DocFinder

(Web Scrapping tool to scrap for Doctors Data)

FINAL REPORT

CMSE 322

PROJECT NO: Web scraper

GROUP NO: 5

PROJECT NAME: DocFinder

PROJECT START DATE: 08-04-2023

PROJECT END DATE: 04-06-2023

SUPERVISOR: Assist.Prof.Dr Duygu Çılık

SEMESTER TERM: Spring 2023

Computer Engineering Department

Eastern Mediterranean University

Project Type: Software Design & Development Project
Template updated: 20.11.2017

ABSTRACT

The comprehensive report for the DocFinder app encompasses all stages of the software development cycle, ensuring a holistic understanding of the scraping process and its outcomes. It begins with a meticulous planning phase, where project objectives, scope, and requirements are defined. The analysis stage involves in-depth exploration of various data sources and the formulation of a robust strategy for extracting doctors' information. Moving to the design phase, the report provides a detailed overview of the system's architecture, including the relationships between its components and the integration of necessary functionalities. The development phase is thoroughly covered, discussing the selection of tools, frameworks, and programming languages, along with code snippets and examples to demonstrate the implementation process. The report also highlights the importance of rigorous testing, describing the strategies and methodologies employed to validate the scraped data and ensure the system's reliability. By condensing all stages of the software development cycle into a single report, readers gain valuable insights into the systematic approach taken to plan, analyze, design, develop, and test the DocFinder app, resulting in a comprehensive overview of the scraping process and its significant outcomes.

Keywords: Web scraping, doctors, clinics, HTML, python, nodeJS, website

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

DOCFINDER is a Tool to Collect Doctors' Data on the Web system is a website that allows patients to easily find and interact with doctors. Patients may search for doctors based on factors such as geography and specialty and examine provider profiles to make educated healthcare decisions. Additionally, the system enables providers to build profiles and update their information, assisting them in attracting new patients and maintaining their internet presence. The system can benefit healthcare organizations by offering an easy-to-use platform for people to identify and interact with doctors, thereby boosting the number of patients such organizations serve. Because the system can be accessed from any device with an internet connection, it is appropriate for use by healthcare organizations of any size. Overall, the Online Doctor Searching system seeks to improve the healthcare experience for patients and doctors alike by providing a convenient and efficient way for patients to find and connect with doctors, as well as useful tools for doctors to promote their services and communicate with patients.

2. PROJECT PLANNING AND MANAGEMENT

A.1. Preliminary Project Information

A.1.1

Project No	1
Project Name	Web Scraping Tool to Collect Doctors’ information
Start Date	03/15/2023
End Date	05/31/2023
Time	56 days

A.1.2

Project Manager			
Name Surname	Hassan El Abdallah	**ID No**	18700656
Title/Role	Project manager / Backend developer / Lead programmer		
Address	Famagusta		
Phone	+90 533 888 20 29		
Email	18700656@emu.edu.tr		

A.2 Group Information

A.2.1

Student 1			
Name Surname	Ghaleb Metal	**ID No**	20801474
Title/Role	Frontend developer / User interface designer		
Address	Famagusta		
Phone	+90 533 888 20 29		
Email	20801474@emu.edu.tr		

Student 2			
Name Surname	Khawlah Alshubati	**ID No**	19701557

Title/Role	Database Manager / Administrator
Address	Famagusta
Phone	+90 533 888 20 29
Email	19701557@emu.edu.tr

Student 2			
Name Surname	Abdulaziz Binafif	**ID No**	19701169
Title/Role	Tester / System analyst		
Address	Famagusta		
Phone	+90 533 888 20 29		
Email	19701169@emu.edu.tr		

A.2.2

List of Completed / Ongoing Projects of Team
Web Scraping Tool to Collect Doctors' information

B.1 Introduction to Project

B.1.1

Summary of Project
A web-based system to extract and collect data about physicians from various health sites/news/blogs. The data to be collected may include physicians' names, contact information, the physician's area of expertise (including main / sub medical specialties), comments/ratings made by his patients with its date info, diseases he has previously diagnosed/treated, institutions/hospitals/clinics he works/worked for information and any other relevant information available on the website or database, to make it easy for patient to search about what they need.

B.1.2

Key Words
Python, HTML, web, data, crawler, scraping, doctors, patients

B.1.3

Aim of Project
The aim of the web scraping project for collecting doctors' information is to build a comprehensive database of doctors and their professional information by extracting and collecting data from various medical websites and databases, so it will be easy for patients to search for what they want by filtering their search to have better results.

B.1.4

Innovative Aspects/Contributions of Project
This system is a web-based and it's going to make it easier for patients to find or look up for doctors who meet their expectations, by applying some methodologies and we are planning to design it to be scalable, which means it can handle large volumes of data from multiple sources. This makes it possible to collect data about a large number of doctors from different medical websites and databases.

B.1.5

Methods to be Applied

In order to plan and analyze the software, we will meet with users and subject-matter experts, interview them, search about the field and analysis it, and assess comparable systems that are presently available on the market. With MS Project, all essential scheduling will be recorded.

In order to implement the essential diagrams that will guide the development team throughout the coding stage, we will use tools like Modelio and draw.io for the design step.

To implement the necessary functions and needs, we will employ a variety of programming languages, including HTML, CSS, JavaScript, and Python. The database system that will maintain the user records will also need certain SQL tools, such as MySQL.

For the testing stage which will be implemented concurrently with the coding stage we will use well-known testing tools, to reach and cover more test cases in a more time efficient way. We will also take into account unusual instances found in the field that can have disastrous consequences for the application.

B.1.6

Economic and National Outcomes

This system will improve the healthcare services by availability of accurate and up-to-date data about doctors can help to improve the quality of healthcare services. With better access to information about doctors' information's, healthcare providers can make more informed decisions about patient care, also Improve the patients' outcomes because with better access to information about doctors, patients can make more informed decisions about their healthcare. Patients can choose doctors based on their specialties, education, and experience, which can lead to better health outcomes.

B.2 Reason of Starting the Project, Methods and R&D Stages

B.2.1

1- Explain the reason of starting this project. (Max 500 charachter)

Nowadays you can nearly find everything online, so we got the idea of supporting healthcare providers and patients. With better access to information about doctors' specialties, education, and experience, healthcare providers and patients can make more informed decisions about patient care. This can lead to better health outcomes and improve the quality of healthcare services. Furthermore, the information gathered can be utilized for marketing purposes. The data may be used by medical firms to identify new clients and generate tailored marketing efforts.

2- Explain the purpose of this project.

The main goal of this project is to collect accurate and up-to-date data on physicians from various medical websites and databases to support healthcare services for example patients can search for doctors anytime and looking for specific thing like reviews, prices, experiences and other more, also this data may then be utilized for a variety of reasons, including medical research, marketing, and analysis.

3- Explain

- **output of project**
- **national / international standards if exist**
- **the specific objectives of the project**
- **success criterias**
- **realistic constraints**

For the output of this project, we hope to have achieved our goal of creating an aesthetically pleasing and functional website. To do this, our website must allow patients to search for any type of doctor across many fields. Also, in order to reach a larger number of people, our website would be multilingual.

Success Criteria: -

Data Accuracy: The project's success is dependent on the correctness of the data collected. The information must be full, up to date, and error-free.

Data Coverage: The collected data should include a considerable fraction of the target area's doctors. The more comprehensive the coverage, the more important the data for medical research, analysis, and marketing.

Data Reliability: Data should be acquired on time and updated on a frequent basis to maintain its relevance and correctness. If data is not updated on a regular basis, it may become obsolete and irrelevant.

Data Security: The information gathered should be kept safe and secure against unwanted access or theft. This is especially critical when gathering sensitive information like medical records.

4- Explain

- **the methods to be applied during R&D activities**
- **applications**
- **technics and tools to be used**
- **standards to be followed under the workflow**

Which SOFTWARE PROCESS MODEL in below will you apply? Why? How? Explain.

Our software will use an agile workflow, in which we will create a somewhat primitive website, but as users see it and provide feedback and requests, the website will be improved with new features that will be added on a timely basis, as well as the fact that this workflow breaks down tasks into smaller tasks, which aids in the discovery of problems and risks.

Explain, Project Workflow:

1. **Feasibility and Pre-research:**

At this stage of the project; several types of research will be conducted, as well as information gathering, to ensure the project's success. By evaluating comparable systems to reduce our system's faults, a system with better and more intuitive characteristics may be established.

2. **System Design:**

In this stage by using the appropriate drawing and designing tools, decisions will be made on system components such as modules, algorithms, and approaches, as well as diagrams such as use case diagrams, ER diagrams, and more. Moreover, the client and developers will work together to create the user design through several prototype iterations.

Working collaboratively throughout the design stage helps us to meet their demands and satisfy them. We generate designs, which the customer reviews, and then we meet to discuss any technical concerns.

3. **Software development:**

The system will be built using the JavaScript as the primary programming language. Along with HTML, CSS, and JavaScript contributing as design languages. Moreover;

our data will be stored in a mySQL database.

4. **Prototype implementation and testing work:**

When we have developed the code for our website, we will run rigorous tests on a local server that will be executing our code to find flaws early on. We will make the website publicly available after it has shown to be sufficiently resilient.

5. **Maintenance:**

In this stage because we are using Agile maintenance is very important to reduce risks and avoid future problems, also by Agile product backlogs, user

stories, task prioritization, coding standards, pair programming, refactoring, daily builds, continuous integration, code reviews, bug tracking, small releases, unit testing, and acceptance testing work to ensure that the final product is efficient and qualified enough to

perform its tasks flawlessly without any risks or failures.

5- Explain

- **the contribution of national/international technological development if exist**
- **starting a new research and development projects within or outside the team**
- **launch new applications or research studies in different technology areas**

With whom we can cooperate?

Expectations:

Published work:

Can your output be an input for other similar national/international projects?

After meeting the stakeholders and knowing what are their ideas, we must provide our requirements structure to clients so that they may determine whether it meets their demands. If required, we will modify the specifications until the client is pleased. Throughout the data collection process from various websites, we will extract the data and develop many databases in which the data will be classified and incorporated into other worldwide initiatives. The doctors listed on our website will be able to confirm appointments.

We will construct our website using comparable current systems as a guide and attempt to eliminate any flaws in the modules that we develop by employing comparable existing systems.

B.3 Innovative and Unique Aspects

B.3.1

1- Describe

- **differences**
- **advantages**
- **superiority**
- **compared to other similar projects.**

- In our project we aim to develop a website that focuses exclusively on collecting information related to medical professionals which is able to extract more detailed and comprehensive information about physicians compared to other similar projects.
- Our project will focus on consistency of data where it can deliver updated data.
- Our website will specialize in physicians' data so it can easily gather larger amount of information while saving time and effort compared to searching manually through other websites to find physicians information.
- Our project will ensure higher levels of data accuracy and completeness than other projects as well as providing real-time access to data which can be important for emergency cases, healthcare research or medical customers.
- Compared to other projects, our project will have a user-friendly interface and navigation capabilities which will ensure the easiness and simplicity for users when searching.
- Most importantly, we will ensure that the performance of our website is as efficient as possible where response and load time are high.

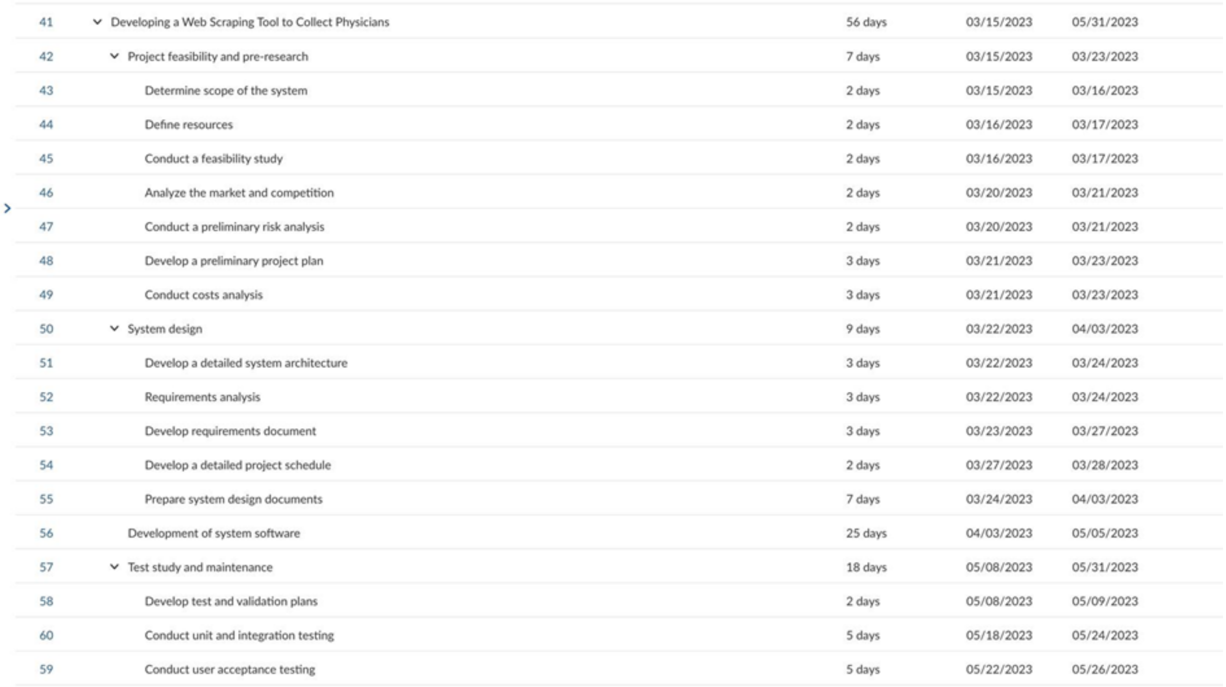
B.4.1

2- Who can contribute to this project in your team?

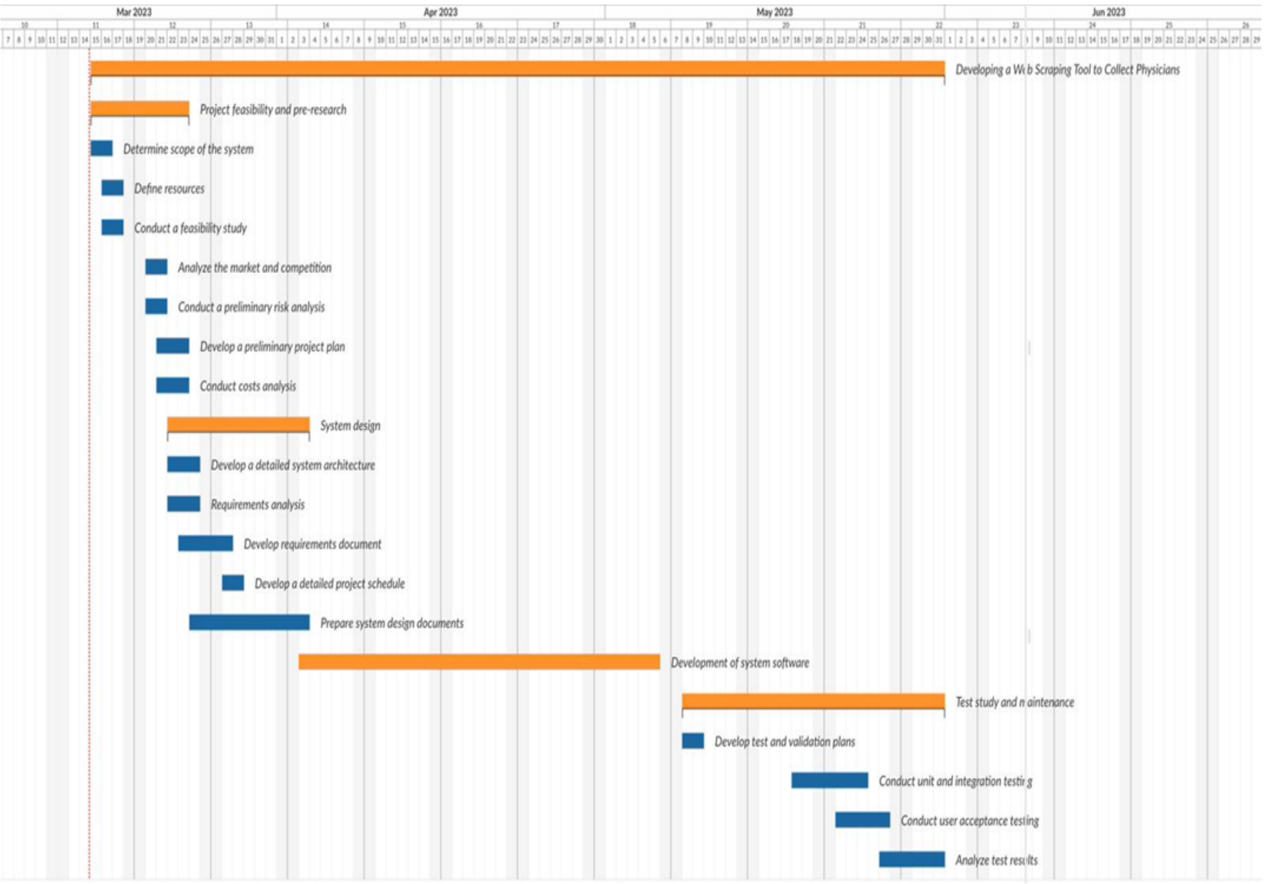
- Project Manager.
- Database Administrator.
- Frontend Developer.
- Backend Developer.
- User Interface Designer.
- System Analyst.
- Tester/ QA Engineer.

C.1 Gantt Chart and Work Packages

C.1.1 Gantt Chart



41	Developing a Web Scraping Tool to Collect Physicians	56 days	03/15/2023	05/31/2023
42	Project feasibility and pre-research	7 days	03/15/2023	03/23/2023
43	Determine scope of the system	2 days	03/15/2023	03/16/2023
44	Define resources	2 days	03/16/2023	03/17/2023
45	Conduct a feasibility study	2 days	03/16/2023	03/17/2023
46	Analyze the market and competition	2 days	03/20/2023	03/21/2023
47	Conduct a preliminary risk analysis	2 days	03/20/2023	03/21/2023
48	Develop a preliminary project plan	3 days	03/21/2023	03/23/2023
49	Conduct costs analysis	3 days	03/21/2023	03/23/2023
50	System design	9 days	03/22/2023	04/03/2023
51	Develop a detailed system architecture	3 days	03/22/2023	03/24/2023
52	Requirements analysis	3 days	03/22/2023	03/24/2023
53	Develop requirements document	3 days	03/23/2023	03/27/2023
54	Develop a detailed project schedule	2 days	03/27/2023	03/28/2023
55	Prepare system design documents	7 days	03/24/2023	04/03/2023
56	Development of system software	25 days	04/03/2023	05/05/2023
57	Test study and maintenance	18 days	05/08/2023	05/31/2023
58	Develop test and validation plans	2 days	05/08/2023	05/09/2023
60	Conduct unit and integration testing	5 days	05/18/2023	05/24/2023
59	Conduct user acceptance testing	5 days	05/22/2023	05/26/2023



C.1.2 List of Work Packages

Work Package No	1
Work Package Name	**Project Feasibility and Pre-Research (Feasibility Analysis)**
Start-End Date and Time	03.15.2023 – 03.23.2023
Related Organizations	
1- List the activities of work packages.	
1.1 Project Process and Economic Feasibility: 1. Determine scone of the system 2. Conduct a feasibility study 3. Analyze the market and competition 4. Conduct a preliminary risk analysis 5. Develop a preliminary project plan **6.** Conduct costs analysis 1.2 Technological Feasibility: Define resources	
2- Describe the methods and parameters that will be used for work package.	
1. Market Research: Analyzing market trends, customer needs and preferences, competitor analysis, and other relevant data. 2. Technical Feasibility: The technical feasibility of the project will be determined by assessing the availability of resources, technology, and expertise required to complete the project successfully.	

3. Financial Feasibility: A thorough financial analysis will be conducted to determine the cost of the project and the potential revenue it can generate. This will include an assessment of the initial investment, operational costs, and expected returns on investment.

Legal and Regulatory Feasibility: Compliance with legal and regulatory requirements is essential for any project. Therefore, a review of applicable laws and regulations will be conducted to ensure that the project complies with all relevant standards and guidelines.

3- List the experiments, tests and analysis in the work package.

- Data analysis
- User research
- Market analysis
- Cost-benefit analysis

4- List the output of work package and its success criteria.

Outputs:

1. Summarizing the findings of the data analysis conducted during the work package, including any trends, patterns, or insights identified.
2. Summarizing the findings of the user research conducted during the work package, including user needs, preferences, and pain points identified.

Success Criteria:

3. Meeting project deliverables on time, within budget, and to the required quality standards.
4. Achieving a certain level of user adoption or customer satisfaction with the final product.

5- Explain the relation of output with other work packages

The success of other work packages within the project is dependent on the completion of this work package, as it serves as a fundamental building block or cornerstone.

Work Package No	2
Work Package Name	**Based System Design Technology (Analysis & Design stage)**
Start-End Date and Time	03.22.2023 – 04.03.2023
Related Organizations	

1- List the activities of work packages.

1. Develop a detailed system architecture
2. Requirements analysis
3. Develop requirements document
4. Develop a detailed project schedule
5. Prepare system design documents

2- Describe the methods and parameters that will be used for work package.

1. Functional and Nonfunctional requirements analysis
2. System architecture design
3. Applications of UML modeling
4. User interface design

3- List the experiments, tests and analysis in the work package.

1. Prototyping
2. Create requirements document
3. Documentation plan

4. System integration plan

4- List the output of work package and its success criterias.

Outputs:

1. Requirements document (SRS).
2. System architecture diagram.
3. Component specifications.
4. Technical documentation.
5. User interface design.

Success Criteria:

Meeting the functional and non-functional requirements: The proposed system design should meet all of the functional and non-functional requirements identified in previous work packages

5- Explain the relation of output with other work packages

The output of the system design work package serves as a critical input for subsequent work packages in the project plan. The system design document provides a detailed blueprint for the development and implementation of the proposed solution, which serves as a guide for subsequent work packages

Work Package No	3
Work Package Name	**Development of System Software (Development Stage)**

Start-End Date and Time	04.03.2023 – 04.03.2023
Related Organizations	

1- List the activities of work packages.

1. Database design.
2. Coding .
3. Implement backend functions.
4. Implement frontend design.

2- Describe the methods and parameters that will be used for work package.

1. Creating a relational database using **MySql**
2. Creating a crawler/scraper to scrap other websites using **Python**
3. Creating backend using **NodeJS**
5. Creating frontend using **HTML-CSS-JS**

3- List the experiments, tests and analysis in the work package.

1. Debugging
2. Requirements testing
3. Develop database
4. Review functional specifications

4- List the output of work package and its success criteria.

Outputs:

1. Sample scraper ready to scrap data from other websites
2. A database ready to read and write data
3. Running sample website to display data scraped to the user

Success Criteria:

4. Performance and scalability
5. Security and privacy

6. Maintainability and scalability
5. Cost-effectiveness

5- Explain the relation of output with other work packages

We would be able to start testing the system after the output of this work package.

Work Package No	4
Work Package Name	**Prototype Implementation and Test Study and Maintenance (Test & Maintenance stage)**
Start-End Date and Time	05.08.2023 – 05.31.2023
Related Organizations	

1- List the activities of work packages.

1. Develop test and validation plans.
2. Conduct unit and integration testing.
3. Conduct user acceptance testing.
4. Analyze test results.

2- Describe the methods and parameters that will be used for work package.

1. Testing methods: Various testing methods will be used to ensure that the software system meets all functional and non-functional requirements. These testing methods include unit testing, integration testing, system testing, and acceptance testing.
2. Testing tools: Automated testing tools will be used to perform the various types of testing required for the software system.

5. Defect reporting and tracking: A defect reporting and tracking system will be used to log and track defects and issues identified during testing.

3- List the experiments, tests and analysis in the work package.

1. Unit testing
2. Integration testing
3. Acceptance testing
4. Usability testing
6. Maintenance analysis

4- List the output of work package and its success criterias.

Outputs:

1. Test reports
2. Defects reports
3. Updated software system

Success Criteria:

1. User acceptance
7. Minimal defects

5- Explain the relation of output with other work packages

Last work package. Project will be ready to be released.

C.1.3 List of Milestones (should be matched in the Gantt chart)

	Description of Output	Expected Time Interval
1	Conduct a feasibility study	03/16/2023 - 03/17/2023
2	Develop a preliminary project plan	03/21/2023 - 03/23/2023
3	Requirements analysis	03/22/2023 - 03/24/2023
4	Prepare system design documents	03/24/2023 - 04/03/2023
5	Database development	04/03/2023 - 04/10/2023
6	Website development	04/20/2023 - 05/05/2023
7	Testing and modifications	05/08/2023 - 05/31/2023
8	Project closure	06/01/2023

C.1.4 List of Risks (see following example, find other risks of your Project!)

Risk	Probability	Effects	Your Strategy
Web scraping can potentially infringe on copyright laws or violate terms of service agreements.	Moderate	Tolerable	Use proxy server to hide IP address of the scraper, so the website cannot track the it's IP and block it.

The quality of the data collected can be affected by various factors such as incomplete information	Moderate	Tolerable	Use multiple trusted articles and blogs to collect data.
The collected data could potentially be at risk of unauthorized access or data breaches if appropriate security measures are not put in place.	Low	Serious	Use reliable and secure database, and apply security testing methods.
Scraper may require regular maintenance to ensure that data collected are legit.	Low	Tolerable	Test the scraper regularly to insure that it collect legit data.
Key staff are ill at critical times in the project.	Moderate	Serious	Reorganize team so that there is more overlap of work and people therefore understand each other's jobs.
The database used in the system cannot process as many transactions per second as expected.	Moderate	Serious	Investigate the possibility of buying a higher-performance database.

Table 1 Risks

C.2 Project Management and Organization

C.2.1 Project Team

Personnel Name	Title	ID	Education Status	Graduation Date	Date of Starting Work
Hassan El Abdallah	Backend Developer/ Project Manager	18700656	Undergrad	01.02.2024	01.03.2023
Ghaleb Mitwalle	Frontend Developer/ UI Designer	20801474	Undergrad	30.06.2024	26.03.2023
Khawlah Alshubati	Database Manager/ Administrator	19701557	Undergrad	01.02.2024	10.04.2023
Abdulaziz Binafif	Tester/System analyst	19701169	Undergrad	01.02.2024	01.15.2023

Table 2 Project Team

C.2.2 Organization Scheme

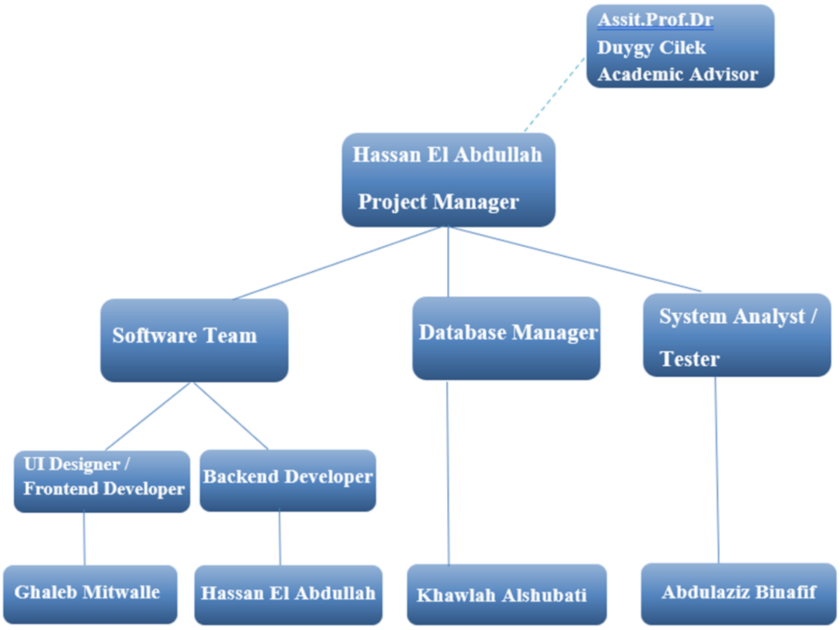


Figure 1 Organization Scheme

D.1 Economic Forecasts

1- Evaluate the commercialization potential of project outcomes. List possible risks here?

Web scarping can be extremely beneficial for content creation and Competitive intelligence but it does pose a few costly risks,

- 1-techinical issues, Web scraping can be technically challenging, and we might need to do a lot of research and maybe even invest in other software

- 2-Ethical risks, Web scraping can be used to collect sensitive information about people, such as their personal preferences and online activities which can be seen a ethically wrong
- 3-legal issues where the program can accidently violate copyright laws and trademarks if not used correctly

2- List your expectations to your team which are come by your project	
Time-to-market (month):	1
The expected increase in sales revenue (%):	50
The expected increase in market share (%):	10
Time to start to gain:	2 months

D.2 National Outcomes

1- Specify the output that may be subject to patent, utility model and industrial design registration in the project.

We are planning to use a novel algorithm that takes unique steps in collecting the data making it fast and efficient, which can be a subject for a patent.

2- Explain the potential of project and its outputs that may have an effect on social life, education, health and etc.

Well depending on the way it is used for it can have big effect in each aspect of life , for instance if it is used for education can be used to gather data on educational trends, student performance, and curriculum effectiveness, which can help teachers make data driven decicsion, and as for health, it can be used to gather data on public health trends, disease outbreaks, and healthcare utilization, which can help public health officials and healthcare providers make more informed decisions about public health.

3- Explain the positive and negative effects of project outputs for environment and human being.

The positive effect is that gathering a lot of information about any topic to help people in power to make data driven decision that benefit the rest on a national level or even bigger in a lot of aspects like health and education , but it also can be used by companies to gather personal data in hopes to make money of their products at the cost of people privacy, or even collect copyrighted information and such.

(M013) Instrument / Equipment / Software / RELEASE PURCHASES

Project Name										
Line no	**Instrument / Equipment / Software / Publication Name**	**No. Of Item**	**Capacity**	**Technical specification**	**Purpose of Project Activities**	**Post-Project Place of Use / Purpose**		**Unit Price (USD)**	**Unit Price (TL)**	**Total Amount (TL)**
						R & D	**Production**			
1	**Work laptops**	**4**	**-**	**Gtx 1080<gpu, cpu i9<cpu, 16gb ram< ram**	**The do all the programming smoothly**	**Research for future updates**	**Updates and fixes**	**1000**	**~18000**	**~72000**

2	**PUBLICATION NAME**	**1**	**-**	**-**	**To preserve the company name**	**Ownership of the app**	**Ownership of the app**	**400**	**7200**	**~7200**
3										
4										
5										
6										
7										
8										
9										
10										
									TOTAL	**72000TL**

Table 3 Instrument / Equipment / Software / RELEASE PURCHASES

(M030) Quarterly Estimated Cost Form (TL)

Project Name :Web scraping				
Cost Item	**2023**		**TOTAL (TL)**	**TOTAL COST RATE OF CONTENTS (%)**
	I	**II**		
Personnel	500	500	1000	~8
Travel	1800	1800	3600	~36
Instrument / Equipment / Software / Publications	0	2000	2000	~18

Domestic Works Made By R & D and Testing Institutions	0	0		
International Works Made By R & D and Testing Institutions	0	0		
Domestic Services Procurement	2000	2000	4000	~40
Overseas Service Procurement	0	0		
Material	0	0		
TOTAL COST	4300	6300	10600	100
CUMULATIVE COST				100
IN THE PROJECT TOTAL MAN-MONTH			2	

Table 4 Quarterly Estimated Cost Form (TL)

APPENDIX

1. CPM (Critical Path Management) analysis by using PERT (defining paths)

Task ID	**Task name**	**Duration (days)**	**Dependency**
A	Conduct a feasibility study	2	
B	Develop a preliminary project plan	2	A

C	Requirements analysis	3	A,B
D	Prepare system design documents	9	C,B
E	Database development	7	D
F	Website development	15	E,D
G	Testing and modifications	15	F
H	Project closure	1	F,G

Table 5 CMP Tasks

Path	**Duration (days)**
A B D E F G H	51
A B D F G H	44
A B D F H	29
A B C D E F G H	**54**
A B C D F G H	47
A B C D F H	32

Table 6 CMP Paths

2. Network Diagram:

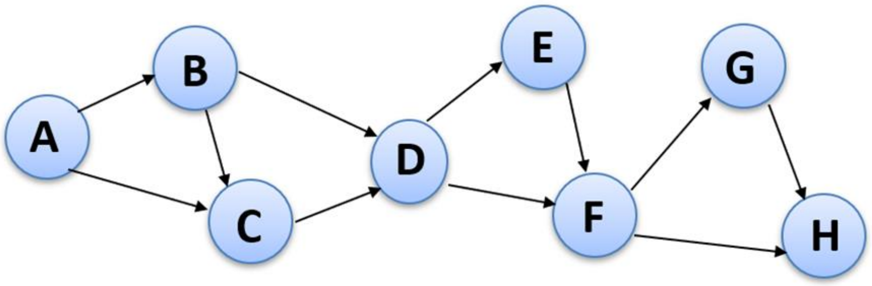
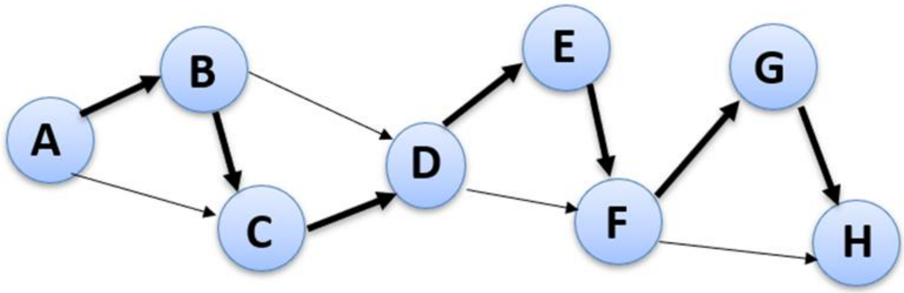


Figure 2 Network Diagram

3. Critical Path:



4. Calculating the Probability For Finishing Project on Time:

c	Predecessor	Optimistic	Most Likely	Pessimistic	Mean	variance	Standard deviation
A	**-**	1	2	4	2.17	0.25	0.5
B	**A**	1	2	4	2.17	0.25	0.5
C	**A,B**	2	3	5	3.17	0.25	0.5

D	**C,B**	5	9	14	9.17	2.25	1.5
E	**D**	4	7	10	7	1	1
F	**E,D**	10	15	20	15	2.78	1.67
G	**F**	10	15	20	15	2.78	1.67
H	**F,G**	1	1	3	1.3	0.11	0.33

Table 7 Calculation of Probabilities

Probability of successful completion rate for all paths:

Expected Project Duration: 45.81 days.

Expected Duration for Critical Path: 54.98.

Variance of Critical Path: sum of variance: 9.67

Standard Deviation for Critical Path: 3.11

Probability the project will finish in 54 days

Z = (x-m)/standard deviation

X = days in question.

M = Expected duration.

1. (54 – 45.81)/3.11 = 2.95
2. P(z) = 0.9957
3. P(z) = 99.57%

Probability of successful completion rate for path ABDEFGH:

Expected Project Duration: **51.81 days.**

Variance: **9.42**

Standard Deviation: **3.06**

Probability the project will finish in 51 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

4. (51 – 51.81)/3.06 = **-o.26**
5. P(z) = 0.3974
6. P(z) = **39.74 %**

Probability of successful completion rate for path ABDFGH:

Expected Project Duration: **44.81 days.**

Variance: **8.42**

Standard Deviation: **2.90**

Probability the project will finish in 44 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

7. (44 – 44.81)/3.06 = **-o.27**
8. P(z) = 0.3936
9. P(z) = **39.36 %**

Probability of successful completion rate for path ABDFH:

Expected Project Duration: **29.81 days.**

Variance: **5.64**

Standard Deviation: **2.37**

Probability the project will finish in 29 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

10. (29 – 29.81)/2.37 = **-o.34**

11. P(z) = 0.3669
12. P(z) = **36.69 %**

Probability of successful completion rate for path ABCDEFGH: (CP)

Expected Project Duration: **54.98 days.**

Variance: **9.67**

Standard Deviation: **3.11**

Probability the project will finish in 54 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

13. (54 – 54.98)/3.11 = **-o.32**
14. P(z) = 0.3745
15. P(z) = **37.45 %**

Probability of successful completion rate for path ABCDFH:

Expected Project Duration: **31.98 days.**

Variance: **5.89**

Standard Deviation: **2.43**

Probability the project will finish in 32 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

16. (32 – 31.98)/2.43 = **-o.40**
17. P(z) = 0.3446
18. P(z) = **34.46 %**

Probability of successful completion rate for path ABCDFGH:

Expected Project Duration: **47.98 days.**

Variance: **8.67**

Standard Deviation: **2.94**

Probability the project will finish in 47 days

Z = (X-M)/standard deviation

X = days in question.

M = Expected duration.

19. (47 – 47.98)/2.94 = **-o.33**
20. P(z) = 0.3707
21. **21.** P(z) = **37.07 %**

5. COCOMO Analysis:

1. Calculate KLOC:

22. KLOC = FP*Language Ratio
23. FP = UFP * [0.65+0.01*DI]

a. Calculating UFP:

Business Functions	Simple	Simple weight	Average	Average weight	Complex	Complex weight	UFPs
User Input (IT)	1	3	2	4	4	6	35
User Output (OT)	2	4	2	5	5	7	53

User Inquiries (QT)	3	3	3	4	8	6	69
Internal Files (FT)	4	7	5	10	10	15	228
External Interfaces(ET)	1	5	4	7	10	10	133
UFP =							518

b. Calculating DI:

Number	Factors	Complexity	Complexity Value
1	Data Communication	Essential	5
2	Distributed data processing	Significant	4
3	Performance Criteria	Essential	5
4	Online Data Entry	Incidental	1
5	High Transaction rate	Essential	5
6	Heavily Utilized Hardware	Incidental	1
7	Maintainability	Average	3
8	Online Updating	Incidental	1
9	Complex Computation	No influence	0
10	Reusability	Average	3
11	Ease of installation	Significant	4

12	Ease of operation	Essential	5
13	Portability	Incidental	1
14	End User Efficiency	Average	3
DI =			40

c. Calculating FP: FP = UFP * [0.65+0.01*DI]

24. FP = 518 *[0.65 + 0.01 * 40] = 543.9

d. Calculating LOC: FP * Language Ration for Python (64 for high level language)

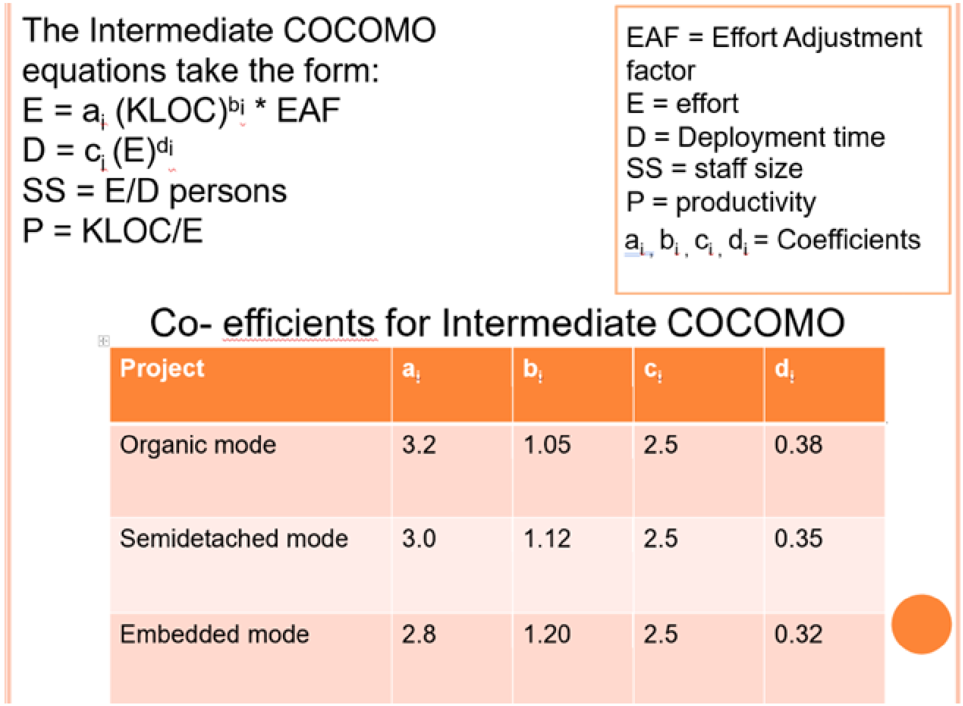
25. 543.9 * 64 = 34809.6 LOC

e. Calculating KLOC: LOC/1000

26. 34809.6 /1000 = 34.81 KLOC.

2. Do the Estimation:

Accordingly this is under the Organic Mode and since our project is considered intermediate we used this table:



The Intermediate COCOMO equations take the form:

$E = a_i (KLOC)^{b_i} * EAF$

$D = c_i (E)^{d_i}$

SS = E/D persons

P = KLOC/E

EAF = Effort Adjustment factor
E = effort
D = Deployment time
SS = staff size
P = productivity
a_i, b_i, c_i, d_i = Coefficients

Co- efficients for Intermediate COCOMO

Project	a_i	b_i	c_i	d_i
Organic mode	3.2	1.05	2.5	0.38
Semidetached mode	3.0	1.12	2.5	0.35
Embedded mode	2.8	1.20	2.5	0.32

Table 8 Intermediate COCOMO

a. Calculate the Cost Drivers:

27. EAF = DATA * TURN * PCAP * LEXP * TOOL = 1*1.07*0.86*0.95*0.83 = 0.73

b. Calculate Effort Estimate:

28. E = ai(KLOC)bi * EAF = 3.2*(34.81)$^{1.05}$*0.73 = 97.11 PM

c. Calculate Duration:

29. D = ci (E)^{di} = 2.5 * 97.11 $^{\mathbf{0.38}}$ = 14.23 M

d. Calculate Staff Size:

30. SS = E/D = 97.11/14.23 = 6.82 Persons

e. Calculate Productivity:

31. P = KLOC/E = 34.81/97.11 = 0.36

3. REQUIREMENTS ANALYSIS

1. Authentication

3.1.1 Description and Priority

The authentication system is an important feature for the website that requires users to identify themselves before being able to add a review to the doctor. It has a high priority to prevent fake reviews and spam.

3.1.2 Stimulus/Response Sequences

Users should be able to register on the website by providing their basic details, such as username, email address, and a password. Then each time the user logs in to the system, a verification code will be sent to their email.

3.1.3 Functional Requirements

REQ-1: User shall be able to sign in into the system
REQ-2: User should be able to create a new account
REQ-3: System shall be able to send verification code to user’s email
REQ-4: User shall be able to reset their password

2. View doctors’ information

3.2.1 Description and Priority

The view doctor feature is one of the core features of the system, where users shall be able to view doctor’s detailed information/profile. It has a high priority so the users can find a healthcare professional who meets their medical needs.

3.2.2 Stimulus/Response Sequences

Users will be able to search for a doctor on the website. Once they find a doctor that matches their criteria, they can view the doctor's profile, which will contain important information such as the doctor's qualifications, experience, areas of expertise, and patient ratings and reviews.

3.2.3 Functional Requirements

REQ-1: User shall be able to view top rated doctors
REQ-2: User will be able to view doctor’s profile
REQ-3: User should be able to view doctor’s area of expertise
REQ-4: User shall be able to view doctor’s clinic

3. Search for doctors

3.3.1 Description and Priority

Search for doctor feature is a critical aspect of any healthcare website or application that aims to connect patients with medical professionals. This feature enables users to search for doctors based on various criteria, such as location, specialty, availability, and rating, and helps them find a medical professional who can meet their healthcare needs.

3.3.2 Stimulus/Response Sequences

On the homepage of the website, users will have access to a search bar that they can use to enter keywords. Additionally, an advanced filter option will be available, enabling users to refine their search results based on specific criteria such as location, specialty, or rating.

3.3.3 Functional Requirements

REQ-1: User should be able to search doctors by name, location, and field
REQ-2: User shall be able to filter doctors by hospitals and clinics
REQ-3: User shall be able to sort doctors by their reviews

4. Doctors' reviews

3.4.1 Description and Priority

Adding a review for a doctor is a high priority core feature that allows patients to share their experiences with a medical professional and provide feedback on the quality of care they received. It helps other patients make informed decisions about which doctor to choose.

3.4.2 Stimulus/Response Sequences

Once a user has searched for and viewed a doctor's profile, there will be a review button for adding a review, if they click it, they will have the ability to submit a review about the doctor if it meets the website's criteria for validity.

3.4.3 Functional Requirements

REQ-1: User shall be able to add a review for a doctor
REQ-2: User should be able to view doctor's reviews
REQ-3: User will be able to compare doctors by reviews
REQ-4: System shall be able to detect fake and spam reviews

5. Scraper

3.5.1 Description and Priority

This feature can allow the healthcare website or application to provide a comprehensive list of doctors in each area or specialty, even if they are not listed on the website itself. It's the highest priority feature, because it should be designed to collect accurate and up-to-date information on doctors and should be regularly maintained to ensure that the data remains relevant.

3.5.2 Stimulus/Response Sequences

The website administrator will have the ability to initiate the web scraper tool, select the desired area of expertise to gather data on, and then the scraper will automatically extract doctors' information from other healthcare websites and store it in the website's database.

3.5.3 Functional Requirements

REQ-1: Scraper will be able to scrap doctor’s profile information
REQ-2: Scraper should be able to scrap doctor’s area of expertise
REQ-3: Scraper should be able to get doctor’s location
REQ-4: Scraper must be able to scrap multiple wesbites at once

6. Admin

3.6.1 Description and Priority

An admin feature allows the website administrator to manage and monitor the website's content, user accounts, and system settings. This feature is critical for ensuring that the website operates smoothly, securely, and in compliance with legal and ethical standards.

3.6.2 Stimulus/Response Sequences

After logging into the admin page, the administrator will have access to a wide range of actions they can perform on the website. These actions could include adding or deleting user accounts, managing doctor profiles, moderating user-generated content, configuring system settings, and generating analytics reports.

3.6.3 Functional Requirements

REQ-1: Admin must be able to add users
REQ-2: Admin shall be able to keep track of new doctors added to the system
REQ-3: Admin shall be able view all users of the system
REQ-4: Admin should be able to view all reviews added by users
REQ-5: Admin must be able to track fake and spam reviews
REQ-6: Admin shall be able to ban/remove users with spam or fake reviews

3.2 Non-Functional Requirements

1. Performance Requirements

- Throughput: The web scraping product may be required to process a large volume of data in a short amount of time. For example, the product may be required to scrape data from hundreds of web pages in under an hour. This requirement is important to ensure that the product can handle large-scale data extraction efficiently.

2. Safety Requirements

- Data Privacy: The web scraping product should not violate any laws or regulations related to data privacy, such as the General Data Protection Regulation (GDPR). The product should not extract any personal information about doctors without their explicit consent or in violation of privacy policies.

- Use of Third-Party Libraries: The web scraping product should not use any third-party libraries that violate safety or security standards. The product should only use libraries that have been thoroughly tested and approved for use by the development team.

3. Security Requirements

- User Authentication: The web scraping product should require users to authenticate themselves before using the product. User authentication may involve username and password authentication, multi-factor authentication, or other methods that verify the identity of the user.
- Data Encryption: The product should use encryption algorithms to secure any sensitive data transmitted between the user and the server, as well as any data stored on the server or user's device.
- Secure Data Storage: The product should store extracted data in a secure and encrypted manner to prevent unauthorized access or data breaches. The product should also have mechanisms in place to protect against data loss or corruption.

4. Software Quality Attributes

- Scalability: The ability to handle an increasing amount of data without a significant decrease in performance. For example, the product should be able to handle scraping data from a large number of websites without crashing or slowing down.
- Maintainability: The ease with which the product can be maintained, modified or updated. This includes code readability, use of clear documentation, and well-structured code.
- Reusability: The extent to which components of the product can be used in other contexts or products. For example, if the product includes a component that extracts data from a particular type of website, it should be possible to reuse that component for other websites of the same type.
- Usability: The ease of use and navigation of the product. This includes features such as a clear and intuitive user interface, clear error messages, and easy-to-understand documentation.

5. **Business Rules**

- Only authorized users with valid login credentials can access and use the product.
- The product should only scrape publicly available data and not violate any laws or regulations.
- The product should scrape data in a manner that is respectful of the websites being scraped, avoiding excessive traffic or server load.
- The product should prioritize accuracy and completeness of scraped data, and alert users if data is missing or incomplete.
- The product should store scraped data securely and protect user privacy.

3.3 Realistic constraints

1. Economic:

- The software should be accessible to all users, regardless of their financial resources. It should not require a significant investment for individuals or organizations to use the software effectively.
- The cost of using the software, such as licensing fees or subscription charges, should be reasonable and affordable for different user groups, including individuals, small clinics, and larger healthcare organizations.

2. Environmental:

- The software should aim to minimize its power consumption during operation. It should be designed to optimize resource usage and minimize unnecessary energy consumption.
- The development and usage of the software should not contribute significantly to pollution or harm the environment. Efforts should be made to minimize the carbon footprint associated with the software's operations.

3. Social:

- The software should not impose any discriminatory or exclusionary restrictions based on social factors such as age, gender, race, or socioeconomic status. It should be accessible and usable by all segments of society.
- Compliance with relevant data protection and privacy regulations is essential to ensure the software respects users' rights and maintains user trust.

4. Political:

- The software should avoid any features or functionality that could be perceived as politically biased, controversial, or potentially create conflicts. It should adhere to applicable laws and regulations in the regions where it operates.

5. Ethical:

- The software should adhere to ethical standards, including respecting intellectual property rights. It should not "borrow" ideas, code, or any other intellectual property from other projects without proper acknowledgment or permission.

6. Health and Safety:

- The software should not pose any risks to the health and safety of users or society. It should be designed to ensure the integrity and accuracy of the scraped data to avoid potential harm caused by incorrect or misleading information.

7. Manufacturability:

- While DocFinder is software, the concept of manufacturability can be translated into the development process. The software should be developed using appropriate software engineering practices, making it maintainable, scalable, and easily deployable. This ensures efficient use of development resources and allows for updates and enhancements to be implemented smoothly.

8. Sustainability:

- The software should be designed and developed with a long-term perspective in mind, considering its longevity and adaptability to evolving technologies and user needs. This will ensure that the software remains relevant and usable over an extended period.

3.4 Ethical issues

When considering the ethical issues related to the DocFinder web scraping software , the following points should be considered:

1. **Privacy and Data Protection**: The software should adhere to strict privacy standards and comply with relevant data protection regulations. Ethical concerns arise if the software collects, stores, or shares user data without informed consent or uses data in ways that violate privacy rights.

2. **Unauthorized Access and Misuse**: If the software enables or facilitates unauthorized access to sensitive information or encourages users to engage in illegal activities, such as identity theft or fraud, it raises ethical concerns. Safeguards should be in place to prevent misuse and ensure that the software is not used for criminal purposes.

3. **Accuracy and Integrity of Data**: The software should strive to provide accurate and up-to-date information about physicians. Ethical concerns arise if the software disseminates inaccurate or misleading data, potentially leading to incorrect medical decisions or patient harm.

4. **Bias and Discrimination**: Ethical considerations arise if the software introduces biases or discriminates against certain groups, such as promoting or excluding specific physicians based on personal characteristics or discriminatory criteria. The software should be designed and tested to ensure fairness and mitigate bias.

5. **Intellectual Property Rights**: Ethical issues may arise if the software infringes upon intellectual property rights by using copyrighted content or proprietary data without proper authorization. Respecting intellectual property rights and giving appropriate credit are important ethical considerations.

6. **Transparency and Informed Consent**: Users should be well-informed about the software's functionality, data collection practices, and potential uses of their data. Ethical concerns arise if users are not adequately informed or if their consent is not obtained transparently.

7. **Responsible Use**: The software should include terms of use and guidelines that promote responsible and ethical behavior by its users. It should discourage the misuse of scraped data for unethical purposes, such as harassment, stalking, or defamation.

8. **Security and Confidentiality**: Ethical concerns arise if the software fails to implement adequate security measures to protect user data from unauthorized access, breaches, or misuse. Confidentiality of sensitive information should be prioritized to maintain trust and prevent harm.

4. DESIGN

4.1 High level design (architectural)

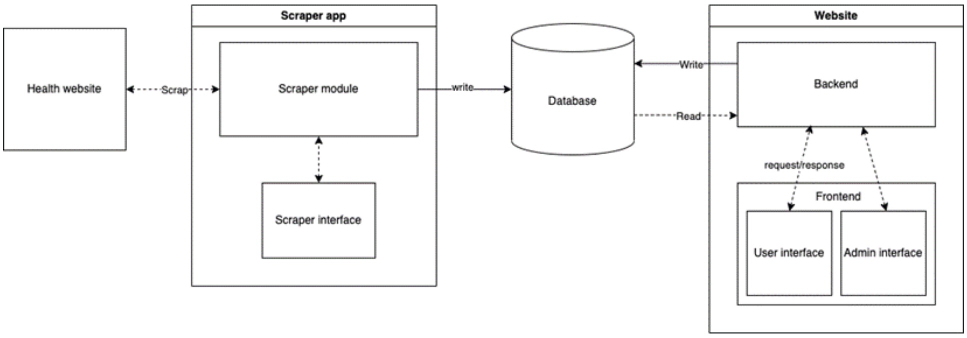


Figure 3 Architecture diagram

Website: This component serves as the primary means of providing users with access to information about doctors, as well as giving administrators the ability to manage and control the system. This component has two parts: the Frontend, which is responsible for the user and admin interface, and the backend, which handles requests, authentication, and database operations.

Database: The Database component is tasked with the responsibility of storing data related to users and doctors. It facilitates the CRUD (Create, Read, Update, and Delete) operations associated with the data.

Scraper app: It is a software program designed to collect data on doctors from health websites and store it in the database for use on the website. It is composed of two main components: the scraping module, which is the central sub-component of the app and contains the script for the scraping process, and an interface that allows users to access and utilize the app's scraping capabilities.

Health website: is the website from which we will gather information on doctors

4.2 Software design

4.2.1 Context Diagram

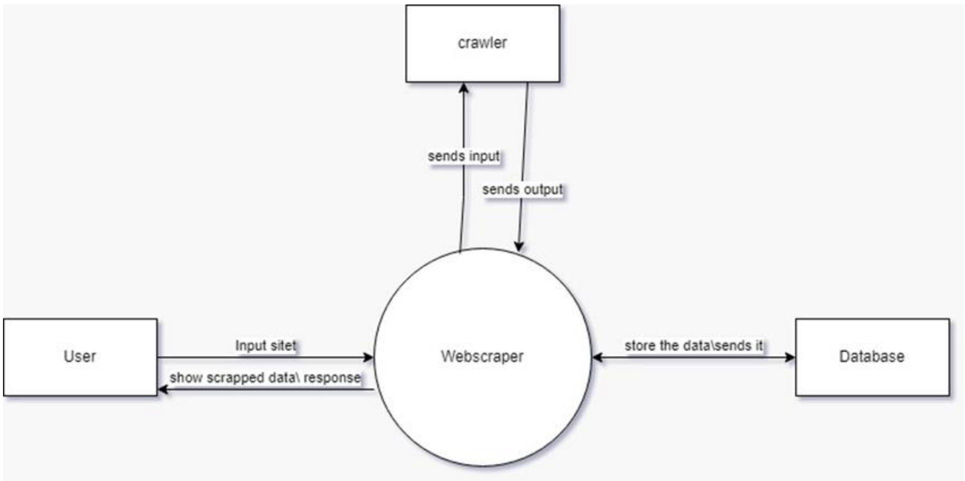


Figure 4 Context Diagram

This context diagram for our web scraping system for doctors' data depicts the system at the center, surrounded by its primary interactions. The system interacts with three main components: the user, the database, and the crawler. The user initiates the scraping process by providing input and receiving output from the system. The database serves as the storage for the extracted doctors' data, enabling efficient data retrieval and management. The crawler component is responsible for navigating the target websites, extracting the relevant information, and feeding it into the system for processing. The context diagram presents a concise overview of how the system interacts with its key components, facilitating a clear understanding of its role and relationships.

4.2.2 DFD Level 0 Diagram:

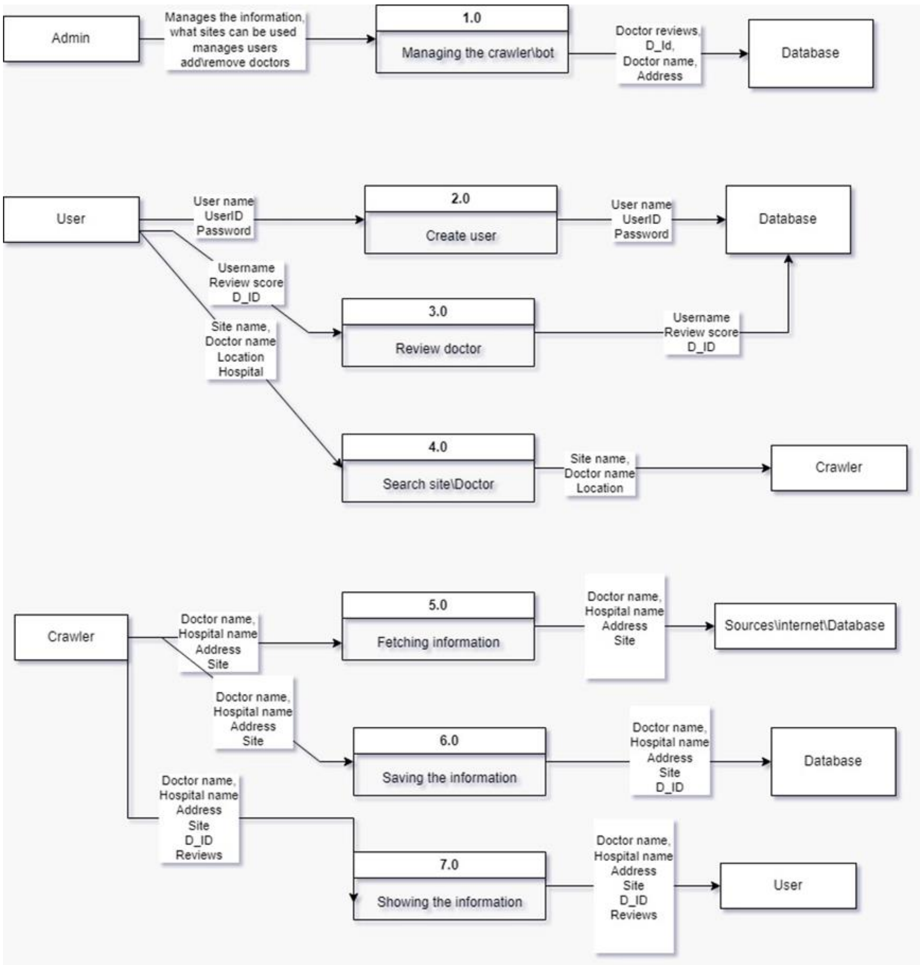

Admin
Manages the information, what sites can be used manages users add\remove doctors
1.0
Managing the crawler\bot
Doctor reviews, D_Id, Doctor name, Address
Database
User
User name UserID Password
2.0
Create user
User name UserID Password
Database
Username Review score D_ID
3.0
Review doctor
Username Review score D_ID
Site name, Doctor name Location Hospital
4.0
Search site\Doctor
Site name, Doctor name Location
Crawler
Crawler
Doctor name, Hospital name Address Site
5.0
Fetching information
Doctor name, Hospital name Address Site
Sources\internet\Database
Doctor name, Hospital name Address Site
6.0
Saving the information
Doctor name, Hospital name Address Site D_ID
Database
Doctor name, Hospital name Address Site D_ID Reviews
7.0
Showing the information
Doctor name, Hospital name Address Site D_ID Reviews
User

Figure 5 DFD Level 0 Diagram

This is data flow diagram (DFD- Level 0) for our web scraping system for doctors' data. It illustrates the flow of information within the system. The main process interacts with the user, receiving input such as website URLs and data requirements. It communicates with the crawler component to extract doctors' data, which is stored in a database. The system provides outputs to the user, such as data reports or notifications. The DFD highlights the system's data flow and interactions with external entities, aiding in understanding its functionality and data storage components.

4.2.3 BPMN Diagram

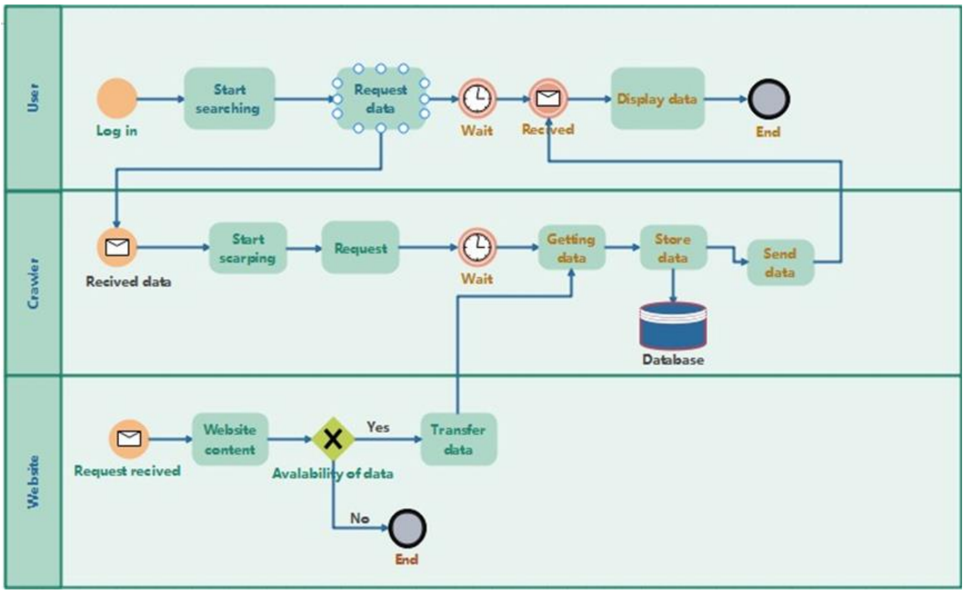


Figure 6 BPMN Diagram

This is the BPMN diagram for our web scraping system for doctors' data. It depicts the system's workflow. İt starts by user loging in to the sysetem, start searching for a doctor which will trigger a reqeust to the database in the system. User will waite a few seconds, data searched for will be displayed. For the crawler side, the requested data recieved, then it starts scrapping, and gets data from different websites and after that sends the data to the user interface.

4.2.4 ER Diagrams:

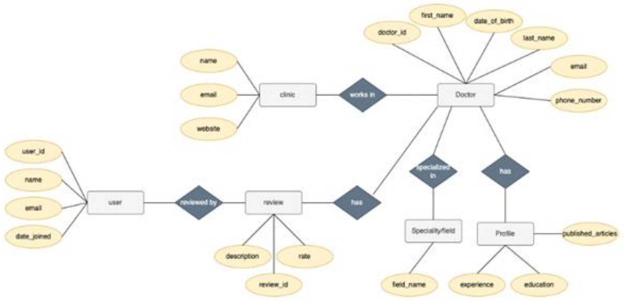
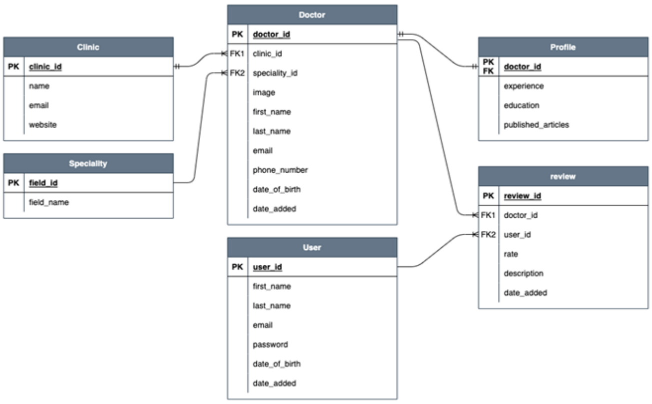


Figure 7 ER Diagram



Clinic	
PK	clinic_id
	name
	email
	website

Speciality	
PK	field_id
	field_name

Doctor	
PK	doctor_id
FK1	clinic_id
FK2	speciality_id
	image
	first_name
	last_name
	email
	phone_number
	date_of_birth
	date_added

User	
PK	user_id
	first_name
	last_name
	email
	password
	date_of_birth
	date_added

Profile	
PK FK	doctor_id
	experience
	education
	published_articles

review	
PK	review_id
FK1	doctor_id
FK2	user_id
	rate
	description
	date_added

Figure 8 ER Diagram

Our ER Diagram has the following entities, the user which has the user_id, name, email, date_joined. Doctors which have some information about those doctors. Review which has the description, rate, and review_id. The SpecialityField which contains field_name. The profile which contains the experiences, education and published articles. Additionally, we have clinics table which has clinics name, email, website columns.

4.2.5 Activity Diagrams

1. Add User Review:

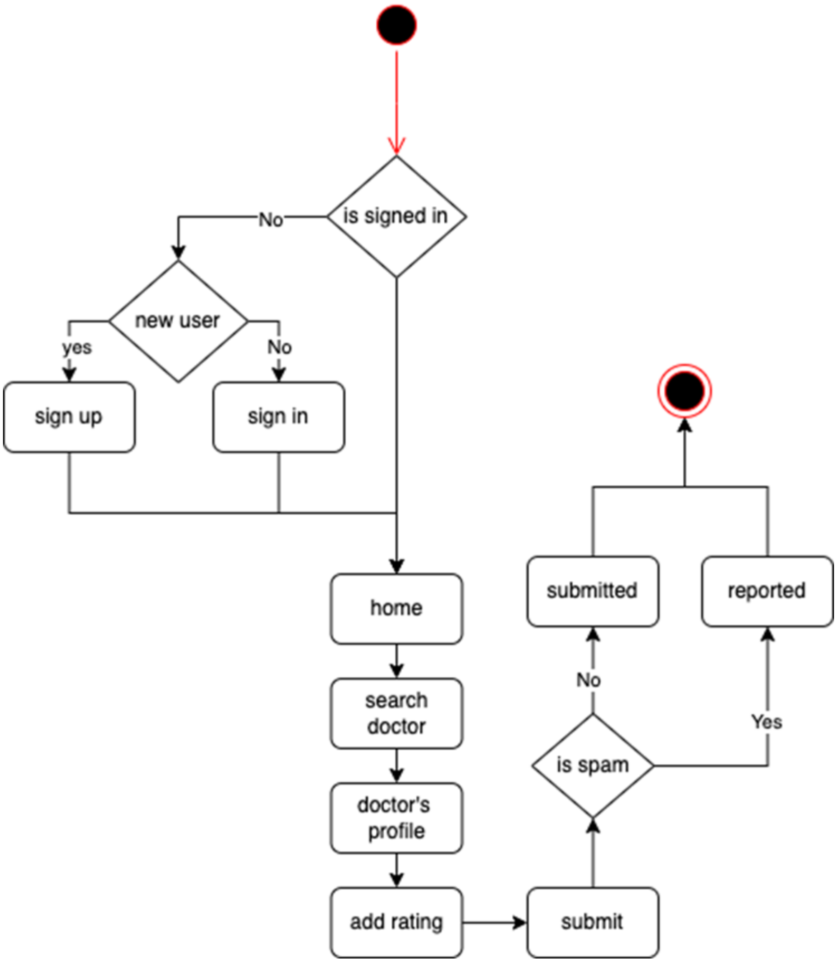


Figure 9 Activity Diagram (Add User Review)

This activity diagram illustrates the process of a user adding a review for a doctor. It starts with the user logs in then searching for a doctor. After that the user selects a doctor, then filling out a review form with feedback, ratings, and comments. The system validates the review, and if it passes, the review is submitted, stored in the database, and associated with the respective doctor and user. The doctor's ratings and statistics are updated accordingly. The activity diagram concludes with the completion of the review addition process.

2. Scrapping for Doctors:

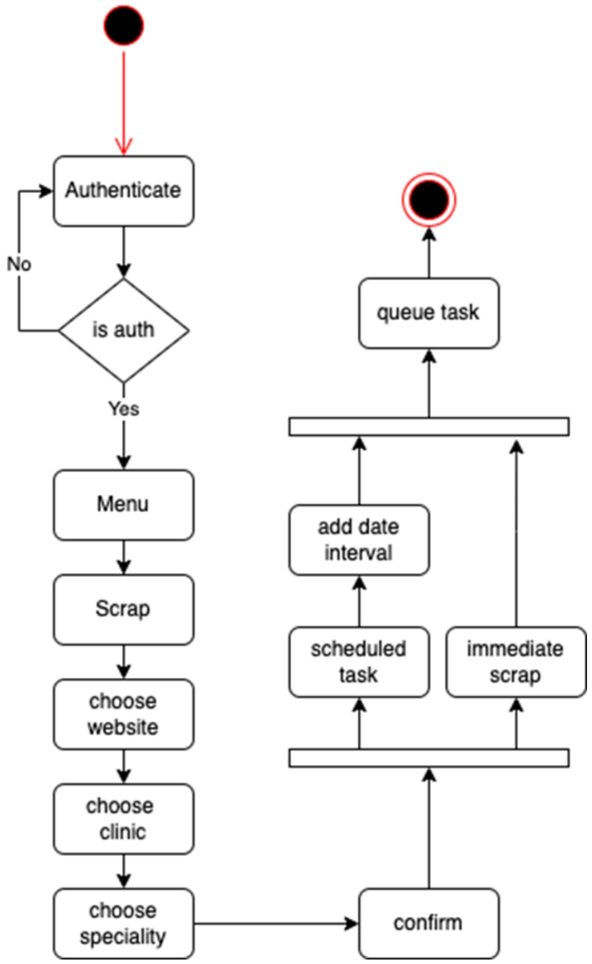


Figure 10 Activity Diagram (Scrapping for Doctors)

This activity diagram illustrates the process of the system scraping doctors' data from external sources. It starts with selecting the source and then proceeds to scrape the data using web scraping techniques. Afterward, the system processes and cleans the scraped data before storing it in the database. The activity diagram concludes with the completion of the scraping process. The flow of the process is shown clearly in this diagram.

3. Admin Track New Doctors:

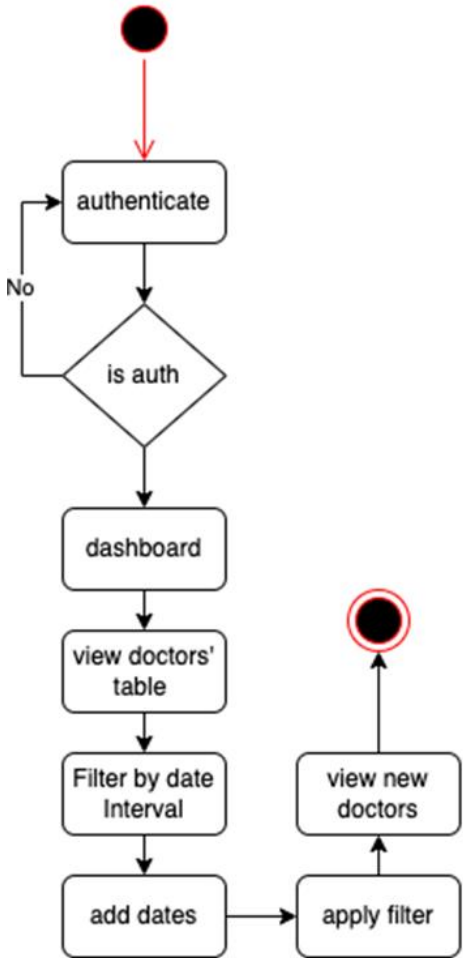


Figure 11 Activity Diagram (Admin Track New Doctors)

This activity diagram is for the admin tracking new doctors. It starts with the authentication process, where the administrator logs into the system. After successful authentication, the system enters the admin’s dashboard and then he can go to view doctors’ table and the "Track New Doctors" activity. It continuously checks for new doctor registrations. If new doctors are found, the system notifies the administrator. The administrator reviews the new doctors' information and decides to approve or reject them. The system updates the doctors' status accordingly. Finally, the administrator views the list of newly approved doctors. Additionally, admin can filter the tracked data by dates.

4.2.6 Use Case Diagram

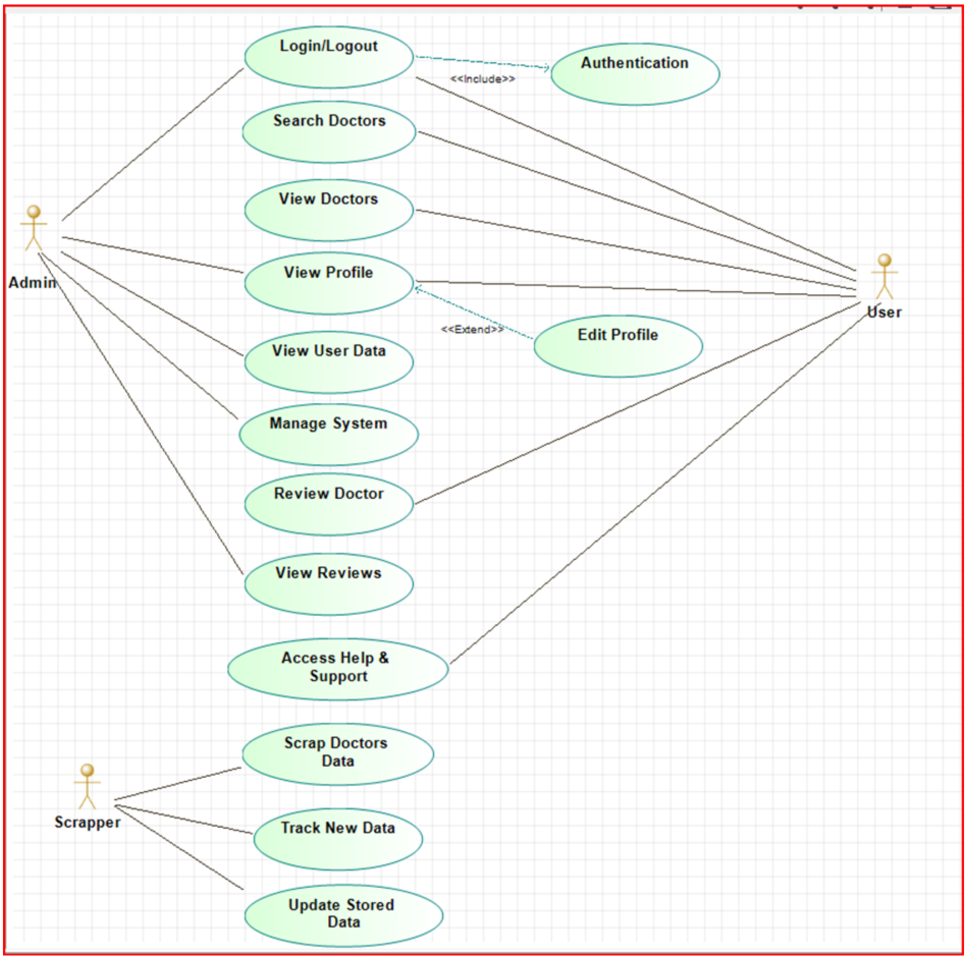


Figure 12 Use Case Diagram

This use case explains major functionalities of the system as shown.

Use case narratives:

Use case1
Title: Search Doctors
Actors: User, the person searching for doctors.
Description: this use case describes the process of searching for doctors on the system. The user wants to find doctors based on specific criteria, such as location, specialty, or name.

Preconditions:
• The user is logged into the system.
• The system is accessible and operational.
Use case scenario:
• The user accesses the search page within the system.
• The user provides search criteria, such as location, specialty, or name, through the user interface.
• The user interface validates and sends the search criteria to the doctor search controller.
• The doctor search controller receives the search criteria and initiates the search process.
• The doctor search controller communicates with the doctor search service to retrieve matching doctors based on the provided criteria.
• The doctor search service performs the search algorithm, querying the database for doctors that meet the search criteria.
• The database retrieves and returns the list of matching doctors to the doctor search service.
• The doctor search service collects the list of doctors and sends it back to the doctor search controller.
• The doctor search controller receives the list of doctors and formats the results.
• The formatted search results are sent to the user interface for display.
• The user interface presents the search results to the user.
• The user can view detailed information about a specific doctor by selecting the corresponding option.
• The user can choose to refine or modify the search criteria and repeat the search process if needed.
• Postconditions:
• The user is presented with a list of doctors matching the search criteria.
• The user can view detailed information about specific doctors.

- The user can refine or modify the search criteria and perform another search if desired.
Alternative scenario:
- If no doctors match the search criteria, the system displays a message indicating no results found.
- If there are any technical issues during the search process, such as a connection failure or database error, an error message is displayed to the user, prompting them to try again later or contact support.
Exceptions:
- If the user is not logged into the system, they are redirected to the login page before accessing the search functionality.

Table 9 Use Case 1

Use Case 2
Title: Scrape Doctors' Data
Actors: Scraper, The component responsible for scraping doctors' data from target websites.
Description: this use case describes the process of scraping doctors' data from target websites using a scraper component. The scraper autonomously retrieves relevant information about doctors, such as their names, specialties, contact details, and locations.
Preconditions:
- The system is accessible and operational.
- The target websites to be scraped are identified and accessible.
Use case scenario:
- The scraper component is initialized within the system.

• The scraper retrieves the list of target websites to be scraped from the configuration or a predefined source.
• For each target website:
o The scraper component navigates to the website using web crawling techniques.
o It identifies and extracts the relevant data based on predefined scraping rules or patterns.
o The extracted data is stored in a temporary data storage area or memory.
o Once all target websites have been scraped, the scraper proceeds to process the extracted data.
• The scraper component performs any necessary data processing or formatting on the extracted data.
• The processed data is stored in the system's database for further use and analysis.
• The scraper component generates a completion message indicating the success of the scraping process.
• The completion message is logged and made available for system monitoring and auditing purposes.
Postconditions:
• The system contains the scraped doctors' data, including their names, specialties, contact details, and locations.
• The extracted doctors' data is stored in the system's database for further use.
Alternative Scenario:
• If there are any technical issues during the scraping process, such as website unavailability or connection errors, the scraper component logs an error and attempts to retry the scraping process after a specified interval.
• In the event of persistent issues or failures, the system may trigger an alert or notification to administrators or technical support for further investigation.
Exceptions:

• If the target websites become inaccessible or undergo significant structural changes, the system may require manual intervention to update the scraping rules or adapt to the changes.

Table 10 Use Case 2

Use Case 3
Title: Review a Doctor
Actors: User, The person providing a review for a doctor.
Description: this use case describes the process of reviewing a doctor in the system. The user, who has had an experience with a specific doctor, shares their feedback, rating, and comments to help others make informed decisions.
Preconditions:
• The user is logged into the system.
• The system is accessible and operational.
• The doctor being reviewed is registered in the system.
Use case main scenario:
• The user navigates to the page for reviewing doctors within the system.
• The user selects the doctor they wish to review from a list or by searching for the doctor’s name.
• The system presents a review form to the user, allowing them to provide their feedback, rating, and comments about the doctor.
• The user fills in the review form, rating the doctor on various aspects and sharing their comments.
• Upon submission, the system validates the review, ensuring all required fields are filled.
• The system stores the review in the database, associating it with the respective doctor and the user who submitted it.

• The system updates the doctor’s overall rating and statistics based on the newly added review.
• The user receives a confirmation message indicating that their review has been successfully submitted.
Postconditions:
• The doctor receives a new review, which is associated with their profile in the system.
• The doctor’s overall rating and statistics are updated based on the received review.
• The user receives a confirmation message indicating the successful submission of their review.
Alternative Scenario:
• If the user attempts to submit a review without filling in all the required fields, the system displays an error message, prompting the user to complete the necessary information.
• In the case of a technical error or database issue during the review submission process, the system displays an error message and advises the user to try again later or contact support.
Exceptions:
• If the user is not logged into the system, they are redirected to the login page before accessing the review functionality.

Table 11 Use Case 3

Use Case Glossary:

Use Case	Description	Actors	Preconditions	Postconditions
Login/Logout	The process of logging into or	User, admin	None	User is logged in or logged

	out of the system			out of the system
Authentication	The process of verifying the identity of a user	User, admin	User has provided valid credentials	User is granted access based on authentication
Search Doctors	The process of searching for doctors based on criteria	Users	User is logged in	User is presented with a list of matching doctors
View Doctors	The process of viewing detailed information about a doctor	Users	User is logged in	User can view a doctor's profile and details
View Profile	The process of viewing the user's own profile information	Users	User is logged in	User can view their own profile information
View User Data	The process of viewing data related to other users	Users	User has appropriate permissions	User can view data of other users
Edit Profile	The process of modifying and	Users	User is logged in	User's profile information is updated

	updating user's profile			
Manage System	The process of managing and configuring system settings	Administrator	Administrator is logged in	System settings and configurations are modified
View Reviews	The process of accessing and viewing reviews for doctors	User	User is logged in	User can read and access reviews for doctors
Access Help/Support	The process of accessing help and support resources	User	User is logged in	User can access help documentation or contact support
Scrape Doctors Data	The process of automatically extracting doctors' data	Scraper	System is operational and target websites are accessible	Doctors' data is extracted and stored in the system's database
Track New Data	The process of monitoring and identifying new doctors' data	Scraper	System is operational and target websites are accessible	New doctors' data is identified for extraction

Update Stored Data	The process of updating existing doctors' data	Scraper	System is operational and target websites are accessible	Existing doctors' data is refreshed and updated

Table 12 Use Case Glossary

4.2.7 Sequence Diagrm:

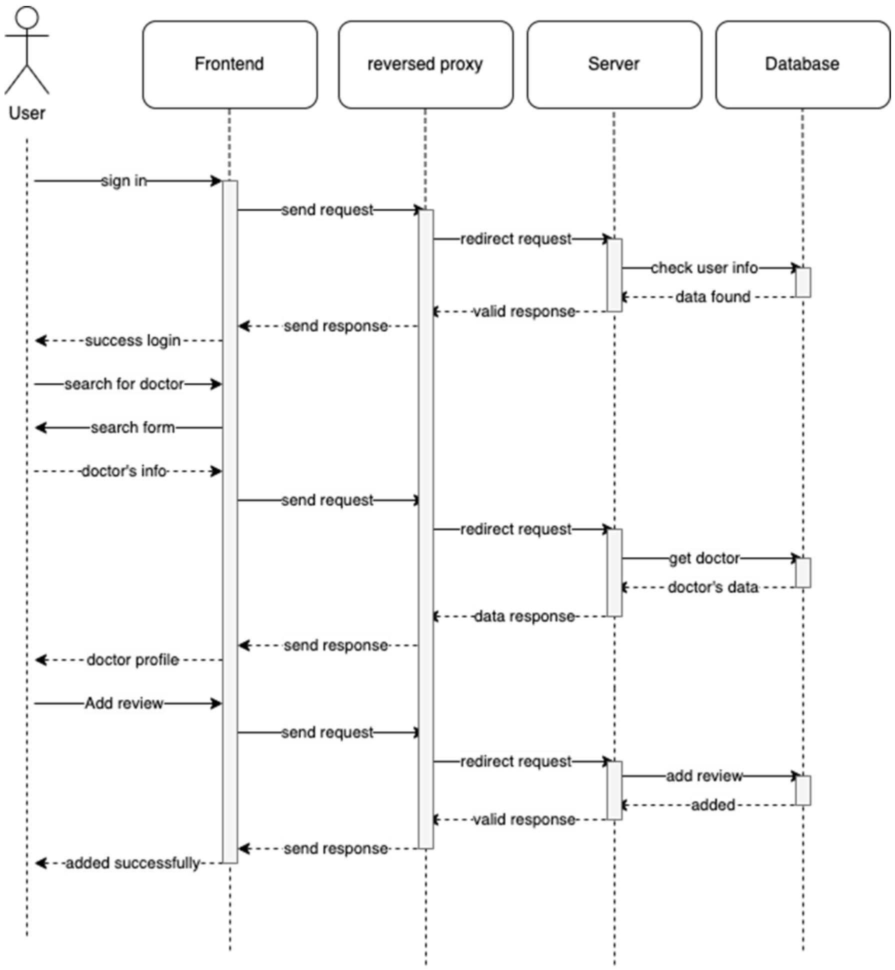


Figure 13 Sequence diagram

This sequence diagram illustrates the interactions and flow of messages between user, frontend, server and database components of the system. This is the case of a user adding a review where the steps evolved in this process are captured. Here is the sequence in details:

1. User signs in to the website.
2. The system’s frontend sends a request to reversed proxy.
3. The reversed proxy directs this request to the server.
4. The server sends this information to be checked in the database.
5. In this case, database finds the user information so it confirms the server that data of this user is found.
6. The server sends a valid response to reversed proxy.
7. The reversed proxy assures that valid response to frontend component.
8. The frontend indicates the success login for the user.
9. Now user will search for a doctor.
10. A search form will appear to user.
11. User looks for doctors’ information.
12. This request is sent to server and thus be found in database of the system.
13. The database retrieves doctor’s data and send it to server.
14. The server sends response to frontend and thus the doctors profile will appear to the user.
15. Now the user can add his review upon that doctor.
16. The review will be added to the database through reversed proxy and system’s server.
17. The database confirms that the review is added and thus sends a valid response back to the server and the server confirms this to frontend.
18. Now the user’s review is added successfully.

4.2.7 Class Diagram:

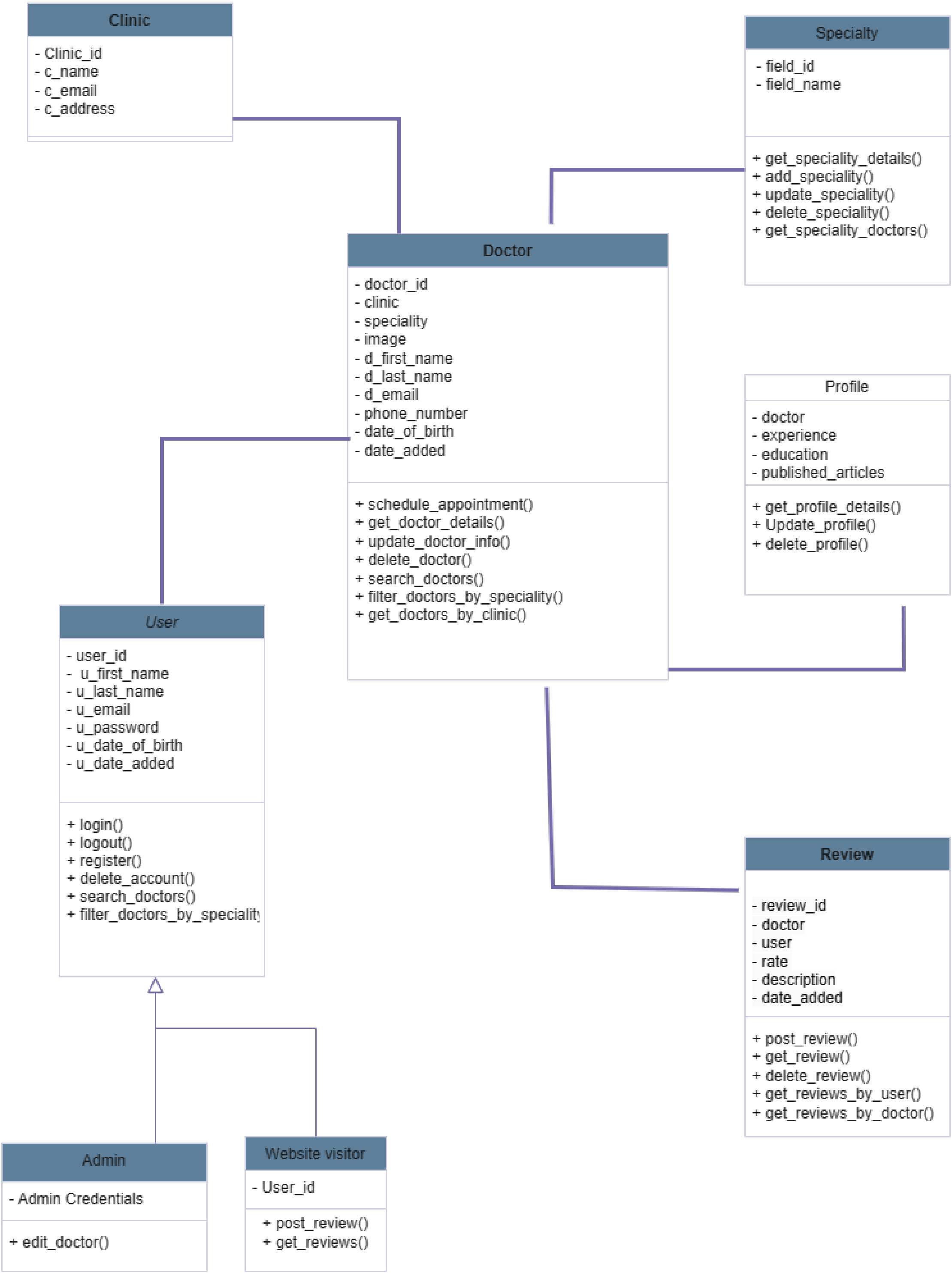


Figure 14: Class Diagram

The class diagram of the system involves the following classes with their methods.

1. Doctor: Represents a doctor with methods for scheduling appointments, retrieving details, updating information, deleting, and searching/filtering doctors.

2. User: Represents a user with methods for login, logout, registration, account deletion, searching/filtering doctors, and posting/retrieving reviews.

3. Clinic: Represents a clinic with attributes for ID, name, email, and website.

4. Speciality: Represents a medical specialty with methods for retrieving details, adding, updating, and deleting specialties.

5. Profile: Represents a doctor's profile with attributes for experience, education, and published articles, and methods for retrieving details, updating, and deleting profiles.

6. Review: Represents a review with attributes for ID, doctor, user, rate, description, and date added, and methods for posting, retrieving, and deleting reviews.

These classes and methods provide a basic structure for implementing a doctor-finding web scraping tool with functionalities for managing doctors, users, clinics, specialties, profiles, and reviews.

4.2.8 Relational Tables:

1. Doctor(doctor_id /**PK**, clinic_id /**FK**, speciality_id /**FK**, image, d_first_name, d_last_name, d_email, phone_number, date_of_birth, date_added**)**

2. User(user_id /**PK**, u_first_name, u_last_name, u_email, u_password, u_date_of_birth, u_date_added**)**

3. Clinic(clinic_id /**PK**, c_name, c_email, c_website**)**

4. Speciality(field_id /**PK**,field_name**)**

5. Profile(doctor_id /**PK**, experience, education, published_articles**)**

6. review(review_id /**PK**, doctor_id /**FK**, user_id /**FK**, rate, description, date_added**)**

4.2.9 Physical Database Management (PDM)

```
CREATE TABLE IF NOT EXISTS `Clinics` (

  `clinic_id` VARCHAR(255) NOT NULL,
  `name` VARCHAR(255) NOT NULL,

  `email` VARCHAR(255),

  `website` VARCHAR(255) NOT NULL,

  `clinic_address` VARCHAR(255) NOT NULL,

  PRIMARY KEY (`clinic_id`)

) ENGINE=InnoDB;


CREATE TABLE IF NOT EXISTS `Doctors` (

  `doctor_id` INTEGER auto_increment,

  `clinic_id` VARCHAR(255),

  `field_name` VARCHAR(255),

  `experience` VARCHAR(1000) NOT NULL,

  `education` VARCHAR(1000) NOT NULL,

  `image` VARCHAR(1000),

  `name` VARCHAR(255) NOT NULL UNIQUE,

  `email` VARCHAR(255) NOT NULL,

  `phone_number` VARCHAR(255) NOT NULL,

  `date_added` DATETIME NOT NULL,

  PRIMARY KEY (`doctor_id`),

  FOREIGN KEY (`clinic_id`) REFERENCES `Clinics` (`clinic_id`),

  FOREIGN KEY (`field_name`) REFERENCES `Specialities` (`field_name`)

) ENGINE=InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `Users` (
  `userId` INTEGER auto_increment,
  `username` VARCHAR(255) NOT NULL UNIQUE,
  `email` VARCHAR(255) NOT NULL UNIQUE,
  `firstName` VARCHAR(255),
  `lastName` VARCHAR(255),
  `password` VARCHAR(255) NOT NULL,
  `date_added` DATETIME NOT NULL,
  `date_of_birth` DATETIME,
  `is_admin` TINYINT(1),
  PRIMARY KEY (`userId`)
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS `Reviews` (
  `review_id` INTEGER auto_increment,
  `doctor_id` INTEGER,
  `user_id` INTEGER,
  `rate` INTEGER NOT NULL,
  `description` VARCHAR(255),
  `date_added` DATETIME NOT NULL,
  PRIMARY KEY (`review_id`),
  FOREIGN KEY (`doctor_id`) REFERENCES `Doctors` (`doctor_id`),

  FOREIGN KEY (`user_id`) REFERENCES `Users` (`userId`)

) ENGINE=InnoDB;


Executing (default): SHOW INDEX FROM `Reviews`;


CREATE TABLE IF NOT EXISTS `Specialities` (

  `field_id` INTEGER auto_increment,

  `field_name` VARCHAR(255) NOT NULL UNIQUE,

  PRIMARY KEY (`field_id`)

) ENGINE=InnoDB;
```

5. IMPLEMENTATION

5.1 Tools, technologies and platforms used

- **Python** and **beautifulsoap** for data scraping
- **NodeJS** for backend
- **ReactJS** for frontend and UI
- **MySql** for database
- **RabbitMQ** for task queueing
- **VsCode** as an IDE

5.2 Algorithms

Data Scraping Algorithm:

The data scraping algorithm is responsible for extracting doctors' information from various sources, such as websites or databases. It typically involves establishing connections to the data sources, retrieving the relevant data, parsing the content to extract the desired information, and formatting it appropriately. The algorithm iterates through the sources, retrieves the data, and appends it to a list or data structure for further processing.

Data Filtering Algorithm:

The data filtering algorithm is used to refine and filter the scraped data based on specific criteria or user preferences. It involves iterating through the scraped data and checking if each piece of information meets the filtering criteria. If the data matches the criteria, it is added to a filtered list or data structure, which contains only the relevant and desired information.

Search Algorithm:

The search algorithm allows users to search for doctors based on various parameters. It takes user input as the search query and iterates through the scraped data. For each doctor's information, it checks if the data matches the search query. If there is a match, the doctor's information is added to the search results list or data structure. The algorithm enables users to find doctors based on specific search criteria, such as name, specialty, location, or other relevant attributes

5.3 Standards

In our development process, we followed the principles of object-oriented design, which allowed us to encapsulate data and functionality into reusable and modular objects. This approach promotes code organization, readability, and maintainability.

Additionally, we implemented the MVC (Model-View-Controller) design pattern on the backend. This pattern separates the application logic into three components: the model, which represents the data and business logic, the view, which handles the presentation and user interface, and the controller, which manages the flow and interactions between the model and view.

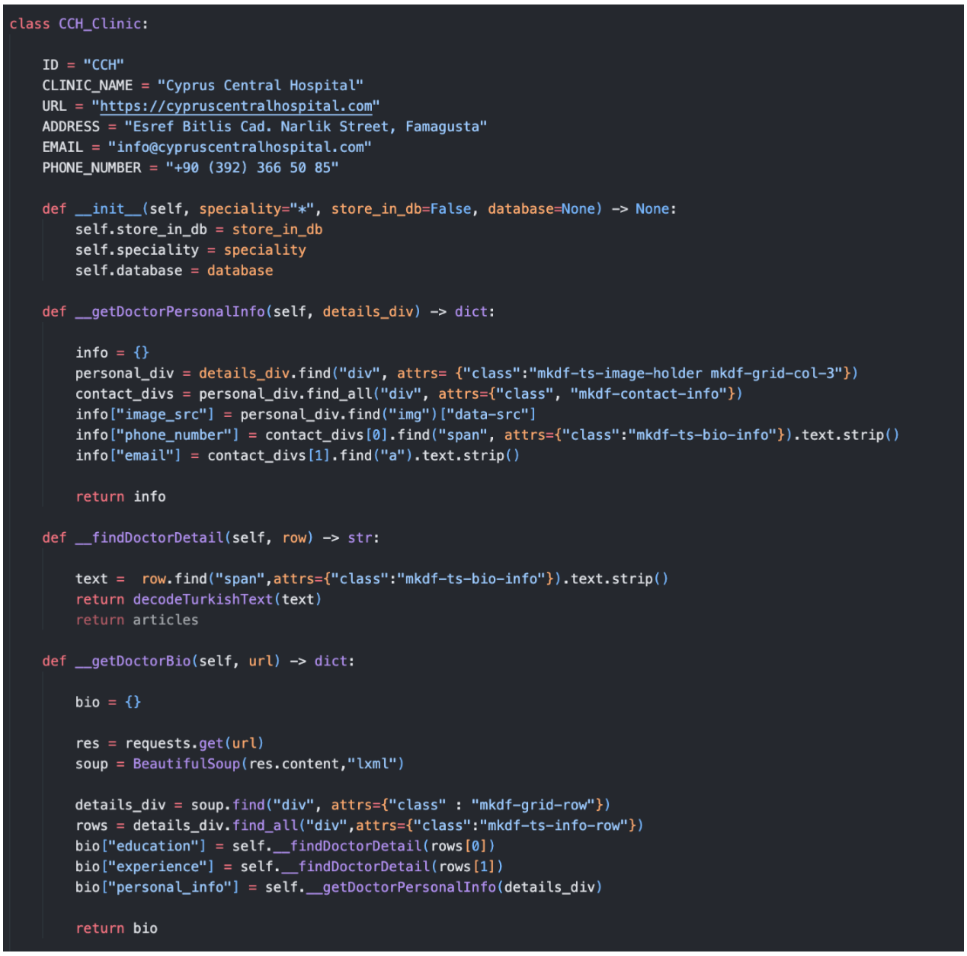
By using the MVC pattern, we achieved a clear separation of concerns, making our code more manageable and enabling us to easily make changes or enhancements to specific components without affecting the entire system. This design pattern also promotes code reusability and facilitates collaboration among team members by providing a well-defined structure for development.

5.4 Detailed description of the implementation (coding)

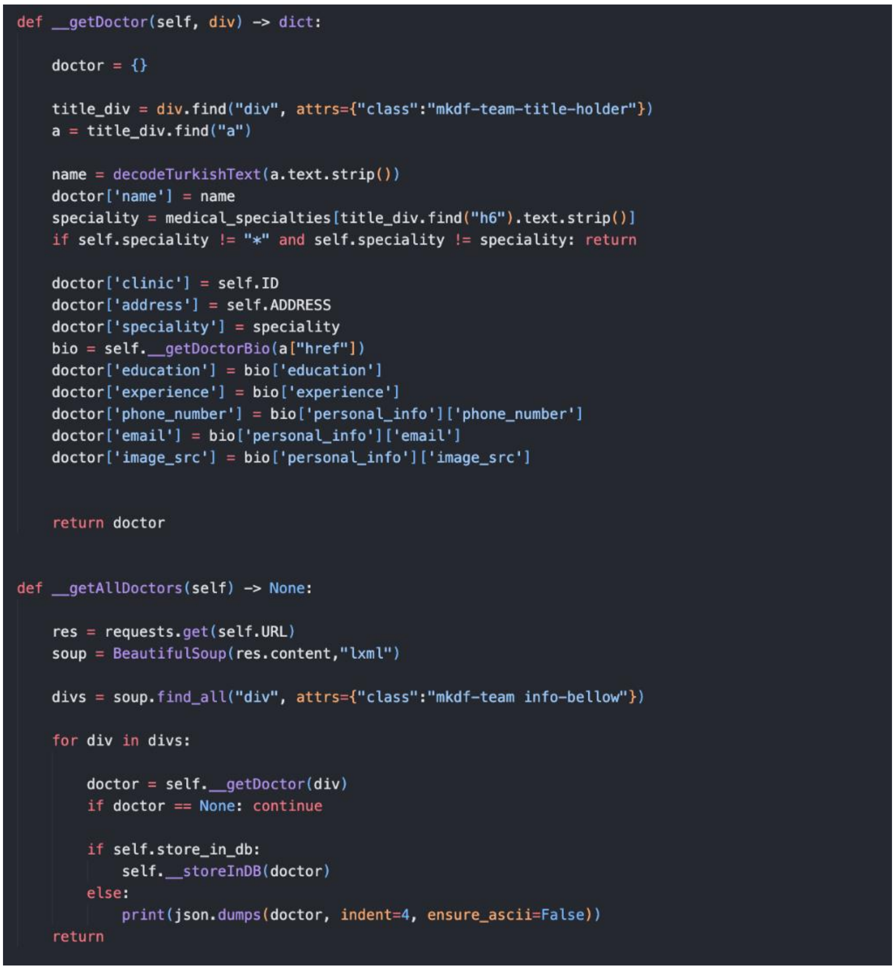
The system comprises three primary components: a scraper application, a backend, and a frontend. The frontend is responsible for presenting the user interface to both users and administrators, allowing them to view and interact with the doctor's information. The backend handles incoming requests, performs CRUD (Create, Read, Update, and Delete) operations with the database, and communicates with the scraper application. The scraper application, on the other hand, is responsible for extracting doctors' data from a clinic website and storing it in the database, ensuring the availability of up-to-date information.

Sample code of the **scraper app**:

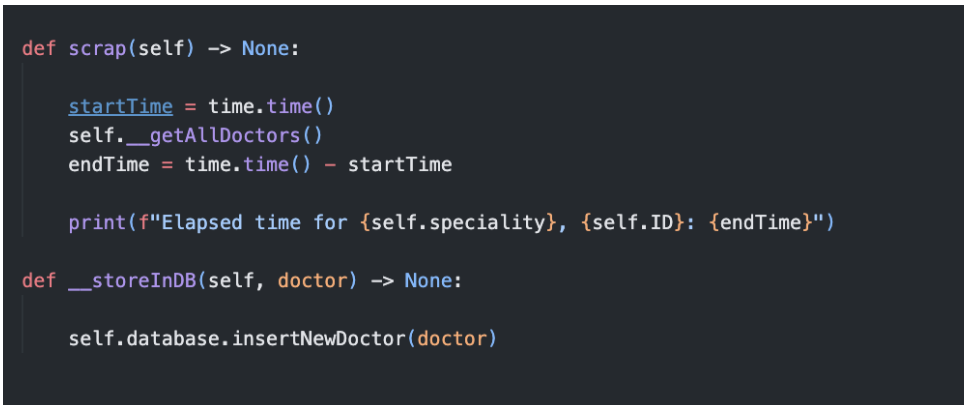
The Scraper class is a crucial component of the scraper app. It is responsible for visiting the website of General Cyprus Hospital and extracting relevant data. By leveraging tools like MySQL and BeautifulSoup, the Scraper class efficiently stores the scraped data in the database. This allows for easy retrieval and utilization of the hospital's information in the application.



```python
class CCH_Clinic:

    ID = "CCH"
    CLINIC_NAME = "Cyprus Central Hospital"
    URL = "https://cypruscentralhospital.com"
    ADDRESS = "Esref Bitlis Cad. Narlik Street, Famagusta"
    EMAIL = "info@cypruscentralhospital.com"
    PHONE_NUMBER = "+90 (392) 366 50 85"

    def __init__(self, speciality="*", store_in_db=False, database=None) -> None:
        self.store_in_db = store_in_db
        self.speciality = speciality
        self.database = database

    def __getDoctorPersonalInfo(self, details_div) -> dict:

        info = {}
        personal_div = details_div.find("div", attrs= {"class":"mkdf-ts-image-holder mkdf-grid-col-3"})
        contact_divs = personal_div.find_all("div", attrs={"class", "mkdf-contact-info"})
        info["image_src"] = personal_div.find("img")["data-src"]
        info["phone_number"] = contact_divs[0].find("span", attrs={"class":"mkdf-ts-bio-info"}).text.strip()
        info["email"] = contact_divs[1].find("a").text.strip()

        return info

    def __findDoctorDetail(self, row) -> str:

        text =  row.find("span",attrs={"class":"mkdf-ts-bio-info"}).text.strip()
        return decodeTurkishText(text)
        return articles

    def __getDoctorBio(self, url) -> dict:

        bio = {}

        res = requests.get(url)
        soup = BeautifulSoup(res.content,"lxml")

        details_div = soup.find("div", attrs={"class" : "mkdf-grid-row"})
        rows = details_div.find_all("div",attrs={"class":"mkdf-ts-info-row"})
        bio["education"] = self.__findDoctorDetail(rows[0])
        bio["experience"] = self.__findDoctorDetail(rows[1])
        bio["personal_info"] = self.__getDoctorPersonalInfo(details_div)

        return bio
```

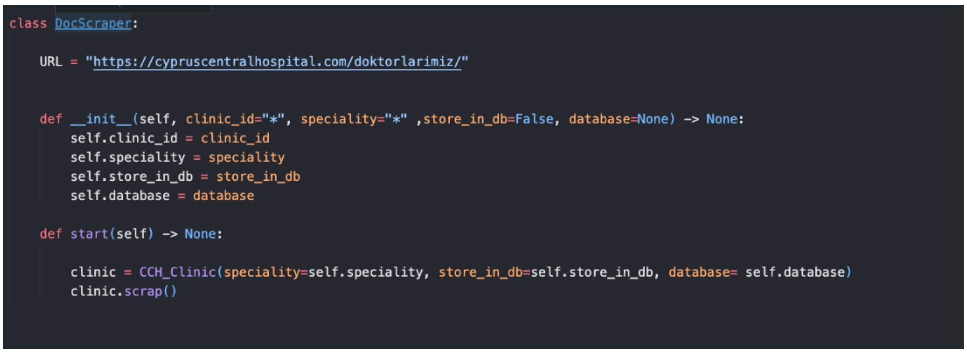


```python
def __getDoctor(self, div) -> dict:

    doctor = {}

    title_div = div.find("div", attrs={"class":"mkdf-team-title-holder"})
    a = title_div.find("a")

    name = decodeTurkishText(a.text.strip())
    doctor['name'] = name
    speciality = medical_specialties[title_div.find("h6").text.strip()]
    if self.speciality != "*" and self.speciality != speciality: return

    doctor['clinic'] = self.ID
    doctor['address'] = self.ADDRESS
    doctor['speciality'] = speciality
    bio = self.__getDoctorBio(a["href"])
    doctor['education'] = bio['education']
    doctor['experience'] = bio['experience']
    doctor['phone_number'] = bio['personal_info']['phone_number']
    doctor['email'] = bio['personal_info']['email']
    doctor['image_src'] = bio['personal_info']['image_src']


    return doctor


def __getAllDoctors(self) -> None:

    res = requests.get(self.URL)
    soup = BeautifulSoup(res.content,"lxml")

    divs = soup.find_all("div", attrs={"class":"mkdf-team info-bellow"})

    for div in divs:

        doctor = self.__getDoctor(div)
        if doctor == None: continue

        if self.store_in_db:
            self.__storeInDB(doctor)
        else:
            print(json.dumps(doctor, indent=4, ensure_ascii=False))
    return
```



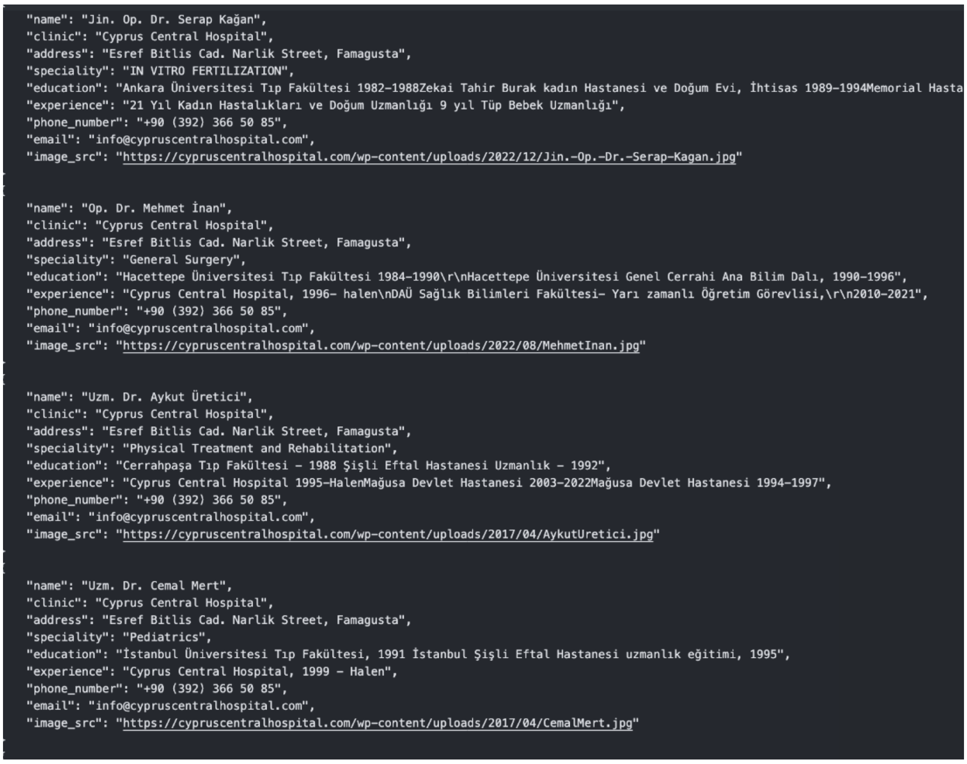
```python
def scrap(self) -> None:

    startTime = time.time()
    self.__getAllDoctors()
    endTime = time.time() - startTime

    print(f"Elapsed time for {self.speciality}, {self.ID}: {endTime}")

def __storeInDB(self, doctor) -> None:

    self.database.insertNewDoctor(doctor)
```

The DB class is a module within the scraper app that plays a pivotal role in receiving the scraped data from the Scraper class and storing it in a database. It achieves this by utilizing the MySQL Connector library in Python. The DB class establishes a connection with the MySQL database, prepares the necessary queries, and executes them to store the scraped data securely. This ensures efficient data management and retrieval for further processing and analysis within the application.



```python
class DocScraper:

    URL = "https://cypruscentralhospital.com/doktorlarimiz/"


    def __init__(self, clinic_id="*", speciality="*" ,store_in_db=False, database=None) -> None:
        self.clinic_id = clinic_id
        self.speciality = speciality
        self.store_in_db = store_in_db
        self.database = database

    def start(self) -> None:

        clinic = CCH_Clinic(speciality=self.speciality, store_in_db=self.store_in_db, database= self.database)
        clinic.scrap()
```

Data sample scraped from the website

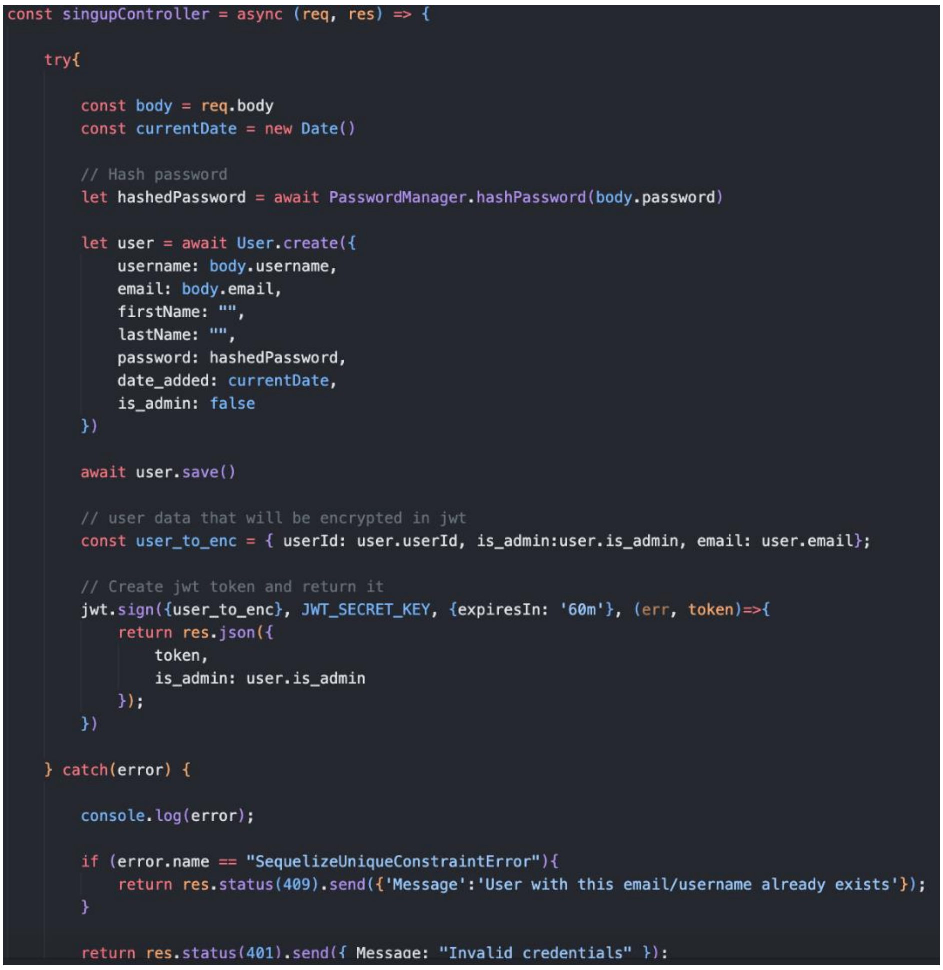


```
   "name": "Jin. Op. Dr. Serap Kağan",
   "clinic": "Cyprus Central Hospital",
   "address": "Esref Bitlis Cad. Narlik Street, Famagusta",
   "speciality": "IN VITRO FERTILIZATION",
   "education": "Ankara Üniversitesi Tıp Fakültesi 1982-1988Zekai Tahir Burak kadın Hastanesi ve Doğum Evi, İhtisas 1989-1994Memorial Hasta
   "experience": "21 Yıl Kadın Hastalıkları ve Doğum Uzmanlığı 9 yıl Tüp Bebek Uzmanlığı",
   "phone_number": "+90 (392) 366 50 85",
   "email": "info@cypruscentralhospital.com",
   "image_src": "https://cypruscentralhospital.com/wp-content/uploads/2022/12/Jin.-Op.-Dr.-Serap-Kagan.jpg"


   "name": "Op. Dr. Mehmet İnan",
   "clinic": "Cyprus Central Hospital",
   "address": "Esref Bitlis Cad. Narlik Street, Famagusta",
   "speciality": "General Surgery",
   "education": "Hacettepe Üniversitesi Tıp Fakültesi 1984-1990\r\nHacettepe Üniversitesi Genel Cerrahi Ana Bilim Dalı, 1990-1996",
   "experience": "Cyprus Central Hospital, 1996- halen\nDAÜ Sağlık Bilimleri Fakültesi- Yarı zamanlı Öğretim Görevlisi,\r\n2010-2021",
   "phone_number": "+90 (392) 366 50 85",
   "email": "info@cypruscentralhospital.com",
   "image_src": "https://cypruscentralhospital.com/wp-content/uploads/2022/08/MehmetInan.jpg"


   "name": "Uzm. Dr. Aykut Üretici",
   "clinic": "Cyprus Central Hospital",
   "address": "Esref Bitlis Cad. Narlik Street, Famagusta",
   "speciality": "Physical Treatment and Rehabilitation",
   "education": "Cerrahpaşa Tıp Fakültesi - 1988 Şişli Eftal Hastanesi Uzmanlık - 1992",
   "experience": "Cyprus Central Hospital 1995-HalenMağusa Devlet Hastanesi 2003-2022Mağusa Devlet Hastanesi 1994-1997",
   "phone_number": "+90 (392) 366 50 85",
   "email": "info@cypruscentralhospital.com",
   "image_src": "https://cypruscentralhospital.com/wp-content/uploads/2017/04/AykutUretici.jpg"


   "name": "Uzm. Dr. Cemal Mert",
   "clinic": "Cyprus Central Hospital",
   "address": "Esref Bitlis Cad. Narlik Street, Famagusta",
   "speciality": "Pediatrics",
   "education": "İstanbul Üniversitesi Tıp Fakültesi, 1991 İstanbul Şişli Eftal Hastanesi uzmanlık eğitimi, 1995",
   "experience": "Cyprus Central Hospital, 1999 - Halen",
   "phone_number": "+90 (392) 366 50 85",
   "email": "info@cypruscentralhospital.com",
   "image_src": "https://cypruscentralhospital.com/wp-content/uploads/2017/04/CemalMert.jpg"
```

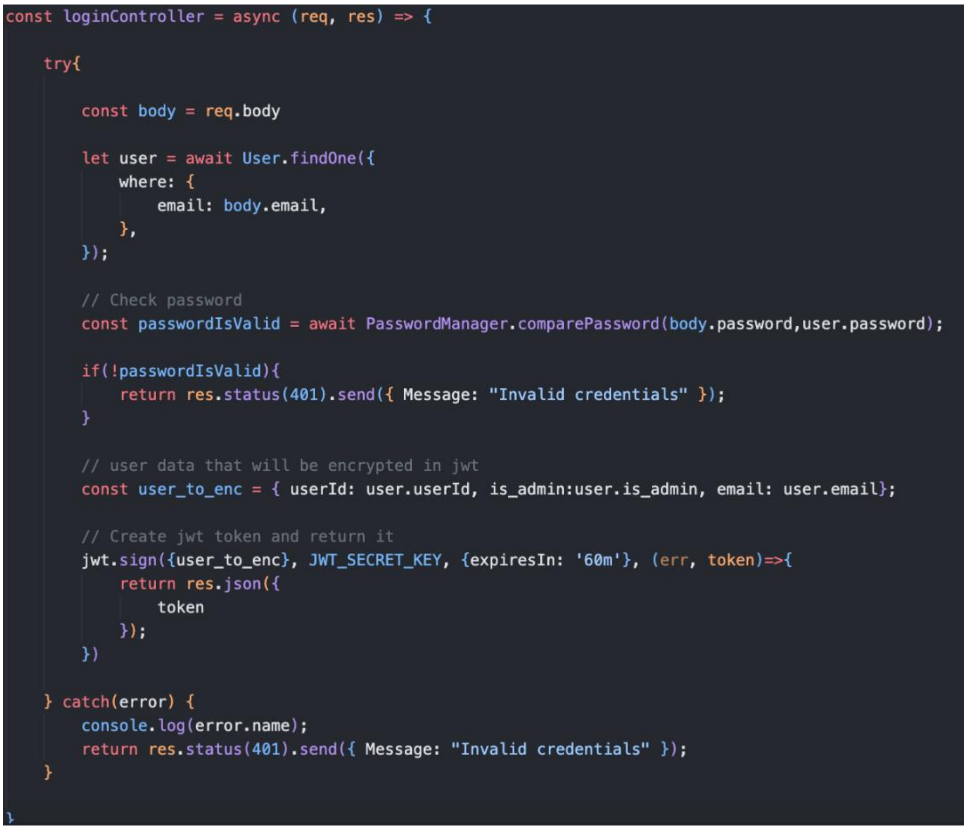
Sample code of the **backend:**

The **Signup** function allows new users to create an account by providing their desired credentials, such as a username, password, and any additional required information

The **Signin** function enables existing users to authenticate themselves by entering their registered credentials, typically their username and password

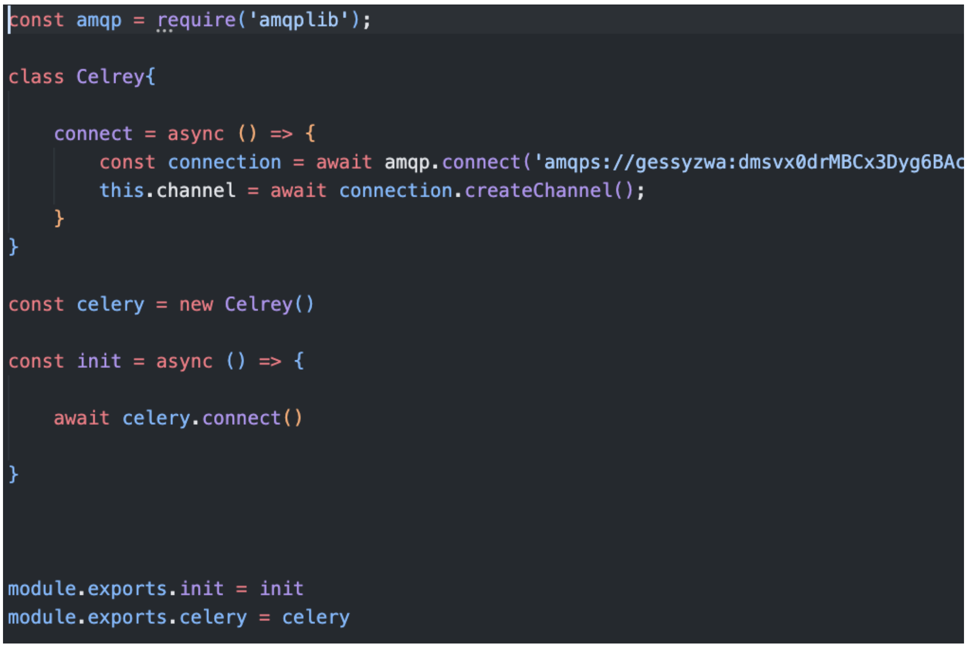


```
const singupController = async (req, res) => {

    try{

        const body = req.body
        const currentDate = new Date()

        // Hash password
        let hashedPassword = await PasswordManager.hashPassword(body.password)

        let user = await User.create({
            username: body.username,
            email: body.email,
            firstName: "",
            lastName: "",
            password: hashedPassword,
            date_added: currentDate,
            is_admin: false
        })

        await user.save()

        // user data that will be encrypted in jwt
        const user_to_enc = { userId: user.userId, is_admin:user.is_admin, email: user.email};

        // Create jwt token and return it
        jwt.sign({user_to_enc}, JWT_SECRET_KEY, {expiresIn: '60m'}, (err, token)=>{
            return res.json({
                token,
                is_admin: user.is_admin
            });
        })

    } catch(error) {

        console.log(error);

        if (error.name == "SequelizeUniqueConstraintError"){
            return res.status(409).send({'Message':'User with this email/username already exists'});
        }

        return res.status(401).send({ Message: "Invalid credentials" });
```

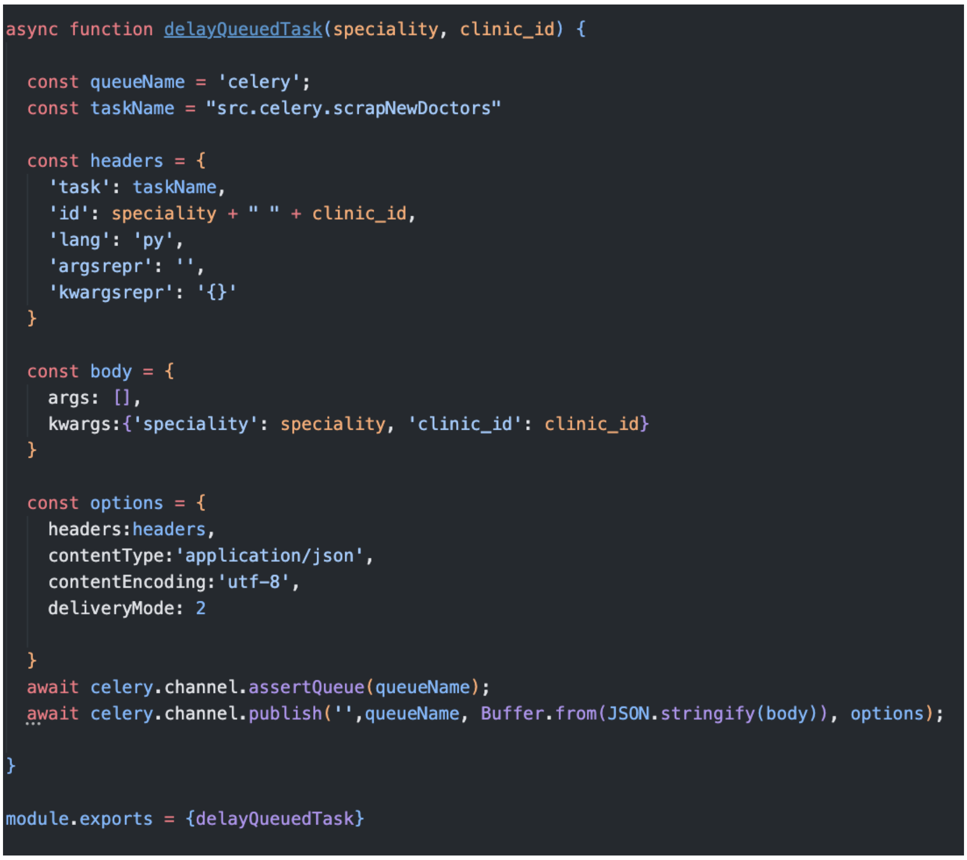


```
const loginController = async (req, res) => {

    try{

        const body = req.body

        let user = await User.findOne({
            where: {
                email: body.email,
            },
        });

        // Check password
        const passwordIsValid = await PasswordManager.comparePassword(body.password,user.password);

        if(!passwordIsValid){
            return res.status(401).send({ Message: "Invalid credentials" });
        }

        // user data that will be encrypted in jwt
        const user_to_enc = { userId: user.userId, is_admin:user.is_admin, email: user.email};

        // Create jwt token and return it
        jwt.sign({user_to_enc}, JWT_SECRET_KEY, {expiresIn: '60m'}, (err, token)=>{
            return res.json({
                token
            });
        })

    } catch(error) {
        console.log(error.name);
        return res.status(401).send({ Message: "Invalid credentials" });
    }

}
```

The **Celery** module is used to connect the backend of a system to RabbitMQ using AMQP protocol, which acts as a message broker. It enables the system to implement task queuing and distributed task execution. In the context of your application, the Celery module is responsible for queuing scraping tasks from the backend to the scraper app.

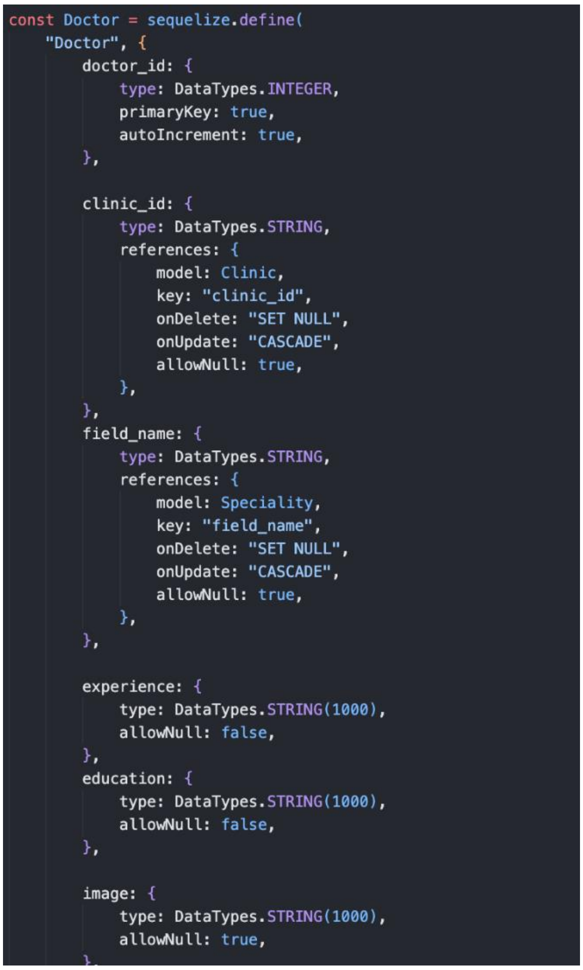


```
const amqp = require('amqplib');

class Celrey{

    connect = async () => {
        const connection = await amqp.connect('amqps://gessyzwa:dmsvx0drMBCx3Dyg6BAc
        this.channel = await connection.createChannel();
    }
}

const celery = new Celrey()

const init = async () => {

    await celery.connect()

}



module.exports.init = init
module.exports.celery = celery
```

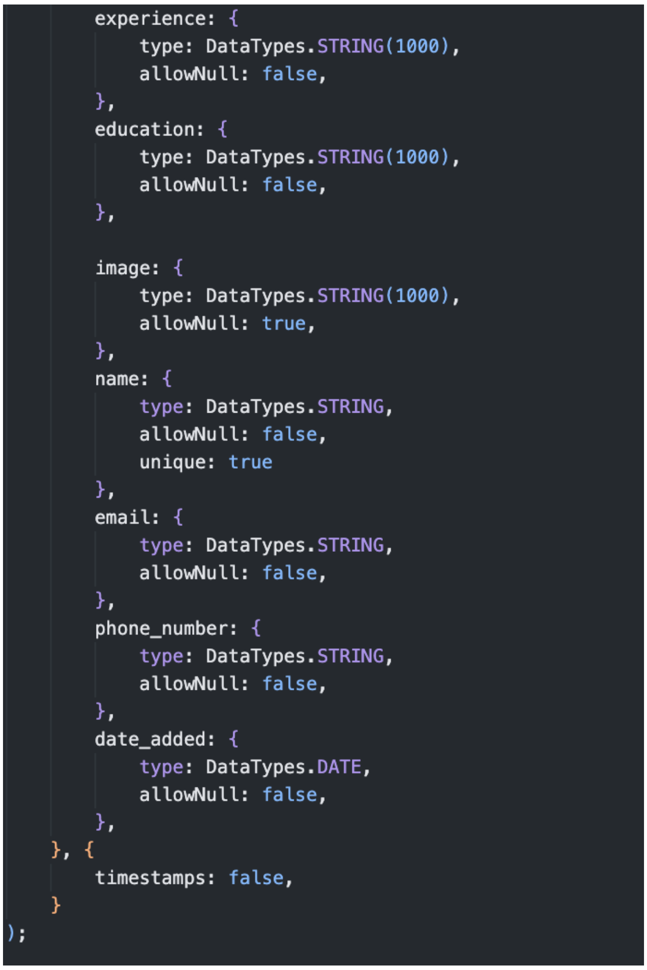


```
async function delayQueuedTask(speciality, clinic_id) {

  const queueName = 'celery';
  const taskName = "src.celery.scrapNewDoctors"

  const headers = {
    'task': taskName,
    'id': speciality + " " + clinic_id,
    'lang': 'py',
    'argsrepr': '',
    'kwargsrepr': '{}'
  }

  const body = {
    args: [],
    kwargs:{'speciality': speciality, 'clinic_id': clinic_id}
  }

  const options = {
    headers:headers,
    contentType:'application/json',
    contentEncoding:'utf-8',
    deliveryMode: 2

  }
  await celery.channel.assertQueue(queueName);
  await celery.channel.publish('',queueName, Buffer.from(JSON.stringify(body)), options);

}

module.exports = {delayQueuedTask}
```

The **Doctor** model in Node.js with Sequelize is used to define the structure of the doctor table in the database. It enables the creation of the doctor table and provides an interface for interacting with the table's data. The Sequelize library, integrated with Node.js, facilitates the implementation of the Doctor model and simplifies database operations.

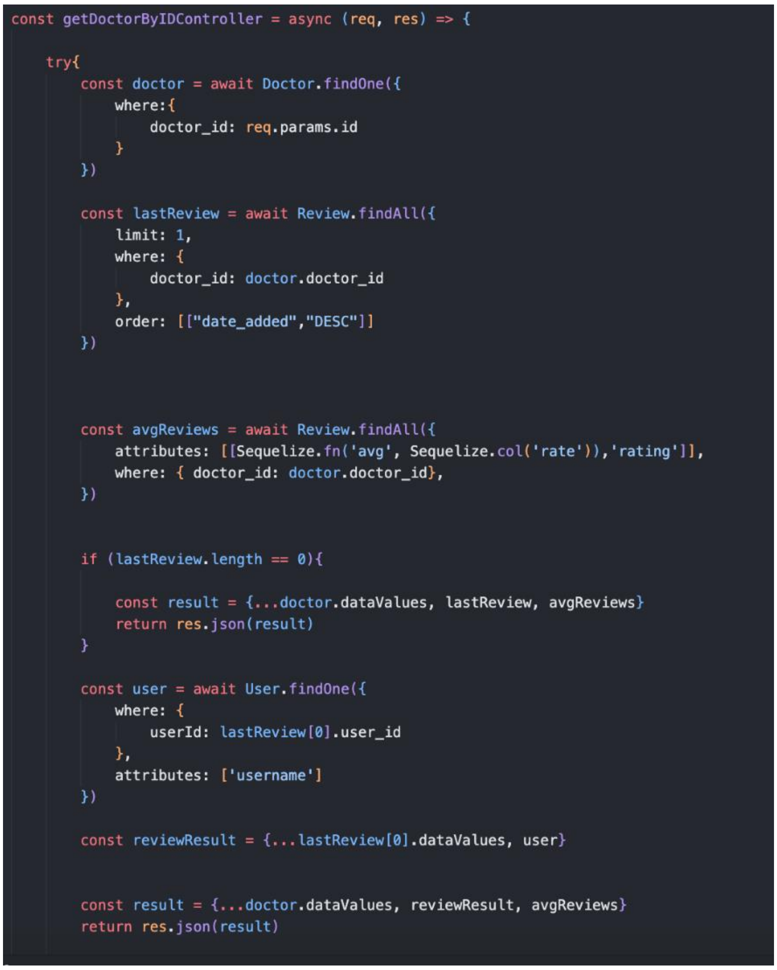


```
const Doctor = sequelize.define(
    "Doctor", {
        doctor_id: {
            type: DataTypes.INTEGER,
            primaryKey: true,
            autoIncrement: true,
        },

        clinic_id: {
            type: DataTypes.STRING,
            references: {
                model: Clinic,
                key: "clinic_id",
                onDelete: "SET NULL",
                onUpdate: "CASCADE",
                allowNull: true,
            },
        },
        field_name: {
            type: DataTypes.STRING,
            references: {
                model: Speciality,
                key: "field_name",
                onDelete: "SET NULL",
                onUpdate: "CASCADE",
                allowNull: true,
            },
        },

        experience: {
            type: DataTypes.STRING(1000),
            allowNull: false,
        },
        education: {
            type: DataTypes.STRING(1000),
            allowNull: false,
        },

        image: {
            type: DataTypes.STRING(1000),
            allowNull: true,
        },
```



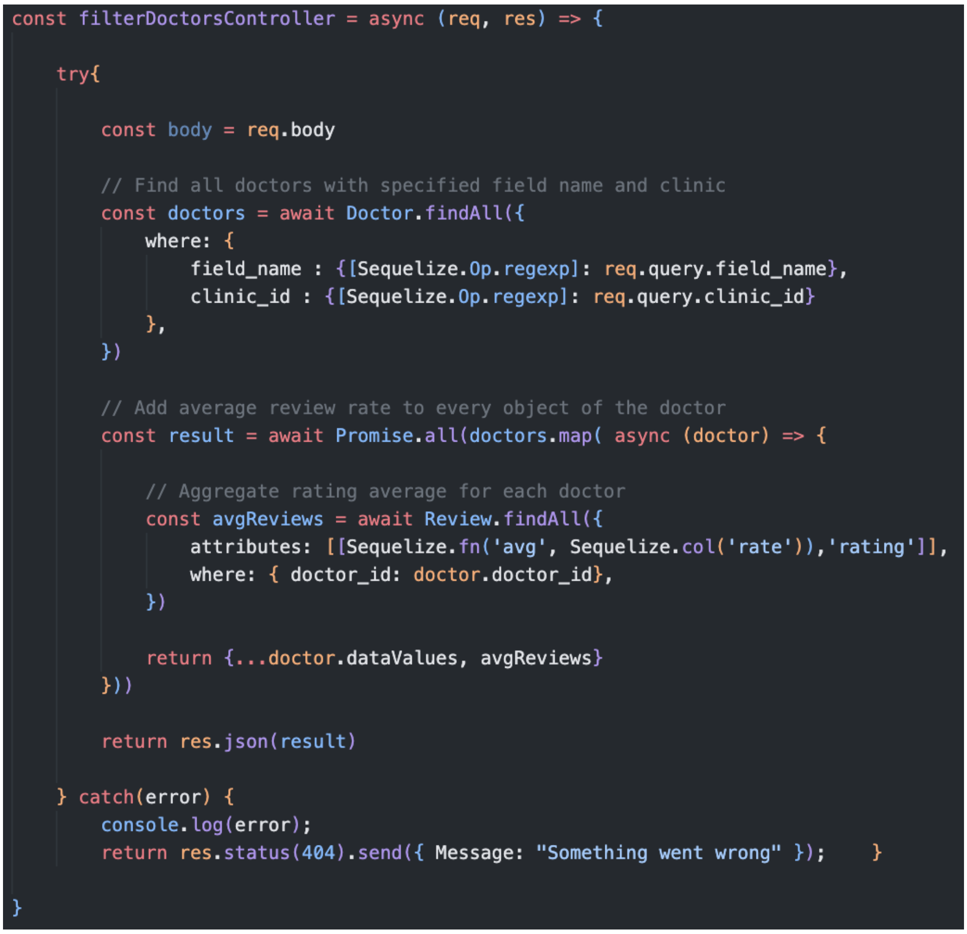
```
        experience: {
            type: DataTypes.STRING(1000),
            allowNull: false,
        },
        education: {
            type: DataTypes.STRING(1000),
            allowNull: false,
        },

        image: {
            type: DataTypes.STRING(1000),
            allowNull: true,
        },
        name: {
            type: DataTypes.STRING,
            allowNull: false,
            unique: true
        },
        email: {
            type: DataTypes.STRING,
            allowNull: false,
        },
        phone_number: {
            type: DataTypes.STRING,
            allowNull: false,
        },
        date_added: {
            type: DataTypes.DATE,
            allowNull: false,
        },
    }, {
        timestamps: false,
    }
);
```

Get doctor function, to get doctor's information from the database and display it to the user.



```javascript
const getDoctorByIDController = async (req, res) => {

    try{
        const doctor = await Doctor.findOne({
            where:{
                doctor_id: req.params.id
            }
        })

        const lastReview = await Review.findAll({
            limit: 1,
            where: {
                doctor_id: doctor.doctor_id
            },
            order: [["date_added","DESC"]]
        })



        const avgReviews = await Review.findAll({
            attributes: [[Sequelize.fn('avg', Sequelize.col('rate')),'rating']],
            where: { doctor_id: doctor.doctor_id},
        })


        if (lastReview.length == 0){

            const result = {...doctor.dataValues, lastReview, avgReviews}
            return res.json(result)
        }

        const user = await User.findOne({
            where: {
                userId: lastReview[0].user_id
            },
            attributes: ['username']
        })

        const reviewResult = {...lastReview[0].dataValues, user}


        const result = {...doctor.dataValues, reviewResult, avgReviews}
        return res.json(result)
```

Filter doctors function, to get doctors from the database and filter them.



```
const filterDoctorsController = async (req, res) => {

    try{

        const body = req.body

        // Find all doctors with specified field name and clinic
        const doctors = await Doctor.findAll({
            where: {
                field_name : {[Sequelize.Op.regexp]: req.query.field_name},
                clinic_id : {[Sequelize.Op.regexp]: req.query.clinic_id}
            },
        })

        // Add average review rate to every object of the doctor
        const result = await Promise.all(doctors.map( async (doctor) => {

            // Aggregate rating average for each doctor
            const avgReviews = await Review.findAll({
                attributes: [[Sequelize.fn('avg', Sequelize.col('rate')),'rating']],
                where: { doctor_id: doctor.doctor_id},
            })

            return {...doctor.dataValues, avgReviews}
        }))

        return res.json(result)

    } catch(error) {
        console.log(error);
        return res.status(404).send({ Message: "Something went wrong" });    }

}
```

6. TESTING

We tested the system both manually and automatically Testing. Manual testing involves executing test cases manually, while automated testing utilizes tools and scripts to automate the testing process, we also manually ensured that the data extracted are up to date and correct. When deficiencies or errors were identified, the team analyzed the root causes and implemented the necessary corrections.

7. USER GUIDE OF THE SYSTEM

7.1 Introduction:

DocFinder is a web scraping website designed to help users find doctors by scraping data from various health websites. It provides a convenient way to search for doctors and access their information quickly and efficiently.

7.2 Getting Started:

Accessing DocFinder: Open your web browser and navigate to the DocFinder website. User Registration: If required, create a user account by providing the necessary information.

7.3 User Interface:

Login Page: where the user can sign in to the system:

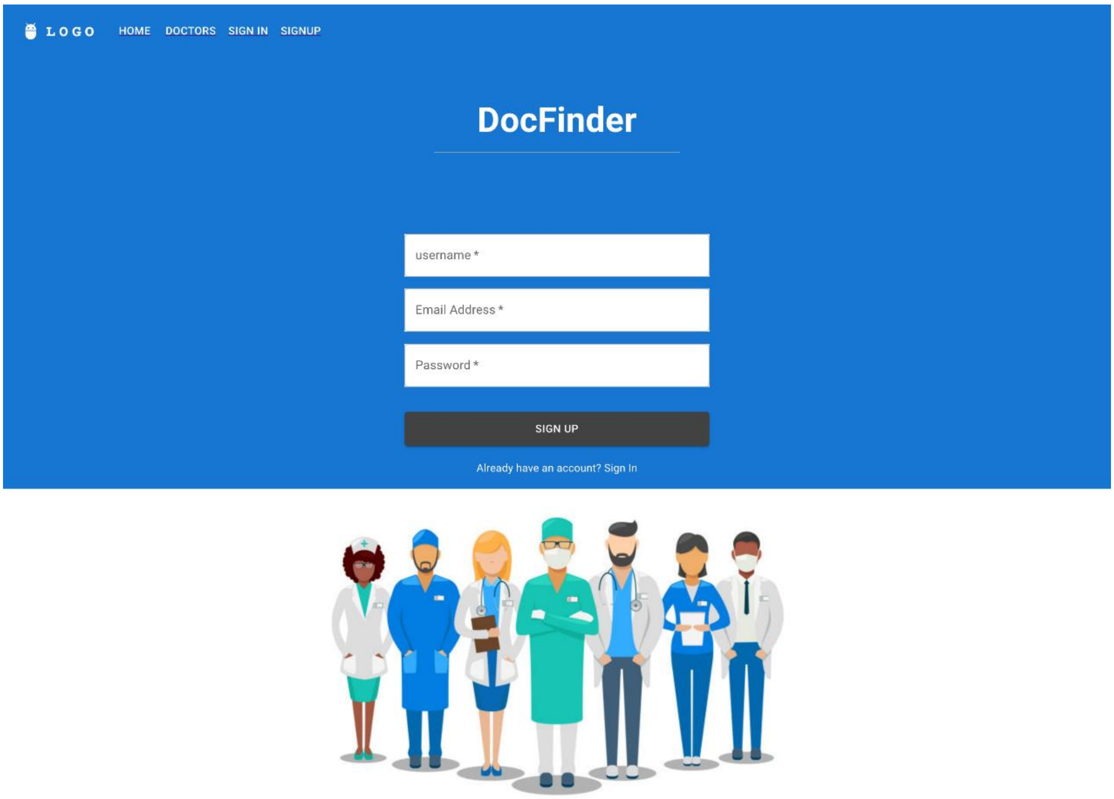


Figure 14 UI (Login Page)

Homepage: Upon accessing DocFinder, users are presented with a search bar and options to filter doctors based on location, specialty, etc.

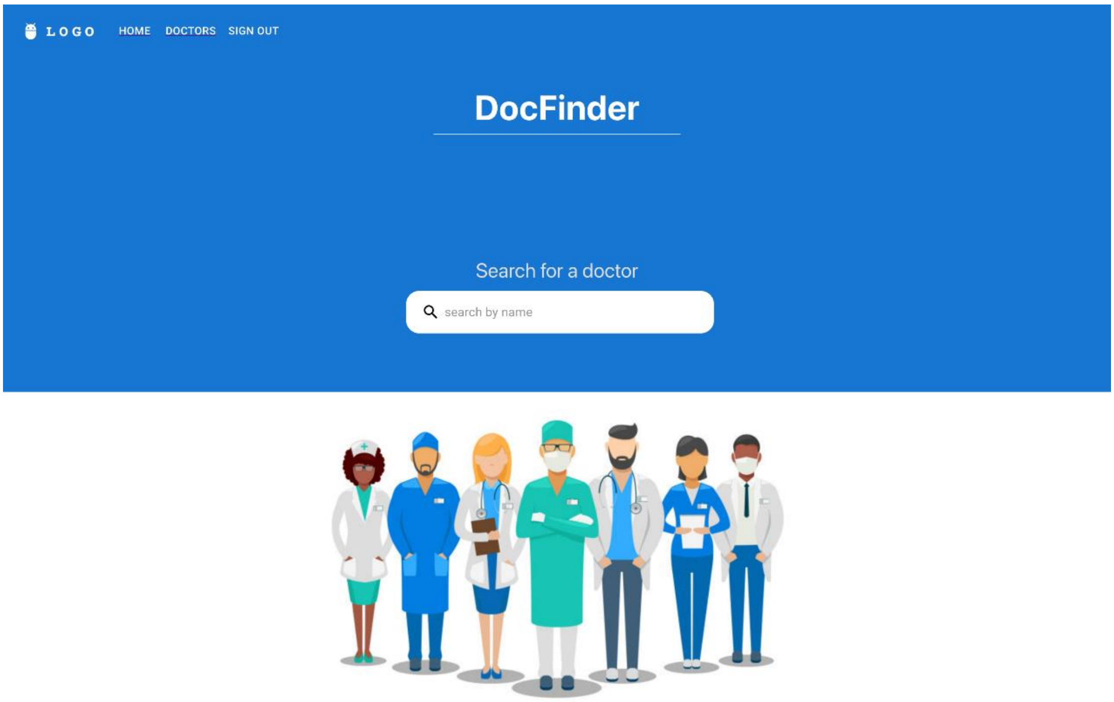
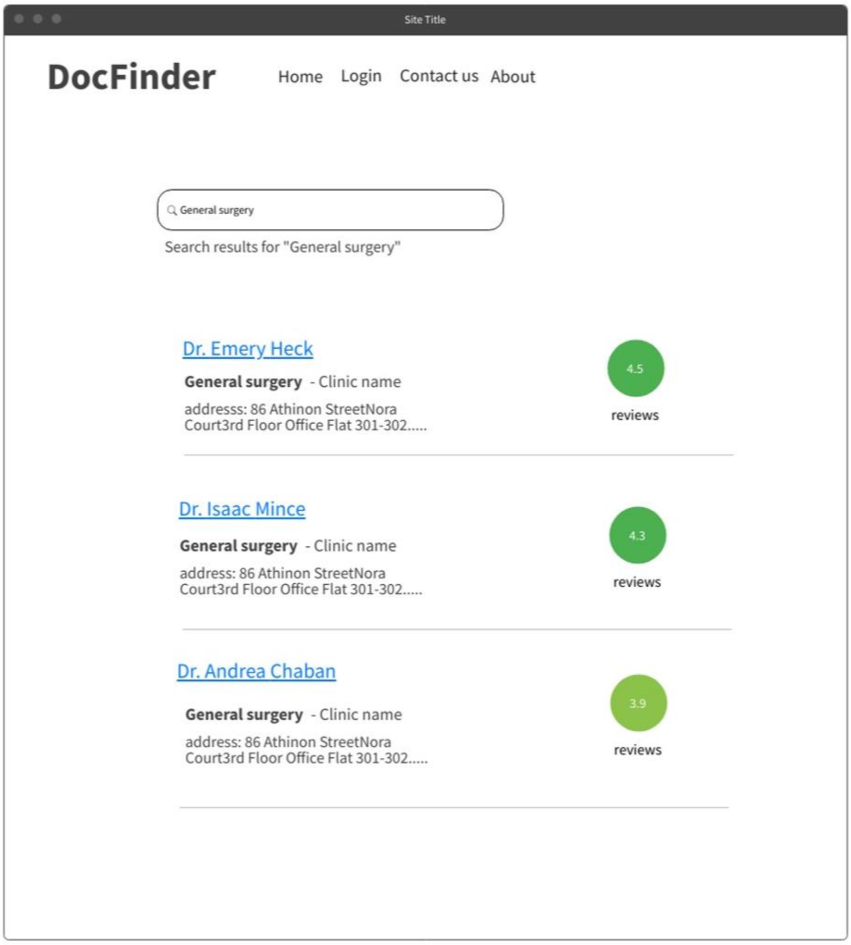


Figure 15 UI (Homepage)

Search Results: After entering search criteria, a list of relevant doctors is displayed, showing key information such as name, specialty, location, and ratings.



Site Title

DocFinder

Home Login Contact us About

General surgery

Search results for "General surgery"

Dr. Emery Heck

General surgery - Clinic name

addresss: 86 Athinon StreetNora
Court3rd Floor Office Flat 301-302.....

4.5

reviews

Dr. Isaac Mince

General surgery - Clinic name

address: 86 Athinon StreetNora
Court3rd Floor Office Flat 301-302.....

4.3

reviews

Dr. Andrea Chaban

General surgery - Clinic name

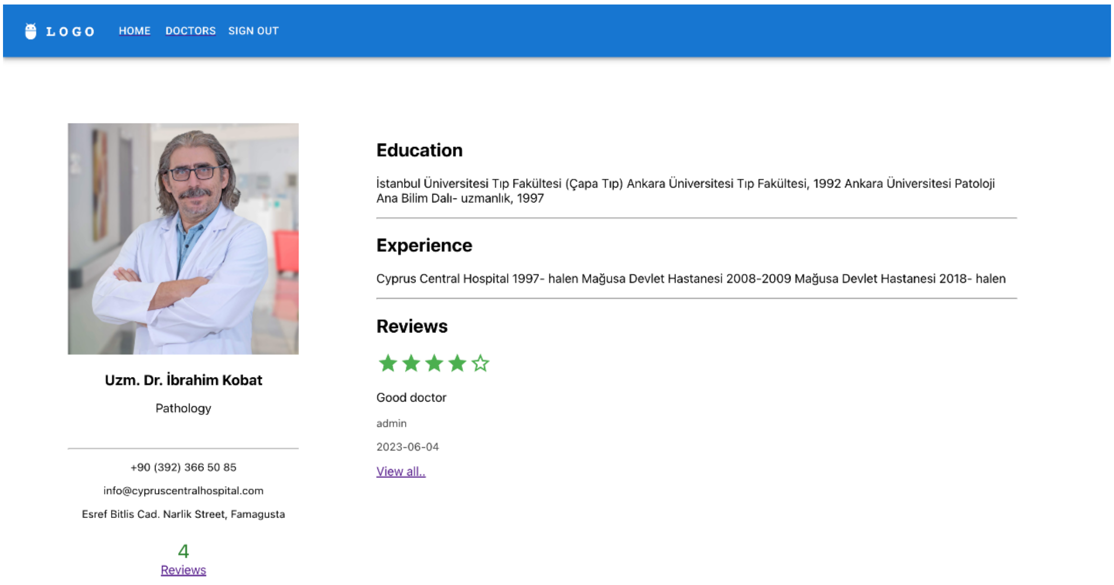
address: 86 Athinon StreetNora
Court3rd Floor Office Flat 301-302.....

3.9

reviews

Figure 16 UI (Search Results)

Doctor Profile: Clicking on a doctor's name or profile picture opens a detailed profile page with additional information, including contact details, qualifications, experience, and patient reviews.



Uzm. Dr. İbrahim Kobat

Pathology

+90 (392) 366 50 85

info@cypruscentralhospital.com

Esref Bitlis Cad. Narlik Street, Famagusta

4

Reviews

Education

İstanbul Üniversitesi Tıp Fakültesi (Çapa Tıp) Ankara Üniversitesi Tıp Fakültesi, 1992 Ankara Üniversitesi Patoloji Ana Bilim Dalı- uzmanlık, 1997

Experience

Cyprus Central Hospital 1997- halen Mağusa Devlet Hastanesi 2008-2009 Mağusa Devlet Hastanesi 2018- halen

Reviews

★★★★☆

Good doctor

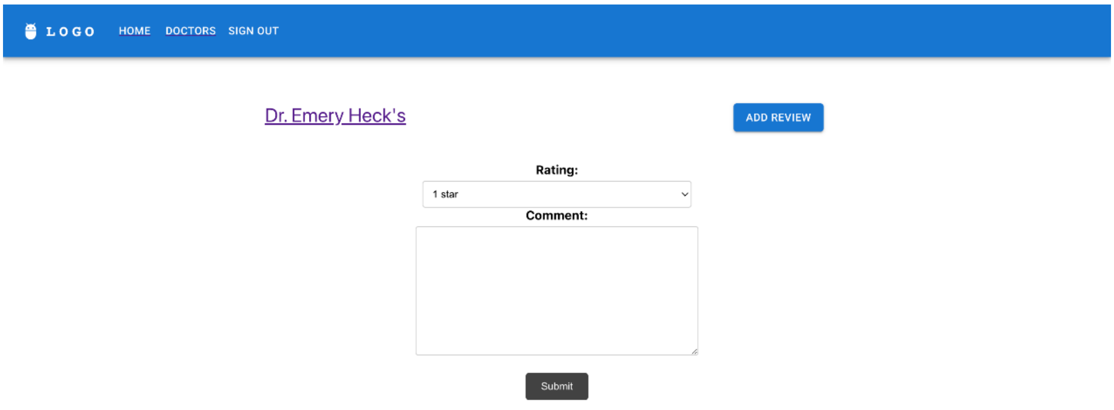
admin

2023-06-04

View all..

Figure 17 UI (Doctor Profile)

Adding a review: here is how the user can add a review to any doctor.



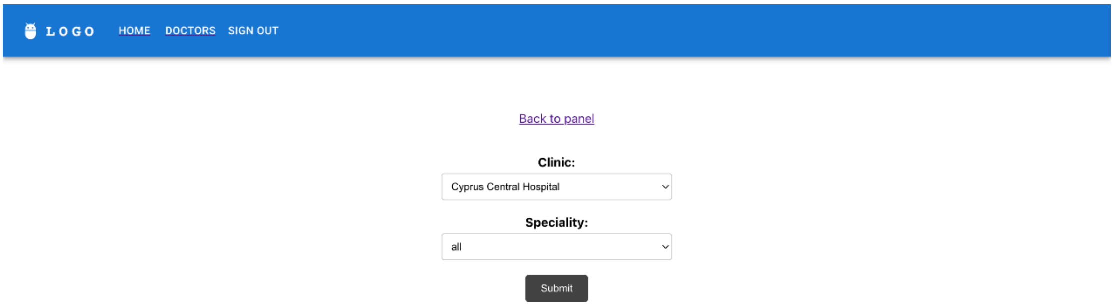
Dr. Emery Heck's

Rating: 1 star

Comment:

Figure 18 UI (Adding a review)

Request data from scrapper app:



Clinic:
Cyprus Central Hospital

Speciality:
all

Figure 19 UI (Request data)

7.4 System Functions:

- Searching for Doctors: Enter relevant keywords, such as a doctor's name, specialty, or location, in the search bar. Click the search button or press Enter to initiate the search.
- Filtering Results: Use the provided filters to refine search results based on location, specialty, ratings, etc.
- Viewing Doctor Profiles: Click on a doctor's name or profile picture in the search results to access their detailed profile page.
- Saving Favorite Doctors: Registered users can save doctors to their favorites list for easy access in the future.

7.5 Conclusion:

DocFinder is a user-friendly web scraping website that simplifies the process of finding and accessing information about doctors. With its intuitive interface, advanced search options, and comprehensive doctor profiles, users can quickly locate the right healthcare professional to meet their needs. Save time and make informed decisions with DocFinder's reliable and up-to-date data from various health websites.

For university: you can refer to the github repositories of this project:

Frontend: hsnkh12/cmse322-frontend (github.com)

Backend: hsnkh12/cmse322-backend (github.com)

8. DISCUSSION

Economic Impact: Docfinder can benefit people and society economically in multiple ways, By providing access to comprehensive data about doctors and hospitals, it enables users to make informed decisions when choosing healthcare providers. This leads to more efficient healthcare choices, potentially reducing medical expenses and improving overall healthcare outcomes. Additionally, businesses in the healthcare industry can leverage the scraped data to gain insights into market trends, competitor analysis, and strategic decision-making, leading to improved efficiency and competitiveness.

Environmental Impact: While Docfinder itself may not directly contribute to environmental conservation; it indirectly supports environmentally friendly practices. By providing users with accurate and comprehensive information about doctors and hospitals, it helps reduce the need for unnecessary physical visits or paperwork. This can result in energy savings, reduced air pollution from transportation, and decreased paper usage, contributing to a more sustainable environment.

Societal Impact: Docfinder can have a positive impact on society by improving access to healthcare information. It allows individuals to easily find and compare doctors and hospitals based on various criteria, such as specialization, location, patient reviews, and availability. This promotes transparency and empowers patients to make informed choices, fostering a patient-centric healthcare system. Additionally, comprehensive healthcare data can facilitate research, public health initiatives, and policymaking, benefiting society.

9. CONCLUSION

DocFinder aims to develop a software solution that extracts data about doctors and hospitals from various websites. The project's core functionality involves scraping information such as doctors' names, specialties, contact details, patient reviews, and hospital information. The collected data is then organized and presented in a user-friendly manner, allowing individuals to search, compare, and make informed decisions when selecting healthcare providers.

10. REFERENCES

- Davidkdryan. (2021, January 20). *Web-scraping for healthcare data: How to pull data from the internet?* Medium. https://medium.com/swlh/web-scraping-for-healthcare-data-how-to-pull-data-from-the-internet-a4353697491

- Davidkdryan. (2021, January 20). *Web-scraping for healthcare data: How to pull data from the internet?* Medium. https://medium.com/swlh/web-scraping-for-healthcare-data-how-to-pull-data-from-the-internet-a4353697491

- *How does web scraping assists in internet healthcare data collection?* (n.d.). LinkedIn. https://www.linkedin.com/pulse/how-does-web-scraping-assists-internet-healthcare-data-collection-
- *Just a moment...* (n.d.). Just a moment... https://www.tandfonline.com/doi/full/10.1080/10691898.2020.1787116
- *Web scraping*. (2023, February 3). Wikipedia, the free encyclopedia. Retrieved June 5, 2023, from https://en.wikipedia.org/wiki/Web_scraping
- *What is web scraping and how to use it?* (2021, November 9). GeeksforGeeks. https://www.geeksforgeeks.org/what-is-web-scraping-and-how-to-use-it/
- *What is web scraping? [A complete step-by-step guide]*. (2021, August 13). CareerFoundry. https://careerfoundry.com/en/blog/data-analytics/web-scraping-guide/

APPENDICES

A. Instructions for installing the system

Scraper app:

Go to https://github.com/hsnkh12/doctors-scraper-app

Clone the repo, and follow Readme.md file instructions to run the scraper

Backend:

Go to https://github.com/hsnkh12/cmse322-backend

Clone the repo, and run the backend by following Readme.md file

Frontend:

Go to https://github.com/hsnkh12/cmse322-frontend

Clone the repo, and run the frontend by following Readme.md file