PROJECT REPORT

Statistical Machine Learning Approach To Liver Disease Prediction

Submitted by

PNT2022TMID41146

K Kasiprasath

M Sathiyanathan

M Hemachandran

D Manikandan

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Diseases related to the liver and heart are becoming more and more familiar with time. Future developments in technology will only lead to an increase in these. Even if more individuals are becoming health-conscious nowadays and are enrolling in dancing and yoga courses, the issue will persist for a long time due to sedentary lifestyles and extravagances that are always being introduced and improved. The ageing of the population and the rise in the incidence and prevalence of chronic diseases result in an increased risk of liver disease-related hospitalisation or death.

Early prediction of liver disease is very important to save human life and take proper steps to control the disease. The correct prediction of liver disease can prevent life threats, and incorrect prediction can prove to be fatal at the same time. Machine learning algorithms have been playing a vital role in solving complex, highly nonlinear classification and prediction problems. Further, different machine learning algorithms are ensembled in order to increase the classification and prediction accuracy. Here different machine learning techniques like

K-Nearest Neighbours (KNN), Decision Tree (DT) and Random Forest (RF) algorithms have been ensembled using the majority voting technique to predict Liver diseases.

1.2 PURPOSE

The purpose of this project is to predict the presence of liver disease with high efficiency. Instead of using individual classifier algorithms, an ensemble model that combines KNN, DT, RF is used to increase accuracy. The main objective of this project is to make socially healthy living. Early prediction of liver disease using classification algorithms is an efficacious task that can help the doctors to diagnose the disease within a short duration of time. It is also helpful for the doctors to get patients treated at the earliest. It is cost effective and user-friendly.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

Today, everyone's health is a very essential concern, so it is necessary to offer medical services that are freely accessible to everyone. Discovering the existence of liver disease at an early stage is a complex task for the doctors. But early treatment may give the liver time to heal. Since there are pre-existing processes to analyze the patient data and the classifier data, the more important fact here is to predict the same with conclusive result with a higher rate of accuracy. Existing Models based on feature selection and classification raised some issues regarding with training dataset and Test dataset.

2.2 REFERENCES

Title: Automated Prediction of Liver Disease using Machine Learning (ML) Algorithms
Author: A. Srivastava, V. V. Kumar, M. T. R and V. Vivek

In this paper, Information systems and strategic tools have lately been offered as new strategies in medical research to improve the disease detection process. A variety of machine learning (ML) algorithms are being used to predict liver diseases. They recommend employing Logistics Regression (LR), Naive Bayes Model (NB), K-Nearest Neighbor (Knn) in the project. The data was separated into two categories: patients with liver disease and sicknesses the most accurate machine learning method was used to predict the final result. The different algorithms are compared against various performance metrics and the best one is identified.

Title : A Machine Learning Based Framework to Identify and Classify Non- alcoholic Fatty Liver Disease in a Large-Scale Population
Author: Weidong Ji, Mingyue Xue, Yushan Zhang, Hua Yao, Yushan Wang

Non-alcoholic fatty liver disease (NAFLD) is a common serious health problem worldwide, which lacks efficient medical treatment. They aimed to develop and validate the machine learning (ML) models which could be used to the accurate screening of large number of people. This paper included 304,145 adults who have joined in the national physical examination and used their questionnaire and physical measurement parameters as model's candidate covariates. Absolute shrinkage and selection operator (LASSO) was used to feature selection from candidate covariates, then four ML algorithms were used to build the screening model for NAFLD, used a classifier with the best performance to output the importance score of the covariate in NAFLD.

Title: Liver Disease Diagnosis Using Machine Learning
Author: Manas Minnoor, Veeky Baths

This paper evaluates the performance of various supervised machine learning algorithms such as Logistic Regression, K-Nearest Neighbors (KNN), Extra Trees, LightGBM as well as a Multilayer Perceptron (MLP) neural network in the detection and diagnosis of liver disease. A total of 11 attributes are used to train the models. The usage of machine learning algorithms alongside human medical expertise may help drastically reduce errors in clinical diagnosis. This paper establishes the feasibility of applying machine learning in various medical fields including the diagnosis of other diseases.

Title: Statistical Machine Learning Approaches to Liver Disease Prediction
Author: Fahad Mostafa, Easin Hasan,Morgan Williamson,Hafiz khan

The study compared binary classifier machine learning algorithms (i.e., artificial neural network, random forest (RF), and support vector machine), which were utilized on a published liver disease data set to classify individuals with liver diseases, which will allow health professionals to make a better diagnosis. The synthetic minority oversampling technique was applied to oversample the minority class to regulate overfitting problems. The purpose of this study was to extract significant predictors for liver disease from the medical analysis of 615 humans using ML algorithms. Thus, this suggests that ML methods predict liver disease by incorporating the risk factors, which may improve the inference-based diagnosis of patients.

Title: Diagnosis of Liver Disease using Machine Learning Models Author: A. Sivasangari; Baddigam Jaya Krishna Reddy; Annamareddy Kiran; P. Ajitha

Liver-related disease poses more problems for people living and is more important nowadays to recognize the causes, and identification phase. So, for early detection of liver disease, an automated program is needed to build with more accuracy and reliability. Specific machine learning models are developed for this purpose to predict the disease. In this paper, the methods of Support Vector Machines (SVM), Decision Tree (DT) and Random Forest (RF) is proposed to predict liver disease with better precision, accuracy and reliability.

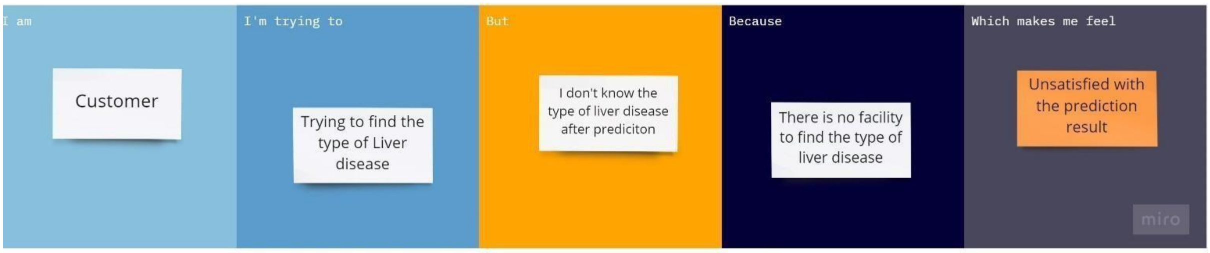
2.3 PROBLEM STATEMENT DEFINITION

Customer Problem Statement Template :

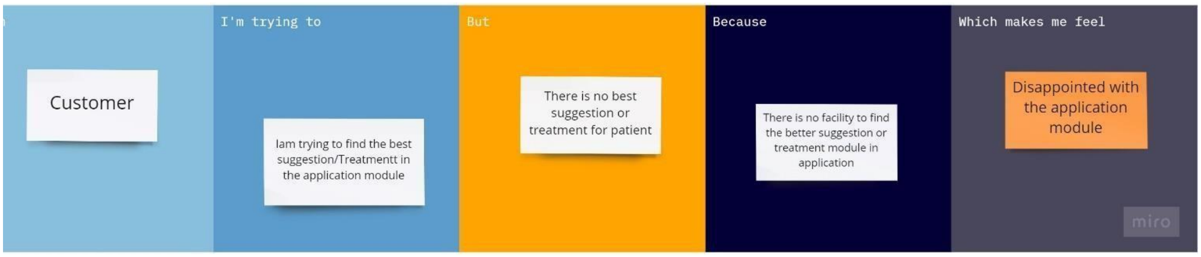
Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Customer	To find the type of liver disease	I can't find the type of liver disease I'am affected	There is no facility to find the type of liver disease	Unsatisfied with the prediction result



I am	I'm trying to	But	Because	Which makes me feel
Customer	Trying to find the type of Liver disease	I don't know the type of liver disease after prediciton	There is no facility to find the type of liver disease	Unsatisfied with the prediction result



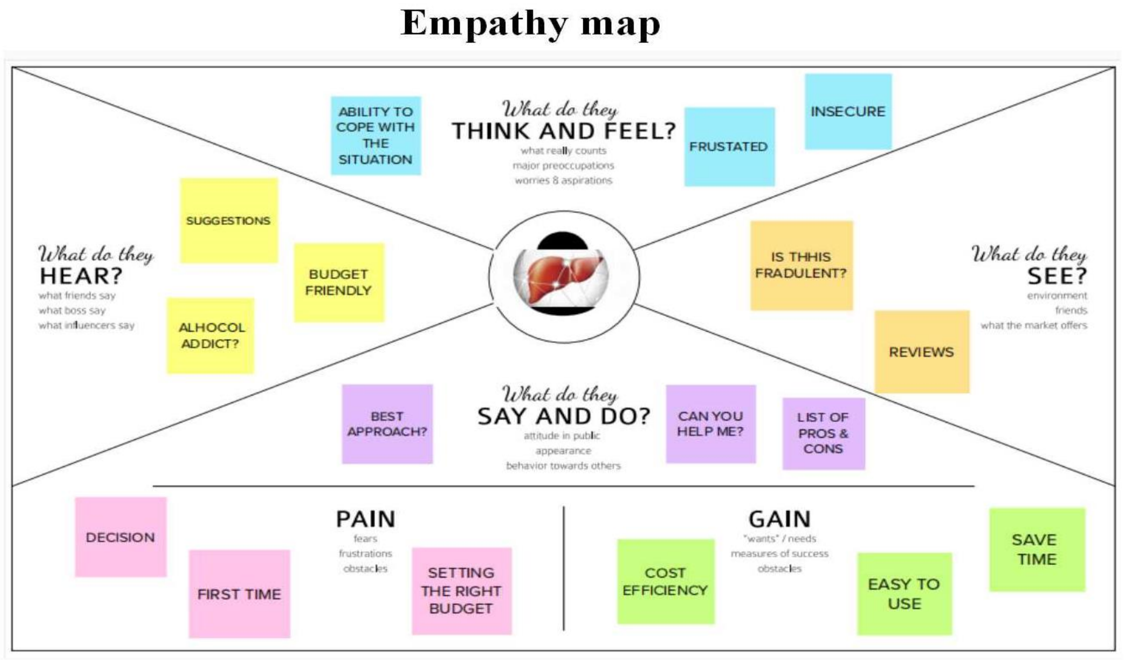
	I'm trying to	But	Because	Which makes me feel
Customer	Iam trying to find the best suggestion/Treatmentt in the application module	There is no best suggestion or treatment for patient	There is no facility to find the better suggestion or treatment module in application	Disappointed with the application module

PS-2	Customer	Find the best suggestion /treatment in application module	There is no better suggestion/treatment for patient	There is no facility to find the better suggestion/module in application module	Disappointed with the application module

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



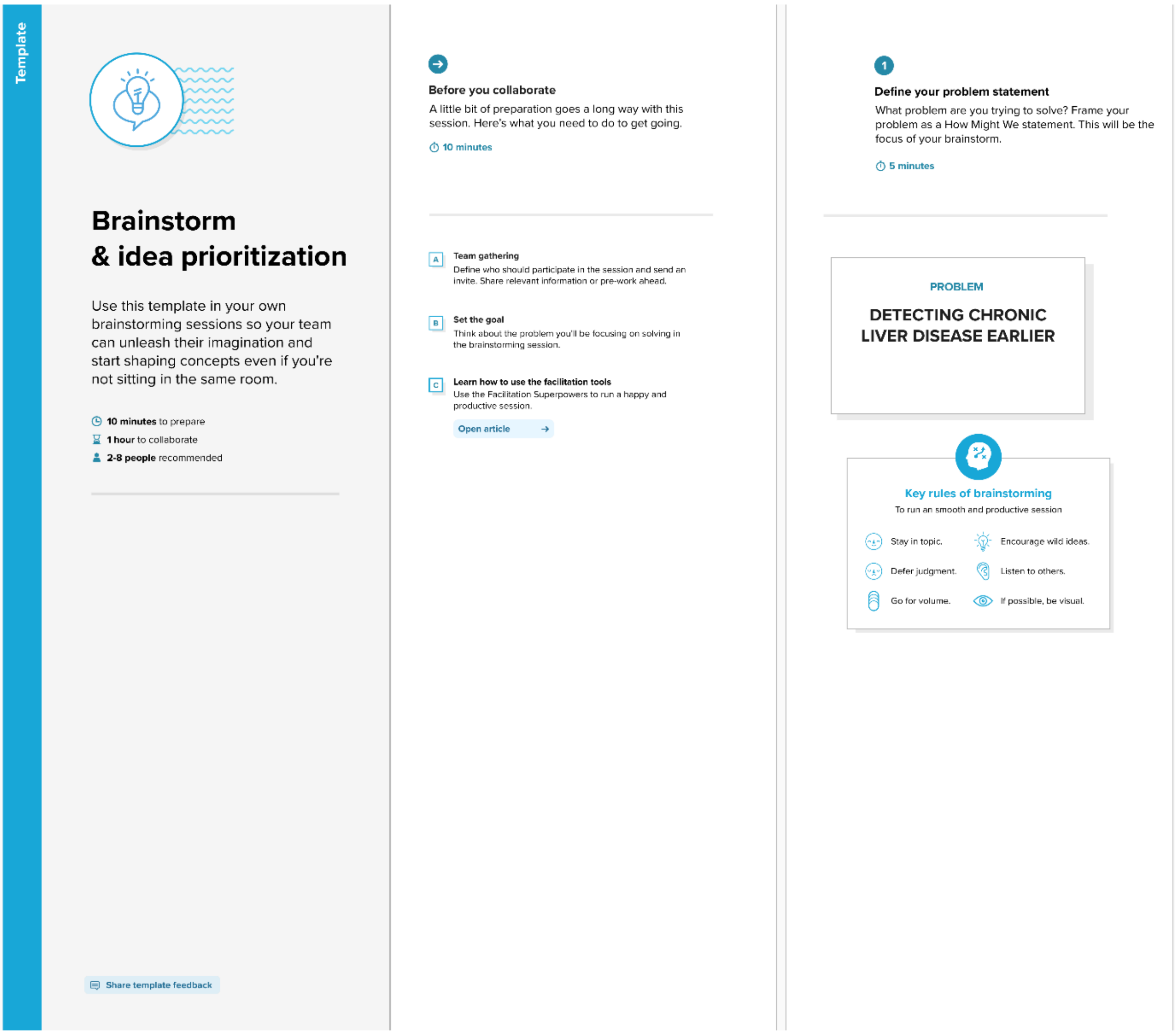
IDEATION & BRAINSTORMING

Brainstorm & Idea Prioritization Template :

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 **Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

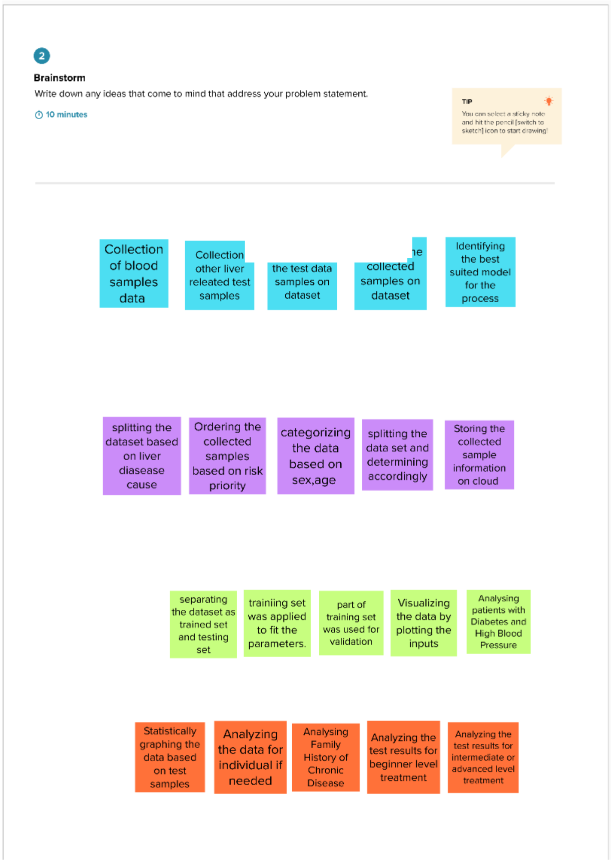
DETECTING CHRONIC LIVER DISEASE EARLIER

Key rules of brainstorming

To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm

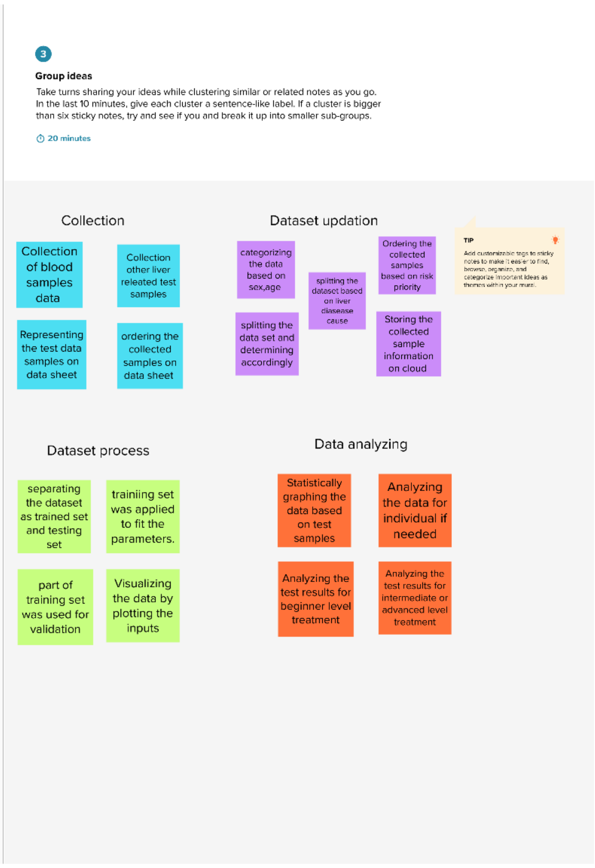
Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Collection of blood samples data
Collection other liver releated test samples
the test data samples on dataset
collected samples on dataset
Identifying the best suited model for the process
splitting the dataset based on liver diasease cause
Ordering the collected samples based on risk priority
categorizing the data based on sex,age
splitting the data set and determining accordingly
Storing the collected sample information on cloud
separating the dataset as trained set and testing set
trainiing set was applied to fit the parameters.
part of training set was used for validation
Visualizing the data by plotting the inputs
Analysing patients with Diabetes and High Blood Pressure
Statistically graphing the data based on test samples
Analyzing the data for individual if needed
Analysing Family History of Chronic Disease
Analyzing the test results for beginner level treatment
Analyzing the test results for intermediate or advanced level treatment



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

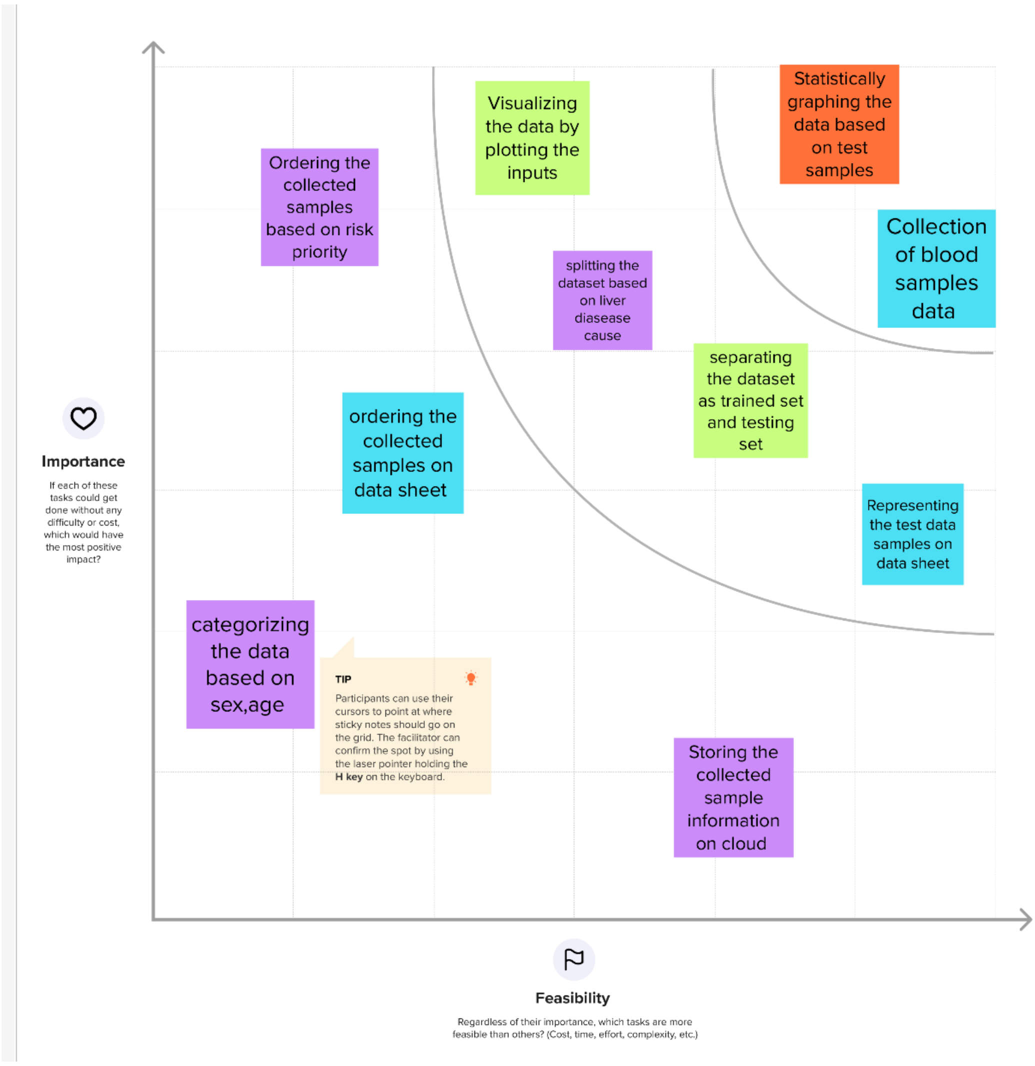
20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Collection
Collection of blood samples data
Collection other liver releated test samples
Representing the test data samples on data sheet
ordering the collected samples on data sheet
Dataset updation
categorizing the data based on sex,age
splitting the dataset based on liver diasease cause
Ordering the collected samples based on risk priority
splitting the data set and determining accordingly
Storing the collected sample information on cloud
Dataset process
separating the dataset as trained set and testing set
trainiing set was applied to fit the parameters.
part of training set was used for validation
Visualizing the data by plotting the inputs
Data analyzing
Statistically graphing the data based on test samples
Analyzing the data for individual if needed
Analyzing the test results for beginner level treatment
Analyzing the test results for intermediate or advanced level treatment

Step-3: Idea Prioritization


Ordering the collected samples based on risk priority
Visualizing the data by plotting the inputs
Statistically graphing the data based on test samples
Collection of blood samples data
splitting the dataset based on liver diasease cause
separating the dataset as trained set and testing set
ordering the collected samples on data sheet
Representing the test data samples on data sheet
categorizing the data based on sex,age
Storing the collected sample information on cloud
Importance
Feasibility

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

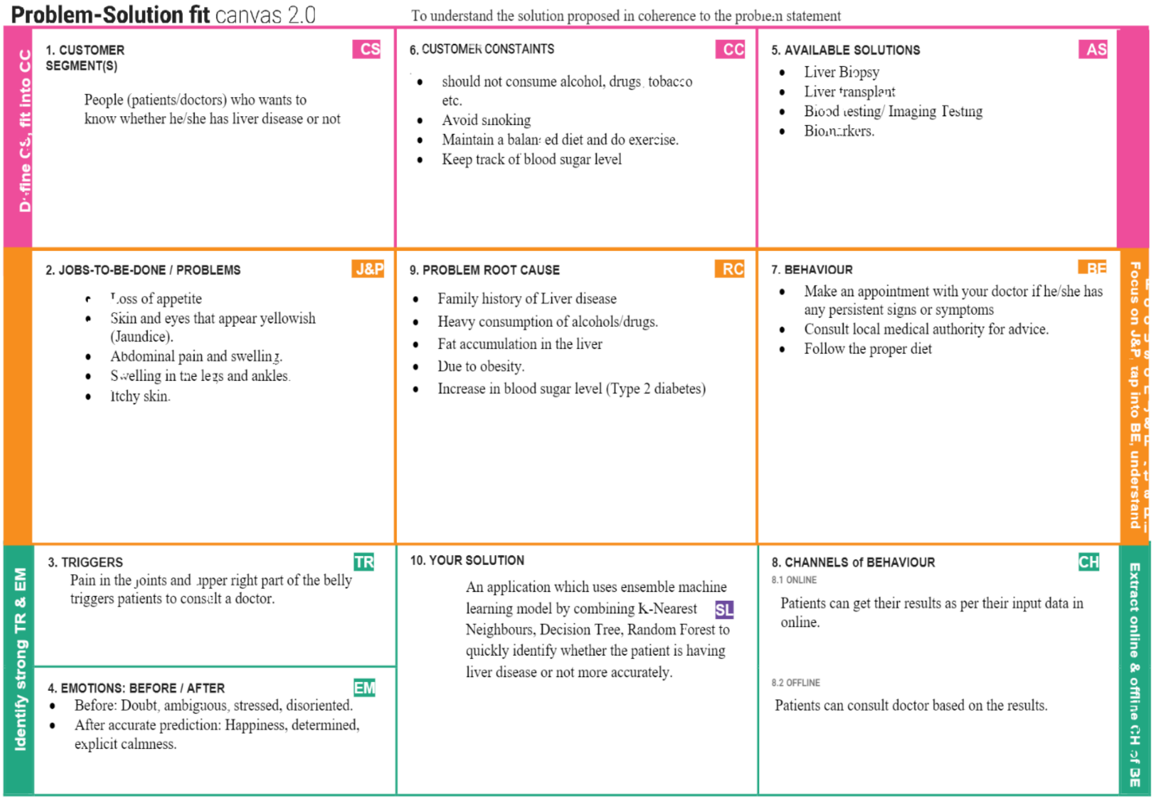
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The number of patients with liver disease has been steadily rising as a result of heavy alcohol usage, exposure to dangerous gases, and use of contaminated food. Health Care Professionals need to obtain patient samples to identify the liver disease, which could be expensive both money and time. The key problem is doctor cannot provide a diagnosis based on test variation results.
2.	Idea / Solution description	The application will accurately and quickly identify which individuals have liver disease and which ones do not by using patient records that include blood test report results.

3.	Novelty / Uniqueness	To predict the presence of Liver disease with high efficiency. Instead of using individual classifier algorithms, an ensemble model that combines KNN, DT, RF is used to increase accuracy. Model is deployed using Heroku cloud platform.
4.	Social Impact / Customer Satisfaction	The proposed system will make socially healthy living by decreasing mortality rate. It is also helpful for the doctors to get patients treated at the earliest.
5.	Business Model (Revenue Model)	• Health Care Sector (Hospitals). • Can generate revenue through direct customers. • Can collaborate with health care sector and generate revenue from their customers.
6.	Scalability of the Solution	It is cost effective and user friendly.

PROBLEM SOLUTION FIT



Problem-Solution fit canvas 2.0

To understand the solution proposed in coherence to the problem statement

Define CS, fit into CC
Focus on J&P, tap into BE, understand
Identify strong TR & EM
Extract online & offline CH of BE

1. CUSTOMER SEGMENT(S) CS

People (patients/doctors) who wants to know whether he/she has liver disease or not

6. CUSTOMER CONSTAINTS CC

- should not consume alcohol, drugs, tobacco etc.
- Avoid smoking
- Maintain a balanced diet and do exercise.
- Keep track of blood sugar level

5. AVAILABLE SOLUTIONS AS

- Liver Biopsy
- Liver transplant
- Blood testing/ Imaging Testing
- Biomarkers.

2. JOBS-TO-BE-DONE / PROBLEMS J&P

- Loss of appetite
- Skin and eyes that appear yellowish (Jaundice).
- Abdominal pain and swelling.
- Swelling in the legs and ankles.
- Itchy skin.

9. PROBLEM ROOT CAUSE RC

- Family history of Liver disease
- Heavy consumption of alcohols/drugs.
- Fat accumulation in the liver
- Due to obesity.
- Increase in blood sugar level (Type 2 diabetes)

7. BEHAVIOUR BE

- Make an appointment with your doctor if he/she has any persistent signs or symptoms
- Consult local medical authority for advice.
- Follow the proper diet

3. TRIGGERS TR

Pain in the joints and upper right part of the belly triggers patients to consult a doctor.

4. EMOTIONS: BEFORE / AFTER EM

- Before: Doubt, ambiguous, stressed, disoriented.
- After accurate prediction: Happiness, determined, explicit calmness.

10. YOUR SOLUTION SL

An application which uses ensemble machine learning model by combining K-Nearest Neighbours, Decision Tree, Random Forest to quickly identify whether the patient is having liver disease or not more accurately.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Patients can get their results as per their input data in online.

8.2 OFFLINE

Patients can consult doctor based on the results.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMEN

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Patients with symptoms of Liver disease	Patient having liver disease dataset contains age of the patient, gender, Alkaline Phosphotase, Total Bilirubin etc.
FR-2	Predicting the liver disease using Ensemble model	Machine Learning
FR-3	Pre-processing of liver disease dataset	Principal Component Analysis (PCA)
FR-4	Ensemble Model Training	K-Nearest Neighbours, Decision Tree, Random Forest
FR-5	Model Evaluation	Predicting the accuracy of our ensemble model and comparing it with other algorithms such as Support Vector Machine (SVM) etc.
FR-6	Model Deployment	Deploying the Machine learning model in cloud platform.

4.2 NON-FUNCTIONAL REQUIREMENTS

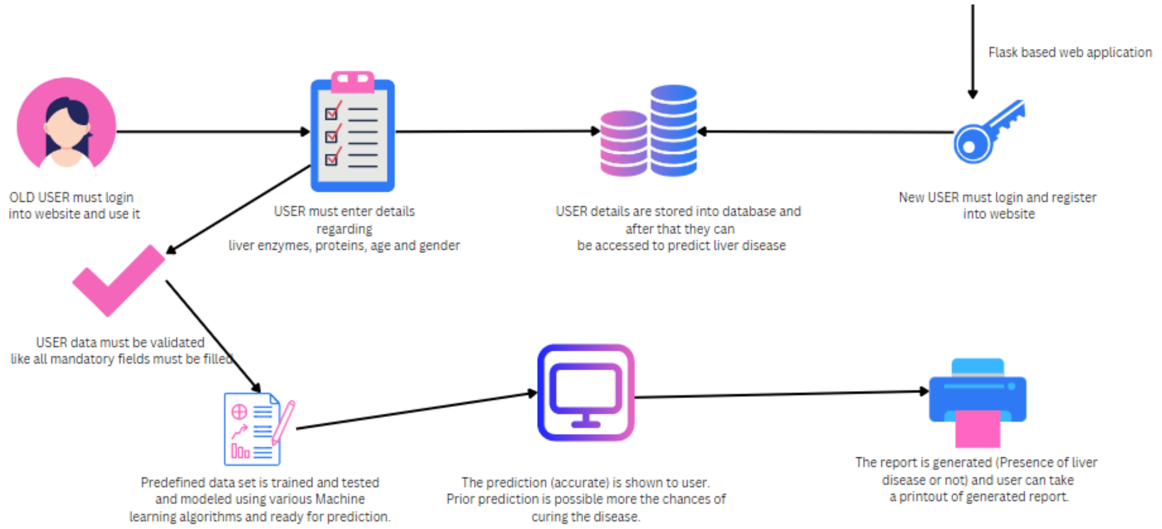
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The system provides a natural interaction with the users. It is user-friendly.
NFR-2	**Security**	The model enables with the high security system, as the user’s data won’t be shared to the other sources. Only the authorised person can access the system.

NFR-3	**Reliability**	As the system is build using a rich Ensemble model, mostly all the user input can be processed without failure in 95 per cent of use cases and since all the processing are done on cloud, the system is consider to be highly reliable.
NFR-4	**Performance**	Our system should run on 32 bit (x86) or 64 bit (x64) Dual-core 2.66-GHZ or faster processor.
NFR-5	**Availability**	The system should be available for the duration of the user access, until the user terminate the access. The system response to request of the user in less time and the recovery is done is less time.
NFR-6	**Scalability**	It provides an efficient outcome and has the ability to increase or decrease the performance of the system based on the datasets. It is cost effective and user friendly.

CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



Flask based web application

OLD USER must login into website and use it

USER must enter details regarding liver enzymes, proteins, age and gender

USER details are stored into database and after that they can be accessed to predict liver disease

New USER must login and register into website

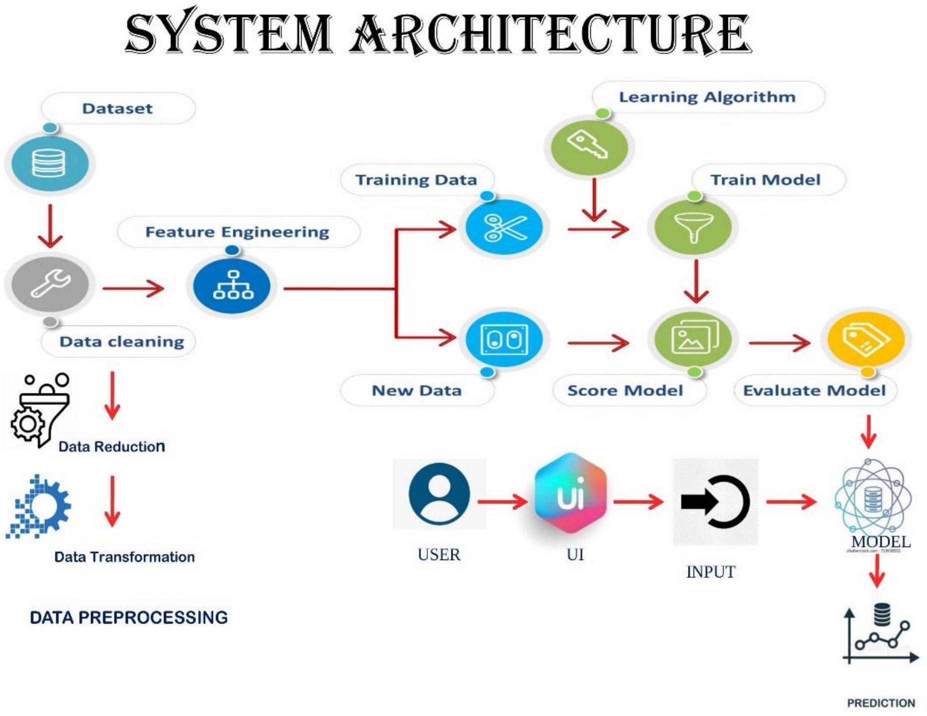
USER data must be validated like all mandatory fields must be filled

Predefined data set is trained and tested and modeled using various Machine learning algorithms and ready for prediction.

The prediction (accurate) is shown to user. Prior prediction is possible more the chances of curing the disease.

The report is generated (Presence of liver disease or not) and user can take a printout of generated report.

5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through website	I can register the website	Low	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login into the website	Medium	Sprint-2
	Dashboard	USN-5	As a user, I can access dashboard	I can get into the dashboard	High	Sprint-2
Customer (web user)		USN-6	As a user, I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	I can predict accurate presence of liver disease based on liver enzymes, proteins, age and gender.	High	Sprint-1
Customer Care Executive		USN-7	As a user, I can get support from admin in case of any issues and also some recommendations.	I can get support from admin in case of any issues and also some recommendations.	High	Sprint-3
Administrator		USN-8	Get all issues solved whatever the issue is.	I can get all issues solved whatever the issue is mostly regarding prediction.	High	Sprint-4

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	High	Kasiprasath K
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	5	High	Sathiyanathan M
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	10	High	Sathiyanathan M
Sprint-2	Input Necessary Details	USN-4	As a user, I can give Input Details to Predict Likeliness of Liver Disease.	15	High	Sathiyanathan M
Sprint-2	Data preprocessing	USN-5	Transform raw data into suitable format for prediction.	5	High	Hemachandran M
Sprint-3	Prediction of Liver Disease	USN-6	As a user, I can predict Liver Disease using machine learning model.	15	High	Kasiprasath K
Sprint-3		USN-7	As a user, I can get accurate prediction of liver disease.		Medium	Manikandan D
Sprint-4	Review	USN-8	As a user, I can give feedback of the application.	20	High	Kasiprasath K
Project Tracker, Velocity & Burn down	Total Story Points	Duration	Sprint Start Date	Sprint End	Story Points Completed (as	Sprint Release Date (Actual)

Chart: (4 Marks) Sprint				Date (Planned)	on Planned End Date)	
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	18	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	17	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	18	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	16 Nov 2022	17	16 Nov 2022

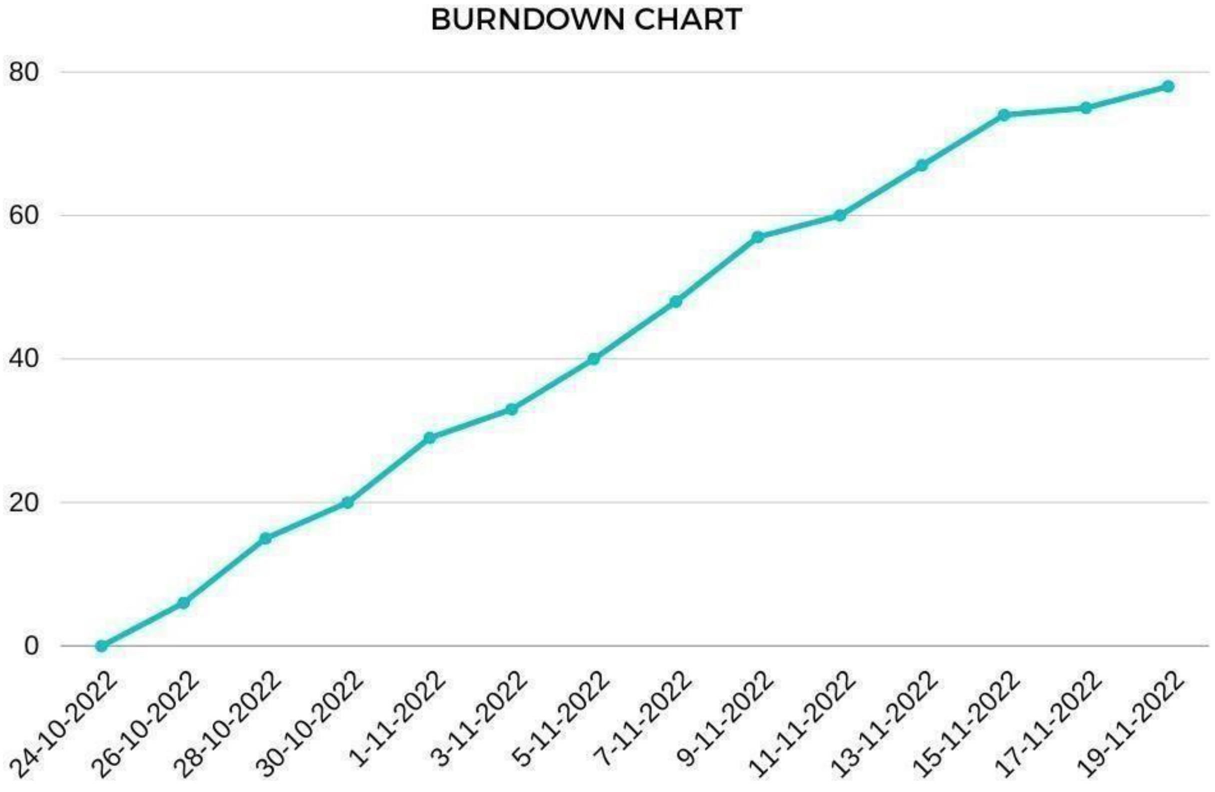
Velocity:

Imagine we have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint).

Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

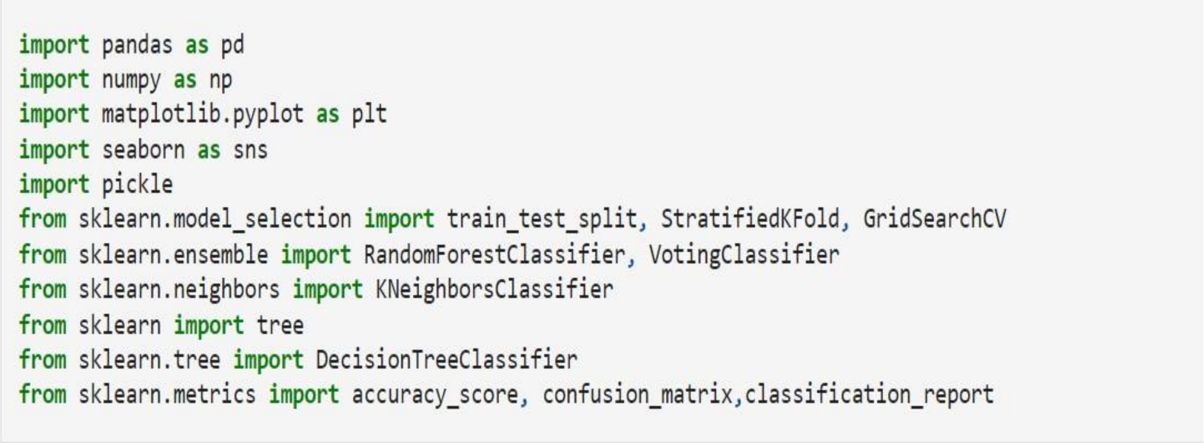
AV=Sprint duration/velocity = 6/20=0.3

Burn down Chart :

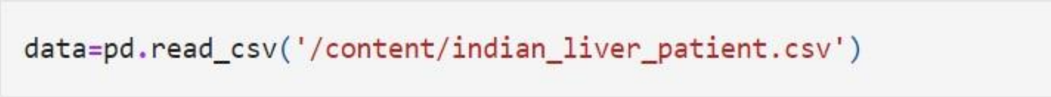


CHAPTER 7

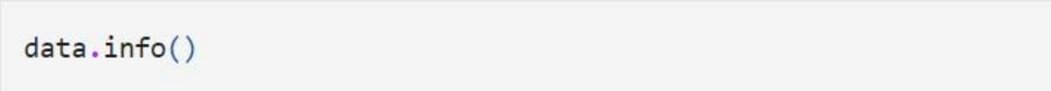
CODING & SOLUTIONING



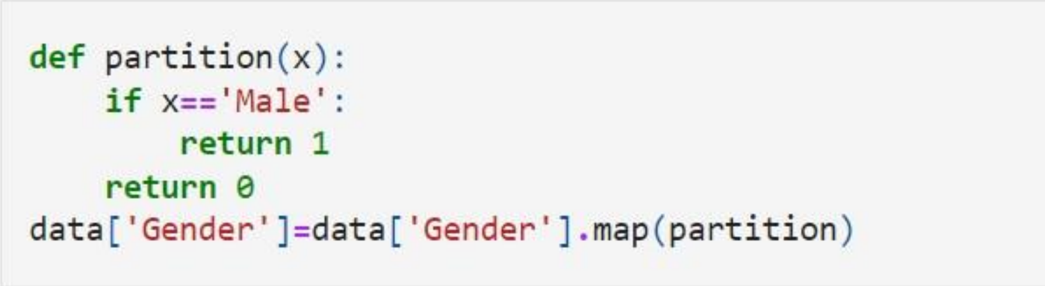
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
from sklearn.model_selection import train_test_split, StratifiedKFold, GridSearchCV
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report
```



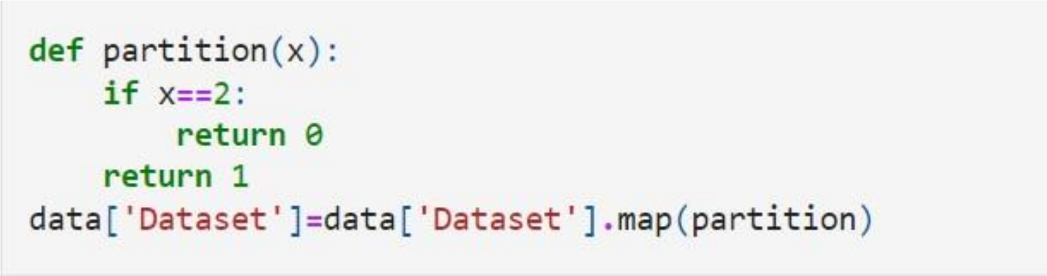
```python
data=pd.read_csv('/content/indian_liver_patient.csv')
```



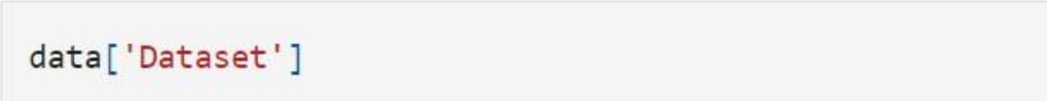
```python
data.info()
```



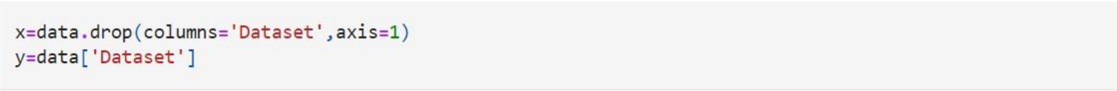
```python
def partition(x):
    if x=='Male':
        return 1
    return 0
data['Gender']=data['Gender'].map(partition)
```



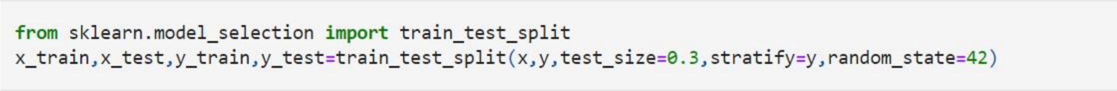
```python
def partition(x):
    if x==2:
        return 0
    return 1
data['Dataset']=data['Dataset'].map(partition)
```



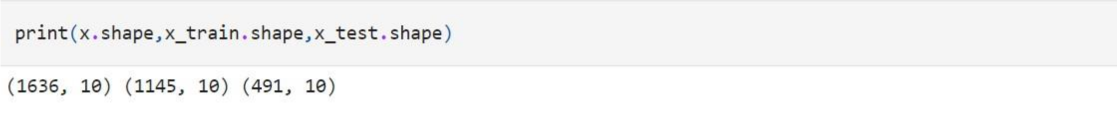
```python
data['Dataset']
```



```
x=data.drop(columns='Dataset',axis=1)
y=data['Dataset']
```

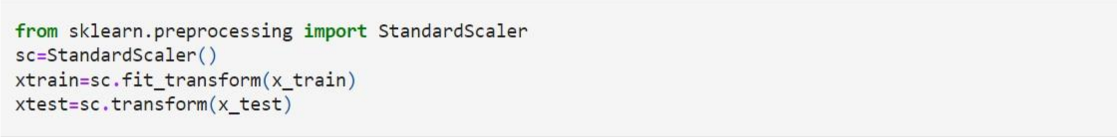


```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,stratify=y,random_state=42)
```

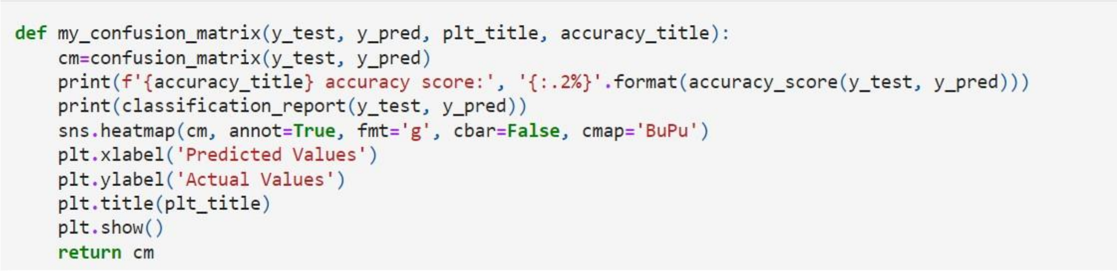


```
print(x.shape,x_train.shape,x_test.shape)
```

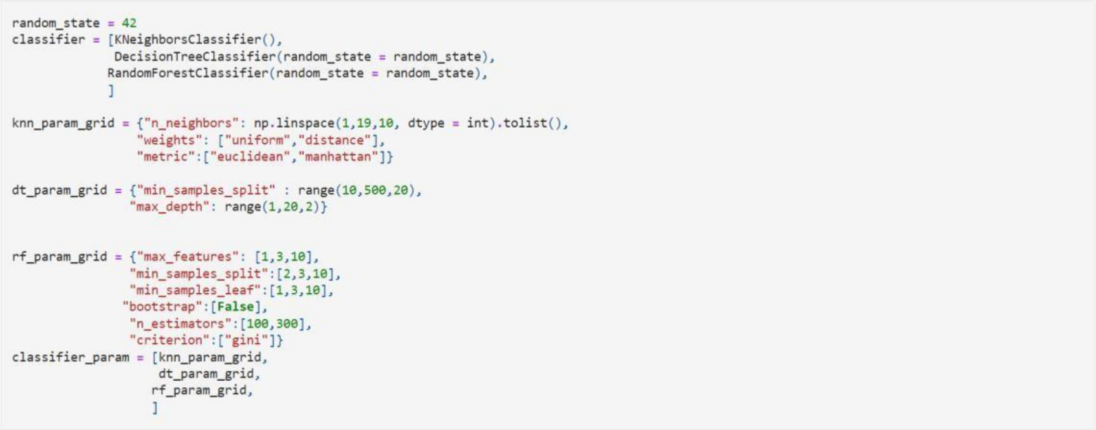
```
(1636, 10) (1145, 10) (491, 10)
```



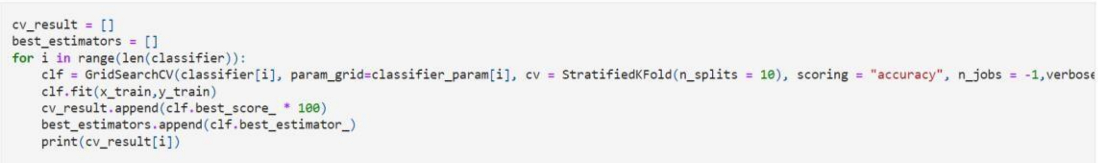
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
xtrain=sc.fit_transform(x_train)
xtest=sc.transform(x_test)
```



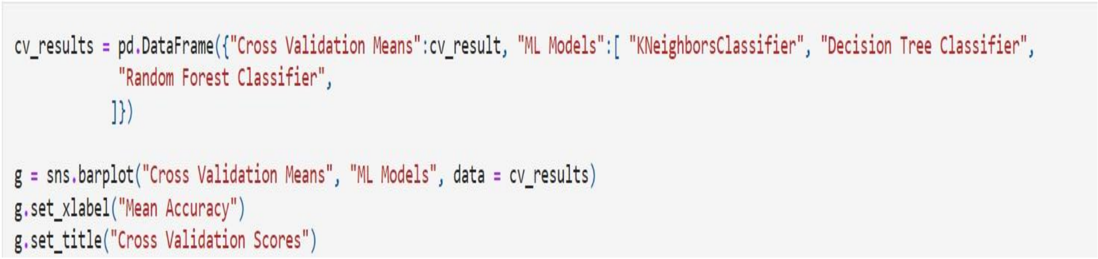
```
def my_confusion_matrix(y_test, y_pred, plt_title, accuracy_title):
    cm=confusion_matrix(y_test, y_pred)
    print(f'{accuracy_title} accuracy score:', '{:.2%}'.format(accuracy_score(y_test, y_pred)))
    print(classification_report(y_test, y_pred))
    sns.heatmap(cm, annot=True, fmt='g', cbar=False, cmap='BuPu')
    plt.xlabel('Predicted Values')
    plt.ylabel('Actual Values')
    plt.title(plt_title)
    plt.show()
    return cm
```



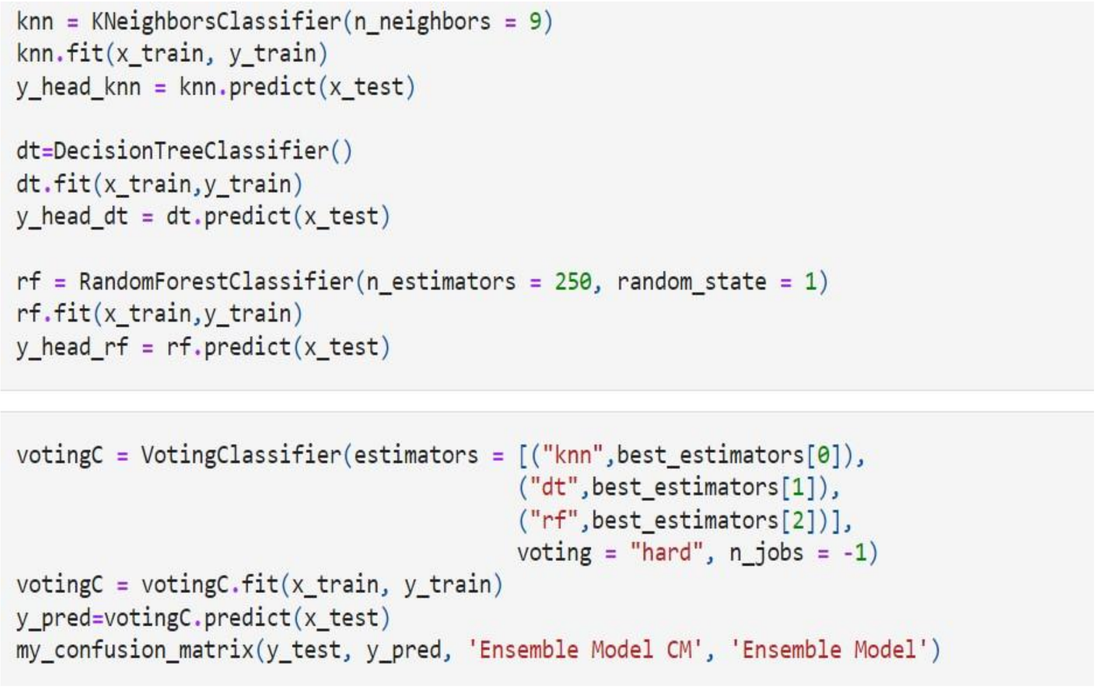
```
random_state = 42
classifier = [KNeighborsClassifier(),
              DecisionTreeClassifier(random_state = random_state),
             RandomForestClassifier(random_state = random_state),
             ]

knn_param_grid = {"n_neighbors": np.linspace(1,19,10, dtype = int).tolist(),
                 "weights": ["uniform","distance"],
                 "metric":["euclidean","manhattan"]}

dt_param_grid = {"min_samples_split" : range(10,500,20),
                "max_depth": range(1,20,2)}


rf_param_grid = {"max_features": [1,3,10],
                "min_samples_split":[2,3,10],
                "min_samples_leaf":[1,3,10],
               "bootstrap":[False],
                "n_estimators":[100,300],
                "criterion":["gini"]}
classifier_param = [knn_param_grid,
                     dt_param_grid,
                    rf_param_grid,
                    ]
```



```
cv_result = []
best_estimators = []
for i in range(len(classifier)):
    clf = GridSearchCV(classifier[i], param_grid=classifier_param[i], cv = StratifiedKFold(n_splits = 10), scoring = "accuracy", n_jobs = -1,verbose
    clf.fit(x_train,y_train)
    cv_result.append(clf.best_score_ * 100)
    best_estimators.append(clf.best_estimator_)
    print(cv_result[i])
```



```
cv_results = pd.DataFrame({"Cross Validation Means":cv_result, "ML Models":[ "KNeighborsClassifier", "Decision Tree Classifier",
             "Random Forest Classifier",
            ]})

g = sns.barplot("Cross Validation Means", "ML Models", data = cv_results)
g.set_xlabel("Mean Accuracy")
g.set_title("Cross Validation Scores")
```



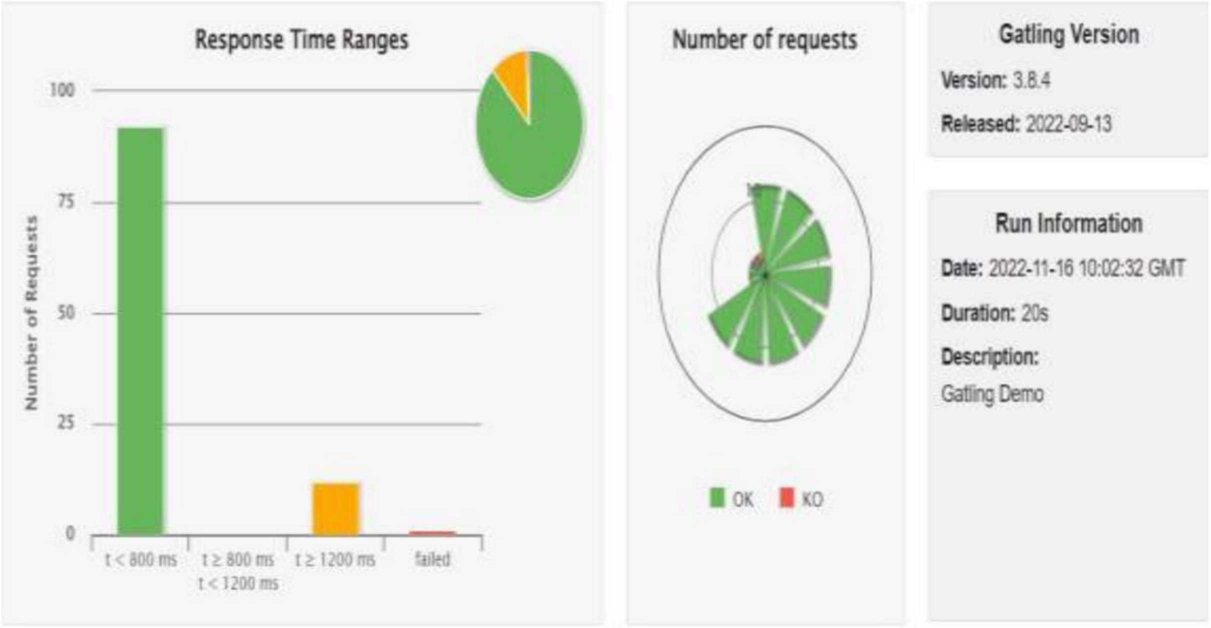
```
knn = KNeighborsClassifier(n_neighbors = 9)
knn.fit(x_train, y_train)
y_head_knn = knn.predict(x_test)

dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_head_dt = dt.predict(x_test)

rf = RandomForestClassifier(n_estimators = 250, random_state = 1)
rf.fit(x_train,y_train)
y_head_rf = rf.predict(x_test)
```

```
votingC = VotingClassifier(estimators = [("knn",best_estimators[0]),
                                        ("dt",best_estimators[1]),
                                        ("rf",best_estimators[2])],
                                        voting = "hard", n_jobs = -1)
votingC = votingC.fit(x_train, y_train)
y_pred=votingC.predict(x_test)
my_confusion_matrix(y_test, y_pred, 'Ensemble Model CM', 'Ensemble Model')
```

CHAPTER 8

TESTING

8.1 TEST CASES



Gatling Version

Version: 3.8.4

Released: 2022-09-13

Run Information

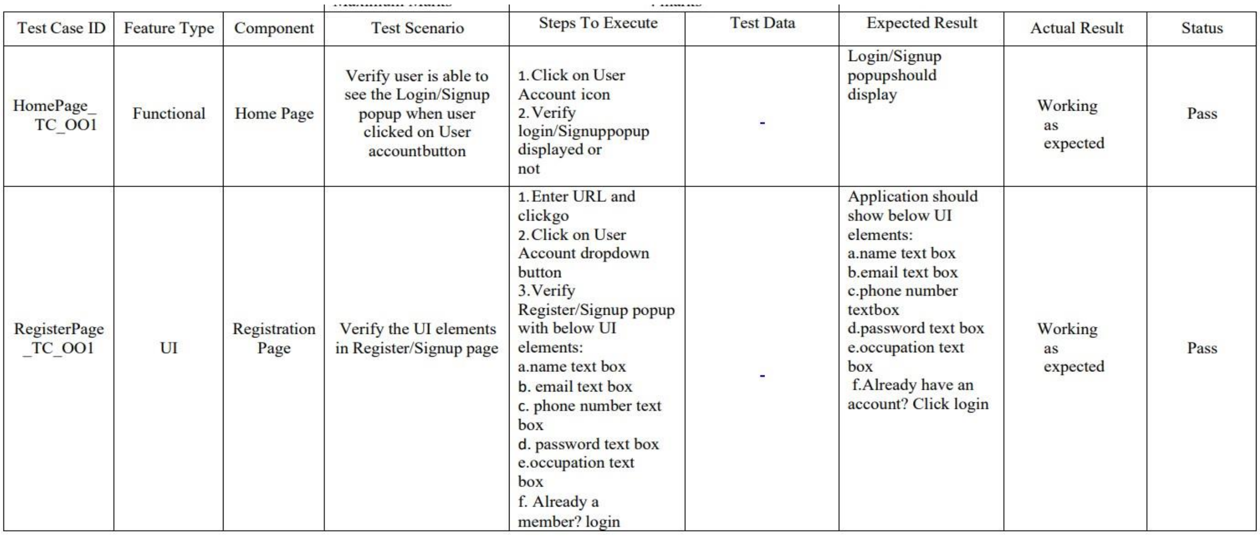
Date: 2022-11-16 10:02:32 GMT

Duration: 20s

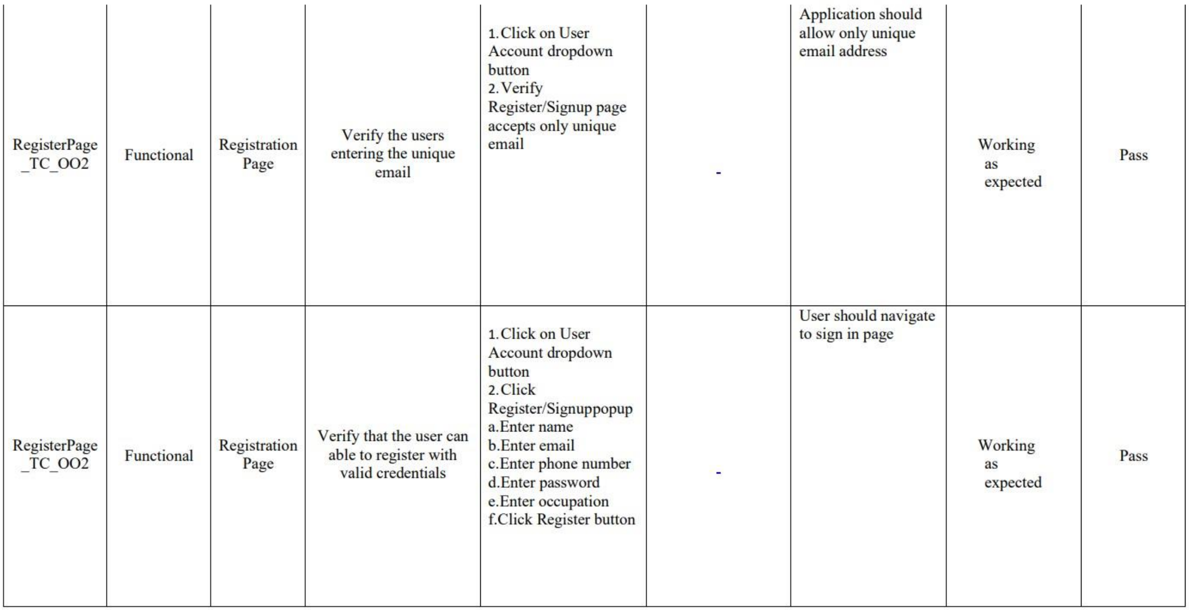
Description:

Gatling Demo

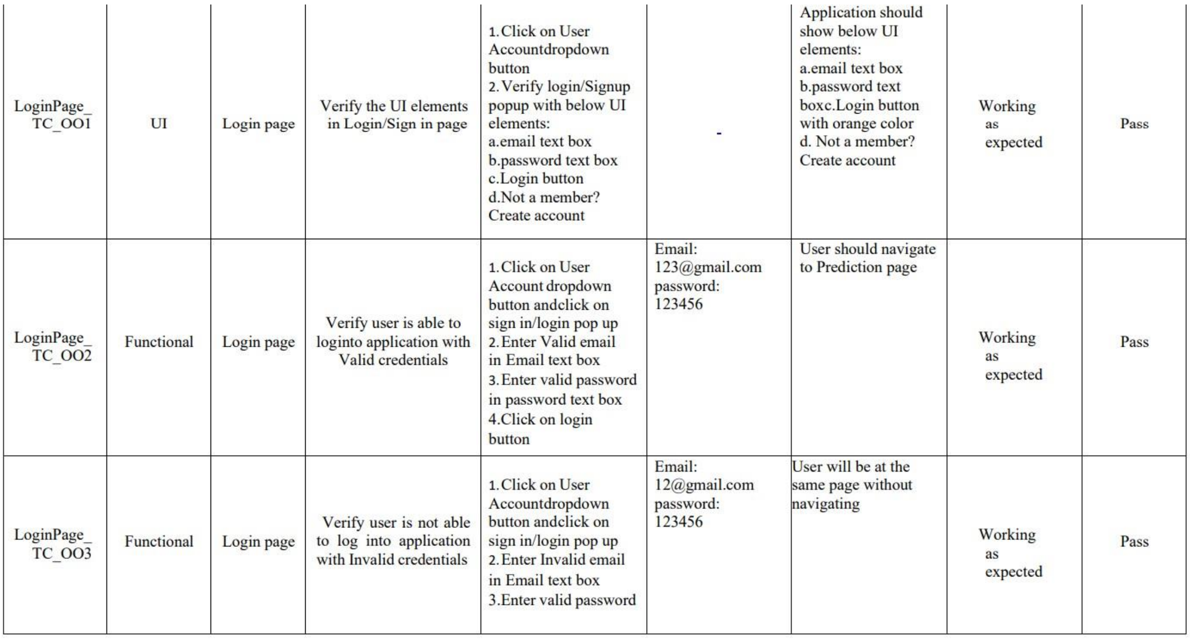
8.2 User Acceptance Testing



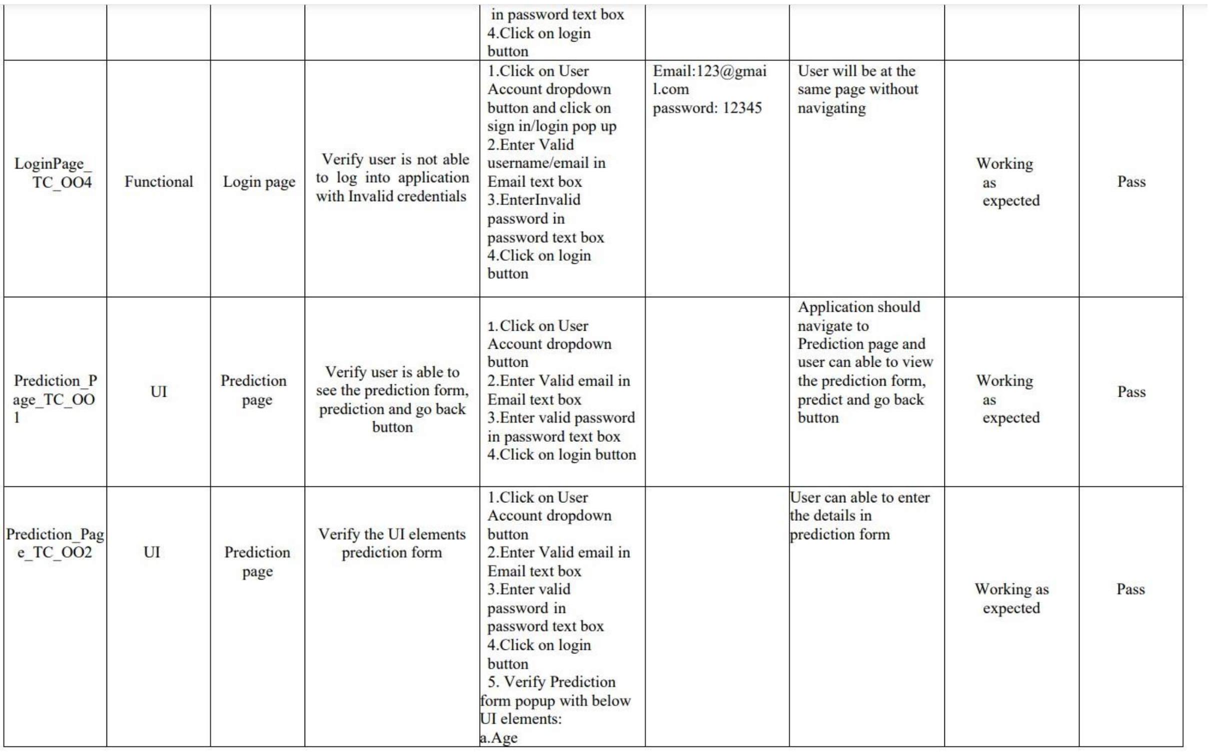
Test Case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status
HomePage_ TC_OO1	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on User accountbutton	1. Click on User Account icon 2. Verify login/Signuppopup displayed or not	-	Login/Signup popupshould display	Working as expected	Pass
RegisterPage _TC_OO1	UI	Registration Page	Verify the UI elements in Register/Signup page	1. Enter URL and clickgo 2. Click on User Account dropdown button 3. Verify Register/Signup popup with below UI elements: a.name text box b. email text box c. phone number text box d. password text box e.occupation text box f. Already a member? login	-	Application should show below UI elements: a.name text box b.email text box c.phone number textbox d.password text box e.occupation text box f.Already have an account? Click login	Working as expected	Pass



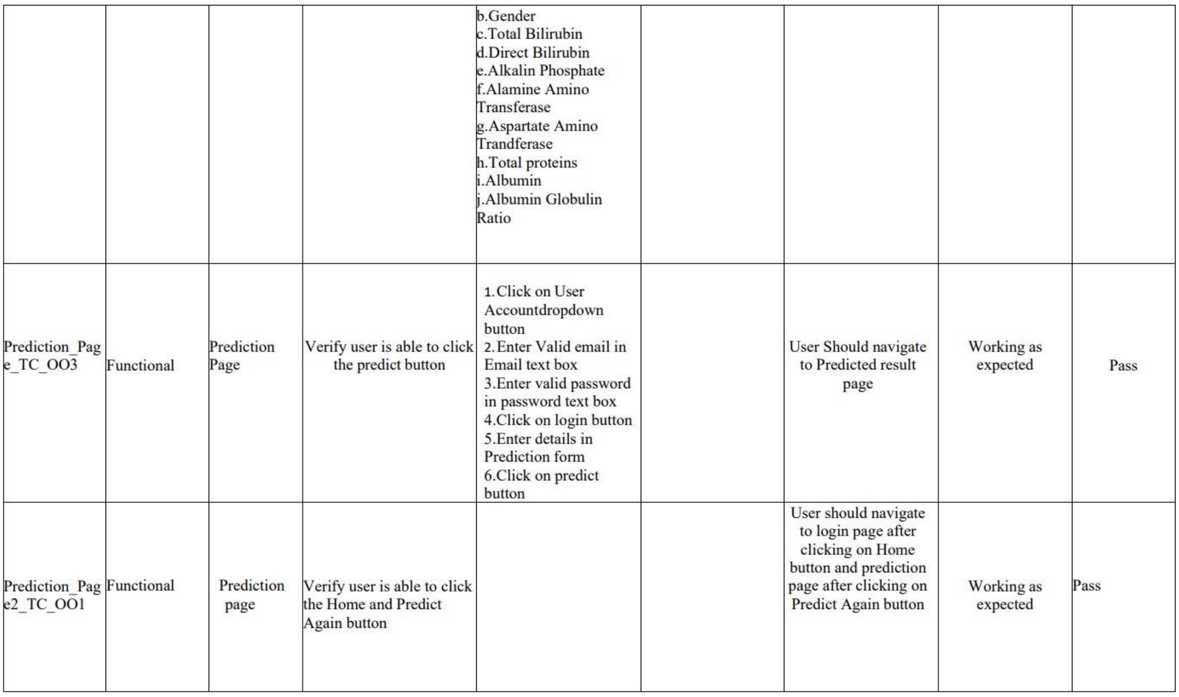
RegisterPage _TC_OO2	Functional	Registration Page	Verify the users entering the unique email	1. Click on User Account dropdown button 2. Verify Register/Signup page accepts only unique email	-	Application should allow only unique email address	Working as expected	Pass
RegisterPage _TC_OO2	Functional	Registration Page	Verify that the user can able to register with valid credentials	1. Click on User Account dropdown button 2. Click Register/Signuppopup a.Enter name b.Enter email c.Enter phone number d.Enter password e.Enter occupation f.Click Register button	-	User should navigate to sign in page	Working as expected	Pass



LoginPage_ TC_OO1	UI	Login page	Verify the UI elements in Login/Sign in page	1. Click on User Accountdropdown button 2. Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Login button d.Not a member? Create account	-	Application should show below UI elements: a.email text box b.password text boxc.Login button with orange color d. Not a member? Create account	Working as expected	Pass
LoginPage_ TC_OO2	Functional	Login page	Verify user is able to loginto application with Valid credentials	1. Click on User Account dropdown button andclick on sign in/login pop up 2. Enter Valid email in Email text box 3. Enter valid password in password text box 4.Click on login button	Email: 123@gmail.com password: 123456	User should navigate to Prediction page	Working as expected	Pass
LoginPage_ TC_OO3	Functional	Login page	Verify user is not able to log into application with Invalid credentials	1. Click on User Accountdropdown button andclick on sign in/login pop up 2. Enter Invalid email in Email text box 3.Enter valid password	Email: 12@gmail.com password: 123456	User will be at the same page without navigating	Working as expected	Pass



				in password text box 4.Click on login button				
LoginPage_ TC_OO4	Functional	Login page	Verify user is not able to log into application with Invalid credentials	1.Click on User Account dropdown button and click on sign in/login pop up 2.Enter Valid username/email in Email text box 3.EnterInvalid password in password text box 4.Click on login button	Email:123@gmail.com password: 12345	User will be at the same page without navigating	Working as expected	Pass
Prediction_Page_TC_OO 1	UI	Prediction page	Verify user is able to see the prediction form, prediction and go back button	1.Click on User Account dropdown button 2.Enter Valid email in Email text box 3.Enter valid password in password text box 4.Click on login button		Application should navigate to Prediction page and user can able to view the prediction form, predict and go back button	Working as expected	Pass
Prediction_Page_TC_OO2	UI	Prediction page	Verify the UI elements prediction form	1.Click on User Account dropdown button 2.Enter Valid email in Email text box 3.Enter valid password in password text box 4.Click on login button 5. Verify Prediction form popup with below UI elements: a.Age		User can able to enter the details in prediction form	Working as expected	Pass



				b.Gender c.Total Bilirubin d.Direct Bilirubin e.Alkalin Phosphate f.Alamine Amino Transferase g.Aspartate Amino Trandferase h.Total proteins i.Albumin j.Albumin Globulin Ratio				
Prediction_Page_TC_OO3	Functional	Prediction Page	Verify user is able to click the predict button	1.Click on User Accountdropdown button 2.Enter Valid email in Email text box 3.Enter valid password in password text box 4.Click on login button 5.Enter details in Prediction form 6.Click on predict button		User Should navigate to Predicted result page	Working as expected	Pass
Prediction_Page2_TC_OO1	Functional	Prediction page	Verify user is able to click the Home and Predict Again button			User should navigate to login page after clicking on Home button and prediction page after clicking on Predict Again button	Working as expected	Pass

8.2.1 DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won’t Fix	1	0	1	0	2
Total	6	1	4	3	14

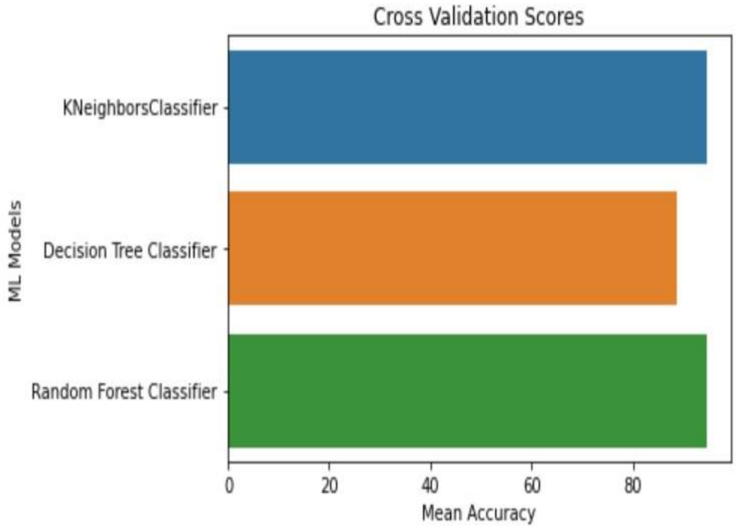
8.2.2 TEST CASE ANALYSIS

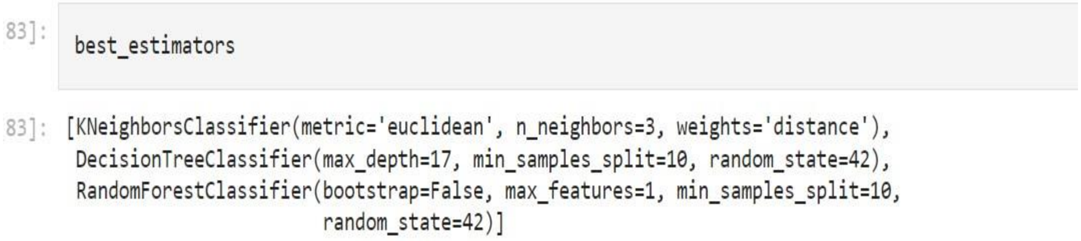
Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

CHAPTER 9

RESULTS

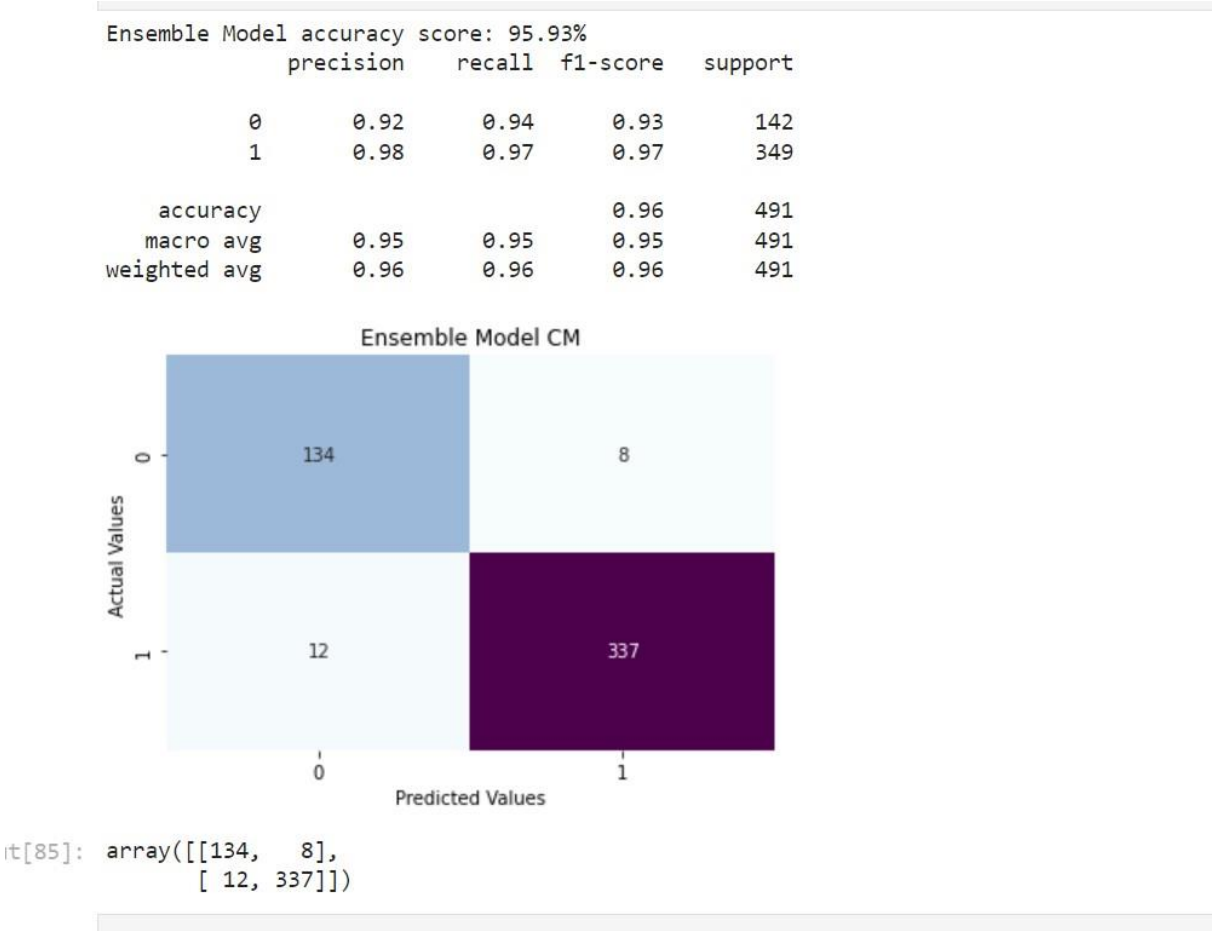
9.1 PERFORMANCE METRICS





```
83]: best_estimators
```

```
83]: [KNeighborsClassifier(metric='euclidean', n_neighbors=3, weights='distance'),
      DecisionTreeClassifier(max_depth=17, min_samples_split=10, random_state=42),
      RandomForestClassifier(bootstrap=False, max_features=1, min_samples_split=10,
                             random_state=42)]
```

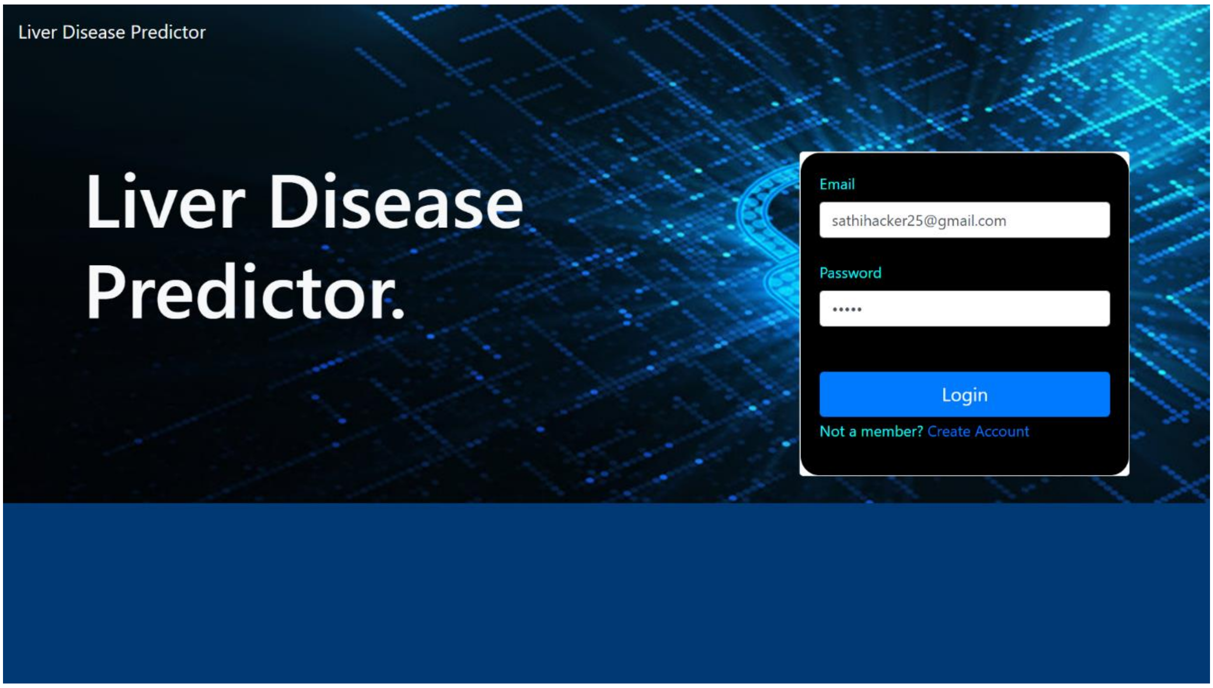


```
Ensemble Model accuracy score: 95.93%
              precision    recall  f1-score   support

           0       0.92      0.94      0.93       142
           1       0.98      0.97      0.97       349

    accuracy                           0.96       491
   macro avg       0.95      0.95      0.95       491
weighted avg       0.96      0.96      0.96       491
```

```
t[85]: array([[134,   8],
              [ 12, 337]])
```

OUTPUT SCREENSHOT FOR LIVER DISEASE

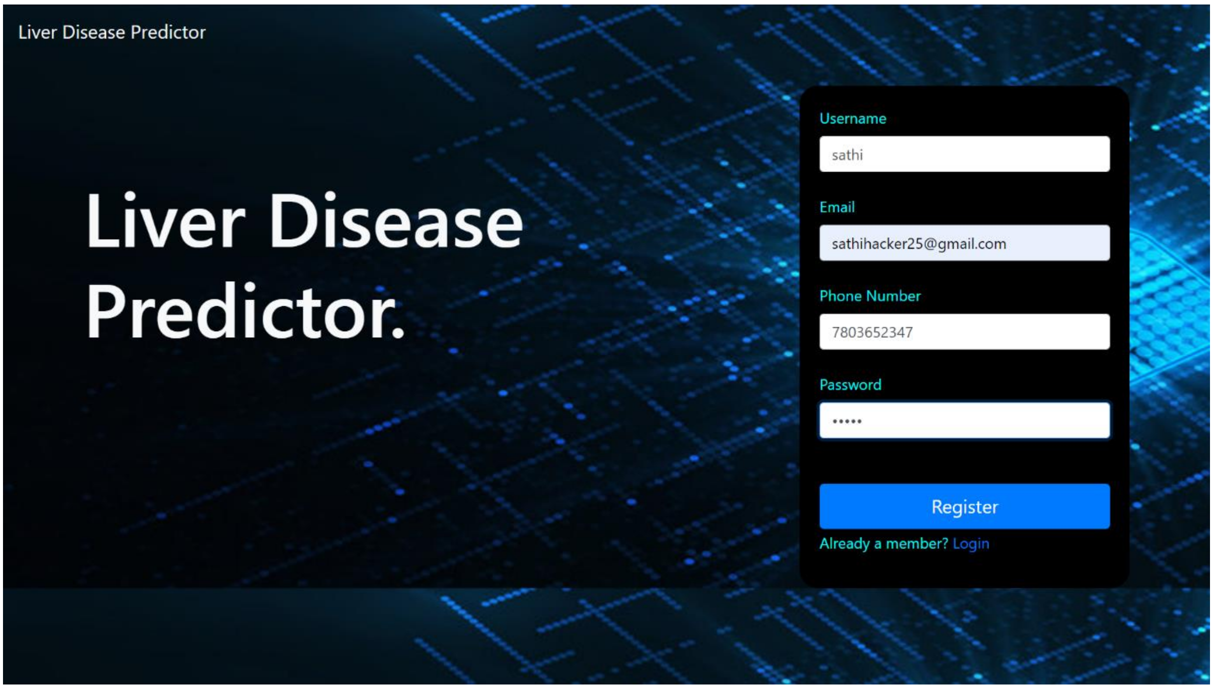
Login



Email: sathihacker25@gmail.com

Password: •••••

Register



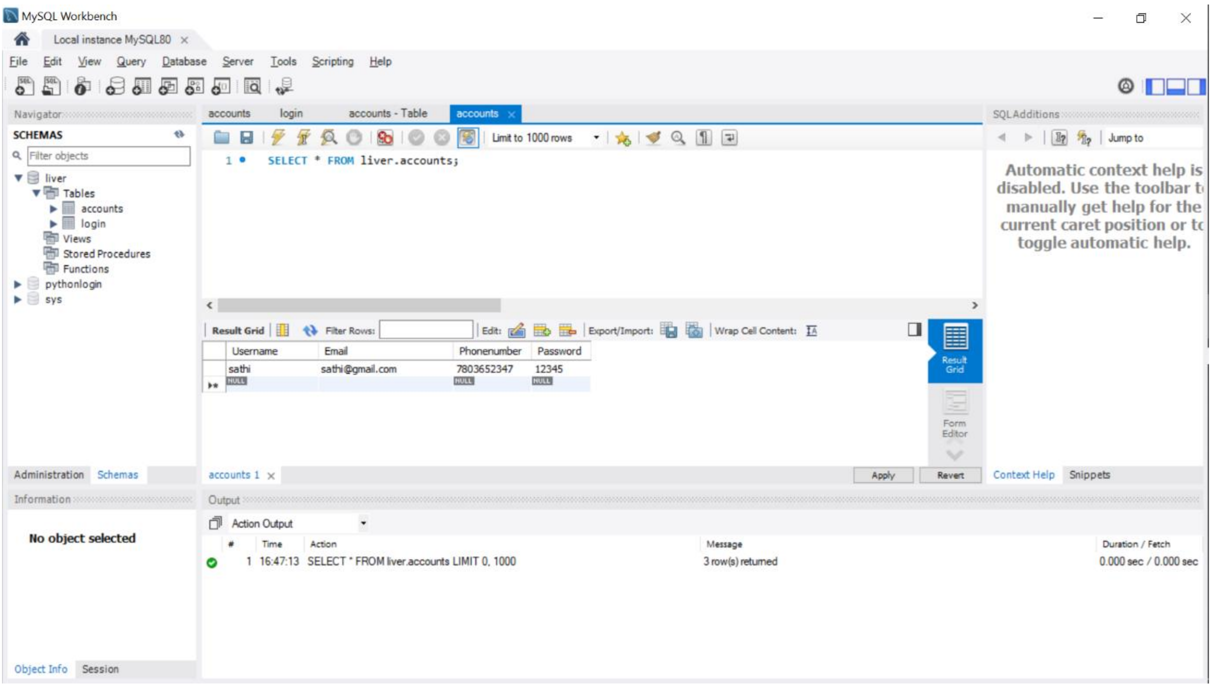
Username: sathi

Email: sathihacker25@gmail.com

Phone Number: 7803652347

Password: •••••

Database Code



```
1 • SELECT * FROM liver.accounts;
```

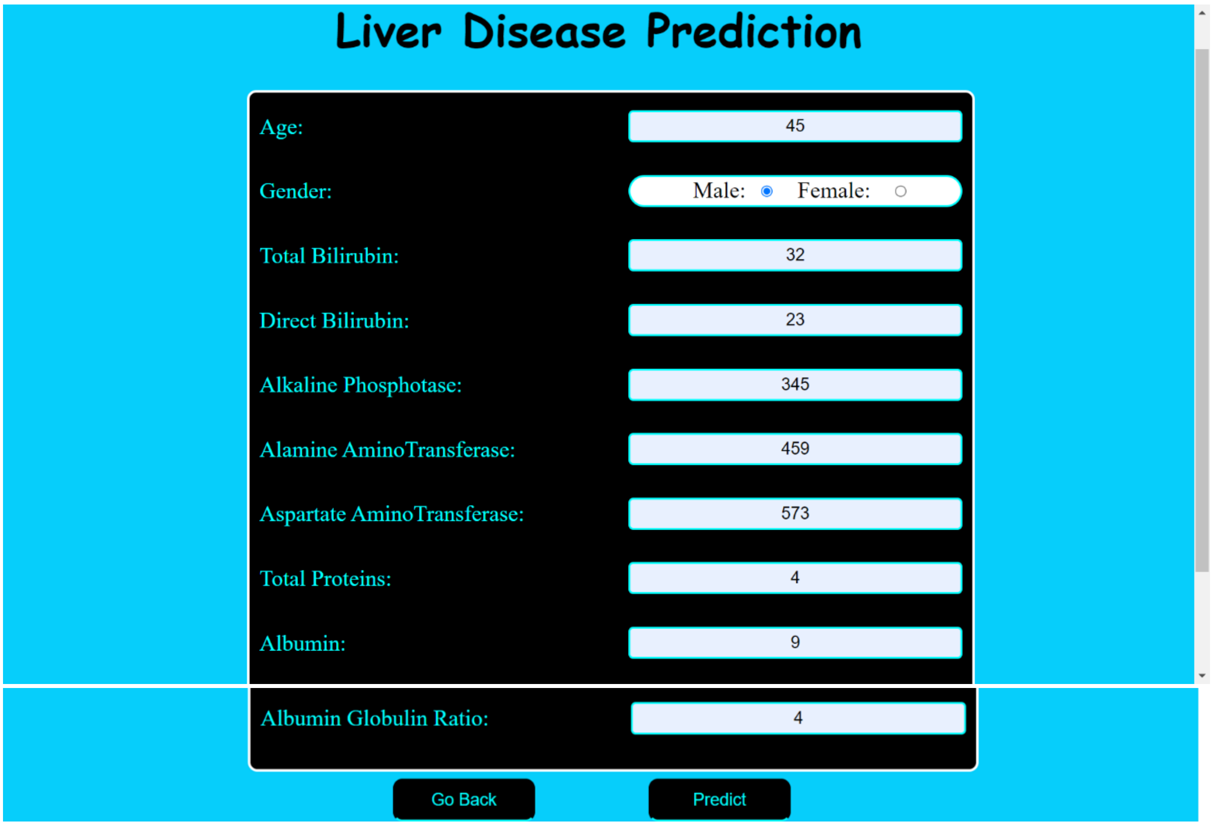
Username	Email	Phonenumber	Password
sathi	sathi@gmail.com	7803652347	12345
NULL		NULL	NULL

```
#   Time      Action                                 Message             Duration / Fetch
1   16:47:13  SELECT * FROM liver.accounts LIMIT 0, 1000   3 row(s) returned   0.000 sec / 0.000 sec
```

No object selected

Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or to toggle automatic help.

Prediction

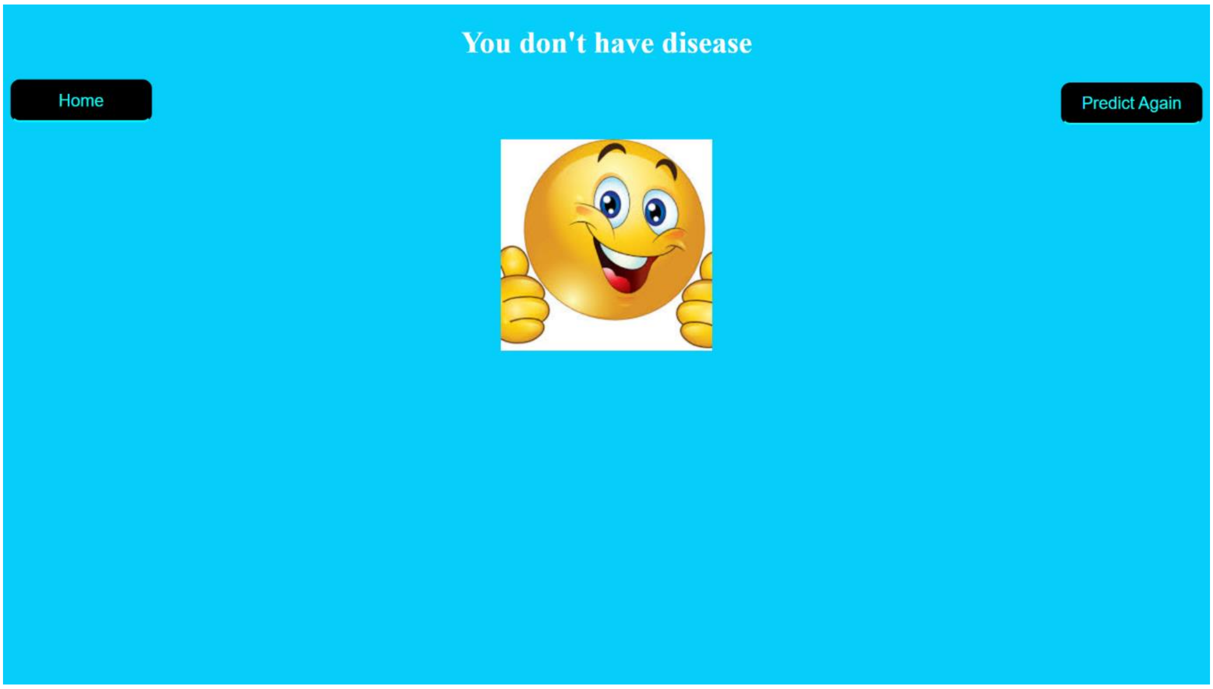


Liver Disease Prediction

Age:	45
Gender:	Male: ● Female: ○
Total Bilirubin:	32
Direct Bilirubin:	23
Alkaline Phosphotase:	345
Alamine AminoTransferase:	459
Aspartate AminoTransferase:	573
Total Proteins:	4
Albumin:	9
Albumin Globulin Ratio:	4

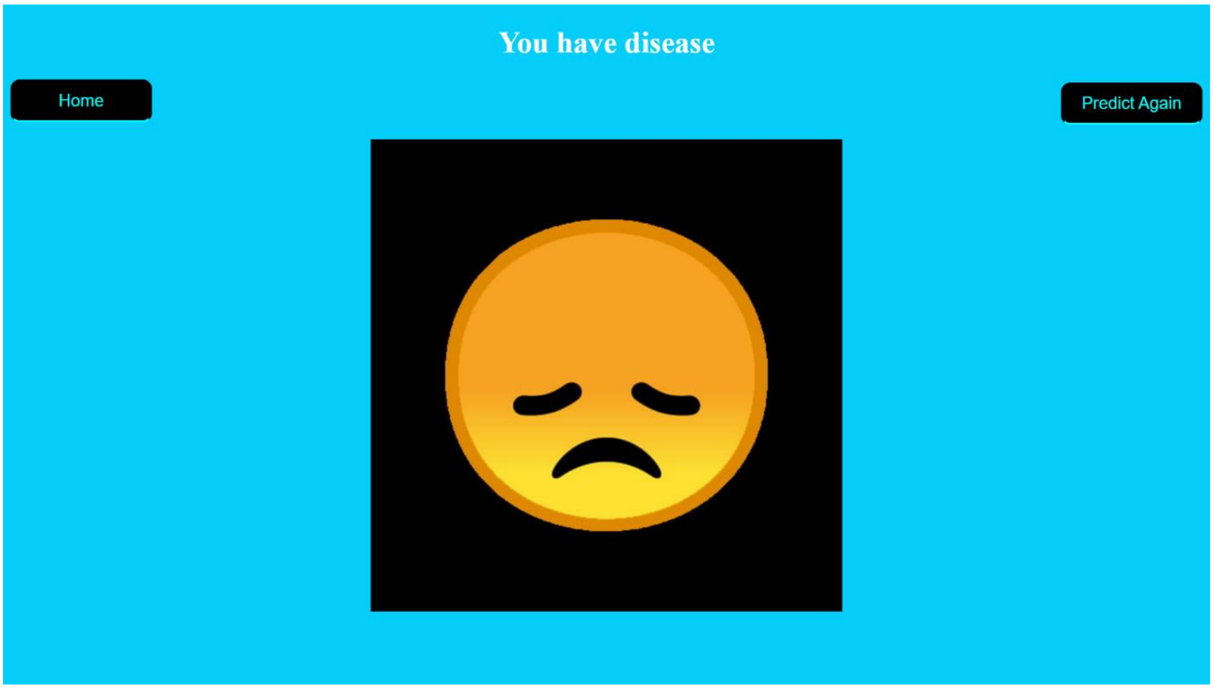
Result :

No Chance Page



You don't have disease

Chance Page



You have disease

CHAPTER 10
ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Results are projected in a matter of seconds after entering the information. In contrast to the conventional procedure, you don't need to wait for a doctor to arrive.
- The application will accurately and quickly identify which individuals have liver disease.
- The performance classification of liver based diseases is further improved in our proposed ensemble model.
- Risky factors can be predicted early by machine learning models.

DISADVANTAGES

- Some approaches are not adoptable for real time collection of database implementation.
- Certain approaches being applicable only for small data.
- Certain combination of classifier over fit with data set while others are under fit.

CHAPTER 11
CONCLUSION

The ageing of the population and the rise in the incidence and prevalence of chronic diseases result in an increased risk of liver disease-related hospitalisation or death. This is notably high for people who have several diseases, which results in large resource consumption. Finding potential high-risk patients is the biggest difficulty in order to increase the quality of medical care and cut expenditures. The main goal of the research is to put ensemble algorithms, such as K-Nearest Neighbours, Random Forest, and Decision Tree, into practise in order to forecast the likelihood of hospitalisation or mortality starting from administrative and socioeconomic information. Our goal is to get prediction on the basis of given datasets of people whether the person is having the chronic disease or liver disease symptoms or not. This system will be very useful for many hospitals and even professional doctors to easily detect the disease. Also, general user can use this system for their finding out the disease. This system will change the way and can be early as possible as it will lead to save the person’s life. This whole work is focused on how we can predict the disease by given datasets so that will help in preventing and curing the disease of the patients.

CHAPTER 12

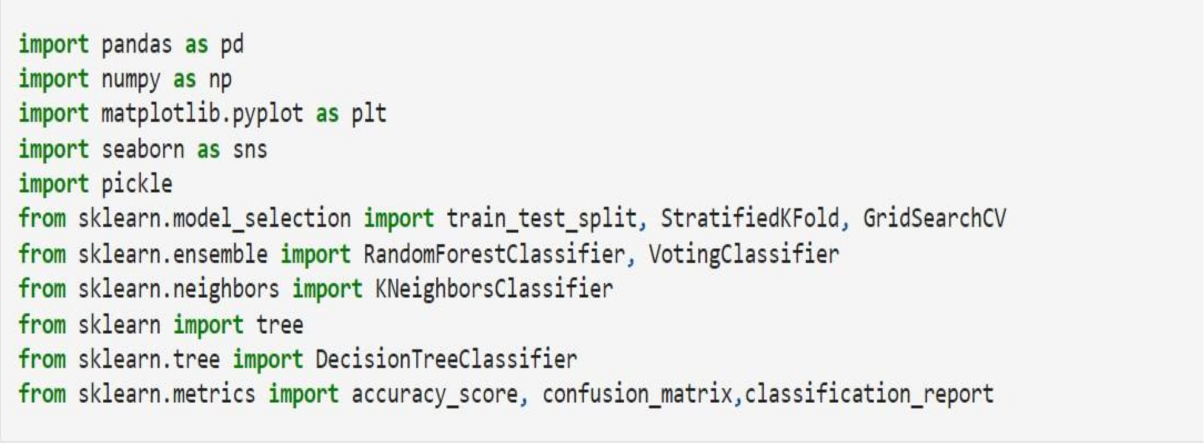
FUTURE SCOPE

This project can be further developed by establishing an alarm system for the patient’s relatives and doctor according to the risk level. Deep learning algorithms can be used to enhance the performance. However, in future we are planning to collect the very recent data from various regions across the world for liver disease prediction.

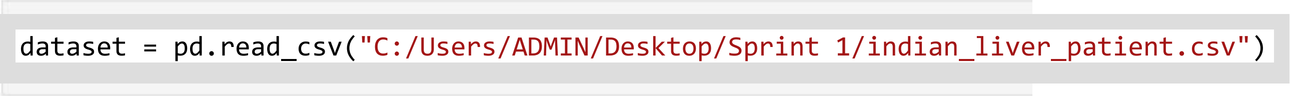
APPENDIX

SOURCE CODE

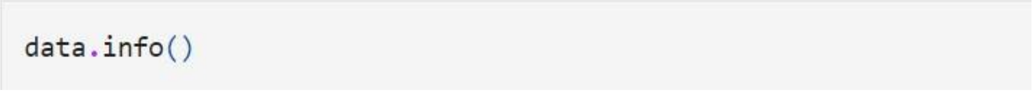
MODEL CREATION



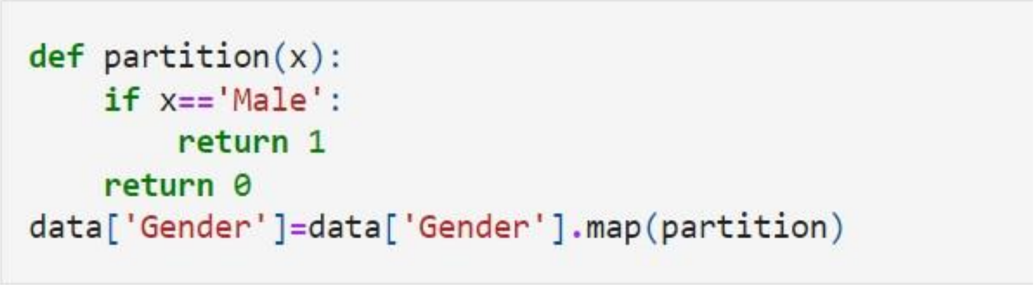
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
from sklearn.model_selection import train_test_split, StratifiedKFold, GridSearchCV
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report
```



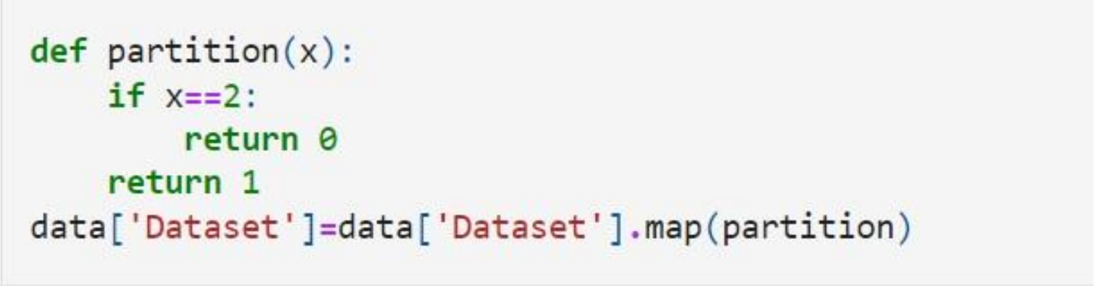
```
dataset = pd.read_csv("C:/Users/ADMIN/Desktop/Sprint 1/indian_liver_patient.csv")
```



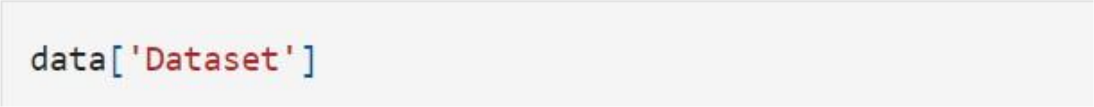
```
data.info()
```



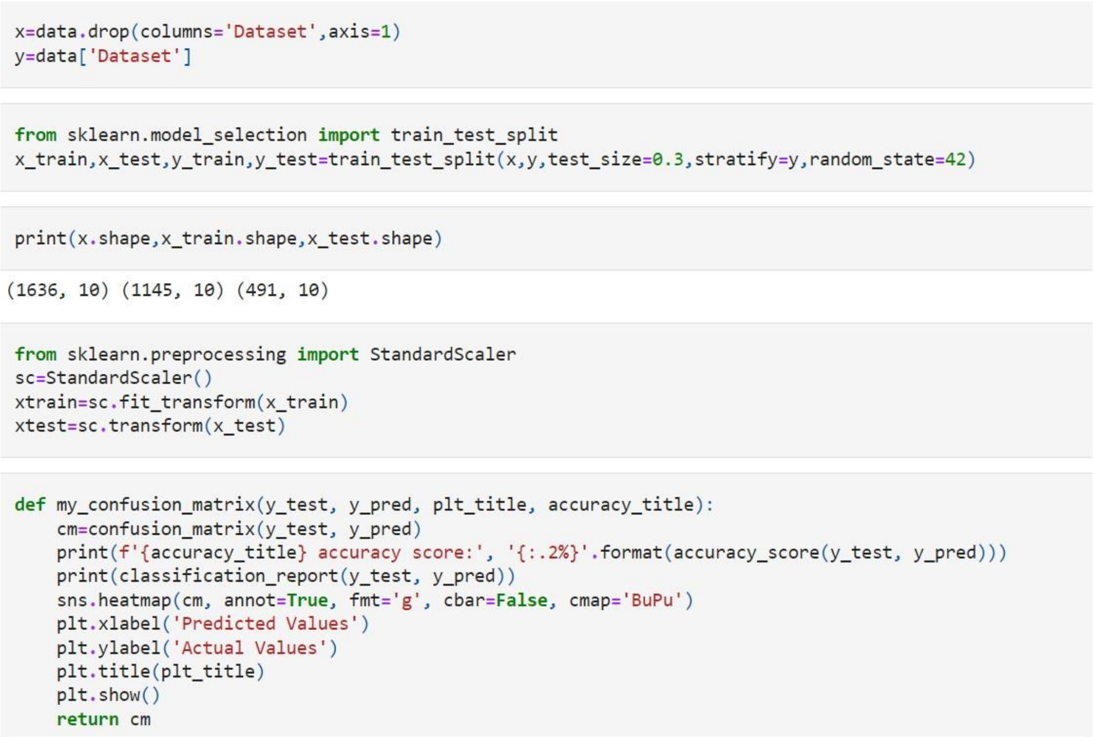
```
def partition(x):
    if x=='Male':
        return 1
    return 0
data['Gender']=data['Gender'].map(partition)
```



```
def partition(x):
    if x==2:
        return 0
    return 1
data['Dataset']=data['Dataset'].map(partition)
```



```
data['Dataset']
```



```
x=data.drop(columns='Dataset',axis=1)
y=data['Dataset']
```

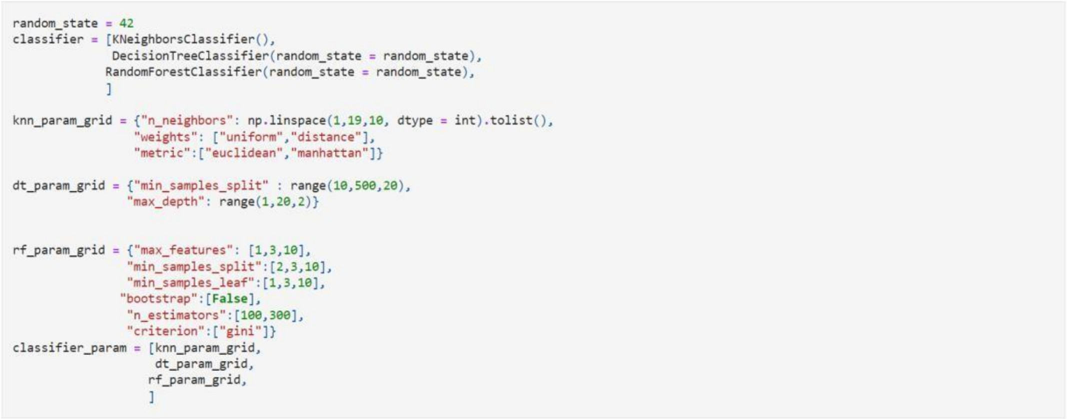
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,stratify=y,random_state=42)
```

```
print(x.shape,x_train.shape,x_test.shape)
```

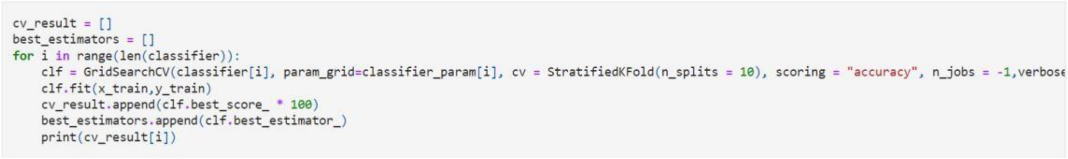
```
(1636, 10) (1145, 10) (491, 10)
```

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
xtrain=sc.fit_transform(x_train)
xtest=sc.transform(x_test)
```

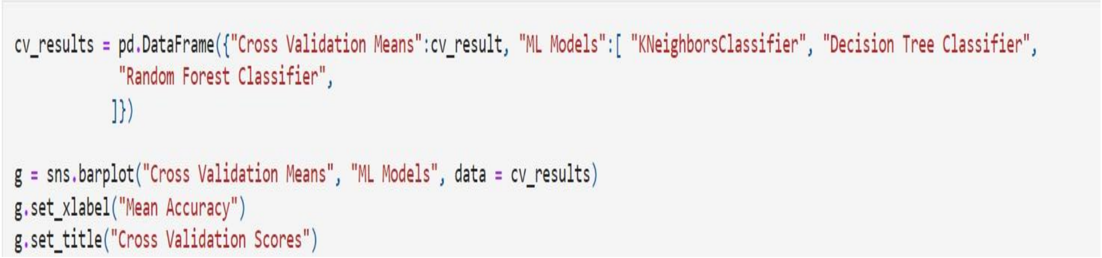
```
def my_confusion_matrix(y_test, y_pred, plt_title, accuracy_title):
    cm=confusion_matrix(y_test, y_pred)
    print(f'{accuracy_title} accuracy score:', '{:.2%}'.format(accuracy_score(y_test, y_pred)))
    print(classification_report(y_test, y_pred))
    sns.heatmap(cm, annot=True, fmt='g', cbar=False, cmap='BuPu')
    plt.xlabel('Predicted Values')
    plt.ylabel('Actual Values')
    plt.title(plt_title)
    plt.show()
    return cm
```



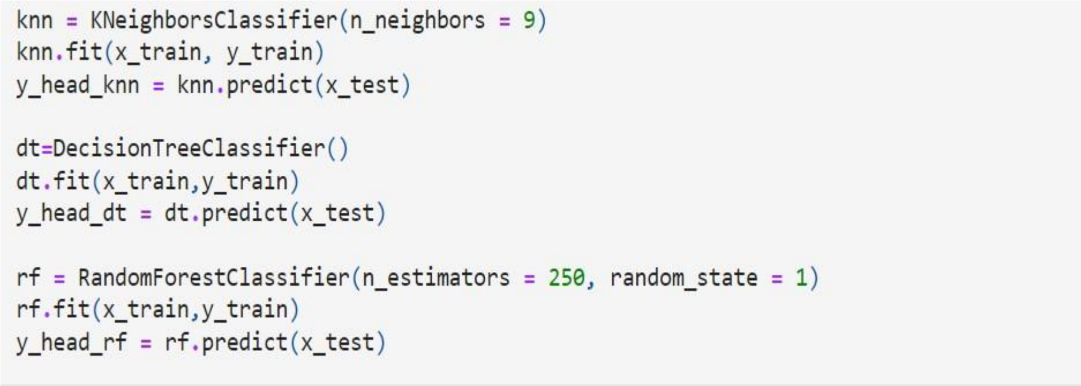
```
random_state = 42
classifier = [KNeighborsClassifier(),
              DecisionTreeClassifier(random_state = random_state),
             RandomForestClassifier(random_state = random_state),
             ]

knn_param_grid = {"n_neighbors": np.linspace(1,19,10, dtype = int).tolist(),
                  "weights": ["uniform","distance"],
                  "metric":["euclidean","manhattan"]}

dt_param_grid = {"min_samples_split" : range(10,500,20),
                 "max_depth": range(1,20,2)}


rf_param_grid = {"max_features": [1,3,10],
                 "min_samples_split":[2,3,10],
                 "min_samples_leaf":[1,3,10],
                "bootstrap":[False],
                 "n_estimators":[100,300],
                 "criterion":["gini"]}
classifier_param = [knn_param_grid,
                     dt_param_grid,
                    rf_param_grid,
                    ]
```



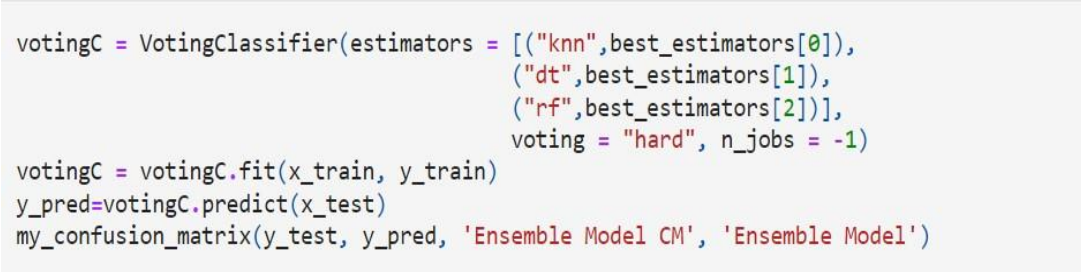
```
cv_result = []
best_estimators = []
for i in range(len(classifier)):
    clf = GridSearchCV(classifier[i], param_grid=classifier_param[i], cv = StratifiedKFold(n_splits = 10), scoring = "accuracy", n_jobs = -1,verbose
    clf.fit(x_train,y_train)
    cv_result.append(clf.best_score_ * 100)
    best_estimators.append(clf.best_estimator_)
    print(cv_result[i])
```



```
cv_results = pd.DataFrame({"Cross Validation Means":cv_result, "ML Models":[ "KNeighborsClassifier", "Decision Tree Classifier",
             "Random Forest Classifier",
            ]})

g = sns.barplot("Cross Validation Means", "ML Models", data = cv_results)
g.set_xlabel("Mean Accuracy")
g.set_title("Cross Validation Scores")
```

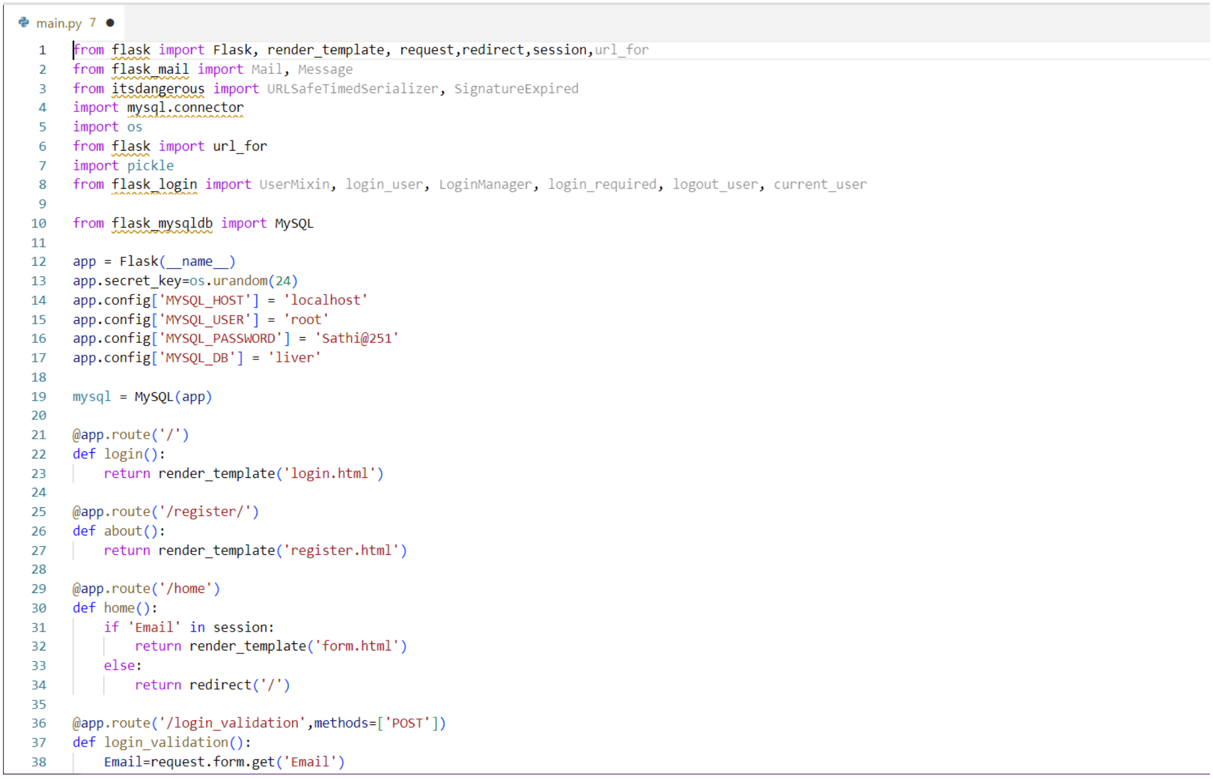


```
knn = KNeighborsClassifier(n_neighbors = 9)
knn.fit(x_train, y_train)
y_head_knn = knn.predict(x_test)

dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_head_dt = dt.predict(x_test)

rf = RandomForestClassifier(n_estimators = 250, random_state = 1)
rf.fit(x_train,y_train)
y_head_rf = rf.predict(x_test)
```

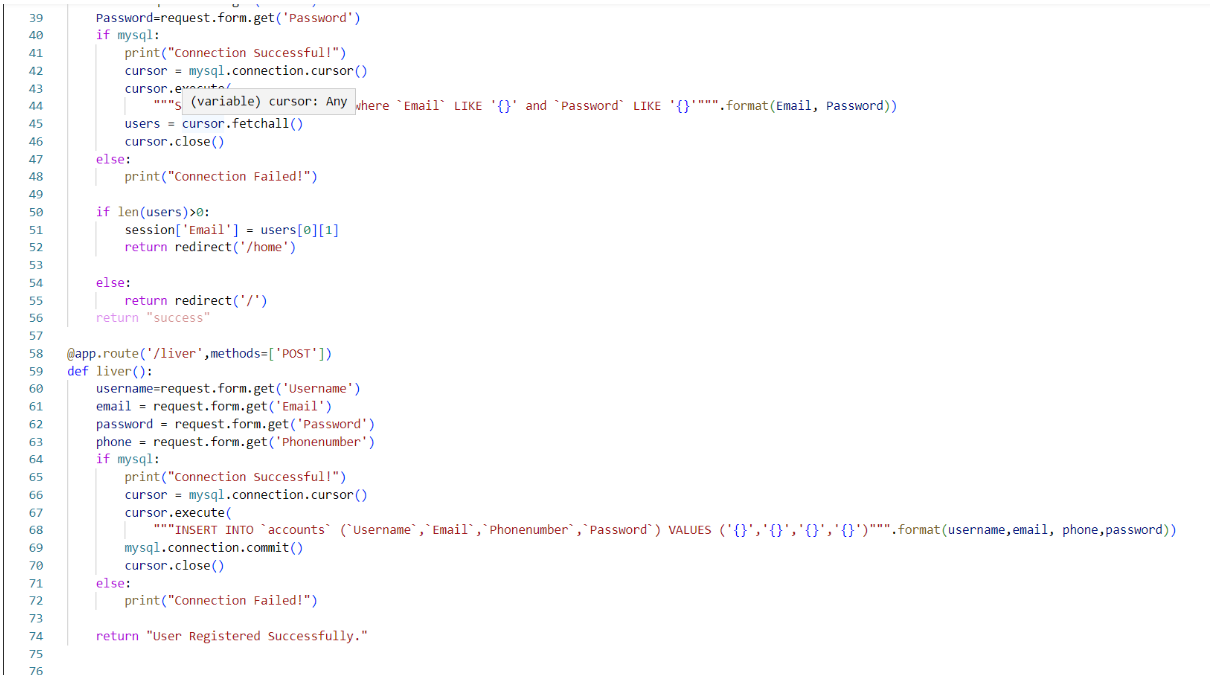


```
votingC = VotingClassifier(estimators = [("knn",best_estimators[0]),
                                        ("dt",best_estimators[1]),
                                        ("rf",best_estimators[2])],
                                        voting = "hard", n_jobs = -1)
votingC = votingC.fit(x_train, y_train)
y_pred=votingC.predict(x_test)
my_confusion_matrix(y_test, y_pred, 'Ensemble Model CM', 'Ensemble Model')
```

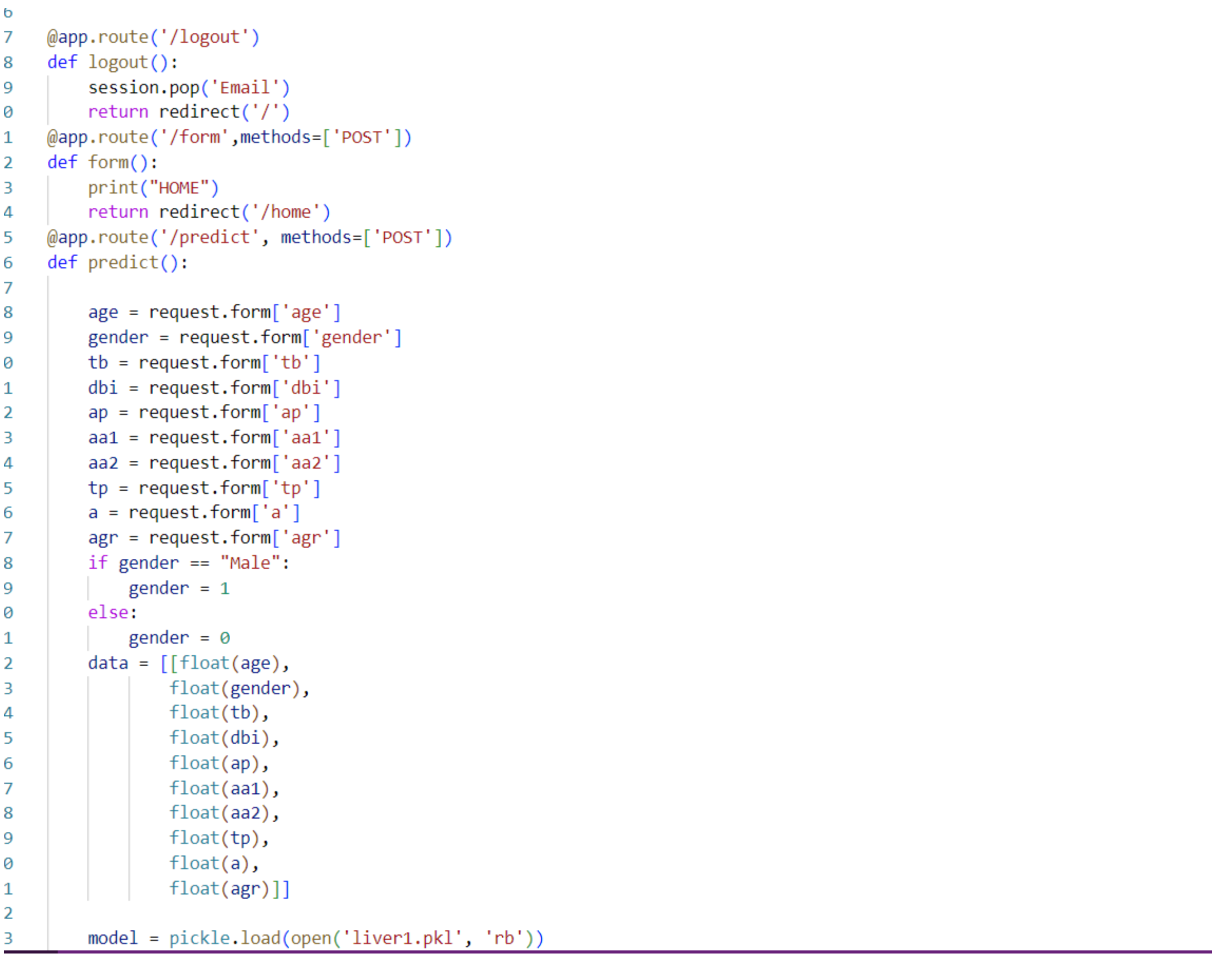
FLASK APP



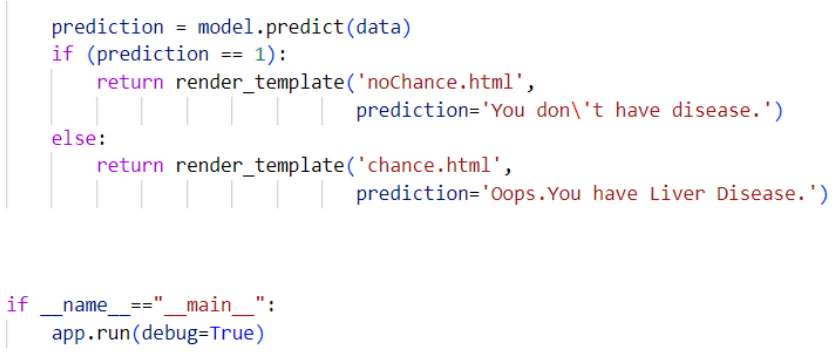
```
from flask import Flask, render_template, request,redirect,session,url_for
from flask_mail import Mail, Message
from itsdangerous import URLSafeTimedSerializer, SignatureExpired
import mysql.connector
import os
from flask import url_for
import pickle
from flask_login import UserMixin, login_user, LoginManager, login_required, logout_user, current_user

from flask_mysqldb import MySQL

app = Flask(__name__)
app.secret_key=os.urandom(24)
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'Sathi@251'
app.config['MYSQL_DB'] = 'liver'

mysql = MySQL(app)

@app.route('/')
def login():
    return render_template('login.html')

@app.route('/register/')
def about():
    return render_template('register.html')

@app.route('/home')
def home():
    if 'Email' in session:
        return render_template('form.html')
    else:
        return redirect('/')

@app.route('/login_validation',methods=['POST'])
def login_validation():
    Email=request.form.get('Email')
```



```
    Password=request.form.get('Password')
    if mysql:
        print("Connection Successful!")
        cursor = mysql.connection.cursor()
        cursor.execute(
            """S (variable) cursor: Any where `Email` LIKE '{}' and `Password` LIKE '{}'""".format(Email, Password))
        users = cursor.fetchall()
        cursor.close()
    else:
        print("Connection Failed!")

    if len(users)>0:
        session['Email'] = users[0][1]
        return redirect('/home')

    else:
        return redirect('/')
    return "success"

@app.route('/liver',methods=['POST'])
def liver():
    username=request.form.get('Username')
    email = request.form.get('Email')
    password = request.form.get('Password')
    phone = request.form.get('Phonenumber')
    if mysql:
        print("Connection Successful!")
        cursor = mysql.connection.cursor()
        cursor.execute(
            """INSERT INTO `accounts` (`Username`,`Email`,`Phonenumber`,`Password`) VALUES ('{}','{}','{}','{}')""".format(username,email, phone,password))
        mysql.connection.commit()
        cursor.close()
    else:
        print("Connection Failed!")

    return "User Registered Successfully."
```

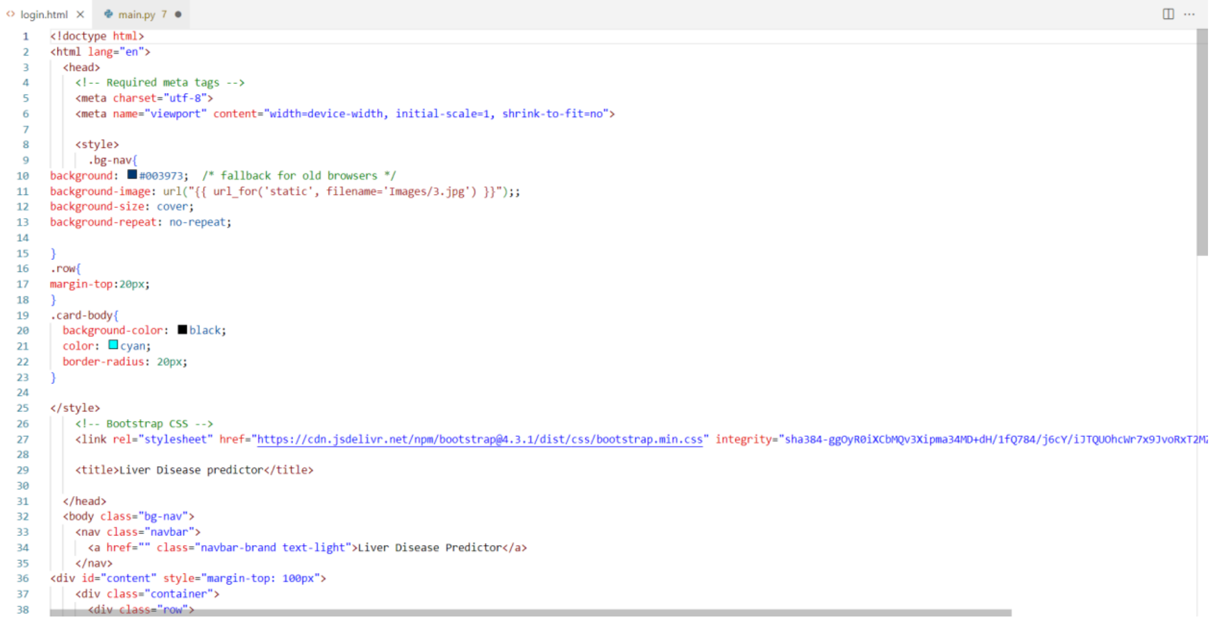


```
@app.route('/logout')
def logout():
    session.pop('Email')
    return redirect('/')
@app.route('/form',methods=['POST'])
def form():
    print("HOME")
    return redirect('/home')
@app.route('/predict', methods=['POST'])
def predict():

    age = request.form['age']
    gender = request.form['gender']
    tb = request.form['tb']
    dbi = request.form['dbi']
    ap = request.form['ap']
    aa1 = request.form['aa1']
    aa2 = request.form['aa2']
    tp = request.form['tp']
    a = request.form['a']
    agr = request.form['agr']
    if gender == "Male":
        gender = 1
    else:
        gender = 0
    data = [[float(age),
            float(gender),
            float(tb),
            float(dbi),
            float(ap),
            float(aa1),
            float(aa2),
            float(tp),
            float(a),
            float(agr)]]

    model = pickle.load(open('liver1.pkl', 'rb'))
```

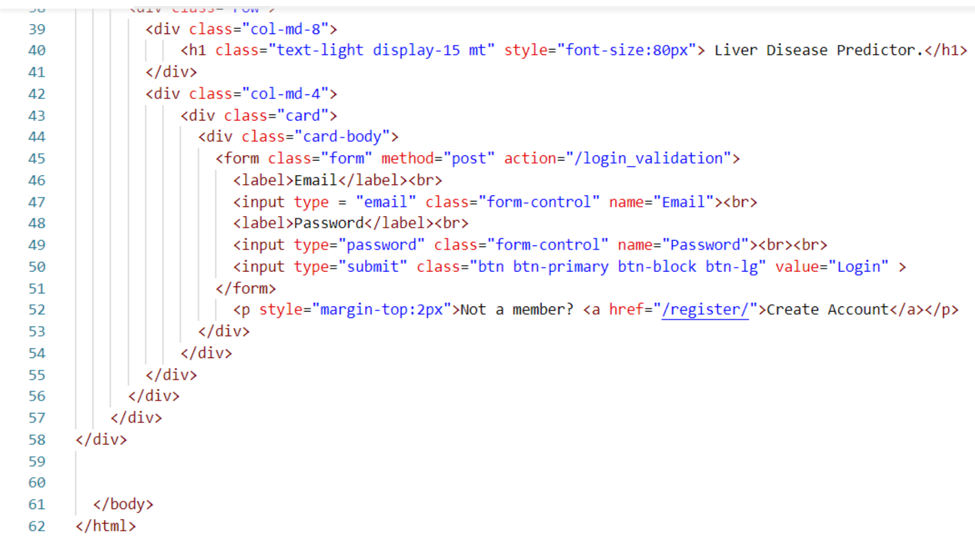


```
    prediction = model.predict(data)
    if (prediction == 1):
        return render_template('noChance.html',
                               prediction='You don\'t have disease.')
    else:
        return render_template('chance.html',
                               prediction='Oops.You have Liver Disease.')


if __name__=="__main__":
    app.run(debug=True)
```

Login (HTML)

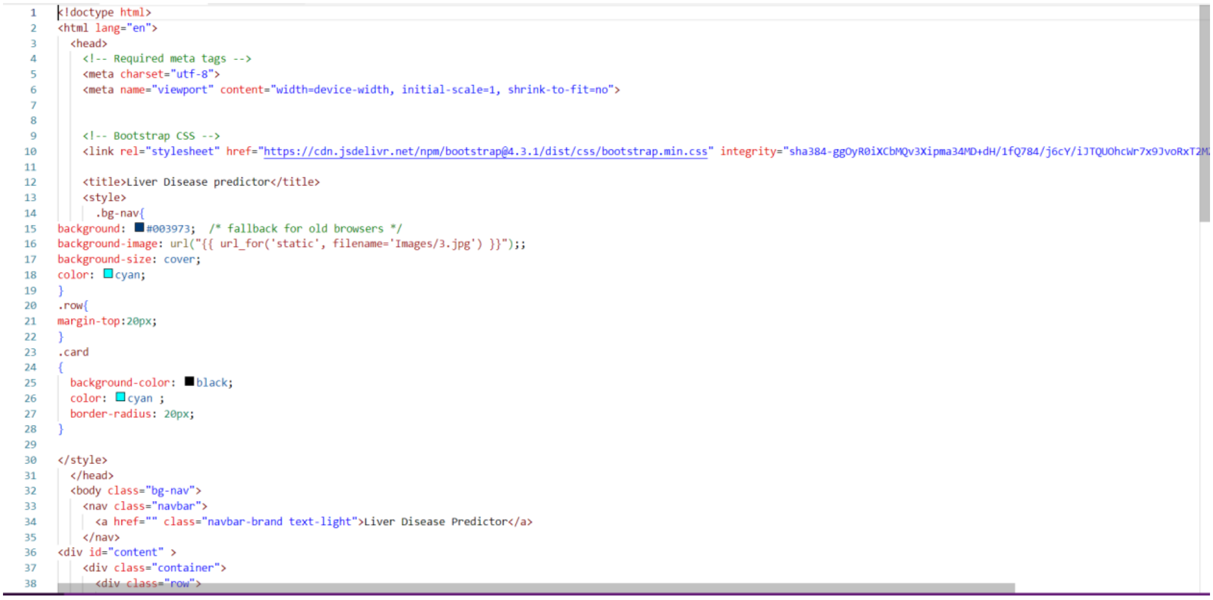


```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <style>
      .bg-nav{
background: ■#003973;  /* fallback for old browsers */
background-image: url("{{ url_for('static', filename='Images/3.jpg') }}");;
background-size: cover;
background-repeat: no-repeat;

}
.row{
margin-top:20px;
}
.card-body{
  background-color: ■black;
  color: □cyan;
  border-radius: 20px;
}

</style>
    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2M

    <title>Liver Disease predictor</title>

  </head>
  <body class="bg-nav">
    <nav class="navbar">
      <a href="" class="navbar-brand text-light">Liver Disease Predictor</a>
    </nav>
<div id="content" style="margin-top: 100px">
    <div class="container">
      <div class="row">
```

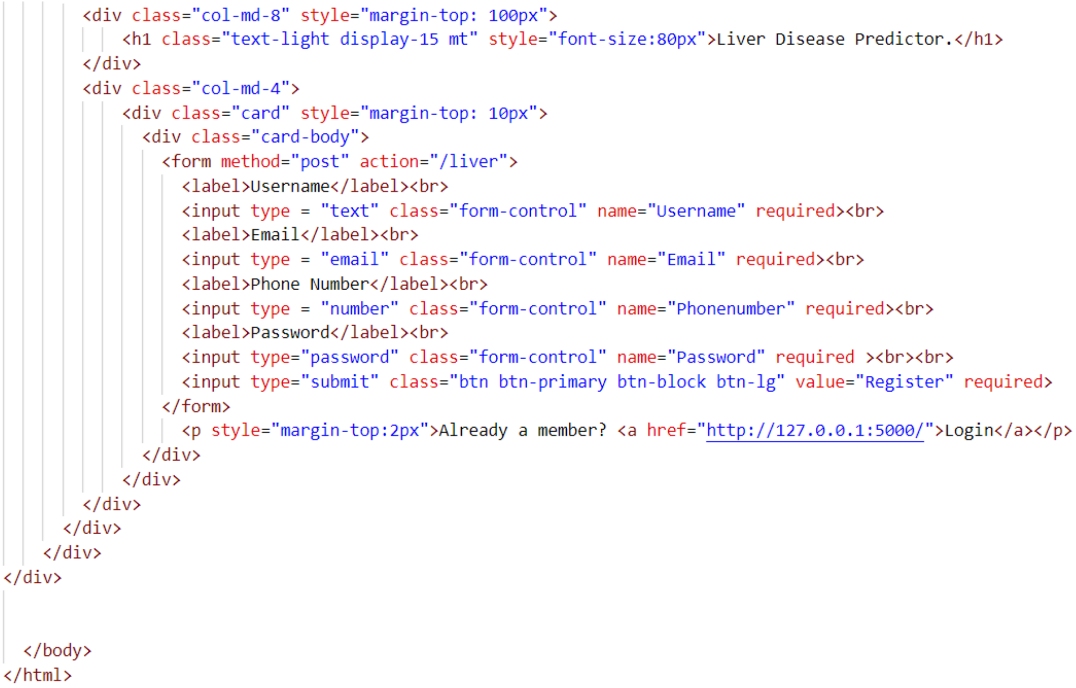


```
        <div class="col-md-8">
            <h1 class="text-light display-15 mt" style="font-size:80px"> Liver Disease Predictor.</h1>
        </div>
        <div class="col-md-4">
            <div class="card">
              <div class="card-body">
                <form class="form" method="post" action="/login_validation">
                  <label>Email</label><br>
                  <input type = "email" class="form-control" name="Email"><br>
                  <label>Password</label><br>
                  <input type="password" class="form-control" name="Password"><br><br>
                  <input type="submit" class="btn btn-primary btn-block btn-lg" value="Login" >
                </form>
                  <p style="margin-top:2px">Not a member? <a href="/register/">Create Account</a></p>
              </div>
            </div>
        </div>
      </div>
    </div>
</div>


  </body>
</html>
```

Register (HTML)

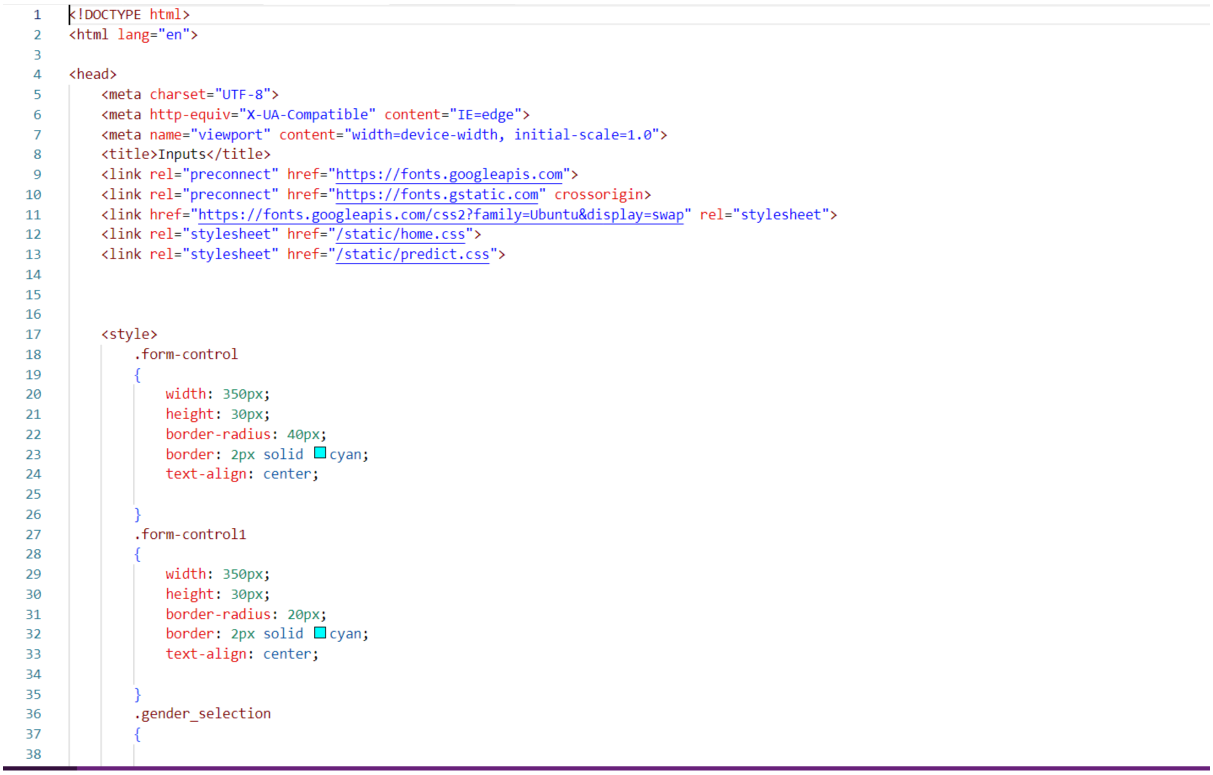


```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">


    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2M

    <title>Liver Disease predictor</title>
    <style>
      .bg-nav{
background: #003973;  /* fallback for old browsers */
background-image: url("{{ url_for('static', filename='Images/3.jpg') }}");;
background-size: cover;
color: cyan;
}
.row{
margin-top:20px;
}
.card
{
  background-color: black;
  color: cyan ;
  border-radius: 20px;
}

</style>
  </head>
  <body class="bg-nav">
    <nav class="navbar">
      <a href="" class="navbar-brand text-light">Liver Disease Predictor</a>
    </nav>
<div id="content" >
    <div class="container">
      <div class="row">
```

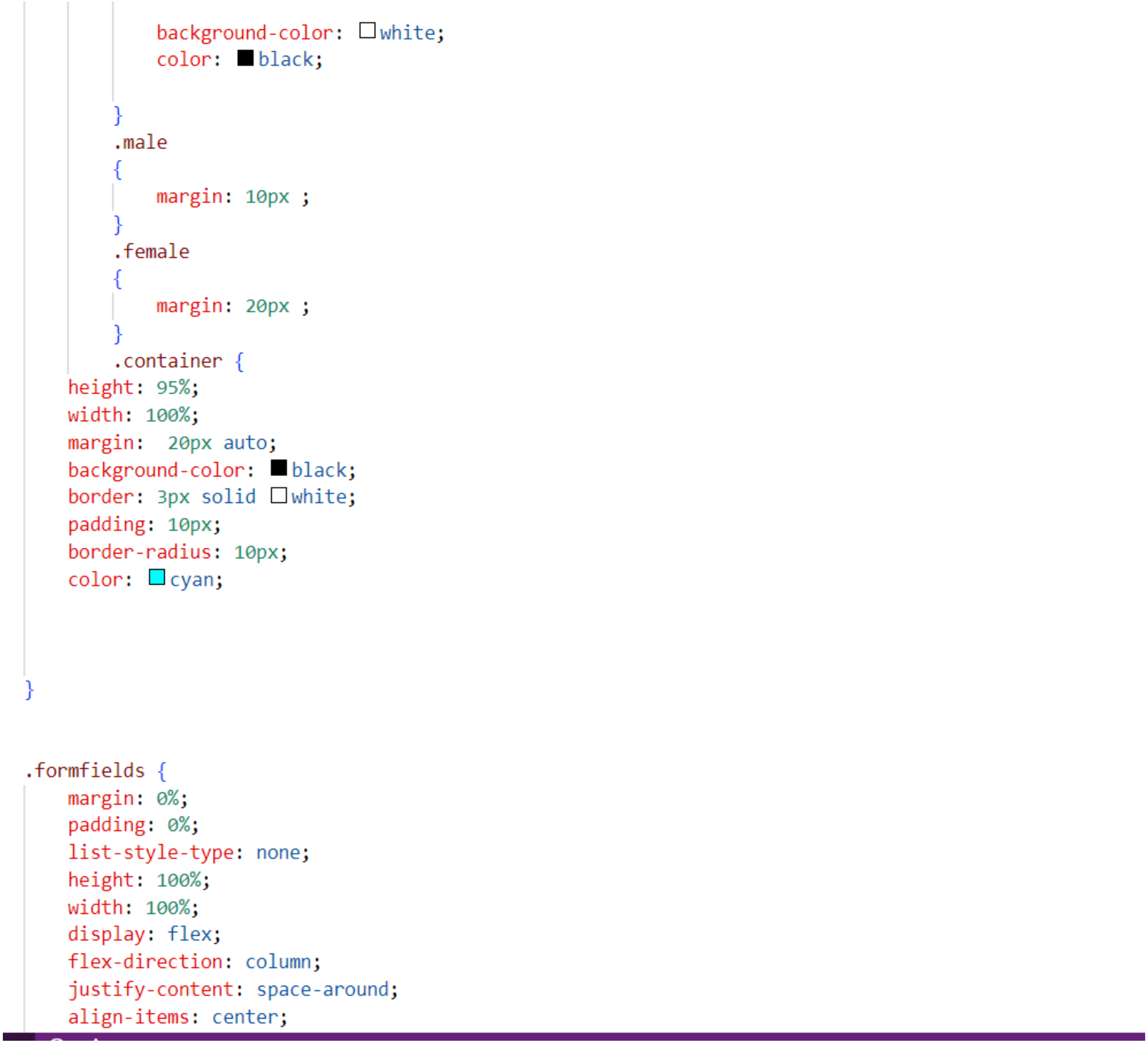


```
        <div class="col-md-8" style="margin-top: 100px">
            <h1 class="text-light display-15 mt" style="font-size:80px">Liver Disease Predictor.</h1>
        </div>
        <div class="col-md-4">
            <div class="card" style="margin-top: 10px">
              <div class="card-body">
                <form method="post" action="/liver">
                  <label>Username</label><br>
                  <input type = "text" class="form-control" name="Username" required><br>
                  <label>Email</label><br>
                  <input type = "email" class="form-control" name="Email" required><br>
                  <label>Phone Number</label><br>
                  <input type = "number" class="form-control" name="Phonenumber" required><br>
                  <label>Password</label><br>
                  <input type="password" class="form-control" name="Password" required ><br><br>
                  <input type="submit" class="btn btn-primary btn-block btn-lg" value="Register" required>
                </form>
                  <p style="margin-top:2px">Already a member? <a href="http://127.0.0.1:5000/">Login</a></p>
              </div>
            </div>
        </div>
      </div>
    </div>
</div>


  </body>
</html>
```

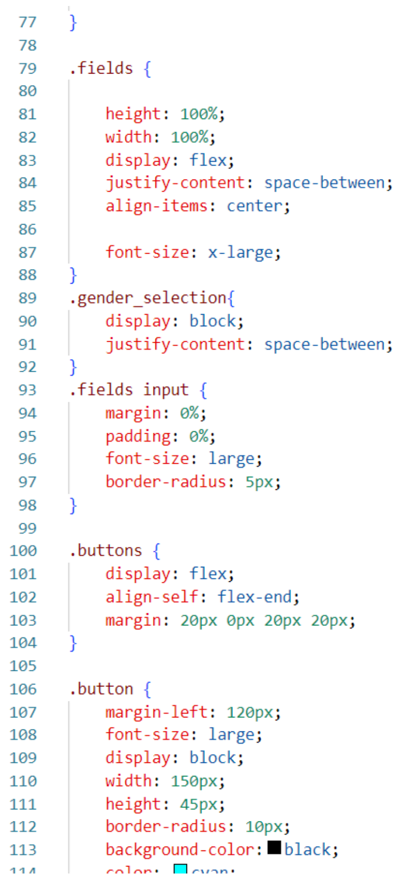
Form (HTML)



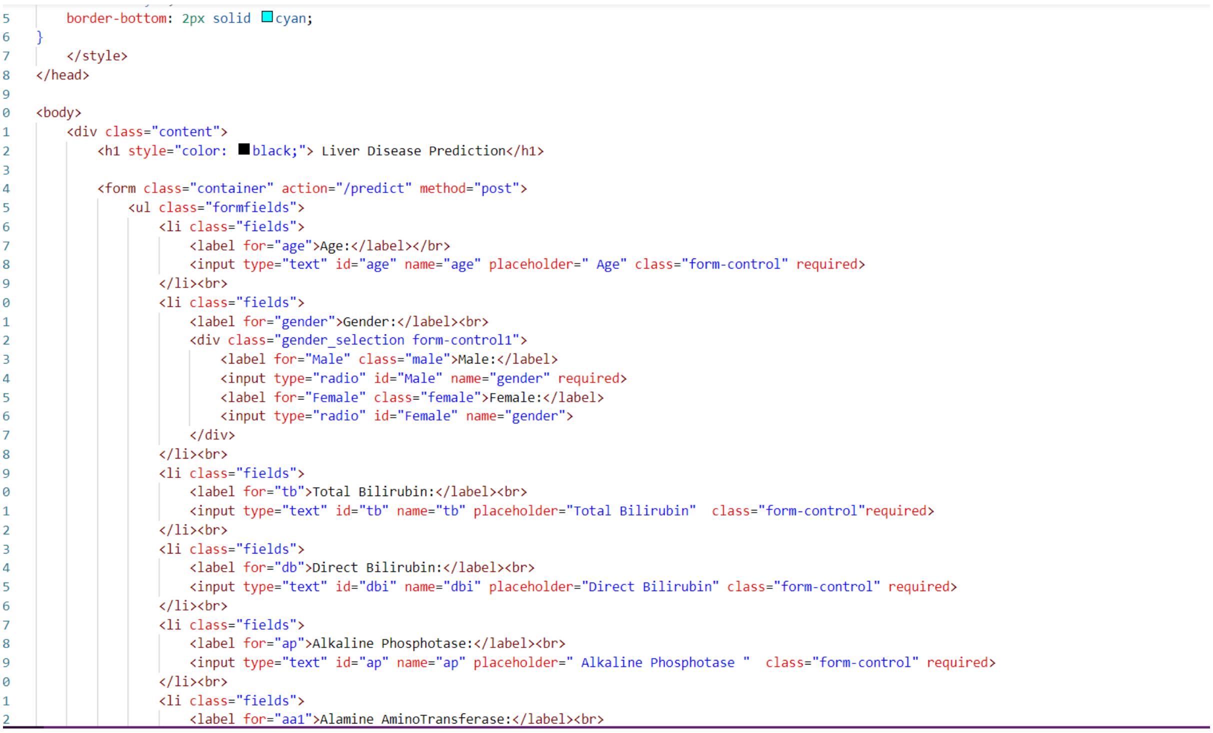
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inputs</title>
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Ubuntu&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="/static/home.css">
    <link rel="stylesheet" href="/static/predict.css">



    <style>
        .form-control
        {
            width: 350px;
            height: 30px;
            border-radius: 40px;
            border: 2px solid cyan;
            text-align: center;

        }
        .form-control1
        {
            width: 350px;
            height: 30px;
            border-radius: 20px;
            border: 2px solid cyan;
            text-align: center;

        }
        .gender_selection
        {

```



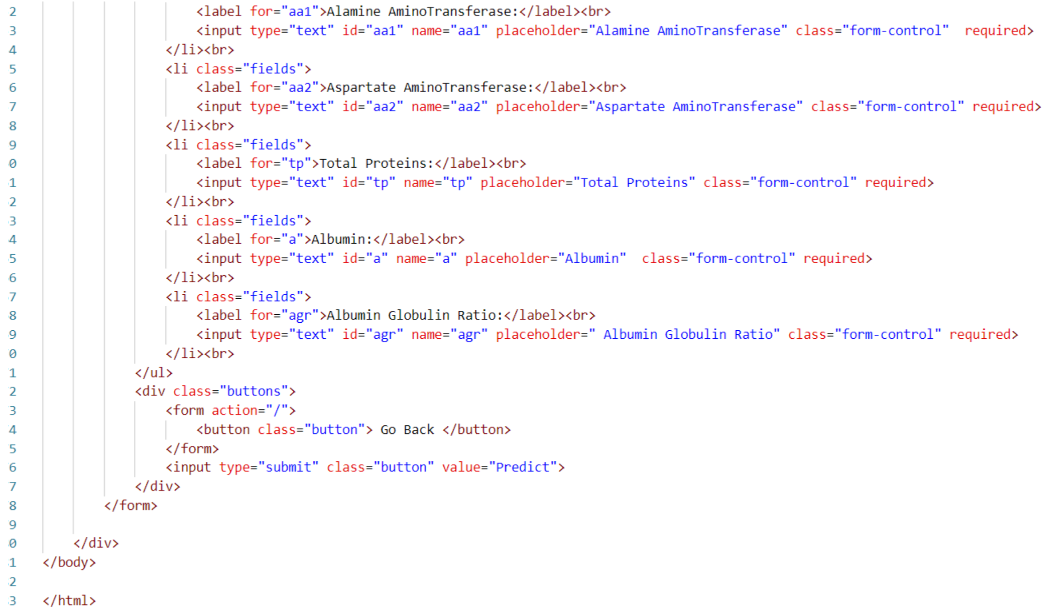
```
            background-color: □white;
            color: ■black;

        }
        .male
        {
            margin: 10px ;
        }
        .female
        {
            margin: 20px ;
        }
        .container {
    height: 95%;
    width: 100%;
    margin:  20px auto;
    background-color: ■black;
    border: 3px solid □white;
    padding: 10px;
    border-radius: 10px;
    color: ■cyan;


}


.formfields {
    margin: 0%;
    padding: 0%;
    list-style-type: none;
    height: 100%;
    width: 100%;
    display: flex;
    flex-direction: column;
    justify-content: space-around;
    align-items: center;
```



```
}

.fields {

    height: 100%;
    width: 100%;
    display: flex;
    justify-content: space-between;
    align-items: center;

    font-size: x-large;
}
.gender_selection{
    display: block;
    justify-content: space-between;
}
.fields input {
    margin: 0%;
    padding: 0%;
    font-size: large;
    border-radius: 5px;
}

.buttons {
    display: flex;
    align-self: flex-end;
    margin: 20px 0px 20px 20px;
}

.button {
    margin-left: 120px;
    font-size: large;
    display: block;
    width: 150px;
    height: 45px;
    border-radius: 10px;
    background-color: ■black;
    color: □cyan;
```

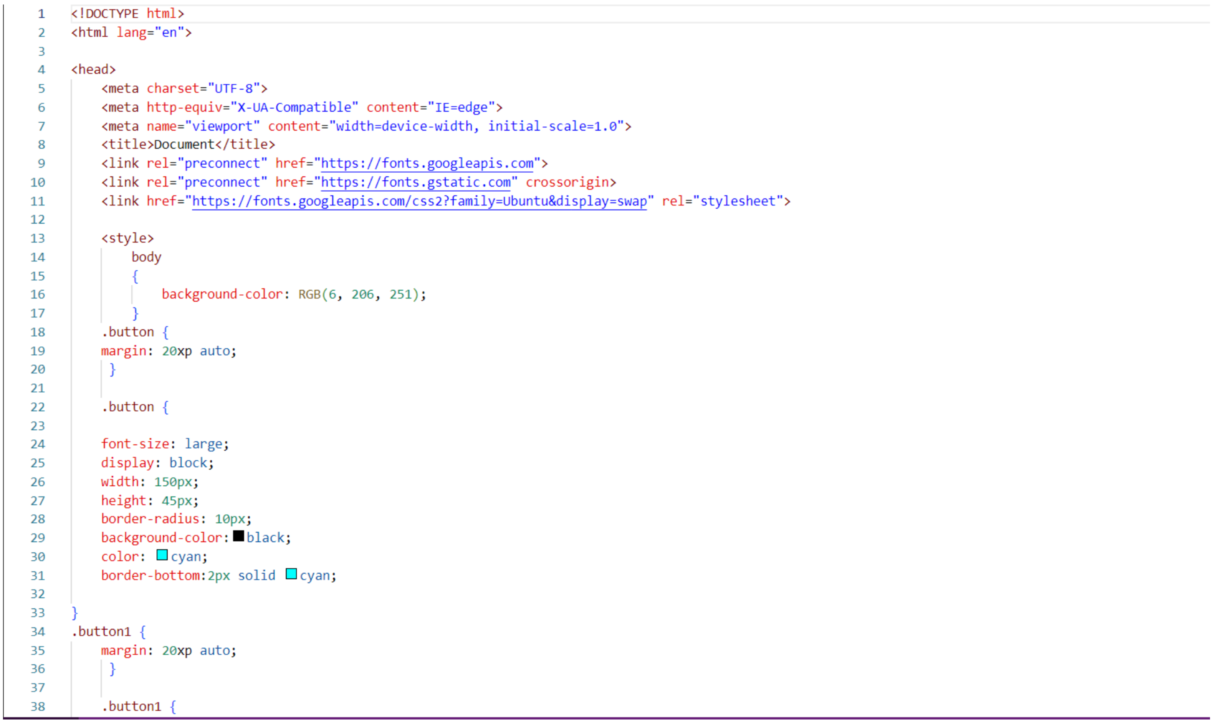


```
    border-bottom: 2px solid □cyan;
}
    </style>
</head>

<body>
    <div class="content">
        <h1 style="color: ■black;"> Liver Disease Prediction</h1>

        <form class="container" action="/predict" method="post">
            <ul class="formfields">
                <li class="fields">
                    <label for="age">Age:</label></br>
                    <input type="text" id="age" name="age" placeholder=" Age" class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="gender">Gender:</label><br>
                    <div class="gender_selection form-control1">
                        <label for="Male" class="male">Male:</label>
                        <input type="radio" id="Male" name="gender" required>
                        <label for="Female" class="female">Female:</label>
                        <input type="radio" id="Female" name="gender">
                    </div>
                </li><br>
                <li class="fields">
                    <label for="tb">Total Bilirubin:</label><br>
                    <input type="text" id="tb" name="tb" placeholder="Total Bilirubin"  class="form-control"required>
                </li><br>
                <li class="fields">
                    <label for="db">Direct Bilirubin:</label><br>
                    <input type="text" id="dbi" name="dbi" placeholder="Direct Bilirubin" class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="ap">Alkaline Phosphotase:</label><br>
                    <input type="text" id="ap" name="ap" placeholder=" Alkaline Phosphotase "  class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="aa1">Alamine AminoTransferase:</label><br>
```

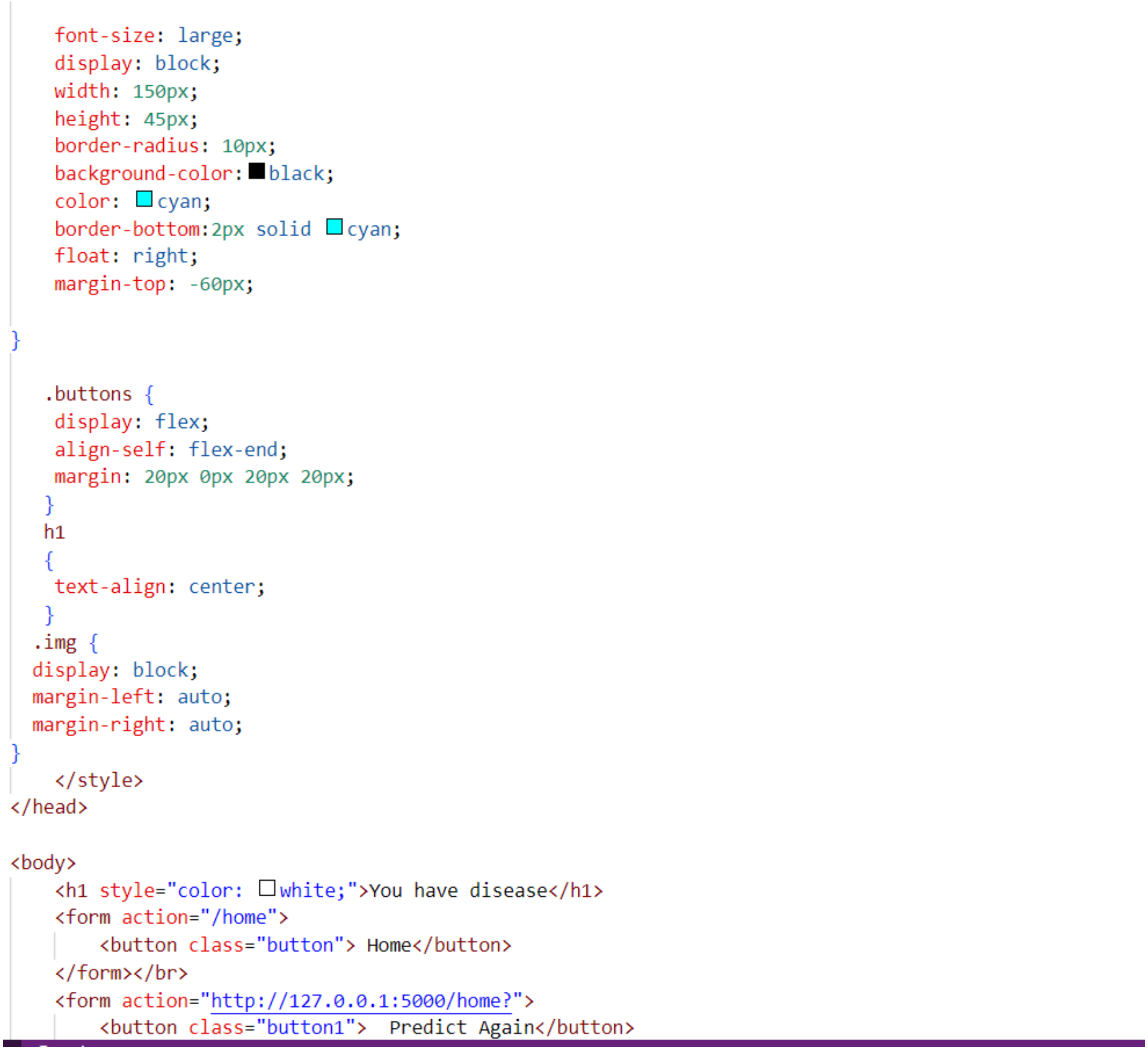


```
                    <label for="aa1">Alamine AminoTransferase:</label><br>
                    <input type="text" id="aa1" name="aa1" placeholder="Alamine AminoTransferase" class="form-control"  required>
                </li><br>
                <li class="fields">
                    <label for="aa2">Aspartate AminoTransferase:</label><br>
                    <input type="text" id="aa2" name="aa2" placeholder="Aspartate AminoTransferase" class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="tp">Total Proteins:</label><br>
                    <input type="text" id="tp" name="tp" placeholder="Total Proteins" class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="a">Albumin:</label><br>
                    <input type="text" id="a" name="a" placeholder="Albumin"  class="form-control" required>
                </li><br>
                <li class="fields">
                    <label for="agr">Albumin Globulin Ratio:</label><br>
                    <input type="text" id="agr" name="agr" placeholder=" Albumin Globulin Ratio" class="form-control" required>
                </li><br>
            </ul>
            <div class="buttons">
                <form action="/">
                    <button class="button"> Go Back </button>
                </form>
                <input type="submit" class="button" value="Predict">
            </div>
        </form>

    </div>
</body>

</html>
```

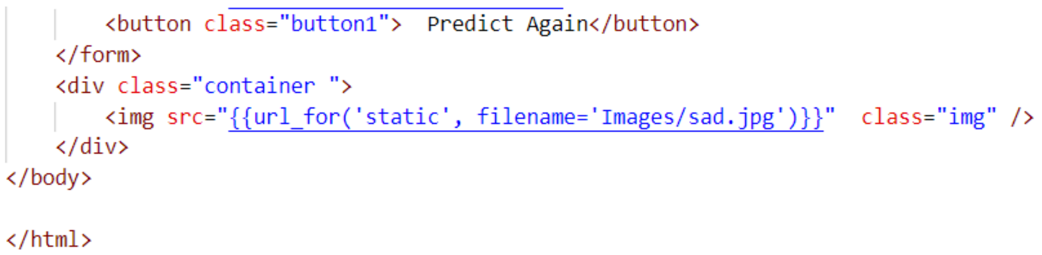
Chance (HTML)



```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Ubuntu&display=swap" rel="stylesheet">

    <style>
        body
        {
            background-color: RGB(6, 206, 251);
        }
    .button {
    margin: 20xp auto;
     }

    .button {

    font-size: large;
    display: block;
    width: 150px;
    height: 45px;
    border-radius: 10px;
    background-color:■black;
    color: ■cyan;
    border-bottom:2px solid ■cyan;

}
.button1 {
    margin: 20xp auto;
     }

    .button1 {
```

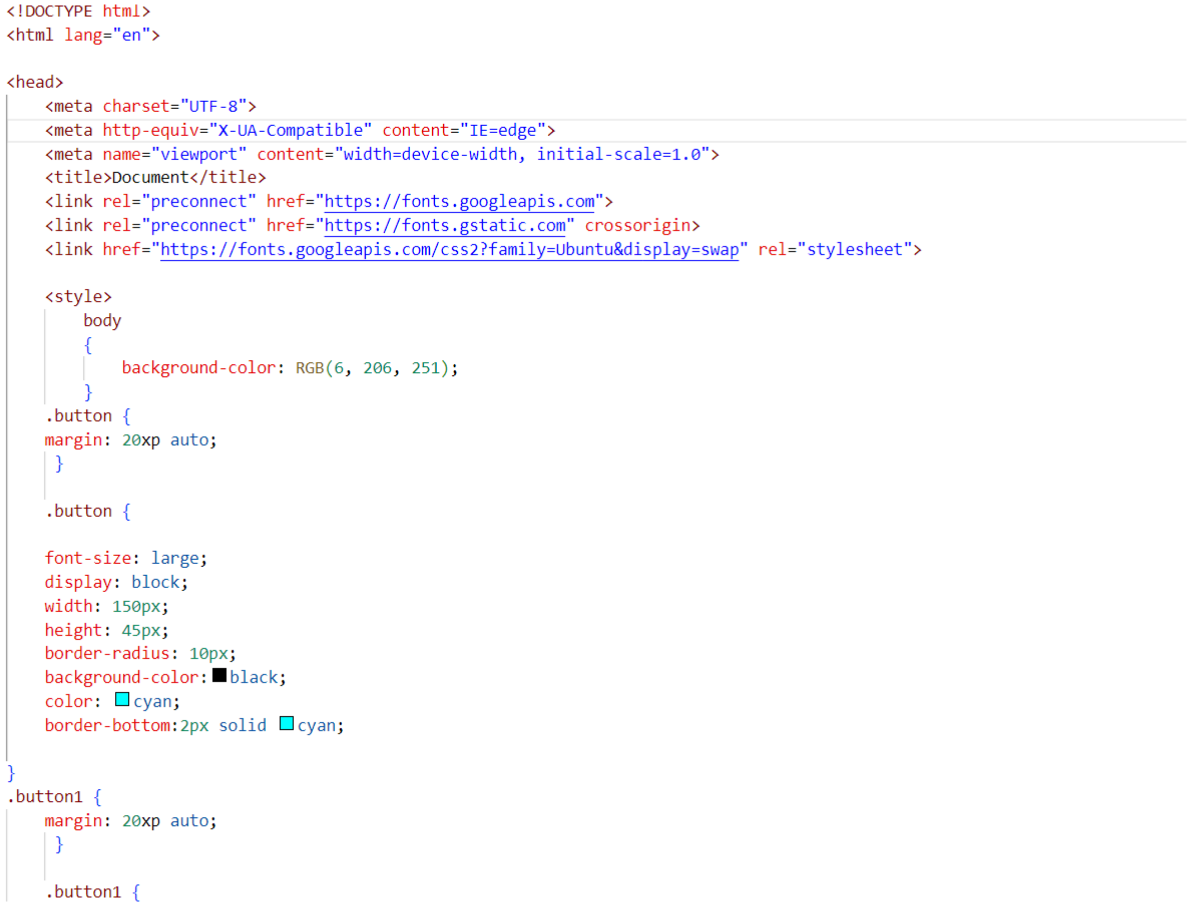


```
    font-size: large;
    display: block;
    width: 150px;
    height: 45px;
    border-radius: 10px;
    background-color:■black;
    color: ■cyan;
    border-bottom:2px solid ■cyan;
    float: right;
    margin-top: -60px;

}

   .buttons {
    display: flex;
    align-self: flex-end;
    margin: 20px 0px 20px 20px;
   }
   h1
   {
    text-align: center;
   }
  .img {
  display: block;
  margin-left: auto;
  margin-right: auto;
}
    </style>
</head>

<body>
    <h1 style="color: □white;">You have disease</h1>
    <form action="/home">
        <button class="button"> Home</button>
    </form></br>
    <form action="http://127.0.0.1:5000/home?">
        <button class="button1">  Predict Again</button>
```

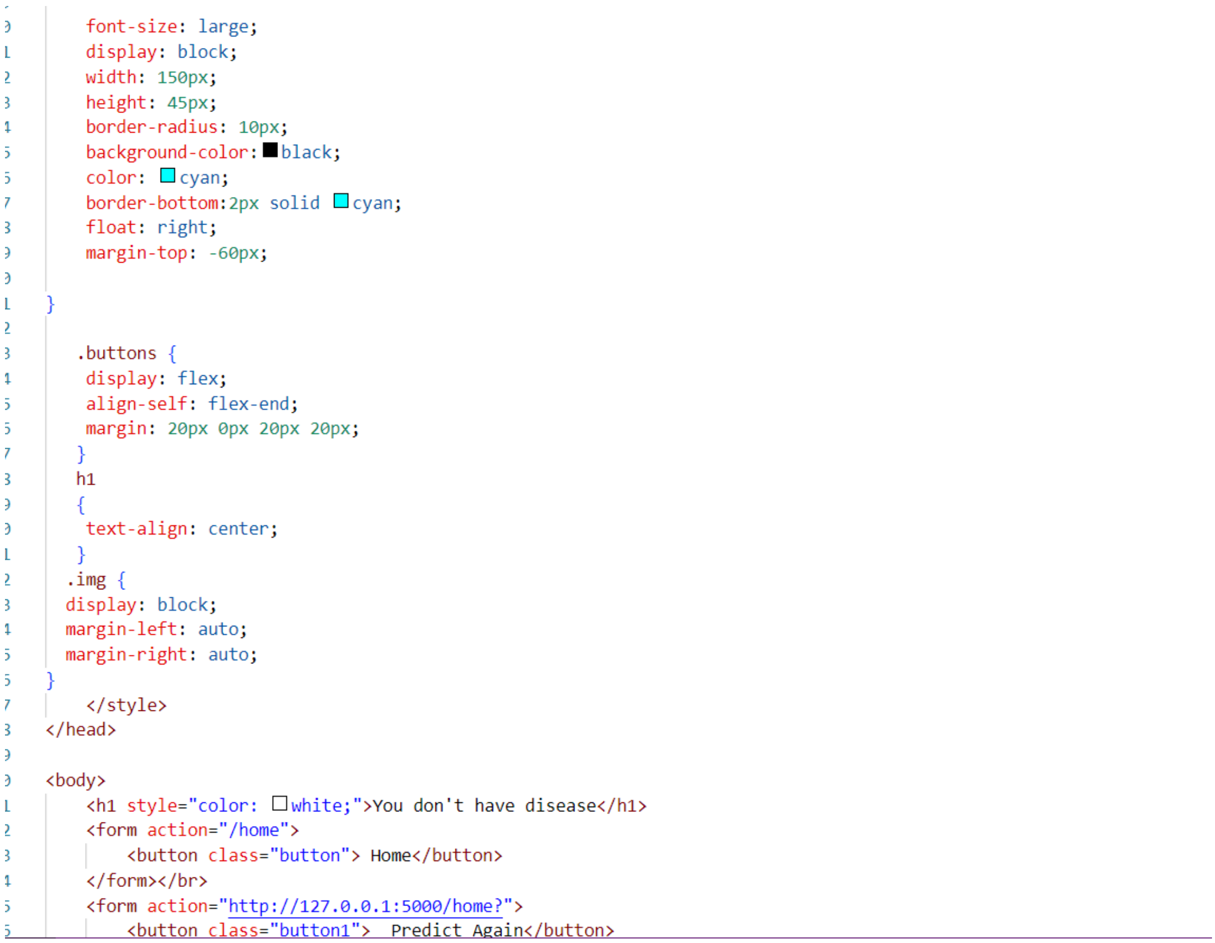


```
        <button class="button1">  Predict Again</button>
    </form>
    <div class="container ">
        <img src="{{url_for('static', filename='Images/sad.jpg')}}"  class="img" />
    </div>
</body>

</html>
```

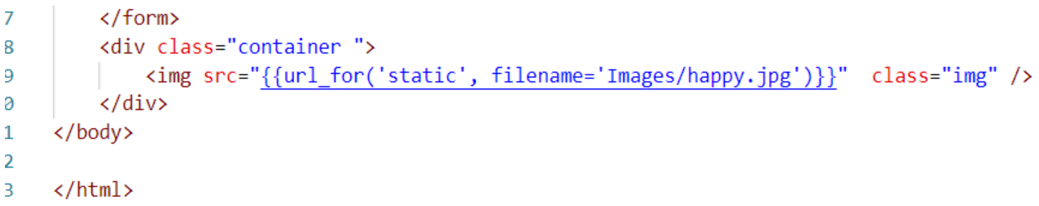
No Chance (HTML)



```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link href="https://fonts.googleapis.com/css2?family=Ubuntu&display=swap" rel="stylesheet">

    <style>
        body
        {
            background-color: RGB(6, 206, 251);
        }
    .button {
    margin: 20xp auto;
     }

    .button {

    font-size: large;
    display: block;
    width: 150px;
    height: 45px;
    border-radius: 10px;
    background-color:■black;
    color: 🟦cyan;
    border-bottom:2px solid 🟦cyan;

}
.button1 {
    margin: 20xp auto;
     }

    .button1 {
```



```
    font-size: large;
    display: block;
    width: 150px;
    height: 45px;
    border-radius: 10px;
    background-color:■black;
    color: ■cyan;
    border-bottom:2px solid ■cyan;
    float: right;
    margin-top: -60px;

}

   .buttons {
    display: flex;
    align-self: flex-end;
    margin: 20px 0px 20px 20px;
   }
   h1
   {
    text-align: center;
   }
  .img {
  display: block;
  margin-left: auto;
  margin-right: auto;
}
    </style>
</head>

<body>
    <h1 style="color: □white;">You don't have disease</h1>
    <form action="/home">
        <button class="button"> Home</button>
    </form></br>
    <form action="http://127.0.0.1:5000/home?">
        <button class="button1">  Predict Again</button>
```



```
    </form>
    <div class="container ">
        <img src="{{url_for('static', filename='Images/happy.jpg')}}"  class="img" />
    </div>
</body>

</html>
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-45942-1660733612