

HEALTH-CARD
FOR MANAGING HEALTH DATA
ONLINE
By
Neel Dani (ID No. 17CEUON039)
Smit Panchal (ID No. 17CEUBS054)
Smit Hapani (ID No. 17CEUOS005)
A project submitted
In
Partial fulfilment of the
requirements for the degree of
BACHELOR OF TECHNOLOGY
In
Computer Engineering

Internal Guide

Prof. Ami M. Shah
Assistant Professor
Dept. Of Comp. Engg.

External Guide

Mr. Rashmin Chhatrala
Team Lead
RapidOps Solutions Pvt.Ltd.



Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University
April 2021

CERTIFICATE

This is to certify that the project work titled

HEALTH-CARD

is the bonafide work of

Neel Dani (ID No. 17CEUON039)

Smit Panchal (ID No. 17CEUBS054)

Smit Hapani (ID No. 17CEUOS005)

Carried out in the partial fulfilment of the degree of Bachelor of Technology in
Computer Engineering at Dharmsinh Desai University in the academic session

December 2020 to April 2021

Prof. Ami M. Shah
Asst. Prof.
Dept. of Computer Engg.

Dr. C. K. Bhensdadia
Head,
Dept. of Computer Engg.



**Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University
April 2021**

COMPANY CERTIFICATE



Date: 9th April '21

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Neel Dani** has completed internship with **Rapidops Solutions Pvt. Ltd.**
From **4th January, 2021 to 9th April, 2021.**

Following is the details of his project and technology:

Project Name: HealthCard

Technology: MEAN Stack

During the internship we found him punctual, hardworking and inquisitive.

Sincerely,



Authorized Signatory

Chitra Kiri

HR Manager

Rapidops Solutions Private Limited

Web & Mobile Development : UI/UX Design : Cloud Services : CRM/ERP Solution : E-commerce
303-309, City Center, Opp. Shukan Mall, Science City Road, Sola, Ahmedabad 380060, India.
web: www.rapidops.com | email: hello@rapidops.com | phone: +91-70434 70009



Date: 9th April '21

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Smit Panchal** has completed internship with **Rapidops Solutions Pvt. Ltd.**
From **4th January, 2021 to 9th April, 2021.**

Following is the details of his project and technology:

Project Name: HealthCard

Technology: MEAN Stack

During the internship we found him punctual, hardworking and inquisitive.

Sincerely,


Authorized Signatory

Chitra Kiri

HR Manager

Rapidops Solutions Private Limited

Web & Mobile Development / UI/UX Design / Cloud Services / CRM/ERP Solution / E-commerce
303-309, City Centre, Opp. Shukan Mall, Science City Road, Sola, Ahmedabad-380060, India
web: www.rapidops.com / email: hello@rapidops.com / phone: +91 70434 70009



Date: 9th April '21

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Smit Hapani** has completed internship with **Rapidops Solutions Pvt. Ltd.**
From **4th January, 2021 to 9th April, 2021.**

Following is the details of his project and technology:

Project Name: HealthCard

Technology: MEAN Stack

During the internship we found him punctual, hardworking and inquisitive.

Sincerely,

A circular stamp with the text "RAPIDOPS SOLUTIONS PVT. LTD." around the perimeter. Inside the stamp is a handwritten signature in blue ink.

Authorized Signatory

Chitra Kiri

HR Manager

Rapidops Solutions Private Limited

Web & Mobile Development | UI/UX Design | Cloud Services | CRM/ERP Solution | E-commerce
303-309, City Center, Opp. Shukan Mall, Science City Road, Sola, Ahmedabad-380060, India.
web: www.rapidops.com | email: hello@rapidops.com | phone: +91-70434 70009

ACKNOWLEDGEMENT

We feel profoundly grateful while presenting the project assignment. The nature of project on the development of “Health-Card” has given us wide opportunity to think, implement and interact with various aspects of management skills as well as the new emerging facilities and the technology used in architecture and the enhancements given to the students with a boon of spirituality and curricular activities.

Every work that one completes successfully stands on the constant encouragement, goodwill and support of the people around. We hereby avail this opportunity to express our gratitude to all the supporters who extended their valuable time, support, energy and cooperation in completing the assigned task.

We feel really obliged while to expressing our gratitude to Prof. Ami M. Shah, Mr. Rashmin Chhatrala for their support and encouragement.

We express deep sense of gratitude towards our project guide Mr. Rashmin Chhatrala towards their innovative suggestions and efforts to make project a success. It is their sincerity that prompted us throughout the project to do hard work using the industry adopted technologies.

We are sincerely thankful to the Head of Computer Engineering Department, Dr. C.K. Bhensdadia for the unconditional and an unbiased support during the whole session of study and development.

Regards,

Neel Dani

Smit Panchal

Smit Hapani

Health-Card

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF TABLES	iv
1. Introduction	2
1.1 Project Details	2
1.2 Purpose	2
1.3 Scope	2
1.4 Objectives	2
1.5 Technology/Platform/Tools used	3
2. About the system	5
2.1 System Requirements	5
2.1.1 Hardware Requirements	5
2.1.2 Software Requirements	5
2.2 Software Development Strategy	5
2.3 Specific Requirements	7
Software Requirements Specification (SRS)	8
3. Analysis	13
3.1 Entity-Relationship Diagram	13
3.2 UML Diagrams	14
3.2.1 Use-Case Diagram	14
3.2.2 Sequence Diagrams	15
3.2.3 Class Diagram	16
3.2.4 Activity Diagram	17
4. Design	19
4.1 Database	19
4.2 Front-end Interface	22
4.3 Validations	32
4.4 Application Navigation	37
5. Implementation	40
5.1 Implementation Environment	40
5.2 Coding Standards	41

5.3 Implementation of Modules	42
5.3.1 Front-end	42
5.3.2 Back-end	44
6. Test Case Design	47
6.1 Testing Plan	47
6.2 Testing Strategy	47
6.3 Test Cases	48
7. Conclusion and Future Extensions	51
7.1 Future Extensions	51
7.2 Conclusion	51
Bibliography	52

ABSTRACT

We have Aadhar card, Pan Card, Voter id, and Driving licence like identity proofs for our day to day transactions, but we have major missing that is our Health card.

Managing of all health related data like different prescriptions, reports for long time is extremely tedious task. Health-Card helps to manage our day to day health history and same card is used everywhere in private and public hospitals and clinics, medical store, laboratory to track your health-related data.

Health-Card System provides information of the Patient's all medical related data, and lets the government use that data and get all statistical analysis of health.

LIST OF FIGURES

Figure	Page No.
Fig 2.1: Incremental model of software development.	6
Fig 3.1: Entity-Relationship Diagram for Health-Card system	13
Fig 3.2: Use-Case Diagram for Health-Card system	14
Fig 3.3: Sequence Diagram for Upload patient's report	15
Fig 3.4: Class Diagram for Health-Card system	16
Fig 3.5: Activity Diagram for Add Prescription	17
Fig 4.1: Home page	22
Fig 4.2: About us	22
Fig 4.3: Login Page	23
Fig 4.4: Signup page	23
Fig 4.5: Doctor's additional details for registration	24
Fig 4.6: Laboratorian additional details for registration	24
Fig 4.7: Patient's home page	25
Fig 4.8: Downloaded health card pdf file	25
Fig 4.9: Patient's view prescriptions details page	26
Fig 4.10: Patient's profile	26
Fig 4.11: Laboratorian side add report page	27
Fig 4.12: Laboratorian side uploaded report page	27
Fig 4.13: Search report by table fields	28
Fig 4.14: Doctor side add diagnosis page	28
Fig 4.15: Add prescription	29
Fig 4.16: Doctor's profile basic details	29
Fig 4.17: Doctor's profile with additional information	30
Fig 4.18: View prescription with search functionality	30
Fig 4.19: View prescription of the selected patient	31
Fig 4.20: Detailed information of the selected patient's prescription	31
Fig 4.21: Downloaded report	32

Fig 4.22: Login fields required validation	33
Fig 4.23: Check credentials valid or not	34
Fig 4.24: Signup fields required validation	34
Fig 4.25: Password and confirm password match validation	35
Fig 4.26: Check email already exists or not	35
Fig 4.27: Fields required	36
Fig 4.28: Check patient-id is valid or not	36
Fig 4.29: Doctor module navigation chart	37
Fig 4.30: Laboratorian module navigation chart	37
Fig 4.31: Patient module navigation chart	38

LIST OF TABLES

Table	Page No.
Table 4.1 – Data Dictionary for Doctor	19
Table 4.2 – Data Dictionary for Prescription	19
Table 4.3 – Data Dictionary for User	20
Table 4.4 – Data Dictionary for Prescription Detail	20
Table 4.5 – Data Dictionary for Laboratorian	21
Table 4.6 – Data Dictionary for Report	21
Table 6.1 – Test Cases	49

Chapter 1

Introduction

1. Introduction

1.1 Project Details

Managing health information using information technology is an important part of the changing health-care system. The aim of this proposed Health-Card system is to improve efficiency, access and Accountability of health-care services. Using the Health-Card all patient's data, doctor's prescription, patient's present and previous health history could be accessible through an easy, responsive web interface.

1.2 Purpose

The purpose of the Health card is to provide an easy, responsive web interface for the patient who want to store health related data online.

1.3 Scope

Health-Card System provides information of the Patient all medical related data, and lets the government use that data and get all statistical analysis of patient health data area wise.

1.4 Objectives

- Main objective of the system is to provide easy interface to user to store their all health related data online instead of managing all the data on papers and it also Eliminates need to remember specific terms, medications, and treatment names.
- Improved communication and information sharing among healthcare providers, Doctors, hospitals, laboratories and pharmacies.

1.5 Technology/Platform/Tools used

Front-End:

- HTML 5
- CSS 3
- Bootstrap 4
- Angular 8

Back-End:

- Node.js
- Express.js

Database:

- Mongo DB

Tools:

- VS Code
- Mongo DB Compass
- Postman

Chapter 2

About the system

2. About the system

2.1 System Requirements

It is platform independent and can be run on Linux, Windows or Mac. However, there are few requirements that need to be matched for different purposes like performance and durability.

2.1.1 Hardware Requirements

Server Side

- GB RAM recommended for good performance.
- Internet Connectivity

Client Side

- Internet Connection

2.1.2 Software Requirements

Server Side

- Node.js
- Angular CLI
- Mongo DB

Client Side

- Web Browser

2.2 Software Development Strategy

Modules developed for Health-Card have been built using the Incremental model of software development.

This approach is based on the idea of rapidly developing an initial software implementation from very abstract specifications and modifying this according to your appraisal. Each program version inherits the best features from earlier versions. Each version is refined based upon feedback from yourself to produce a system which satisfies your needs.

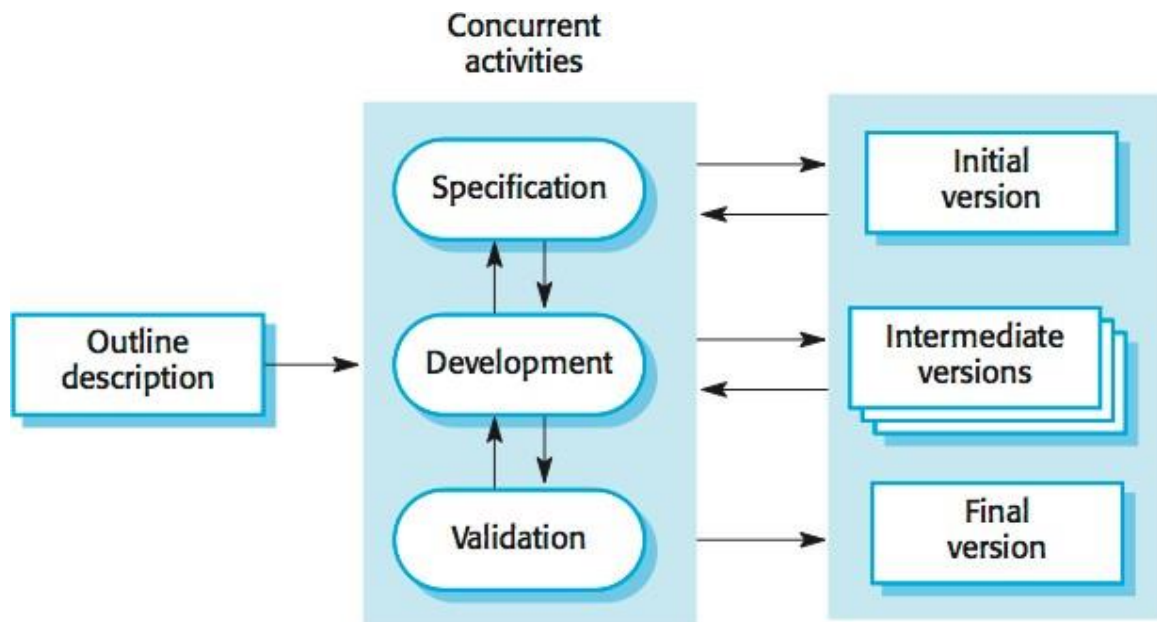


Figure 2.1: Incremental model of software development.

Advantages:

- This is the only method appropriate for situations where a detailed system specification is unavailable.
- Effective in rapidly producing small systems and developing sub-components of larger system.

Disadvantages:

- It is difficult to measure progress and produce documentation reflecting every version of the system as it evolves. This paradigm usually results in badly structured programs due to continual code modification. Production of good quality software using this method requires highly skilled and motivated programmers.

2.3 Specific Requirements

After the extensive analysis of the problems in the system, we are familiarized with the requirements that the current system needs. The requirements that the system needs is categorized into functional and non-functional requirements. These requirements are listed below.

Functional Requirements:

Functional requirements describe how a product must behave, what its features and functions.

- User should be able to login/signup as the doctor or patient or laboratorian to the system
- Doctor should be able to add prescriptions to the respective patient
- Patient should be able to view the prescriptions added by the doctor
- Patient should be able to download his health-card
- The system must allow doctor and patient to download the report which is uploaded by the laboratorian
- Laboratorian should be able to upload the report mentioned by the doctor for a particular prescription
- Doctor and patient should be able to view his profile

Non-Functional Requirements:**Performance:**

The system must be interactive and the delays involved must be less. So, in every action-response of the system, there are no immediate delays. In case of opening App components, popping error messages and the settings or sessions there is delay much below 3 seconds.

Security:

User's sensitive details should be secured to the server. The main security concern is for user account hence proper login mechanism should be used to avoid being hacked.

Reliability:

As the system provide the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

Maintainability:

The newly developed modules do not affect the existing modules. Bugs if found in these modules can be dealt with separately without affecting the other Modules. The modules being even internally modular and with easily understandable design (including proper comments and proper coding standards as per the company guidelines), functionality addition to these modules in future will be much easier.

Software Requirements Specification (SRS)

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform.

There are three modules Doctor, Patient and Laboratorian in the system and software requirements are as per the modules specified.

R.1 Doctor:**R.1.1 Registration**

Description: If doctor wants to register to the Health-Card system then he can register by giving valid proofs like licence number, email, licence expiration date etc.

Input: User details

Output: Sign in page

R.1.2 Login

Description: Doctor can login to the system by giving valid email-id and password.

Input: User credentials

Output: Home page

R.1.3 View Health History

Description: Doctor can view previously added prescriptions of the patient.

Input: User Selection

Output: Displays Selected Patient's Health History

R.1.4 Add Prescription

Description: Doctor can add new diagnosis and prescription of the patient.

Input: Prescription and Diagnosis details

Output: Acknowledgement of whether prescription added or not

R.1.5 Download Lab Report

Description: Doctor can download the lab report of the patient which is uploaded by laboratorian.

Input: User Selection

Output: File Downloaded

R.2 Laboratorian

R.2.1 Registration

Description: If Laboratorian wants to Register to the Health-Card system then he can register by giving valid proofs like licence number, lab-name, licence expiration date etc.

Input: User details

Output: Sign in page

R.2.2 Login

Description: Laboratorian can login to the system by entering valid email id and password.

Input: User credentials

Output: Home page

R.2.3 Add Lab Report

Description: Laboratorian can add lab reports of respective patient.

Input: Report Selection

Output: Acknowledgement of whether report is added or not

R.2.4 View Uploaded Report

Description: Laboratorian can view the uploaded reports.

Input: User Selection

Output: List of uploaded reports

R.2.5 Search Report

Description: Laboratorian can search reports by patient name or date.

Input: Search input

Output: Search related data

R.3 Patient:

R.3.1 Registration

Description: Patient can register himself by giving valid data.

Input: User details

Output: Sign in page

R.3.2 Login

Description: Patient can login to the system by entering valid email id and password.

Input: User credentials

Output: Home page

R.3.3 View and download Health-card

Description: Patient can view as well as download his Health-card.

Input: User command

Output: downloaded health card in pdf format

R.3.4 View Prescriptions

Description: Patient can view prescriptions added by the doctor.

Input: User selection

Output: Display selected prescription

R.3.5 Download Lab Report

Description: Patient can download the lab report which is uploaded by laboratorian.

Input: User command

Output: File Downloaded

Chapter 3

Analysis

3. Analysis

3.1 Entity-Relationship Diagram

ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

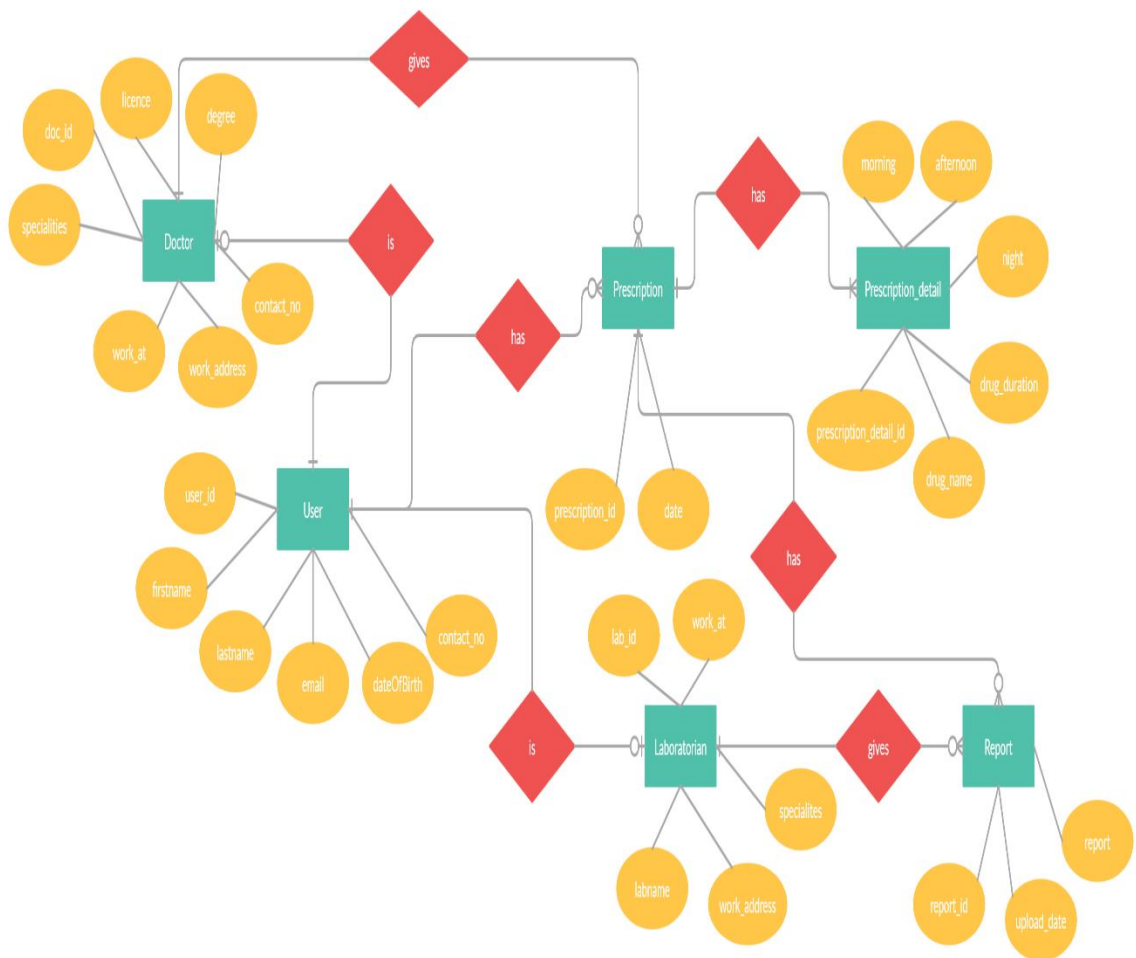


Figure 3.1: Entity-Relationship Diagram for Health-Card system.

3.2 UML Diagrams

3.2.1 Use-Case Diagram

A use-case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and different use cases in which the user is involved. A use case diagram can identify the different type of users of a system and the different use cases and will often be accompanied by other type of diagram as well.

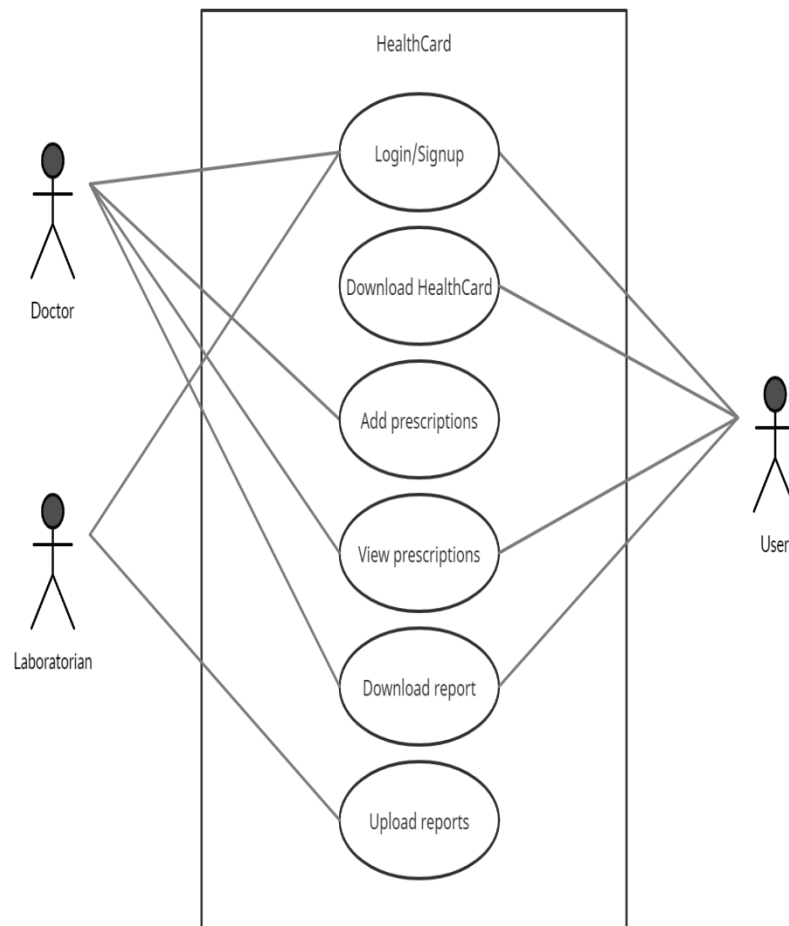


Figure 3.2: Use-Case Diagram for Health-Card system.

3.2.2 Sequence Diagrams

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence.

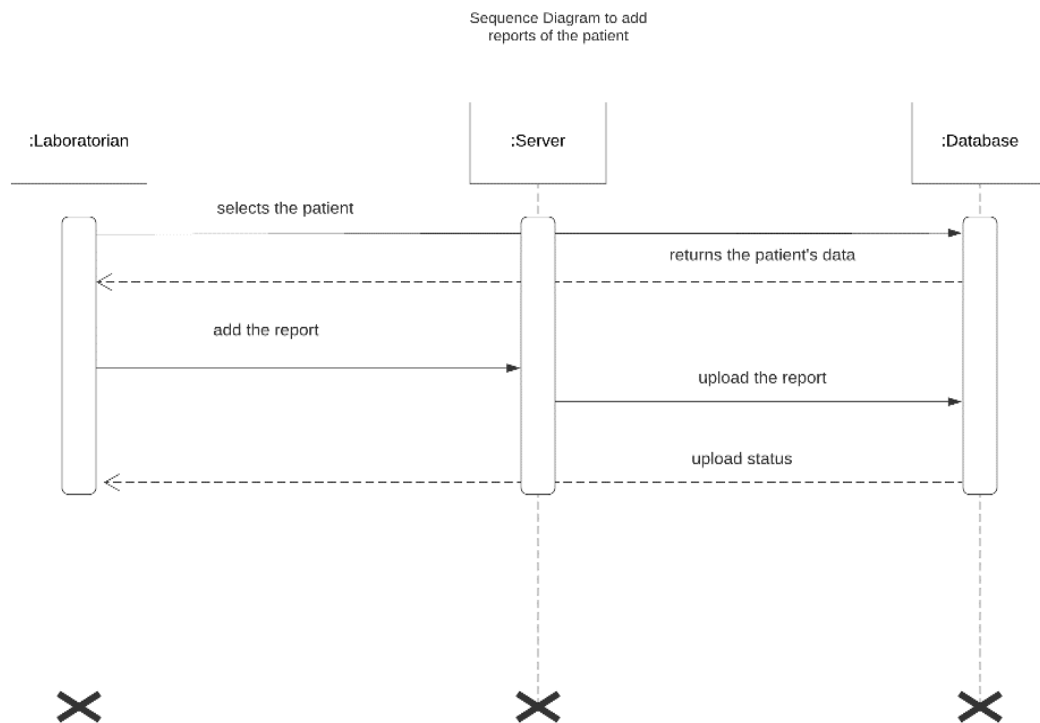


Figure 3.3: Sequence Diagram for Upload patient's report

3.2.3 Class Diagram

The class diagram is a graphical notation used to construct and visualize object oriented systems. A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's:

- classes,
- their attributes,
- operations (or methods),
- the relationships among objects.

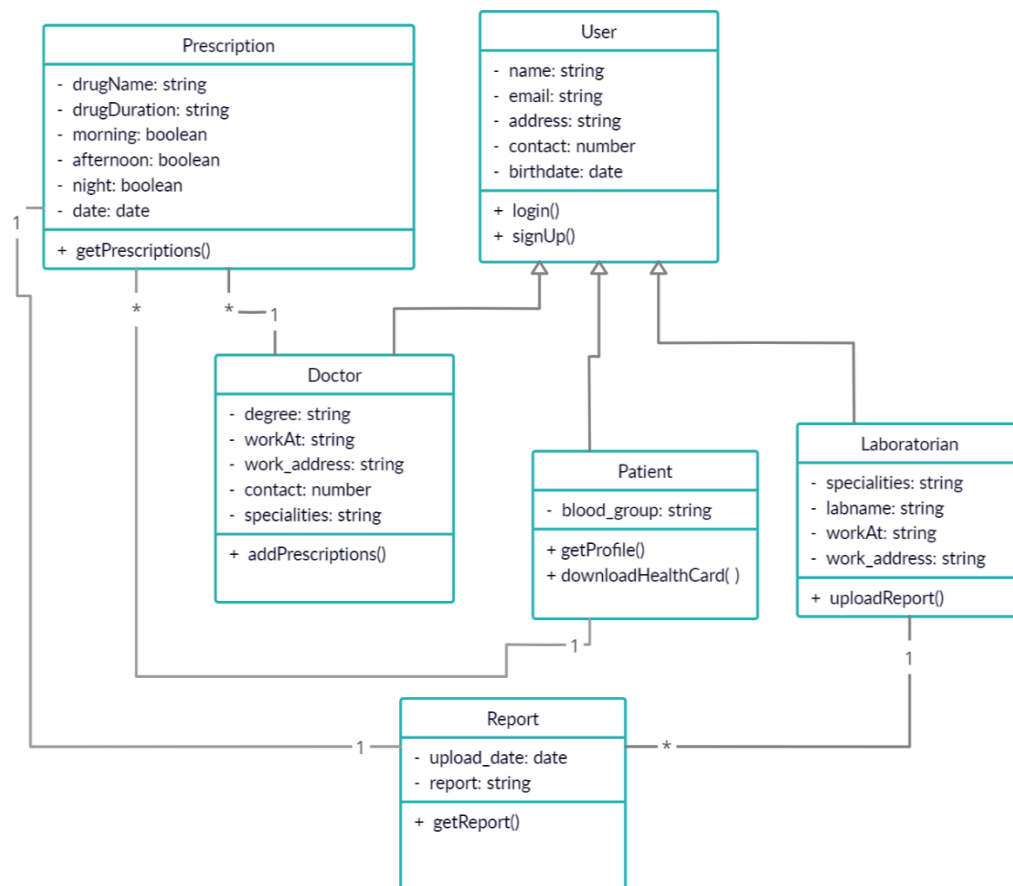


Figure 3.4: Class Diagram for Health-Card system.

3.2.4 Activity Diagram

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

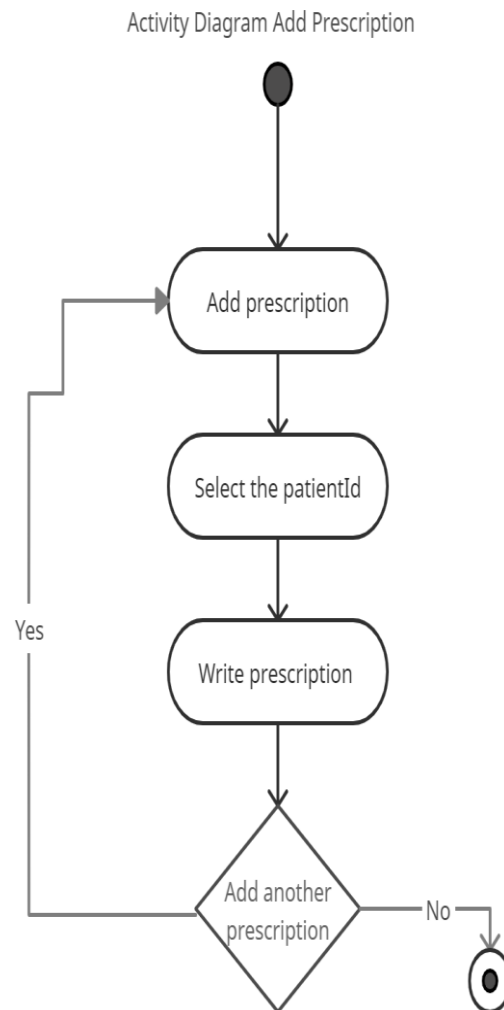


Figure 3.5: Activity Diagram for Add Prescription.

Chapter 4

Design

4. Design

4.1 Database

Data Dictionary:

A data dictionary contains metadata (Data about the database). The data dictionary is very important as it contains information such as what is in the database, who is allowed to access it, where is the database physically stored etc. The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators.

Doctor				
Field Name	Data Type	PK/FK	Referenced Table	Description
doctor_id	Number	PK	-	Unique Id for doctor
user_id	Number	FK	User	-
licence	Varchar2	-	-	-
degree	Varchar2	-	-	-
workat	Varchar2	-	-	-
work_address	Varchar2	-	-	-
contact_no	Number	-	-	-
specialities	Varchar2	-	-	-

Table 4.1 – Data Dictionary for Doctor

Prescription				
Field Name	Data Type	PK/FK	Referenced Table	Description
prescription_id	Number	PK	-	Unique Id for prescription
user_id	Number	FK	User	-
doctor_id	Number	FK	Doctor	-
date	Date	-	-	-

Table 4.2 – Data Dictionary for Prescription

User				
Field Name	Data Type	PK/FK	Referenced Table	Description
user_id	Number	PK	-	Unique Id for user
firstname	Varchar2	-	-	-
lastname	Varchar2	-	-	-
email	Varchar2	-	-	-
address	Varchar2	-	-	-
contact_no	Number	-	-	-
dateOfBirth	Date	-	-	-

Table 4.3 – Data Dictionary for User

Prescription_detail				
Field Name	Data Type	PK/FK	Referenced Table	Description
prescription_detail_id	Number	PK	-	Unique Id for prescription details
prescription_id	Number	FK	Prescription	-
drug_name	Varchar2	-	-	-
drug_duration	Number	-	-	-
morning	Boolean	-	-	-
afternoon	Boolean	-	-	-
night	Boolean	-	-	-
after_before_meal	Varchar2	-	-	-

Table 4.4 – Data Dictionary for Prescription Detail

Laboratorian				
Field Name	Data Type	PK/FK	Referenced Table	Description
lab_id	Number	PK	-	Unique Id for laboratorian
user_id	Number	FK	User	-
workAt	Varchar2	-	-	-
specialities	Varchar2	-	-	-
labname	Varchar2	-	-	-
work_address	Varchar2	-	-	-

Table 4.5 – Data Dictionary for Laboratorian

Report				
Field Name	Data Type	PK/FK	Referenced Table	Description
report_id	Number	PK	-	Unique Id for report
prescription_id	Number	FK	Prescription	-
lab_id	Number	FK	Laboratorian	-
upload_date	Date	-	-	-
report	Blob	-	-	-

Table 4.6 – Data Dictionary for Report

4.2 Front-end Interface

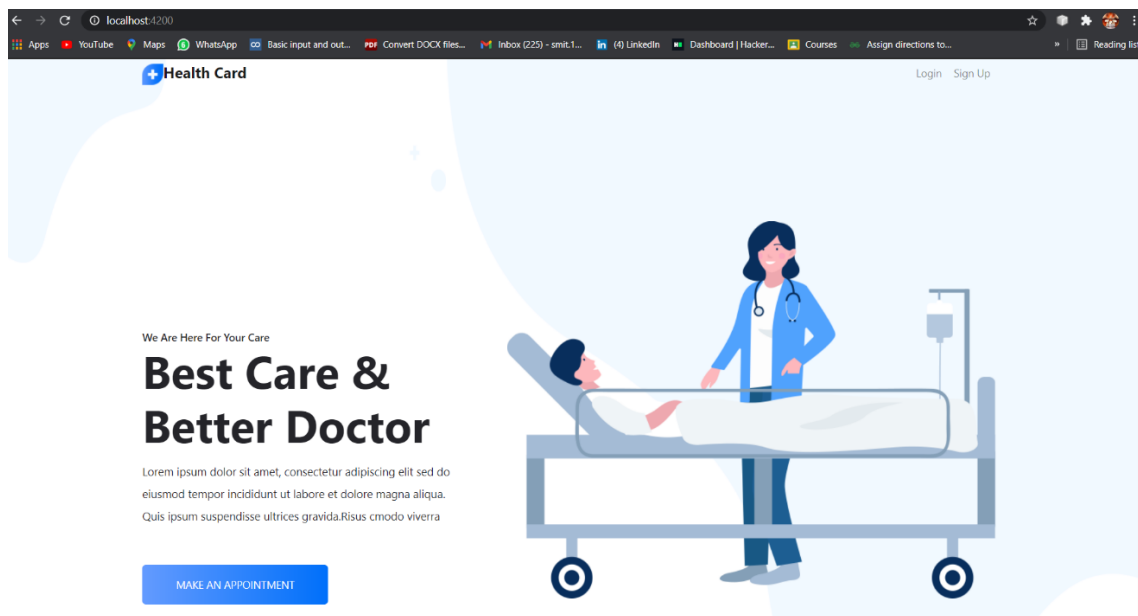


Figure 4.1: Home page

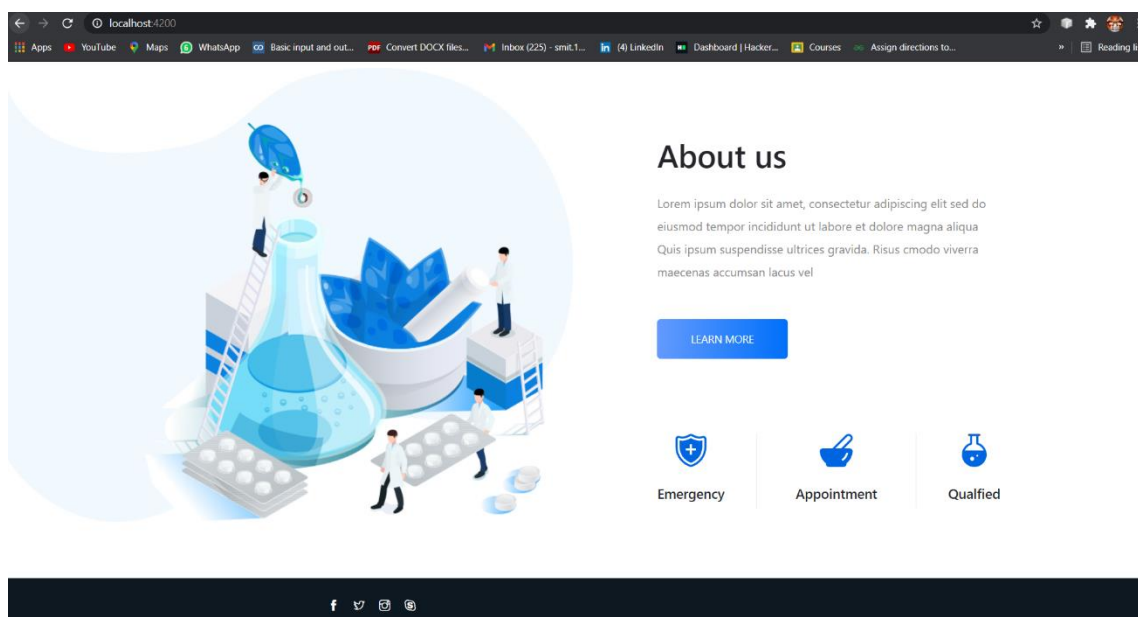
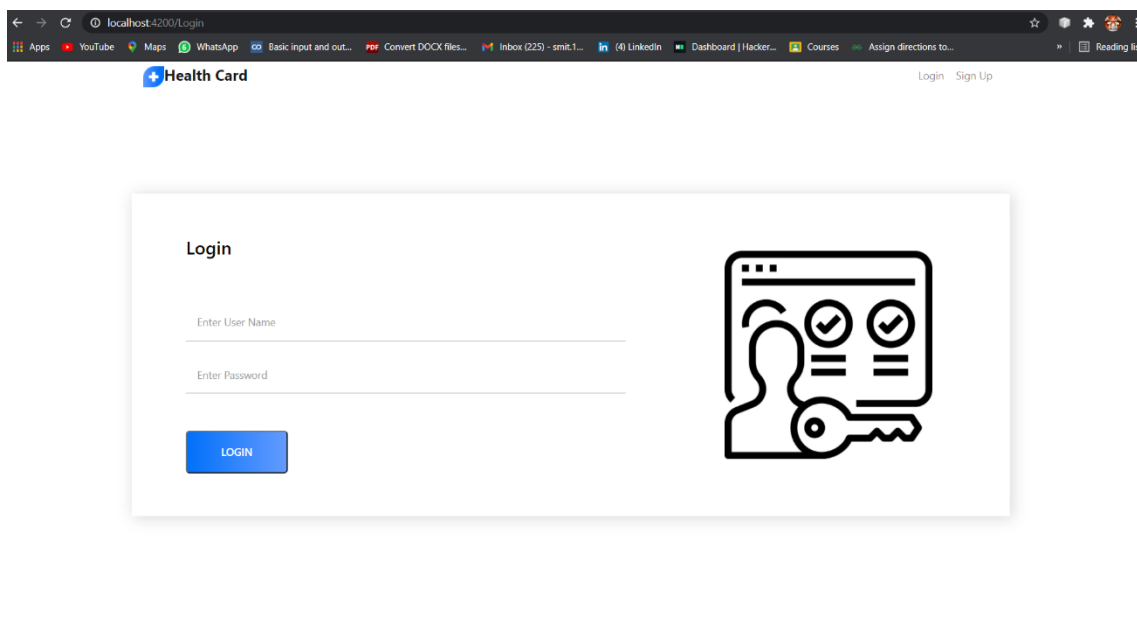
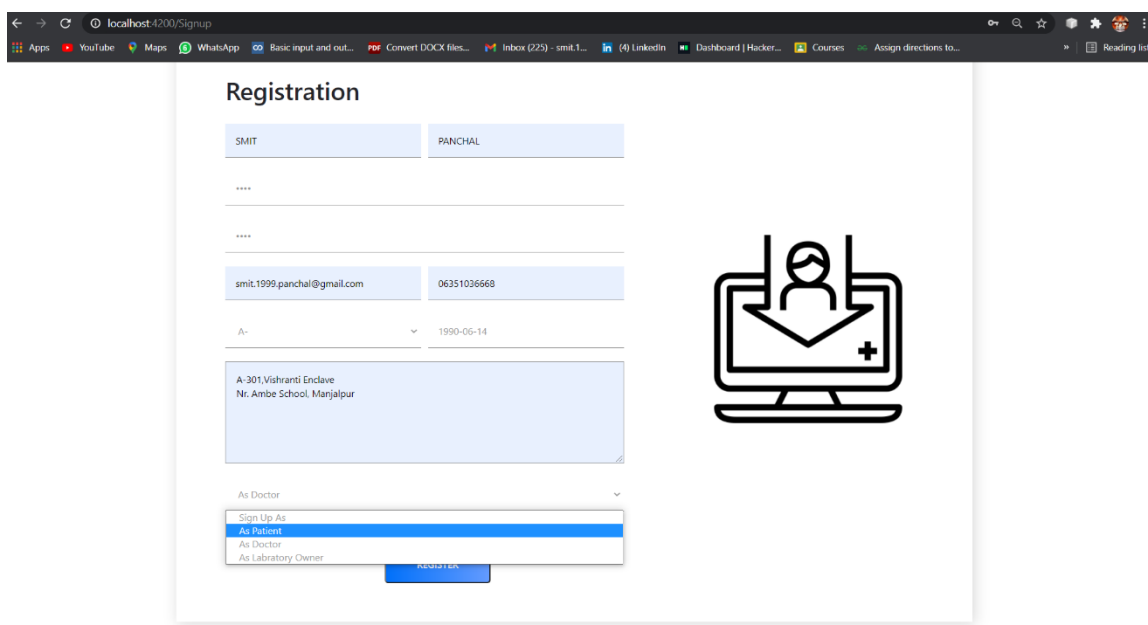


Figure 4.2: About us



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/Login'. The page title is 'Health Card'. In the top right corner, there are links for 'Login' and 'Sign Up'. The main content area features a 'Login' form with two input fields: 'Enter User Name' and 'Enter Password'. Below these fields is a blue 'LOGIN' button. To the right of the form is a large icon depicting a person's profile with a keyhole and a key, symbolizing access or security.

Figure 4.3: Login Page



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/Signup'. The page title is 'Registration'. The form is divided into two sections: 'SMIT' and 'PANCHAL'. The 'SMIT' section has two input fields for first and last names, followed by a dropdown menu for gender (currently showing 'A-') and a date of birth field (currently showing '1990-06-14'). The 'PANCHAL' section has two input fields for email address (currently showing 'smit.1999.panchal@gmail.com') and phone number (currently showing '06351036668'). Below these fields is a large text area for the address (currently showing 'A-301 Vishranti Enclave, Nr. Ambe School, Manjalpur'). At the bottom, there is a dropdown menu for 'Sign Up As' with options: 'As Patient' (selected), 'As Doctor', and 'As Laboratory Owner'. A blue 'REGISTER' button is located at the bottom right of the form. To the right of the form is a large icon depicting a person's profile with a plus sign, symbolizing registration or addition.

Figure 4.4: Signup page

Doctor Basic Information

Doctor Licence Number Degree/Major In

Doctor Specialities

New Speciality Remove

Work Place Information

Work Place Name Work Place Contact

Work Place Address

REGISTER

Figure 4.5: Doctor's additional details for registration

Lab Registration

Lab Name

Licence Number Licence Expiry Date

Lab Address

Full Blood Examination

REGISTER

Figure 4.6: Laboratorian additional details for registration

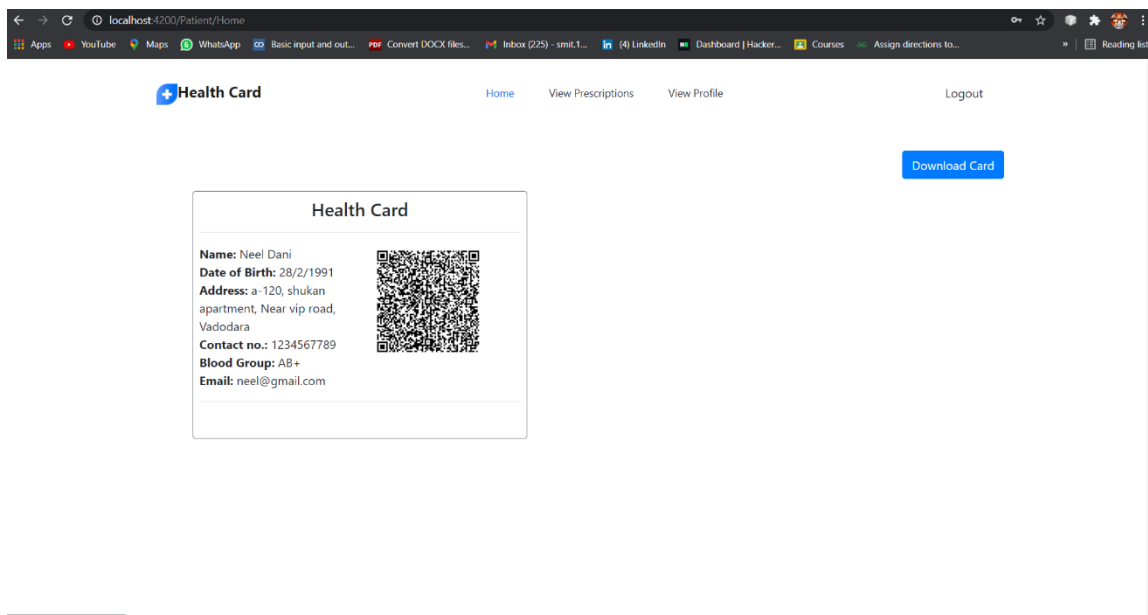


Figure 4.7: Patient's home page

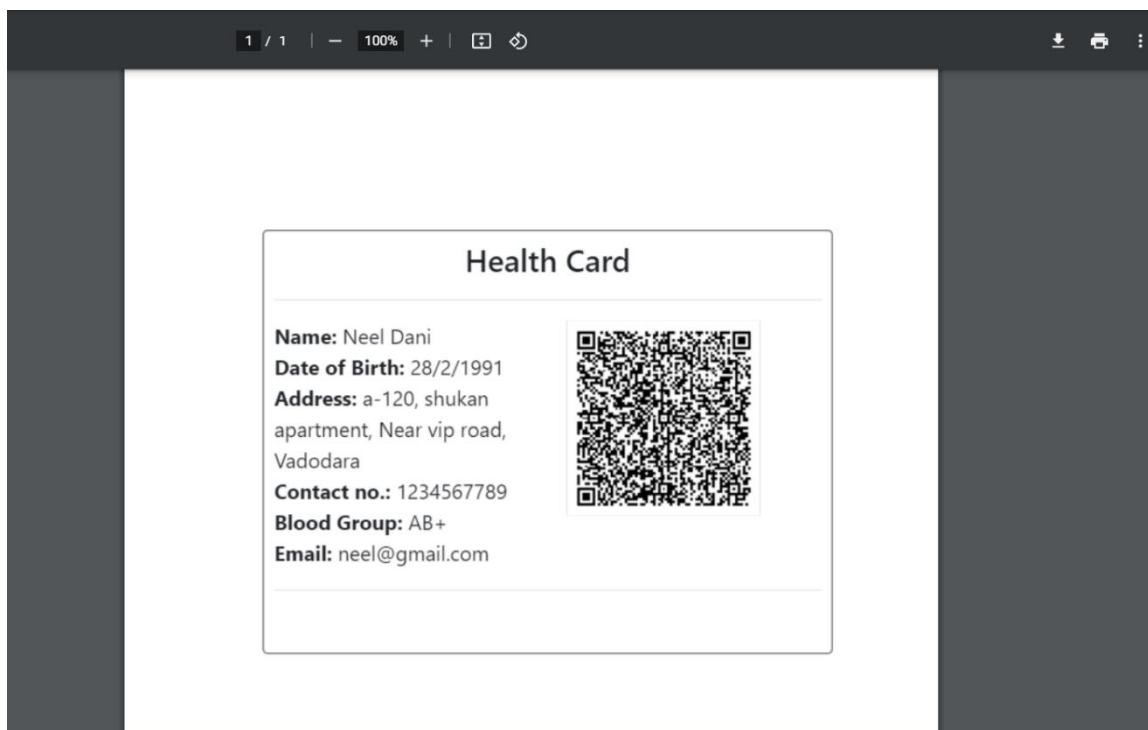


Figure 4.8: Downloaded health card pdf file

Health Card Home View Prescriptions View Profile Logout

Prescription List

Doctor	Doctor Id	Reports	Date	Actions
Dr.SMIT	DSMP00001406	1	August 4, 2021	Get Details
Dr.SMIT	DSMP00001406	1	August 4, 2021	Get Details

Prescription Details

Hospital	Date	time	Medicines	Doses												
Guru hospital	08/4/2021	14:47	<table border="1"> <thead> <tr> <th>Name</th> <th>Days</th> <th>description</th> </tr> </thead> <tbody> <tr> <td>dolo650</td> <td>12</td> <td>abc</td> </tr> </tbody> </table>	Name	Days	description	dolo650	12	abc	<table border="1"> <thead> <tr> <th>Morning</th> <th>Afternoon</th> <th>Night</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	Morning	Afternoon	Night	1	1	0
Name	Days	description														
dolo650	12	abc														
Morning	Afternoon	Night														
1	1	0														

Reports:
health_Corona.pdf [Download](#)

Figure 4.9: Patient's view prescriptions details page

Health Card Home View Prescriptions View Profile Logout

Patient Profile

User Id:	PNED00002802
Name:	Neel Dani
Email:	neel@gmail.com
Contact:	1234567789
Blood Group:	AB+
Date of Birth:	28/2/1991
Address:	a-120, shukan apartment, Near vip road, Vadodara

Figure 4.10: Patient's profile

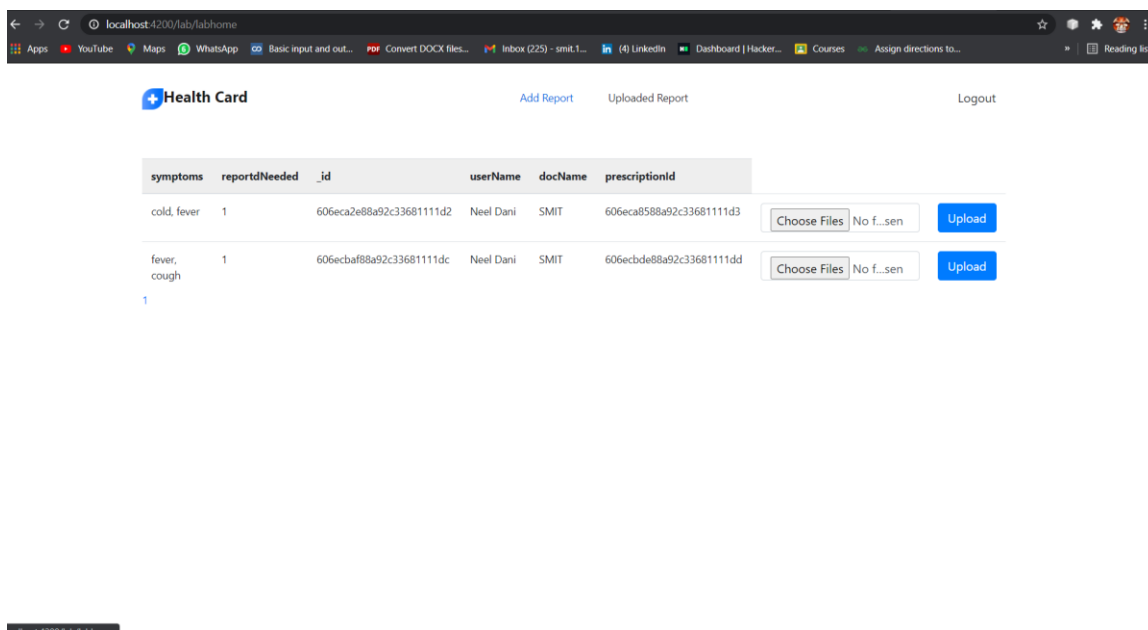


Figure 4.11: Laboratorian side add report page

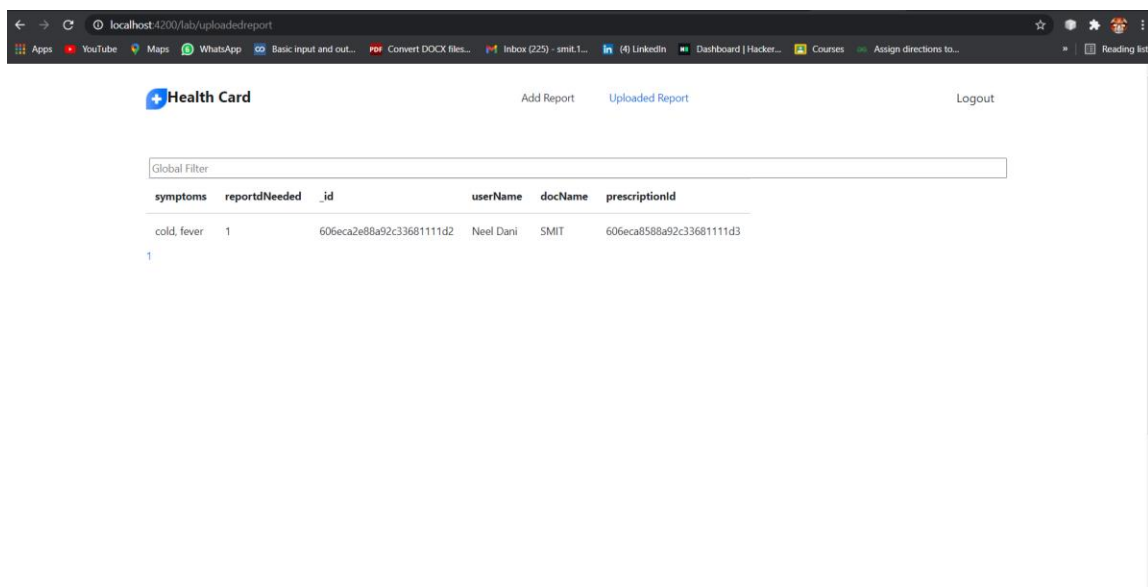


Figure 4.12: Laboratorian side uploaded report page

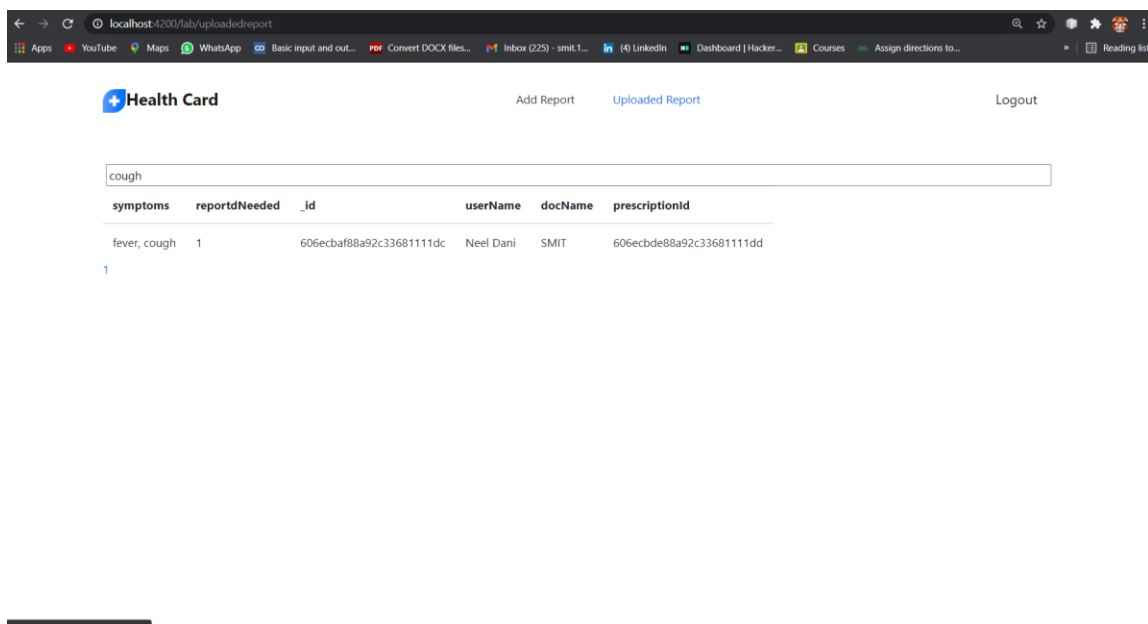


Figure 4.13: Search report by table fields

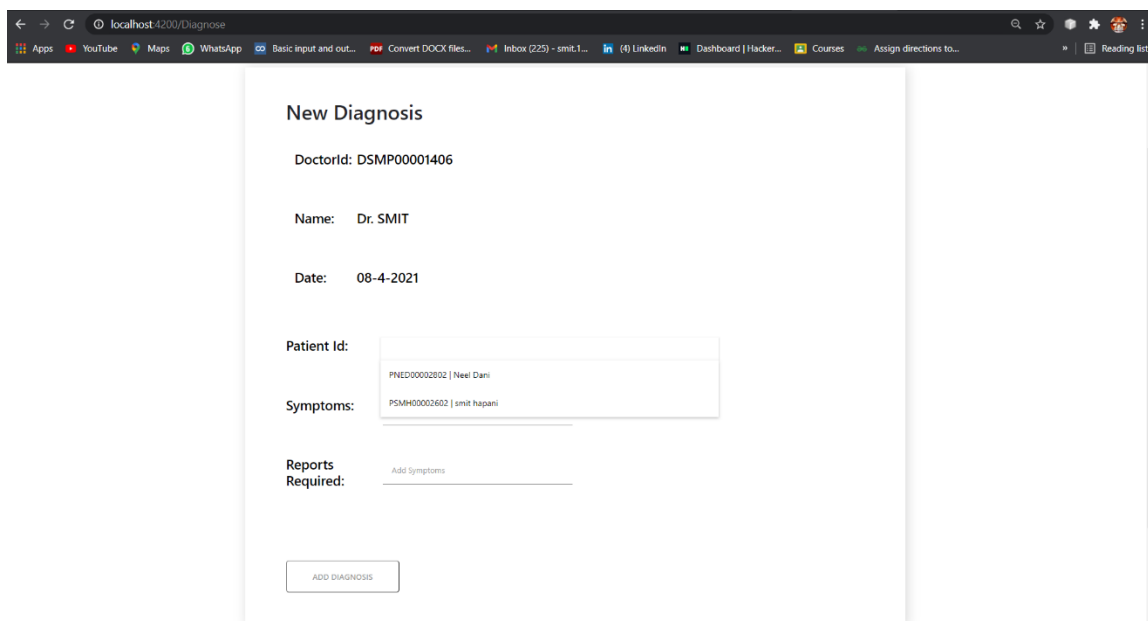


Figure 4.14: Doctor side add diagnosis page

The screenshot shows a web browser at localhost:4200/Prescription/Add. The page has a header with 'Health Card' and navigation links: 'New Diagnosis', 'View Profile', 'View Prescription', and 'Logout'. The main form is titled 'Add Prescription' and contains the following fields:

- Hospital:** A text input field with the placeholder 'Hospital name'.
- Date:** A date picker showing '08-04-2021' with a 'Today's Date' button.
- Time:** A time picker showing '14:47' with a 'Current Time' button.
- Medicine name:** A text input field with a '1' character and a 'If any caution' label.
- Frequency:** Radio buttons for 'Morning', 'Afternoon', and 'Night', followed by a 'Days' input field and a 'Remove' button.
- Submit:** An 'ADD PRESCRIPTION' button.

Figure 4.15: Add prescription

The screenshot shows a web browser at localhost:4200/Doctor/Profile/View. The page has a header with 'Health Card' and navigation links: 'New Diagnosis', 'View Profile', 'View Prescription', and 'Logout'. The main form is titled 'Basic Details' and contains the following information:

- User Id:** DSMP00001406
- Name:** SMIT PANCHAL
- Email:** smit.1999.panchal@gmail.com
- Contact:** 06351036668
- Blood Group:** A-
- Date of Birth:** 14/6/1990
- Address:** A-301,Vishranti Enclave Nr. Ambe School, Manjalpur

There is a large icon on the right side of the form, depicting a person sitting at a desk with a computer monitor, with a plus sign next to it.

Figure 4.16: Doctor's profile basic details

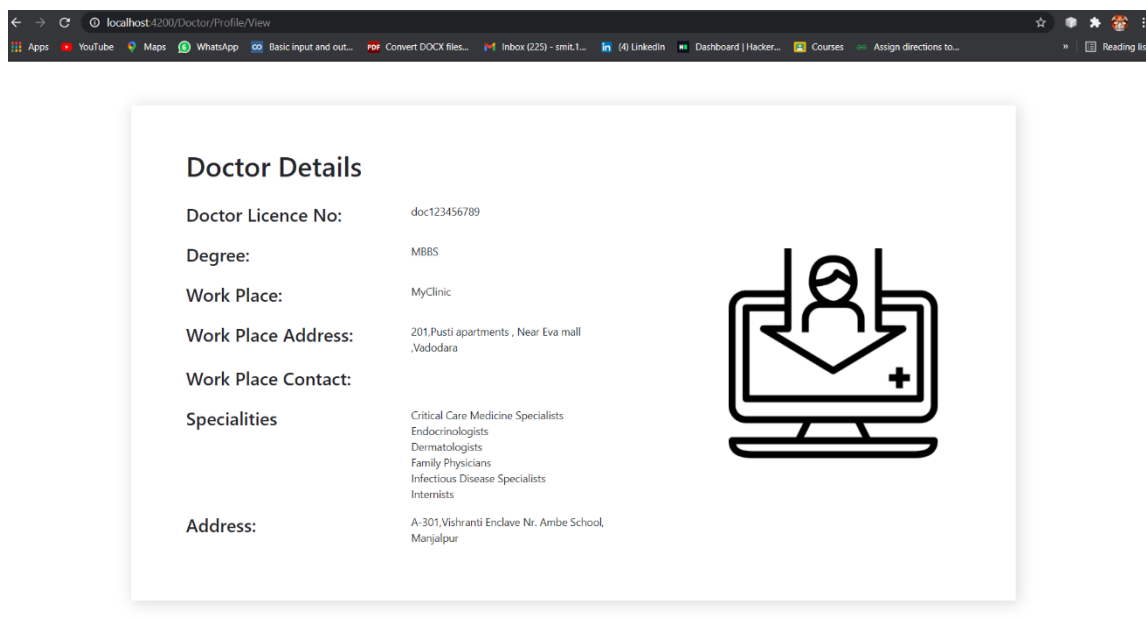


Figure 4.17: Doctor's profile with additional information

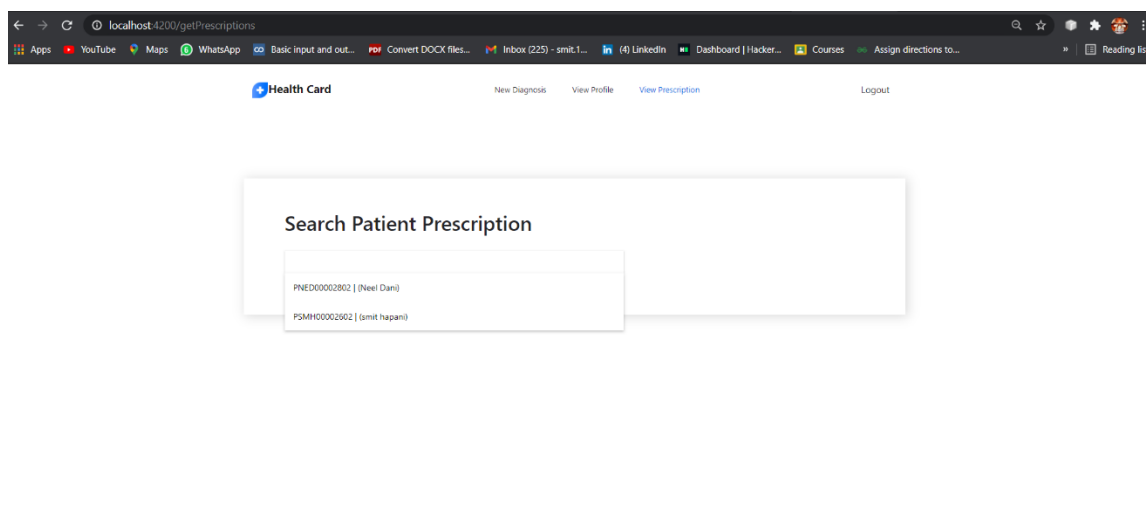


Figure 4.18: View prescription with search functionality

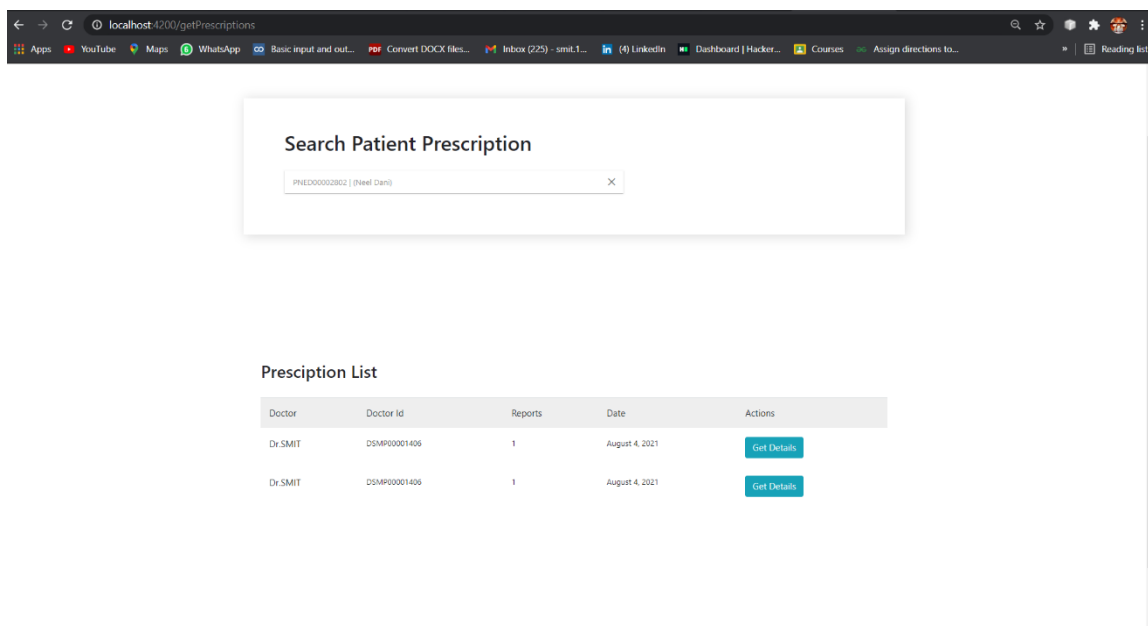


Figure 4.19: View prescription of the selected patient

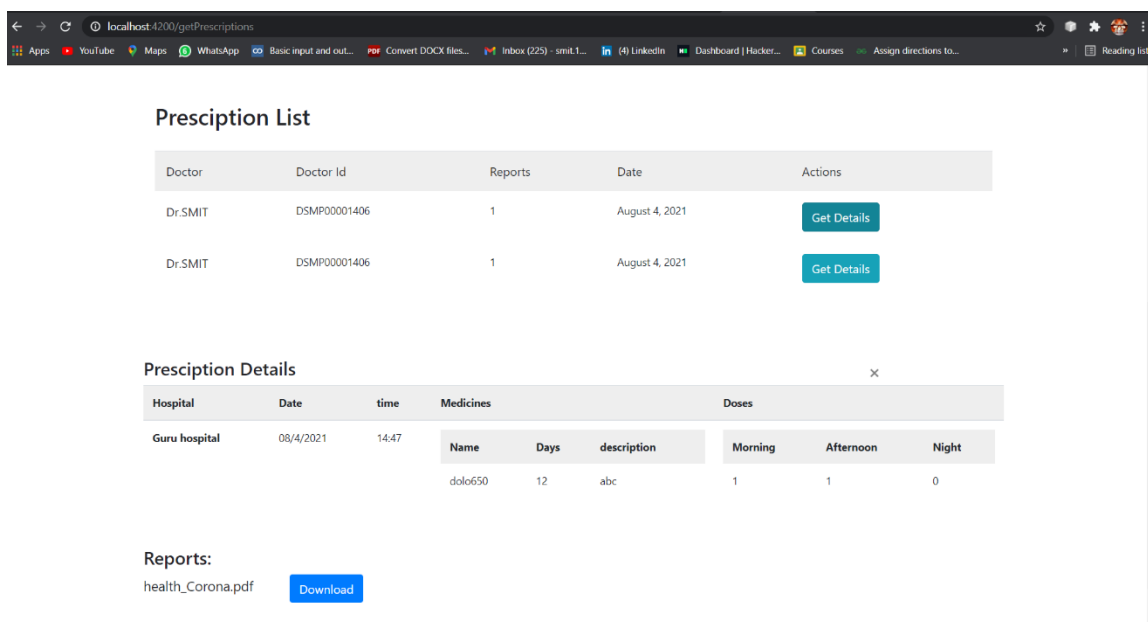


Figure 4.20: Detailed information of the selected patient's prescription

Prescription List

Doctor	Doctor Id	Reports	Date	Actions
Dr.SMIT	DSMP00001406	1	August 4, 2021	Get Details
Dr.SMIT	DSMP00001406	1	August 4, 2021	Get Details

Prescription Details

Hospital	Date	time	Medicines	Doses												
Guru hospital	08/4/2021	14:47	<table border="1"> <thead> <tr> <th>Name</th> <th>Days</th> <th>description</th> </tr> </thead> <tbody> <tr> <td>dolo650</td> <td>12</td> <td>abc</td> </tr> </tbody> </table>	Name	Days	description	dolo650	12	abc	<table border="1"> <thead> <tr> <th>Morning</th> <th>Afternoon</th> <th>Night</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	Morning	Afternoon	Night	1	1	0
Name	Days	description														
dolo650	12	abc														
Morning	Afternoon	Night														
1	1	0														

Reports:
health_Corona.pdf [Download](#)

health_Corona.pdf [Show all](#)

Figure 4.21: Downloaded report

4.3 Validations

Server-side Validation

With server-side validation, all form information entered by the user is sent to the server to be validated upon form submittal. In the event of a validation error, the original page is reloaded along with the error information. When combined with a one field at a time approach, this is the least user-friendly form of validation there is because the user must wait for the data to make a round trip to the server a back on each form submittal. On the positive side, it does bolster security, as server-side validation cannot be bypassed like client-side validation can. There is also the added benefit that the exact validation used is harder for a would-be-hacker to get at. So to sum up:

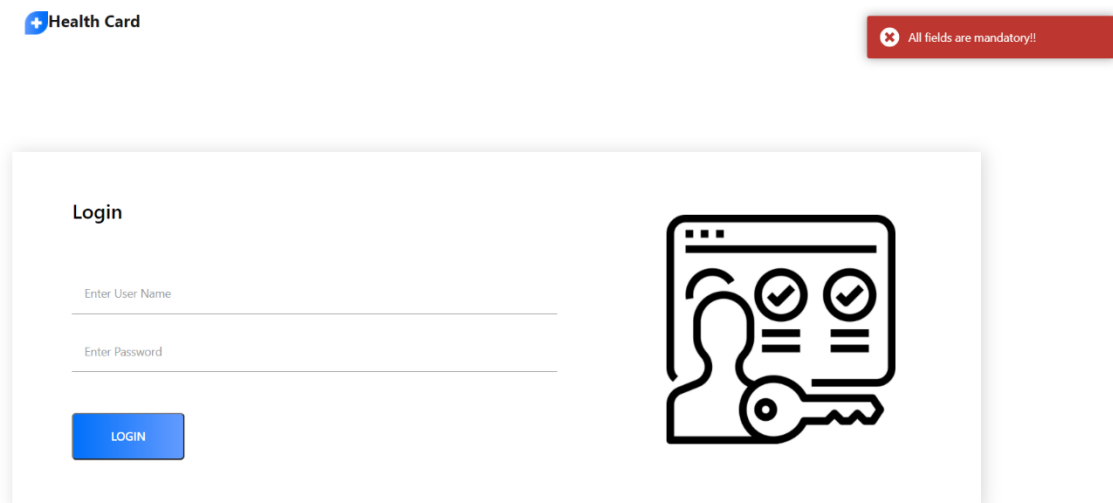
Server-side Validation = Slow Response but Strong Security

Client-side Validation

Although HTML5 has begun to take over some of the common client-side validation duties, JavaScript is still the language of choice for client-side validation today. Unfortunately, as long as users can disable JavaScript in their browser, you cannot rely solely on JavaScript as a method of validation. Rather, client-side validation should be thought of more as an extra layer on top of server-side validation. That means you've now got duplicate validation code.


Client-side Validation = Fast Validation but Code Duplication

Validations in Health-Card System



The screenshot shows a web interface for a 'Health Card' system. At the top left is a blue header with a plus icon and the text 'Health Card'. At the top right is a red error message box with a white 'x' icon and the text 'All fields are mandatory!!'. Below the header is a white login form. The form has the title 'Login' and two input fields: 'Enter User Name' and 'Enter Password'. A blue 'LOGIN' button is at the bottom left of the form. To the right of the form is a black line-art icon depicting a person's profile, a key, and a document with two checkmarks, symbolizing authentication and security.

Figure 4.22: Login fields required validation

 Health Card

Invalid email and password

Login

smit.1999.panchal@gmail.com

LOGIN





Figure 4.23: Check credentials valid or not

 Health Card

All fields are mandatory!

Registration

First Name

Last Name

Enter Password

Confirm Password

Enter Email Id

Enter Contact Number

Select Blood Group

Date of Birth

Address

Sign Up As

REGISTER




Figure 4.24: Signup fields required validation

Registration

SMIT

PANCHAL

+

smit.1999.panchal@gmail.com

06351036668


A-

2021-04-07

A-301,Vishranti Enclave
Nr. Ambe School, Manjalpur

As Patient

REGISTER



✖ Password and confirm Password does not match

Figure 4.25: Password and confirm password match validation

Registration

SMIT

PANCHAL

smit.1999.panchal@gmail.com

06351036668


A-

2021-04-07

A-301,Vishranti Enclave
Nr. Ambe School, Manjalpur

As Patient

REGISTER



✖ Email already exists

Figure 4.26: Check email already exists or not

Patient Id and Symptoms are mandatory!

New Diagnosis

DoctorId: DSMP00001406

Name: Dr. SMIT

Date: 08-4-2021

Patient Id:

Symptoms:

Add Symptoms

Reports

Required:

Add Symptoms

ADD DIAGNOSIS

Figure 4.27: Fields required

Invalid Patient-id

New Diagnosis

DoctorId: DSMP00001406

Name: Dr. SMIT

Date: 08-4-2021

Patient Id:

Symptoms:

Reports
Required:

[ADD DIAGNOSIS](#)

Figure 4.28: Check patient-id is valid or not

4.4 Application Navigation

Doctor Module Navigation chart

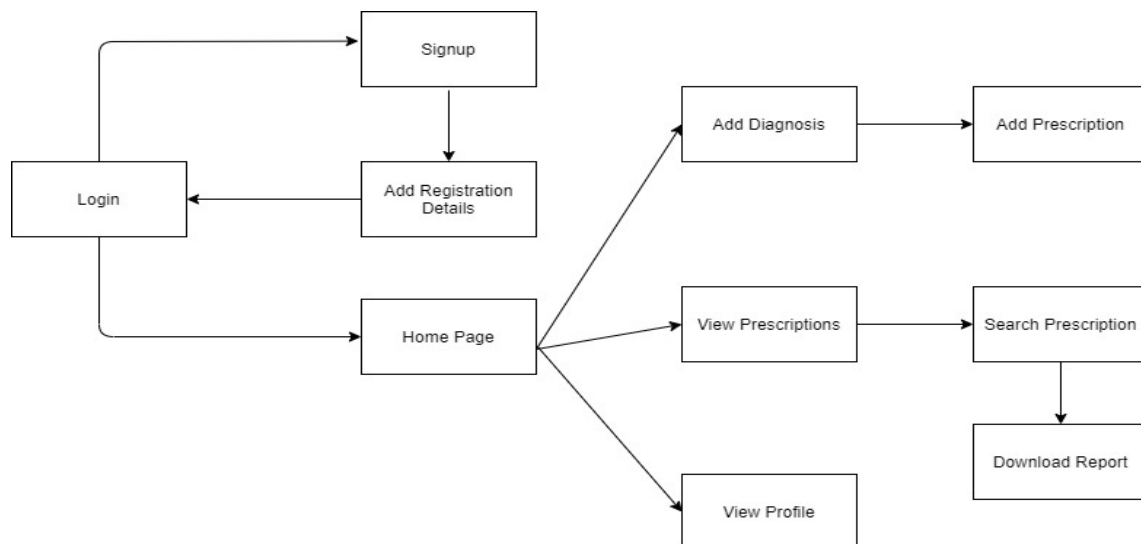


Figure 4.29: Doctor module navigation chart

Laboratorian Module Navigation chart

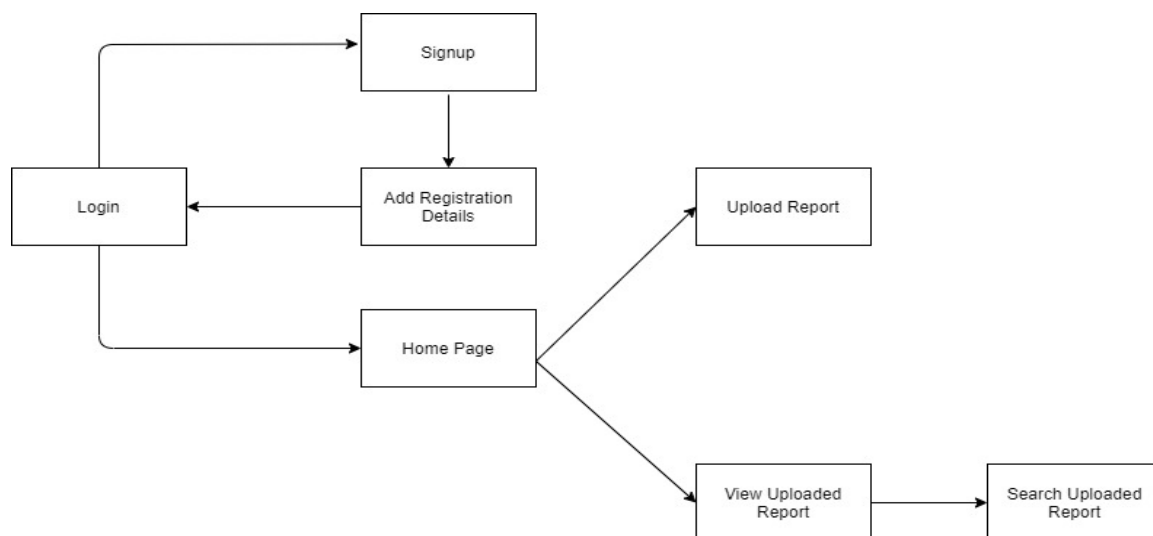


Figure 4.30: Laboratorian Module Navigation chart

Patient Module Navigation chart

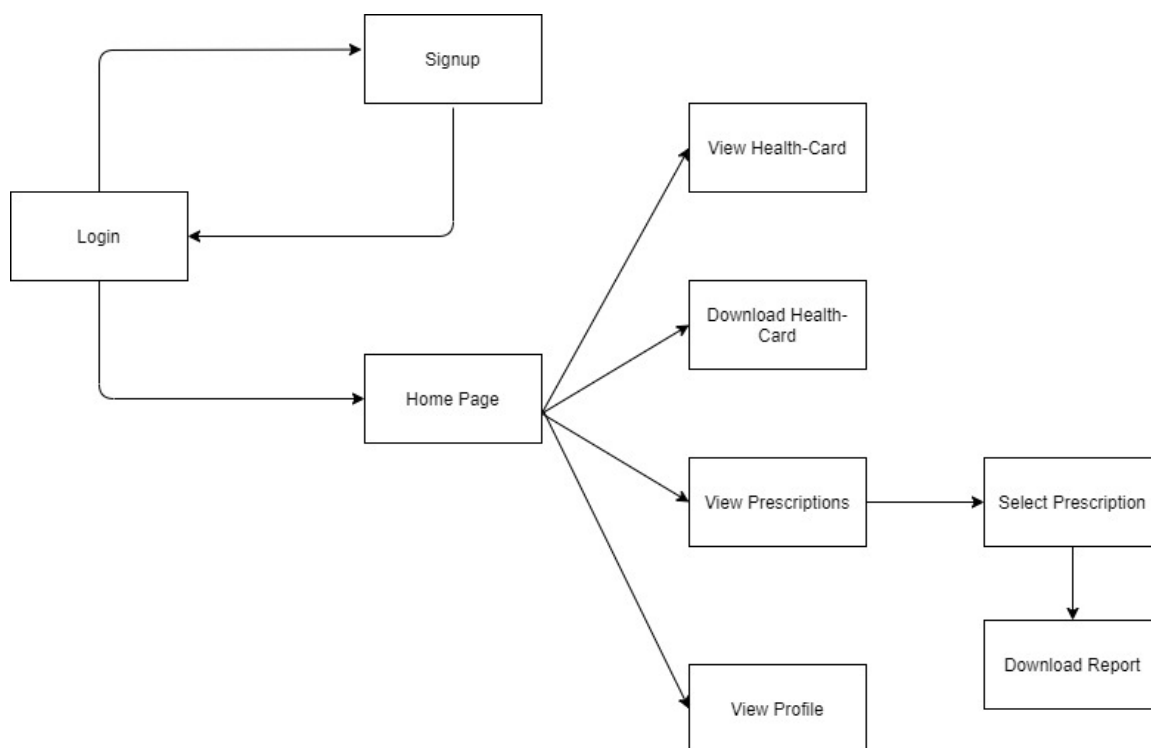


Figure 4.31: Patient Module Navigation chart

Chapter 5

Implementation

5. Implementation

5.1 Implementation Environment

In this project, the implementation environment is MEAN stack (Mongo DB, Express.js, Angular, and Node.js)

Mongo DB:

Mongo DB is a document-oriented NoSQL **database** used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, Mongo DB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in Mongo DB. Collections contain sets of documents and function which is the equivalent of relational database tables.

Express.js:

Express is the most popular Node web framework, and is the underlying library for a number of other popular Node web frameworks. It provides mechanisms to:

- Write handlers for requests with different HTTP verbs at different URL paths (routes).
- Integrate with "view" rendering engines in order to generate responses by inserting data into templates.
- Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.
- Add additional request processing "middleware" at any point within the request handling pipeline.

Angular:

Angular is a Typescript based **frontend web application framework**. One of the major advantage is that the Angular support for web application that can fit in any screen resolution. Angular application is fully compatible for mobiles, tablets, laptops or desktops. Angular has an excellent user interface library for web developers which contains reusable UI components. This functionality helps us to create Single Page Applications (SPA). SPA is reactive and fast application.

Node.js:

Node.js is an open source, cross-platform runtime environment for developing **server-side** and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

5.2 Coding Standards

Coding standards contribute to an improved comprehension of source code. Perhaps one of the most influential aids to understanding the logical flow of an application is how the various elements of the application are named. A name should tell “what” rather than “how”. By avoiding names that expose the underlying implementation, which can change, you preserve a layer of abstraction that simplifies the complexity.

Reasons for using the coding standards are:

- Uniform distribution
- Sound understanding
- Encourager good programming skills

5.3 Implementation of Modules

5.3.1 Front-end

The basic building blocks of the Angular framework are **Angular components** that are organized into **NgModules**.

NgModules:

An NgModule declares a compilation context for a set of components that is dedicated to an application domain, a workflow, or a closely related set of capabilities. An NgModule can associate its components with related code, such as services, to form functional units.

Every Angular app has a root module, conventionally named AppModule, which provides the bootstrap mechanism that launches the application. An app typically contains many functional modules.

Module in Health-Card system:

- **App Module:** Packages all the dependencies required by the components of the system

Components:

Each component defines a class that contains application data and logic, and is associated with an HTML template that defines a view to be displayed in a target environment.

Every Angular application has at least one component, the root component conventionally named AppComponent that connects a component hierarchy with the page document object model (DOM).

Components in Health-Card system:

- Header Component: Acts a master header since it is used in every page which reduces coding redundancy.
- Footer Component: Acts a master footer since it is used in every page which increase code reusability.
- Signup Component: Component that allows users to register using required custom credentials.
- Login Component: Authorize the user to access the application once it is being successfully registered in the system.

Components of Doctor Module:

- Diagnosis Component: Component that provides the functionality of add new diagnosis of patient and also load as home component when any doctor login successfully.
- Add Prescription Component: Component that provides the functionality of add new prescription of patient.
- View Prescription Component: Component that provides the functionality to view previously added prescriptions of patient and download report uploaded by laboratorian.
- View Profile Component: Component that allows to view doctor's basic and work details.

Components of Patient Module:

- Home Component: Component that loads after patient's successfully login and provides the functionality of view and download the health-card.
- View Prescription Component: Component that provides the functionality to view previously added prescriptions of that patient and download report uploaded by laboratorian.
- View Profile Component: Component that allows to view patient's basic details.

Components of Laboratorian Module:

- Home Component: Component that loads after laboratorian's successfully login and also lists the repots which need to be upload by that laboratorian.
- Uploaded Reports Component: Component that lists the reports which are uploaded by that laboratorian and also provides searching among the list of reports.

5.3.2 Back-end

Express.js is used to support backend code logic. Restful APIs have been created that are being consumed by the frontend.

APIs in Health-Card system:

registerUser : For registration of a new user as patient/doctor /laboratorian.

login : For validating the registered user.

getUserId : To generate unique user id using firstname, lastname, user type and date of birth fields of user.

doctorDetail : To add doctor's additional work details.

labDetail : To add laboratorian's additional work details.

getUserData : To get all the details of the user for given user id.

addDiagnosis : To add patient's new diagnosis in doctor module.

addPrescription : To add new prescription and update the diagnosis with generated prescription in doctor module.

getPrescriptionsById : To get list of the prescriptions for the provided user id.

getPrescriptionsByDoc : To get list of the prescriptions for the provided doctor id.

getDetailedPrescriptionById : To get prescription in detail for the given prescription id.

getReports : To get list of reports which are remaining to upload by laboratorian.

uploadReport : To upload the report of patient on server and update the diagnosis with report uploaded.

uploadedReport : To get list of uploaded reports by laboratorian.

downloadReport : To download the selected report from the server in doctor and patient module.

Chapter 6

Test Case Design

6. Test Case Design

6.1 Testing Plan

The objective of the system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and ensure that the computer system and the associated clerical and other procedures work together. Systems are not designed as entire systems but they are tested as single system. The analyst must perform both unit and system testing.

Different types of testing methods are available. We have tested our system for different aspects like Does the application meet the goals for which it has been designed ? This was a very important question that stood before us as the application was designed to be implemented on such a large network.

To fulfil its goal of being able to run on different system we went through a series of test at different places where this is supported to be used the most. As we need to make our system efficient enough, we need to test it thoroughly.

Finally, we tested the system with the real-time data, for which it is actually designed. We are almost successful in satisfying our needs as it was designed according to their requirements. But it is very necessary to maintain this application and so our work is not still over.

6.2 Testing Strategy

The development process repeats this testing sub-process a number of times for the following phases.

- a) Unit Testing.
- b) Integration Testing.

Unit Testing tests a unit of code (module or program) after coding of that unit is completed.

Integration Testing tests whether the various programs that make up a system, interface with each other as desired, fit together and whether the interfaces between the programs are correct.

Testing is carried out in such a hierarchical manner to ensure that each component is correct and the assembly/combination of components is correct. Merely testing a whole system at the end would most likely throw up errors in components that would be very costly to trace and fix.

6.3 Test Cases

Test Id	Test Scenario	Test Steps	Test Data	Precondition	Expected result	Pass/Fail
1.	Check user login with valid data	1. Open login page 2. Enter user credentials 3. Click on the login button	Valid email and password	The user must already be registered with an email address and password	User should login to the system	Pass

2.	Check user login with invalid data	<ol style="list-style-type: none"> 1. Open login page 2. Enter user credentials 3. Click on the login button 	Invalid email and password	-	User should not be able to login and he should get invalid credential message	Pass
3.	Check user wants to upload a report	<ol style="list-style-type: none"> 1. Open Home page 2. Choose a file 3. Click on the upload button 	-	The user must be logged in as laboratorian	The selected file or report should get uploaded to the server	Pass
4.	Check user wants to download a report	<ol style="list-style-type: none"> 1. Open view prescription 2. Select a prescription which has a report available 3. Click on the download button 	-	The user must be logged in as patient or doctor	The selected report should get downloaded to the user's machine	Pass

Table 6.1 – Test Cases

Chapter 7

Conclusion and Future Extensions

7. Conclusion and Future Extensions

7.1 Future Extensions

Health-Card has a lot of room for improvement as well as addition of new features.

Some of features that can be inculcated into Health-Card are:

- Add pharmacist module
- Allow only verified doctors and laboratorians
- Allow user to update the basic and work information
- By gathering all health data, train the model and generate some statistical analysis

7.2 Conclusion

Hereby, we declare that the project has been performed by understanding all the modules. All the modules mentioned in the thesis were developed separately and further integrated together in a standard module with limited functionalities and features. The main concept behind creating communication between two systems lies in the usage of API of the respective system and the stability of that API. This features were developed by keeping in mind all the modules of the system. Hence the quality of features desired by the user is received.

Bibliography

1. For Html, CSS, Bootstrap, JavaScript.

<https://www.w3schools.com>

2. For mongoose database

<https://mongoosejs.com/>

3. For understanding Angular.

<https://angular.io>

4. For solving doubts.

<https://stackoverflow.com>

5. For download and setup of Node.js.

<https://nodejs.org/en/>