Date: 15/11/2021

SOFTWARE DESIGN DOCUMENT

For the software application,

CHRONIC DISEASE PREDICTOR - AN ML BASED APPLICATION

DONE BY:

TEAM NO: 4

19Z202 - Anita Priyadarshini
19Z211 - Divya Darshini
19Z217 - Hemavarshini B
19Z227 - Monirhithika
19Z240 - Sarayumithira
19Z243 - Samyuktha S

TABLE OF CONTENTS

DATA DESIGN

4.1 Data Description

4.2 Data Dictionary

COMPONENT DESIGN

HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

6.2 Screen Images

6.3 Screen Objects and Actions

REQUIREMENTS MATRIX

APPENDICES

8.1 Analysis Model – DFD Diagram

****

1. Introduction

1.1 Purpose

This document describes the software specification requirements of our software "CHRONIC DISEASE PREDICTOR". The document also emphasizes how the software collects the data from the users and uses the data to estimate whether the user is at risk of the chronic diseases. The document contains the functional behavior and non-functional requirements of the project. This also plays as a guidance and framework for system engineers and programmers to build the software with the functionalities listed in a limited time frame. The software is intended to benefit the society by making them more aware of their body health.

1.2 Scope

The software product is a disease prediction effort aimed at 3 most common chronic diseases at present, viz stroke, cardiovascular diseases and diabetes. The system incorporates the fundamentals of Machine Learning in its working; by training the algorithm model with data sets and then analyzing the user data to provide the result.

The system's objective is to empower the public with a tool to get themselves checked periodically and to stay proactive about their wellbeing. The software can be extended to accommodate multiple chronic disorders of varying data sets. A separate database can be created listing the contacts of doctors specializing in the particular disease and displayed to users at risk of the same, by grouping the contacts based on the region they entered in their demographics profile. These features sum up the scope of our system.

1.3 Overview

The purpose of our Project is to Predict the risk of an individual to be diagnosed with a particular Chronic Disease. Today, the scientific enterprise performs an essential function in curing patients' ailments and is beneficial to customers after they no longer need to visit a medical institution or different clinic, so customers input signs and symptoms and all different beneficial information. You can study contamination, and the scientific enterprise can benefit from this application sincerely through querying the person's signs and symptoms and getting them into the application, and subsequently the person might be knowledgeable about destiny contamination predictions

The current software system at place comes with a user registration and login experience. The users may then be prompted to a menu screen where the radio buttons of the diseases are present. The application redirects the users for extracting the data required for prediction and provides the users with a dependable estimate of their health in an ML driven environment. The back-end and front-end are seamlessly interfaced along with the database to provide a smooth-functioning application.

1.4 Reference Material

i. *IEEE Std 10161998*

ii. https://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf

iii. https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.06-Linear-Regression.ipynb

1.5 Definitions andAcronyms

1. *SDD: Software Design Document*

2. *SRS: Software Requirements Specification document*

3. *DFD: Data Flow Diagrams*

4. *Chronic disease: A disorder in the body caused by internal factors and is ongoing as opposed to an infection*

5. *Disease parameter: A risk factor associated with a particular illness*

6. *Classification problem: A type of problem encountered in ML, where the output is expected to be binary*

7. *Logistic Regression Model: The name given to an ML algorithm / model which solves the classification problem with high accuracy.*

8. *Model training: An ML model needs to be trained with existing and clean data sets for it to function properly in the application*

9. *Framework: A framework is a container having tools, libraries and technologies targeted to build an application.*

10. *Model evaluation: It is a procedure to quantify the quality of the model's predictions*

2. System Overview

This is a preventive app that enables in those tough instances while it is now no longer so handy to visit the medical institution and make an appointment. Based

on the input provided, you will be able to find the likelihood of the user suffering from the particular chronic disease. We also aspire to expand the system by offering suggestions and treatments from the local doctor to make it more convenient for you. The main goal is to go to the hospital during a pandemic, help people when they cannot make a convenient appointment, and provide medical assistance. The users of disease prediction web applications must be equipped with a device. A stable internet connection is recommended for a smooth user experience. The application is free of cost and is made available to all users.The Application predicts the closest disease based on the user inputs although it cannot guarantee an exact diagnosis of the disease.

In this project, an automated disease diagnosis model is designed using the machine learning models. In this project, we have selected three critical diseases such as Stroke, General cardiovascular disease, and diabetes. In this project, the data are entered into an web Application, the analysis is then performed in a real-time database using a pretrained machine learning model which was trained on the same dataset and deployed in MySQL Workbench, and finally, the disease detection result is shown in the Web Application. Logistic regression is used to carry out computation for prediction.

The speed and availability of a user's internet connection affects the response speed of the application. Our current software "Chronic disease predictor" is limited by certain technologies, tools, and databases used to the best of our knowledge and also the lack of funding for professional system development. The time frame available for development, number, and experience of developers also restricts our system's working when compared to the professional systems present on the market and internet.

3.System Architecture

3.1 Architectural Design

A typical three-layer structure is used in the system: the database layer, the application service layer, the user interface layer. System architecture as shown in Figure 1.

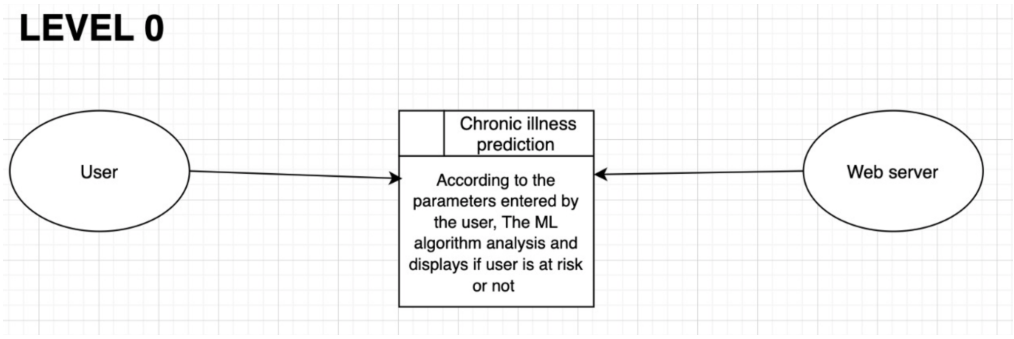
- *The database layer : The database is used to hold data, including user registration information, login information, etc*

- *The application service layer : The application service layer is the core of this three-layer structure, the system functions and business logic are handled in this layer. In this layer, the system's business logic is encapsulated, the application service interface is provided for the user interface layer and the system modules between the function calls. The application service layer also updates data in the database, according to the service request of the top layer.*

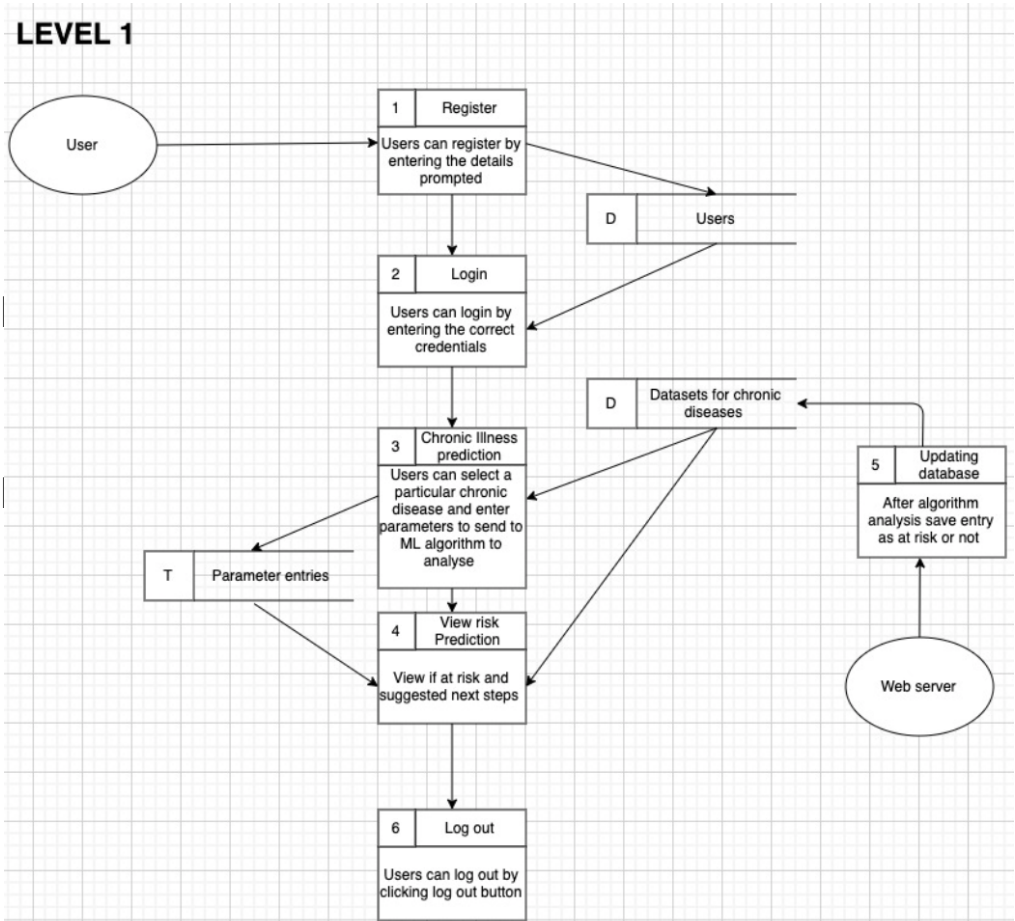
- *The user interface layer : The user interface layer is a program that runs on a remote user computer. It displays the provided services by the server to the user. When the user selects a service, this program sends a request to the server. When the server returns the processed result, this program shows it to the user.*

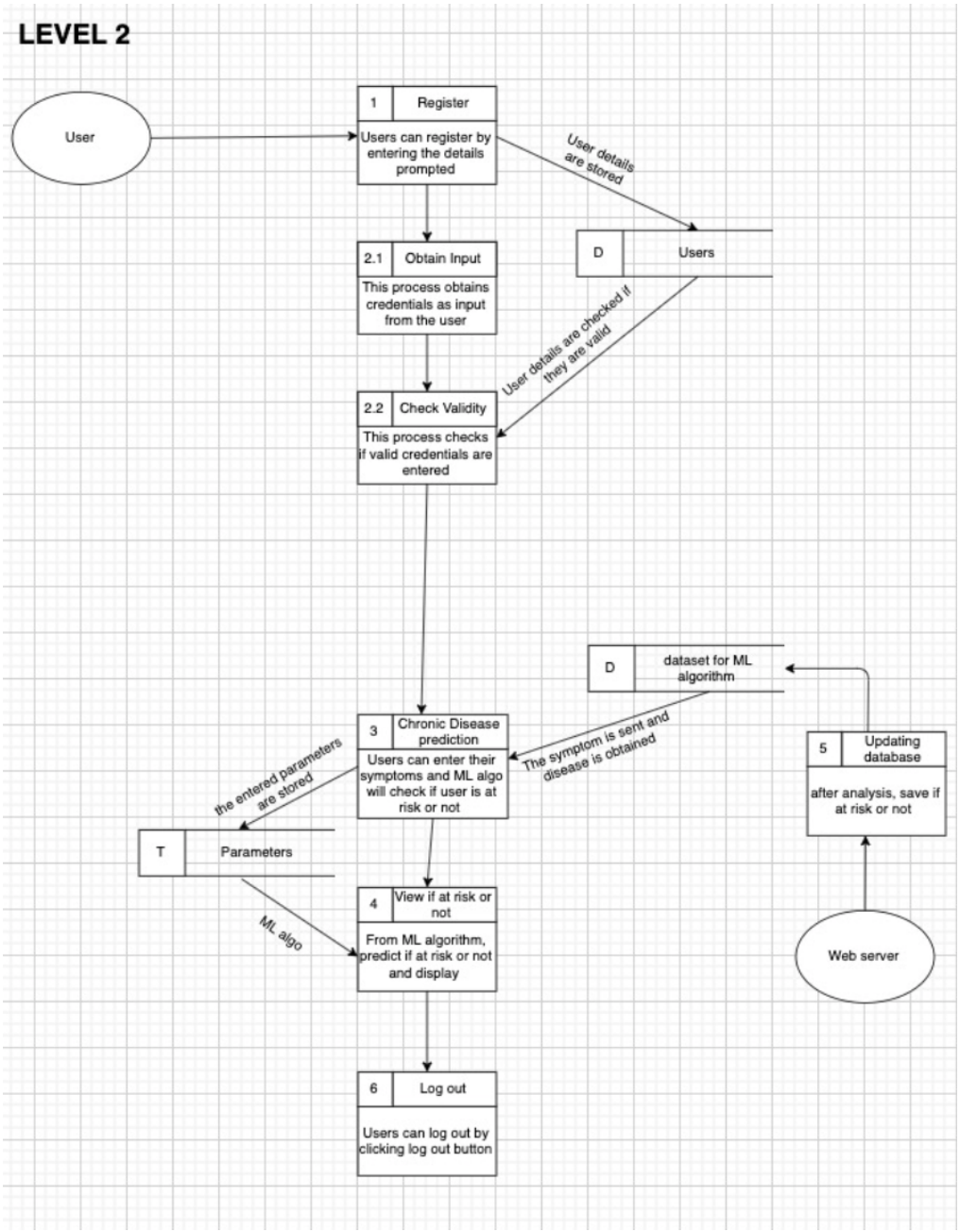
3.2 Decomposition Diagram

Level 0 Data flow diagram

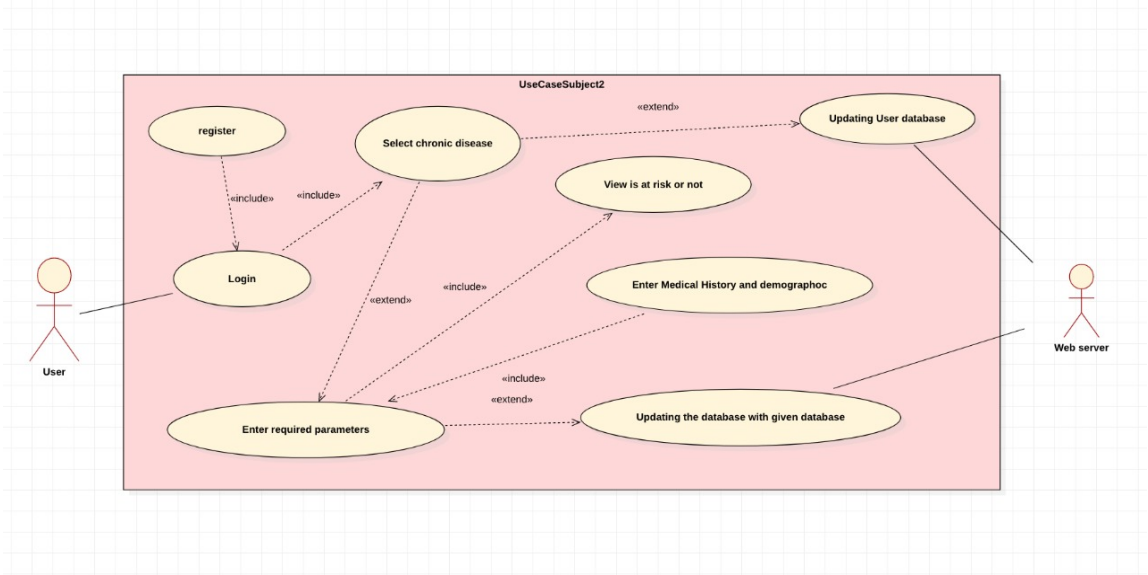


Level 1 Data flow diagram

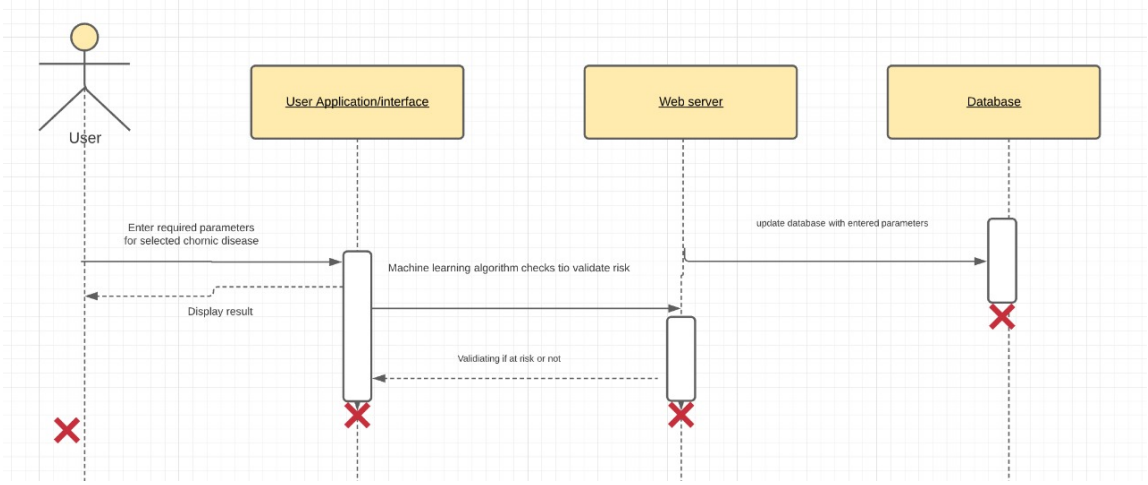



LEVEL 2
User
1
Register
Users can register by entering the details prompted
User details are stored
D
Users
2.1
Obtain Input
This process obtains credentials as input from the user
User details are checked if they are valid
2.2
Check Validity
This process checks if valid credentials are entered
D
dataset for ML algorithm
3
Chronic Disease prediction
Users can enter their symptoms and ML algo will check if user is at risk or not
The symptom is sent and disease is obtained
5
Updating database
after analysis, save if at risk or not
the entered parameters are stored
T
Parameters
ML algo
4
View if at risk or not
From ML algorithm, predict if at risk or not and display
Web server
6
Log out
Users can log out by clicking log out button

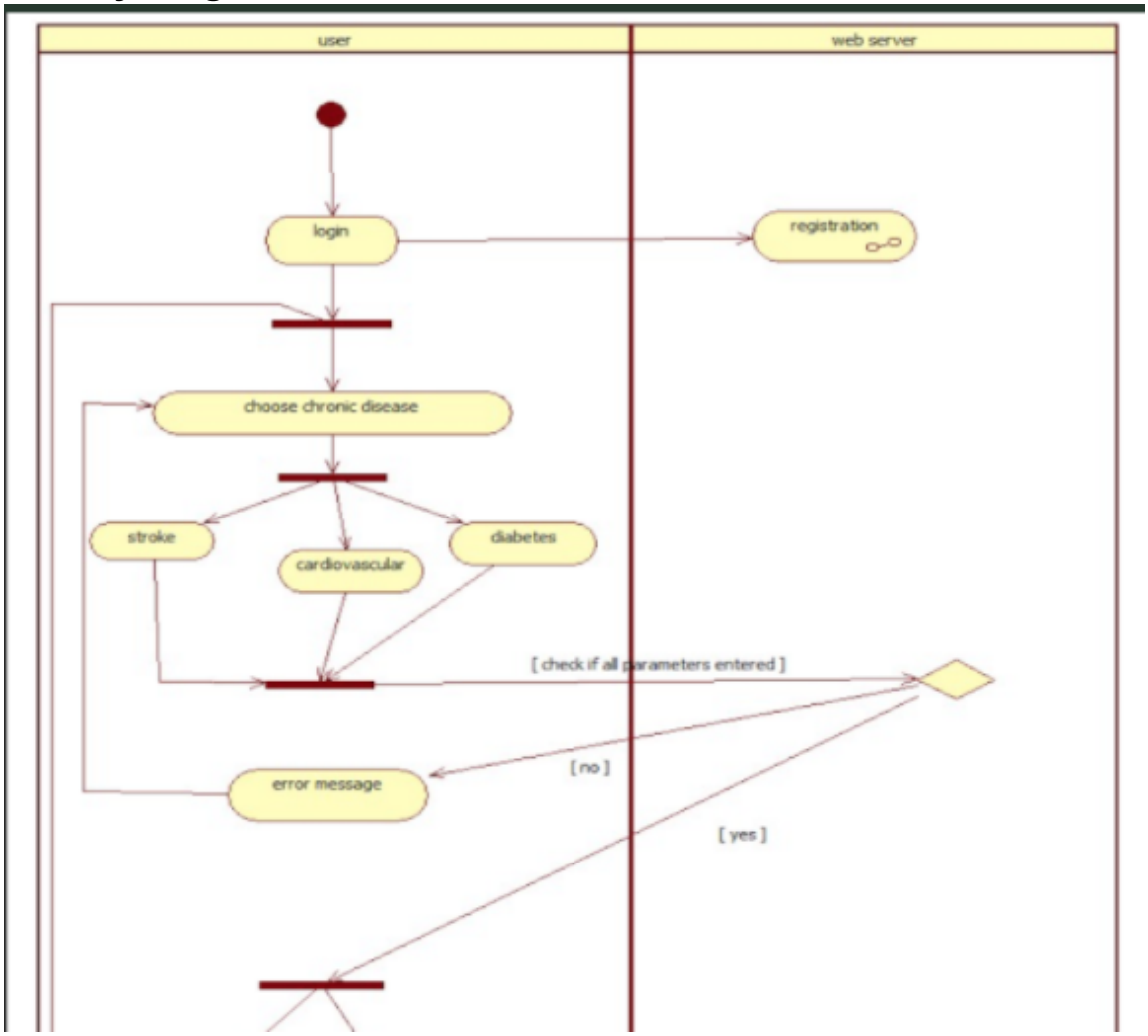
Use case diagram:

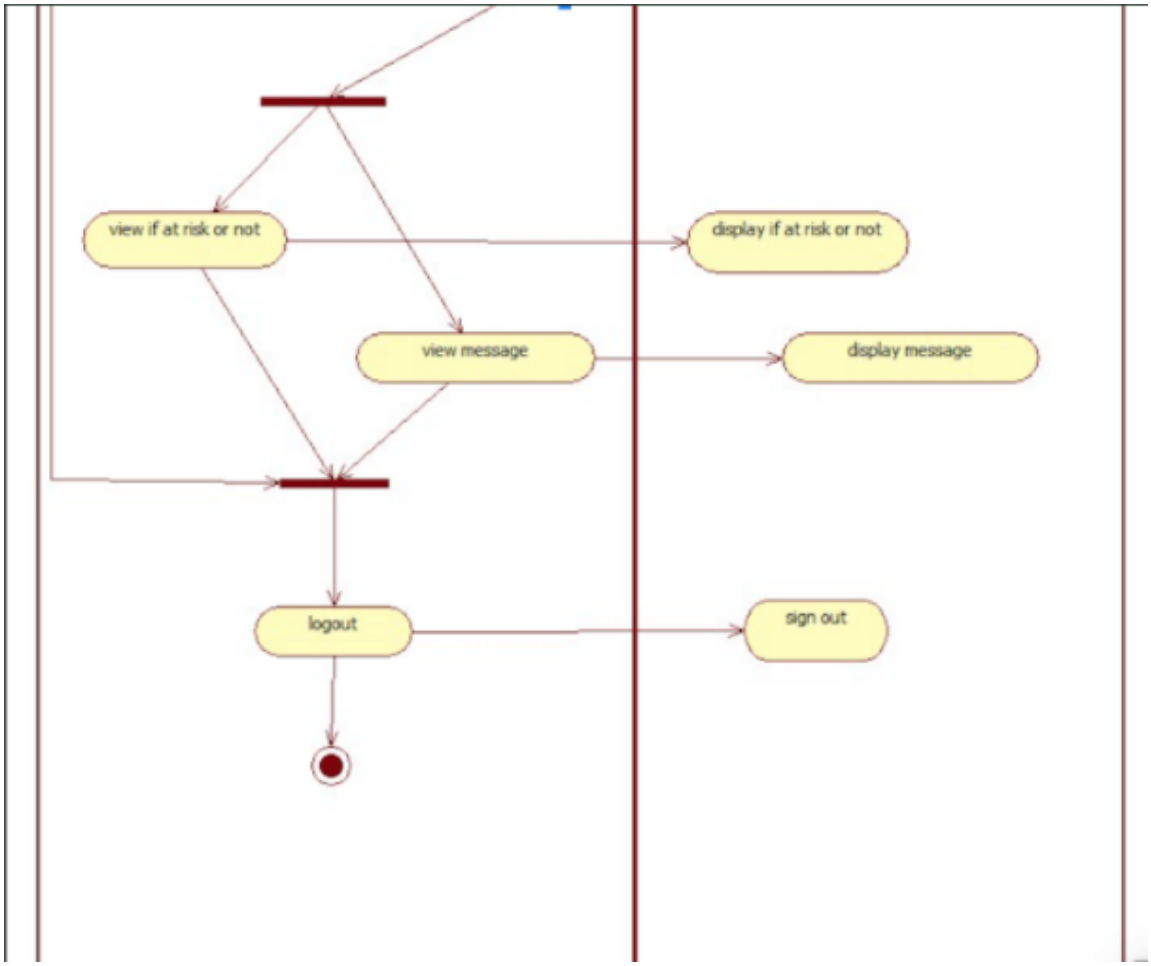


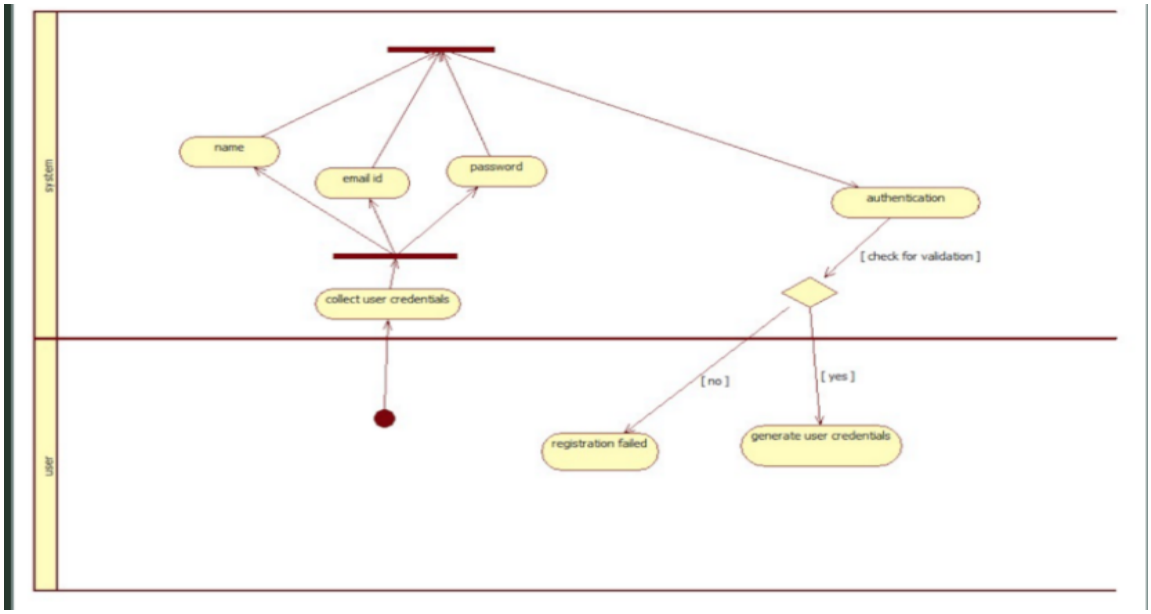
Sequence Diagram:



Activity Diagram:




view if at risk or not
display if at risk or not
view message
display message
logout
sign out


system
user
name
email id
password
collect user credentials
authentication
[check for validation]
[no]
[yes]
registration failed
generate user credentials


user
login
collect user details for updating their profile
name
age
phone number
medical records
[checks if details are valid]
receive error message
[no]
update profile
[yes]
enter symptoms
disease database
view predicted disease
display predicted disease
view suggested treatments
display suggested treatments
logout
sign out

3.3 Design Rationale

The structure of the project consists of the data access layer , detection layer using machine learning approaches to identify the disease of the person. It is a non profitable project that does not aim at business monetization of any sort.

- *The Application will only serve as a preliminary measure to visiting a doctor or hospital.*
- *The accuracy of the disease predicted is parallel to the accuracy of the symptoms entered by the user.*
- *The application does not substitute visiting the hospital or doctor.*
- *The application attempts to make the "most probable choice". This is important because the user must not use this as a final diagnosis and must consult their doctor for figuring out their final course of treatment.*

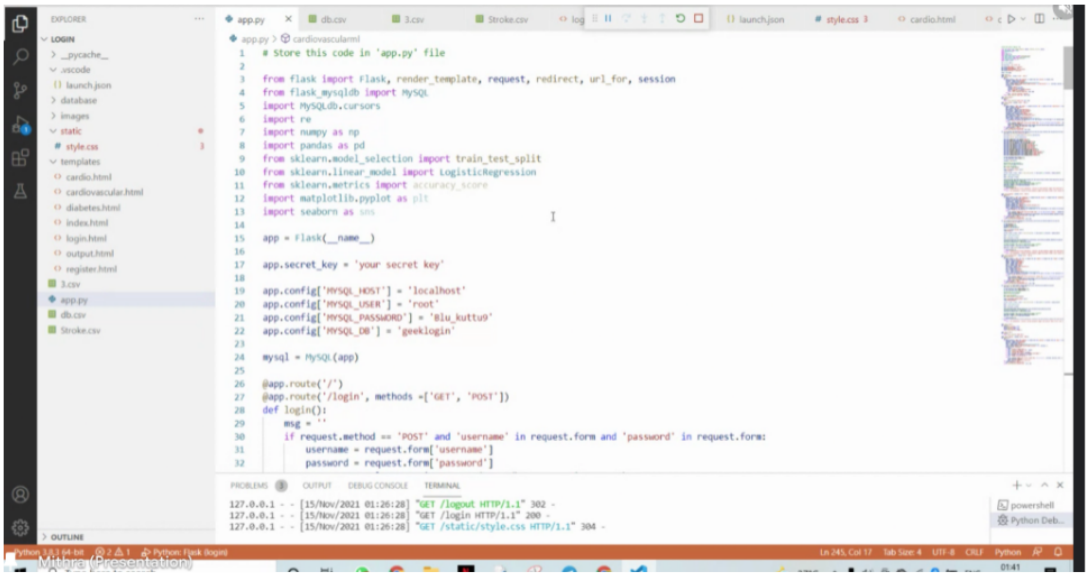
DATA DESIGN

4.1 Data Description

A library known as Flask is used for prediction. Flask is a micro web framework written in Python because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

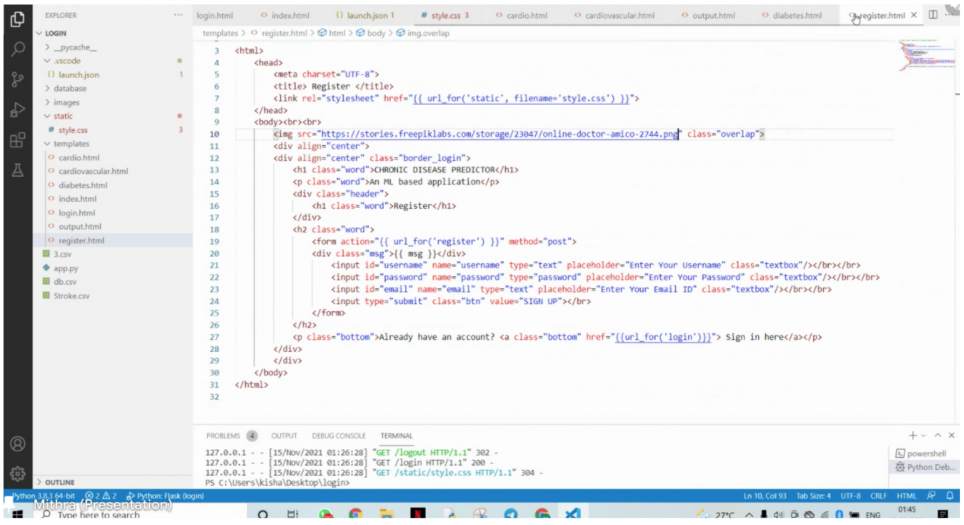
5.Component Design

Connection MYSql To VSCode



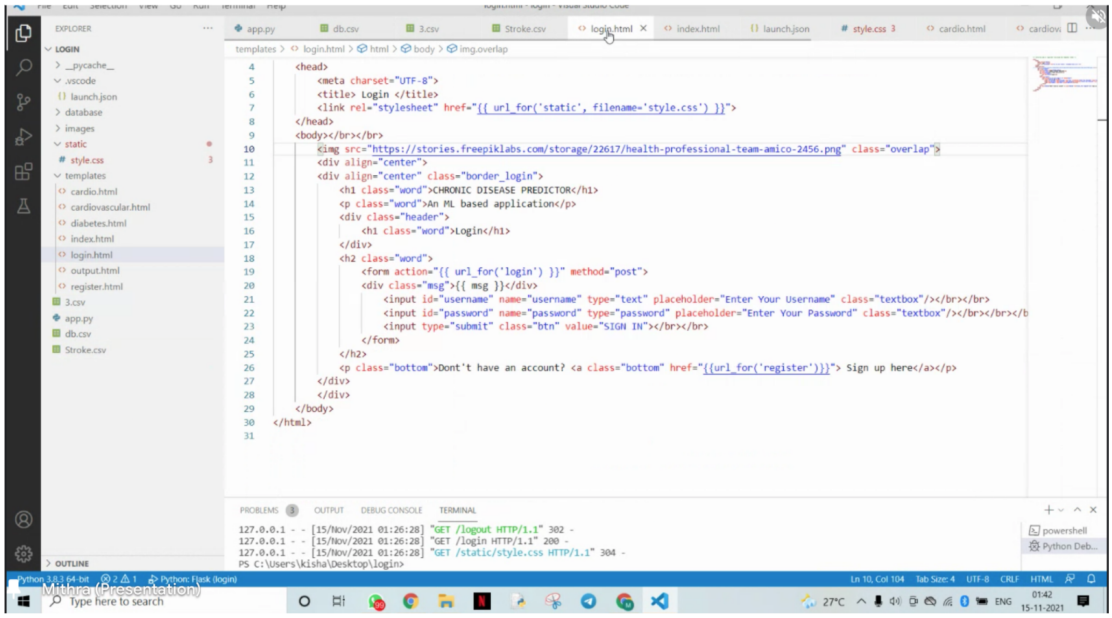
```
# Store this code in 'app.py' file

from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns

app = Flask(__name__)

app.secret_key = 'your secret key'

app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'Blu_kuttu9'
app.config['MYSQL_DB'] = 'geeklogin'

mysql = MySQL(app)

@app.route('/')
@app.route('/login', methods =['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
        username = request.form['username']
        password = request.form['password']
```

Register HTML

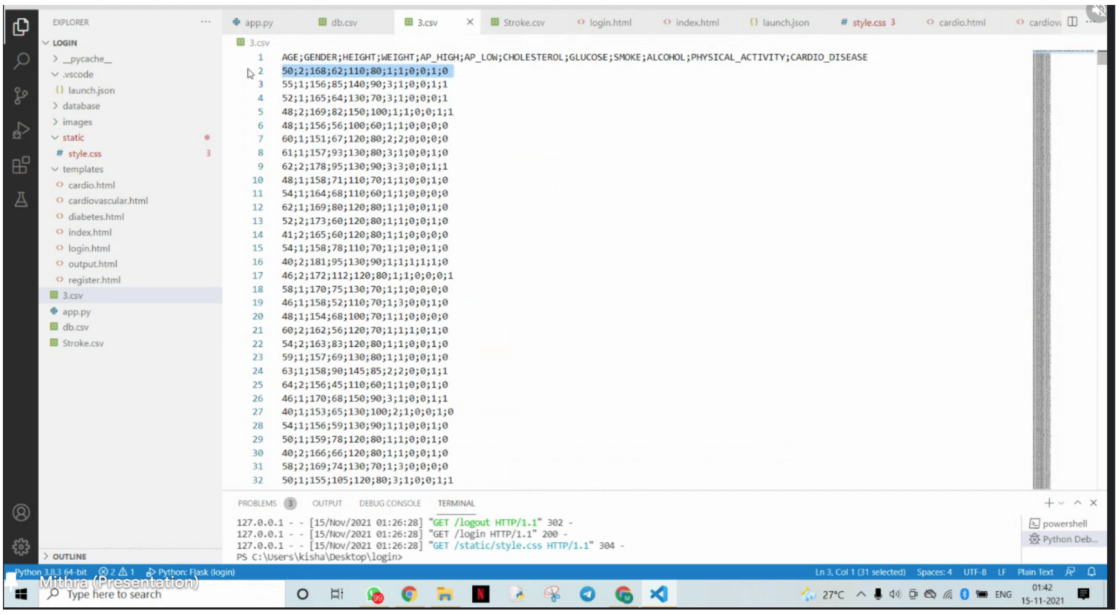


```
<html>
    <head>
        <meta charset="UTF-8">
        <title> Register </title>
        <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    </head>
    <body><br><br>
        <img src="https://stories.freepiklabs.com/storage/23047/online-doctor-amico-2744.png" class="overlap">
        <div align="center">
        <div align="center" class="border_login">
            <h1 class="word">CHRONIC DISEASE PREDICTOR</h1>
            <p class="word">An ML based application</p>
            <div class="header">
                <h1 class="word">Register</h1>
            </div>
            <h2 class="word">
                <form action="{{ url_for('register') }}" method="post">
                <div class="msg">{{ msg }}</div>
                    <input id="username" name="username" type="text" placeholder="Enter Your Username" class="textbox"/></br></br>
                    <input id="password" name="password" type="password" placeholder="Enter Your Password" class="textbox"/></br></br>
                    <input id="email" name="email" type="text" placeholder="Enter Your Email ID" class="textbox"/></br></br>
                    <input type="submit" class="btn" value="SIGN UP"></br>
                </form>
            </h2>
            <p class="bottom">Already have an account? <a class="bottom" href="{{url_for('login')}}"> Sign in here</a></p>
        </div>
        </div>
    </body>
</html>
```

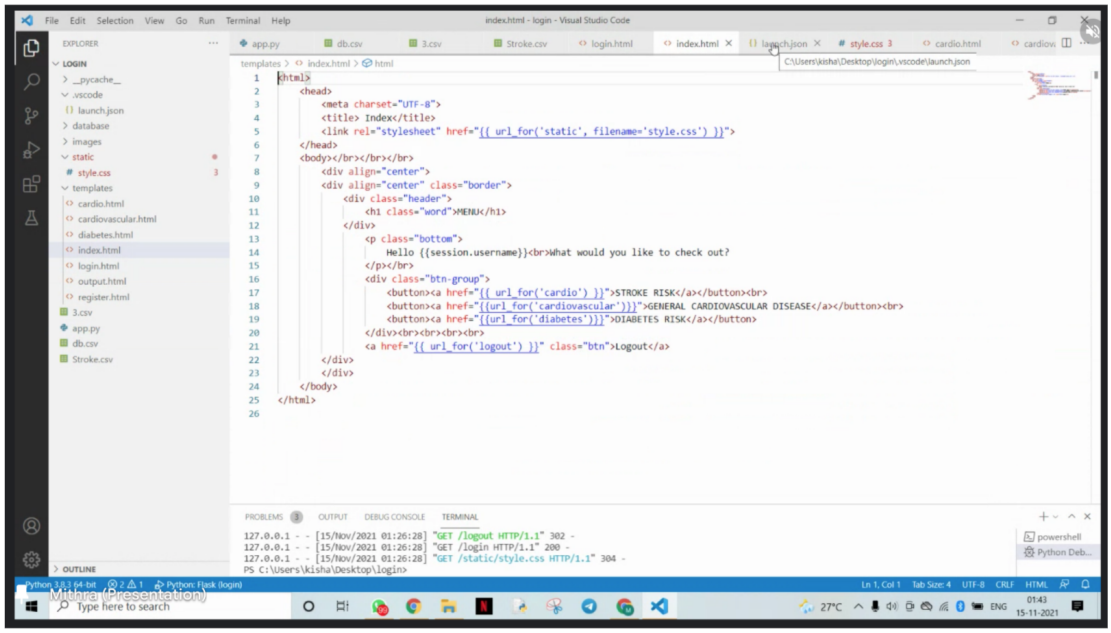
Login HTML



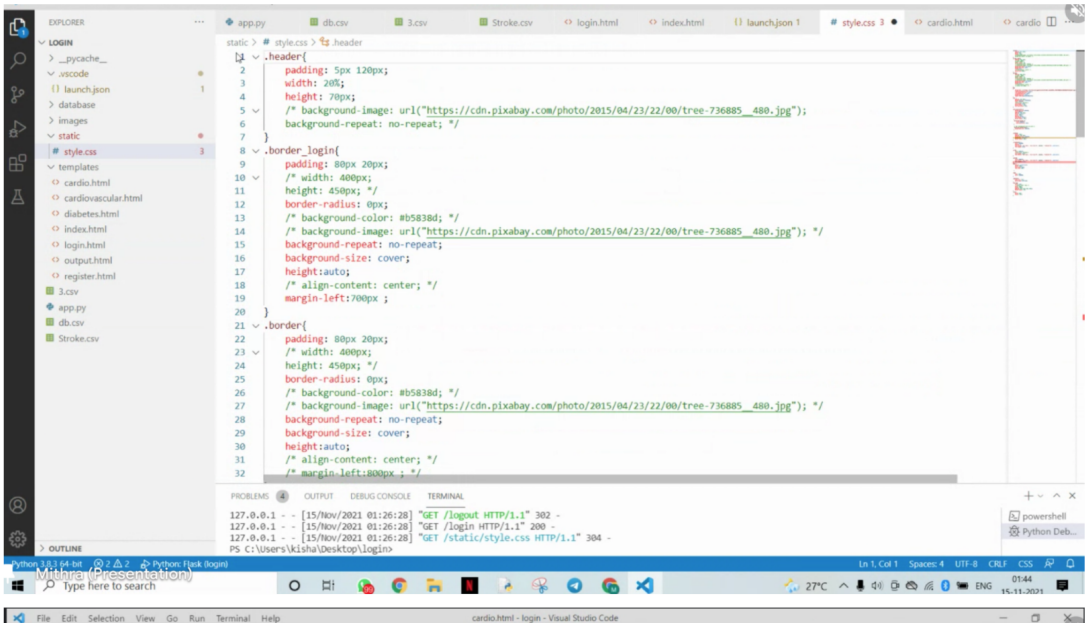
Database for ML Model



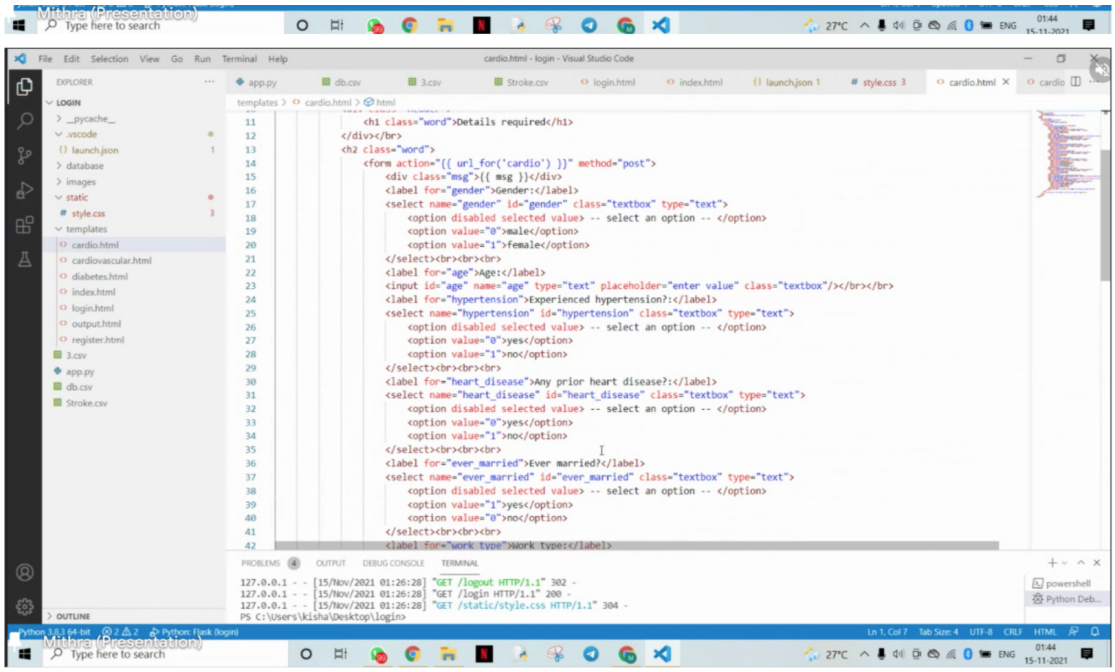
Disease Portal



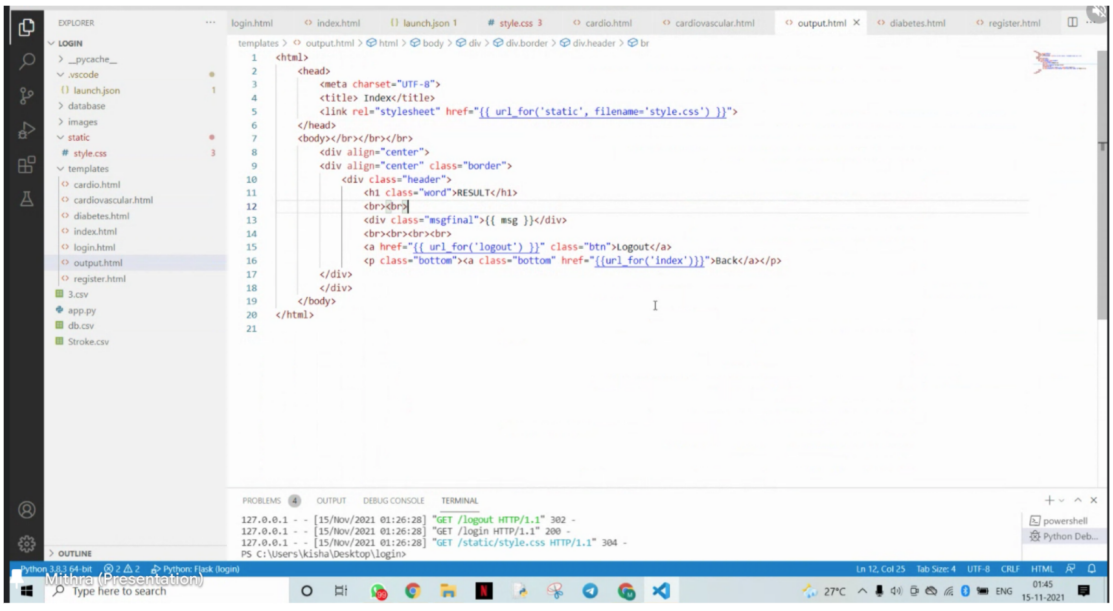
CSS Styling of pages



Input parameters for each disease



Output page



6. Human Interface design

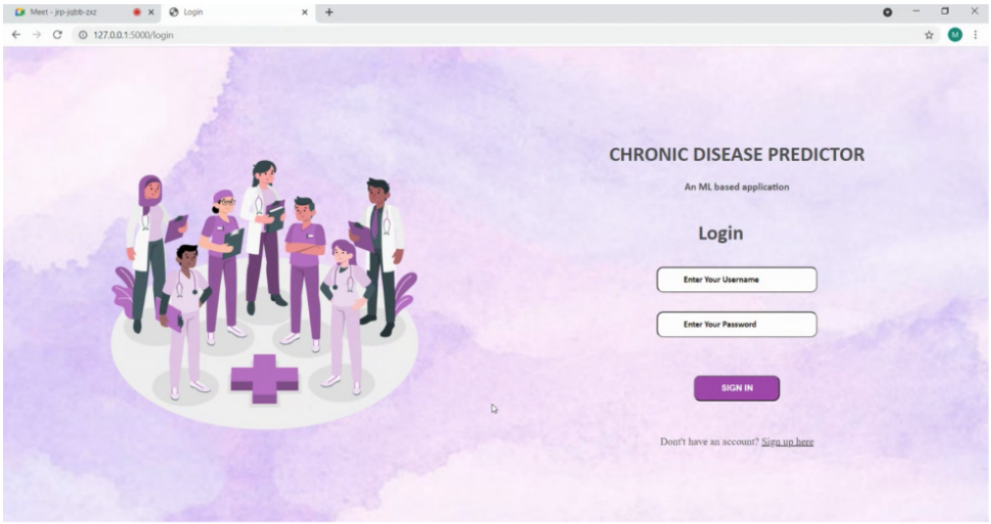
6.1 Overview of User Interface

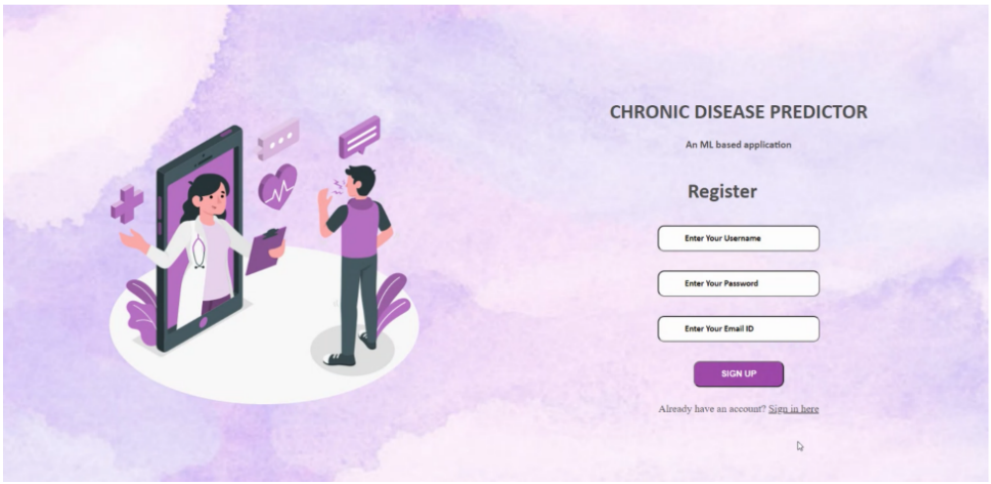
The User Interface is designed with an aim to make it simple for the user with no technical background to access our system freely without hassle or uncertainty. The UI consists of 6 webpages for each disease we offer to predict risk.

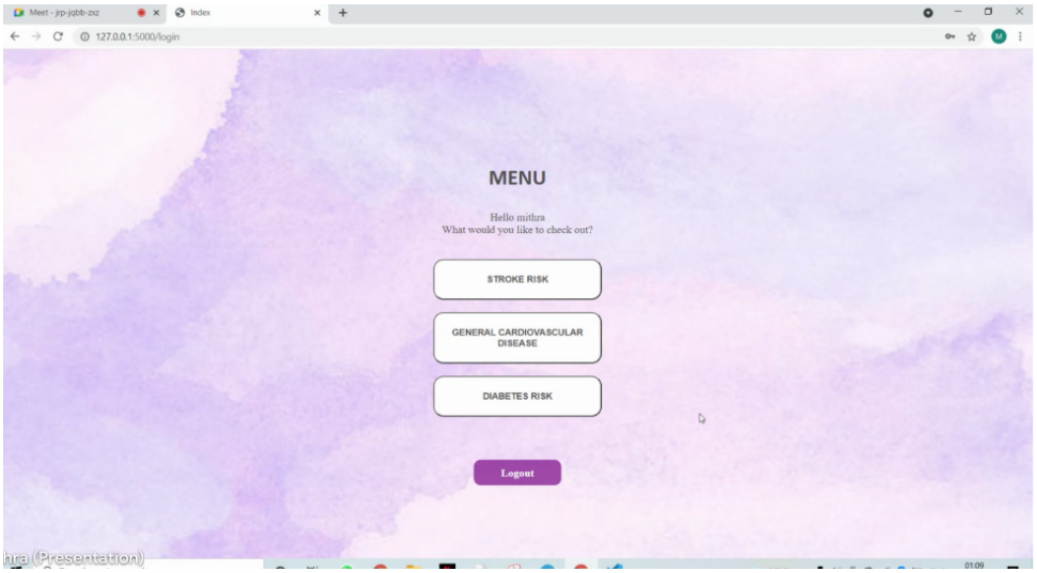
1. *Register page*
2. *Login page*
3. *Menu page*
4. *Data collection page*
5. *Result display page with 2 use cases*

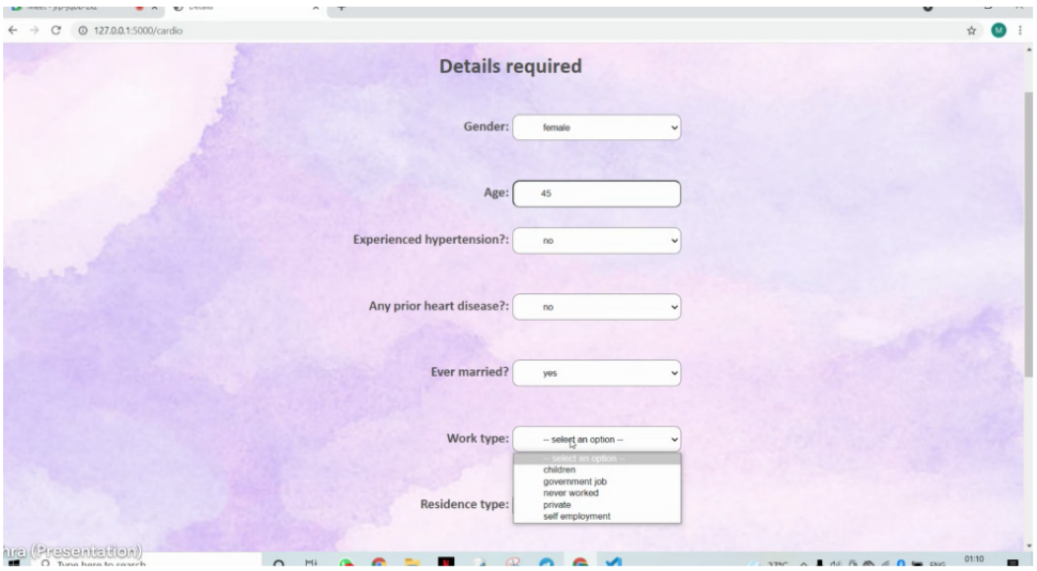
The register page which is the opening page of the application collects the username, email id and password of the new user while also providing a space for old users to sign into the application with the help of a button. This button leads to the login page collecting the username and password alone which is then processed in order to log the user into the main application which is the menu page. A simple yet sophisticated interface, the menu leads the users into the specific disease' homepage. This homepage is made to collect the demographics, lifestyle and body parameters data of the user for feeding into the back end model. The output thrown by the ML model determines which result page is displayed, by deciding between the two pre-built pages. The result display page has a convenient 'Logout' button and a 'Back' button that directs the user outside and into the application respectively. The screenshots are attached below for reference.

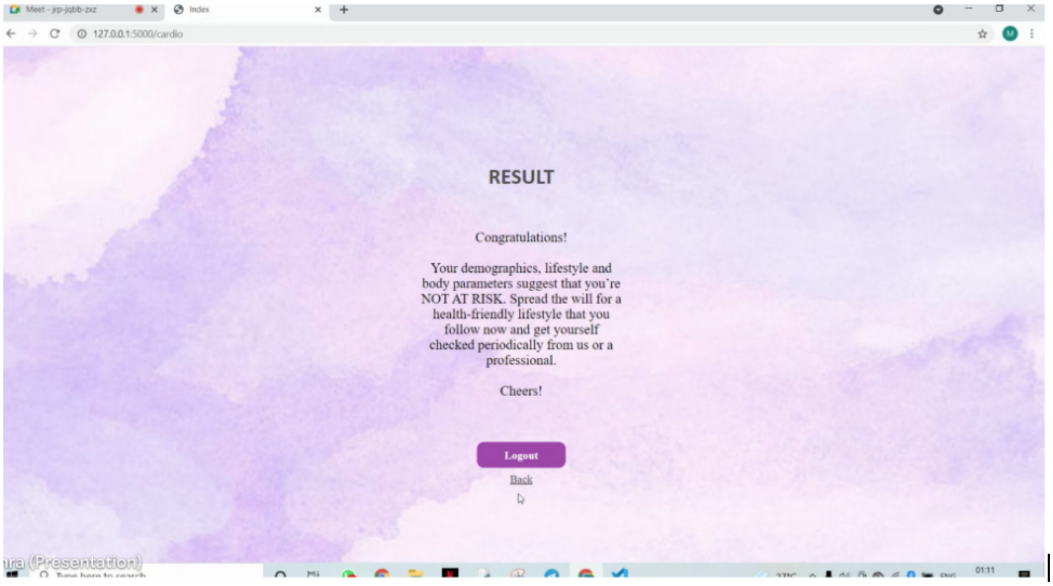
6.2 Screen Images

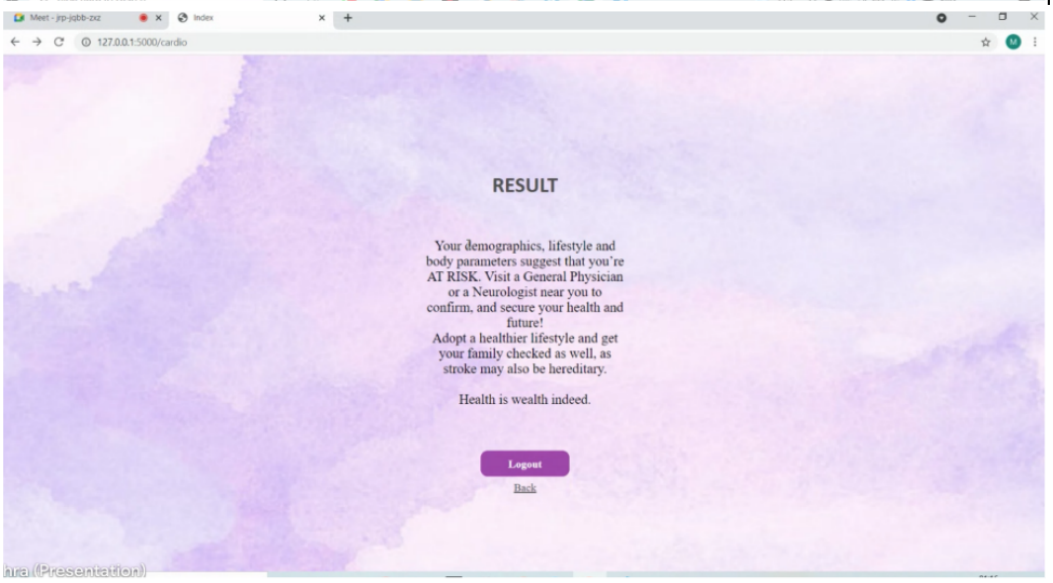

Meet - jrp-jqbb-zxz
Login
127.0.0.1:5000/login
CHRONIC DISEASE PREDICTOR
An ML based application
Login
Enter Your Username
Enter Your Password
SIGN IN
Don't have an account? Sign up here

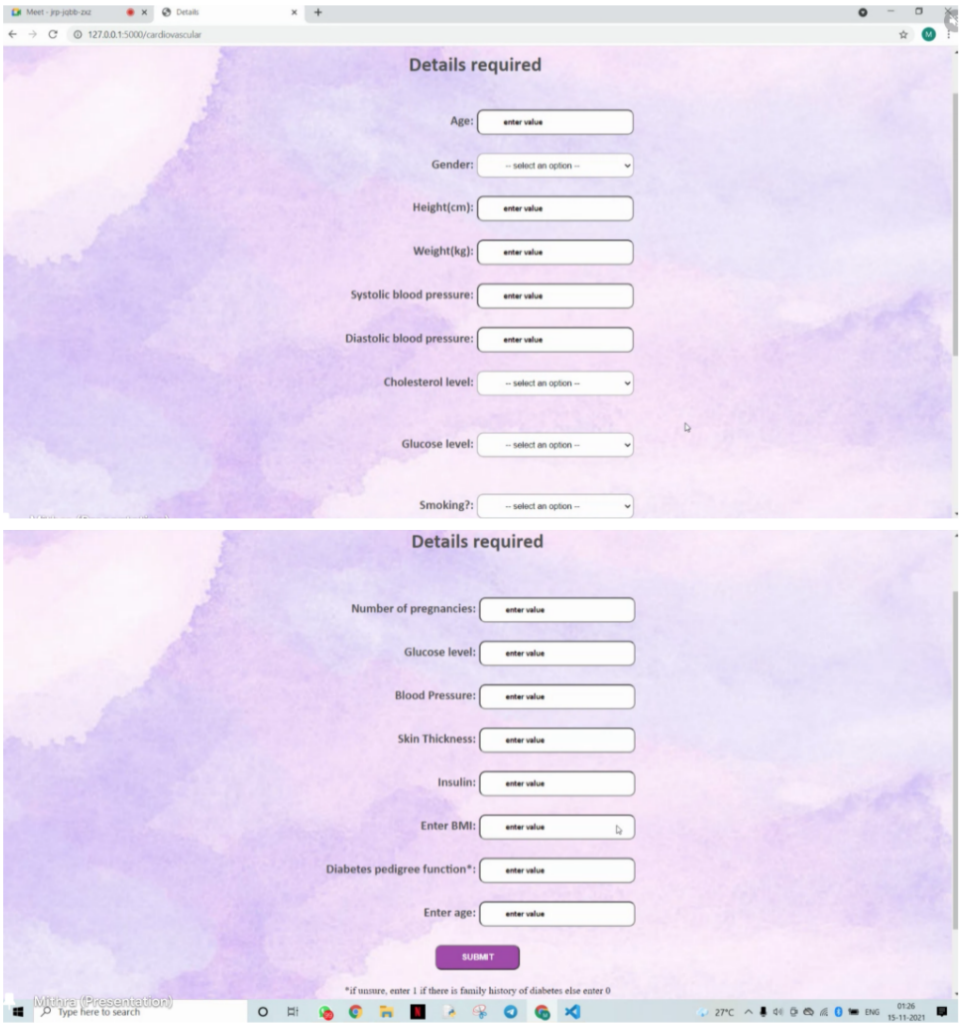

CHRONIC DISEASE PREDICTOR
An ML based application
Register
Enter Your Username
Enter Your Password
Enter Your Email ID
SIGN UP
Already have an account? Sign in here


127.0.0.1:5000/login
MENU
Hello mithra
What would you like to check out?
STROKE RISK
GENERAL CARDIOVASCULAR DISEASE
DIABETES RISK
Logout


127.0.0.1:5000/cardio
Details required
Gender:
female
Age:
45
Experienced hypertension?:
no
Any prior heart disease?:
no
Ever married?
yes
Work type:
-- select an option --
children
government job
never worked
private
self employment
Residence type:


Meet - jrp-jqbb-zxz
Index
127.0.0.1:5000/cardio
RESULT
Congratulations!
Your demographics, lifestyle and body parameters suggest that you're NOT AT RISK. Spread the will for a health-friendly lifestyle that you follow now and get yourself checked periodically from us or a professional.
Cheers!
Logout
Back


Meet - jrp-jqbb-zxz
Index
127.0.0.1:5000/cardio
RESULT
Your demographics, lifestyle and body parameters suggest that you're AT RISK. Visit a General Physician or a Neurologist near you to confirm, and secure your health and future!
Adopt a healthier lifestyle and get your family checked as well, as stroke may also be hereditary.
Health is wealth indeed.
Logout
Back



6.3 Screen Objects and Actions

1. *Registration:*

It admits new users into the application using text boxes for the fields username, email id and password which is then stored in the database at the backend. It also has a provision that redirects old users to the login object.

2. *Login:*

It admits old users into the portal by prompting them to fill the username and password text box fields. On verifying with the backend database, the user is directed to

the disease portal object. A provision for new users to access the 'registration' object is also present.

3. *Disease portal:*
It displays the menu of chronic illnesses for the user to choose from.

4. *Data collection portal:*
Each chronic illness in the system has a data collection object that does the work of collecting the disease parameters, feeding it to the database and the ML model for prediction

5. *Prediction:*
The final object in the UI object cascade that displays the risk triggered by the model output.

7. Requirements Matrix

This Requirements Traceability Matrix document demonstrates the relationship between requirements and other artifacts. It's used to prove that requirements have been fulfilled. It maps and traces user requirements with test cases. It captures all requirements proposed by the client and requirement traceability in a single table and is delivered at the conclusion of the Software development life cycle. A simplified matrix for our system is shown below:

S.No	*Requirement Description*	*Module to test it with*
1.	*Username, Email ID, Password*	*User Module*
2.	*The Choice of disease (from the click of the Radio button)*	*Disease Portal module*
3.	*Disease parameters unique to each disorder*	*Data collection module*

8. APPENDIX

Refer 3.2 System architecture for the Data flow diagrams.
