

A Project Report on

MEDPREDICT CRYPTO PAY

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

In

Computer Science and Engineering

Submitted by

B. MANI CHANDRA
(20H51A0506)

B. SATHWIK
(20H51A0559)

K.SAI HARSHA
(20H51A05C7)

Under the esteemed guidance of
MS. ARCHANA BATHULA
(ASSISTANT PROFESSOR)



Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)

*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A⁺ Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2020- 2024

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Major Project report entitled "**MEDPREDICT CRYPTOPAY**" being submitted by **B. MANI CHANDRA (20H51A0506)**, **B. SATHWIK(20H51A0559)**, **K. SAI HARSHA (20H51A05C7)** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

MS. ARCHANA BATHULA
Assistant Professor
Dept. of CSE

Dr. Siva Skandha Sanagala
Associate Professor
& HOD Dept. of CSE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project a grand success.

We are grateful to **Ms. Archana Bathula**, Assistant Professor , Department of Computer Science and Engineering for her valuable technical suggestions and guidance during the execution of this project work.

We would like to thank, **Dr. Siva Skandha Sanagala**, Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana**, Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary& Correspondent, CMR Group of Institutions, and **Shri Ch Abhinav Reddy**, CEO, CMR Group of Institutions for their continuous care and support.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly or indirectly in completion of this project work.

B.MANI CHANDRA	20H51A0506
B.SATHWIK	20H51A0559
K.SAI HARSHA	20H51A05C7

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	ii
	LIST OF TABLES	iii
	ABSTRACT	iv
1	INTRODUCTION	1
	1.1 Problem Statement	2
	1.2 Research Objective	2
	1.3 Project Scope and Limitations	2
2	BACKGROUND WORK	3
	2.1. Human Disease Prediction using Random Forest	4
	2.1.1.Introduction	4
	2.1.2.Merits,Demerits and Challenges	4
	2.1.3.Implementation	5
	2.2. Online Doctor Management System Synopsis	6
	2.2.1.Introduction	6
	2.2.2.Merits,Demerits and Challenges	6
	2.2.3.Implementation	6
	2.3. Doctor Appointment Online Booking	7
	2.3.1.Introduction	7
	2.3.2.Merits,Demerits and Challenges	7
	2.3.3.Implementation of	8
3	PROPOSED SYSTEM	9
	3.1. Objective of Proposed Model	10
	3.2. Algorithms Used for Proposed Model	10
	3.3. Designing	11
	3.3.1.Patient Architecture	11
	3.3.2.Doctor Architecture	11
	3.3.3.Admin Architecture	12
	3.4. Stepwise Implementation and Code	12-30
4	RESULTS AND DISCUSSION	31-38
	4.1 Result	31-37

	4.2. Performance metrics	38
5	CONCLUSION	39
	5.1 Conclusion	40
	5.2 Future Enhancement	40-41
	REFERENCES	42-43
	APPENDIX	44-54
	PUBLICATION	55

List of Figures**FIGURE**

NO.	TITLE	PAGE NO.
2.1.	Methodology of Random Forest Algorithm	21
3.1	Patient Architecture	11
3.2	Doctor Architecture	11
3.3	Admin Architecture	12
4.1	Admin Login	35
4.2	Admin View	35
4.3	Patient Login	36
4.4	Patient Profile	36
4.5	Patient Consulting a Doctor	36
4.6	Patient Payment has Succeeded	36
4.7	Chat Window	37
4.8	Doctor Login	37
4.9	Doctor Profile	37
4.10	Payment Window	38
4.11	Payment Method	38
4.12	User Interface-1	40
4.13	User Interface-2	40

List of Tables

FIGURE

NO.	TITLE	PAGE NO.
4.1	Performance metrics	41

ABSTRACT

In the evolving landscape of healthcare, efficient diagnosis and payment mechanisms play pivotal roles in enhancing patient care and operational efficacy. This introduces MedPredict CryptoPay, a novel system integrating disease prediction with cryptocurrency payment, underpinned by blockchain technology. It enables users to input symptoms for disease prediction, select specific healthcare professionals, and seamlessly pay consultation fees using cryptocurrencies. Leveraging advanced machine learning algorithms, the system accurately predicts diseases based on user-provided symptoms, facilitating timely medical intervention. The integration of blockchain ensures transparent, secure, and immutable transactions, fostering trust among stakeholders while safeguarding sensitive healthcare data. It represents a pioneering initiative at the intersection of healthcare, predictive analytics, and blockchain technology, promising transformative impact on healthcare accessibility, efficiency, and security.

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1.Problem Statement

In contemporary healthcare, there is a need to enhance disease diagnosis and medical consultation processes. Many patients seek accurate and timely disease predictions, while healthcare providers require secure and transparent payment methods for their services. Combining the power of machine learning with medical expertise, "Med Predict Crypto Pay" aims to address these needs by offering a system that predicts diseases from patient symptoms, enables cryptocurrency-based payments, and fosters a collaborative approach to medical prescription. To securely pay the fee and to get prescription for the symptoms and to securely transfer the information.

1.2.Research Objective

It is to develop an machine learning with medical expertise and utilizes blockchain technology for secure payment transactions. Specifically, the primary research objectives include:

Machine Learning Model Development: Develop and refine machine learning models that can accurately predict diseases from patient-reported symptoms, encompassing textual descriptions.

Secure Payments: Integrating blockchain and cryptocurrency technology to provide patients with a secure and transparent method of paying for medical service.

Cryptocurrency Integration: Explore the integration of various cryptocurrencies and blockchain technologies into the system, ensuring a seamless and user-friendly payment process for patients and healthcare providers.

1.3.Project Scope

1. The disease prediction system have 3 users such as doctor, patient and admin.
2. Each user of the system are authenticated by the system.
3. There is a role based access to the system.
4. The system allows the patient to give symptoms and according to those symptoms the system will predict a disease.
5. The system suggests doctors for predicted diseases.
6. The system allows online consultation for patients.
7. The system helps the patients to consult the doctor at their convenience by sitting at home.

CHAPTER 2

BACKGROUND

WORK

CHAPTER 2

BACKGROUND WORK

2.1 Human Disease Prediction using Random Forest

2.1.1. Introduction

The proposed model is providing an enhanced and accurate model for predicting human diseases from the symptoms. The dataset from Kaggle is used, and the methods used to train the models are the [5] Rainforest algorithm, LSTM algorithm and SVM algorithm to train our data.

The working model will be as follows:

1. The human will enter his/her symptoms.
2. The symptoms will then be inputted into our model.
3. The model will then yield the possible disease.

The novelty of the proposed work is that tweaking the Random forest model by using hyperparameters, improves the efficacy of the model. Hence, it is providing more accuracy.

2.1.2. Merits, Demerits and Challenges

Merits

In the database, the author has modified the symptoms (inputs) based on the following parameters:

1. Rarity: The rarer a symptom is, the more weight is given to it. Thus, the Random Forest Model predicts a disease more accurately according to the symptoms.
2. Location: Some diseases are only bound to happen in a particular geographic location.
3. Thus, the database is set in such a way that the algorithm discards all the diseases that are not present in the inputted location.

Demerits

1. Execution time: It requires huge execution time and space for the compilation of the decision trees.
2. Stability: It works better in a stable environment where the dataset is less noisy and subjected to be less dynamic.
3. Overfitting: It may lead to an overfitted model when provided with noise.

Challenges

No blockchain is used and no feedback from doctor.

2.1.3 Implementation

The random forest produces decision trees from multiple data using their average for regression and most of the voting for categorization. The research reported by Paul et al. used the Random Forest Algorithm as the main algorithm. The random forest algorithm is used to train the model with the dataset which contains a combination of symptoms this methodology is illustrated in figure 2.1 and the corresponding diseases. The driving force behind using the random forest algorithm is that it has the capacity to handle data sets with continuous variables, as in regression, and categorical variables, as in classification. It produces superior results with regard to classification problems. The working method of the Random Forest is .

Step 1: Select arbitrary samples from a given data set or training set.

Step 2: This method will create a decision tree for every training data set.

Step 3: Using the decision tree's average, voting will be done.

Step 4: Lastly, select the predicted outcome that garnered the greatest support as the final prediction outcome.

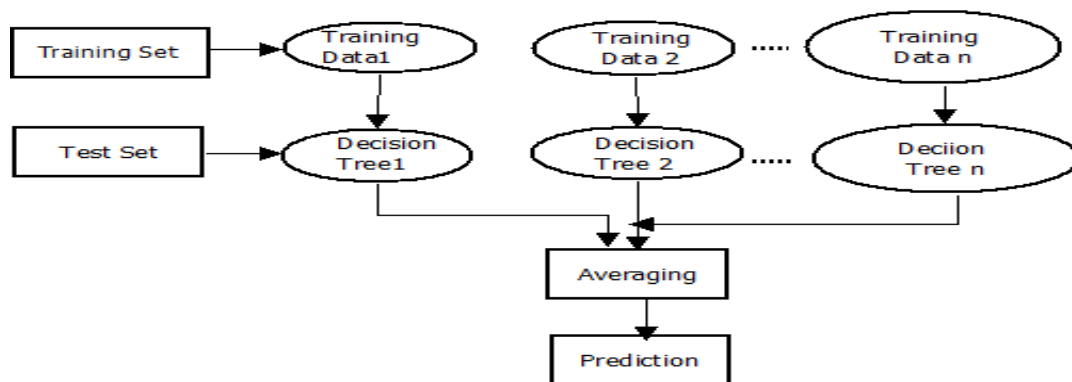


Figure 2.1. Methodology of Random Forest Algorithm

2.2 Online Doctor Management System Synopsis

2.2.1. Introduction

The establishment and improvement of [4]doctor-patient interaction system is a very important requirement, especially now when the mobile communication technology is developing rapidly. The advantages of mobile web can be made full use of to make up the time and distance gap between doctors and patients and to provide fast and adequate medical services. Through the connection between mobile terminals and specific service, both doctors and patients are able to obtain required data to achieve a better interaction. Android is a Linux based open source operating system which is mainly used in portal devices with excellent performance thus making its market share growing. The platform, a services and database technology are all gradually maturing, so that we can develop a doctor- patient interaction system on Android platform to meet the needs of the patient and provide doctors more efficient and convenient means of communication with patients.

2.2.2 Merits, Demerits and Challenges

Merits

Doctor patient interaction system.

Demerits

No machine learning.

Challenges

Verifying the identity of patients and doctors is essential to prevent fraud and protect patient information.

2.2.3 Implementation

To make a truly online doctor system to have meet with online doctors, all manual process has been automated through this system. Patient have to fill online form by which id and password created and sended to their email and upon accepting data, automatic login to patient panel. Through this panel, patients can select the doctors and have appointment with them on their time from their own place. Patients will get all their reports and medicine prescriptions in their inbox by notification indication just after appointment session. There is no need of cash and a secure payment gateway has been used to pay the required fees using their account or debit or credit card.

2.3. Doctor Appointment Online Booking System

2.3.1 Introduction

The proposed work in this paper is an Online Hospital Management Application that uses an [3] android platform that makes the task of making an appointment from the doctor easy and reliable for the users. Android based online doctor appointment application contains two modules. One module is the application designed for the patient that contains a login screen. The patient has to register himself before logging in to the application. After logging in, the patient can select a hospital and can view the hospital details. The patient has the option of selecting a doctor from the list of doctors and can view the doctor's details. The patient can request for an appointment on his/her preferred day/time. The selected day/time slot will be reserved and patient will receive the notification of the successfully added appointment. The patient can view the location of the hospital on map. In addition, the patient can contact to the hospital and the doctor by making a call or may send an email to the doctor. There are considerable online scheduling tools in the internet, a few of which are trait loaded, simple to setup and economical For practitioners, online appointment reservation and scheduling delivers a lot of merit added benefits and services, like captivating the patient, composing the patient to feel welcomed, and being capable to save patients' details safely for future information. But the most admirable and useful preference is that online appointment reservation and scheduling is remarkably in expensive .Both doctors and patients can access the portal through their unique ID's.

2.3.2 Merits, Demerits and Challenges

Merits

Doctor appointment allotting

Demerits

No blockchain

Challenges

Waiting time simply means a period of time which one must wait in order for a specific action to occur, after that action is requested or mandated . Patients' waiting time has been defined as the length of time from when the patient entered the outpatient clinic to the time the patient actually received his or her prescription.

2.3.3 Implementation

The proposed system consists of two panels: Doctor and Patient. The users will first have to download the application and install it in their mobile devices. Once installed, this application will remain into the device permanently until the user deletes it or uninstalls it. The patient will have to register into the application for the first time. On registering, the patient will receive a username and password. The patient can use this username and password for logging into the app each time he uses it. After logging in, the patient will have to select a filtration type. The filtration is done on two bases: Area wise and Specialty wise. After selecting the filtration type, the doctors list will be displayed. The patient can select any particular doctor and view his profile. Also the patient can view the doctor's schedule and look for an appointment according to his convenience. The patient will then send a request for appointment. The doctor can either accept the appointment or reject it. The database will get updated accordingly and the patient will get a confirmation message. The add-on to this system is that the patient will receive a notification 2 hours before the actual appointment. This will be very useful in case the patient tends to forget the appointment. The duration a patient waits from the given time of their schedule to the time that they must actually receive the service is known as direct waiting time. The patients use this technique and waste much waiting time just by standing in queue at the registration counter to make sure a successful registration of the appointment has been made with a certain doctor.

CHAPTER 3

PROPOSED SYSTEM

CHAPTER 3

PROPOSED SYSTEM

3.1 Objective of Proposed Model

1. Med predict crypto pay is the system that is used to predict the diseases from the symptoms which are given by the patients.
2. The system processes the symptoms provided by the user as input and it generates the probability of the disease.
3. User can select specific doctor and can pay the fees by using crypto currency with blockchain, and the patient and doctor can chat by using chat box.

3.2 Algorithms Used for Proposed Model

3.2.1 Algorithm 1: A disease prediction model based on Naïve Bayes

Input: Symptoms, Trained model
Output: Disease name

1. *Procedure* ML_model(*data*={*symptom1*,*symptom2*,.....,*symptomN*}, *Trained model*)
2. *Data* <- *load_data(data_symptoms)*
3. *Data* <- *Data_preprocessing(Data)*
4. *X_train* <- *Data[Symptom]*, *Y_train* <- *Data[Disease]*
5. *Model* <- *MultinomialNB()*
6. *Model* <- *Model.fit(X_train,Y_train)*
7. *Prediction* <- *Model.predict(X_train)*
8. *Accuracy* <- *accuracy_score(Y_train,Prediction)*
9. *Disease* <- *classification_report(Y_train, Prediction)*
10. *Display Accuracy and Disease*
11. *End*

3.2.2 Algorithm 2: Disease Prediction Using naïve bayes

For each instance Xi in Dataset:

P_class = *calculate_prior_probability(class)*

P_data_given_class = *calculate_likelihood(Xi, class, Symptoms)*

P_data = *calculate_evidence(Xi, Symptoms)*

$\phi = \text{Find_Naive_Bayes_Prediction}(P_class_given_data)$ End

3.3 Designing

3.3.1 Patient Architecture:

1. Patient has to signup/ signin
2. After signin patient have choose the symptoms and have to submit
3. Patient can choose the suggested doctor/any doctor.
4. After selecting the doctor, patient have to pay the consultation fee using crpto currency
5. After successful payment, patient and doctor can have conversation using chatbox.

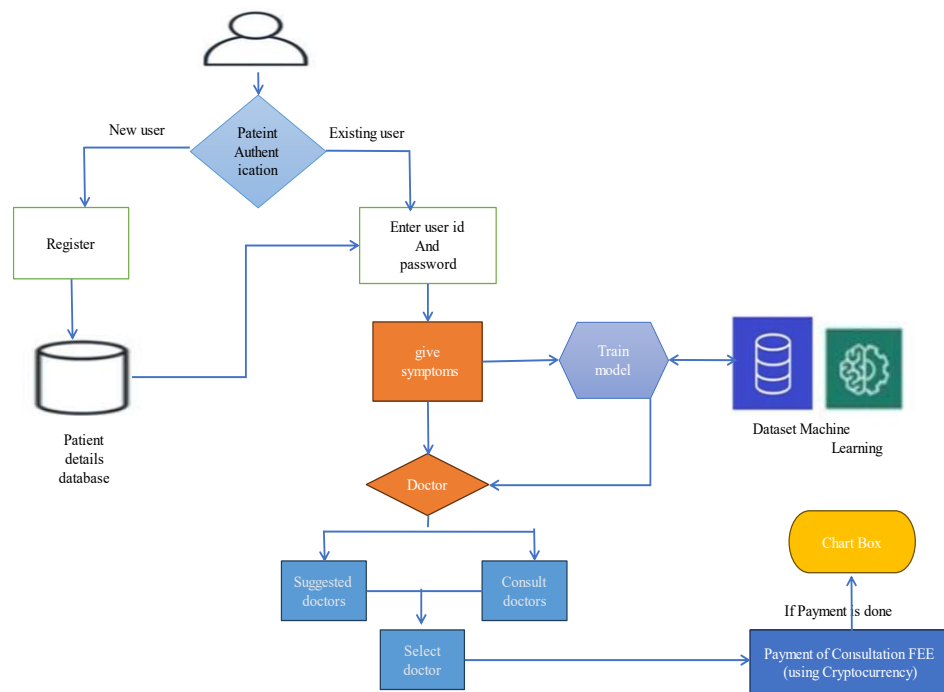


Figure 3.1 Patient Architecture

3.3.2 Doctor Architecture

1. Doctor has to signup/ signin
2. After signin doctor can view the patient in the consultation history .
3. If patient pays the fee then doctor can view the patient symptoms and predicted disease and also can have chat with the patient.

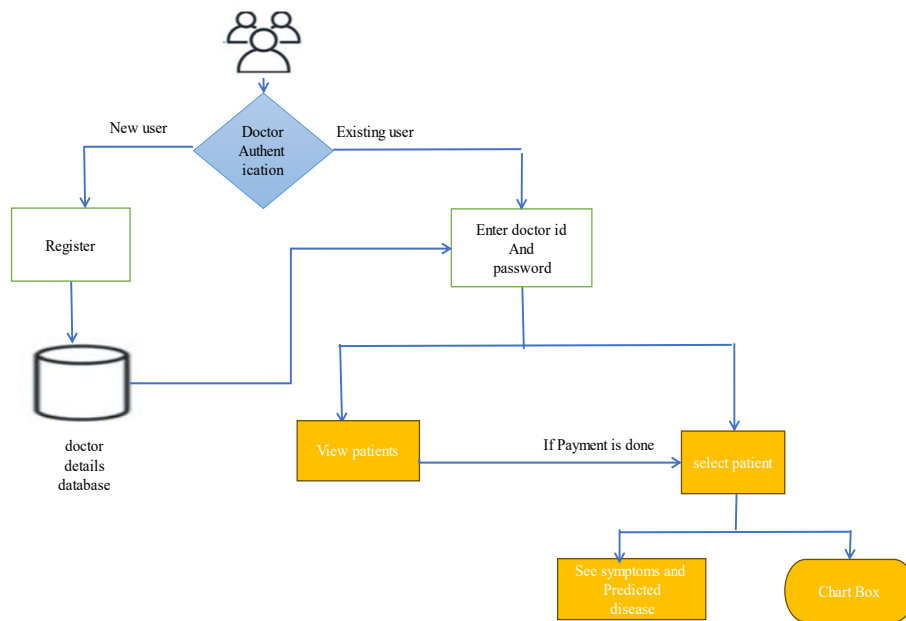


Figure 3.2 Doctor Architecture

3.3.3 Admin Architecture

1. Admin have to login
2. Admin can create/view and delete doctors and patient details.

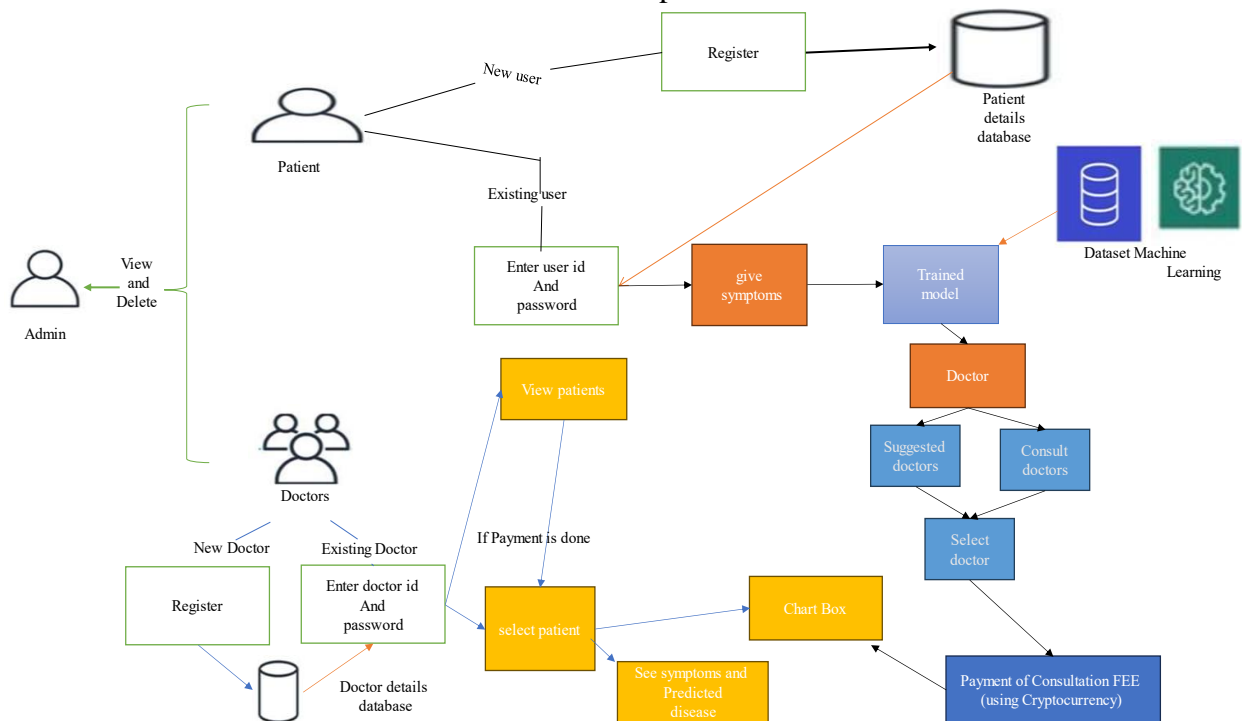


Figure 3.3 Admin Architecture

3.4 Stepwise Implementation and code

1)Open the terminal in the VS code and type the command
” python manage.py runserver”.

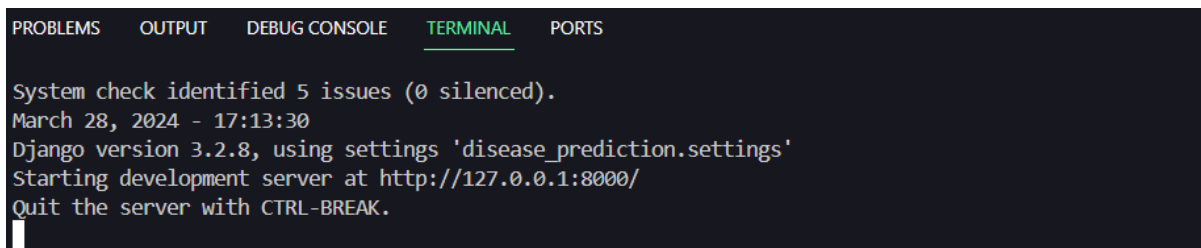


```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Microsoft Windows [Version 10.0.22631.3235]
(c) Microsoft Corporation. All rights reserved.

F:\1\Medpredict_Cryptopay_Final\Medpredict_Cryptopay_Final>python manage.py runserver
```

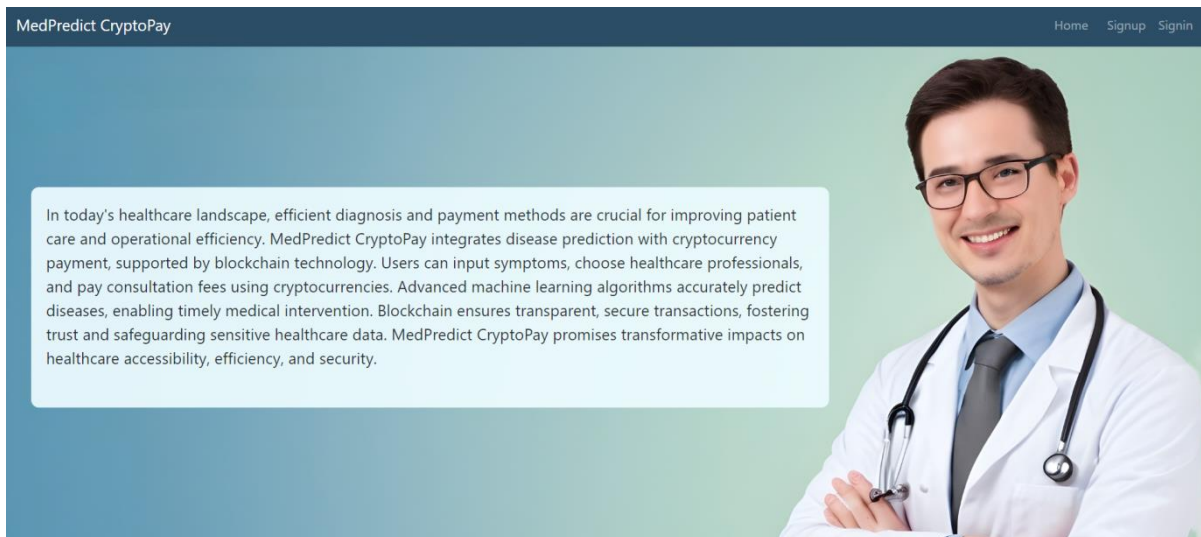
2)Now click on the generated link



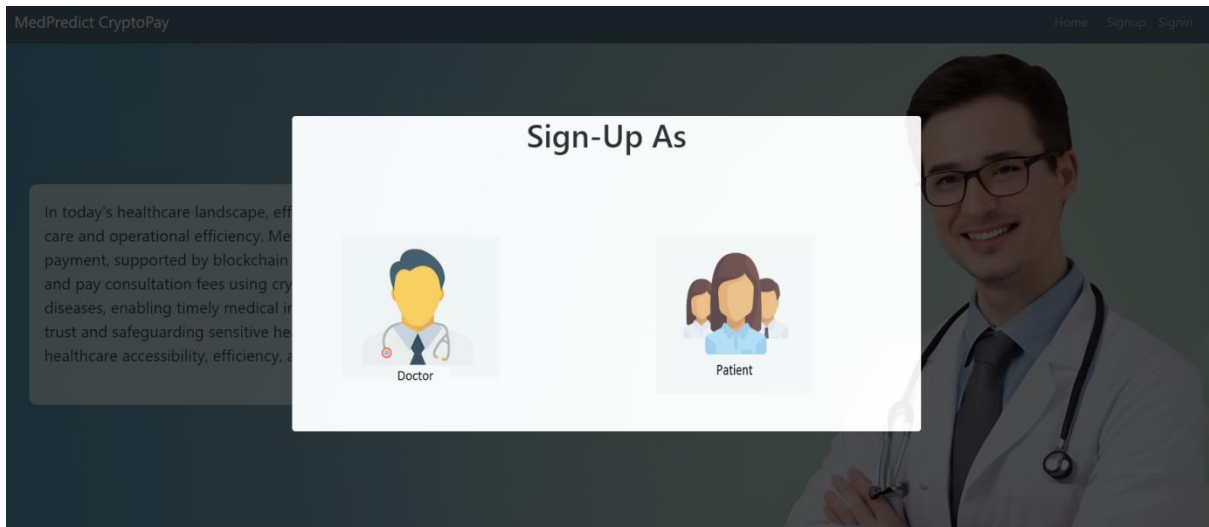
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

System check identified 5 issues (0 silenced).
March 28, 2024 - 17:13:30
Django version 3.2.8, using settings 'disease_prediction.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

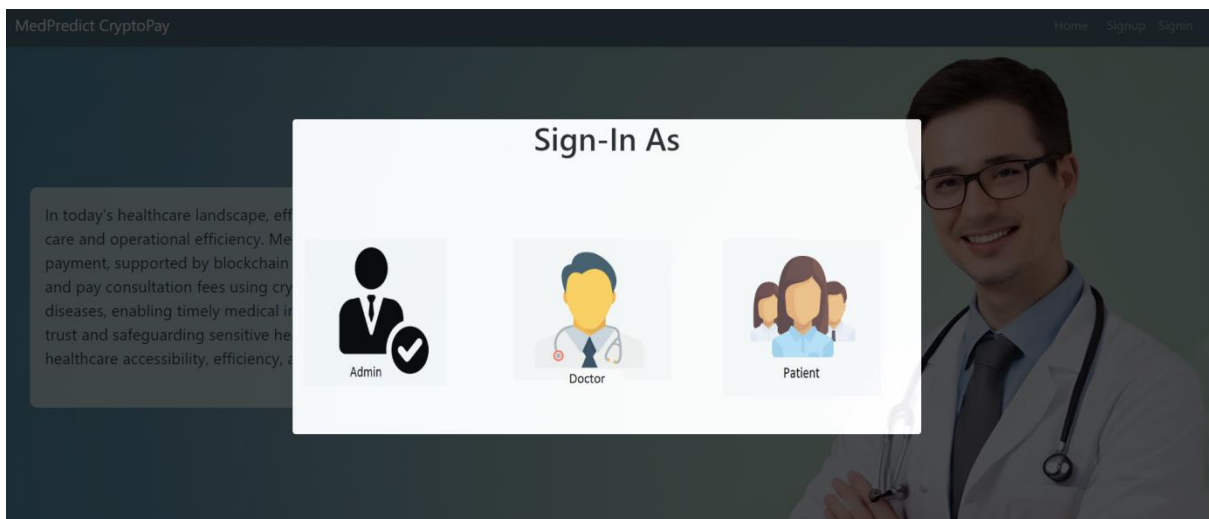
3)The website will open in the browser



4) Doctor and Patient can signup

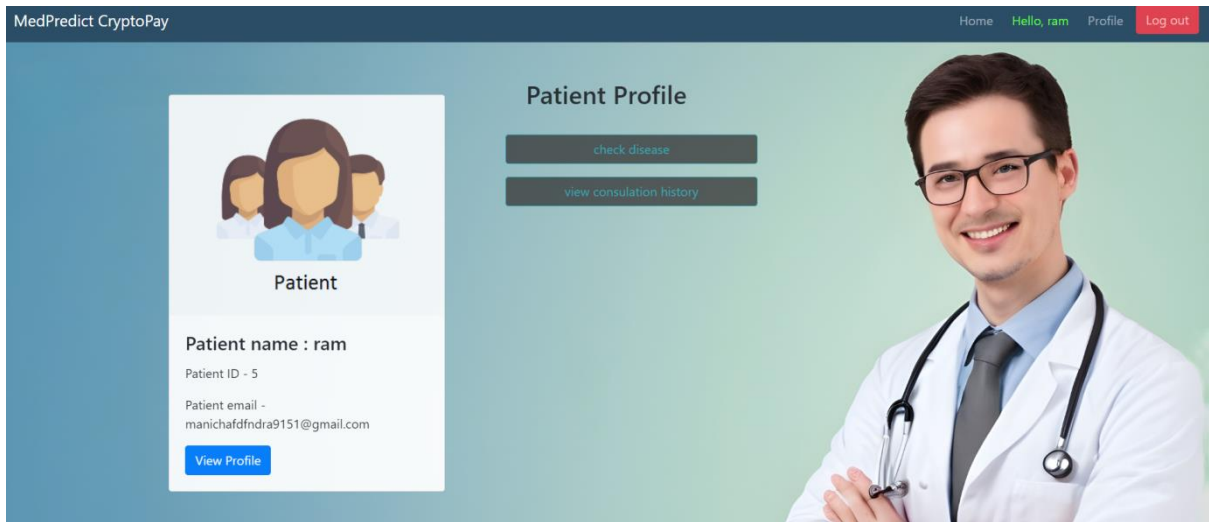


5) Admin, Doctor and Patient can signin

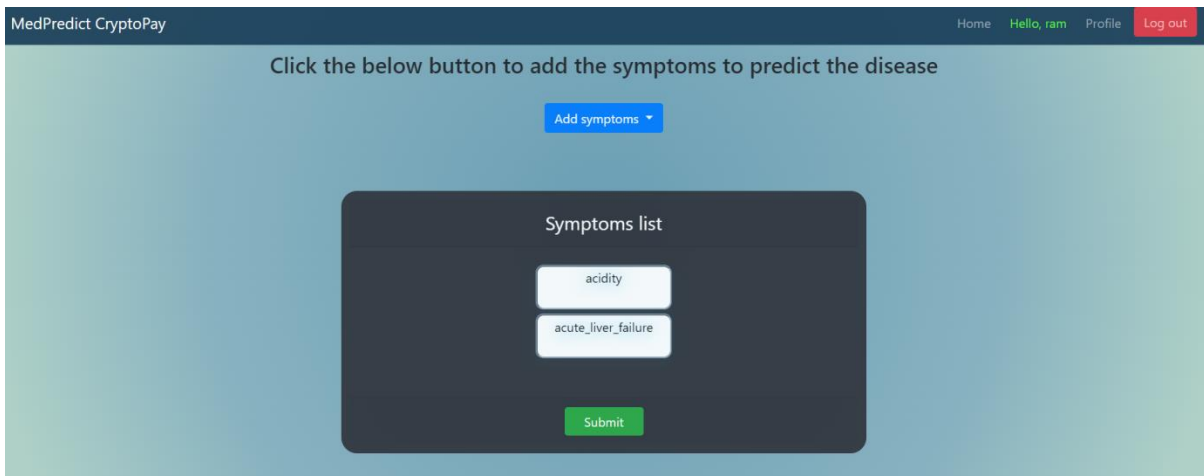


Administrative functionalities form the backbone of the system, requiring secure login credentials for access. Admins wield the authority to create, view, and delete doctor and patient details, akin to administrative privileges in a centralized administrative setup. This structured approach streamlines administrative tasks, ensuring effective management of user data and system functionality.

6)After patitent authentication,the patient can check for disease and view previous consultation history



7)In check disease functionality ,the patient can add symptoms



8)After adding symptoms ,patient can consult the suggested doctor or can consult from all the available doctors

MedPredict CryptoPay

Home Hello, ram Profile Log out

Add symptoms

Symptoms list

acidity

acute_liver_failure

Submit

Patient name: ram
Age: 5

Consult a doctor Suggested Doctors

9) Patient has to pay the doctor fee for the consultation

MedPredict CryptoPay

Home Hello, ram Profile Log out

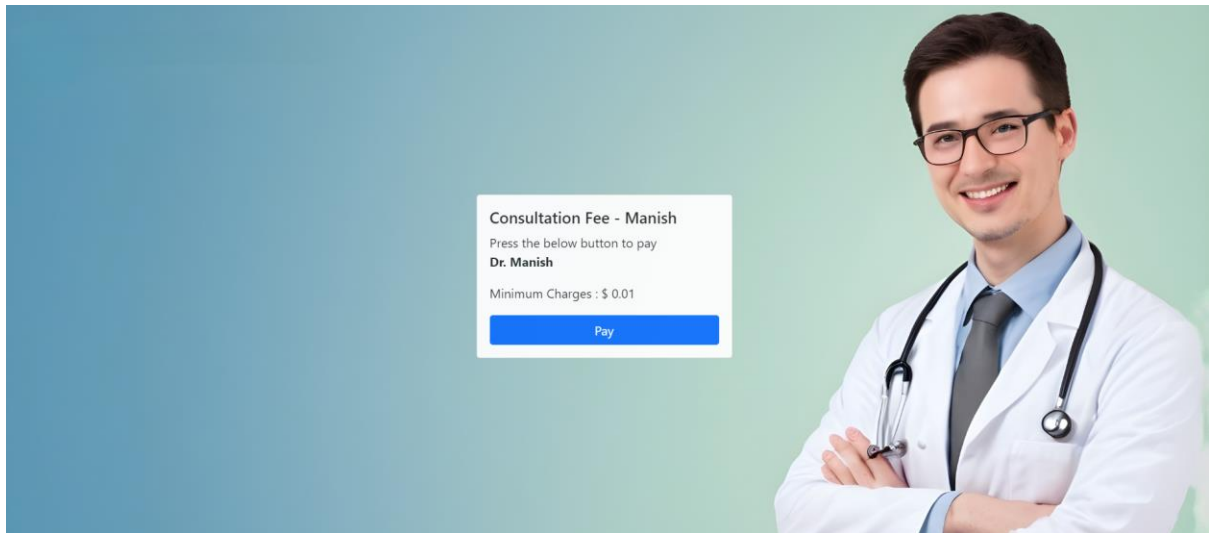
Consult a Doctor

Doctor name	Specialization	Email	View profile	Minimum charges	Pay here	Status
doctor2	Rheumatologist	manichandwhra9151@gmail.com	View profile	\$5.00	Pay here for doctor2	Available
Mahendra	Rheumatologist	mahendra@gmail.com	View profile	\$5.00	Pay here for Mahendra	Offline
Abhinav	Gastroenterologist	Abhinav@gmail.com	View profile	\$5.00	Pay here for Abhinav	Available
Doctor1	Neurologist	manichandra9151@gmail.com	View profile	\$5.00	Pay here for Doctor1	Available
Kiran	Cardiologist	kiran@gmail.com	View profile	\$5.00	Pay here for Kiran	Available

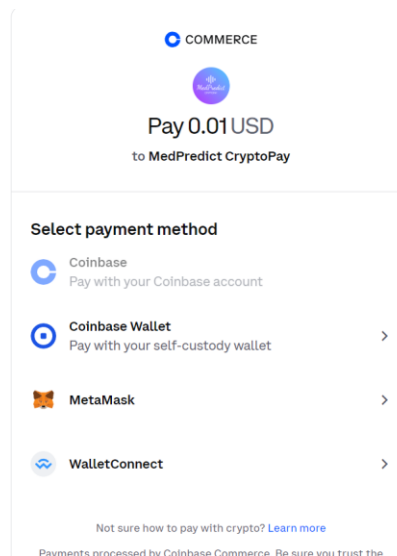
Start

Patient functionalities align with the seamless user experience, necessitating signup/signin processes for authentication. Once authenticated, patients can effortlessly select symptoms and submit them for disease prediction, reflecting the streamlined administrative processes. The option to choose suggested or preferred doctors further enhances user autonomy and convenience, akin to user-centric administrative practices.

10) Pay button redirects to coinbase

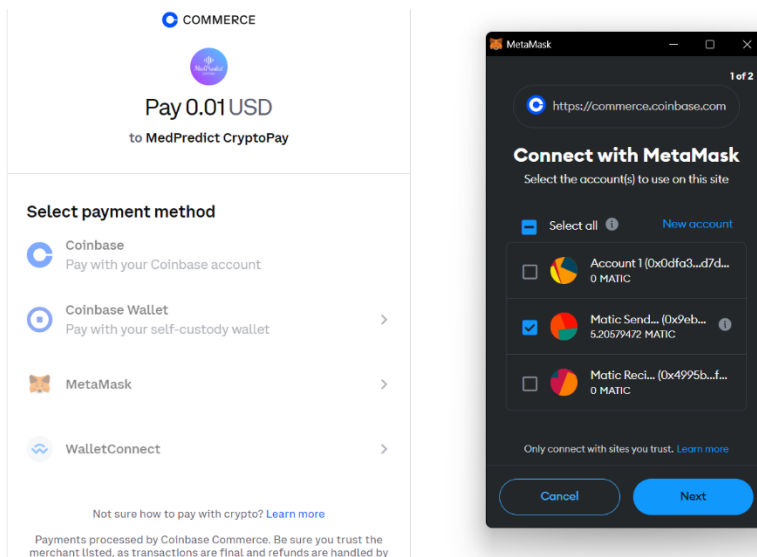


11) Patient has to connect the wallet for payment

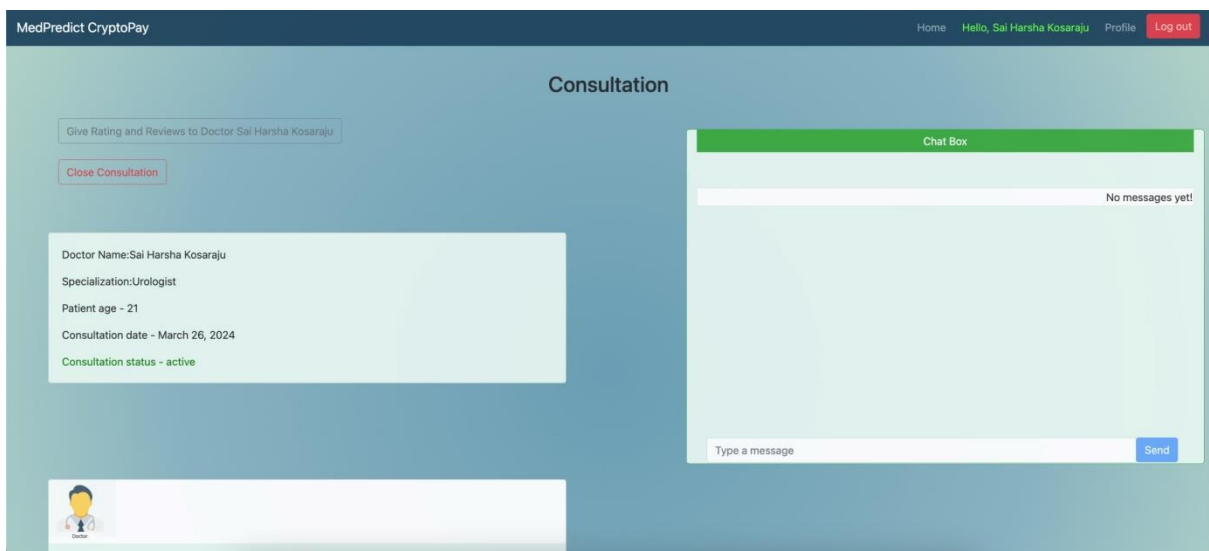


Following patient selection of a doctor, the system seamlessly facilitates crypto currency payment for consultation fees, mirroring efficient administrative transaction handling. Upon successful payment, patients and doctors engage in conversation using a chatbox feature, fostering effective communication and collaboration, reminiscent of administrative communication protocols.

12) On payment page, patient has to connect their respective metamask wallet to proceed with the payment process



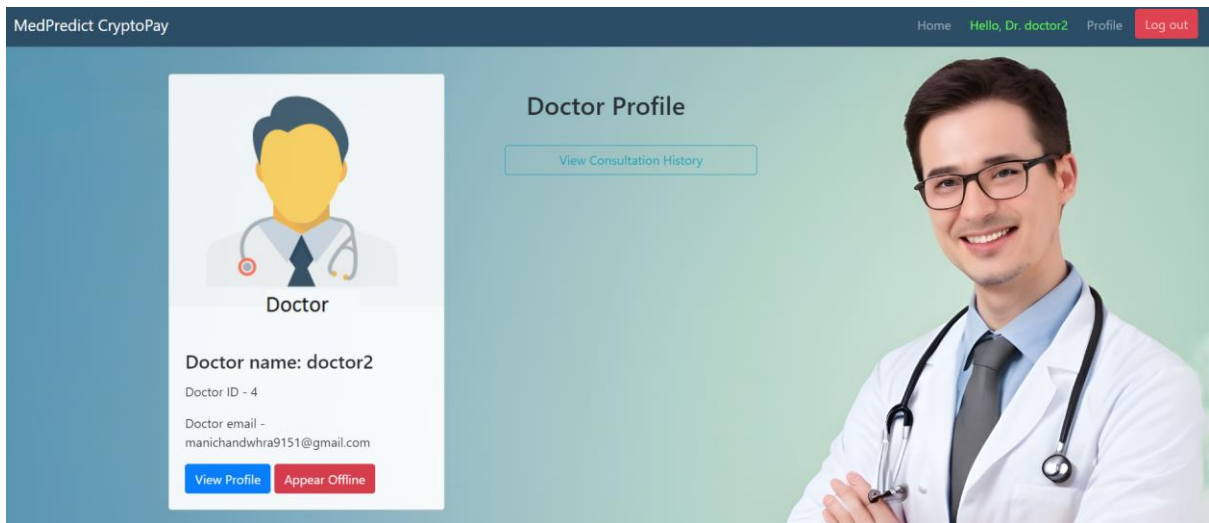
13) After payment, chat window will be accessible for communication between doctor and patient



Doctor functionalities mirror the structured administrative approach, requiring signup/signin processes for authentication. Upon authentication, doctors gain access to patient requests in the consultation history, akin to accessing administrative records for efficient management. Post-payment, doctors view patient symptoms and predicted diseases, enabling informed decision-making and treatment planning, reflecting the organized administrative framework's focus on

data accessibility and functionality.

14)After doctor authentication ,doctor can appear offline/online and can view consultation history



15)Doctor can consult patients

MedPredict CryptoPay

Home Hello, Dr. doctor2 Profile Log out

Consultation History

Patient name	Patient Email	View Patient's profile	Predicted Disease Name	Consultation Date	Resume Consultation
patient1	manichandra91wue1@gmail.com	view profile	Migraine	Dec. 28, 2023	Consult
patient1	manichandra91wue1@gmail.com	view profile	Impetigo	Jan. 7, 2024	Consult
patient1	manichandra91wue1@gmail.com	view profile	Hepatitis E	Jan. 14, 2024	Consult
patient1	manichandra91wue1@gmail.com	view profile	Hepatitis E	Jan. 14, 2024	Consult
patient1	manichandra91wue1@gmail.com	view profile	GERD	Jan. 16, 2024	Consult
patient1	manichandra91wue1@gmail.com	view profile	Hepatitis E	Jan. 16, 2024	Consult

16)Once the payment is done,Doctor can view the payment status as completed in the coinbase payments window

COMMERCE Payments

Payments

Checkouts

Reports

Terms of Service

By accessing or using the cloud products you are agreeing to the [Coinbase Commerce Terms of Service](#).

Search all payments

Date	Description	Status	Amount	Action
Mar 29, 2024 10:02 PM PST	Consultation Fee - Ramya12345	Completed	0.1 USDC \$0.10	
Mar 29, 2024 9:48 PM PST	Consultation Fee - Ramya12345	Completed	0.1 USDC \$0.10	
Mar 29, 2024 9:41 PM PST	Consultation Fee - Ramya12345	Completed	0.1 USDC \$0.10	

Developers

Integrate Commerce API

Use API to programmatically accept cryptocurrency

Third Party Integrations

Integrations guide

Learn more about 3P integrations

Create a payment link

Generate a secure, custom link to receive crypto payments

17) Doctor has all the information regarding the payment done by the patient

COMMERCE

Payments

Checkouts

Reports

Payment amount: \$0.01

Date & time: Mar 07, 2024 9:33 PM PST

Status: Completed

Network: Polygon

Coinbase fee: 0.0001 USDC

Total: 0.0099 USDC

Link to receipt: <https://commerce.coinbase.com/pay/e13baa53-2f...>

View block explorer: <https://polygonscan.com/tx/0x7b4a0b425f9d076b...>

Payment initiated
Mar 07, 2024

Payment in progress
Mar 07, 2024

Payment confirmed
Mar 07, 2024

18) Link to the receipt is available to the doctor in the coinbase payments window

Receipt:

Payment complete

Paid 0.04 USD

MedPredict CryptoPay

From: MATIC balance

Order code: 5V7LK7YP

Network: Polygon

Subtotal: \$0.01 + 0.008958 MATIC

Network fee: \$0.03 + 0.02826 MATIC

Date: 11:03 AM - Mar 07, 2024

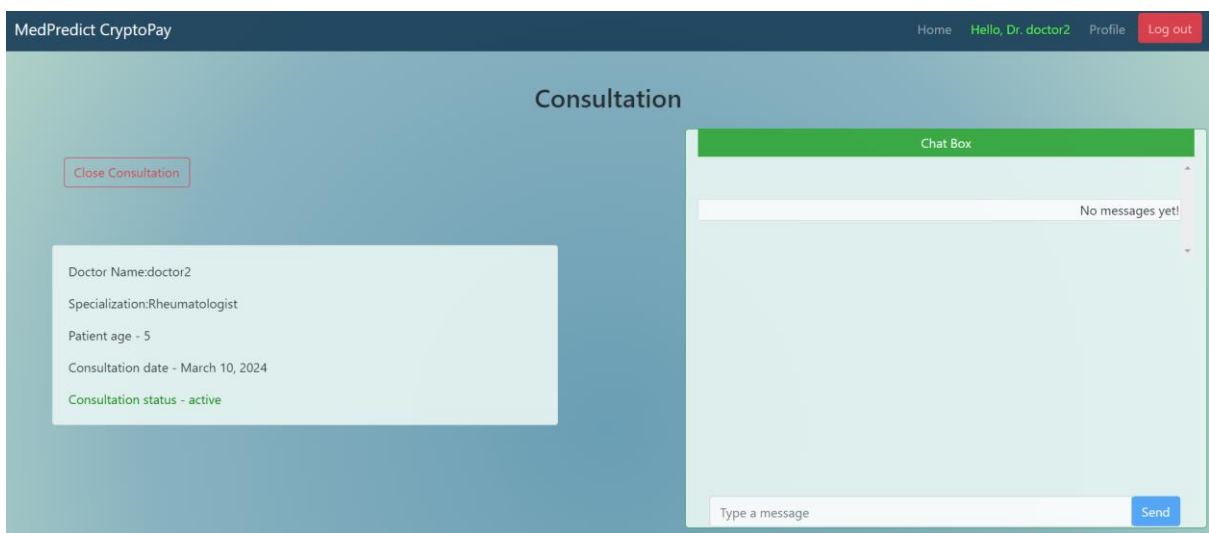
Status: Completed

[View on block explorer](#)

Need Help? Contact sethusndp@gmail.com

The integration of cryptocurrency payment systems and chatbox features into healthcare platforms streamlines administrative processes by facilitating seamless payment for consultation fees after patients select a doctor, while the chatbox fosters effective communication and collaboration reminiscent of administrative protocols. Patients easily convey their symptoms, concerns, and questions to doctors, who can provide timely responses and medical advice, ensuring ongoing support and guidance. Furthermore, the chatbox enables interdisciplinary collaboration among healthcare professionals, promoting knowledge sharing and innovation.

19))Consultation happens through the chat window and after consultation ,doctor can close the consultation.



Front-end Technologies used:Html,Css,JavaScript

Back-end Technologies used : JavaScript,Python,PostgressSql(Database).

Homepage consits of three users: Admin,Doctor,Patient

3.4.1 Code

Admin_url.html

```
{
% extends "basic.html" %
}
{
% load static %
}
{
% block head %
}
<style>
#box
{
padding-left: 15%;
padding-right: 15%;
}
</style>
{% endblock %}
{% block body %}
<br>
<div id="box" class="container mt-5 mb-5">
<h2>Admin - {{ auser.username }}
</h2>
<div class="col" >
<div class="row">
<a id="links" class="btn btn-outline-info btn-block" href="{% url 'admin:index' %}">Manage
user's data</a>
<br>
</div>
<div class="row">
```

```
<button class="btn btn-outline-info btn-block" data-toggle="modal" data-  
target="#myModal2">
```

```
View user's feedback
```

```
</button>
```

```
</div>
```

```
</div>
```

```
<!-- The Modal -->
```

```
<div class="modal fade" id="myModal2">
```

```
<div class="modal-dialog modal-xl">
```

```
<div class="modal-content">
```

```
<!-- Modal Header -->
```

```
<div class="modal-header">
```

```
<h4 class="modal-title">Feedback's</h4>
```

```
<button type="button" class="close" data-dismiss="modal">&times;</button>
```

```
</div>
```

```
<!-- Modal body -->
```

```
<div class="modal-body">
```

```
{% for i in Feedback %}
```

```
<div class="row" style="border-bottom: 1px solid #ccc;">
```

```
<div class="col">
```

```
<p class=""><mark>Date created</mark>: {{i.created}}</p>
```

```
</div>
```

```
<div class="col">
```

```
<p class=""><mark>Feedback</mark>: {{i.feedback}}</p>
```

```
</div>
```

```
<div class="col">
```

```
<p class=""><mark>Sender</mark>: {{i.sender.username}}</p>
```

```
</div>
```

```
</div>
```

```
<br>
```

```
{% endfor %}
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
{% endblock %}
```

SIGNIN.html

```
{% extends "basic.html" %}
```

```
{% load static %}
```

```
{% block head %}
```

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<style>
```

```
body {font-family: Arial, Helvetica, sans-serif;}
```

```
form {border: 3px solid #f1f1f1;}
```

```
#box{
```

```
width:30%;
```

```
height:30%;
```

```
}
```

```
input[type=text], input[type=password] {
```

```
width: 100%;
```

```
padding: 12px 20px;
```

```
margin: 8px 0;
```

```
display: inline-block;
```

```
border: 1px solid #ccc;
```

```
box-sizing: border-box;
```

```
}
```

```
button {
```

```
background-color: #4CAF50;
```

```
color: white;
```

```
padding: 14px 20px;
```



```
margin: 8px 0;
border: none;
cursor: pointer;
width: 100%;
}
button:hover {
opacity: 0.8;
}
.cancelbtn {
width: auto;
padding: 10px 18px;
background-color: #f44336;
}
.imgcontainer {
text-align: center;
margin: 24px 0 12px 0;
}
img.avatar {
width: 40%;
border-radius: 50%;
}
.container {
padding: 16px;
}
span.psw {
float: right;
padding-top: 16px;
}
/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
span.psw {
```

```
display: block;
float: none;
}
.cancelbtn {
width: 100%;
}
}
</style>
{% endblock %}
{% block body %}
<div class="container" id="box" >
<center>
<h2> Admin Signin</h2>
</center>
<form action="sign_in_admin" method="post">
{% csrf_token %}
<div class="imgcontainer">

</div>
<center>
<div>
{% for message in messages %}
<h3 style="color: red;">{{ message }}</h3>
{% endfor %}
</div>
</center>
<br>
<div class="container">
<label for="uname"><b>Username</b></label>
<input type="text" placeholder="Enter Username" name="username" required>
<label for="psw"><b>Password</b></label>
```

```
<input type="password" placeholder="Enter Password" name="password" required>
<button type="submit">Signin</button>
<label>
<input type="checkbox" checked="checked" name="remember"> Remember me
</label>
</div>
</form>
</div>
{% endblock %}
```

Chat_body.html

```
{% load static %}
{% for obj in chat %}
{% if obj.sender == request.user %}
<div class="outgoing_msgs">
<div class="sent_msg">
<span class="time_date"> {{ obj.created }} </span>
<li class="text-right list-group-item">{{ obj.message }} </li>
</div>
</div>
{% else %}
<div class="incoming_msg">
<div class="incoming_msg_img"> 
</div>
<div class="received_msg">
<div class="received_withd_msg">
<span class="time_date"> {{ obj.created }} </span>
<li class="text-left list-group-item">{{ obj.message }} </li>
<span class="time_date"> {{ obj.sender }} </span>
</div>
</div>
```

```
</div>
{% endif %}
{% empty %}
<br><br>
<li class="text-right list-group-item">No messages yet!</li>
<br><br>
{% endfor %}
```

Profile.html

```
{% extends "basic.html" %}
{% load static %}
{% block head %}
{% endblock %}
{% block body %}
<br>
<div class="container mt-3 mb-3">
<center><h2>Doctor Profile</h2>
</center><br>
<div class="row">
<div class="col" >
<div class="card" style="width:350px">

<div class="card-body">
<h4 class="card-title">Doctor name : {{user.doctor.name}}</h4>
<p class="card-text">Doctor ID - {{user.doctor.user_id}}</p>
<p class="card-text">Doctor email - {{user.email}}</p>
<a href="{% url 'dviewprofile' user.username %}" class="btn btn-primary">View Profile</a>
</div>
</div>
</div>
<div class="col" >
```

```
<div class="row">
  <a class="btn btn-outline-info btn-block" href="{% url 'dconsultation_history' %}">view
  consulation history</a><br>
</div>
<div class="row">
  <button class="btn btn-outline-info btn-block" data-toggle="modal" data-target="#myModal-
  feedback">Give feedbacks </button><br>
</div>
<!-- The Modal -->
<div class="modal fade" id="myModal-feedback">
  <div class="modal-dialog modal-xl ">
    <div class="modal-content">
      <!-- Modal Header -->
      <div class="modal-header">
        <h4 class="modal-title">Feedbacks</h4>
        <button type="button" class="close" data-dismiss="modal">&times;</button>
      </div>
      <!-- Modal body -->
      <div class="modal-body">
        <form action="post_feedback" method="POST"> { % csrf_token % }
        <div class="form-group">
          <label for="comment">Give feedback:</label>
          <textarea class="form-control" rows="5" id="feedback" name="feedback"></textarea>
        </div>
      </div>
      <!-- Modal footer -->
      <div class="modal-footer">
        <button id="submit" type="submit" class="btn btn-success" data-dismiss="modal"
        style="color: white;">Submit</button>
      </div>
    </form>
```

```
</div>
</div>
</div>
</div>
</div>
<script>
$(document).ready(function() {
$('#submit').click( function(event){
$.ajax({
url : "{ % url 'post_feedback' % }",
type : "POST",
data : { feedback : $('#feedback').val(),
csrfmiddlewaretoken : $('input[name=csrfmiddlewaretoken']).val()
},
success : function(data){
alert(data);
}
});
});
});
});
</script>
{ % endblock % }
```

CHAPTER 4

RESULTS AND DISCUSSION

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Result

The integration of Medpredict CryptoPay marks a significant leap forward in healthcare technology, revolutionizing diagnostic practices and financial transactions within the medical sector. Through the utilization of machine learning algorithms, the system predicts diseases accurately based on patient-reported symptoms, leading to enhanced diagnostic accuracy and timely medical intervention. Additionally, by incorporating blockchain technology, the platform ensures secure cryptocurrency transactions, bolstering user trust and privacy. Furthermore, the emphasis on personalized healthcare experiences empowers users to select specific doctors, facilitating tailored medical prescriptions and accurate diagnoses. This collaborative approach, combining machine learning predictions with clinical expertise, fosters informed decision-making and customized treatment plans. Moreover, the system's global accessibility promotes inclusivity and equality in healthcare, breaking down geographical barriers and ensuring individuals worldwide can access quality medical services. Overall, Medpredict CryptoPay represents a transformative initiative with the potential to redefine healthcare delivery, accessibility, and patient care on a global scale.

4.1.1 Admin Integration

The Admin can login which can be seen in figure 4.1, and after login the admin can create or view and delete doctors and patient details.



Fig. 4.1 Admin Login

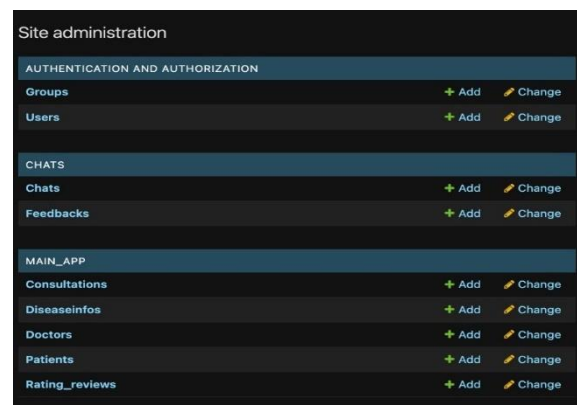
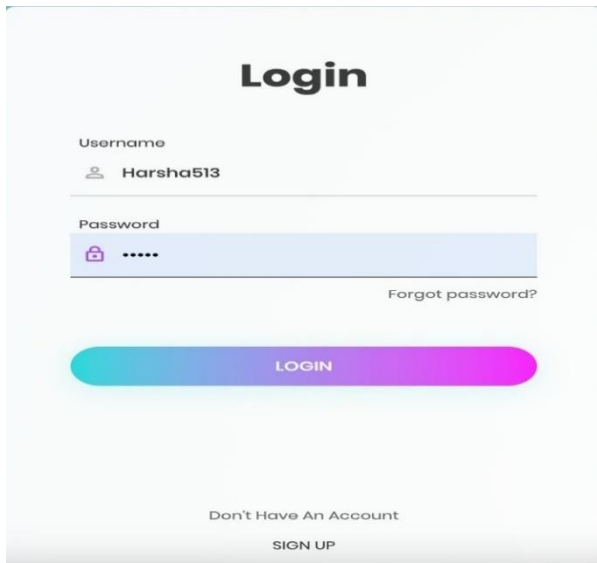


Fig. 4.2 Admin view

4.1.2 Patient Integration

The patient has to signup or signin. After signin the patient have to chose the symptoms and have to submit, and based on the symptoms doctor are suggested. The patient can choose any suggested doctor or any doctor. After selecting the doctor, patient have to pay the consultation fee using crpto currency. After successful payment, patient and doctor can have conversation using chatbox.



Login

Username
Harsha513

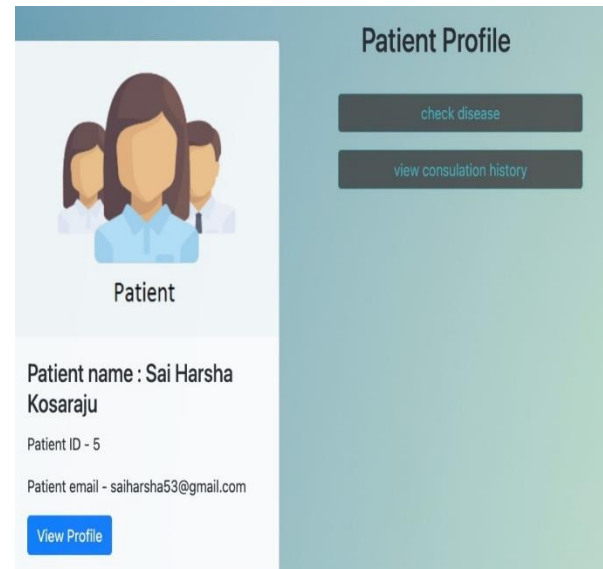
Password
.....

[Forgot password?](#)

LOGIN

Don't Have An Account
SIGN UP

Fig. 4.3 Patient Login



Patient Profile

[check disease](#)

[view consultation history](#)

Patient

Patient name : Sai Harsha Kosaraju

Patient ID - 5

Patient email - saiharsha53@gmail.com

[View Profile](#)

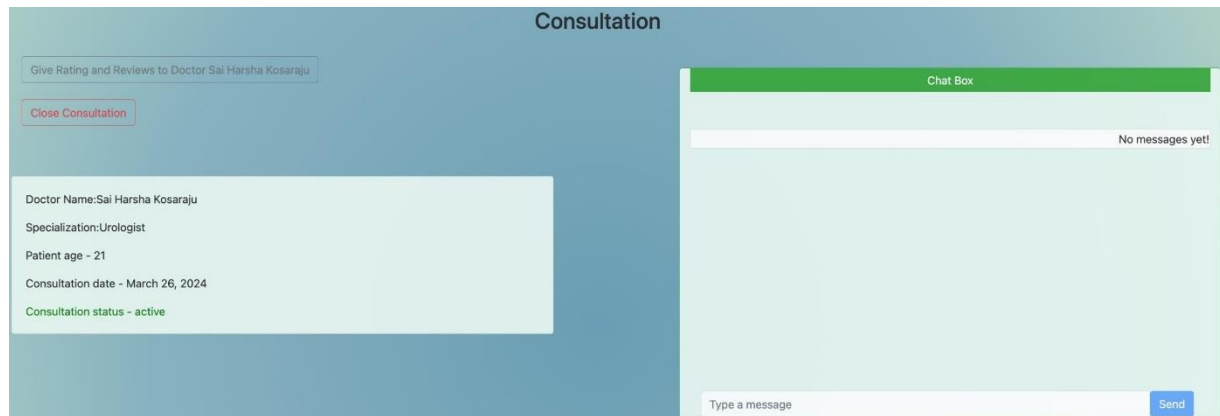
Fig. 4.4 Patient Profile

Consult a Doctor						
Doctor name	Specialization	Email	View profile	Minimum charges	Pay here	Status
Sai Harsha Kosaraju	Urologist	saiharsh4@gmail.com	View profile	\$5.00	Pay here for Sai Harsha Kosaraju	Available

Fig. 4.5 Patient Consulting a Doctor

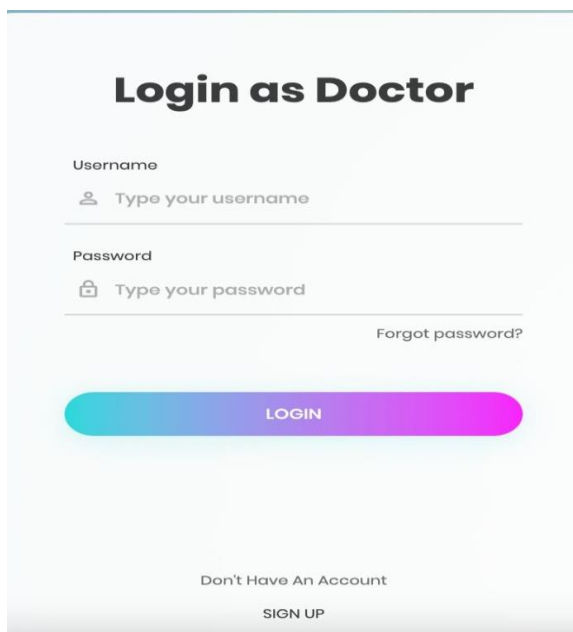
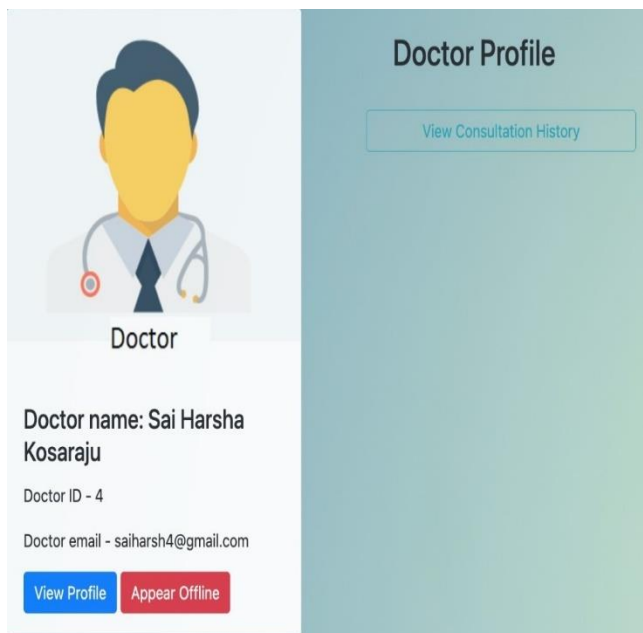
Payment has Succeeded				
Doctor name	Specialization	Email	View profile	Consult
Sai Harsha Kosaraju	Urologist	saiharsh4@gmail.com	View profile	Click to Chat

Fig. 4.6 Patient Payment has Succeeded

**Fig. 4.7** Chat Window

4.1.3 Doctor Integration

Doctor has to signup or signin. After signin doctor can view the patient in the consultation history. If patient pays the fee then doctor can view the patient symptoms and predicted disease and also can have chat with the patient using chat window.

**Fig. 4.8** Doctor Login**Fig. 4.9** Doctor Profile

4.1.4 Payment Integration

In implementing the payment system for our project, Medpredict Cryptopay, we drew inspiration from the approach outlined in the provided link. We utilized Django, a high-level Python web framework, to develop our payment functionality, integrating it with Coinbase, a popular cryptocurrency exchange platform. Firstly, we set up a Django project and created necessary Django apps to handle user authentication, patient-doctor interactions, and payment processing. Following the instructions outlined in the provided link, we configured our Django project to interact with the Coinbase API.

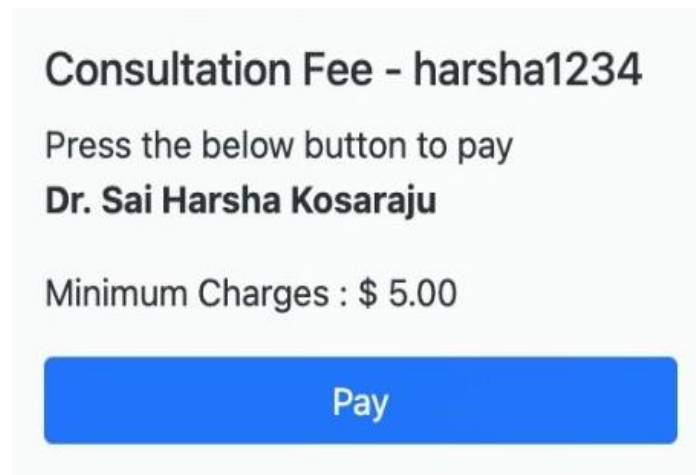


Fig.4.10 Payment Window

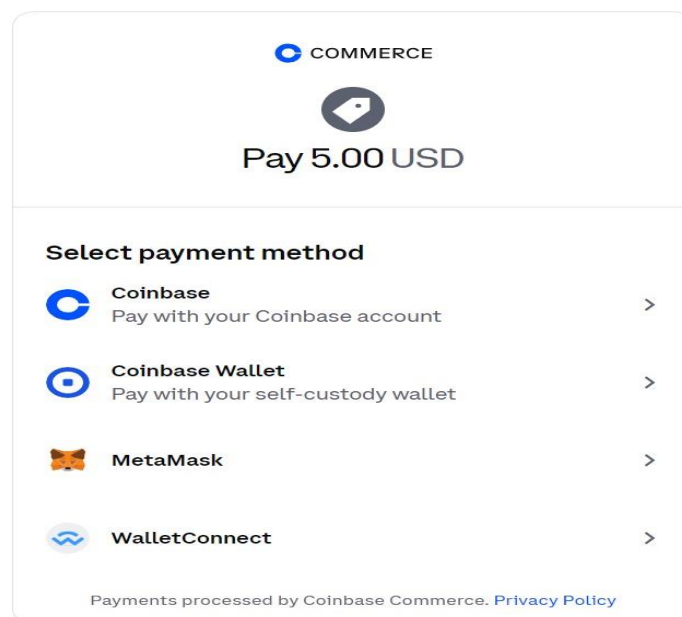


Fig.4.11 Payment Method

We implemented user authentication functionalities, allowing patients and doctors to securely sign up and sign in to their accounts within our system. Once authenticated, patients could select their symptoms, submit them for disease prediction, and choose a doctor for consultation.

For payment processing, we integrated Coinbase's API into our system, enabling patients to pay consultation fees using cryptocurrencies. Patients were provided with a seamless payment experience, with their transactions securely processed through Coinbase's platform.

Upon successful payment, patients and doctors could engage in conversation using a chatbox feature, fostering effective communication and collaboration.

In summary, our payment system implementation for Medpredict CryptoPay leveraged Django for web development and integrated Coinbase's API for cryptocurrency payment processing. This allowed us to provide users with a secure and seamless payment experience within our healthcare platform.

4.1.5. Prediction Integration

The presented system is a Django web application designed to facilitate disease prediction based on symptoms provided by patients. Upon receiving a submission through a form, the system processes the symptoms by converting them into a binary vector representation where each element corresponds to a specific symptom. This vector is then fed into a pre-trained machine learning model, likely trained on a dataset associating symptoms with various diseases. The model predicts the most probable disease based on the input symptoms and calculates a confidence score for the prediction using `predict_proba()`. Additionally, the system recommends a type of doctor for consultation based on the predicted disease.

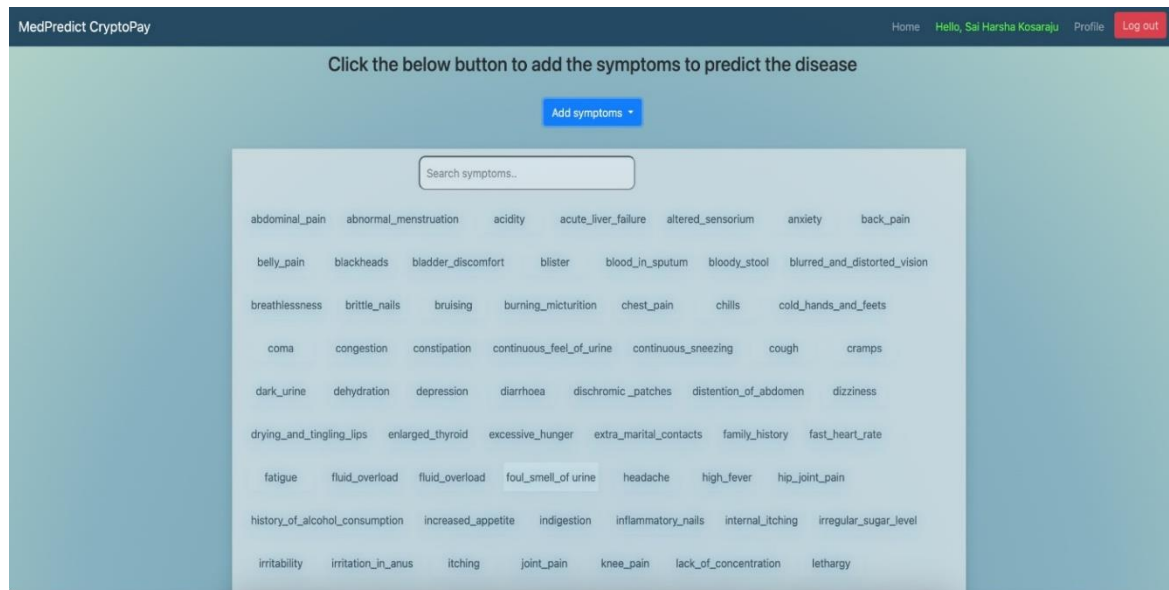


Figure 4.12 User interface-1

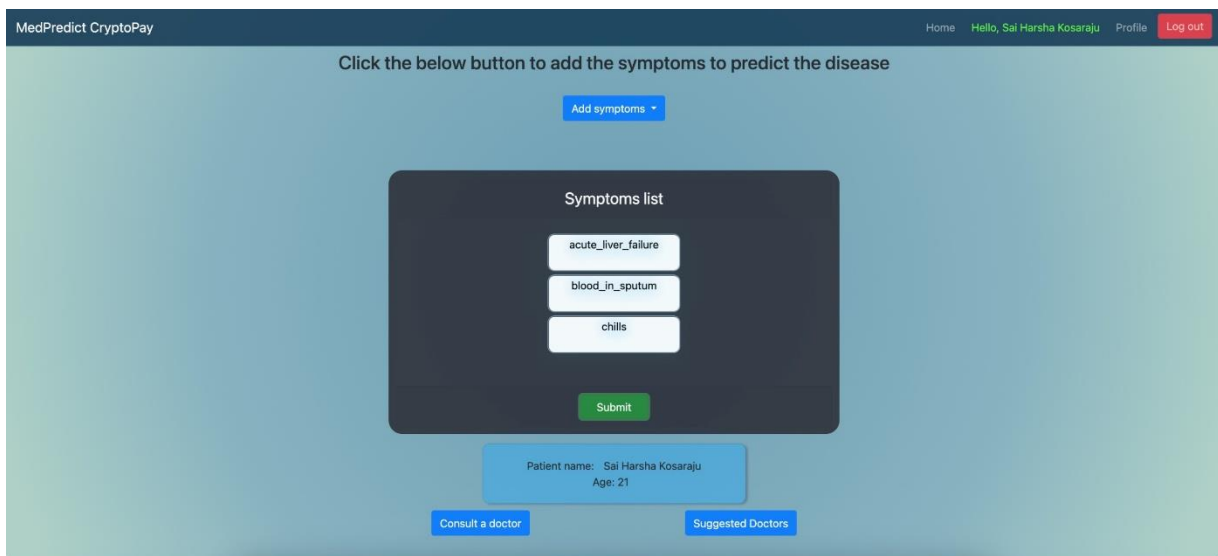


Figure 4.13 User interface-2

4.2 Performance metrics

1. **Rinkal Keniya, vruddhi shah et.al 2020:** They used the Weighted KNN algorithm, achieving an accuracy of 89.5%. In this No blockchain or payment components were involved.
2. **Dhanashri Gujar et.al 2021:** Employed the Decision Tree algorithm with an accuracy of 89.75%. In this No blockchain or payment components were involved.
3. **Venkatesh Rallapalli et.al 2022:** In this no algorithm is used, but blockchain technology was utilized, with no mention of machine learning or payment components.
4. **Ramandeep Singh Sethi et.al 2023:** Utilized the Naïve Bayes algorithm with an accuracy of 90.34%. In this No blockchain or payment components were involved.
5. **Rahul Deo Sah1 et.al 2021:** Utilized the SubSpace KNN algorithm with an accuracy of 90.89%. In this No blockchain or payment components were involved.
6. **Our 2024:** Presenting a project in 2024, using the MultiNomial Naïve Bayes algorithm with an accuracy of 91.5%. we employed both machine learning and blockchain technology, and integrated payment components into our project.

S.No	Authors and Journal Name and year of publication	Algorithm Used	Accuracy	Machine Learning	Blockchain	Payment
1	[4]Rinkal Keniya, vruddhi shah <i>et.al</i> 2020	Weighted KNN	89.5%	✓	✗	✗
2	[7]Dhanashri Gujar <i>et.al</i> 2021	Decision tree	89.75%	✓	✗	✗
3	[5]Venkatesh Rallapalli <i>et.al</i> 2022	-	-	✗	✓	✗
4	[8]Ramandeep Singh Sethi <i>et.al</i> 2023	Naïve Bayes	90.34%	✓	✗	✗
5	[6]Rahul Deo Sah1 <i>et.al</i> 2021	SubSpace KNN	90.89%	✓	✗	✗
6	Our 2024	MultiNomial Naïve Bayes	91.5%	✓	✓	✓

Table 4.1 Performance metrics

CHAPTER 5

CONCLUSION

CHAPTER 5

CONCLUSION

5.1 Conclusion

Our system, Med Predict Crypto Pay, is a groundbreaking advancement in disease prediction. It uses advanced machine learning to minimize reliance on human expertise, reducing misdiagnosis risks and enabling timely health issue identification. With blockchain-powered secure payments, patients can easily handle transactions, while doctors gain precise disease predictions for prompt treatment. This innovative tech fusion transforms diagnostics and creates a globally accessible healthcare platform, emphasizing early detection and personalized care. It represents a pioneering leap forward in disease prediction, harnessing advanced machine learning to mitigate reliance on human expertise, thereby minimizing misdiagnosis risks and facilitating timely identification of health issues. By integrating blockchain-powered secure payments, the system ensures seamless transactions for patients, while equipping doctors with precise disease predictions to expedite treatment. This innovative fusion of technology transforms diagnostic practices, establishing a globally accessible healthcare platform. Emphasizing early detection and personalized care, It prioritizes proactive healthcare interventions, ultimately enhancing patient outcomes and fostering a more efficient healthcare ecosystem. Through its groundbreaking approach, this system not only revolutionizes diagnostics but also sets a new standard for healthcare accessibility and effectiveness in the digital era.

5.2 Future Enhancement

1. Enhanced Disease Prediction Models:

- Broaden coverage and accuracy through advanced machine learning algorithms.
- Incorporate diverse datasets and genetic/biomarker data for precision.

2. Integration of Emerging Technologies:

- Utilize AI with IoT devices and wearables for real-time health monitoring.
- Explore immersive healthcare experiences with AR/VR technologies.

3. Continuous Improvement of Blockchain Payment System:

- Strengthen security protocols for data privacy and compliance.
- Streamline transactions and reduce fees for widespread adoption.

4. Collaborative Research and Development:

- Forge partnerships with healthcare institutions and technology companies.

Conduct clinical validation studies for predictive models.

REFERENCES

REFERENCES

- [1].Bathula A, Muhuri S, Gupta SK, Merugu S. Secure certificate sharing based on Blockchain framework for online education. Multimedia Tools and Applications. 2023 May;82(11):16479-500.
- [2].Bathula A, Merugu S, Skandha SS. Academic Projects on Certification Management Using Blockchain-A Review. In2022 International Conference on Recent Trends in Microelectronics, Automation, Computing and Communications Systems (ICMACC) 2022 Dec 28 (pp. 1-6). IEEE.
- [3]. Akinode, J.L. and Oloruntoba, S.A., 2017. Design and implementation of a patient appointment and scheduling system. Department of Computer Science, Federal Polytechnic Ilaro Nigeria.
- [4].Rinkal Keniya, vruddhi shah Sukarno, Karmila Sari. "The use of cryptocurrency as a Payment Instrument." In 3rd International Conference on Law and Governance (ICLAVE 2020), pp. 366-370. Atlantis Press, 2020
- [5]. Bezovski, Zlatko, Venkatesh Rallapalli , Ljupco Davcev, and Mila Mitreva. "Current adoption state of cryptocurrencies as an electronic payment method." Management Reseach and Practice 13, no. 1 (2021): 44-50.
- [6]. Rahul Deo Sah1, Deepthi, Y., Kalyan, K.P., Vyas, M., Radhika, K., Babu, D.K. and Krishna Rao, N.V., 2021. Disease prediction based on symptoms using machine learning. In Energy Systems, Drives and Automations
- [7]. Dhanashri Gujar S, S. A., Bhat, S., K, S. V., Karnik, S., & M, N.. (2021). <i>Disease Prediction Chatbot</i>. <https://doi.org/10.32628/CSEIT2173172>
- [8]. Ramandeep Singh Sethi, Rahman, Md Atikur, Tania Ahmed Nipa, and Md Assaduzzaman. "Predicting disease from several symptoms using machine learning approach." International Research Journal of Engineering and Technology (IRJET) 10, no. 7 (2023): 836-841.
- [9]. Muthu, BalaAnand, C. B. Sivaparthipan, Gunasekaran Manogaran, Revathi Sundarasekar. "IOT based sensor for diseases prediction and symptom analysis in healthcare sector." Peer-to-peer networking and applications 13, no. 6 (2020): 2123.

APPENDIX

Index.html

```
<div class="limiter">
<div class="container-login100">
<div class="wrap-login100 p-l-55 p-r-55 p-t-65 p-b-54">
<form class="login100-form validate-form" action="sign_in_doctor" method="POST" >
{% csrf_token %}
<span class="login100-form-title p-b-49">
Login as Doctor
</span>
<div class="wrap-input100 validate-input m-b-23" data-validate = "Username is required">
<span class="label-input100">Username</span>
<input class="input100" type="text" name="username" placeholder="Type your username"
required>
<span class="focus-input100" data-symbol="">□</span>
</div>
<div class="wrap-input100 validate-input" data-validate="Password is required">
<span class="label-input100">Password</span>
<input class="input100" type="password" name="password" placeholder="Type your
password" required>
<span class="focus-input100" data-symbol="">□</span>
</div>
<div class="text-right p-t-8 p-b-31">
<a href="#">
Forgot password?
</a>
</div>
<center>
<div>
```

```
{% for message in messages %}
<h3 style="color: red;">{{ message }}</h3>
{% endfor %}
</div>
</center>
<br>
<div class="container-login100-form-btn">
<div class="wrap-login100-form-btn">
<div class="login100-form-bgbtn"></div>
<button class="login100-form-btn" type="submit">
Login
</button>
</div>
</div>
<div class="flex-col-c p-t-155">
<span class="txt1 p-b-17">
Don't Have An Account
</span>
<a href="signup_doctor" class="txt2">
Sign Up
</a>
</div>
</form>
</div>
</div>
</div>
<div id="dropDownSelect1"></div>
Basic.html
<div class="container-fluid">
<h1>PREDICO</h1>
```

```
<h6>Be your own doctor </h6>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent">
<span class="navbar-toggler-icon"></span>
</button>
</div>
<nav class="navbar navbar-expand-lg navbar-dark" style="background-color: #1b405a;">
<a class="navbar-brand" href="{ % url 'home' % }">Predico </a>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav ml-auto">
<li class="nav-item">
<a class="nav-link" href="{ % url 'home' % }">Home</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{ % url 'home' % }">About</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{ % url 'home' % }">Contact</a>
</li>
{ % if user.is_authenticated % }
{ % if user.patient % }
<li class="nav-item ml-2">
<li class="nav-link" href="{ % url 'patient_ui' % }" style="color: rgb(91, 252, 50);">Hello,
{ {user.patient.name} }</li>
</li>
<li class="nav-item ml-2">
<a class="nav-link" href="{ % url 'patient_ui' % }">Profile</a>
</li>
{ % endif % }
{ % if user.doctor % }
```

```
<li class="nav-item ml-2">
<li class="nav-link" href="{% url 'doctor_ui' %}" style="color: rgb(91, 252, 50);">Hello, Dr.
{{ user.doctor.name }}</li>
</li>
<li class="nav-item ml-2">
<a class="nav-link" href="{% url 'doctor_ui' %}">Profile</a>
</li>
{% endif %}
{% if user.is_superuser %}
<li class="nav-item ml-2">
<li class="nav-link" href="{% url 'admin_ui' %}" style="color: rgb(91, 252, 50);">Admin:
{{ user.username }}</li>
</li>
<li class="nav-item ml-2">
<a class="nav-link" href="{% url 'admin_ui' %}">Profile</a>
</li>
{% endif %}
<li class="nav-item ml-2">
<button class="btn btn-danger btn-xs" data-toggle="modal" data-target="#logout-modal"
style="color: rgb(247, 190, 188);">Log out</button>
</li>
<!-- Small modal -->
<div class="modal fade" id="logout-modal" tabindex="-1" role="dialog" aria-hidden="true">
<div class="modal-dialog modal-sm">
<div class="modal-content">
<div class="modal-header"><h4>Logout <i class="fa fa-lock"></i></h4></div>
<div class="modal-body"><i class="fa fa-question-circle"></i><span style="color: rgb(42,
187, 6);">{{ user.patient.name }}</span>, Are you sure you want to log-off?</div>
<div class="modal-footer"><a href="{% url 'logout' %}" class="btn btn-primary btn-
block">Logout</a></div>
```

```
</div>
</div>
</div>
{% else %}
<li class="nav-item ml-2">
<a class="nav-link" data-toggle="modal" data-target=".bd-example-modal-lg2">Signup</a>
</li>
<li class="nav-item ml-1">
<a class="nav-link" data-toggle="modal" data-target=".bd-example-modal-lg">Signin</a>
</li>
{% endif %}
</ul>
</div>
</nav>
<div class="modal fade bd-example-modal-lg" tabindex="-1" role="dialog" aria-
labelledby="myLargeModalLabel" aria-hidden="true">
<div class="modal-dialog modal-lg">
<div class="modal-content">
<center>
<h1>Sign-In As</h1>
</center>
<br><br>
<center>
<form>
<div class="container mt-5 mb-5">
<div class="row">
<div class="col">
<a href="{% url 'sign_in_admin' %}">

<div>
```



```
<div class="col">
<a href="{% url 'sign_in_doctor' %}">

</a>
</div>
<div class="col">
<a href="{% url 'sign_in_patient' %}">

</a>
</div>
</div>
</form>
</center>
</div>
</div>
</div>
{% block body %}
{% endblock %}
</div>
```

Patient.html

```
{% block body %}
<div class="limiter">
<div class="container-login100">
<div class="wrap-login100 p-l-55 p-r-55 p-t-65 p-b-54">
<form class="login100-form validate-form" action="sign_in_patient" method="POST" >
{% csrf_token %}
<span class="login100-form-title p-b-49">
Login
</span>
```

```
<div class="wrap-input100 validate-input m-b-23" data-validate = "Username is required">
<span class="label-input100">Username</span>
<input class="input100" type="text" name="username" placeholder="Type your username"
required>
<span class="focus-input100" data-symbol="&#xf206;"></span>
</div>

<div class="wrap-input100 validate-input" data-validate="Password is required">
<span class="label-input100">Password</span>
<input class="input100" type="password" name="password" placeholder="Type your
password" required>
<span class="focus-input100" data-symbol="&#xf190;"></span>
</div>

<div class="text-right p-t-8 p-b-31">
<a href="#">
Forgot password?
</a>
</div>

<center>
<div>
{ % for message in messages % }
<h3 style="color: red;">{ { message } }</h3>
{ % endfor % }
</div>
</center>

<br>
<div class="container-login100-form-btn">
<div class="wrap-login100-form-btn">
<div class="login100-form-bgbtn"></div>
<button class="login100-form-btn" type="submit">
Login
```

```
</button>
</div>
</div>
<div class="flex-col-c p-t-155">
  <span class="txt1 p-b-17">
    Don't Have An Account
  </span>
  <a href="signup_patient" class="txt2">
    Sign Up
  </a>
</div>
</form>
</div>
</div>
</div>
<div id="dropDownSelect1"></div>
```

Settings.py

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'v3v5vfn0xxjtmb=eoawoiw$5br4g0r&jy_l39995h_93l+-z5'
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition
INSTALLED_APPS = [
    'chats.apps.ChatsConfig',
```

```
'accounts.apps.AccountsConfig',
'main_app.apps.MainAppConfig',
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions'
'django.contrib.messages',
'django.contrib.staticfiles',
]
MIDDLEWARE = [
'django.middleware.security.SecurityMiddleware',
'django.contrib.sessions.middleware.SessionMiddleware',
'django.middleware.common.CommonMiddleware',
'django.middleware.csrf.CsrfViewMiddleware',
'django.contrib.auth.middleware.AuthenticationMiddleware',
'django.contrib.messages.middleware.MessageMiddleware',
'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
ROOT_URLCONF = 'disease_prediction.urls'
TEMPLATES = [
{
'BACKEND': 'django.template.backends.django.DjangoTemplates',
'DIRS': [os.path.join(BASE_DIR, 'templates')],
'APP_DIRS': True,
'OPTIONS': {
'context_processors': [
'django.template.context_processors.debug',
'django.template.context_processors.request',
'django.contrib.auth.context_processors.auth',
'django.contrib.messages.context_processors.messages',
```

],

},

},

]

WSGI_APPLICATION = 'disease_prediction.wsgi.application'

CMRCET

B. Tech (CSE)

Page No 33

Medpredict CryptoPay

Database

<https://docs.djangoproject.com/en/2.2/ref/settings/#databases>

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.postgresql',

'NAME': 'predico',

'USER': 'postgres',

'PASSWORD': '1234',

'HOST': 'localhost'

}

}

Password validation

<https://docs.djangoproject.com/en/2.2/ref/settings/#auth-password-validators>

AUTH_PASSWORD_VALIDATORS = [

{

'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',

},

{

'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',

},

{

'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',

```
{
'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
},
]
LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
def doctor_ui(request):
if request.method == 'GET':
doctorid = request.session['doctorusername']
duser = User.objects.get(username=doctorid)
return render(request,'doctor/doctor_ui/profile.html',{'duser':duser})
def dviewprofile(request, doctorusername):
if request.method == 'GET':
duser = User.objects.get(username=doctorusername)
r = rating_review.objects.filter(doctor=duser.doctor)
return render(request,'doctor/view_profile/view_profile.html', {"duser":duser, "rate":r} )
def consult_a_doctor(request):
if request.method == 'GET':
doctortype = request.session['doctortype']
print(doctortype)
dobj = doctor.objects.all()
return render(request,'patient/consult_a_doctor/consult_a_doctor.html',{'dobj':dobj})
def make_consultation(request, doctorusername):
if request.method == 'POST':
patientusername = request.session['patientusername']
puser = User.objects.get(username=patientusername)
patient_obj = puser.patient
#doctorusername = request.session['doctorusername']
duser = User.objects.get(username=doctorusername)
status = "active"
```

PUBLICATION



PAPER ACCEPTED

[About Us](#) | [Aim & Scope](#) | [Check Paper Status](#)



Dear Author/Research Scholar,

I am pleased to inform you that IJRASET would like to publish your manuscript **“MEDPREDICT CRYPTOPAY”** in Volume 12 Issue III March 2024. Acceptance for the paper is sent on the recommendation of experts after peer review.