with other pages of the website. The primary focus of this page is on textual content, where the emphasis has been placed on coloring the text to enhance its visibility and visual appeal. Additionally, the navigation bar at the top of the page features a logo that represents the website, providing brand recognition and reinforcing the overall identity of the platform.

By selectively applying colors to the texts, the high fidelity wireframe ensures that important information regarding medicines stands out and is easily readable for users. This approach aims to facilitate user engagement and comprehension, allowing them to quickly access relevant details about medications.

6 IMPLEMENTING THE DESIGN INTO CODE

After designing the prototype using Figma, the next step was to implement the design into code. I utilized HTML, CSS, Bootstrap, and JavaScript technologies to transform the visual design into a functional website..

The home page is separated into different sections including:

1. Navigation bar which is used to navigate through different pages
Bootstrap responsive navigation is used to implement it.
2. Hero section its purpose is to explain the purpose of the website to the user and give them an idea about the website by showing them a simple explanation with an image and CTA buttons.
3. Services section is used to give the user information about the services that the website will offer them it's made by using grid layout system of bootstrap with different breakpoints for the purpose of responsiveness.
4. Schedule appointment section this part is for the educational purpose it explains about how to make an appointment it also focuses on that the user should be authenticated to schedule appointments with a button that will take them to the appointments page.
5. Doctors section in this section the most specialized doctors will be shown to gain the trust of the users it displays 4 cards of doctors that are done through bootstrap grid system.
6. Footer Section this section it includes useful links like links of the pages available in the website and contact information with social media buttons it is made through bootstrap grid system to be responsive on different devices and screens.

6.1 Home Page

```
1 var nav = document.querySelector(".navbar");
2
3 window.addEventListener("scroll", () => {
4   if (window.scrollingY >= 20) {
5     console.log("Add");
6     nav.classList.remove("primary-bg-light");
7     nav.classList.add("bg-white");
8     nav.classList.add("shadow-sm");
9   } else {
10    nav.classList.add("primary-bg-light");
11    nav.classList.remove("bg-white");
12    nav.classList.remove("shadow-sm");
13  }
14 });
```

Figure 39: JS Code For Navigation Scroll

The attached JavaScript file serves a specific purpose on the home page, which is to dynamically modify the background color of the navigation bar as the user scrolls down the website. Its primary function is to enhance the user experience by providing a visual cue that distinguishes the navigation bar from other sections.

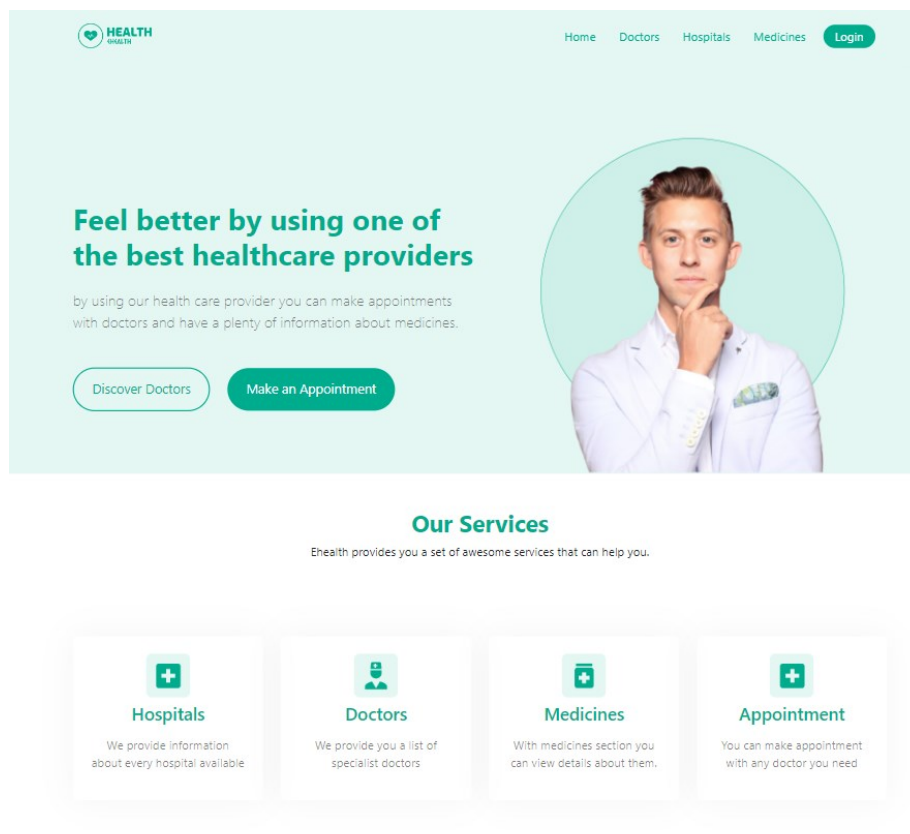


Figure 40: Home Page 1/2

The script achieves this by monitoring the scrolling position of the webpage. When the user scrolls beyond an offset of 20 pixels, the background color of the navigation bar is changed to white. Additionally, a subtle shadow effect is applied to further emphasize the separation between the navigation bar and other content sections.

This dynamic behavior enhances the visual aesthetics and usability of the website, allowing users to easily identify and navigate through the website's sections. By adapting the appearance of the navigation bar based on the user's scrolling activity, the JavaScript file adds a responsive and interactive element to the home page.

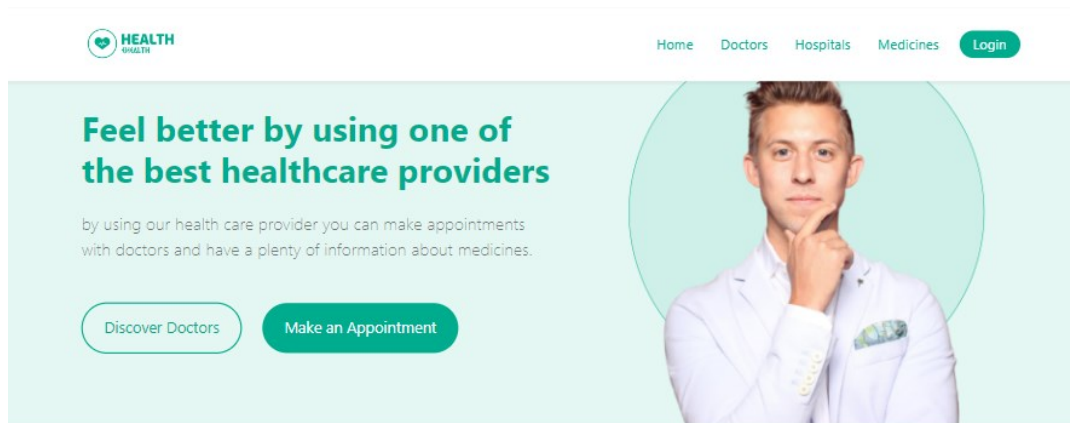


Figure 41: Home Page 2/2

6.2 Doctor Login

The process of doctor registration necessitates the input of distinctive credentials that are unique to each individual practitioner

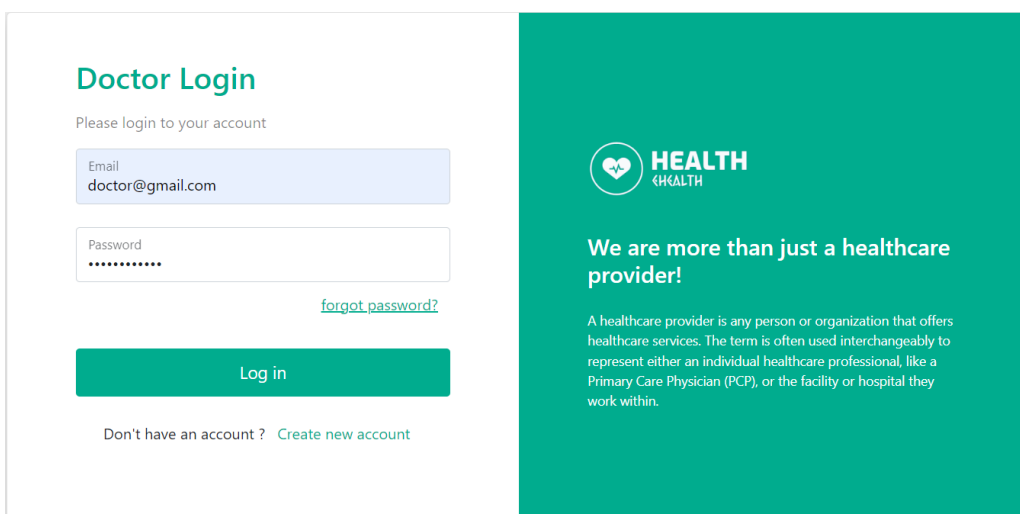


Figure 42: Doctors Log In

The JS code appears to handle the form submission event and perform authentication-related actions. It checks if the entered email and password match any of the allowed credentials. If a match is found, it stores the authentication status in the localStorage and redirects the doctor to the home page. Otherwise, it displays a toast notification and resets the form fields.

Additionally, there is a check for the authentication status stored in the localStorage. If the doctor is authenticated, they are redirected to the home page. If not, and there is no form element present, the doctor is redirected to the doctor login page.

This code snippet demonstrates a basic implementation of doctor authentication and session management in a web application.

```
const authKey = "doctor_authenticated";

const form = document.getElementById("form-submit");
const email = document.getElementById("emailInput");
const password = document.getElementById("passwordInput");
const logoutButton = document.getElementById("logout");

const allowedCredintials = [
  {
    email: "doctor@gmail.com",
    password: "doctor123456",
  },
  {
    email: "doctor2@gmail.com",
    password: "doctor2123456",
  },
];
```

Figure 43: JS Code for Doctor's Login

6.3 Patient Login

The process of patient login necessitates the input of distinctive credentials that are unique to each individual practitioner.

Figure 44: Login Page

```
const authKey = "is_authenticated";

const form = document.getElementById("form-submit");
const email = document.getElementById("emailInput");
const password = document.getElementById("passwordInput");
const logoutButton = document.getElementById("logout");

const allowedCredintials = [
  {
    email: "ornella@gmail.com",
    password: "ornella123456",
  },
  {
    email: "user@gmail.com",
    password: "user123456",
  },
];
```

Figure 45: JS Code For Patient Information

In the Figure 45 it is the JS code is a part of an authentication system implemented in a web application. The system is responsible for verifying user credentials and maintaining the authentication status.

6.4 User-Specific Dashboard: Doctor

Upon doctor login, a distinct dashboard tailored to the needs of doctors is presented. This specialized dashboard provides doctors with access to various features and information

specific to their role. The doctor's dashboard prominently includes sections such as patient management, appointment management, and doctor profiles.

The patient management section allows doctors to view and manage their assigned patients. It provides a comprehensive overview of patient records, medical histories, and any relevant notes or updates. This enables doctors to track and monitor the health progress of their patients, review past diagnoses, and access vital information necessary for effective treatment and care.

In the appointment management section, doctors can efficiently handle their schedule and appointments. They can view upcoming appointments, reschedule or cancel appointments if necessary, and allocate time slots for new patient consultations. This feature enables doctors to effectively manage their time and ensure optimal patient care and engagement.

Furthermore, the doctor's dashboard includes a dedicated area for doctor profiles. This section allows doctors to update their professional information, such as qualifications, specializations, areas of expertise, and contact details. Patients and colleagues can access these profiles to gain insights into the doctor's background and make informed decisions regarding their healthcare needs.

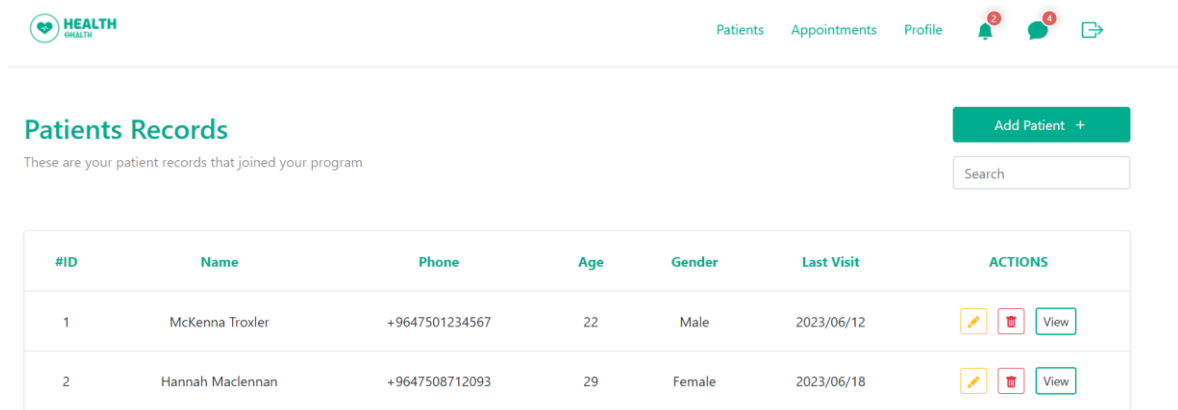


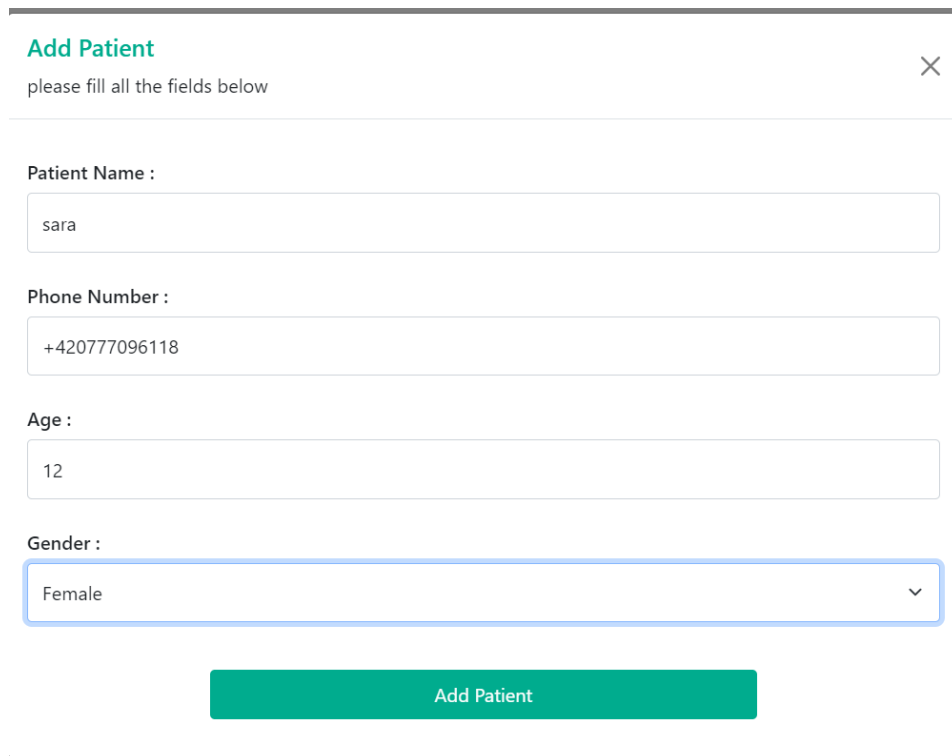
Figure 46: Doctor's Dashboard After Login

```
submitButton.addEventListener("click", function (e) {
  const patientName = patientField.value;
  const age = ageField.value;
  const phone = phoneField.value;
  const gender = genderField.value;

  if (patientName && age && phone) {
    const lastItemId = data.length > 0 ? data[data.length - 1].id : 0;
    const jsonData = {
      id: lastItemId + 1,
      lastVisit: getCurrentDate(),
      patient: patientName,
      age,
      phone,
      gender,
    };
    data.push(jsonData);
    generateTableItem(jsonData, true);
  }
});
```

Figure 47: JS Code for Add Patient

The following code snippet demonstrates the functionality of the submitButton event listener. When the submitButton is clicked, the values entered in the patientField, ageField, phoneField, and genderField are retrieved. If all the required fields (patientName, age, and phone) have been filled, a new item is created with an incremented ID based on the existing data. The newly created item includes details such as the patient's name, age, phone number, and gender, as well as the current date as the last visit. This item is then added to the data array, and the generateTableItem function is called to generate and display a new table row with the item's data.



Add Patient ×

please fill all the fields below

Patient Name :

Phone Number :

Age :

Gender :

Add Patient

Figure 48: Add Patient Field

Figure 48 depicts the user interface component for entering patient information into the website. The form includes various fields for capturing essential details, such as the patient's name, age, phone number, and gender. Upon clicking the submit button, an event listener is triggered, which retrieves the values entered in the respective fields.

6.5 Schedule An Appointment: Doctors Dashboard

The input field displayed in the doctors dashboard is specifically designed for scheduling appointments. In this context, the doctor utilizes the field to input relevant information pertaining to the patient for whom they wish to create an appointment. Once the necessary patient details are entered, the doctor proceeds to initiate the appointment creation process by clicking the "Create an Appointment" button. This action triggers the underlying functionality responsible for generating a new appointment within the system, incorporating the provided patient information. The purpose of this feature is to facilitate efficient appointment management and enable seamless communication between doctors and patients regarding scheduling arrangements.

Schedule an Appointment

please fill all the fields below

Patient :

Jameson Chaknis

Pick a Date :

02/16/2000

Pick a Time :

08:00 --

Additional information

Lactose

Create Appointment

Figure 49: Schedule An Appointment Form

```
function fillTheTable(data) {  
  tableBodyElement.innerHTML = "";  
  data.forEach((e) => {  
    generateTableItem(e);  
  });  
}
```

Figure 50: JS Code For Fill The Table

The function takes a data parameter, which is an array containing the appointment information. It clears the existing contents of the table by setting the innerHTML of the tableBodyElement (presumably a reference to the table body) to an empty string ("").

Then, it iterates over each element (e) in the data array and calls the generateTableItem function, passing in the current element as an argument. This function is responsible for generating the HTML markup for a single table row representing an appointment, and it likely appends this markup to the table body.

In summary, the fillTheTable function is used to refresh the appointment table by clearing its contents and generating new table rows based on the provided appointment data.

In the doctor's appointment management system, the doctor is provided with options to edit, cancel, and update appointments. These options are presented through buttons available for each appointment in the table.

To update an appointment, the doctor can click on the "edit" button, which triggers a modal window or form to appear. Within this modal, the doctor can modify the appointment's time and date according to the desired changes. After making the necessary adjustments, the doctor can submit the form, and the updated appointment details will be saved.

In case the doctor needs to cancel an appointment, they can simply click on the "cancel" button associated with the respective appointment. This action will remove the appointment from the system, effectively canceling it.

On the other hand, if the doctor wants to accept an appointment, they can click on the appropriate button, usually labeled as "accept." This action indicates that the doctor has confirmed their availability and willingness to see the patient at the scheduled time.

Overall, these functionalities provide the doctor with the flexibility to manage appointments efficiently, allowing them to update, cancel, or accept appointments based on their availability and preferences.

6.6 Add Consultations

A crucial functionality available to doctors within the eHealth system is the ability to conduct consultations with their patients. This feature enables doctors to remotely connect with patients in a secure and convenient manner, offering medical advice, diagnosis, treatment recommendations, and ongoing support.

Add Consultation
please fill all the fields below

Problem :
Enter the issue

Medicines :
Enter the medicines separated by comma

Tests :
Enter the tests separated by comma

Condition :
Select Condition

Note

Figure 51: Add Consultation Form

```
submitButton.addEventListener("click", (e) => {
  const problem = problemField.value;
  const tests = testsField.value;
  const medicines = medicinesField.value;
  const condition = conditionField.value;
  const note = noteField.value;

  consultations.unshift({
    problem,
    tests,
    medicines,
    condition,
    note,
    date: getCurrentDate(),
  });

  generateConsoltations(consultations, true);
});
```

Figure 52: JS Code For Add Consultations

The provided code snippet includes an event listener attached to the submitButton element. Upon clicking the submit button, the listener function is executed, facilitating the following functionalities:

1. **Extraction of Input Values:** The code retrieves the entered values from various input fields, namely `problemField`, `testsField`, `medicinesField`, `conditionField`, and `noteField`. These input fields likely pertain to relevant information related to a consultation or medical record.
2. **Creation of Consultation Object:** A new consultation object is constructed, incorporating the extracted values from the input fields, along with the current date obtained from the `getCurrentDate()` function. This ensures that the consultation object includes the necessary details and a timestamp indicating when it was recorded.
3. **Addition of Consultation to Array:** The newly created consultation object is added to the beginning of the consultations array using the `unshift()` method. This allows for the efficient storage and management of multiple consultations, with the most recent one appearing at the top of the array.
4. **Generation of Consultations Display:** The `generateConsoltations()` function is invoked to generate and display the updated list of consultations on the webpage. By passing the argument `true`, the function is instructed to clear the existing consultations before generating the new list, ensuring an accurate representation of the latest data.

6.7 User-Specific Dashboard Patient

Upon successfully logging into their individual accounts using unique credentials, patients are granted access to a personalized dashboard within the eHealth system. This dashboard serves as a central hub for patients, providing them with convenient access to crucial information related to their healthcare. The patient dashboard encompasses two significant components: appointments and medical records.

1. **Appointments:** The patient dashboard prominently displays upcoming appointments scheduled with healthcare providers. This feature allows patients to conveniently view and manage their scheduled medical appointments, ensuring they are well-informed and prepared for their upcoming healthcare visits. By presenting appointment details such as date, time, and healthcare provider information, patients

can easily keep track of their scheduled consultations and plan their healthcare activities accordingly.

2. **Medical Records:** In addition to the appointment management functionality, the patient dashboard provides access to the patient's comprehensive medical records. These records include vital information related to the patient's health history, such as previous diagnoses, prescribed medications, laboratory test results, and relevant treatment plans. By having their medical records readily available on the dashboard, patients can review their health information, track their medical progress, and gain insights into their overall well-being. This feature empowers patients to actively participate in their healthcare decision-making process and enables them to communicate effectively with healthcare professionals regarding their medical history and ongoing treatments.

By offering a dedicated dashboard for patients, the eHealth system aims to enhance the patient experience by providing easy access to vital healthcare information. This personalized and secure platform enables patients to stay informed about their appointments, manage their healthcare schedule, and have a comprehensive overview of their medical records. Ultimately, the patient dashboard serves as a valuable tool in fostering patient engagement, facilitating efficient healthcare communication, and promoting patient-centered care within the eHealth ecosystem.

6.8 Patient Dashboard Appointments

The screenshot shows the 'Appointments' section of a patient dashboard. At the top, there is a navigation bar with the 'HEALTH HEALTH' logo on the left and menu items for 'Appointments', 'Medical Records', 'Doctors', and 'Medicines' in the center. On the right side of the navigation bar, there are notification icons for a bell and a speech bubble, both with red dots indicating unread notifications, and a share icon. Below the navigation bar, the main content area is titled 'Appointments' in a teal color. Underneath the title, it says 'These are your recent scheduled appointments'. To the right of this text, there is a teal button labeled 'Schedule An Appointment' with a calendar icon. Below the button is a search bar with the placeholder text 'Search'. The main content area contains a table with the following columns: '#ID', 'DOCTOR', 'LOCATION', 'DATE', 'TIME', 'STATUS', and 'ACTIONS'. The table has one row of data: '#ID' 1, 'DOCTOR' Dr. Frank Ralat, 'LOCATION' 8 Kensington Road, 'DATE' 2023/06/12, 'TIME' 3:23 AM, 'STATUS' Canceled (in a red box), and 'ACTIONS' with two icons: a pencil (edit) and a trash can (delete).

#ID	DOCTOR	LOCATION	DATE	TIME	STATUS	ACTIONS
1	Dr. Frank Ralat	8 Kensington Road	2023/06/12	3:23 AM	Canceled	

Figure 53: Patients Dashboard

Upon logging into the eHealth website, patients are immediately directed to a dedicated page that serves as their primary point of interaction. This page, designed to streamline the patient experience, focuses primarily on appointments and provides patients with seamless access to their scheduled medical consultations. The intuitive design of this page ensures that

patients can effortlessly navigate through the various functionalities and efficiently manage their healthcare appointments.

The central component of this page is the comprehensive display of the patient's appointments. By presenting a clear and organized overview of upcoming appointments, patients can quickly ascertain the date, time, and healthcare provider associated with each scheduled consultation. This allows patients to effectively plan their healthcare activities and ensure timely attendance at their appointments. Additionally, patients can easily access detailed information about each appointment, such as the purpose or nature of the visit, enabling them to adequately prepare for their medical consultations.

In addition to viewing existing appointments, patients have the convenience of scheduling new appointments directly from this page. Through a user-friendly interface, patients can select the desired date and time for their appointment, as well as choose their preferred healthcare provider. This streamlined scheduling process ensures that patients can secure appointments that best suit their availability and preferences.

Furthermore, patients have the ability to modify or cancel their existing appointments. By simply selecting the respective appointment, patients can easily edit the appointment details, such as rescheduling the date or time. Additionally, patients can opt to cancel an appointment if necessary, providing flexibility and control over their healthcare scheduling.

This patient-focused page within the eHealth website prioritizes the efficient management of appointments and empowers patients to take an active role in their healthcare journey. By providing a seamless user experience and an array of functionalities, patients can effortlessly navigate, schedule, modify, and cancel appointments as needed. This user-centric approach not only enhances patient convenience and flexibility but also fosters effective communication and collaboration between patients and healthcare providers, ultimately contributing to a more patient-centered healthcare experience.

As shown in the Figure The variable `consoltationsListElement` is used to store a reference to the HTML element with the ID "consoltations-list". It is likely that this element represents a container or list where the consultations or medical records of the patient are displayed.

And the following Storing data array in Figure 39 the variable `consoltations` is an array that stores a collection of consultation objects. Each object represents a specific consultation or medical record and contains properties such as date, tests, medicines, problem, condition, and note.

The `consoltations` array serves as a data structure to hold multiple consultations for a patient. Each consultation object within the array represents a specific instance of a consultation, capturing relevant information such as the date of the consultation, the tests conducted, prescribed medicines, the patient's problem or condition, and any additional notes or observations.

By organizing the consultations within an array, it allows for easy retrieval and manipulation of the data. The array can be iterated over to display the consultations in the patient's dashboard, populate a list, or perform other operations such as filtering, sorting, or updating the consultations.

```
const consoltations = [
  {
    date: "12/16/2022",
    tests: "TSH,CBH,WBC,HMB",
    medicines: "AOES 2 doses a day,ALWEI,FALKKS",
    problem: "Alergic",
    condition: "Stable",
    note: "His condition has been updated to a better condition since last time but for the next time we have to check about the conditio
  },
  {
```

Figure 54: JS Code For Consultation

6.9 The CSS Code

After designing the user interface in Figma, the implementation phase involved translating the design into code using CSS. During this process, a deliberate decision was made to adopt a color scheme centered around shades of green, carefully chosen to align with the visual identity and atmosphere associated with hospitals.

The selection of green shades was purposeful, considering the common associations of green with health, well-being, and nature. By incorporating various shades of green, the objective

was to create a visually pleasing and cohesive color palette that resonates with the hospital environment.

Throughout the CSS implementation, special attention was given to maintaining consistency and harmony among the chosen green shades across different elements of the user interface. This consistent use of color not only enhances the overall aesthetic appeal of the website but also establishes a unified visual identity that aligns with the hospital context.

By incorporating the appropriate green shades into the CSS, I aimed to create a visually engaging and cohesive user interface that resonates with the hospital setting, evoking a sense of calmness, healing, and trust.

```
:root {  
  --primary: #00ac8e;  
  --primary-light: rgba(0, 172, 142, 0.1);  
  --primary-light-60: rgba(0, 172, 142, 0.6);  
}
```

Figure 55: The Main Colors Of The Website

In Figure 55, the CSS implementation includes the use of the `:root` selector, which allows me to define and use colors consistently throughout the entire CSS document. This approach ensures that the defined colors can be easily accessed and utilized across various elements of the website.

The three main colors chosen for the website are shades of green: dark green, light green, and light cyan. These colors were thoughtfully selected to align with the desired visual aesthetic and thematic elements associated with the website.

The dark green shade was specifically chosen to provide a sense of depth and richness to certain elements, adding a touch of elegance and sophistication. The light green shade was selected to evoke a feeling of freshness, vibrancy, and vitality, which aligns with the concept of health and well-being. Additionally, the light cyan color was incorporated to introduce a subtle and soothing element to the overall color scheme. This hue complements the green shades, creating a harmonious and visually pleasing combination.

By defining these colors using the `:root` selector, I ensured their availability for use throughout the CSS document, allowing for consistent application and easy maintenance. This approach promotes a unified visual experience across the website and contributes to the overall cohesiveness and professionalism of the design.

6.9.1 Nav Login Button Style

```
.nav-login-btn {  
  padding: 4px 16px !important;  
  background-color: var(--primary);  
  color: white !important;  
  border-radius: 20px;  
  font-size: 16px;  
}
```

Figure 56: Login Button

In Figure 56, the CSS style is applied to the login button in the navbar. The button is customized with specific visual properties to enhance its appearance and ensure it stands out to users.

The color chosen for the button is dark green, which adds a sense of sophistication and elegance to the overall design. This color selection is intended to create a visually appealing contrast against the background and other elements of the navbar.

The text color is set to white, providing a clear and readable contrast against the dark green background. This ensures that the text is easily visible and legible for users.

To give the button a slightly rounded shape, a border radius of 20px is applied. This border radius creates a subtle curvature at the corners of the button, softening its appearance and adding a touch of visual interest.

By combining the dark green background color, white text color, and border radius, the login button in the navbar is visually distinctive and inviting to users. These design choices help to create a cohesive and aesthetically pleasing user interface.

7 DESCRIBE THE SECURITY OPTIONS FOR WEB APPLICATION

The first security option of securing the website is to use HTTPS/SSL. In order to prevent hackers from stealing data, HTTPS uses the SSL/TLS protocol. In addition, SSL/TLS authentication validates the identity of a website server, ensuring its authenticity and verifying that it is indeed the entity it claims to be., preventing impersonations. This prevents numerous types of cyber attacks[30].

The Second security option is the password policy. In order to decide if the new password is valid, a password policy defines the strength rules for a password.

Which is:

3. the minimum numbers of characters in the password can be specified by means of the password strength rule, such as 5,
4. the rule could also provide for the maximum number of characters to be 10[31].

Authorization is a crucial mechanism employed by servers to ascertain whether a client possesses the necessary permissions to utilize a specific resource or gain access to a file. It is commonly intertwined with authentication, which enables the server to verify the identity of the client making the request[32].

By employing authorization protocols, the server evaluates the credentials and privileges of the client to determine if they are allowed to perform the requested action. This process involves comparing the client's authorization credentials, such as roles or permissions, against the access control policies defined by the server[32].

Authentication and authorization work together to enforce security measures and safeguard sensitive information. Authentication validates the client's identity, while authorization regulates the client's actions based on their authenticated identity and associated privileges. This dual approach ensures that only authorized and authenticated clients are granted appropriate access to resources and files, bolstering the overall security of the system[32].

The input validation process shall be used to analyze the inputs and exclude those which are not suitable for use. To fulfill their requirements usually all websites and applications require user inputs in many cases. the main purpose of the input validation is to prevent attackers from entering inputs that are designed to cause harm to the system, by only allowing inputs which meet certain criteria[33].

7.1 HTML INJECTION

is a method employed by attackers to manipulate the content of a web page presented to users. This technique takes advantage of the absence of proper input validation, allowing malicious individuals to transmit HTML-formatted text that alters the displayed content of the website[34].

Web pages often rely on user interactions to generate dynamic content. When web applications fail to validate user input, attackers can exploit this vulnerability by injecting HTML code that modifies the content visible to other users. By crafting specific queries, attackers can gain control over the incorporation of HTML elements, thereby changing how the application's content is presented on the web[34].

In essence, HTML injection enables attackers to manipulate the appearance and behavior of a web page by taking advantage of the lack of input validation. This poses a significant security risk as it allows unauthorized individuals to modify the content viewed by users, potentially leading to misinformation, unauthorized access, or other detrimental consequences. To mitigate HTML injection attacks, it is crucial for web applications to implement rigorous input validation techniques to ensure the integrity and security of the displayed content[34].

Cross-site Scripting (XSS) and HTML injection attacks share a close relationship as they both exploit weaknesses in user input validation. While HTML injection involves defacing a web page by injecting HTML code, XSS takes advantage of the ability to inject JavaScript code into the page. In both cases, the attacks are made possible by inadequate validation of user inputs[34].

HTML injection attack manipulates the content of a web page by inserting HTML elements, which can alter the appearance and structure of the page. On the other hand, XSS attack inserts JavaScript code into the page, enabling attackers to execute arbitrary actions on behalf of the user, such as stealing sensitive information or performing unauthorized operations[34].

Despite the differences in the injected code (HTML vs. JavaScript), both attacks rely on the same fundamental vulnerability: insufficient validation of user input. This vulnerability allows attackers to bypass security measures and inject malicious code that is then executed by the user's browser[34].

7.1.1 How To Protect Website Against HTML Injection

1. To ensure security, it is essential to validate user input on the server side and take precautions against the execution of malicious code. This involves thoroughly examining and validating any data provided by the user before rendering it on the page. Additionally, it is crucial to properly escape all user-supplied data to prevent any unintended code execution[35]
2. Implementing Content Security Policy (CSP) is a recommended security measure that involves defining a set of rules to govern the permissible content sources that can be accessed on a web page. By employing CSP, organizations can effectively mitigate the risk of HTML injection attacks by limiting the authorized sources from which content can be loaded on their websites. This proactive approach to web security helps establish a robust defense mechanism and safeguard against potential vulnerabilities arising from unauthorized content sources[35].
3. Use a reliable library or function to remove harmful HTML code from user input, especially when dealing with user-generated content displayed on a website. The OWASP Java Encoder Project is a popular choice for input sanitization and protection against HTML vulnerabilities. By incorporating these measures, organizations can effectively mitigate the risks of HTML-related attacks and ensure the security of their websites[35].

7.2 JAVASCRIPT INJECTION

During a JavaScript injection attack, malicious code is directly inserted into the client-side JavaScript of a vulnerable website. When the victim accesses the compromised website, their browser executes the injected code. The attacker can use various methods to inject the malicious code:

1. Utilizing the browser's developer console to introduce or modify JavaScript code.
2. Inputting JavaScript code, including SCRIPT elements, into the address bar of the victim's browser.
3. Exploiting cross-site scripting vulnerabilities to insert code into comments or form fields.

Once the injected code is executed, it can perform different malicious actions depending on its intended purpose. These actions may include:

1. Gathering sensitive user information when it is submitted through input fields.
2. Manipulating the victim into carrying out unwanted actions.
3. Eavesdropping on the entirety of user sessions.
4. Manipulating parameters on websites.

JavaScript injection exploits can have significant ramifications, such as:

1. Loss of sensitive information: JavaScript injection vulnerabilities enable attackers to obtain personally identifiable details like names, addresses, and credit card information.
2. Financial repercussions: Successful JavaScript injection attacks can adversely impact user experience, disrupt user traffic, erode user trust, and result in financial losses.
3. Compliance considerations: The exposure of sensitive data due to JavaScript injection can subject organizations to fines and legal consequences imposed by security and compliance frameworks such as PCI-DSS and GDPR[36].

7.2.1 How To Protect Website Against JavaScript Injection

1. **Whitelist:** The implementation of a whitelist entails establishing a predefined set of permissible values for user selection. This approach restricts the options available to end-users, effectively mitigating the risk of code injection in JavaScript. While whitelisting is considered a highly secure method, its practicality may be limited due to the necessity of predefining and maintaining an exhaustive list of acceptable values[37]
2. **Input Validation:** Offers greater flexibility in terms of user options compared to whitelisting. It involves identifying and disallowing specific characters or patterns that could potentially lead to code injection. While whitelisting is a more rigorous approach for safeguarding web applications compared to blacklisting, it is frequently the preferred and necessary choice[37].
3. **WAF:** The web application firewall (WAF) is a highly effective safeguard for web applications, serving as a preventive measure against malicious attacks and preserving the confidentiality of sensitive information for end users. Incorporating regular configuration of the WAF is strongly advised, as it can mitigate potential challenges and complications in the long term[37].

7.3 SQL INJECTION

SQL injection (SQLi) is a type of web security flaw that enables attackers to manipulate the database queries performed by an application. This vulnerability grants unauthorized access to data that would typically be inaccessible to the attacker[38].

1. SQL injection is a commonly used technique for attacking websites[39]
2. SQL injection involves maliciously inserting code into SQL statements through input received from a web page[39]

7.3.1 How To Protect A Website Against SQL Injection

3. Verify the user input: A fundamental principle in safeguarding websites against malicious attacks is to adopt a cautious approach towards user input, refraining from placing blind trust in its authenticity. Despite the presence of authentication mechanisms, thorough validation of user inputs is imperative. This validation applies not only to conventional input fields like text areas and text boxes but also extends to file uploads, checkboxes, hidden inputs, and other input components[40]
4. Input validation and parameterized queries: Which encompass prepared statements. These security practices have proven to be the most reliable in mitigating the risk of SQL Injection[40].
5. Employing an automated SQL injection scanner is an additional and uncomplicated approach to identify and mitigate SQL injection vulnerabilities[40].

In conclusion, ensuring the security of web applications is of utmost importance to protect against HTML, JavaScript, and SQL injections. By implementing robust security measures such as input validation, parameterized queries, and proper sanitization of user input, the risk of these types of attacks can be significantly reduced. Additionally, leveraging techniques such as content security policies, whitelisting, and utilizing web application firewalls further enhances the overall security posture. Regular security audits, vulnerability assessments, and staying updated with the latest security best practices are essential for maintaining a secure web environment. By prioritizing security measures and adopting a proactive approach, organizations can effectively mitigate the risks associated with HTML, JavaScript, and SQL injections, safeguard sensitive data, and ensure the integrity and reliability of their web applications.

CONCLUSION

In conclusion, the development of a successful eHealth website is a complex undertaking that requires careful planning, meticulous attention to user requirements, and the effective utilization of relevant technologies. This study has delved into the multifaceted nature of eHealth websites, highlighting the paramount importance of providing high-quality content and creating a user-friendly design. By employing prototyping techniques, implementing robust security measures, and harnessing a diverse range of technologies such as HTML, CSS, JavaScript, and Bootstrap, developers can create a well-designed and fully functional eHealth website.

The research conducted in this study also underscores the significance of intuitive navigation as a critical component of the user experience. By ensuring that users can easily navigate through the website and locate the desired information or services, the risk of user abandonment is mitigated, fostering a positive and engaging user experience.

Furthermore, the study emphasizes the importance of data security in the context of eHealth websites. With the sensitive nature of healthcare information, it is imperative to prioritize the implementation of robust security measures to safeguard user data and maintain confidentiality. This can be achieved through techniques such as encryption, access control mechanisms, and regular security audits.

By bringing together the elements of user-centric design, seamless functionality, and stringent security measures, eHealth websites can effectively bridge the gap between healthcare services and the digital realm. The ultimate goal is to provide efficient and accessible healthcare services to users, enhancing convenience and facilitating remote access to medical resources.

It is assumed that the findings and insights presented in this study will contribute to the continuous evolution of eHealth websites, fostering their ongoing improvement and adaptation to meet the ever-changing needs and expectations of users. By prioritizing user satisfaction, data security, and technological advancements, eHealth websites can play a pivotal role in transforming and revolutionizing healthcare delivery, making healthcare services more accessible, efficient, and user-centric in the digital era.

BIBLIOGRAPHY

- [1] [’ARVIND SINGHAL’, ’Martin Cowie’]. What is e-Health? *What is e-Health?* [online]. Available from: <https://www.escardio.org/Journals/E-Journal-of-Cardiology-Practice/Volume-18/what-is-e-health>
- [2] What is an electronic health record (EHR)? | HealthIT.gov. *What is an electronic health record (EHR)?* | *HealthIT.gov* [online]. 10 September 2019. Available from: <https://www.healthit.gov/faq/what-electronic-health-record-ehr>
- [3] Benefits of Telemedicine. *Benefits of Telemedicine* | *Johns Hopkins Medicine* [online]. 18 January 2022. Available from: <https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/benefits-of-telemedicine>
- [4] mHealth Technology? What is the Definition of Mobile Health. *eVisit* [online]. Available from: <https://evisit.com/resources/what-is-mhealth>
- [5] postDICOM. 10 Advantages and Disadvantages of EHR [EHR vs EMR Differences] | PostDICOM. *postDICOM* [online]. Available from: <https://www.postdicom.com/en/blog/advantages-and-disadvantages-of-ehr>
- [6] Advantages and Disadvantages of EHRs. *Wheel* [online]. Available from: <https://www.wheel.com/companies-blog/advantages-and-disadvantages-of-ehrs>
- [7] Benefits Of Telemedicine | 2023 Advantages & Disadvantages. *Benefits Of Telemedicine* | *2023 Advantages & Disadvantages* [online]. 1 May 2023. Available from: <https://www.selecthub.com/telemedicine/telemedicine-benefits/#4>
- [8] RESNICK, Richard. What are the Pros and Cons of mHealth? *What are the Pros and Cons of mHealth?* [online]. 17 July 2019. Available from: <https://blog.cureatr.com/pros-and-cons-mhealth>
- [9] The Benefits and Challenges of a Health Information Exchange. *Tangible Solutions* [online]. 15 July 2021. Available from: <https://www.tangible.com/blog/data-sharing/the-benefits-and-challenges-of-a-health-information-exchange/>
- [10] What is HTML? Definition of Hypertext Markup Language - javatpoint. *www.javatpoint.com* [online]. Available from: <https://www.javatpoint.com/what-is-html>
- [11] SINGHAL, Piyush. What Are the Advantages of HTML?- Scaler Topics. *Scaler Topics* [online]. 17 October 2022. Available from: <https://www.scaler.com/topics/advantages-of-html/>
- [12] Advantages and Disadvantages of HTML - GeeksforGeeks. *GeeksforGeeks* [online]. 25 October 2020. Available from: <https://www.geeksforgeeks.org/advantages-and-disadvantages-of-html/>
- [13] Unstop - Competitions, Quizzes, Hackathons, Scholarships and Internships for Students and Corporates. *Unstop - Competitions, Quizzes, Hackathons, Scholarships and Internships for Students and Corporates* [online]. Available from: <https://unstop.com/blog/advantages-and-disadvantages-of-html>
- [14] 1. USER, Devmountain. What Is CSS and Why Should You Use It? *Devmountain* [online]. 22 April 2019. Available from: <https://devmountain.com/blog/what-is-css-and-why-use-it/>
- [15] BRADLEY, Steven. The Benefits Of Cascading Style Sheets - Vanseo Design. *Vanseo Design* [online]. 15 March 2006. Available from: <https://vanseodesign.com/css/benefits-of-cascading-style-sheets/>
- [16] Unstop - Competitions, Quizzes, Hackathons, Scholarships and Internships for Students and Corporates. *Unstop - Competitions, Quizzes, Hackathons,*

- Scholarships and Internships for Students and Corporates* [online]. Available from: <https://unstop.com/blog/advantages-and-disadvantages-of-css>
- [17] What is JavaScript? - Learn web development | MDN. *What is JavaScript? - Learn web development | MDN* [online]. Available from: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript
- [18] admin. JavaScript advantages and disadvantages. *W3schools* [online]. 24 April 2018. Available from: <https://www.w3schools.blog/advantages-disadvantages-javascript>
- [19] Advantages and Disadvantages of JavaScript. *Advantages and Disadvantages of JavaScript* [online]. Available from: <https://www.tutorialspoint.com/advantages-and-disadvantages-of-javascript>
- [20] Benefits of Using Bootstrap for Web Design. *Benefits of Using Bootstrap for Web Design* [online]. 6 May 2020. Available from: <https://www.clarity-ventures.com/blog/benefits-of-using-bootstrap-for-web-design>
- [21] What is Bootstrap? Pros and Cons Of This Framework. *Hackr.io* [online]. Available from: <https://hackr.io/blog/what-is-bootstrap-framework>
- [22] What is Unified Modeling Language (UML)? *What is Unified Modeling Language (UML)?* [online]. Available from: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>
- [23] What are Functional Requirements: Examples, Definition, Complete Guide - Visure Solutions. *Visure Solutions* [online]. Available from: <http://visuresolutions.com/blog/functional-requirements/>
- [24] ROME, Paula. What are Non Functional Requirements — With Examples | Perforce Software. *Perforce Software* [online]. Available from: <https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples>
- [25] Use-case diagrams in UML modeling. *Use-case diagrams in UML modeling* [online]. Available from: <https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case>
- [26] T, Neha. What is Class Model? Objects, Class, Relations - Binary Terms. *Binary Terms* [online]. 28 May 2020. Available from: <https://binaryterms.com/class-model.html>
- [27] IBM Developer. *IBM Developer* [online]. Available from: <https://developer.ibm.com/articles/the-sequence-diagram/>
- [28] What is Wireframe? *What is Wireframe?* [online]. Available from: https://www.visual-paradigm.com/support/documents/vpuserguide/2822/2613/83691_whatswirefr.html
- [29] What is Figma? A Design Crash Course [2021 Tutorial]. *freeCodeCamp.org* [online]. 21 June 2021. Available from: <https://www.freecodecamp.org/news/figma-crash-course/>
- [30] What is HTTPS? - SSL.com. *SSL.com* [online]. 12 October 2021. Available from: <https://www.ssl.com/faqs/what-is-https/>
- [31] Password policies. *Password policies* [online]. Available from: <https://www.ibm.com/docs/en/spim/2.0.0?topic=administration-password-policies>
- [32] Understanding Authentication, Authorization, and Encryption : TechWeb : Boston University. *Understanding Authentication, Authorization, and Encryption : TechWeb : Boston University* [online]. Available from: <https://www.bu.edu/tech/about/security->

- resources/bestpractice/auth/#:~:text=Authorization%20is%20a%20process%20by,i
s%20that%20is%20requesting%20access.
- [33] What Is Input Validation and Why Is It Important? *MUO* [online]. 31 January 2023. Available from: <https://www.makeuseof.com/what-is-input-validation/>
- [34] HTML Injection | Imperva. *Learning Center* [online]. Available from: <https://www.imperva.com/learn/application-security/html-injection/>
- [35] pandaquests. Defend Your Website from HTML Injection: A Comprehensive Guide. *Medium* [online]. 6 February 2023. Available from: <https://pandaquests.medium.com/defend-your-website-from-html-injection-a-comprehensive-guide-60764c35f9dc>
- [36] **【Javascript Injection】** Definition, Examples, and Prevention. *Crashtest Security* [online]. 4 October 2022. Available from: <https://crashtest-security.com/js-injection-attack/>
- [37] admin. Code Injection in Javascript: Prevention and Remediation. *Bright Security* [online]. 1 March 2022. Available from: <https://brightsec.com/blog/code-injection-in-javascript-prevention-and-remediation/>
- [38] SQL Injection | OWASP Foundation. *SQL Injection | OWASP Foundation* [online]. Available from: https://owasp.org/www-community/attacks/SQL_Injection
- [39] SQL Injection. *SQL Injection* [online]. Available from: https://www.w3schools.com/sql/sql_injection.asp
- [40] Protect your application against SQL Injection - Bright Security. *Bright Security* [online]. Available from: <https://brightsec.com/protect-your-application-against-sql-injection/>

LIST OF ABBREVIATIONS

CSP - Content Security Policy

CSS - Cascading Style Sheets

EHR - Electronic Health Records

eHealth - Electronic Health

GDPR - General Data Protection Regulation

HIE - Health Information Exchange

Hi-Fi - High Fidelity

HTML - Hypertext Markup Language

HTTPS - Hypertext Transfer Protocol Secure

Lo-Fi - Low Fidelity

mHealth - Mobile Health

OWASP - Open Web Application Security Project

PCI-DSS - Payment Card Industry Data Security Standard

SEO - Search Engine Optimization

SQL - Structured Query Language

SQLi - Structured Query Language Injection

SSL - Secure Sockets Layer

TLS - Transport Layer Security

UML - Unified Modeling Language

WAF - Web Application Firewall

XSS - Cross-Site Scripting

XML - Extensible Markup Language

LIST OF FIGURES

Figure 1: Doctor's Management Functional Requirements 1/2	25
Figure 2: Doctor's Management Functional Requirements 2/2	26
Figure 3: Patients Functional Requirements	26
Figure 4: Non-Functional Requirement	27
Figure 5: Use Case Diagram	28
Figure 6: Class Model	37
Figure 7: Create Account Sequence Diagram	38
Figure 8: Login Sequence Diagram	39
Figure 9: Create Appointment Sequence Diagram	40
Figure 10: View Appointment Sequence Diagram	42
Figure 11: Update Appointment Sequence Diagram	43
Figure 12: Cancel Appointment Sequence Diagram	45
Figure 13: Medical Record Sequence Diagram	46
Figure 14: Create Prescription Sequence Diagram	48
Figure 15: View Prescription Sequence Diagram	49
Figure 16: Update Prescription Sequence Diagram	50
Figure 17: Add Patient Sequence Diagram	52
Figure 18: Delete Patient Sequence Diagram	53
Figure 19: Update Patient Sequence Diagram	54
Figure 20: Hero Wireframe	55
Figure 21: Service Section Wireframe	56
Figure 22: Doctor's Page Wireframe	57
Figure 23: Doctors Page After Log In	58
Figure 24: Lo-Fi Hero Section Wireframe	60
Figure 25: Lo-Fi Service Section Wireframe	61
Figure 26: Lo-Fi Footer Section Wireframe	62
Figure 27: Lo-Fi Login Page Wireframe	62
Figure 28: Lo-Fi Sign Up Wireframe	63
Figure 29: Lo-Fi Hospitals Page Wireframe	64
Figure 30: Lo-Fi Doctor's Page Wireframe	65
Figure 31: Lo-Fi Medicines Page Wireframe	66
Figure 32: Hi-Fi Hero Section Wireframe	67
Figure 33: Hi-Fi Service Section Wireframe	68
Figure 34: Hi-Fi Footer Section Wireframe	69

Figure 35: Hi-Fi Doctors Login Wireframe.....	70
Figure: 36 Hi-Fi Doctors Page Wireframe	71
Figure 37: Hi-Fi Hospitals Page Wireframe	72
Figure 38: Hi-Fi Medicines Page Wireframe	73
Figure 39: JS Code For Navigation Scroll.....	75
Figure 40: Home Page 1/2	75
Figure 41: Home Page 2/2	76
Figure 42: Doctors Log In	76
Figure 43: JS Code for Doctor's Login	77
Figure 44: Login Page.....	78
Figure 45: JS Code For Patient Information.....	78
Figure 46: Doctor's Dashboard After Login	79
Figure 47: JS Code for Add Patient	80
Figure 48: Add Patient Field.....	81
Figure 49: Schedule An Appointment Form.....	82
Figure 50: JS Code For Fill The Table	82
Figure 51: Add Consultation Form.....	84
Figure 52: JS Code For Add Consultations	84
Figure 53: Patients Dashboard.....	86
Figure 54: JS Code For Consultation.....	88
Figure 55: The Main Colors Of The Website	89
Figure 56: Login Button	90

APPENDICES

Appendix P I: fulltext.PDF

Appendix P II: Website.zip

APPENDIX P I: APPENDIX TITLE

Website.zip

CD