University of Mumbai

DoCure: The Smart Way to Maintain your Health Records

Submitted in partial fulfillment of requirements
for the degree of

Bachelors in Technology

by

Anushka Darade
Roll No: 1814012

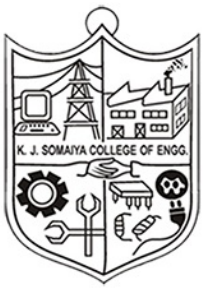
Tirth Thaker
Roll No: 1814061

Dhairya Umrania
Roll No: 1814063

Utsav Parekh
Roll No: 1924002

Guide:

Prof. Nandana Prabhu



Department of Information Technology
K.J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Batch 2018-2022

K.J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Certificate

This is to certify that the dissertation report entitled **DoCure: The Smart Way to Maintain your Health Records** submitted by Anushka Darade, Tirth Thaker, Dhairya Umarania and Utsav Parekh at the end of semester VIII of LY B. Tech is a bona fide record for partial fulfillment of requirements for the degree of Bachelors of Technology in Information Technology of University of Mumbai.

Guide

Head of Department

Principal

Date:
Place: Mumbai-77

K.J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Certificate of Approval of Examiners

We certify that this dissertation report entitled **DoCure: The Smart Way to Maintain your Health Records** is bona fide record of project work done by Anushka Darade, Tirth Thaker, Dhairya Umarania and Utsav Parekh. This project is approved for the award of Bachelors in Technology Degree in Information Technology of University of Mumbai.

_____
Internal Examiner

_____
External Examiner

Date:
Place: Mumbai-77

K.J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

DECLARATION

We declare that this written thesis submission represents the work done based on our and / or others' ideas with adequately cited and referenced the original source. We also declare that we have adhered to all principles of intellectual property, academic honesty and integrity as we have not misinterpreted or fabricated or falsified any idea/data/ fact/source/original work/matter in my submission.

We understand that any violation of the above will be cause for disciplinary action by the college and may evoke the penal action from the sources which have not been properly cited or from whom proper permission is not sought.

Signature of the Student **1814012** **Roll No.**	**Signature of the Student** **1814061** **Roll No.**
Signature of the Student **1814063** **Roll No.**	**Signature of the Student** **1924002** **Roll No.**

Date:
Place: Mumbai-77

Abstract

Health is an important aspect of every life. For all the necessary records related to health, medical records are employed. Traditionally these are in the form of paper and can often be misplaced, or suffer from tear and damage The project aims at resolving this problem by using an Electronic Health Record system that scans and stores the medical report data on the database for anytime access from the user. Users can directly upload their report in the form of images or PDFs, that are processed and the data is stored in the database. When the user wants to access this data, they can directly see the data as well as several analysis that are available on the platform. The proposed approach is designed keeping in mind the different technological intricacies involved to provide a fast, secure, and most importantly a very user-friendly application that can be operated with ease.

Keywords: Optical Character Recognition, Portable Document Format, Electronic Health Record, Pytesseract.

Contents

List of Figures

List of Tables

Nomenclature

AI Artificial Intelligence

API Application Program Interface

CBC Complete Blood Count

$CRUD$ Create, Read, Update and Delete

EHR Electronic Health Record

OCR Optical Character Recognition

$OLAP$ Online Analytical Processing

PDF Portable Document Format

SQL Structured Query Language

Chapter 1

Introduction

This chapter gives a brief introduction to the thesis presented in manner of the BE Project undertaken.The problem definition is stated and is followed by motivation and scope of the thesis.Lastly,the organization of the thesis is written to help understand the entire document.

1.1 Problem Definition

Revolving around today's era, there is a lot of medical research surfacing new diseases, new symptoms as well as new medical reports. Layman faces several problems in maintaining their medical data efficiently and updating it time to time without loss of any reports. These reports are not only required for medical purposes but in the pandemic, such reports hold a lot of significance for daily activities. In such a situation, having an alternative to store medical reports, and related data eases several aspects for people. In an ideal case, the reports are to be maintained appropriately and latest reports be available for doctor consultation.
Some of the commonly noticed problems that have been discovered are:

- Frequent expensive consultations
- Loss of medical data due to inefficient maintenance
- Irregular updation of data
- Difficult access of data by patients and doctors
- Improper interpretation of report data
- Insecure storage of data

DoCure will be a Electronic Health Record (EHR) system that will have the ability to read patients' reports from both PDF and image formats and storing them securely in an encrypted format available to the user at any given time. The user can access the reports and view them in a dashboard format. Apart from this, they can get doctor consultation for the reports that they request.

1.2 Motivation of the thesis

Medical health covers several different and important aspects of human life. Medical reports are maintained by paper basis which are traditionally exposed to wear and tear. Due to several problems faced in maintenance and updating of medical data, there is an inaccuracy in the provision of health services which can often prove to be fatal. We are attempting to reduce this issue by the introduction of an Electronic Health Record system which stores the data in a digital format. Our project aims at helping those who aren't well versed with medical jargon to understand and analyse their reports in an easier and simpler way.

1.3 Scope Of the Project

The scope of our project covers laymen and people who are connected to the medical industry. We aim at easing their work while storing and retrieving medical data history. Our project deliverable include an application that can take user reports as input, read them and detect medical data variables, store the data securely and make it available for the user on demand. Our centralized task is to create a system where the user doesn't h ave to undergo the hassle of searching their reports and the latest data is updated automatically and easily accessible. We also aim to provide them with additional insights about their report that they aren't able to realise. This can even help doctors and researchers in quick access of medical data for research.

1.4 Organization of the Synopsis

The synopsis is the overall documentation of the project research and documentation performed in semester VII and VIII. The synopsis begins with the abstract of the project to provide an essence. This is followed by the table of contents and the list of figures and tables. The introduction covers the problem definition, the motivation and the scope of the project. The literature survey provides an overview of the several research papers that we have centrally used as anchors to design our project idea. This is followed by the Software Project Management Plan. This section of the document includes the overview and deliverable of the project. The process model, roles and responsibilities of each member and the tools and technology used. Lastly it contains the Tasks, Assignments and Timeline of completion of the implementation. Following this is the Software Requirements Specification which comprises the different requirements of the project. Next is the Software Design Document which comprises of the different component description, data flow diagrams and system architecture information of the project. Followed by this is the Implementation where we have covered the several aspects of the project came to life. Lastly the Software Test Document, comprises of the test approach, the test plan, and the test cases of the different features of the project.

Chapter 2

Literature Review

This chapter provides a comprehensive literature survey used for the designing and completion of the project. Papers from reputed journals such as IEEE and Springer are referenced and cited here.

Paper Title	**Survey**
Building Structured Personal Health Records from Photographs of Printed Medical Records(2015)	This paper talks about how to use OCR to scan patient records from images clicked from mobile phones in China. In this paper, the OCR was performed using IBM Datacap Taskmaster Capture 8.11 For medical term entities, a dictionary-based approach was used to detect the matches of dictionary terms in text, which required a dictionary with high coverage and good quality. The system could identify over 80
Custom Made Medical Data Reporting Tool(2009)	In this paper, methods of medical data extraction from databases/data warehouses using OLAP tools, and then generating a report from that data is mentioned. The authors of this paper have created their own Custom made Medical report generation software, which does the above said things and generates a report for the same.
Text Detection and Recognition for Images of Medical Laboratory Reports with a Deep Learning Approach	This paper explains how various CNN techniques can be used to extract text from documents. It also compares the results of the different CNN techniques being used.

OCR with Tesseract, Amazon Textract, and Google Document AI: a benchmarking experiment	This paper explores 3 different OCR engines: Tesseract, Amazon Textract and Google Document AI on images of English and Arabic document images. The paper concluded that Document AI had the lowest error, followed by Textract, followed by Tesseract. The error rate went up for all the OCR engines when noise was implemented in the dataset. Another observation was that the engines performed worse for Arabic documents than English documents.
Study of Information Extraction and Optical Character Recognition (2017)	This paper explains different ways of text mining, and various tools which can be used for them. It also talks about OCR, its uses, benefits and disadvantages.
Extract and analysis of semi structured data from websites and documents	This paper proposes a method to extract data from semi structured PDFs. The pipeline they have proposed is as follows: The PDF file will be passed to a PDF parser. After that, a line picker will iterate through the text line by line, and on each line there will be a boundary extractor which will extract the boundaries, and a pattern matcher such as Regular Expressions will be used to locate text that is to be extracted. The extracted text will be entered into the database
An OCR Post-correction Approach using Deep Learning for Processing Medical Reports	Using OCR for medical reports can prove to be less accurate because the terms used in medical reports are not common words which are given to the OCR to identify from. This paper attempts to solve this by creating a Deep Learning Processing model to train an OCR with all the medical terms to improve the accuracy of the OCR algorithm. This paper achieved to get lower WER and CER using their deep learning post processing model, and concluded that training an OCR with domain specific terms/words proves to improve accuracy.

Table 2.1: Literature Survey

Chapter 3

Software Project Management Plan

This chapter illustrates the Software Project Management Plan document. Content for project deliverable, project organization, roles and responsibilities of the team members, tools and techniques to be used and finally the essential tasks and phases of the project; is presented here. The Gantt charts for project schedule are attached in this chapter.

3.1 Introduction

3.1.1 Project Overview

The Goal of our project is to help patients understand their diagnostic reports with ease Patients can input their reports in the system in the form of an image or PDF. This will be read via OCR and PDF recognition technique in our system and the parameters can be simplified for the user to understand. Maintenance of the medical data via ease of updating Our system will house a database that will be made which will store all the report data of the patient. When the patient receives new reports, they can upload them to our system, and the database will update and display their latest records.

3.1.2 Project Deliverable

1. Product Problem Statement
2. Software Requirement Specification
3. Software Design Document
4. OCR
    - Image
    - PDF
5. Database CRUD Operations
    - Create new user
    - Retrieve data

- Display fetched report data (decryption)
- Store data (encryption)

6. Report processing
7. Dashboard
    - Report analysis
    - Data visualization
8. User Interfaces
9. Software Test Document

3.2 Project Organisation

3.2.1 Software Process Model

Docure will be developed using the Concurrent Process Model. Figure 1 provides a schematic representation of one software engineering activity within the modeling activity using a concurrent modeling approach. The activity modeling may be in any one of the states noted at any given time. For example,early in our project the communication activity has completed its first iteration and exists in the awaiting changes state. The modeling activity which existed in the inactive state while initial communication was completed, will now make a transition into the under development state. If, however, the customer indicates that changes in requirements must be made, the modeling activity moves from the under development state into the awaiting changes state. The event analysis model correction, will trigger the requirements analysis action from the done state into the awaiting changes state.

Since the features of the model are well suited with the development process that we are implementing, we are using the Concurrent Process Model. At the time of finalizing the conduct of development, a particular module will be in the inactive state, which after receiving clearance, will begin its development i.e. will enter under development phase. After different rounds for revision, changes and review the process will be base-lined and marked as 'Done'. The model offers flexibility for our project as different modules are being developed by different micro teams within our team.

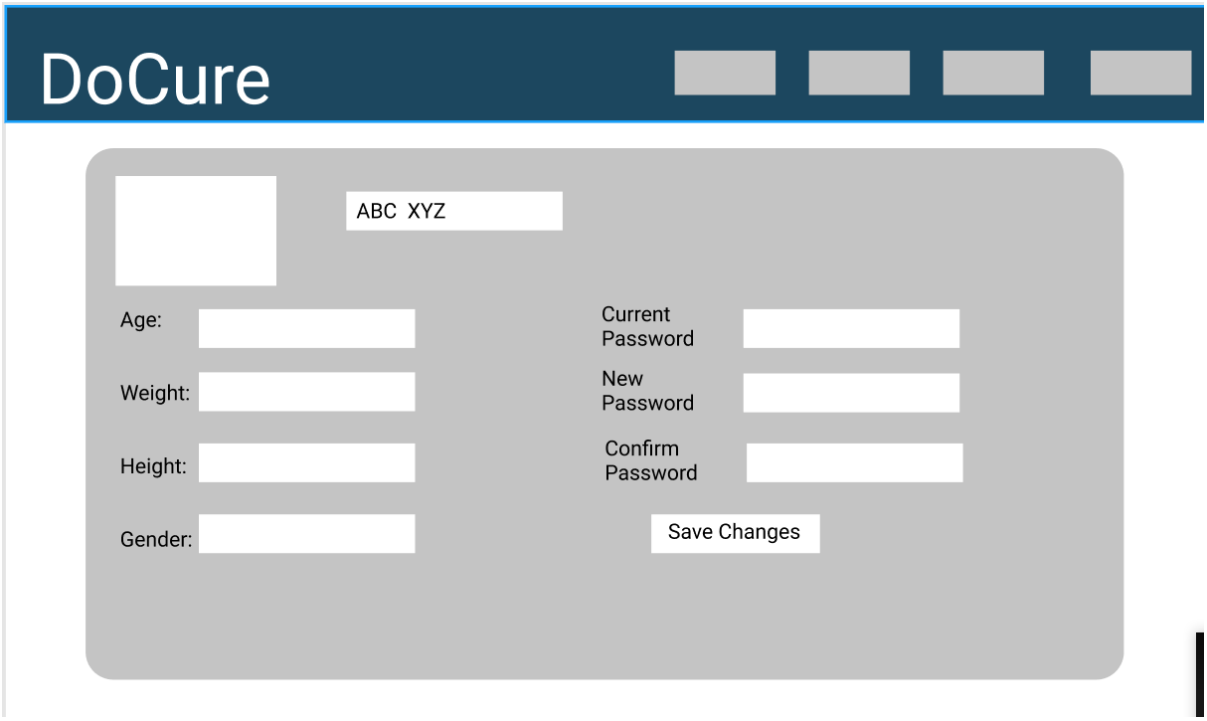


Figure 3.1: Concurrent Process Model

3.2.2 Roles and Responsibilities

- **Project Manager (Utsav Parekh):** Project Manager is responsible for working and assigning duties to the team members. Communication with faculty about progress and timely execution of the assignments. Establish the vision and scope of the project. Develop a plan to manage the project,gather the requirements,and document the plan.
- **Designer (Dhairya Umrania, Tirth Thaker):** Designer works as part of a collaborative development team to manage the look-and feel of the project,and ensure that the project's GUI adheres to design standards. Analyze user needs and developer requirement specifications.Transform requirements into software design documentation.
- **Developer (Dhairya Umrania, Anushka Darade, Tirth Thaker, Utsav Parekh):** Developers get familiar with applications and resources that may be available. Support to make the team more effective, and learn about the languages that the team will use. Code for application and programs for backend processing systems to build a working project fulfilling the requirements specified in System Requirements Specification..
- **Tester (Dhairya Umrania, Anushka Darade, Tirth Thaker):** Document issues, determine testing methodologies and responsibilities, and manage testing

schedules. Develop and define the system test plan and preparing use case procedures.

3.2.3 Tools and Technology:

- Tools
  - Various text editors, Overleaf and ShareLaTeX for documentation.
  - StarUML and LucidChart to make all the UML diagrams.
  - GanttPro (an online tool to make Gantt charts) to map the process scheduling Gantt chart.
  - Visual Studio Code for programming.
  - XAMPP control panel to access databases.
- Technologies
  - Python for base coding alongwith libraries.
  - Django for Python back-end with database.
  - Spacy
  - OpenCV for image processing and OCR
  - Pytesseract for OCR processing.
  - Bootstrap for the front-end of the application.
  - SQLite Database

3.3 Project Management Plan

3.3.1 Tasks:

3.3.1.1 Defining a Business Case

- **Description :** Brainstroming ideas for projects, and doing research on all of them and shortlisting the best one.
- **Deliverables and Milestones :** Problem Definition Draft
- **Resources needed :** Internet, Web Browser and Text Editor
- **Dependencies and Constraints :** Project should be feasible
- **Risks and Contingencies :** Project may look feasible from outside, but may not be feasible after conducting research

3.3.1.2 Finalizing Problem Statement

- **Description :** Defining a problem statement for the shortlisted topic in the previous step.
- **Deliverables and Milestones :** Problem Definition
- **Resources needed :** Google docs, Text Editor, Internet
- **Dependencies and Constraints :** Problem definition should be well defined and understood.
- **Risks and Contingencies :** Problem statement could be misunderstood.

3.3.1.3 Project Approval

- **Description :** Making a short presentation of project topic where we explain the project in brief and gain approval for it from teachers.
- **Deliverables and Milestones :** Brief project PPT and presentation.
- **Resources needed :** PowerPoint, internet, google drive.
- **Dependencies and constraints :** Presentation must be accurate description of project.
- **Risks and Contingencies :** Project topic could be rejected.

3.3.1.4 Project Documentation

- **Description :** Creation of documentation required for the project, such as: SPMP, SRS, SDD and STD.
- **Deliverables and Milestones :** SPMP, SRS, SDD, STD.
- **Resources needed :** Text Editor, LaTeX, Internet.
- **Dependencies and constraints :** Documentation must be well defined and planned and provide with a blueprint for the project.
- **Risks and Contingencies :** Documentation may be ill-defined or may not be accurate representation of the project.

3.3.1.5 Dataset preparation

- **Description :** Generating a self-made dataset to test and tweak OCR, and finding online datasets for CBC report, and other medical report data.
- **Deliverables and Milestones :** Dataset for OCR, and CBC
- **Resources needed :** Camera, Medical Reports, Google Drive, Photo Editor

- **Dependencies and constraints :** Dataset must contain all the relevant data to its corresponding module.
- **Risks and Contingencies :** Data may be insufficient or incorrect. Garbage in garbage out result.

3.3.1.6 Finalization and implementation of techniques/tools for OCR for images and PDF data extraction

- **Description :** : Implementing various tools for OCR for extracting data using it
- **Deliverables and Milestones :** :Testing OCR
- **Resources needed :** PDF, Images, Text Editor
- **Dependencies and constraints :** Quality of Images and PDFs should be good for extraction
- **Risks and Contingencies :** Incorrect data could be extracted.

3.3.1.7 Front-end development for modules developed

- **Description :** :Implementing Dashboard and web design using front end technology and bootstrap framework
- **Deliverables and Milestones :** :Web pages of implemented modules
- **Resources needed :** Text Editor, Internet, Browser, Django, Bootstrap
- **Dependencies and constraints :** All front End module should properly integrated and implemented
- **Risks and Contingencies :** Web design needs to be compatible with back-end.

3.3.1.8 Database implementation and related operations

- **Description :** Implementing of database schema to store user report data and its CRUD functions.
- **Deliverables and Milestones :** Database backend of project.
- **Resources needed :** Text Editor, Internet, MySQL, Django, Xampp
- **Dependencies and constraints :** Database should have all the necessary parameters.
- **Risks and Contingencies :** Database may be poorly implemented.

3.3.1.9 Implementation of encryption module

- **Description :** : Implementing Encryption in our work using algorithms like RS4,Hash Functions To provide security
- **Deliverables and Milestones :** Encryption of data to avoid third party attacks
- **Resources needed :** Text Editor, Storage Space to store Encryption keys, Internet
- **Dependencies and constraints :** Encryption Algorithms Should properly integrated into system.
- **Risks and Contingencies :** After Hosting ,Encryption algorithms should work to have security to data

3.3.1.10 Implementation of analytics and interfacing with related front-end

- **Description :** Implementation of the analytics module, where we create a dashboard for users so that they can interpret their medical reports data and also make the necessary front-end for the same.
- **Deliverables and Milestones :** Analytics dashboard module and its front-end.
- **Resources needed :** Text Editor, Django, MySQL, Xampp, Web Browser, Internet.
- **Dependencies and constraints :** Dashboard and UI must be convenient to use and see.
- **Risks and Contingencies :** Analytics module may be incorrectly implemented, UI may not have come out as intended.

3.3.1.11 UI development for rest of modules and refining already implemented UI

- **Description :** :Develop the UI for remaining modules such as Database and Analytics Dashboard.
- **Deliverables and Milestones :** : Front-end of DoCure Website
- **Resources needed :** Text editor, Django, MySQL, Xampp
- **Dependencies and constraints :** UI needs to be convenient to use and must cover all the specified functions.
- **Risks and Contingencies :** Flow of website must be correct. UI must be an accurate representation of the data.

3.3.1.12 Testing of individual modules

- **Description :** :All the modules are to be tested individually to ensure all functions and features are working properly.
- **Deliverables and Milestones :** :Individually tested modules.
- **Resources needed :** Text-editors, website testing tools such as Selenium or Jmeter.
- **Dependencies and constraints :** All modules must be tested individually, seperate from other modules.
- **Risks and Contingencies :** Modules may not work as intended.

3.3.1.13 Module integration into single system

- **Description :** :All the modules are to be integrated into one single system.
- **Deliverables and Milestones :** :Integrated system of all modules
- **Resources needed :** Internet, Web Browser and Text Editor, Django, MySQL, Xampp
- **Dependencies and constraints :** All modules must be integrated seamlessly.
- **Risks and Contingencies :** Modules may malfunction after integration or errors may occur.

3.3.1.14 Testing of system

- **Description :** Full system is to be tested as a whole to ensure proper working and functioning of entire system
- **Deliverables and Milestones :** Fully tested system
- **Resources needed :** Testing tools
- **Dependencies and constraints :** Testing must be done on full system as one single unit.
- **Risks and Contingencies :** Errors and bugs might appear.

3.3.1.15 Product Deployment

- **Description :** Project is to be finished and be deployed.
- **Deliverables and Milestones :** Deployed project
- **Resources needed :** Web server for hosting
- **Dependencies and constraints :** Hosting must be free of cost, and must have enough storage space for all project files.
- **Risks and Contingencies :** After hosting security risks

3.3.2 Assignment

- **Defining a Business Case:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Finalizing Problem Statement:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Project Approval:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Project Documentation:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Dataset preparation:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Finalization and implementation of techniques/tools for OCR for images and PDF data extraction:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Front-end development for modules developed :** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Database implementation and related operations:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Implementation of encryption module:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Implementation of analytics and interfacing with related front-end:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **UI development for rest of modules and refining already implemented UI:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Testing of individual modules:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Module integration into single system:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Testing of system:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania
- **Deployment:** Anushka Darade, Tirth Thaker, Utsav Parekh, Dhairya Umrania

3.3.3 Timeline

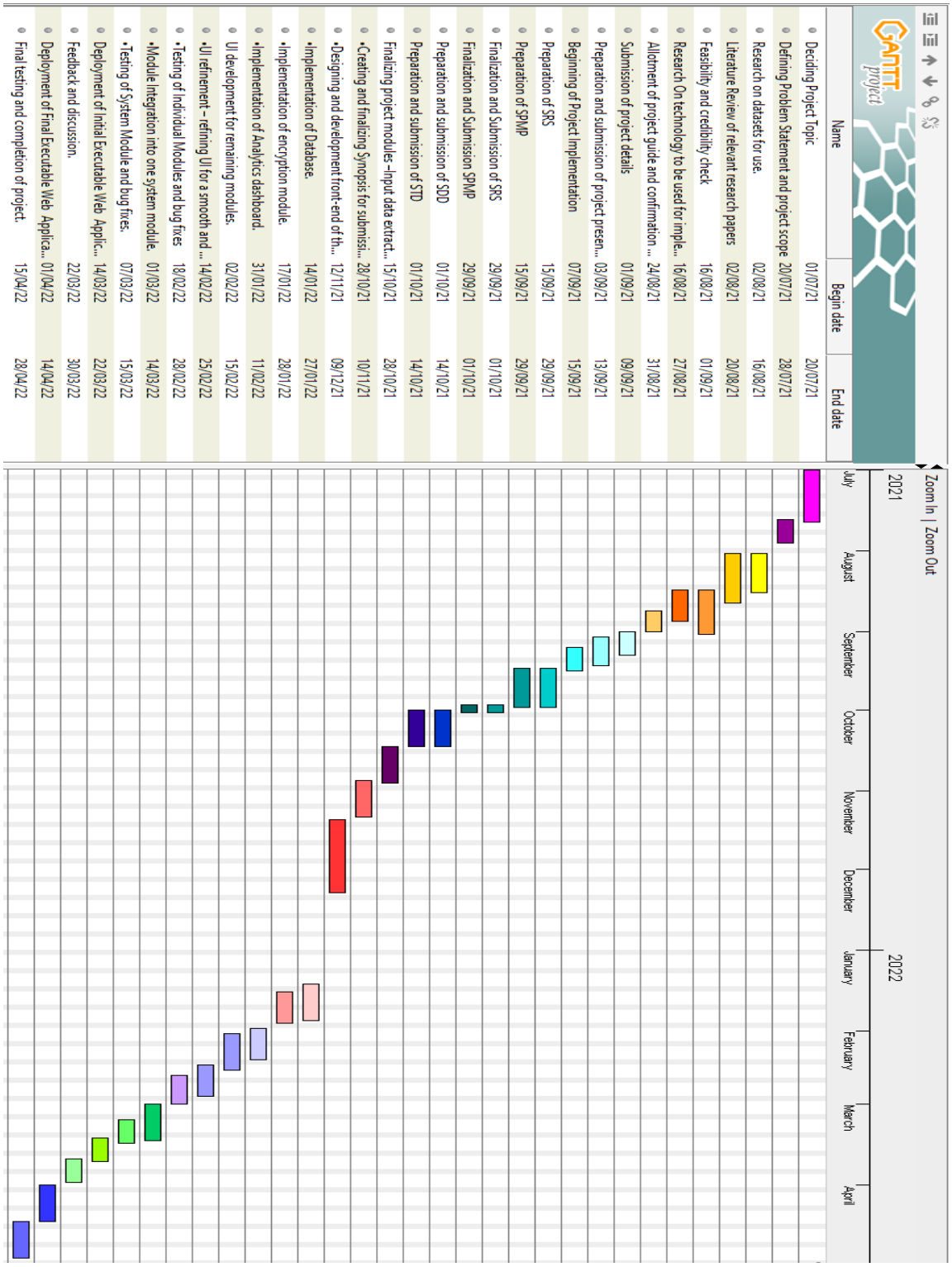


Figure 3.2: Project Timeline

Chapter 4

Software Requirements Specification

This chapter provides the Software Requirements Specifications document; and explains the requirements of the project - both software and hardware components. Content for product features, system attributes and database requirements wraps up the chapter.

4.1 Introduction

4.1.1 Product Overview

The aim is to create an application that will be used for report storage and analysis. The system will help patients understand their diagnostic reports with ease. It will ease the maintenance of reports for the user by updating it from newly entered reports. All the report information will be easily accessible for the user without hassle. This data will be stored securely in the database and safe from any third party access, as well as will maintain the integrity of the user data.

4.2 Specific Requirements

4.2.1 External Interface Requirements

4.2.1.1 User Interfaces

The complete application will run on a web browser (preferably Google Chrome). The user will have an option to sign-in for services in order to enter user information. The System should be user-friendly such that a user with no prior experience of using this software should be able to use and generate the report easily. The user will have selection of uploading new report or checking old report information and users' dashboard.
The interface while interacting with the user will comprise of a do's and don'ts list,hence all these requirements can be backtracked with the designer and the tester.

4.2.1.2 Hardware Interface

Personal Computer Laptop

- Processor: Pentium Dual Core 1.66 GHz and above.
- Active Internet Connection.
- System should possess at least 2 GB memory.
- CAT 5 Ethernet cable for wired network connection or wireless network connection is available.

4.2.1.3 Software Interfaces

1. Operating System
    - Microsoft Windows 8+
2. Web Browser
    - Google Chrome Browser 85+

4.2.1.4 Communication Protocols

The application will use client-server communication protocols over the internet. The client will try to establish a Web-Socket connection if possible, and will fall back on HTTP long polling if not. Web-Socket is a communication protocol which provides a full-duplex and low-latency channel between the server and the browser. It is expected to use TCP/IP protocol for secure transmission of data.

4.2.2 Software Product Features

1. User Account Creation: Users can create an account to use DoCure's service. The users will have to login before uploading their medical records. After logging in, they can view their previous medical records, and can delete their records anytime. The system would deny access if the user attempts to login with incorrect credentials as the database reflects invalid entry.
2. Electronic Heath Records (EHR): This is the base feature of our service. Stable Internet connectivity would ensure proper uploading and storing of health reports. The user can upload a PDF or an image of his medical report. The user can retrieve their report data as a part of the EHR system.
3. Medical Report Analysis: The user will be able to view data via a dashboard, wherein the users' medical data will be displayed in an organized format. Apart from this, the user will be able to receive insights on their report as per the analytics performed on their report via the system.

4.2.3 Software System Attributes

1. Performance/Response Time: It is expected that the OCR which is used for data extraction should be working with minimum error. Also, it is expected that the

interpretation of the medical reports should be delivered with accuracy. Hence, to ensure performance, high security and higher OCR accuracy is required.

2. Portability: Since it's a web based service, portability will be delivered as convenience. Hence, it can be accessed anywhere over the internet with the defined hardware, software and network constraints mentioned.

3. Maintainability/Modifiability: Maintenance will be crucial part of the service and hence, any complexity while doing so will be dealt within the designing process; hence, minimizing liabilities.

4. Security: The user credentials such as user information or passwords will be encrypted and stored, therefore, third-party interruption would be prohibited. Secure encryption algorithm will be put to practice to avoid any security lapses for user information as well as medical data.

5. Reliability: The application will be reliable for use, based on a secure database and being a web application. It will have minimal downtime with quick recovery in case of any issues.

6. Availability: The application will be available for usage at any given time since it is a web based application.

7. Usability: The system UI will be user friendly and will allow all users to easily access the different features of the system.

4.2.4 Database Requirements

The database services that will be integrated will meet all database requirements including information storage, accessibility, integrity throughout the system. The security of the database will be efficient to meet the requirements of the system.

Chapter 5

Software Design Description

This chapter includes the Software Design Description document. Content for the design overview, system architectural design, and detailed description of the components of the web application with UI design wire-frames is presented here.

5.1 Introduction

This chapter outlines the detailed structure of our project components and the precise details of implementation needed to fulfill the specifications set out in the Software Requirements Specification (SRS). It is assumed that the reader has read the SRS, since this document also defines the implementation details of the desired behaviour given the requirements within it. This document will build heavily on the Software Architecture and Design and so knowledge of the general system architecture is recommended prior to commencing this document.

5.1.1 Design Overview

The project has two dimensions : Client and Server. The client is a desktop application which will take user medical report images/PDFs and display the results received by the server. The server receives the request from the client,it processes the input and applies the algorithm to the input data and then returns the output to the client accordingly.

5.1.2 Requirements Traceabilty Matrix

	User	Authentication	Database
Login/Registration	Y	Y	Y
Uploading a Report	Y	Y	Y
Storing Data in Database		Y	Y
Displaying Analysis Dashboard		Y	Y

Table 5.1: Requirement Traceability Matrix

5.2 System Architectural Design

5.2.1 Chosen System Architecture

The architecture for DoCure is based on Layered architecture. Components within the layered architecture pattern are organized into horizontal layers, each layer performing a specific role within the application (e.g., presentation logic or business logic). Although the layered architecture pattern does not specify the number and types of layers that must exist in the pattern, most layered architectures consist of four standard layers: presentation, business, persistence, and database (Figure 1-1). In some cases, the business layer and persistence layer are combined into a single business layer, particularly when the persistence logic (e.g., SQL or HSQL) is embedded within the business layer components. Thus, smaller applications may have only three layers, whereas larger and more complex business applications may contain five or more layers.

Each layer of the layered architecture pattern has a specific role and responsibility within the application. For example, a presentation layer would be responsible for handling all user interface and browser communication logic, whereas a business layer would be responsible for executing specific business rules associated with the request. Each layer in the architecture forms an abstraction around the work that needs to be done to satisfy a particular business request. For example, the presentation layer doesn't need to know or worry about how to get customer data; it only needs to display that information on a screen in particular format. Similarly, the business layer doesn't need to be concerned about how to format customer data for display on a screen or even where the customer data is coming from; it only needs to get the data from the persistence layer, perform business logic against the data (e.g., calculate values or aggregate data), and pass that information up to the presentation layer.

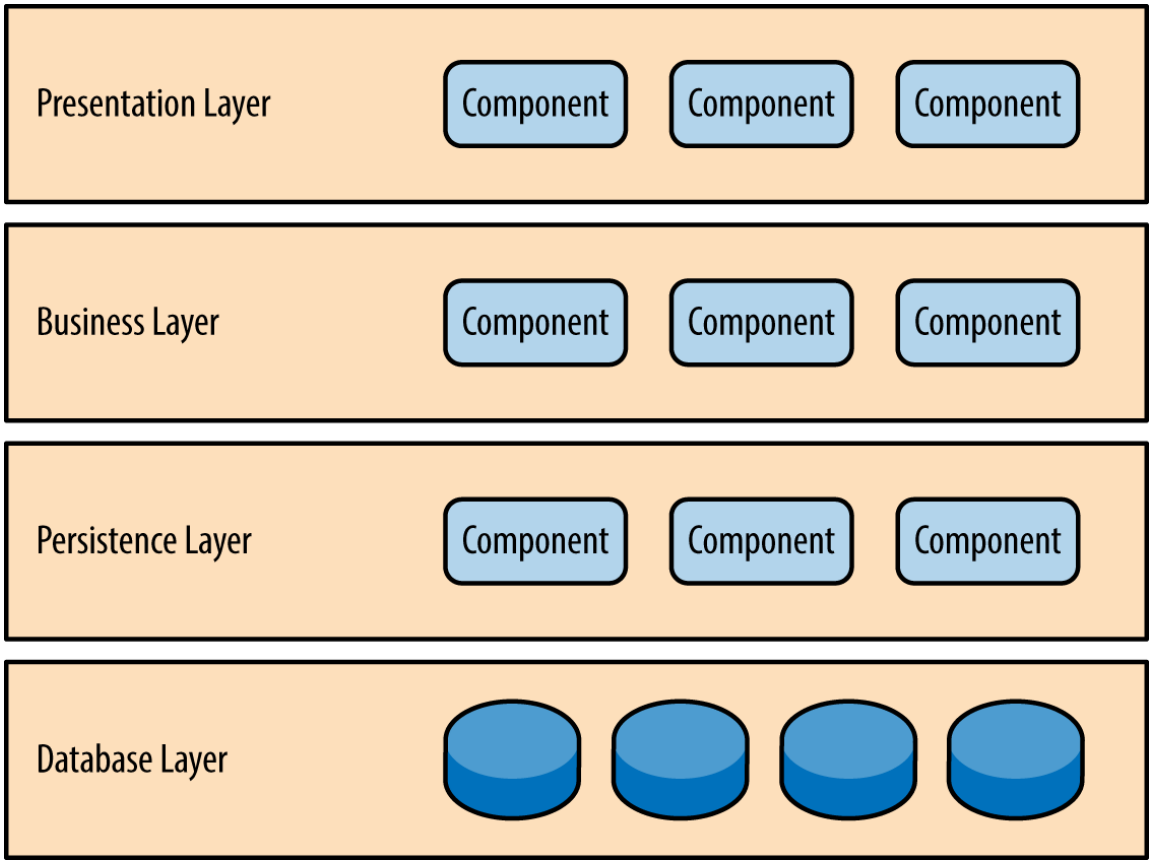


Figure 5.1: Layered Architecture

5.2.2 Discussion of Alternative Designs

A web application can also have 2-tier and 3-Tier architecture, where the layers are:
- Presentation Tier
- Business Tier
- Data Access Tier

In 2-tier architecture, the business layer does not exist, it only has the presentation layer and the data layer.

Our application suits the layered architecture approach more, hence we will be using layered architecture.

5.2.3 System Interface Description

1. User Interface DoCure shall be a web application system, hence the user must interact with it using a laptop/desktop.User has to first login to access the functionality of the system. After successfully logging the user will be directed to the home page where the user will be given the option to upload a medical report or view their analysis dashboard. Note, the analysis dashboard won't be available for new users who haven't uploaded any report to the database.

   If the user clicks on Upload report, they will be redirected to the Upload screen,

where they can drag and drop, or browse from their file directory to choose the image file or PDF file of their report. Once they have chosen the file, the report will be uploaded to the server, and the user will be redirected to the home page.

If user clicks generate report dashboard, they will be redirected to the analysis dashboard page, where their report data will be visualized and inferences from their report will be generated for them to see.

2. Hardware Interface The application is developed for Desktops and laptops. The user should possess a desktop/laptop with a web browser which supports Django.

3. Software Interface Web Browser We have chosen Google Chrome Browser for their best support and user- friendliness.

4. Communication Interface As the software supports online database connectivity, any changes the user wants to commit need to be done online. A good Internet connection is required.

5.3 Detailed Description of Components

5.3.1 Component 1 - Uploading Report

Responsibilities	Give an interface to user to upload documents
Constraints	User must login first and enter image or PDF, report must be of given format and is verified.
Composition	Front-end: HTML5, CSS3, bootstrap; Back-end: Django , MYSQL,OCR;
Interactions	Users will upload medical report image/PDF to the server
Resources	Developer

Table 5.2: Component 1 - Uploading Report

5.3.2 Component 2 - Store report data in database

Responsibilities	The extracted report data will be encrypted and stored in database.
Constraints	User must have logged in and a report must have been uploaded.
Composition	Frontend: HTML5, CSS3, bootstrap; Backend: Django, MYSQL;
Interactions	-
Resources	Developer

Table 5.3: Component 2 - Store report data in database

5.3.3 Component 3 - Analysis Dashboard

Responsibilities	Give an analysis of reports to user and metrics user care about
Constraints	Users Information must be accurate and up to date
Composition	Frontend: HTML5, CSS3, bootstrap; Backend: MYSQL, Django;
Interactions	Real-time insight of data and help users to visualize, filter, and dig deeper into their data
Resources	Developer

Table 5.4: Component 3 - Analysis Dashboard

5.4 User Interface Design

5.4.1 Description of User Interface

5.4.1.1 Screen Images

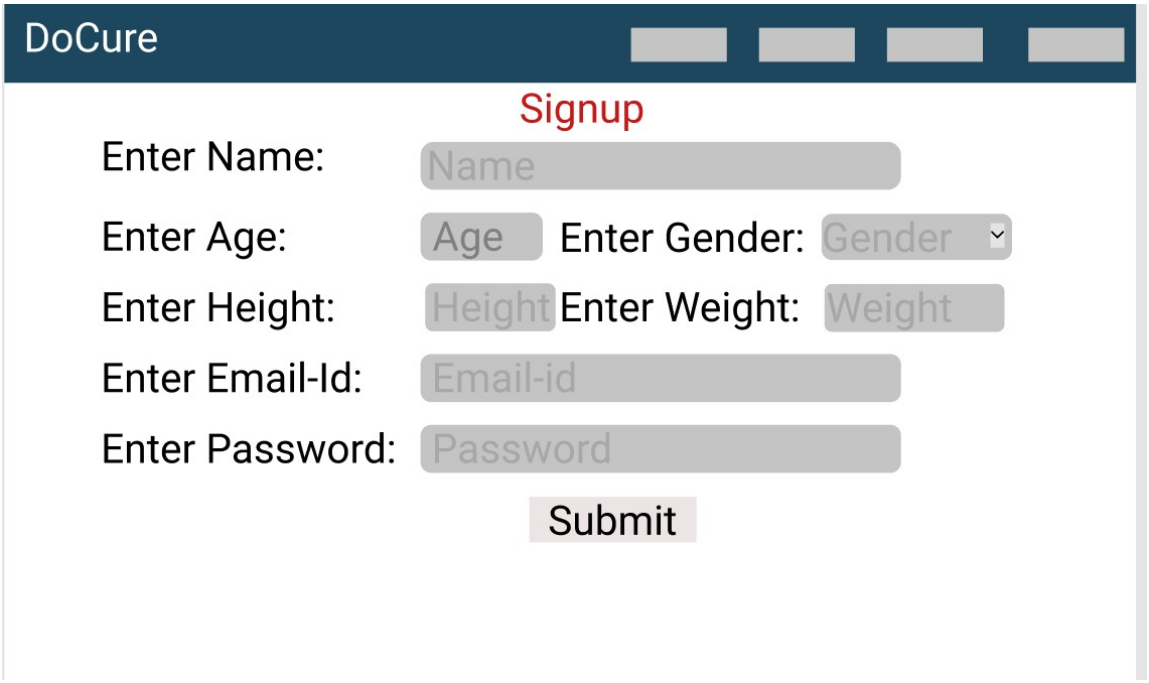


Figure 5.2: Sign Up page

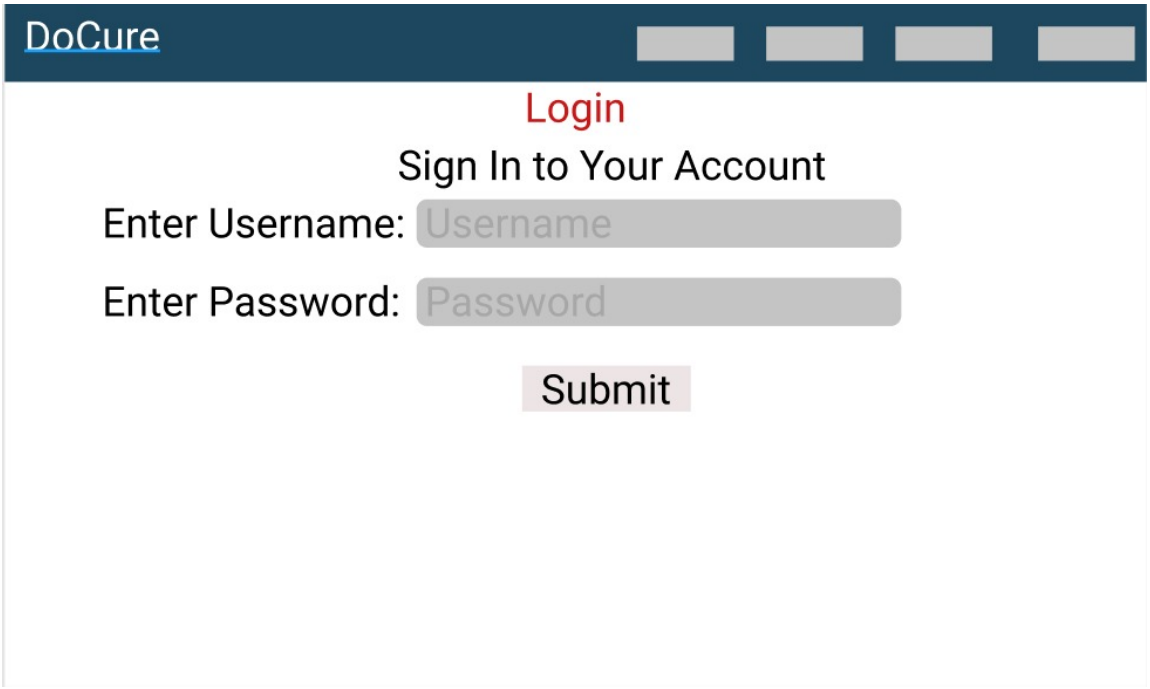


Figure 5.3: Login Page

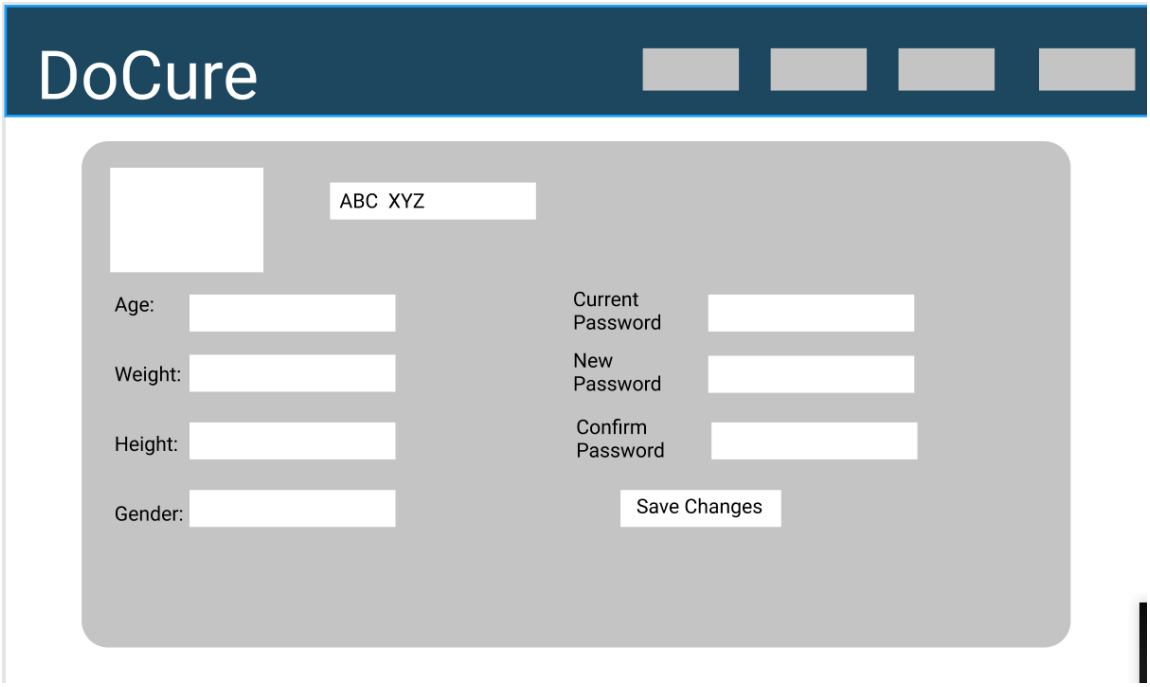


Figure 5.4: Edit User Profile

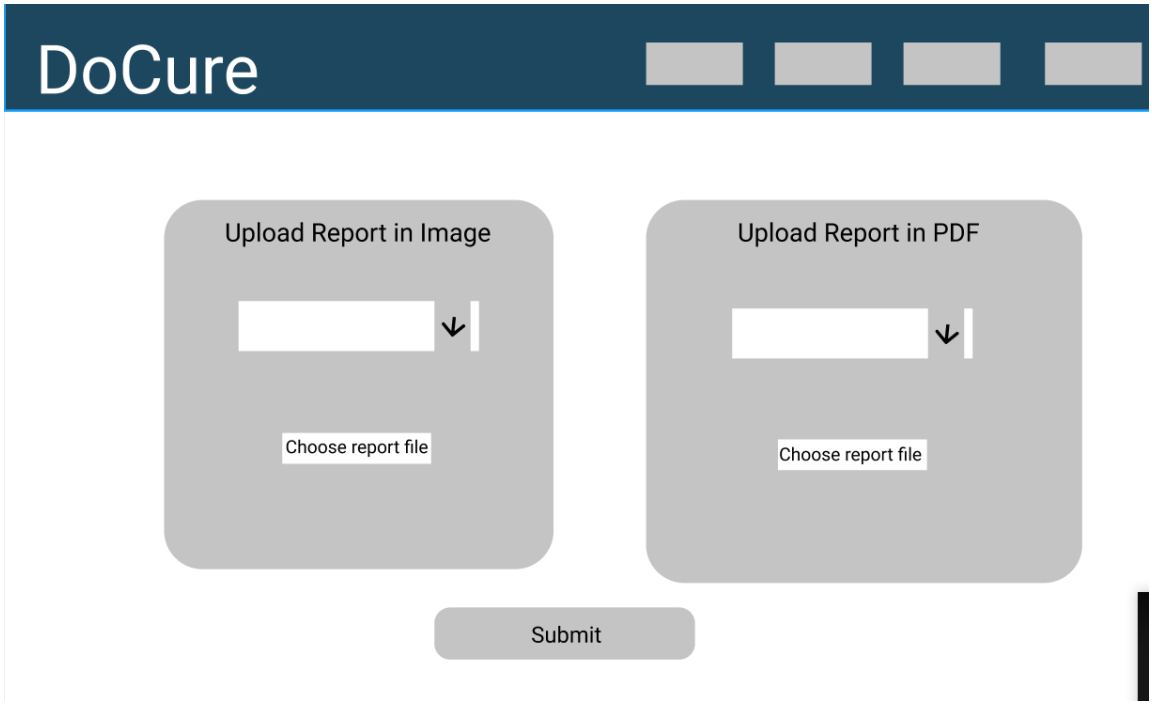


Figure 5.5: Report Upload

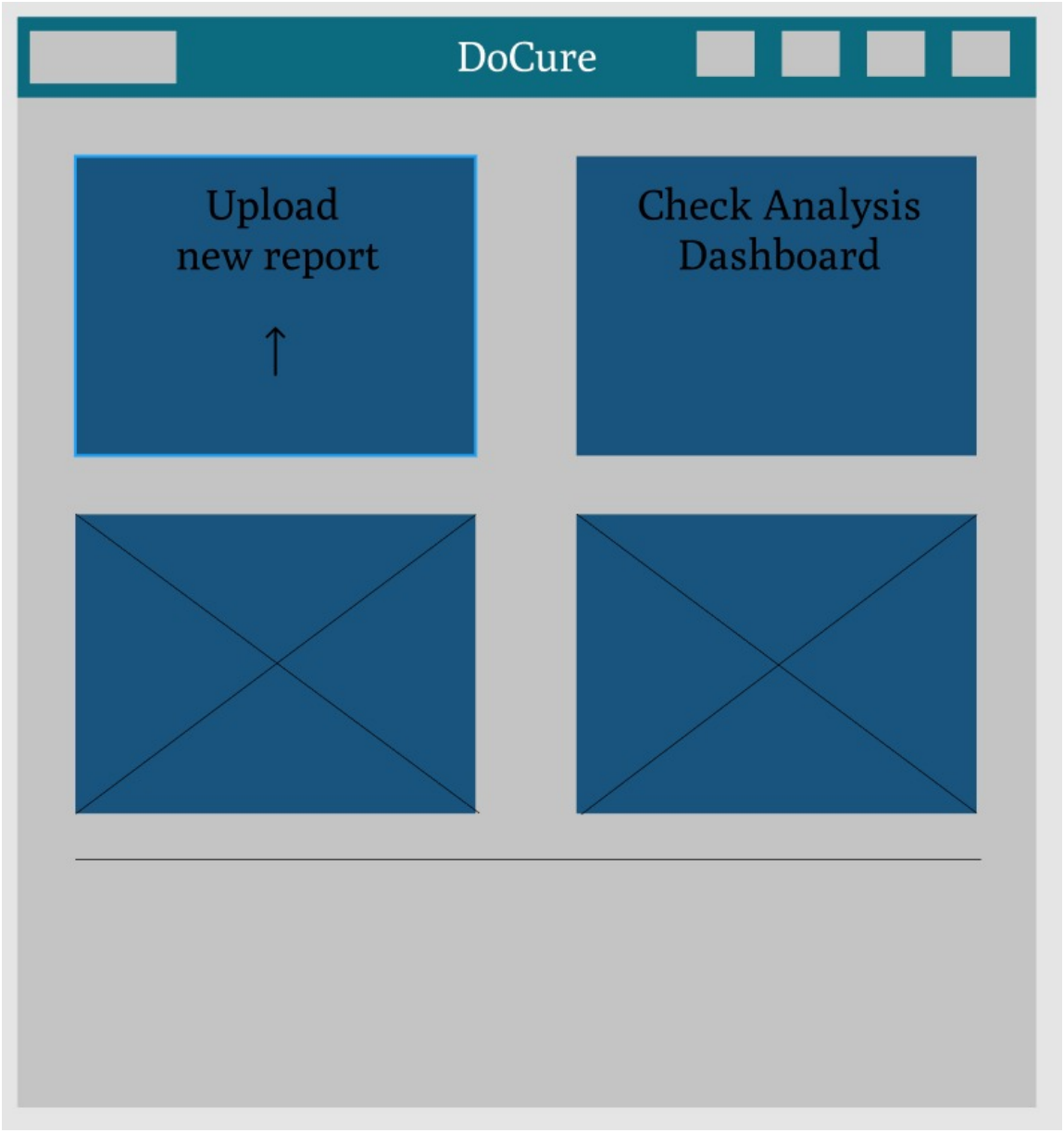


Figure 5.6: Home Page

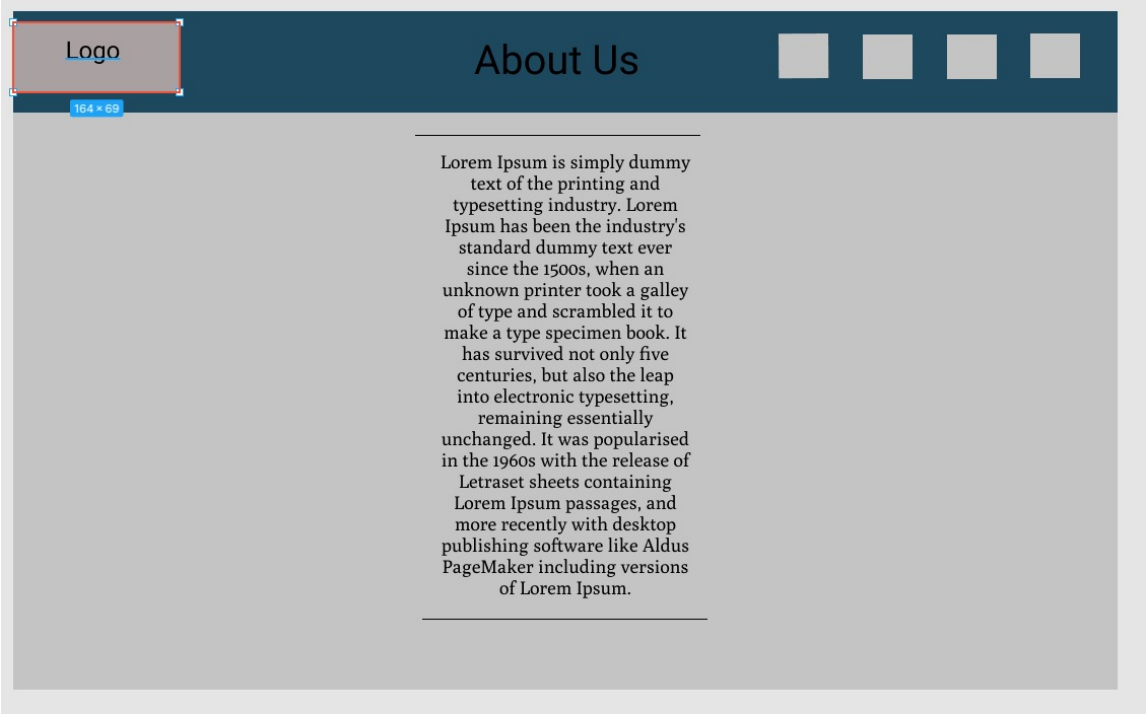


Figure 5.7: About Us

Figure 5.8: Dashboard

5.4.1.2 Objects And Actions

Use Case ID:		1	
Use Case Name		Login	
Created By:	Dhairya	Last Updated By:	Dhairya
Created On:	24/9/21	Last Updated On	25/9/21
Primary Actor :		User	
Secondary Actor :		-	
Description :		Used to log in into the system if the user is already registered.	
Trigger :		Valid login credentials are provided and sign in button is pressed	
Preconditions :		Registration is done.	
Post Conditions :		Allow user to access system.	
Normal Flow :		System verifies the sign in credentials and allows to find the flight details	
Alternative Flow :		Redirect to enter the valid credentials	
Exceptions :		-	
Includes :		Register	
Priority :		High	
Frequency of use :		Medium	
Special Requirements :		-	
Open issues :		User might have not completed registration process.	
Assumptions :		User has internet connectivity and will enter the credentials correctly	
Notes issues :		-	

Table 5.5: Use Case 1

Use Case ID:		2	
Use Case Name		Upload Report	
Created By:	Anushka	Last Updated By:	Anushka
Created On:	25/9/21	Last Updated On	31/9/21
Primary Actor :		Registered User	
Secondary Actor :		System Administrator	
Description :		To upload user reports to server.	
Trigger :		"Upload report" is selected.	
Preconditions :		User is logged-in to the system, and has a scanned report available	
Normal Flow :		1. User will click on "Upload Image" button, 2. Users will upload a medical report image/pdf to the server, 3. After successfully validating the uploaded image it will redirect to the report generation page.	
Alternative Flow :		Report not uploaded successfully, wrong report uploaded	
Exceptions :		File uploaded is not an image or pdf.	
Priority :		High	
Frequency of use :		High	
Open issues :		-.	
Assumptions :		User has scanned report and internet connection.	
Notes issues :		-	

Table 5.6: Use Case 2

Use Case ID:		3	
Use Case Name		Store report data into database.	
Created By:	Tirth	Last Updated By:	Tirth
Created On:	31/9/21	Last Updated On	1/10/21
Primary Actor :		System	
Secondary Actor :		System Administrator	
Description :		To encrypt and store extracted data from reports into database.	
Trigger :		OCR extraction is completed.	
Preconditions :		Report must be uploaded	
Post Conditions:		Data extracted should be verified once.	
Normal Flow :		1. Extracted data is encrypted, 2. Data is stored in database.	
Alternative Flow :		Server error or database error, in which case error message to be shown to user.	
Exceptions :		OCR extracts data incorrectly.	
Priority :		High	
Frequency of use :		High	
Open issues :		-	
Assumptions :		OCR has extracted data correctly, and database back-end is correctly configured.	
Notes issues :		-	

Table 5.7: Use Case 3

Use Case ID:		4	
Use Case Name		Analysis Dashboard	
Created By:	Utsav	Last Updated By:	Utsav
Created On:	1/10/21	Last Updated On	3/10/21
Primary Actor :		User	
Secondary Actor :		System	
Description :		To generate an Analysis dashboard with visualization for the user report data.	
Trigger :		"Generate analysis dashboard" is selected.	
Preconditions :		User data exists in database.	
Post Conditions:		Dashboard is generated correctly.	
Normal Flow :		1. Retrieve data from database, 2. Decrypt data, 3. Generate dashboard.	
Alternative Flow :		If data cannot be retrieved, show error.	
Exceptions :		User data does not exist in database.	
Priority :		High	
Frequency of use :		High	
Open issues :		-	
Assumptions :		-	
Notes issues :		-	

Table 5.8: Use Case 4

5.5 Data Flow Specifications

5.5.1 Level 0 DFD

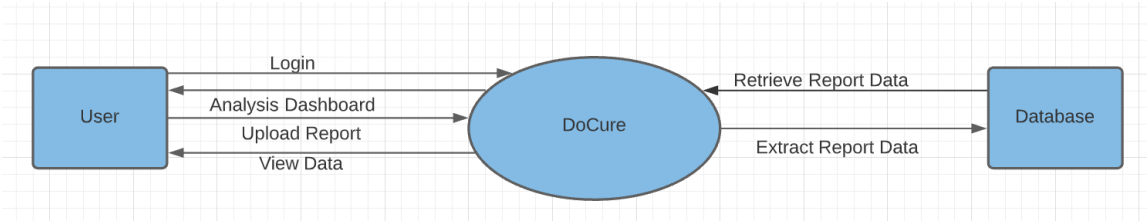


Figure 5.9: Level 0 DFD

It displays the project context diagram. It is intended to be an abstract view, illustrating the system as a single process with its connection to external entities.This represents the entire system as a single bubble, indicated by inncoming/outgoing arrows with input and output data. "User" , "DoCure" and "Databse" are the entities, and our system.User uses DoCure to Login, upload reports and DoCure sends back the Analysis Dashboard to the user. DoCure interacts with the Database to store and retrieve data.

5.5.2 Level 1 DFD

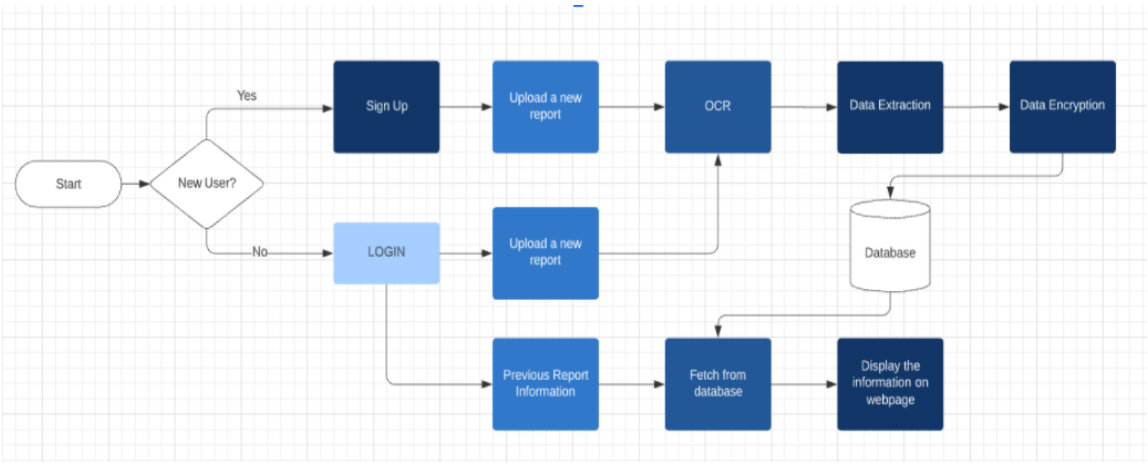


Figure 5.10: Level 1 DFD

The context diagram is decomposed into several bubbles/processes in this figure. At this stage, we highlight the system's key functions and break down the level-0 DFD's high-level process into subprocesses. The User will either log in or sign up, if they don't have an account. After authorization is done, they will be redirected to the homepage, where they can choose to upload a new report or view the analytics dashboard of their uploaded data.

Chapter 6

Implementation

This chapter discusses the implementation of the project in detail. It lists down the technology stack used with an algorithmic implementation. A detailed implementation procedure for each of the modules is presented through this chapter.

6.1 Technologies Used

The DoCure application is a web application. It is developed using the Django framework and hosted over the Heroku PaaS. The text recognition module of the project is developed by using the Nanonets API for images and the PDFPlumber library in Python. The Nanonets API works over an HTTP request which returns the recognized text. The retrieved text is put under text extraction which is performed using Named Entity Recognition. . This produces the labelled data. The data is encrypted using the Advanced Encryption Standard (AES) algorithm and stored onto the SQLite database integrated with our project. The project is built on HTML, CSS, JavaScript based front end. Lastly, testing is automated with the use of Lighthouse and manually tested for any errors.

Web Application	Django Web-Development Framework
Web Hosting	Heroku PaaS
Text Recognition	Nanonets, PDFPlumber
Text Extraction	Spacy
Front-End	HTML, CSS, JavaScript
Testing	Lighthouse

Table 6.1: Technologies Used

6.2 Algorithm

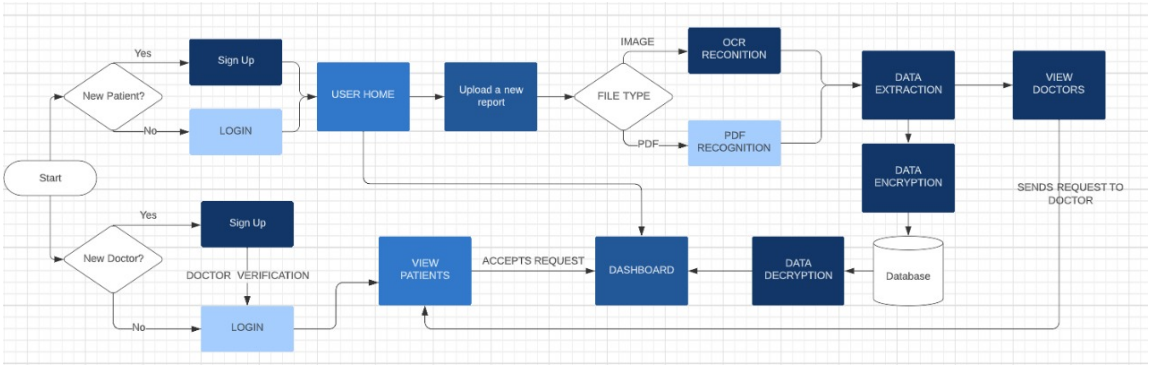


Figure 6.1: Block Diagram

1. Patients' Module:
   (a) The patient can login or if new, can sign up on the system to use it.
   (b) The patient is lead to the home page wherein the user can either upload their new reports or use the dashboard to view their previous reports.
   (c) If user chooses to upload new report, go to step (d), else go to (j)
   (d) User uploads report file, if file is image, go to step (e), else go to step (f).
   (e) OCR is applied on image using Nanonets API. The text is given for text extraction.
   (f) PDFPlumber recognises the PDF text and it is provided for text extraction.
   (g) Data is extracted from the recognized text and is produced in a labelled format.
   (h) The data is encrypted and stored in the database.
   (i) The user can view doctors on the portal and request for consultation.
   (j) The user can view his reports on the dashboard wherein all details are mentioned in a graphical and easy to understand format. The patient can also view the doctor's comments on the reports.
2. Doctors' Module:
   (a) The doctor can login or if new, can sign up on the system to use it.
   (b) The doctor can view the patients and accept or reject the requests.
   (c) The doctor if accepts the request, can view the reports and add his comments on the same on the dashboard which is in turn visible to the patient.

6.3 Implementation

The web application is built using the Django web framework that succors rapid development of the application. The framework helps in integrating the several text recognition, extraction and interpretation modules together as well as facilitate a simple and intuitive UI. The UI comprises of two main modules:

Patients' side: The patients' side makes use of the UI for logging in, uploading reports and checking analysis for reports. The module makes use of text recognition in form of Nanonets, or PDFPlumber. The reports are uploaded, processed, and stored in an encrypted form on the SQLite database. The user can then access the reports by retrieving them from the database, decrypted and can view remedies and related analytics regarding the same. The user can request doctor consultation based on their reports and view it in form of the doctors' comments.

Doctors' side: The doctor can register and login, on basis of which he can be requested for consultation by the patients. The doctor can accept or reject patients. On accepting requests, the doctor can view the reports and add comments based on the same.

Text recognition involves identification of text from a document. Our project includes the recognition of images and PDFs. Since reports can be in paper or digital format, provision for images and PDFs is necessary to provide reports to the system. Our system identifies the file type and runs the recognition on the document accordingly. Optical Character Recognition is the recognition of text from images by scanning the image and retrieving all the text data from the same. Our system makes use of Nanonets for performing OCR on images. Nanonets is an automated tool for data recognition and labeling on the web. It employs the Nanonets API which passes the image and gets the OCR response over HTTPS protocol in the form of response. Our application stores the uploaded image which is passed onto the OCR module. The OCR module detects and returns all the necessary strings over which the processing and extraction procedures take place. The system also has provision for scanning PDF files and retrieving the data from them for processing. This is performed using inbuilt libraries and provides for password-locked files too. To get the parameters and their values from the raw text generated by the OCR or the PDF reader, named entity recognition (NER) along with regular expressions (Regex) is used. For named entity recognition, a machine learning approach is implemented, where the training data consists of the text, the named entities in the text and their spans. The dataset used for this is the text of 49 CBC reports generated from the OCR, along with their entities and spans. The data is shuffled and split into 70:30 for training and testing, and the training data is further split into 50:50 for training and validation. The NER pipeline consists of a tokenizer, tagger, parser and NER. Tokenizer, tagger and parser components are used to pre-process the text before feeding it to the NER component. The NER is done using word embeddings using subword features and bloom embeddings which is a deep convolutional network with residual connections.
HomeBefore Page is the initial Page of our Web App which contains links to the Login Page, and the Register Page for patients and Doctors. After clicking on the Register Page We ask Patients for general information to enter. Then after proper verification and validation of the Form, it will go to the Login Page. The Login Page can only be logged in to the Registered Patient who has registered on our Website, after verification of credentials, it will log in to our Website home page. Home Page contains all links to Upload Reports Page, My Reports page, to view all the reports uploaded by patients, about us page, to get information about the website, Feedback page to give feedback about our WebApp. The upload page contains the two input boxes: the name of the file uploaded by the Patient and a Password input. The Patient can enter the password if the PDF file requires a Password. After the file is uploaded, the report is sent to the OCR

and text extraction module, from which the data is extracted. After extracting the report data, it will direct to the ConfirmForm page. The ConfirmForm Page takes the Extracted Data from PDF/IMAGE in the form where the Patient can see the extracted values of his/her Reports. It helps Patients to enter values of reports which were misinterpreted by our Text Mining Techniques or OCR Techniques. It also helps the Patient to change the name of the report which Patient wants to set. After Clicking on Submit, the data will be stored in the database in encrypted form and will be directed to the "My Reports" page. My reports contain all Reports Uploaded by Patients in a tabular Format. The Table contains the upload time of the Report, the Name of the Report, the Type of Report, a link of "View Report " to see the Dashboard analysis of the report, and "Delete Report " from which a Patient Can delete Reports. After clicking on View Reports, Dashboard Page pops up and it contains the extracted data from the Files uploaded by the Patient. The dashboard shows the extracted values with color-coding. Red Color for any values that are above the normal range, Green if values are between the normal range, and Yellow if values are at the boundaries of the normal range. The Dashboard Page also has a table with four columns which are the Metrics, the Normal Range of Metrics, the Extracted Data of Each Metric (Report Value) and Remarks, where the Patient can see information about home Remedies and other notifications. The patient can see the Doctors' Comments on the report that they uploaded. To book aforementioned doctors, the Patient can go to the View Doctor page in which all the doctors available for the patient are visible. Patients can see all the Doctors with their names, Specialization, and Gender and can book the Doctor according to their requirements. The patient Profile Page gives an overview of the Patient information which the Patient has registered during registration. On this Page the If Patient Wants to change any information then they can click on the Edit Profile button and can go to "EditProfile" Page. EditProfile Page helps the Patient to change their personal information and save it to the database.
A new doctor will first register on the Doctor Registration page. After registration they will be confirmed by the admin. There will be verification of doctor details, the doctor will be able to login with the given username and password. After logging in the doctor can see the list of patients who have sent him a request. The doctor can further accept or reject the request. Viewing Patient Reports – the doctor can see all patient reports on this page. At last the doctor can view the dashboard for each report and comment if those comments needed, will be reflected on the patient dashboard.
User Table contains general information about patients with UID as primary key of the table and it is referenced as foreign key to Reports Table and View Doctor Table. The Reports table has the primary key as Report ID and information about report type and it is referenced as Foreign key to CBC Report and Urine Report table. CBC, Urine Reports are two types of report which our Web App is able to extract using Text mining and OCR Techniques. Doctor table where Doctor ID is Primary key and general information about Doctor is stored in the table. After Verification we have Confirm Doctor where Doctor ID is Foreign key referred from Doctor table. In the View Doctor We have User ID and Doctor ID as foreign keys referring to the user and Confirm Doctor table.

To add usage for more reports, we have prepared a pipeline which can be followed. The first step is to collect reports of the type you desire to add, there is no fixed number for the number of reports, but with more reports, the text extraction model will be trained

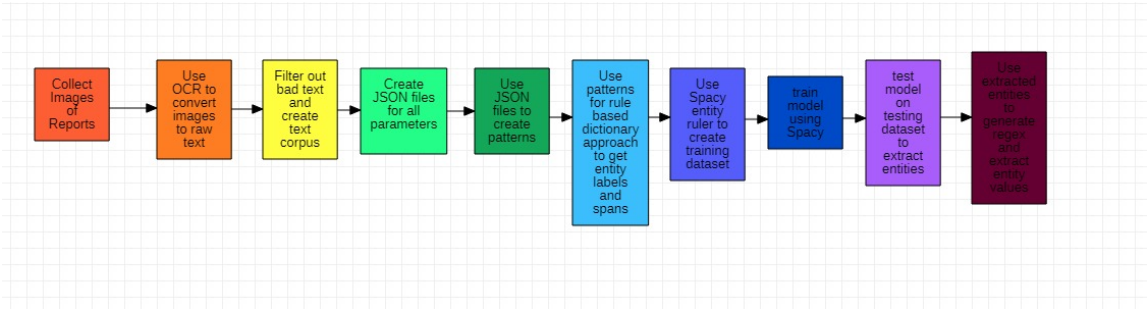


Figure 6.2: Reports Pipeline

better. After collecting the reports, the raw text of the reports need to be extracted. For that we have provided a jupyter notebook, which contains the Nanonets OCR, which is extracting the raw text from the report images and storing the text in .txt files that are named according to the number in which they appear in the image folder, and then they are stored in a directory of your choice. If you want to use some other OCR, you can use that too. After the raw text has been extracted from the images, the txt files have to be checked if the data has been correctly identified because there could be cases where the text has not been detected correctly due various reasons such as bad image quality, bad lighting, OCR mistakes, incorrect interpretation, etc. The txt files with bad text must be filtered out and only those which are correctly identified must be kept. Next step is to train the text mining model. For this we have used Spacy, this can be found in the cbc_ml.ipynb jupyter file. But before training the model, a rule based dictionary approach must be done to get the desired text data entities and their spans in corresponding text, because spacy needs the training data in the format:
{text, entities: (label1, span), (label2, span)
To do this first the parameters of the report that are to be extracted must be listed down and for each separate entity a JSON file must be created with all the names that the entity could take according to the user. Then open the cbc_ml.ipynb jupyter file and read the JSON files to get the entities and their patterns, the stems of the words in the JSON files are modified a bit to cover more patterns for the dictionary approach. After all the patterns are mapped out, they have to be put in the following format:

$$label1 : entity1, pattern : pattern1, label1 : entity2, pattern : pattern2, \ldots$$

This is done using the spacy entityruler which can be found in cbc.ml_ipynb jupyter file. After the entity ruler has been used to map out the patterns, the dictionary approach can be used to get the entities and their spans from the text corpus collected from the report images. Once the entities and their spans are collected they are to be put in the format that we defined previously: {text, entities: (label1, span), (label2, span) This can be done using the function provided in the jupyter file. Once all the txt files from the text corpus have been parsed, the data will be created in the format mentioned above. This data can be split into training, validation and testing datasets. The training and validation datasets need to be converted into .spacy format, which can be done using the save_data function in the jupyter file.
After the training and validation datasets have been saved in the .spacy format, the

training of the model has to be done on the command line. The commands and steps for training can be referenced from the following site:
https://spacy.io/usage/training
Here, a Spacy NER base.cfg file must be downloaded, and the paths of the training and validation datasets must be provided. The base.cfg file must be converted to a config.cfg file using the following command on command line:
python -m spacy init fill-config base_config.cfg config.cfg
After this, the training must be done using the following command:
python -m spacy train config.cfg –output ./output
Once the model is trained, the training scores will be visible in the command prompt, and the trained model will be saved to the folder named "output". That model can be called and tested on the testing dataset. The function for generating the confusion matrix and calculating the accuracy on the testing data is given in the jupyter file. After looking at the metrics, if the model is well trained, the next step is to use those extracted entities to create a regex expressions to extract their corresponding values. The code for generating the regex is given in the jupyter file. Once the regex for the specific parameter is created the value of that parameter from the report can be extracted and stored in a variable. This pipeline can be repeated for as many report types as required provided the data is sufficiently available.

6.4 User Interfaces:

6.4.1 Patients Module

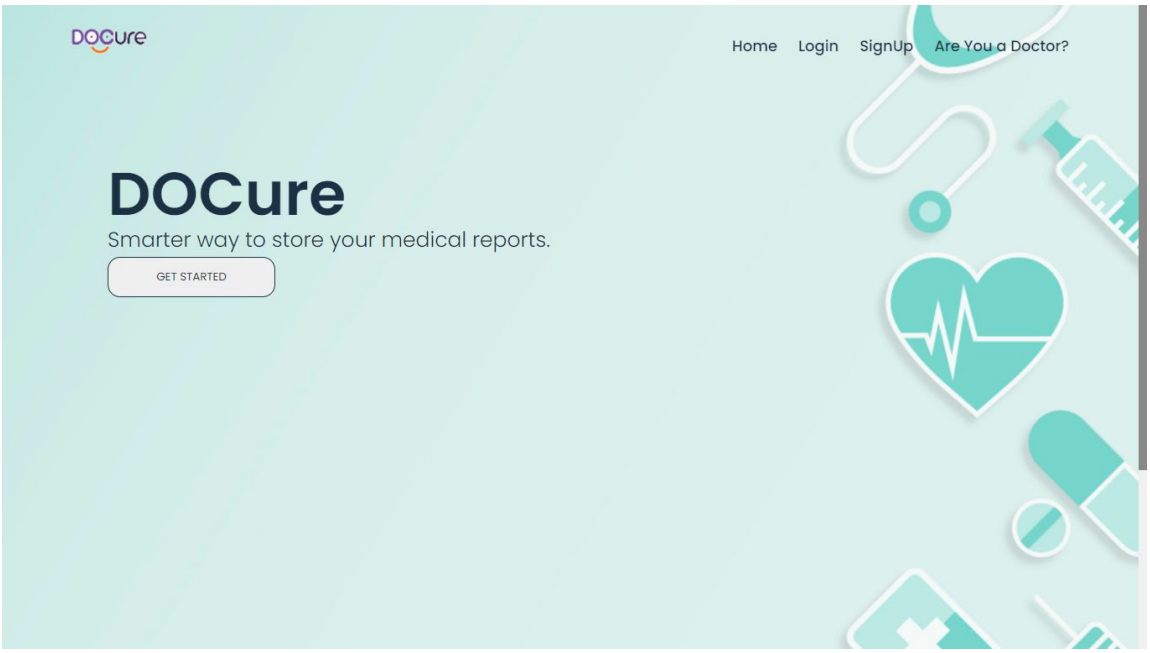


Figure 6.3: Welcome Page

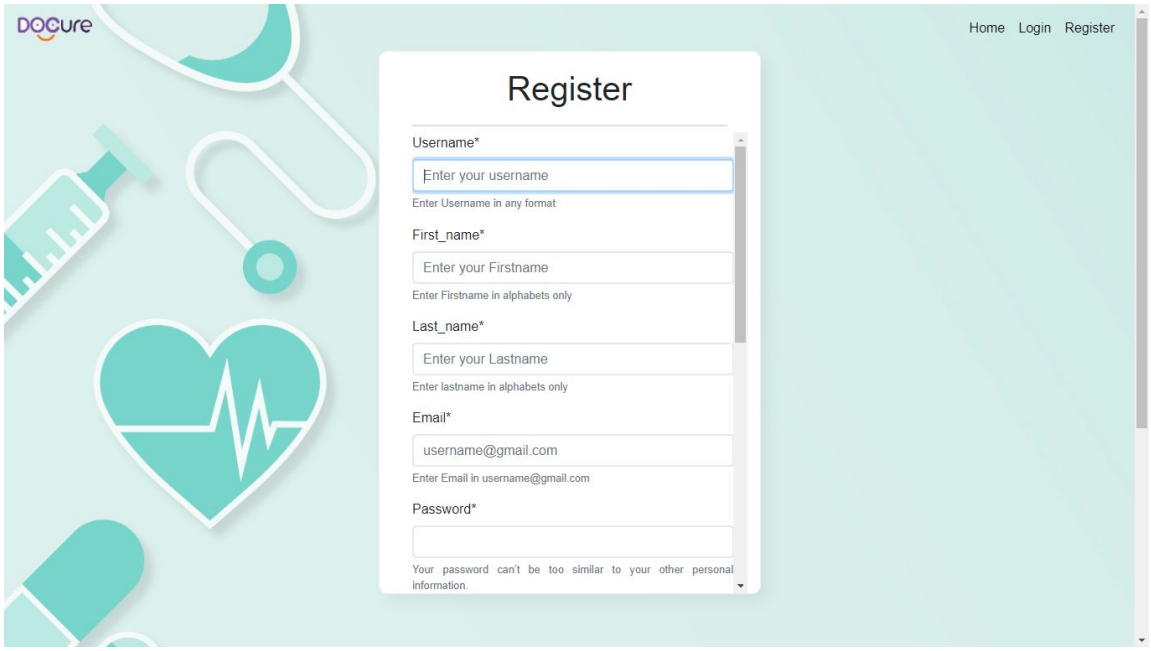


Figure 6.4: Sign Up Form

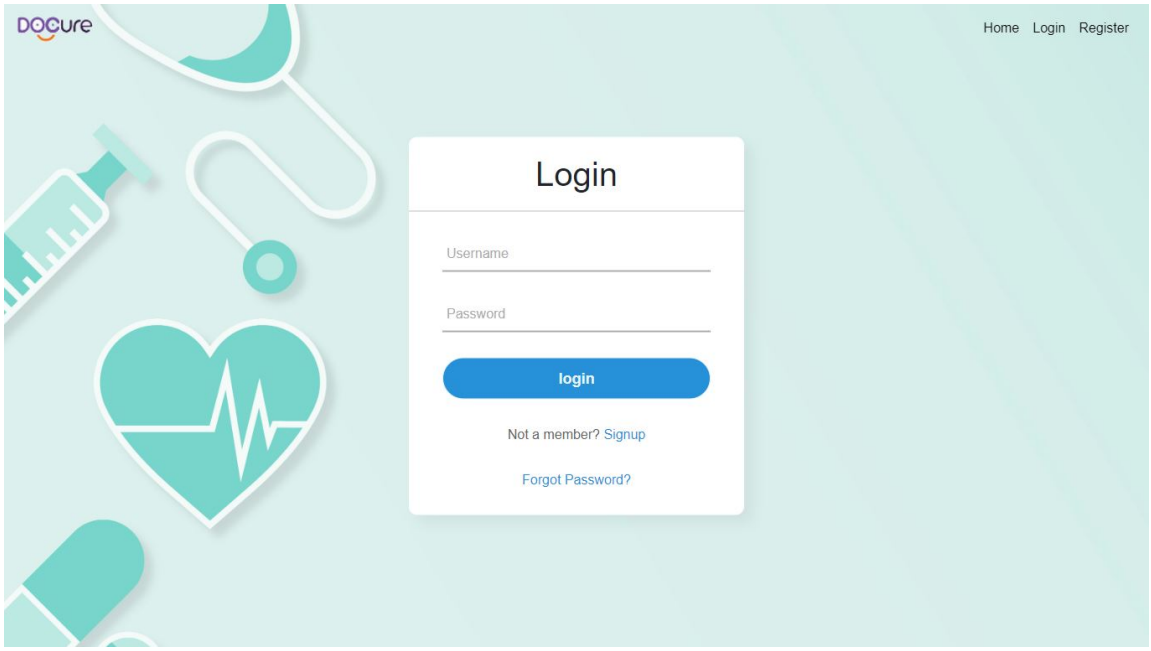


Figure 6.5: Login Form

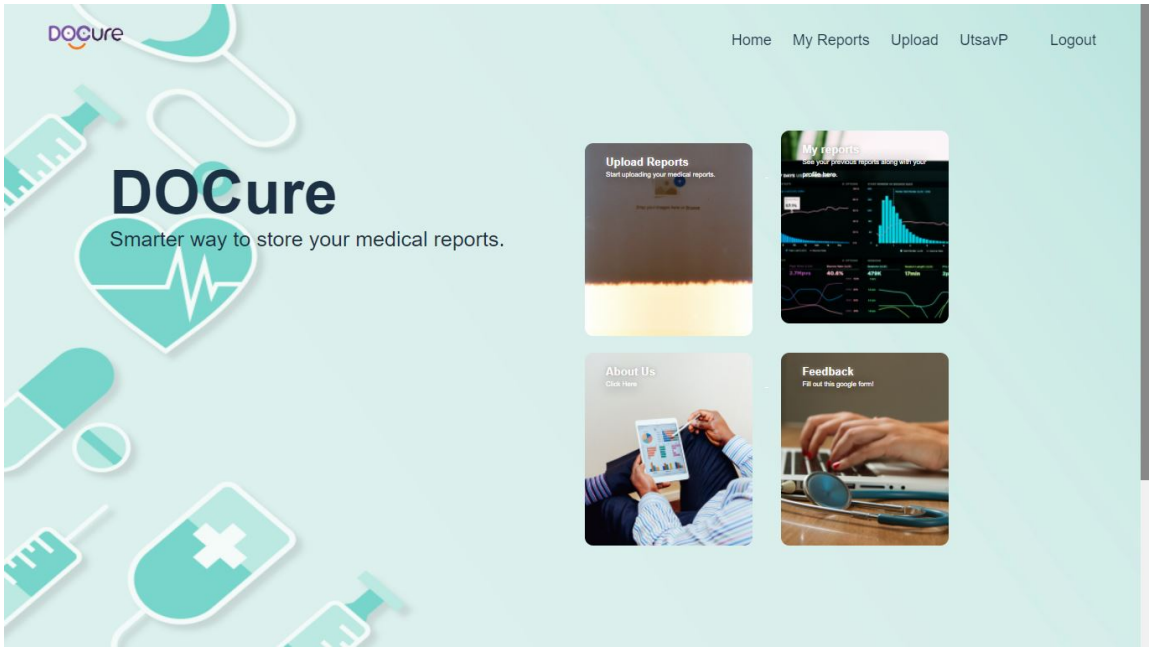


Figure 6.6: Home Page

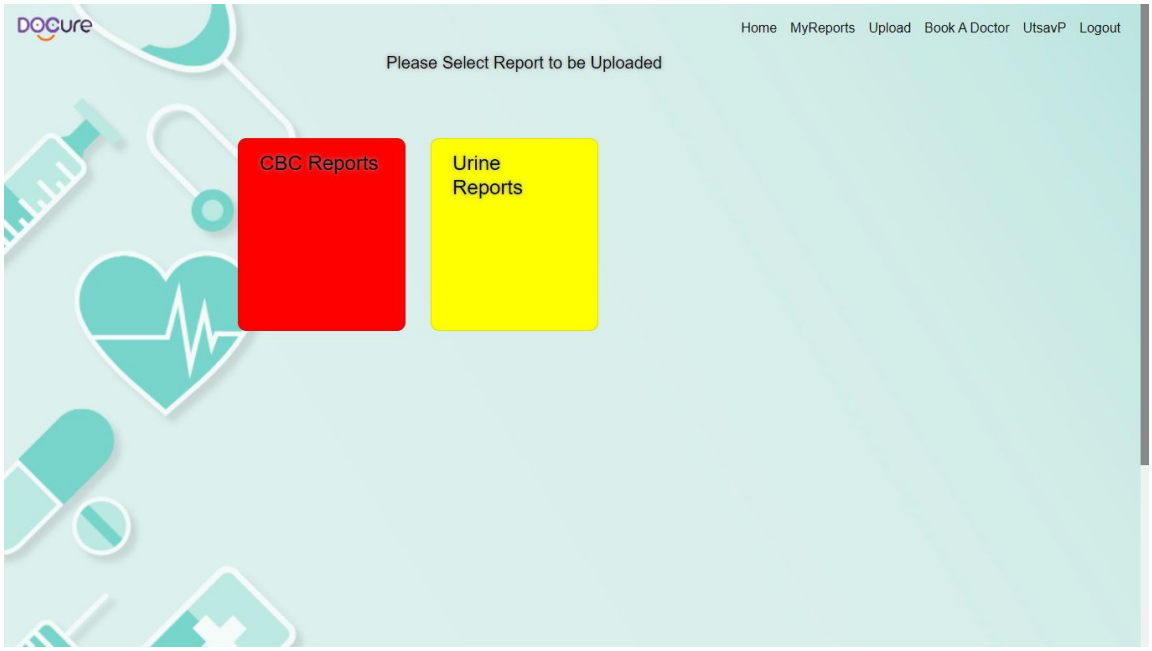


Figure 6.7: Report Type Selection

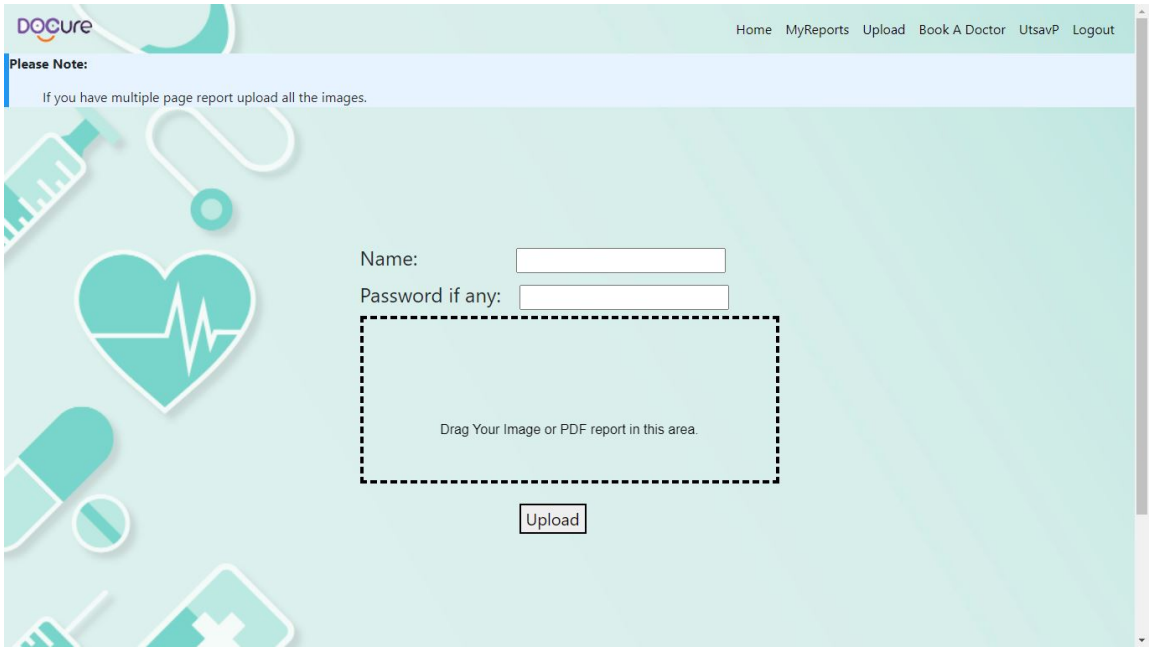


Figure 6.8: Report Upload Page

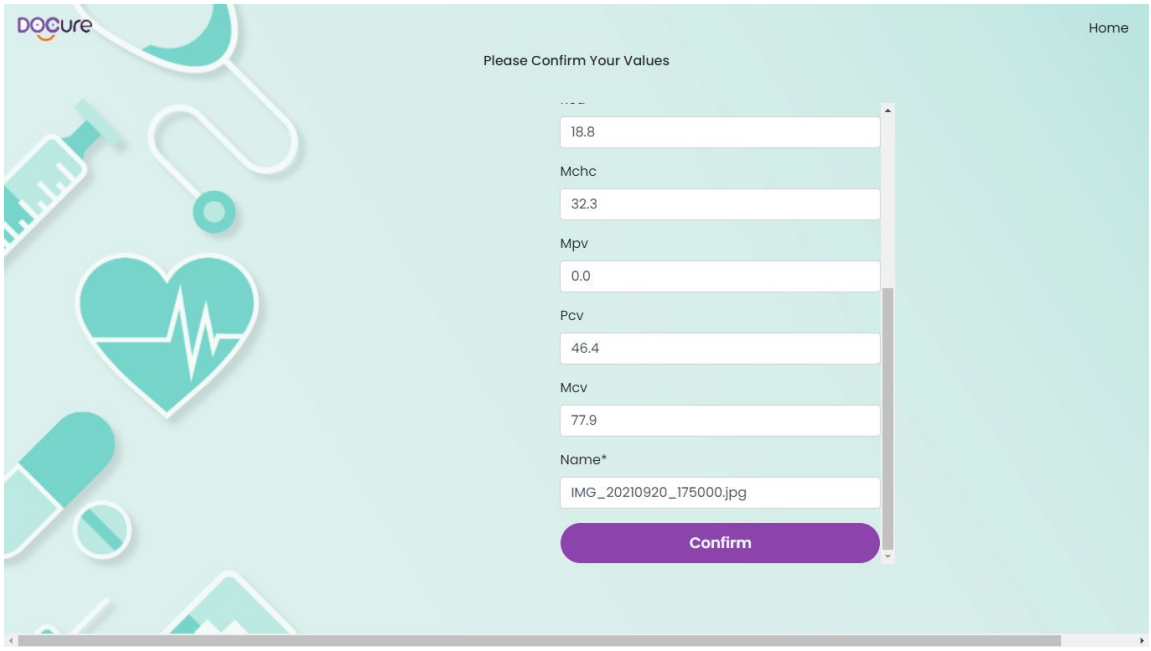


Figure 6.9: Confirm Value page

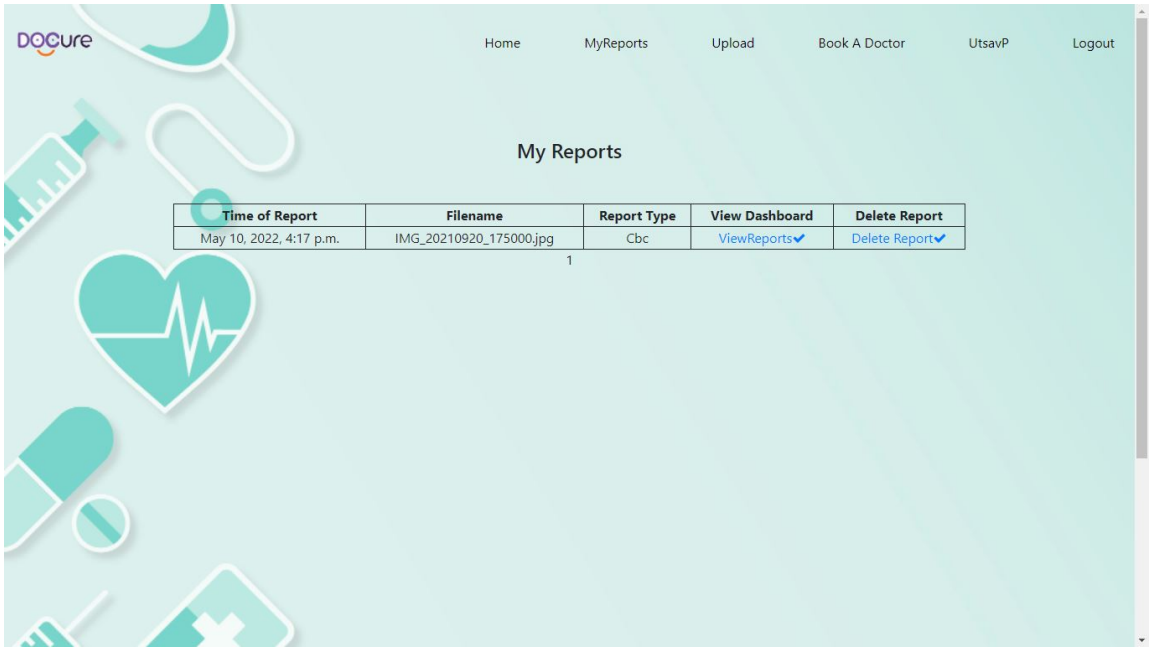


Figure 6.10: View Reports

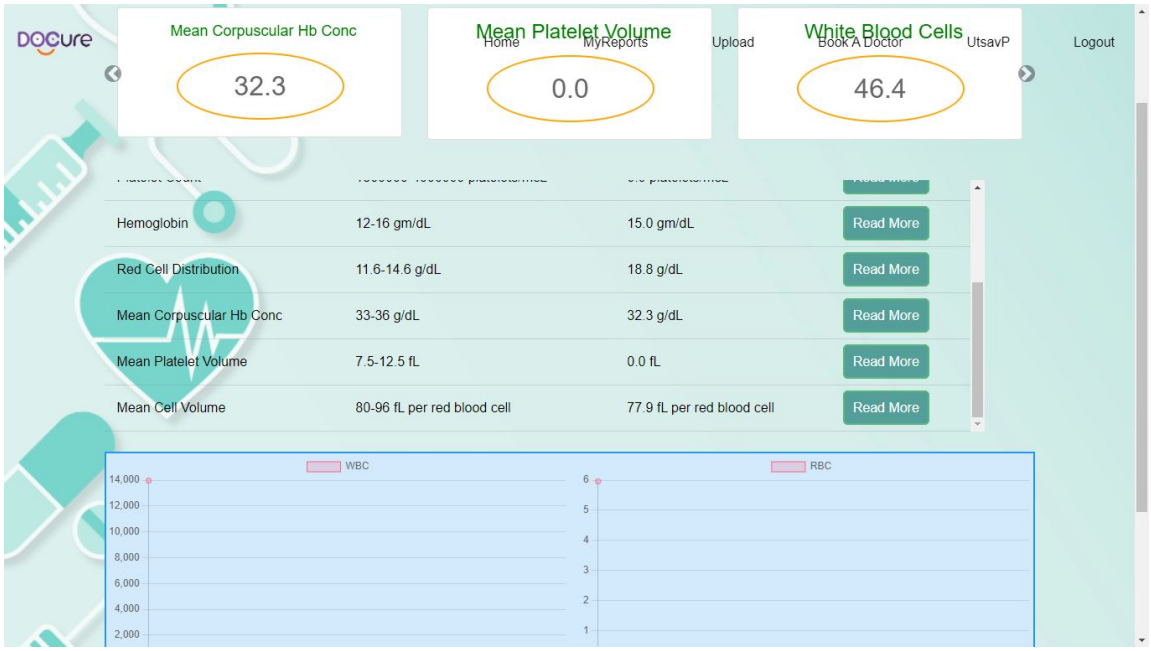


Figure 6.11: Report Dashboard

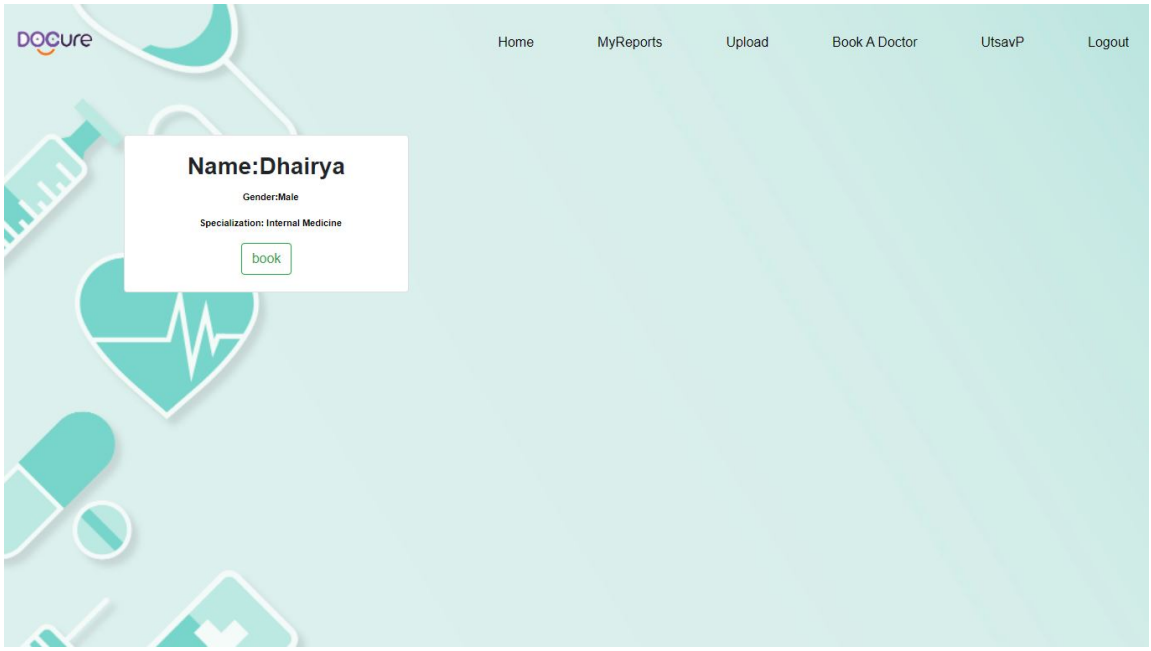


Figure 6.12: Book Doctors

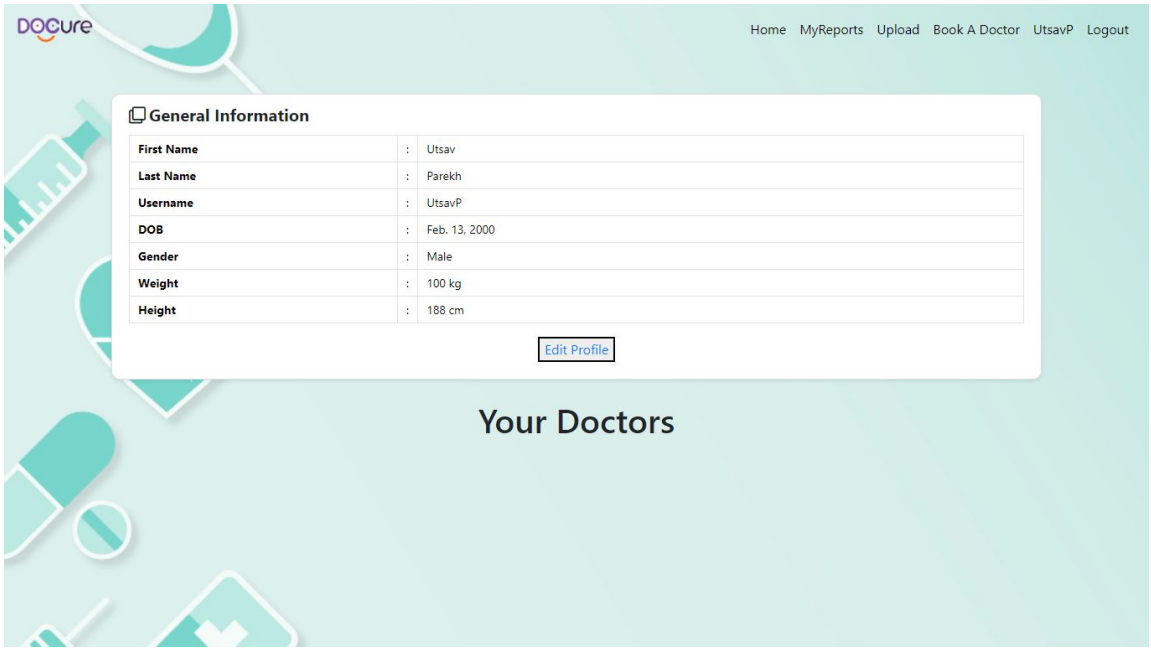


Figure 6.13: Patient Profile

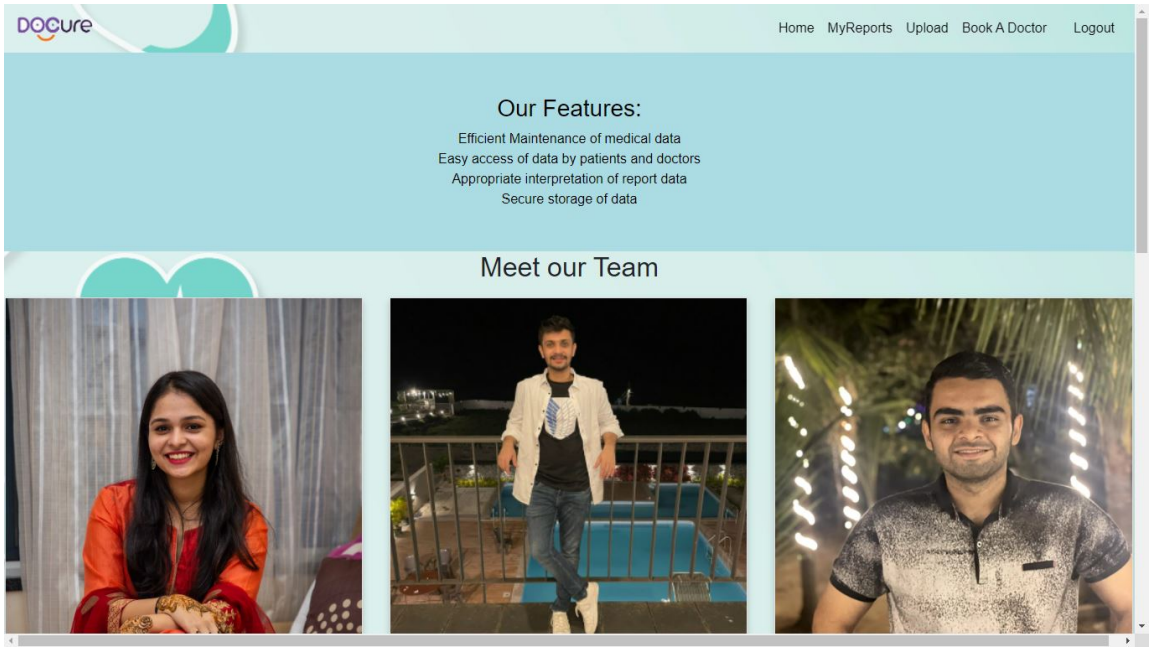


Figure 6.14: About Us

6.4.2 Doctors' Module

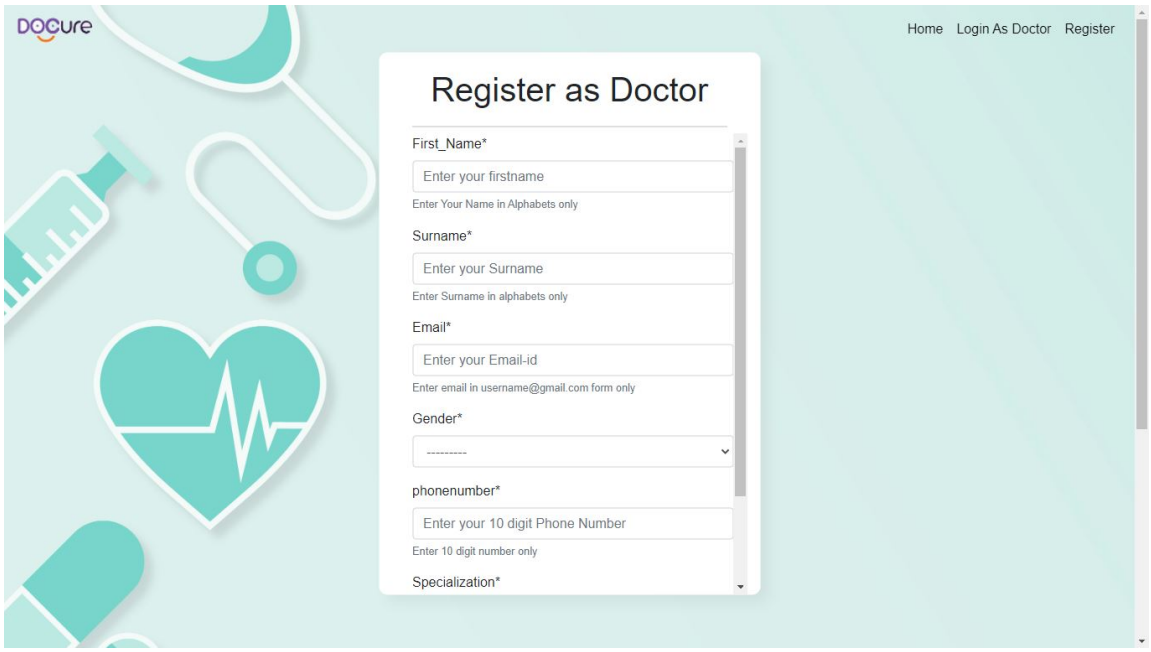


Figure 6.15: Doctor Registration

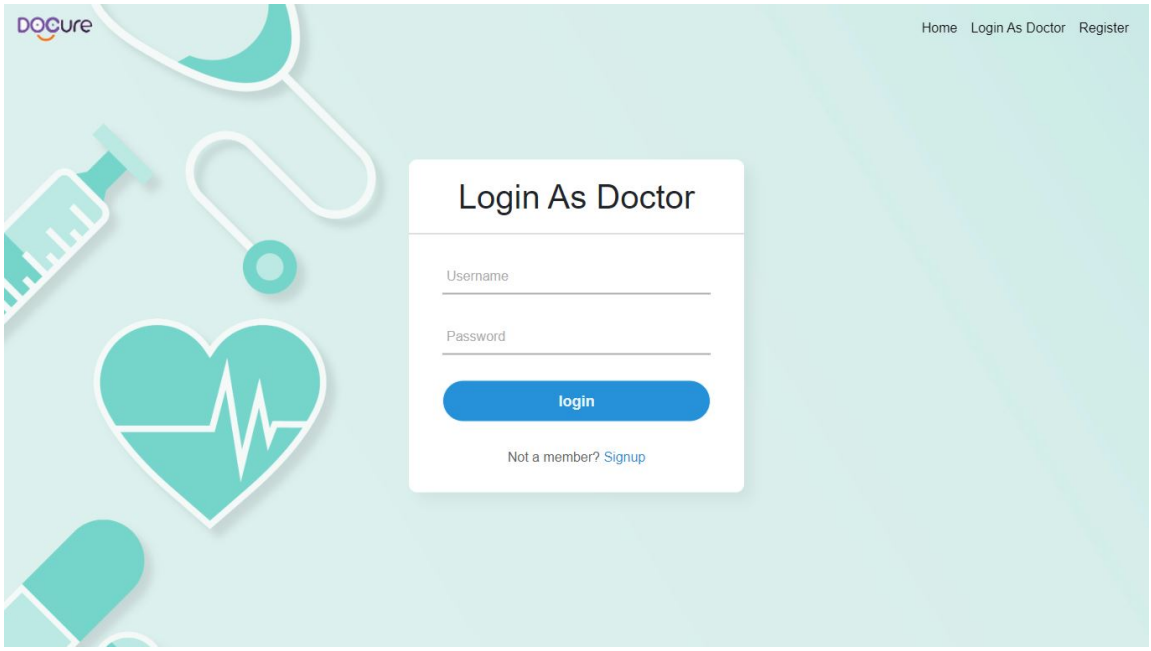


Figure 6.16: Doctor Login

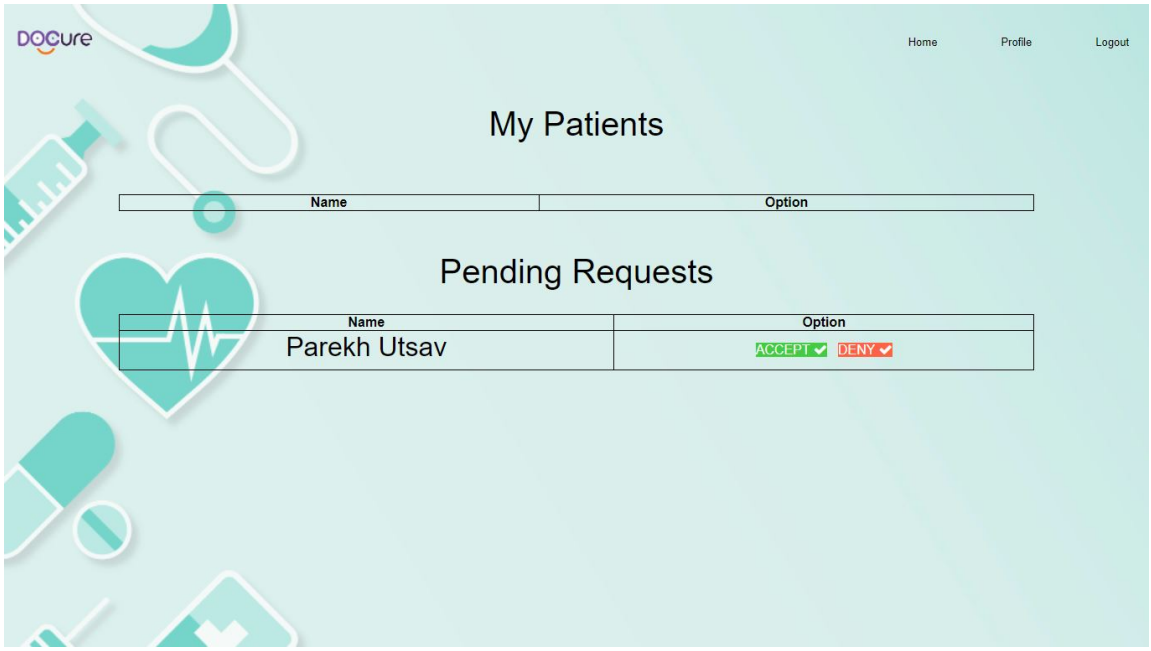


Figure 6.17: View Patients

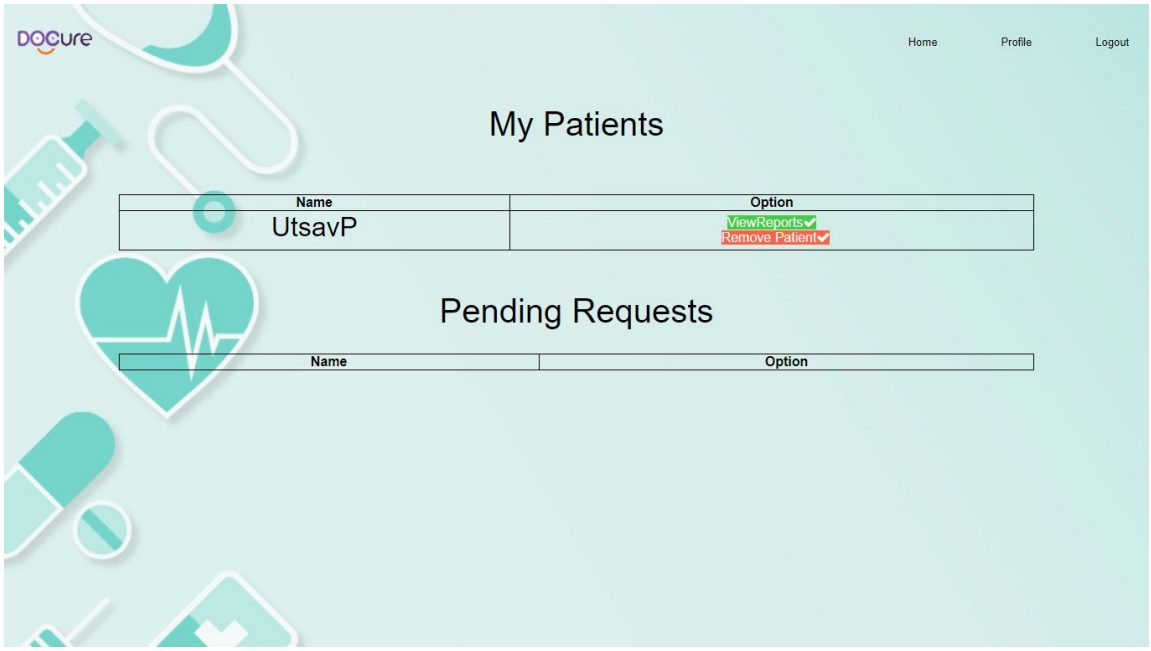


Figure 6.18: View Patient Reports

Figure 6.19: Doctor's Dashboard

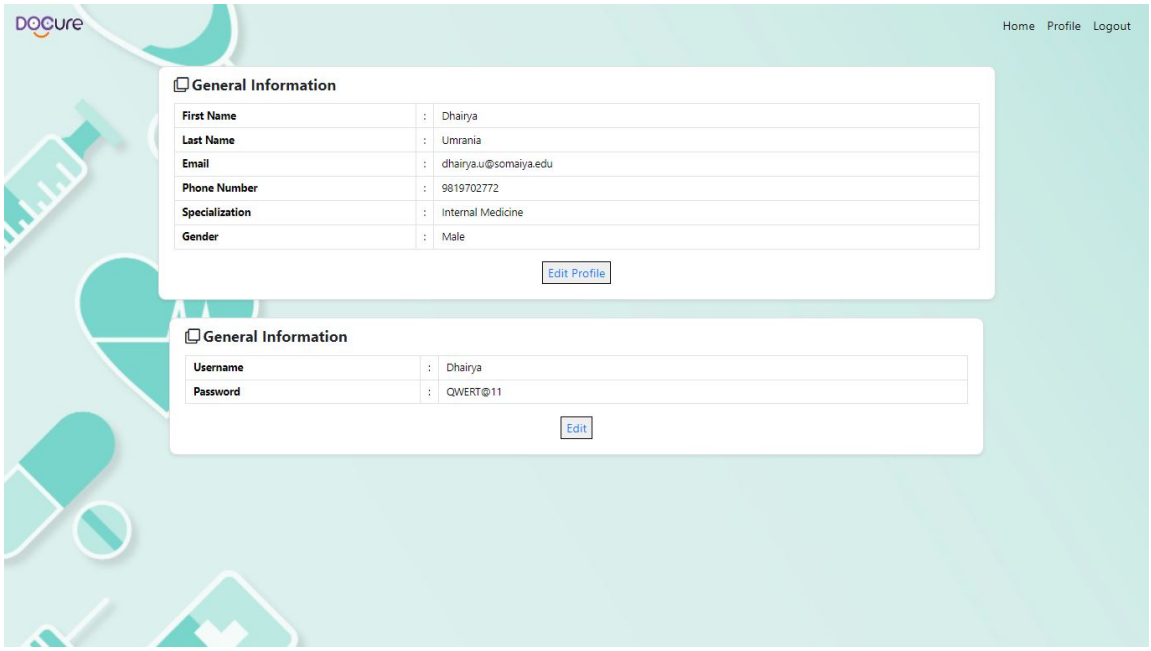


Figure 6.20: Doctor's Profile

Chapter 7

Software Test Document

This chapter provides the System Test Document which illustrates the test approach by stating the features to be tested and not to be tested. Testing tools and environment under which the system will go under test is explained with a details pertaining to the test cases to be performed.

7.1 Introduction

7.1.1 System Overview

DoCure is a web application that aims at easing the process of uploading medical reports while storing and retrieving medical data history. The application centralizes on safe storage and management on a web application with a database. The system is a web application that is based on DJango and SQL based database. The items to be tested include validation of the sign up/login page, validation of medical reports, correct interpretation of the report.

7.1.2 Test Approach

We are using the technique of proactive approach where the testing is started as early as possible in order to find and fix the defects before the build is created. For validation of the sign up/login page, creating the user profile and adding medical reports to the user profile, we use White box testing which is the software testing method in which internal structure is being known to tester who is going to test the software. For report analysis, we are using Black box testing which is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester.

7.2 Test Plan

7.2.1 Functions to be tested

Feature	Description	Test Cases
Design	Expected rendering and connection of Web Pages	TC 1.1, TC 1.2
Authentication and validation	Appropriate verification and validation of input fields and authentication of user details for access	TC 2.1, TC 2.2, TC 2.3
Upload Report	Uploading of Report Type in File Format Accepted	TC 3.1, TC 3.2, TC 3,3
Report processing	Processing of Report Uploaded Before Storing and Interpretation	TC 4.1, TC 4.2, TC 4.3, TC 4.4
Report data Interpretation	Interpretation of Report for Insights and Statistics	TC 5.1, TC 5.2
Report Display	Display of Report Statistics and Insights	TC 6.1
Doctors' access	Interaction of doctor with patient records and appointments	TC 7.1, TC 7.2, TC 7.3, TC 7.4

7.2.2 Functions not to be tested

- User Hardware Components - The hardware of the user is logistically impossible to test, we therefore rely on the user to use components similar or higher as prescribed in our SRS.
- Database storage capabilities - Since the database service in use is well established, its capabilities needn't be tested.

7.3 Testing Tools and Environment

Testing is done in a parallel approach, wherein the development schedule is in line with testing and improvisations based on the testing. For testing, we have used open-source testing tools and integration testing environments (ITEs). Testing approaches for automated and manual testing are used according to the nature of the software being tested. The majority of testing is manual apart from a few automated testing procedures for usability, load and performance testing. 'Lighthouse' was one such tool used as an automated testing tool for auditing the performance, which gives metrics for the accessibility, load time, etc. of the website.

7.4 Test Cases

7.4.1 Test Case 1

7.3.1.1 Test Case 1.1

Purpose: To check if interfaces are consistently rendered.
Input: None.
Expected Outputs and Pass/Fail Criteria: If the components in each of the interfaces are loaded properly in the defined structure without any spelling errors then the test will be passed else failed.
Test Procedure: Each page will be scrutinized properly, to ensure that every functionality is depicted properly using various icons/text fields as it is equally essential to ensure every functionality has a depiction. Also, it is necessary to ensure that no ambiguity is present.

7.3.1.2 Test Case 1.2

Purpose: To check if all pages are interconnected as required and all hyperlinks are working properly.
Input: Clicking the hyperlink.
Expected Outputs and Pass/Fail Criteria: Each hyperlink should lead to its desired destination page and should be able to carry any data if required from page to page. If all links are working as expected, the test will be passed else failed.
Test Procedure: Links on each page will be clicked and checked for the expected destination against the implemented destination to ensure that it is correct.

7.4.2 Test Case 2

7.3.2.1 Test Case 2.1

Purpose: To check if user is able to Login to the system using only correct credentials.
Input: Login Credentials
Expected Outputs and Pass/Fail Criteria: Upon entering the correct credentials, the user should be able to log into their accounts, else ale
Test Procedure: Initially, correct credentials are entered, to ensure the redirection, then invalid credentials are entered to check the generation of alerts.

7.3.2.2 Test Case 2.2

Purpose: To ensure the Sign-Up page is working as required.
Input: Sign-Up credentials.
Expected Outputs and Pass/Fail Criteria If the credentials provided, are unique and adhere to the several conditions such as the password field, then the User is created in the database else the user is alerted of changes to be made.
Test Procedure: Each field in the Sign-Up page is tested and checked for valid entries, initially the correct credentials are provided to check the redirection after which the invalid credentials are provided to check for alerting already existing users.

7.3.2.3 Test Case 2.3

Purpose: To ensure all form data is verified.

Input: Form credentials.
Expected Outputs and Pass/Fail Criteria If the credentials provided adhere to the different norms set for the data to be input, they are accepted else alerts are issued for them to be changed.
Test Procedure: Each field is tested and checked for valid entries, initially the correct credentials are provided to check the redirection after which the invalid credentials are provided to check for the alerts.

7.4.3 Test Case 3

7.3.3.1 Test Case 3.1
Purpose: To check if file is of supported type.
Input: File
Expected Outputs and Pass/Fail Criteria: The image file entered is checked if it is supported by the system. If the file is supported, it is accepted else user is alerted.
Test Procedure: The correct file is uploaded on the upload page, to check if the appropriate processing is performed, and unaccepted file type is uploaded to check the system alerting the user.

7.3.3.2 Test Case 3.2
Purpose: To check if report uploaded by user is of the selected type.
Input: Report type selection and file.
Expected Outputs and Pass/Fail Criteria: If the user has selected the correct file type, and uploaded the necessary file, the file is accepted and processed else the file is rejected and user is alerted.
Test Procedure: The report type is selected and the file is uploaded to check for the system response.

7.3.3.3 Test Case 3.3
Purpose: To check if locked file password provided by user is correct.
Input: Password for file.
Expected Outputs and Pass/Fail Criteria: The user provided password if correct opens the file, else the user is alerted for the filename.
Test Procedure: The password is entered while the user uploads the file on the upload page.

7.4.4 Test Case 4

7.3.4.1 Test Case 4.1
Purpose: To check if all data is correctly recognized.
Input: Image/PDF file
Expected Outputs and Pass/Fail Criteria: The system has to extract all required data from the report document.
Test Procedure: The image/PDF is uploaded and the system processes it.

7.3.4.2 Test Case 4.2

Purpose: To check if multi-page PDFs are processed.
Input: PDF file
Expected Outputs and Pass/Fail Criteria: The system is expected to parse through all pages in the document and extract its data.
Test Procedure: The PDF file is provided to the system through the upload page.

7.3.4.3 Test Case 4.3

Purpose: To ensure units in reports and system database are matched.
Input: Report uploaded by user.
Expected Outputs and Pass/Fail Criteria: The dashboard should display values from the user's uploaded report which are converted as per requirement of the system. If the dashboard shows values with correct units, the test is passed else the test is failed.
Test Procedure: The dashboard is loaded and report values are checked with the parameters' units.

7.3.4.4 Test Case 4.4

Purpose: To check if reports that are filtered are processed.
Input: Report with filters.
Expected Outputs and Pass/Fail Criteria: The reports that are filtered are processed by the system. If the report is correctly processed, the test case is passed, else it is failed.
Test Procedure: Report scanned using apps like Adobe Lens, are uploaded to the system.

7.4.5 Test Case 5

7.3.5.1 Test Case 5.1

Purpose: To check if the data is correctly interpreted.
Input: Extracted data from reports.
Expected Outputs and Pass/Fail Criteria: The data that is extracted from report is correctly interpreted and stored.
Test Procedure: The report is provided to the system for processing and storage.

7.3.5.2 Test Case 5.2

Purpose: To ensure dashboard values match report parameters.
Input: Report parameters
Expected Outputs and Pass/Fail Criteria: The dashboard should only display values from the user's uploaded report. If the dashboard shows appropriate values, the test is passed else the test is failed.
Test Procedure: The dashboard is loaded and report values are checked with the parameters.

7.4.6 Test Case 6

7.3.6.1 Test Case 6.1

Purpose: To ensure all data is corrected displayed in the dashboard.

Input: Data from database.
Expected Outputs and Pass/Fail Criteria: The data from the database is retrieved and graphically displayed appropriately and simply for the user to understand.
Test Procedure: The dashboard is opened by user for a particular report.

7.4.7 Test Case 7

7.3.7.1 Test Case 7.1

Purpose: To ensure doctor usernames are unique for logging in.
Input: Data from database and user input.
Expected Outputs and Pass/Fail Criteria: The doctor is alerted if their username isn't unique. If doctor's usernames aren't duplicated, the test case is passed else the test is failed.
Test Procedure: The doctor has to register on the system.
Result: Pass

7.3.7.2 Test Case 7.2

Purpose: To check if doctor's status is reflected on the user end.
Input: None.
Expected Outputs and Pass/Fail Criteria: The doctor's appointment status is reflected on the user's end as accepted, rejected or pending. If the status is reflected correctly, the test case is passed else it is failed.
Test Procedure: The doctor has to accept or reject the patient who has applied for appointment.
Result: Pass

7.3.7.3 Test Case 7.3

Purpose: To check doctor can access patient reports.
Input: Data from the database
Expected Outputs and Pass/Fail Criteria: The doctor should be able to access the patient's reports. If the doctor is able to access the patients' reports then the test case is passed, else it is failed.
Test Procedure: The doctor has to access the user's patients.
Result: Pass

7.3.7.4 Test Case 7.4

Purpose: To check doctor's comments are reflected on user dashboard.
Input: None.
Expected Outputs and Pass/Fail Criteria: The doctor's comments should be accessible on patients' dashboard. If the user is able to access the patients' dashboard for comments then the test case is passed, else it is failed.
Test Procedure: The user has to access the user's dashboard.
Result: Pass

Test Images:

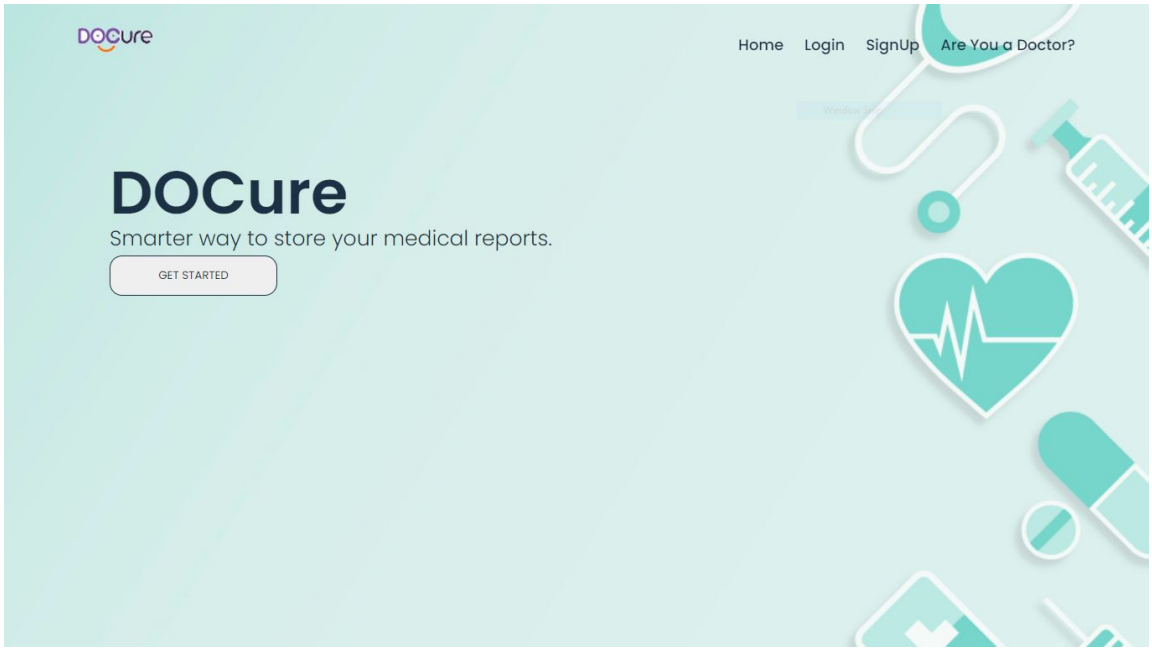


Figure 7.1: Test Case 1.1

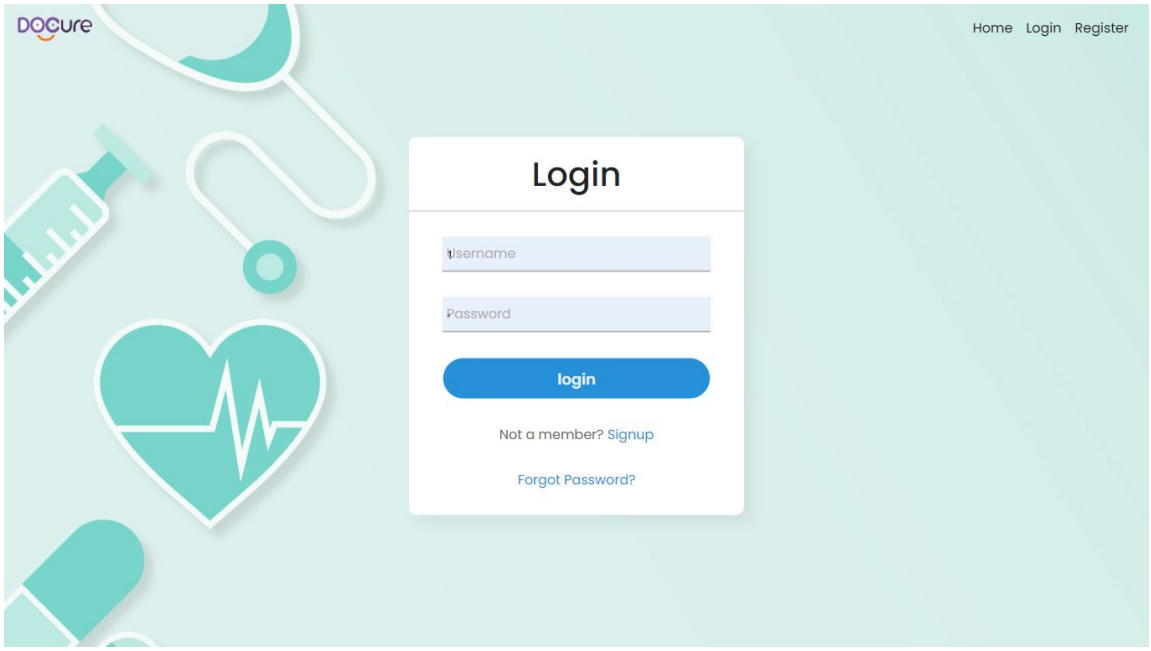


Figure 7.2: Test Case 1.2

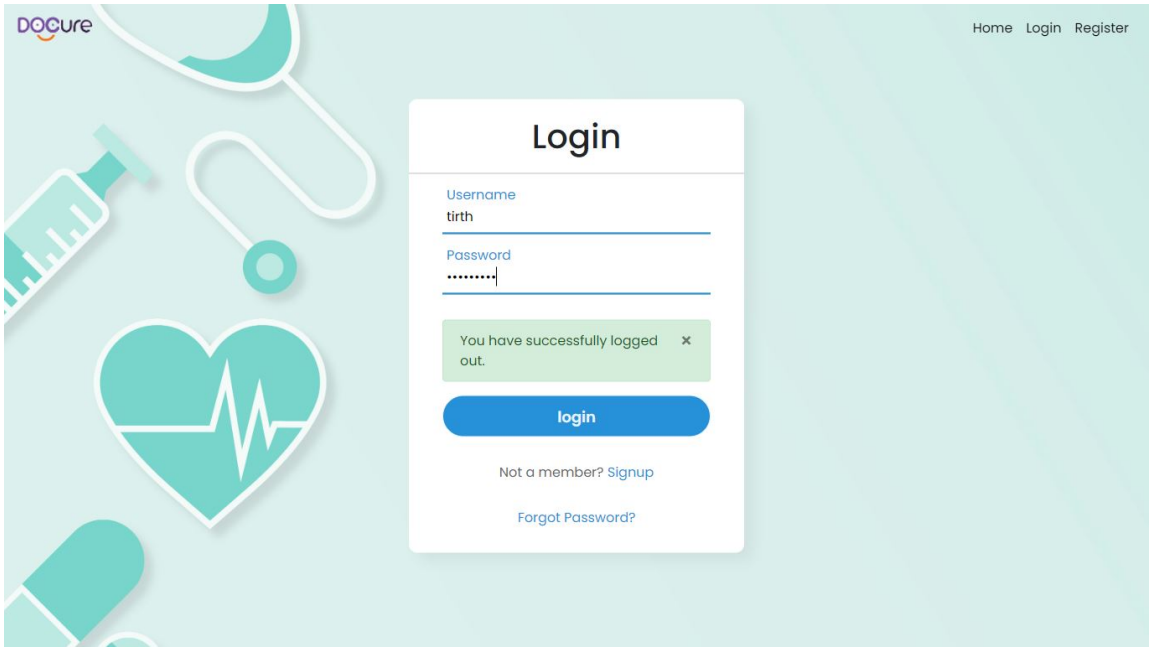


Figure 7.3: Test Case 2.1

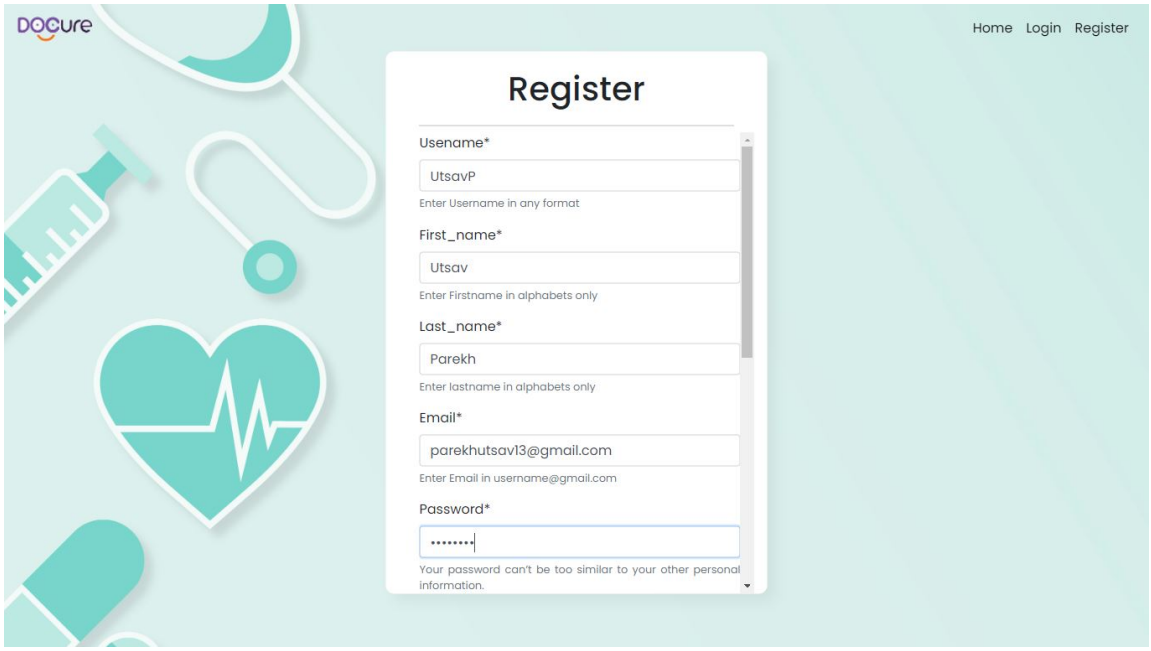


Figure 7.4: Test Case 2.2

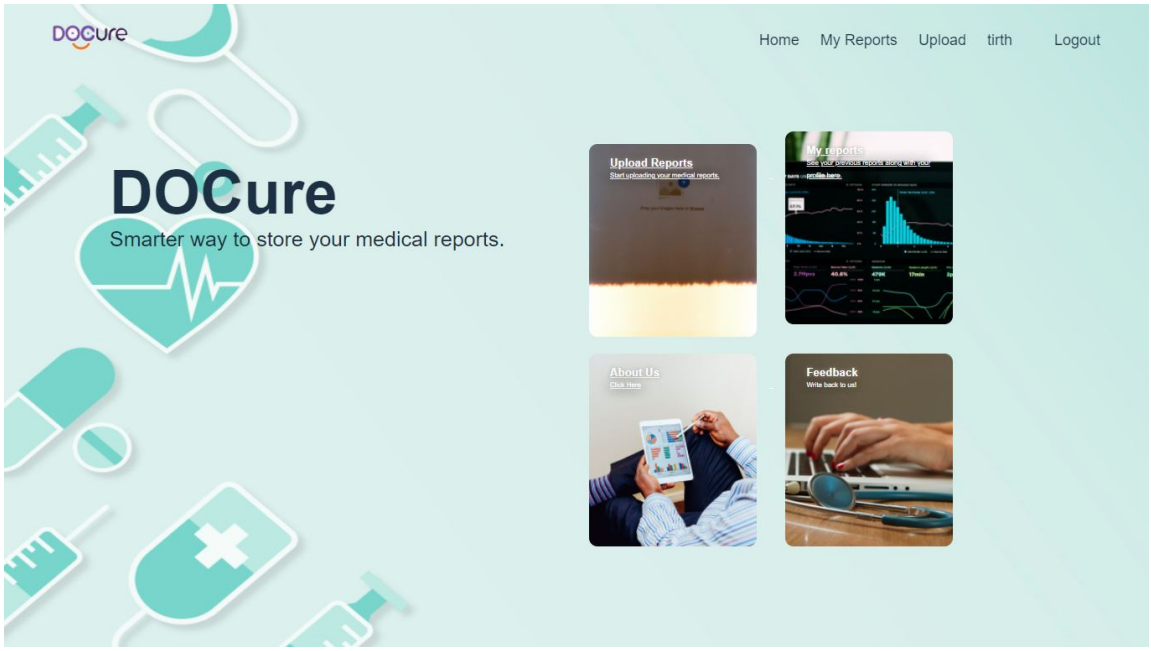


Figure 7.5: Test Case 2.3

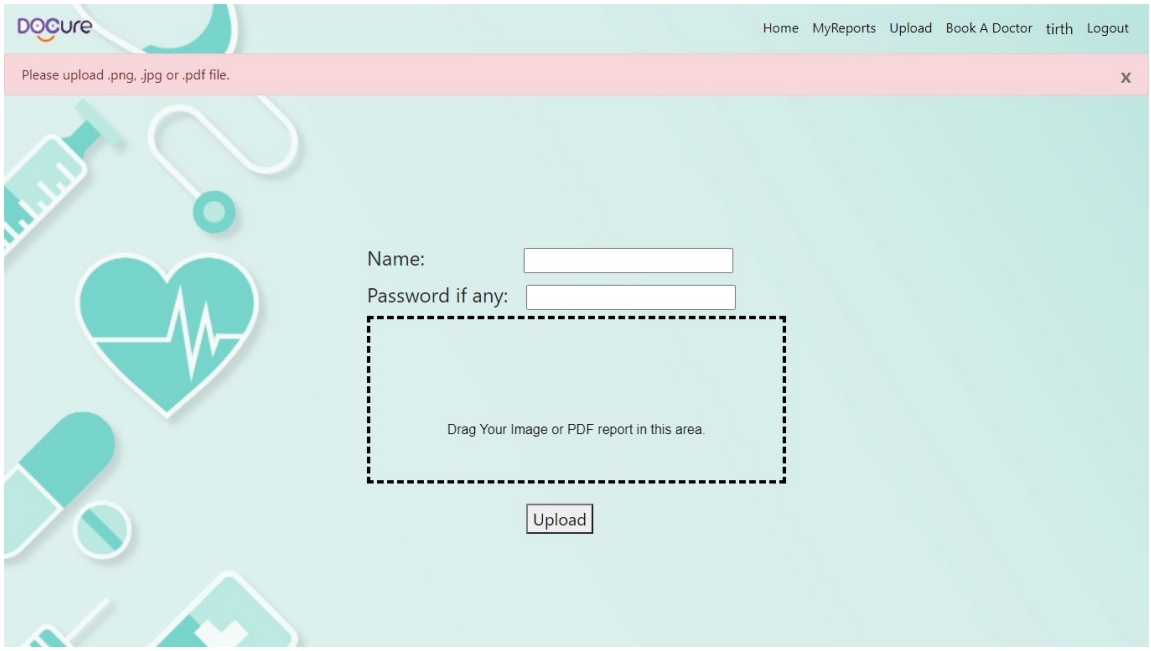


Figure 7.6: Test Case 3.1

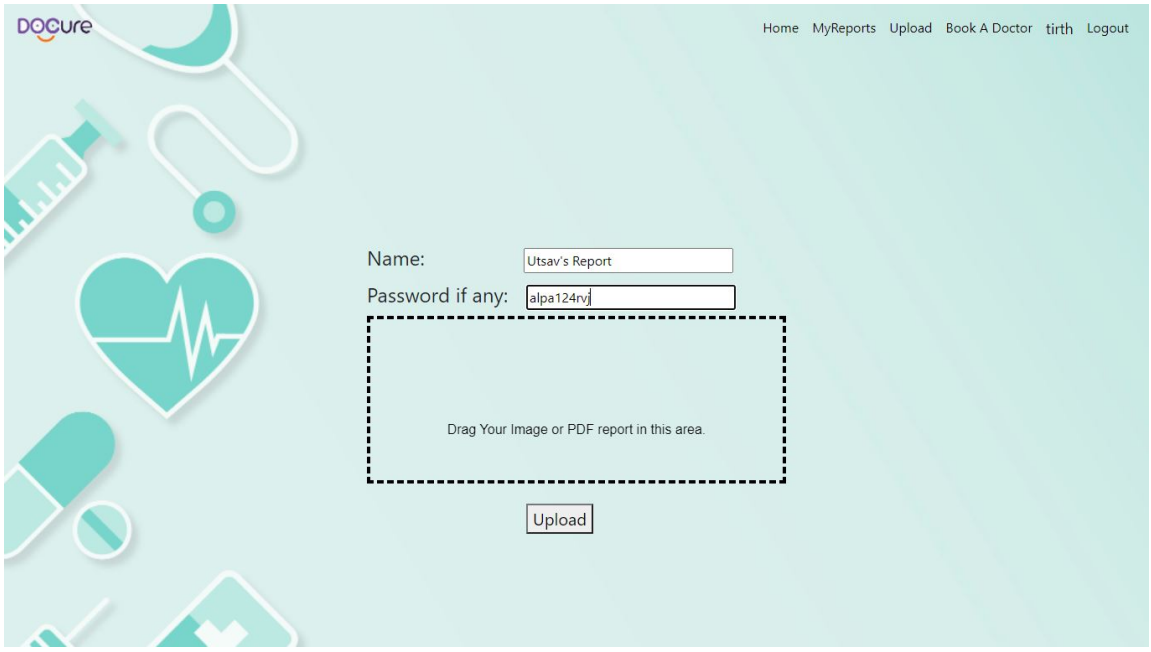


Figure 7.7: Test Case 3.3

Figure 7.8: Test Case 4.1

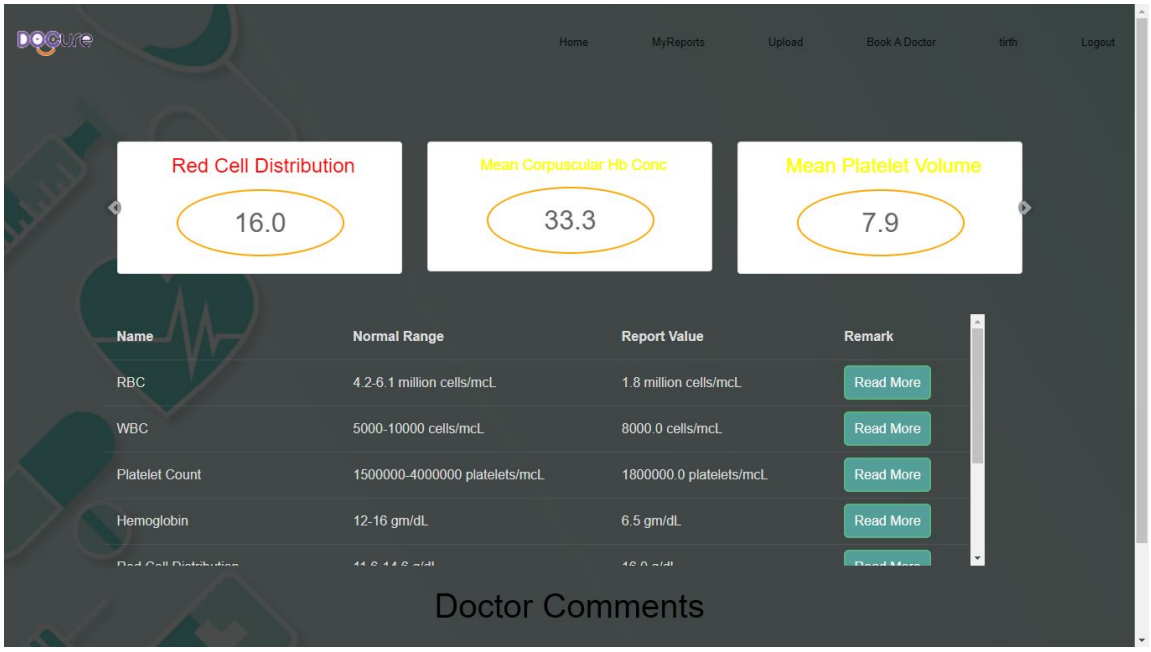


Figure 7.9: Test Case 6.1

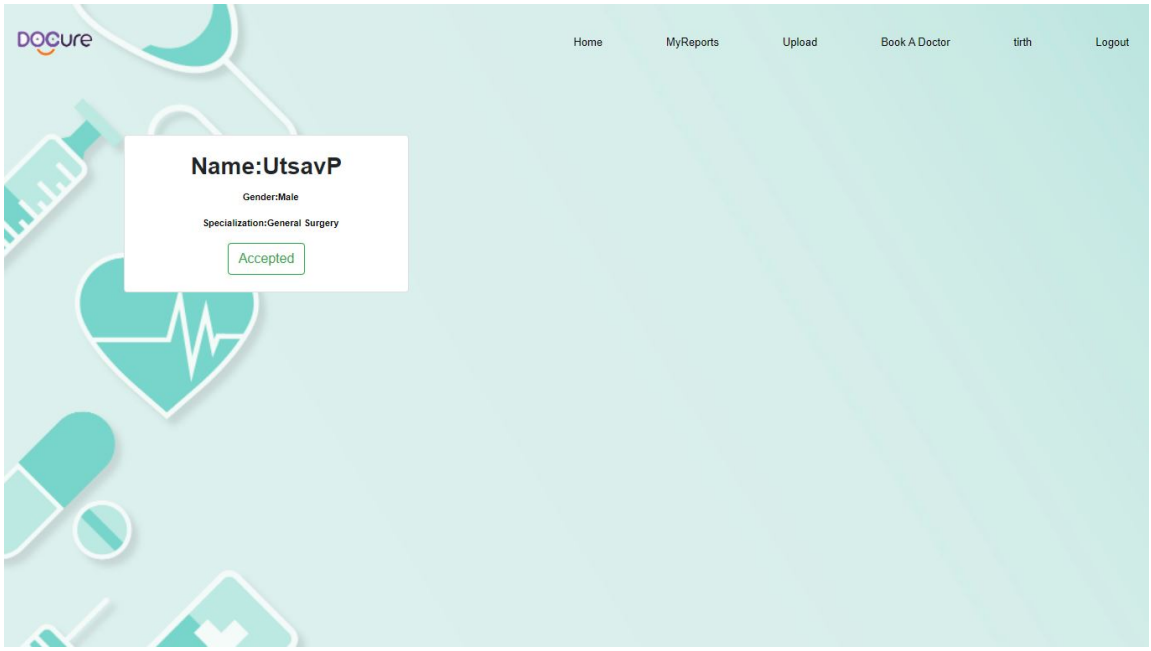


Figure 7.10: Test Case 7.1

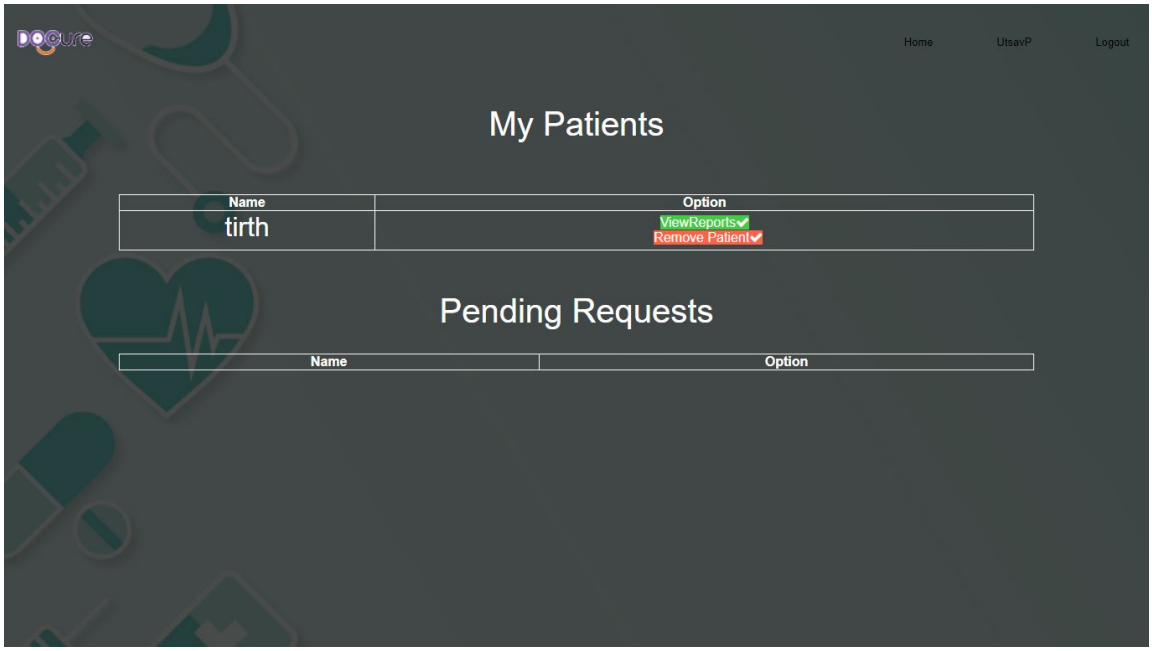


Figure 7.11: Test Case 7.2

Chapter 8

Conclusion and Future Work

The final chapter concludes the thesis with a detailed conclusion and explaining the future scope of the project undertaken by the team.

8.1 Conclusion

We have successfully implemented the modules of Data Recognition, Data Extraction, Data Storage, and Visualization. Apart from this, we have implemented the Doctor's module too. Along with them, the documentation i.e., the SPMP, SRS, SDD and STD have also been completed and compiled. We have built text recognition from Images and PDFs using Optical Character Recognition and PDF Plumber respectively. We created a dataset of medical reports for OCR and extracted the text to create text corpus for text mining machine learning algorithm which utilizes Named Entity Recognition (NER), combined with Regex to extract the parameters of CBCs. The SQLite database has been integrated to store data in an encrypted format. The data is visualized and interpreted to be displayed on the dashboard. The doctor's module includes choosing patients and accessing their reports alongside adding comments for the patient.

8.2 Future Work

For future work, we look forward to integrate more report types into our application. We also aim to work on producing better OCR readability accuracy which can in turn increase the overall efficiency of the project. We could work with medical institutions for better doctor involvement in our application which helps with our patient interaction. We also aim to improve our User Interface and make it more interactive and user friendly.

Bibliography

[1] https://www.djangoproject.com/

[2] https://www.python.org/

[3] https://jupyter.org/

[4] https://github.com/tesseract-ocr/

[5] https://github.com/jsvine/pdfplumber

[6] https://www.sqlite.org/index.html

[7] An OCR Post-correction Approach using Deep Learning for Processing Medical Reports

[8] Building Structured Personal Health Records from Photographs of Printed Medical Records Xiang Li, PhD1, Gang Hu, MS1, Xiaofei Teng, PhD1, Guotong Xie, MS1 1IBM Research, Beijing, China

[9] Extracting information from the text of electronic medical records to improve case detection: a systematic review Elizabeth Ford,1 John A Carroll,2 Helen E Smith,1 Donia Scott,2 and Jackie A Cassell1

[10] HEDEA: A Python Tool for Extracting and Analysing Semi-structured Information from Medical Records Anshul Aggarwal, BE, Sunita Garhwal, PhD, Ajay Kumar, PhD Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, India

[11] Medical Document Reader on Android Smartphone Arrart Kongtaln, Sutthipong Minsakorn, Lalita Yodchaloemkul, Sirasit Boontarak, Sukanya Phongsuphap Faculty of Infonnation and Communication Technology Mahidol University Salaya, Nakhon Pathom 73170, Thailand

[12] Structured Information Extraction of Pathology Reports with Attention-based Graph Convolutional Network Jialun Wu1,2,3, Kaiwen Tang4, Haichuan Zhang1,2,3, Chunbao Wang5, Chen Li1,2,3* 1 School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China 2 National Engineering Lab for Big Data Analytics Xi'an Jiaotong University, Xi'an, Shaanxi 710049

[13] Study of Information Extraction and Optical Character Recognition Swayanshu Shanti Pragnya Dept. of Computer Science and Engineering, CUTM, Bhubaneswar, Odisha, India

[14] Table Analysis and Information Extraction for Medical Laboratory Reports Wenyuan Xue, Qingyong Li, Zhen Zhang, Yulei Zhao, Hao Wang Beijing Key Lab of Transportation Data Analysis and Mining Beijing Jiaotong University

[15] Text Detection and Recognition for Images of Medical Laboratory Reports with a Deep Learning Approach

[16] A Blockchain Framework for Secure Electronic Health Records in Healthcare Industry Mohammad Tabrez Quasim1, Alaa Abd Elhamid Radwan1, Goram Mufareh M Alshmrani1, Mohammad Meraj21 College of Computing Information Technology University of Bisha, Bisha, Saudi Arabia 2 PhD Scholar, NIMS University

[17] Custom Made Medical Data Reporting Tool Petar J. Rajkovic1, Dragan S. Jankovic2

[18] A New Supervision Strategy based on Blockchain for Electronic Health Records

Authors' Publication

This chapter includes the Research Paper titled DoCure: The Smart Way to Maintain your Health Records based on the facts and findings found towards the end of the project. Essential metrics and factors of considerations are included in the paper.

DoCure: The Smart Way to Maintain your Health Records

Anushka Darade, Dhairya Umrania, Tirth Thaker, Utsav Parekh

[1] Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India
[2] Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India
[3]Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India
[4] Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India

***Abstract*— Health is an important aspect of every life. For all the necessary records related to health, medical records are employed. Traditionally these are in the form of paper and can often be misplaced, or suffer from tear and damage The project aims at resolving this problem by using an Electronic Health Record system that scans and stores the medical report data on the database for anytime access from the user. Users can directly upload their report in the form of images or PDFs, that are processed and the data is stored in the database. When the user wants to access this data, they can directly see the data as well as several analyses that are available on the platform. The proposed approach is designed keeping in mind the different technological intricacies involved to provide a fast, secure, and most importantly a very user-friendly application that can be operated with ease.**

***Keywords*— Medical report data extraction, medical report data storage, OCR (Optical Character Recognition), EHR (Electronic Health Records) , EMR (Electronic Medical Records), PHR (Personal Health Records), semi-structured document extraction.**

I. INTRODUCTION

Revolving around today's era, there is a lot of medical research surfacing new diseases, new symptoms as well as new medical reports. Layman faces several problems in maintaining their medical data efficiently and updating it time to time without loss of any reports. These reports are not only required for medical purposes but in the pandemic,
such reports hold a lot of significance for daily activities. In such a situation, having an alternative to store medical reports, and related data eases several aspects for people.
Some of the commonly noticed problems that have been discovered are:
• Frequent expensive consultations
•Loss of medical data due to inefficient maintenance
• Irregular updation of data
• Difficult access of data by patients and doctors
• Improper interpretation of report data
• Insecure storage of data
Medical health covers several different and important aspects of human life. Medical reports are maintained by paper basis which are traditionally exposed to wear and tear. Due to several problems faced in maintenance and updating of medical data, there is an inaccuracy in the provision of health services which can often prove to be fatal. We
are attempting to reduce this issue by the introduction of an Electronic Health Record system which stores the data in a digital format. Our project aims at helping those who aren't well versed with medical jargon to understand and analyze their reports in an easier and simpler way.

II. LITERATURE SURVEY

In [1], the authors have discussed how the lack of medical terms, specific to the domain, while training OCR's can cause inaccuracies in the OCR. An OCR is trained on general words used in the English language, so particular medical terms may not get recognised. This paper suggests a Post Processing method, to counter this problem.
The paper uses OCR Tesseract Engine, and uses the RoBERTa language model to find wrongly predicted words. For evaluation, the dataset on MiBio is used, and a few medical reports collected from the NHS are used and to measure the quality of the OCR the Word Error Rate (WER) and Character Error Rate(CER) are calculated.
After running experiments, the paper found that the WER and CER were reduced by implementing post processing using their proposed method. Before post processing the Average insertions, substitutions and deletions were 0.93, 0.812 and 0.858 respectively. After post processing it reduced to 0.813, 0.645 and 0.711 respectively.
[2] proposes a pipeline to extract text from scanned medical records. The Dataset Corpus consists of 100 medical report documents collected from hospitals in China. The images of these images were taken from several different mobile devices such as iPhone, Nexus, Xiaomi, etc. In total, the dataset consists of over 1400 images.
The pipeline starts with clicking a picture, then, there is the image pre-processing stage. Here the images are denoised, binarized and deskewed. After that, the OCR is run on the images. The OCR engine used is the IBM Datacap Taskmaster Capture 8.1. After this, it's the post-processing stage, where word resegmentation and Multi-engine Synthesis occurs. After this the annotations are done, where each entity, such as numbers and labels are annotated. Finally the Personal Health Record (PHR) is created where a confidence threshold is

measured, and if it passes it, the document is constructed and stored in the database.
For evaluation, the precision, recall and F-measure was calculated. The results were calculated after each step in the pipeline for all the different categories. They achieved a result of F1 score of 0.918 for Diagnosis name detection, 0.845 for medication name detection and 0.922 for test item name detection for entity detection.
[3] explores 3 different OCR engines: Tesseract, Amazon Textract and Google Document AI on images of English and Arabic document images.
The dataset comprises 322 color page scans from books printed between 1853 and 1920, which are part of the English Language dataset. For the Arabic dataset, 4587 articles were printed from wikipedia and scanned to pdf. From this 100 randomly selected pages were chosen. The authors also discussed how important it is to have noise in the dataset so that it reflects real world scenarios more accurately. So to implement this, the authors created 43 copies of each image which had different types of noise embedded in them such as blurs, watermarks, scribbles, etc. In total this generated a total corpus of over 14000 English documents and 4400 Arabic documents.
For testing metrics of each OCR, the accuracy was measured using the ISRI tool, known as Ocreval. It calculates the Levenshtein distance which is then used to calculate the Word Accuracy.
The paper concluded that Document AI had the lowest error, followed by Textract, followed by Tesseract. The error rate went up for all the OCR engines when noise was implemented in the dataset. Another observation was that the engines performed worse for Arabic documents than English documents.
The paper has explained in detail a document image, which is a kind of camera-based image obtained from an Android smartphone. The method utilizes the Tesseract OCR engine, whose engine consists of 3 main components, Connected Component Analysis, Line and Region Analysis, and Word Recognition, which works together to extract text from an image.
This word recognition module begins the recognition process by first identifying how a word should be separated into characters. The results from line analysis are classified first. The non-fixed pitch text is then processed by the rest of the word recognition. Then the Linguistic Analysis unit is called on.
As described in [4], a lot of clinical data in regards to the archival data of a patient is in the form of clinical reports (CR) which are inserted into medical records. The texts describe the patient details, histories and findings during procedures. An efficient system is required for extracting information in a structured way so that it can be analyzed accurately. The authors have firstly worked on the process of extraction of information. CRs have been collected and analysis has been performed, these CR contain observations from certain examinations, measurements, etc. The authors assembled a corpus of hundred CRs wherein 50 reports were training corpus, and the rest for the test phase. The major goal was the automatic structuring of the clinical report text into sections that were defined earlier to serve as a pre-processing step for the entity extraction. The report begins with patient identification details, history, text giving information about symptoms, findings and evolution of the patient, etc. A rule based algorithm was developed for the segmentation of the report into sections. The linguistic characteristics that are a part of the grammatical categories of the nominal phrases (NP) serve as a source for the recognition. The authors first focus on tokenizing and tag words using the TreeTagger tool. The tagged medical report is parsed to detect nominal phrases. A filter is applied to favor the longest NP over others. The template is shown for the MedIX tools which is a structured object representing the entities and properties.
Process extraction is based on the rule approach and context surrounding the information. The authors have incorporated several dictionaries from several resources for tagging the NP components. There are four features decided to encode for each NP. The first one represents part of speech information of each token present in NP that can be extended with a second feature. The third feature is the NP length and fourth being the contextual information. Privacy is a major concern and so the patient data is anonymized before using it. In the research, a new anonymous identifier is generated by the Standard Hash Algorithm for each patient, in the structured database, the generated code is used for identification.
As a result, 50 reports were processed and out of 651 entities and properties, MedIX accurately matched 450 and missed 201, identified 3 entities erroneously. This gave a precision of 98.9% and a recall of 68.8%.
[4] proposes a method to extract data from semi structured PDFs. The PDF file will be passed to a PDF parser, which will extract the raw text from it. After that, the text will be passed to a line picker which will iterate through the text line by line, and on each line there will be a boundary extractor which will extract the boundaries, and a pattern matcher such as Regular Expressions will be used to locate text that is to be extracted. After that, the extracted text will be entered into the database, from which it will be available for analysis.
[6] proposes a cloud based Electronic Health Record (EHR) system which facilitates a storage for patient medical report data and since it is on the cloud, it provides a key generation system with the Key Attributes Method. This also allows the patient to control access points to their data, which means they can grant access to their doctors or relatives as well.
The authors have also provided a framework for the cloud which will host the EHR.

In [7], methods of medical data extraction from databases/data warehouses using OLAP tools, and then generating a report from that data is mentioned. The authors of this paper have created their own Custom made Medical report generation software, which does the above said things and generates a report for the same.
OLAP stands for Online Analytical Processing and it is used in various complex forms of data analysis such as reporting, summaries, KPI's, analysis of trends, time series forecasting and top down hierarchical analysis. The OLAP server has proven to be the most important component, it is present between the client and the database management systems. OLAP servers understand the types of data structures present in the databases and have special functions and methods which can be used for analysis of those data structures.
Electronic Health Records (EHR), are used for storage of patient health records and tracking of patient medical health. For that a database is required, in this paper, the authors have proposed a EHR metadata model which consists of 6 tables, named: 'profile', 'Tables', 'MeasuementUnit', 'ProfileItem', 'Fields' and ranges, in the form of a relational schema.
Lastly, the authors propose a report generating tool which will retrieve patient data and display it to the user in the form of tables and charts, which the user can download for reference, in the form of XML files or PDF files.
[8] mentions how the current state of storage of medical reports is and what are the subsequent problems of the same. It mentions how to store medical data in a structured fashion and why it is better than conventional methods.
If report data is stored in a structured format, it has many advantages such as: user customized data, historical data retrieval, interpretation analysis and structured medical data can also be used to practice evidence based medicine.
This paper proposes a standardized structure for report data, and mentions that it is an absolute prerequisite for structured medical data. It will also ease the process of data mining, data retrieval and data output and make it quicker and more versatile for different processes and requirements.
This paper proposes a table like timeline for patient data with fixed parameters. The table contains parameters such as: Date, Modality, Anatomy, Findings, Clinical Significance, Interval change and Follow-up recommendations. It is tracked through time and since it is in a table format, which is similar to DBMS relational tables, the data can be filtered as well as per need.
This paper believes that standardizing structured medical data across every platform and institution can improve efficiency of data retrieval, facilitate data driven analysis, improve workflow, and many other aspects which will ultimately improve clinical outcomes and patient safety.
In [9], Medical institutions store a lot of medical data, while more and more data is being collected to support the growing medical informatics. Good record keeping helps in smooth communication and better understanding between patients, doctors and other related professionals. Electronic Health Records (EHR) is introduced for sharing this information, making it available and securely to authorized users. Health Data Visualization is a way to gain insights derived from EHR. People can understand and act upon information by seeing the results in a visual context.
The deformable human body model is built using a three step statistical analysis method. The Statistical shape model uses the Generalized Procrustes Analysis to analyze real differentiation of human body data as well as changes of parameters such as height and weight of the mode. The Principal Component Analysis divides the samples into male and female data. The internal deformation parameters are extracted between these by PCA. The interpolation algorithm is used on height, weight and other parameters. The customized 3D model of the human body is able to be generated when the real data is provided. The user can generate their 3D digital human body with organs, muscles, etc. after the parameters are entered and the model is generated on their smartphone. The procedure has 4 main parts. The image preprocessing is performed so that the irrelevant data can be eliminated, the useful information can be saved and enhanced. OCR is used for digitizing the medical records The third step is the OCR results analysis, page ranking is implemented on the OCR results, the page style is recognized and the page is digitally recreated with the obtained information. The header and footer content is recognized when the contents are in the same position. Lastly, the OCR results and the medical database double check method is applied, to ensure the correctness of the data. The database is built by completing the schema with the help of professionals including entities and relationships. Filling the database with entities, relationships and related properties to complete the relation and filling the database after data scrubbing by medical professionals.

III. DATASET DESCRIPTION

OCR dataset: Old-books dataset-Old scanned books dataset with groundtruth. The groundtruth was built with Project Gutenberg ebooks. All the .tiff pages were converted from project Internet Archive's books (PDFs).
As there are no online datasets for medical report images, we will be using the strategy used by the authors of "Building Structured Personal Health Records from Photographs of Printed Medical Records", where they created their own dataset by clicking pictures of their personal medical reports. They took images of the reports from different angles and in different lighting conditions to generate a larger corpus of training data for the OCR and also to improve efficacy of OCR.

Our dataset:

Total	CBC	Biochemistry	Urine	Other
180	73	50	12	45

Table - 1: Dataset records

IV. Implementation

A. Recognition

IV.A.1 Optical Character Recognition

Text recognition involves identification of text from a document. Our project includes the recognition of images and PDFs. Since reports can be in paper or digital format, provision for images and PDFs is necessary to provide reports to the system. Our system identifies the file type and runs the recognition on the document accordingly.

Optical Character Recognition is the recognition of text from images by scanning the image and retrieving all the text data from the same. Our system makes use of Nanonets and Tesseract OCR Engine for performing OCR on images. Nanonets is an automated tool for data recognition and labeling on the web. It employs the Nanonets API which passes the image and gets the OCR response over HTTPS protocol in the form of response. Tesseract is an OCR engine that uses a Python library to pass the image and retrieve the text in the form of string from it. Tesseract supports various outputs like plain text, hOCR (HTML), etc.

Our application stores the uploaded image which is passed onto the OCR module. The OCR module detects and returns all the necessary strings over which the processing and extraction procedures take place.

IV.A.2 PDF Recognition

The system also has provision for scanning PDF files and retrieving the data from them for processing. This is performed using inbuilt libraries and provides for password-locked files too.

B. Text Extraction

To get the parameters and their values from the raw text generated by the OCR or the PDF reader, named entity recognition (NER) along with regular expressions (Regex) is used. For named entity recognition, a machine learning approach is implemented, where the training data consists of the text, the named entities in the text and their spans. The dataset used for this is the text of 49 CBC reports generated from the OCR, along with their entities and spans. The data is shuffled and split into 70:30 for training and testing, and the training data is further split into 50:50 for training and validation.

The NER pipeline consists of a tokenizer, tagger, parser and NER. Tokenizer, tagger and parser components are used to pre-process the text before feeding it to the NER component. The NER is done using word embeddings using subword

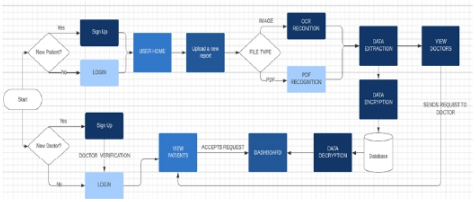


Fig 1: Block diagram

features and bloom embeddings which is a deep convolutional network with residual connections.

C. Web Application

HomeBefore Page is the initial Page of our Web App which contains links to the Login Page, and the Register Page for patients and Doctors. After clicking on the Register Page We ask Patients for general information to enter. Then after proper verification and validation of the Form, it will go to the Login Page. The Login Page can only be logged in to the Registered Patient who has registered on our Website, after verification of credentials, it will log in to our Website home page. Home Page contains all links to Upload Reports Page, My Reports page, to view all the reports uploaded by patients, about us page, to get information about the website, Feedback page to give feedback about our WebApp. The upload page contains the two input boxes: the name of the file uploaded by the Patient and a Password input. The Patient can enter the password if the PDF file requires a Password. After the file is uploaded, the report is sent to the OCR and text extraction module, from which the data is extracted.

After extracting the report data, it will direct to the ConfirmForm page. The ConfirmForm Page takes the Extracted Data from PDF/IMAGE in the form where the Patient can see the extracted values of his/her Reports. It helps Patients to enter values of reports which were misinterpreted by our Text Mining Techniques or OCR Techniques. It also helps the Patient to change the name of the report which Patient wants to set. After Clicking on Submit, the data will be stored in the database in encrypted form and will be directed to the “My Reports" page. My reports contain all Reports Uploaded by Patients in a tabular Format. The Table contains the upload time of the Report, the Name of the Report, the Type of Report, a link of “View Report " to see the Dashboard analysis of the report, and “Delete Report " from which a Patient Can delete Reports. After clicking on View Reports,

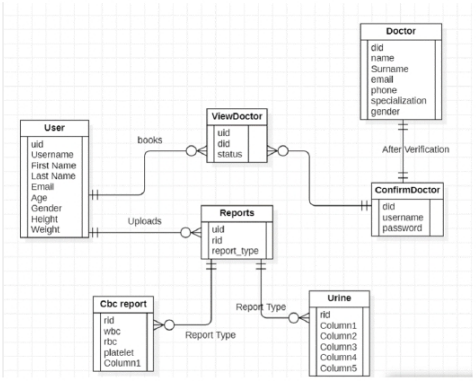


Fig 2- Database Schema

Dashboard Page pops up and it contains the extracted data from the Files uploaded by the Patient. The dashboard shows the extracted values with color-coding. Red Color for any values that are above the normal range, Green if values are between the normal range, and Yellow if values are at the boundaries of the normal range. The Dashboard Page also has a table with four columns which are the Metrics, the Normal Range of Metrics, the Extracted Data of Each Metric (Report Value) and Remarks, where the Patient can see information about home Remedies and other notifications. The patient can see the Doctors' Comments on the report that they uploaded. To book aforementioned doctors, the Patient can go to the View Doctor page in which all the doctors available for the patient are visible. Patients can see all the Doctors with their names, Specialization, and Gender and can book the Doctor according to their requirements. The patient Profile Page gives an overview of the Patient information which the Patient has registered during registration. On this Page the If Patient Wants to change any information then they can click on the Edit Profile button and can go to "EditProfile" Page. EditProfile Page helps the Patient to change their personal information and save it to the database.

A new doctor will first register on the Doctor Registration page. After registration they will be confirmed by the admin. There will be verification of doctor details, the doctor will be able to login with the given username and password. After logging in the doctor can see the list of patients who have sent him a request. The doctor can further accept or reject the request. Viewing Patient Reports – the doctor can see all patient reports on this page. At last the doctor can view the dashboard for each report and comment if those comments needed, will be reflected on the patient dashboard.

User Table contains general information about patients with UID as primary key of the table and it is referenced as foreign key to Reports Table and View Doctor Table. The Reports table has the primary key as Report ID and information about report type and it is referenced as Foreign key to CBC Report and Urine Report table. CBC, Urine Reports are two types of report which our Web App is able to extract using Text mining and OCR Techniques. Doctor table where Doctor ID is Primary key and general information about Doctor is stored in

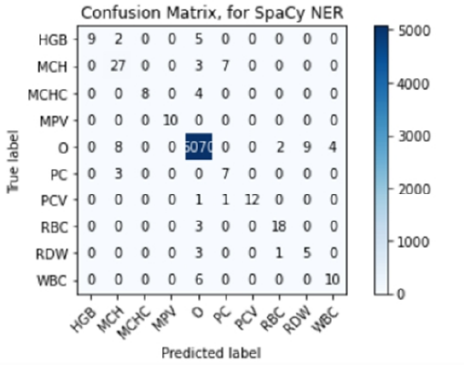


Fig 3- Confusion Matrix

the table. After Verification we have Confirm Doctor where Doctor ID is Foreign key referred from Doctor table. In the View Doctor We have User ID and Doctor ID as foreign keys referring to the user and Confirm Doctor table.

V. RESULTS AND DISCUSSION

The word error rate of Pytesseract on our dataset was 0.1672081326 and nanonets gave a WER of 0.04656046305. The cloud based OCR gave better outputs for most images, but for text mining, it is crucial that the values of parameters is after the parameters. Nanonets sometimes parsed the values before the named entity parameters, which meant that the text mining algorithm couldn't detect the value of the named entity. In those cases Pytesseract performs better. Hence for the webapp, both OCRs will be used in tandem, first nanonets, and if it is unable to detect the parameters, then Pytesseract.

The Named Entity recognition model has given the following results mentioned in Fig. 3.

And the NER gave an accuracy of 0.98.

The web app has been compiled and hosted. For future work we want to add compatibility with more types of reports.

VI. CONCLUSION

We have successfully implemented the modules of Data Recognition, Data Extraction, Data Storage, and Visualization. Apart from this, we have implemented the Doctor's module

too. We have built text recognition from Images and PDFs using Optical Character Recognition and PDF Plumber respectively. We created a dataset of medical reports for OCR and extracted the text to create a text corpus for text mining machine learning algorithm which utilizes Named Entity Recognition (NER), combined with Regex to extract the parameters of CBCs. The SQLite database has been integrated to store data in an encrypted format. The data is visualized and interpreted to be displayed on the dashboard. The doctor's module includes choosing patients and accessing their reports alongside adding comments for the patient.

VII. References

[1] An OCR Post-correction Approach using Deep Learning for Processing Medical Reports Srinidhi Karthikeyan1 2, Alba G. Seco de Herrera1, Faiyaz Doctor1 and Asim Mirza 2 1School of Computer Science and Electronic Engineering, University of Essex, Colchester, CO4 3SQ UK 2Firza Group, London E10 5FA.

[2] Building Structured Personal Health Records from Photographs of Printed Medical Records Xiang Li, PhD1, Gang Hu, MS1, Xiaofei Teng, PhD1, Guotong Xie, MS11IBM Research, Beijing, China.

[3] Hegghammer, T. OCR with Tesseract, Amazon Textract, and Google Document AI: a benchmarking experiment. J Comput Soc Sc (2021). https://doi.org/10.1007/s42001-021-00149-1

[4] B. Fatiha, B. Bouziane and A. Baghdad, "MedIX: A named entity extraction tool from patient clinical reports," 2011 International Conference on Communications, Computing and Control Applications (CCCA), 2011, pp. 1-6, doi: 10.1109/CCCA.2011.6031494.

[5] Prashant M Ahire, Anil P Gagare, Yogesh B Pawar, Savan S Vidhate, "EXTRACT AND ANALYSIS OF SEMI STRUCTURED DATA FROM WEBSITES AND DOCUMENTS", www.ijcsmc.com, Vol 4, Issue 2

[6] P. J. Rajkovic and D. S. Jankovic, "Custom made medical data reporting tool," 2009 9th International Conference on Telecommunication in Modern Satellite, Cable, and Broadcasting Services, 2009, pp. 306-309, doi: 10.1109/TELSKS.2009.5339518.

[7] Reiner B. I. (2010). Customization of medical report data. Journal of digital imaging, 23(4), 363–373. https://doi.org/10.1007/s10278-010-9307-4

[8] Liu, Nan & Wang, Chuan & Miao, Xinyu & Bai, Hua & Wang, Yunan & Yang, Limin & Lei, Yiming & Zhang, Wei & Wang, Hong. (2020). A New Data Visualization and Digitization Method for Building Electronic Health Record. 2980-2982. 10.1109/BIBM49941.2020.9313116.

[9] "Impact of Cloud Database in Medical Healthcare Records based on Secure Access." International Journal of Engineering and Advanced Technology (2019): n. pag.

Acknowledgements

We take this opportunity to express our profound gratitude and deep regards to our guide Prof. Nandana Prabhu for her exemplary guidance,monitoring and constant encouragement throughout the course of this project.
We are obliged to the Project Committee and all the Staff Members of K. J. Somaiya College of Engineering, Somaiya Vidyavihar University for the valuable information provided by them in their respective fields.We are grateful for their cooperation during the period of our project.