TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

. Regarding the different causes of heart disease, analyzing what causes heart disease has become mainstream nowadays.After an in-depth understanding of data analysis and machine learning-related knowledge, data analysis and data training are carried out on a dataset.This project proposes a prediction model to predict whether a people have a heart disease or not and visualize it using a dashboard.

1.2 PURPOSE

With the ever increasing population of the world, diseases and their possibilities are also increasing at an alarming rate. As time passes by, diagnosing diseases and providing appropriate treatment at the right time has become quite a challenge. Heart diseases, for one, have been a major cause of death worldwide. Finding an efficient way to predict the chances of a heart failure is indeed a need.

CHAPTER-2

REQUIREMENTS

2.1 EXISTING PROBLEM

Existing solutions are found to have,

1. Inefficient and inaccurate heart disease prediction system
2. No proper assistance of medical professionals in evaluating a patient's heart disease based on the clinical data of the patient.

2.2 REFERENCES

1 A. Ed-Daoudy and K. Maalmi, "Real-time machine learning for early detection of heart disease using big data approach," 2019 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS), 2019, pp. 1-5, doi: 10.1109/WITS.2019.8723839.

2 K. G. Dinesh, K. Arumugaraj, K. D. Santhosh and V. Mareeswari, "Prediction of Cardiovascular Disease Using Machine Learning Algorithms," 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT), 2018, pp. 1-7, doi: 10.1109/ICCTCT.2018.8550857.

3 P. Motarwar, A. Duraphe, G. Suganya and M. Premalatha, "Cognitive Approach for Heart Disease Prediction using Machine Learning," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-5, doi: 10.1109/ic-ETITE47903.2020.242.

4 A. Rahim, Y. Rasheed, F. Azam, M. W. Anwar, M. A. Rahim and A. W. Muzaffar, "An Integrated Machine Learning Framework for Effective Prediction of Cardiovascular Diseases," in IEEE Access, vol. 9, pp. 106575-106588, 2021, doi: 10.1109/ACCESS.2021.3098688.

5 R. Indrakumari, T. Poongodi, Soumya Ranjan Jena,Heart Disease Prediction using Exploratory Data Analysis,Procedia Computer Science,Volume 173,2020,Pages 130-139,ISSN 1877-0509,https://doi.org/10.1016/j.procs.2020.06.017.

6 Renugadevi, G & Priya, G & Sankari, B & Rajamanickam, Gowthamani. (2021). Predicting heart disease using hybrid machine learning model. Journal of Physics: Conference Series. 1916. 012208. 10.1088/1742-6596/1916/1/012208.

7 Xinyu Zhang. 2021. Using Data Visualization to Analyze the Correlation of Heart Disease Triggers and Using Machine Learning to Predict Heart Disease. In 2021 3rd International Conference on Intelligent Medicine and Image Processing (IMIP

'21). Association for Computing Machinery, New York, NY, USA, 127–132. https://doi.org/10.1145/3468945.3468966

8 Habib, Sumaya, Moin, Maisha Binte, Aziz, Sujana. 2018-12.Heart failure risk prediction and medicine recommendation system using exploratory analysis and big data analytics.BRAC University http://hdl.handle.net/10361/11446

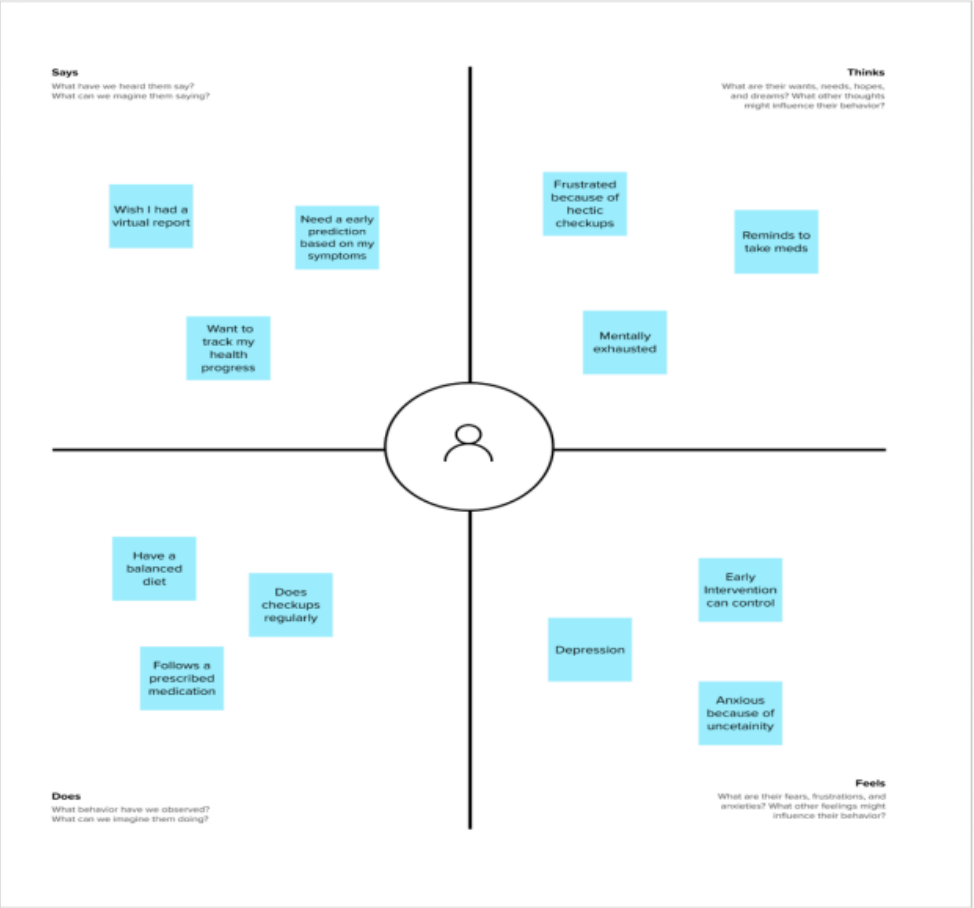
2.3 PROBLEM STATEMENT DEFINITION

Day by day the cases of heart diseases are increasing at a rapid rate and it's very important and concerning to predict any such diseases beforehand. This diagnosis is a difficult task i.e. it should be performed precisely and efficiently.

CHAPTER 3

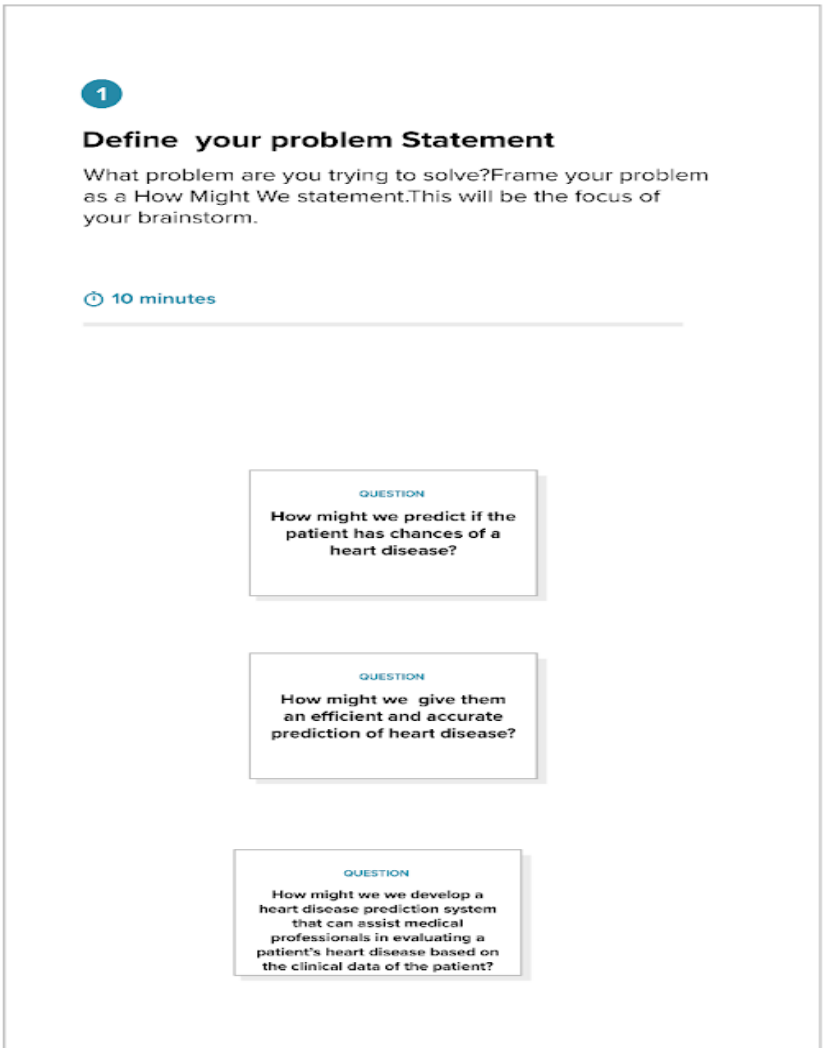
IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

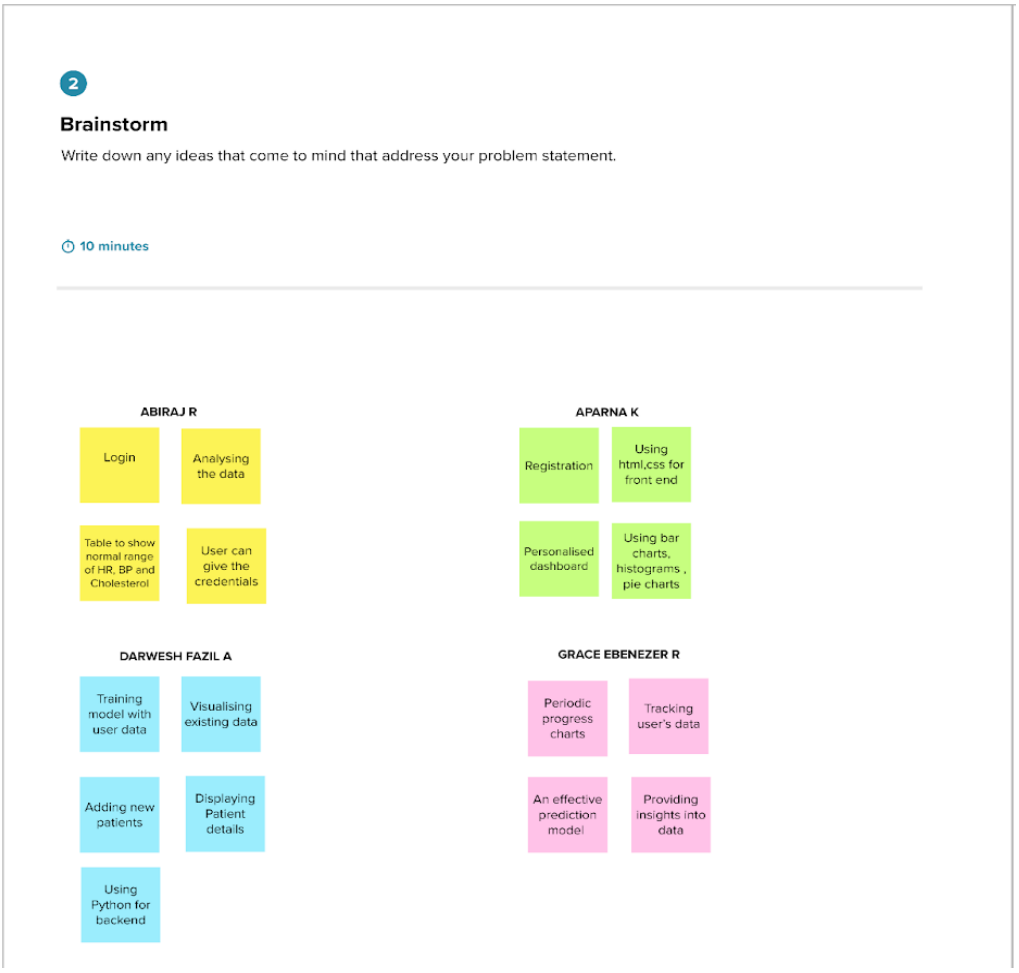


3.2 IDEATION & BRAINSTORMING

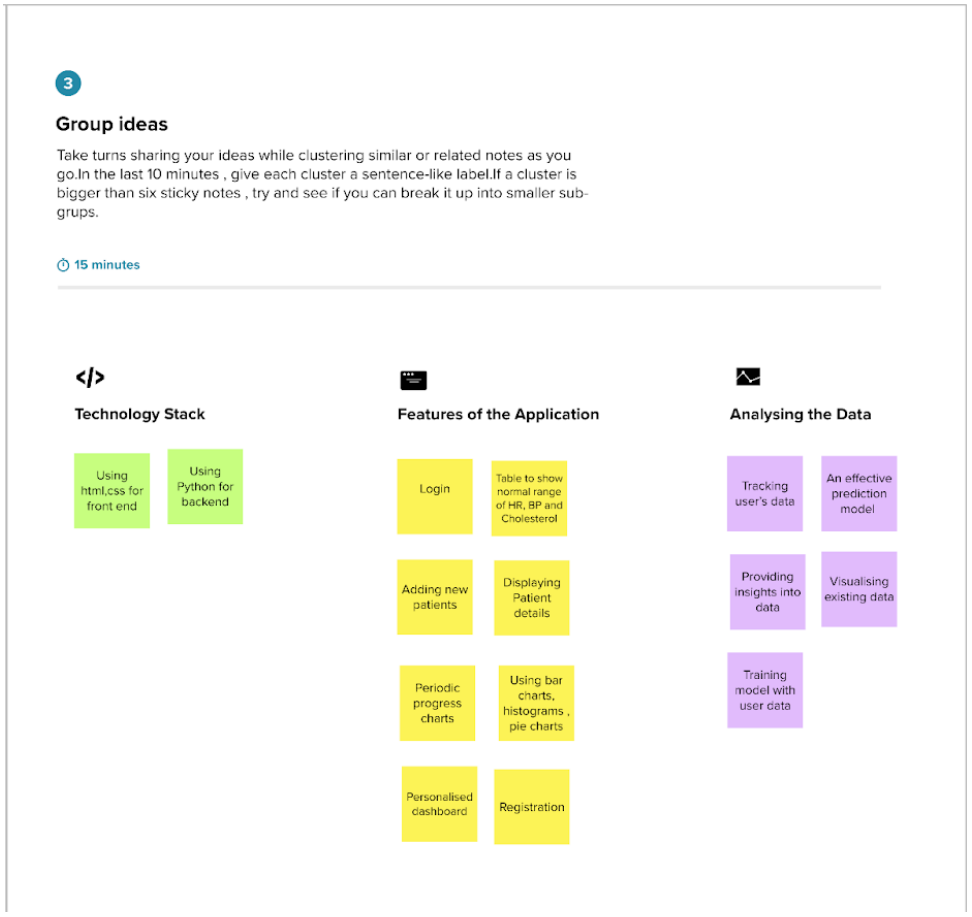
Step-1: Team Gathering and Select the Problem Statement



Step-2: Brainstorm, Idea Listing



Step-3: Grouping ideas



3.3 PROPOSED SOLUTION

3.3.1 Problem Statement

Risk because of heart disease is increasing throughout the world.Many approaches were suggested for prediction of heart disease, but none of them were satisfactory in clinical terms.

3.3.2 Idea/Solution Description

A web application which is able to predict the presence and state of heart diseases using the patients' readings and also provide an analysis, overview of their health over a period of time.

3.3.3 Novelty/Uniqueness

When each patient's readings are provided for prediction and analysis, the model gets trained with the new data, resulting in more accuracy. The patient is able to view their health progress with the help of visualization tools.

3.3.4 Social Impact/ Customer Satisfaction

i) Helps to check whether the patient is likely to be diagnosed with any cardiovascular heart diseases based on their medical attributes.

ii) Reduce in cost and manual interventions
iii) Auto suggestion of methods to take

3.3.5 Business Model

i) Monthly/Yearly Subscriptions

ii) Analysis and prediction fee
iii) Online sessions with medical staffs

3.3.6 Scalability of the solution

i) More the prediction done, more the accuracy

ii) Can be used in any platform since it is a web app
iii) Can be provided as API, to extend as various forms of applications
iv) Scalable three-tier architecture

3.4 PROBLEM SOLUTION FIT

CUSTOMER PROBLEM STATEMENT:

ABSTRACT :

Risk because of heart disease is increasing throughout the world. According to the World Health Organisation report, the number of deaths because of heart disease is drastically increasing as compared to other diseases. Multiple factors are responsible for causing heart-related issues. Many approaches were suggested for prediction of heart disease, but none of them were satisfactory in clinical terms. Day by day the cases of heart diseases are increasing at a rapid rate and it's very Important and concerning to predict any such diseases beforehand. This diagnosis is a difficult task i.e. it should be performed precisely and efficiently.

CHAPTER-4

REQUIREMENTS ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

i) User Registration

ii) User Login

iii) Personal Details Form

iv) Periodic progress charts

v) Visualizing data

vi) Generating report

4.2 NON-FUNCTIONAL REQUIREMENTS

i) Usability

ii) Security

iii) Reliability

iv) Performance

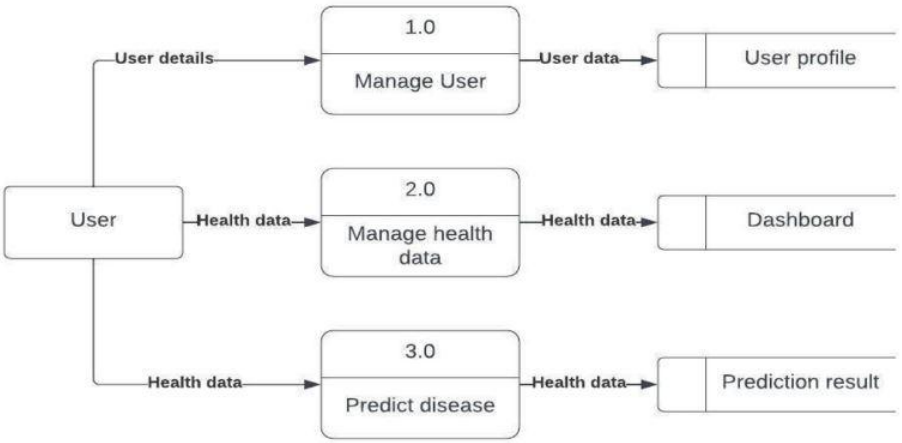
v) Availability

vi) Scalability

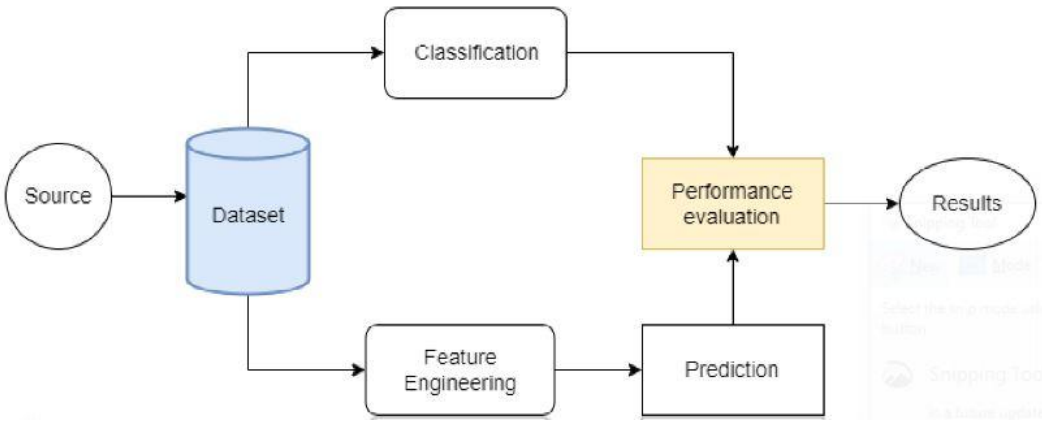
CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

i) **USER STORY NUMBER** : USN-1

USER TYPE : Customer(Web User)

FUNCTIONAL REQUIREMENT : Registration

USER STORY : I may sign up for the programme as a user by providing my email address , a password, and a password confirmation.

ACCEPTANCE CRITERIA : I can access my dashboard or account

PRIORITY : High

RELEASE : Sprint-1

ii) **USER STORY NUMBER** : USN-2

USER TYPE : Customer(Web User)

FUNCTIONAL REQUIREMENT : Registration

USER STORY : When I register for the application as a user, I will get a confirmation email.

ACCEPTANCE CRITERIA : I can get a confirmation email and confirm it.

PRIORITY : High

RELEASE : Sprint-1

iii) **USER STORY NUMBER** : USN-3

USER TYPE : Customer(Web User)

FUNCTIONAL REQUIREMENT : Login

USER STORY : I may access the application as a user by providing my email address and password.

ACCEPTANCE CRITERIA : When I logged in,I can view my account

dashboard.

PRIORITY : High

RELEASE : Sprint-1

iv) **USER STORY NUMBER** : USN-4

USER TYPE : Customer(Web User)

FUNCTIONAL REQUIREMENT : Dashboard

USER STORY : The user can examine his or her accurate illness prognosis and comprehensive medical analysis.

ACCEPTANCE CRITERIA : My medical analysis is displayed in the dashboard.

PRIORITY : High

RELEASE : Sprint-2

v) **USER STORY NUMBER** : USN-5

USER TYPE : Customer(Web User)

FUNCTIONAL REQUIREMENT : Dashboard

USER STORY : The user may examine their whole medical analysis and illness prediction accuracy.

ACCEPTANCE CRITERIA : In the dashboard , I can see the accuracy of heart illness.

PRIORITY : High

RELEASE : Sprint-2

vi) **USER STORY NUMBER** : USN-6

USER TYPE : Administrator

	FUNCTIONAL REQUIREMENT	**:**	User Profile
	USER STORY	**:**	He or She can change a user's health information as an admin.
	ACCEPTANCE CRITERIA	**:**	I can see my current health information.
	PRIORITY	**:**	High
	RELEASE	**:**	Sprint-3
vii)	**USER STORY NUMBER**	**:**	USN-7
	USER TYPE	**:**	Administrator
	FUNCTIONAL REQUIREMENT	**:**	User Profile
	USER STORY	**:**	The admin has the power to add and remove users.
	ACCEPTANCE CRITERIA	**:**	When I logged in , I can view my account and dashboard.
	PRIORITY	**:**	High
	RELEASE	**:**	Sprint-3
viii)	**USER STORY NUMBER**	**:**	USN-8
	USER TYPE	**:**	Administrator
	FUNCTIONAL REQUIREMENT	**:**	User Profile
	USER STORY	**:**	He or She can control user information as an admin.
	ACCEPTANCE CRITERIA	**:**	I have access to my own structured data.
	PRIORITY	**:**	High
	RELEASE	**:**	Sprint-4

CHAPTER 6

PROJECT PLANNING & SCHEDULING

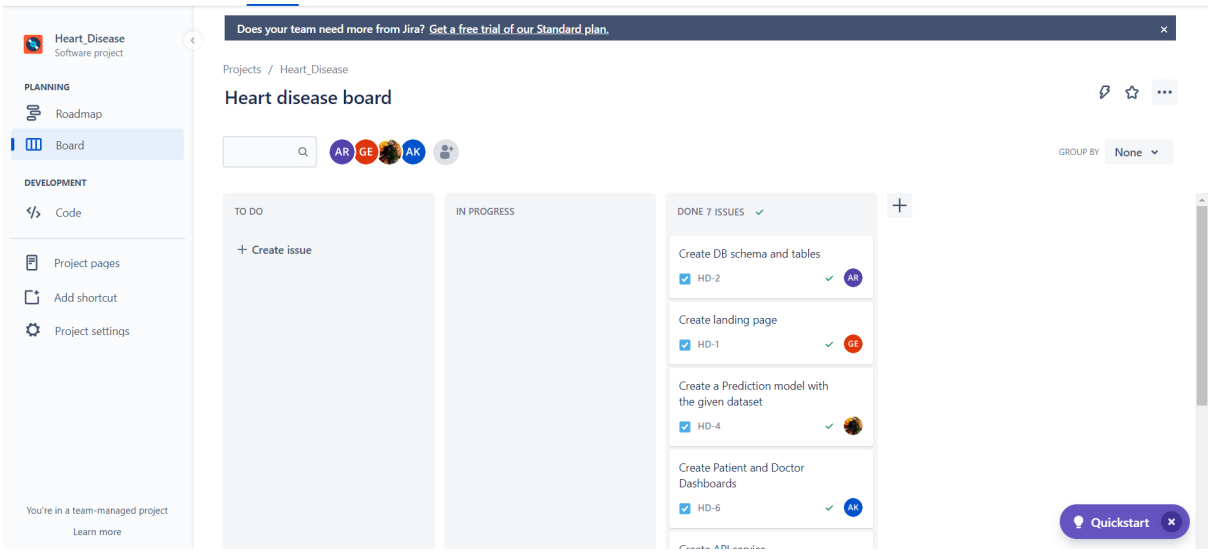
6.1 SPRINT PLANNING & ESTIMATION

SPRINT	FUNCTIONAL REQUIREMENT	USER STORY NUMBER	USER STORY	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint - 1	UI	USN-1	Authentication	5	Low	Aparna, Grace Ebenezer
Sprint-1	UI	USN-2	Doctor form, Patient form	15	High	Grace Ebenezer, Aparna
Sprint-2	Dashboard	USN-3	Homepage for Doctor and Patient	10	Medium	Grace Ebenezer, Aparna
Sprint-2	Classification Model	USN-4	Model for Classifying and Predicting heart diseases	10	Medium	Darwesh Fazil,Abiraj
Sprint-3	Database	USN-5	Table Creation and DB connectivity	5	Low	Darwesh Fazil,Abiraj
Sprint-3	Data API	USN-6	Backend interface for providing data	15	High	Darwesh Fazil,Abiraj
Sprint-4	Visualization and Charts	USN-7	Visualizing User data using different types of chart	10	Medium	Darwesh Fazil, Aparna, Grace Ebenezer
Sprint-4	API Integration	USN-8	Populating UI with Dynamic data	5	Low	Abiraj, Aparna, Grace Ebenezer
Sprint-4	Unit Testing	USN-9	Testing the core functions	5	Low	Darwesh Fazil, Aparna, Grace Ebenezer, Abiraj

6.2 SPRINT DELIVERY SCHEDULE

SPRINT	TOTAL STORY POINTS	DURATION	SPRINT START DATE	SPRINT END DATE	STORY POINTS COMPLETED	SPRINT RELEASE DATE
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	20	6 Days	14 Nov 2022	19Nov 2022		

6.3 REPORTS FROM JIRA



CHAPTER-7

CODING & SOLUTIONING

7.1 FEATURE 1

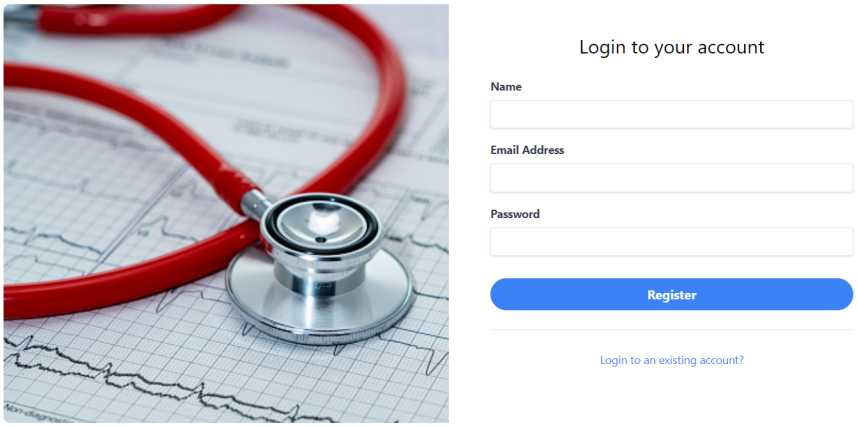


Figure 7.1 Patient Login and Register

7.2 FEATURE 2

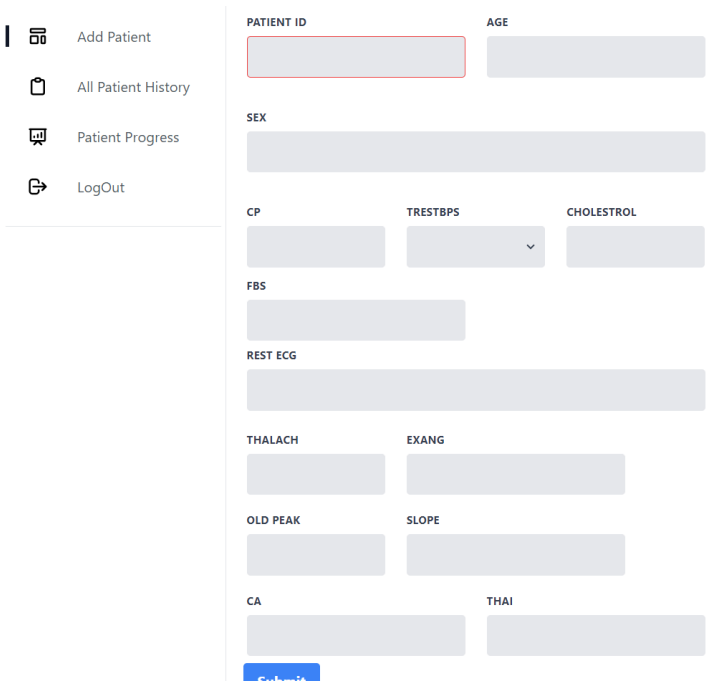
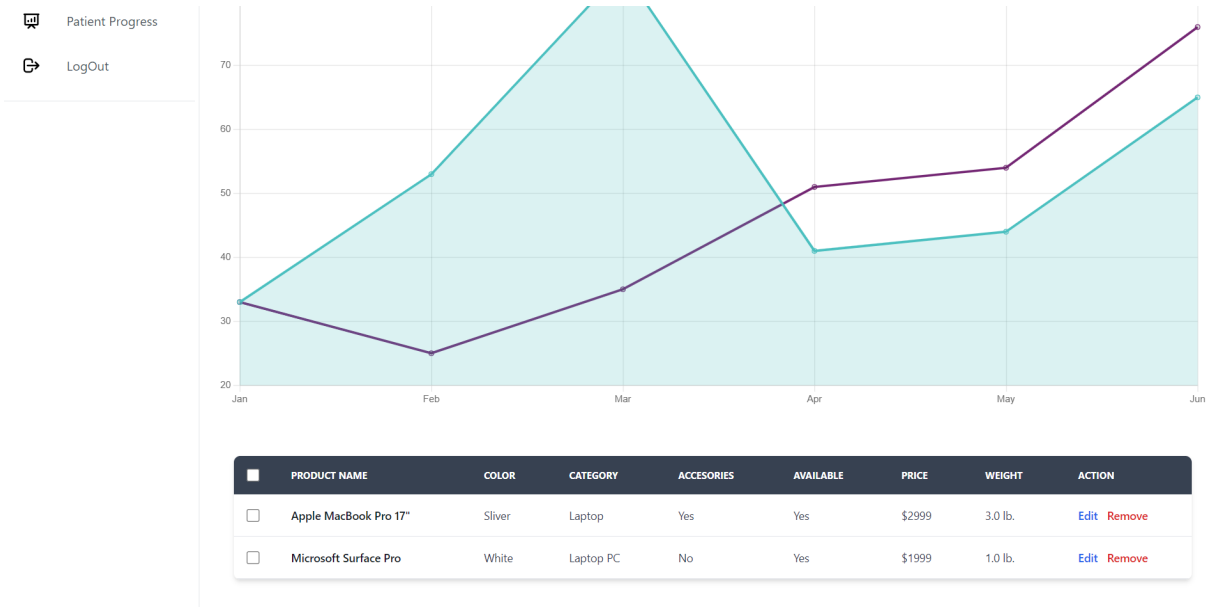


Figure 7.2 Predict patient's chance of risk of heart disease

7.3 FEATURE 3

Figure 7.3 Visualize each patient's progress

7.4 FEATURE 4



PRODUCT NAME	COLOR	CATEGORY	ACCESORIES	AVAILABLE	PRICE	WEIGHT	ACTION
Apple MacBook Pro 17"	Sliver	Laptop	Yes	Yes	$2999	3.0 lb.	Edit Remove
Microsoft Surface Pro	White	Laptop PC	No	Yes	$1999	1.0 lb.	Edit Remove

Figure 7.4 View each patient's records

CHAPTER-8

TESTING

8.1 TEST CASES

```
df = pd.read_csv('heart.csv')

X = df.drop("target", axis=1)
y = df["target"]
X_test = np.array([58.0,0.0,0.0,100.0,248.0,0.0,0.0,122.0,0.0,1.0,1.0,0.0,2.0])
X_test = X_test.reshape((1,-1))

model = RandomForestRegressor()
model.fit(X, y)

val = model.predict(X_test)

print(int(val*100),"%")
```

✓ 0.3s

```
73 %
```

```
df = pd.read_csv('heart.csv')

X = df.drop("target", axis=1)
y = df["target"]
X_test = np.array([52.0,1.0,0.0,125.0,212.0,0.0,1.0,168.0,0.0,1.0,2.0,2.0,3.0])
X_test = X_test.reshape((1,-1))

model = RandomForestRegressor()
model.fit(X, y)

val = model.predict(X_test)

print(int(val*100),"%")
```

✓ 0.3s

```
0 %
```

```
df = pd.read_csv('heart.csv')

X = df.drop("target", axis=1)
y = df["target"]
X_test = np.array([204.0,1.0,0.0,210.0,239.0,0.0,1.0,138.0,1.0,2.8,1.0,1.0,3.0])
X_test = X_test.reshape((1,-1))

model = RandomForestRegressor()
model.fit(X, y)

val = model.predict(X_test)

print(int(val*100),"%")
```

✓ 0.3s

```
5 %
```

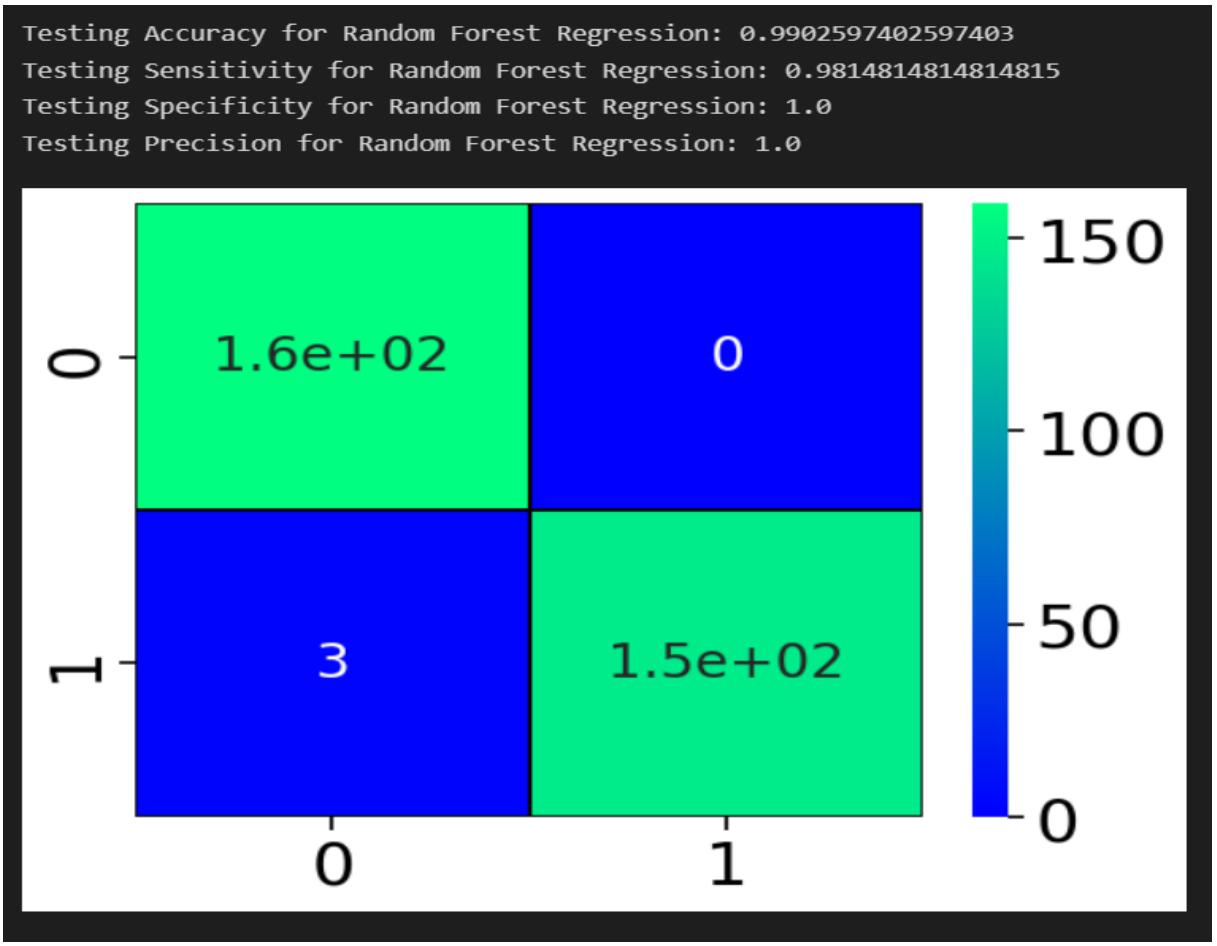
8.2 USER ACCEPTANCE TESTING

CHAPTER-9

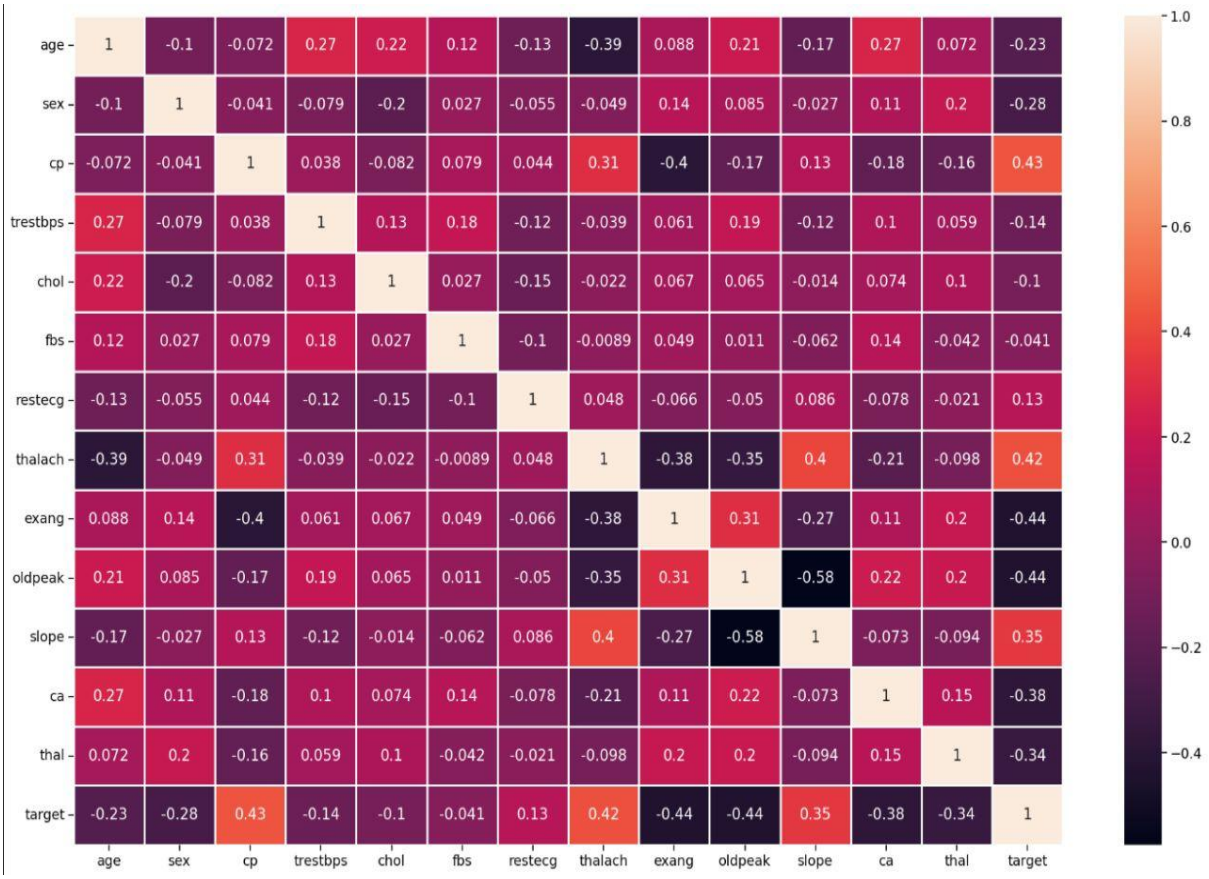
RESULTS

9.1 PERFORMANCE METRICS

Various performance metrics of the model is analyzed and documented as below



i) Confusion Matrix


age 1 -0.1 -0.072 0.27 0.22 0.12 -0.13 -0.39 0.088 0.21 -0.17 0.27 0.072 -0.23
sex -0.1 1 -0.041 -0.079 -0.2 0.027 -0.055 -0.049 0.14 0.085 -0.027 0.11 0.2 -0.28
cp -0.072 -0.041 1 0.038 -0.082 0.079 0.044 0.31 -0.4 -0.17 0.13 -0.18 -0.16 0.43
trestbps 0.27 -0.079 0.038 1 0.13 0.18 -0.12 -0.039 0.061 0.19 -0.12 0.1 0.059 -0.14
chol 0.22 -0.2 -0.082 0.13 1 0.027 -0.15 -0.022 0.067 0.065 -0.014 0.074 0.1 -0.1
fbs 0.12 0.027 0.079 0.18 0.027 1 -0.1 -0.0089 0.049 0.011 -0.062 0.14 -0.042 -0.041
restecg -0.13 -0.055 0.044 -0.12 -0.15 -0.1 1 0.048 -0.066 -0.05 0.086 -0.078 -0.021 0.13
thalach -0.39 -0.049 0.31 -0.039 -0.022 -0.0089 0.048 1 -0.38 -0.35 0.4 -0.21 -0.098 0.42
exang 0.088 0.14 -0.4 0.061 0.067 0.049 -0.066 -0.38 1 0.31 -0.27 0.11 0.2 -0.44
oldpeak 0.21 0.085 -0.17 0.19 0.065 0.011 -0.05 -0.35 0.31 1 -0.58 0.22 0.2 -0.44
slope -0.17 -0.027 0.13 -0.12 -0.014 -0.062 0.086 0.4 -0.27 -0.58 1 -0.073 -0.094 0.35
ca 0.27 0.11 -0.18 0.1 0.074 0.14 -0.078 -0.21 0.11 0.22 -0.073 1 0.15 -0.38
thal 0.072 0.2 -0.16 0.059 0.1 -0.042 -0.021 -0.098 0.2 0.2 -0.094 0.15 1 -0.34
target -0.23 -0.28 0.43 -0.14 -0.1 -0.041 0.13 0.42 -0.44 -0.44 0.35 -0.38 -0.34 1
age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal target
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

ii) Heat Map

CHAPTER-10

ADVANTAGES & DISADVANTAGES

ADVANTAGES :

- Medical professionals can quickly recognize and respond to potential dangers with the right visualization tools.
- Doctors may better define patient populations and allocate resources by displaying health data in real-time.
- Doctors may better forecast their patients' health and make more accurate diagnoses using data visualization software equipped with predictive analytics technologies.
- Clear and simple charts and infographics improve patient awareness and increase participation, making visualization a helpful tool for patient education.
- Visualization tools can significantly improve presentations and reports.
- Good data visualization aids in making the call to impose health regulations.

DISADVANTAGES :

- Due to the risk of malpractice, a visualization intended for use by health care professionals must be so clear and precise as to render misinterpretation virtually impossible.
- Insufficient data or poor structuring of data objects in the database may lead to inaccurate interpretation of visualized data.

CHAPTER-11

CONCLUSION

Diagnosis of cardiac disease is the sternest challenge in the medical profession. It is based on the thorough review by medical experts of the various clinical and medical data of the patient.The methods which are used for comparison of ML models are confusion matrix, heat map.. For the 13 features which were in the dataset, KNeighbors classifier performed better in the ML approach when data preprocessing is applied.

CHAPTER-12

FUTURE SCOPE

- The current project does for prediction and visualization of heart diseases. It can be extended to add features related to hospital management.
- The dataset size can be increased and then deep learning with various other optimizations can be used and more promising results can be achieved.
- More ways could be found where we could integrate heart-disease-trained ML and DL models with certain multimedia for the ease of patients and doctors.
- If a large dataset is present, the results can increase very much in deep learning and ML as well
- The prediction of heart diseases by using advanced techniques and algorithms in less time complexity

CHAPTER-13

APPENDIX

13.1 SOURCE CODE

prediction.py

```
import numpy as np
import pandas as pd
import pickle
from matplotlib import rcParams
from matplotlib.cm import rainbow
import warnings
warnings.filterwarnings('ignore')

from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
import os
def initialize_dataset():
    global df
    df = pd.read_csv('prediction/heart.csv')

def train_model():
    from sklearn.ensemble import RandomForestRegressor

    initialize_dataset()

    X = df.drop("target", axis=1)
```

```
    y = df["target"]

    model = RandomForestRegressor()
    model.fit(X, y)
    print("Train complete")
    pickle.dump(model, open('prediction/model.pkl', 'wb'))

def get_prediction(record):

    initialize_dataset()

        X_test = np.array([record.age, record.sex, record.cp, record.trestbps, record.chol, record.fbs, record.restecg, \
       record.thalach, record.exang, record.oldpeak, record.slope, record.ca, record.thal])
    X_test = X_test.reshape((1,-1))
    pickled_model = pickle.load(open('prediction/model.pkl', 'rb'))

    val = pickled_model.predict(X_test)
    val = val*100

    add_data(X_test, int(val), df)
    #add data to the csv
    #initiate training with new data
    record.target = int(val)
    return record
```

```
def add_data(X_test, val, df):
    temp = np.append(X_test, val)
    temp = temp.reshape((1,-1))
      temp = pd.DataFrame(temp, columns=['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'])
    df = df.append(temp)
    open("prediction/heart.csv", "w").close()
    df.to_csv('prediction/heart.csv', mode='a', index=False, header=True)
```

main.py (API)

```
from fastapi import FastAPI,HTTPException,status,BackgroundTasks
from utils.auth_utils import genToken, verifyToken
from db.db_enum import Queries
from db.db_helpers import *
from models.fast_api_models import *
from models.models import *
from prediction.prediction import *
import uvicorn
from fastapi.middleware.cors import CORSMiddleware
app=FastAPI()

origins = [
    "http://localhost:3000",
]

app.add_middleware(
```

```
    CORSMiddleware,
    allow_origins=origins,
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

@app.get("/")
async def root():
    return {"message": "Hello World"}

@app.post('/doctor/login', status_code=status.HTTP_200_OK)
def doctor_login(request:LoginRequest):

                                                                    present                    =
user_present(Queries.DOCTOR_TABLE_NAME.value,(request.email,request.password))

    if present:
        token = genToken(request.dict())
        data = {"message":"SUCCESS","result":[{"token":token}]}
        response = LoginResponse(**data)
        return response
    else:
        data = {"message":"User not found"}
        response = LoginResponse(**data)
        raise HTTPException(404,detail=[response.dict()])
```

```
@app.post('/doctor/register',status_code=status.HTTP_201_CREATED)
def doctor_register(request:Doctor):
                                                    present =
user_present(Queries.DOCTOR_TABLE_NAME.value,(request.email,request.password))
    if not present:
        #register
        result = insert_operation(Queries.ADD_DOCTOR,request.tuple_for_insert())
        if result:
            return {"message":"success"}
        else:
            raise HTTPException(404,{"Some error occurred"})
    else:
        data = {"message":"User already exists"}
        raise HTTPException(404,data)



@app.post('/doctor/get-patients')
def doctor_get_patients(token:str):
    present,id = verifyToken(token)
    if present:
        result = fetch_operation(Queries.FETCH_PATIENTS_OF_DOCTOR,(id,))
        return {"message":"Success","result":result}
    #fetch operation
    else:
```

```
        data = {"message":"Invalid token","result":None}
        raise HTTPException(404,data)


@app.post('/doctor/all-records')
def doctor_all_records(token:str):
    present,id = verifyToken(token)
    if present:
        result = fetch_operation(Queries.MY_PATIENTS_ALL_RECORDS,(id,))
        #have to refine columns
        return {"message":"Success","result":result}
    #fetch operation
    else:
        data = {"message":"Invalid token","result":None}
        raise HTTPException(404,data)
    pass




#Patient Routes
@app.post('/patient/register')
def patient_register(request:Patient):
                                                        present =
user_present(Queries.PATIENT_TABLE_NAME.value,(request.email,request.password))
    if not present:
        #register
```

```
        result = insert_operation(Queries.ADD_PATIENT,request.tuple_for_insert())
        if result:
            return {"message":"success"}
        else:
            raise HTTPException(404,{"Some error occurred"})
    else:
        data = {"message":"User already exists"}
        raise HTTPException(404,data)


@app.post('/patient/login')
def patient_login(request:LoginRequest):
                                                    present =
user_present(Queries.PATIENT_TABLE_NAME.value,(request.email,request.password))


    if present:
        token = genToken(request.dict())
        data = {"message":"SUCCESS","result":[{"token":token}]}
        response = LoginResponse(**data)
        return response
    else:
        data = {"message":"User not found"}
        response = LoginResponse(**data)
        raise HTTPException(404,detail=[response.dict()])


@app.post('/patient/predict')
async def predict_and_add_data(record:Record, background_tasks: BackgroundTasks):
```

```
    result = get_prediction(record)
    print("prediction done")

background_tasks.add_task(insert_operation,Queries.INSERT_PATIENT_SINGLE_RECORD,result.tuple_for_insert())
    print("insert started")
    background_tasks.add_task(train_model)
    return {"message":"Success","result":[{"prediction":result.target}]}


@app.post('/patient/my-records')
def patient_my_records(token:str):
    present,id = verifyToken(token)
    if present:
        result = fetch_operation(Queries.FETCH_PATIENT_ALL_RECORDS,(id,))
        #have to refine columns
        return {"message":"Success","result":result}
    #fetch operation
    else:
        data = {"message":"Invalid token","result":None}
        raise HTTPException(404,data)
    pass


@app.get('/patient/record')#req params
def patient_date_records(request:TokenRequest):
    print(request.dict())
    return {"message":"suscces"}
```

```
if __name__ == "__main__":
  uvicorn.run("main:app", reload=True)
```

Landing.js

```
import React from 'react'
import { Radio } from '../components/buttons/Radio'
export const Landing = () => {
  return (
    <div class="flex flex-col h-screen my-auto items-center">
    <div class="text-center mt-10">
       <h1 className='text-3xl font-bold mt-5'>Login to your account</h1>
       <Radio/>
    </div>
    </div>
  )
}
```

AddPatient.js

```
import React from 'react'
import { AddPatientForm } from '../components/forms/AddPatientForm'
import NavBar from '../components/navbar/NavBar'
import Header from '../components/header/Header'
import LeftPart from '../components/leftPart/LeftPart'

export const AddPatient = () => {
  return (
    <div>
```

```
      <div className="App overflow-y-hidden ">
     <Header/>
     <div className='w-full min-h-[90vh] grid grid-cols-12'>
     <NavBar/>
     <div className='grid grid-cols-1 xl:grid-cols- col-span-10 w-full'>

      <AddPatientForm/>
     </div>
     </div>
    </div>
    </div>
  )
}
```

REFERENCES

GitHub Repository Link :

https://github.com/IBM-EPBL/IBM-Project-15341-1659597507

Project Demo Link :

https://heart-disease-prediction-dashboard.vercel.app/

IEEE Paper:

A. Rahim, Y. Rasheed, F. Azam, M. W. Anwar, M. A. Rahim and A. W. Muzaffar, "An Integrated Machine Learning Framework for Effective Prediction of Cardiovascular Diseases," in IEEE Access, vol. 9, pp. 106575-106588, 2021, doi: 10.1109/ACCESS.2021.3098688.