INTERNAL PROJECT

7. INTRODUCTION TO PROJECT

7.1 PROJECT PROFILE

- **Project Title:** HEALTHCARE APP
- **Project Definition:** Healthcare app is a website which uses Machine Learning models to predict whether the user has disease or not based on the inputs given by the user.
- **Project Guide:** Prof. Krishna Samdani
- **Project Duration:** 12 weeks
- **Submitted to:** Mukesh Patel School of Technology Management and Engineering, Mumbai
- **Developed by :** Tanvi Wagle

7.2 INTRODUCTION TO SYSTEM

Healthcare app is a website that uses Machine learning model which will help the users to predict if they suffer from any disease like heart disease, liver disease based on the inputs given by them.
The website shows some analytical graphs which will give information like: how many people suffer from the disease?
This website gives information about the symptoms for the disease and also it gives preventive measures that one can follow.

7.3 SCOPE AND APPLICABILITY OF THE PROJECT

The main objective of this project is to reduce the workload of doctors by using machine learning model to predict if any user is suffering from certain disease or not. Information about the symptoms of the disease and what preventive measures one can take is given on the website.

7.4 TECHNOLOGIES USED

- Python 3.7
- Django framework
- HTML 5, CSS 3, Bootstrap 4, JavaScript
- Chart.js JavaScript Library
- Python Libraries
    - Numpy, Pandas, Matplotlib, Seaborn
    - Sklearn, Imblearn
    - Pickle

IDE: Visual Studio Code, Jupyter Notebook

8. Implementation

8.1 FEATURES/ MODULES

8.1.1 Modules of Project:

- **Models**:
    - **Heart disease prediction:**
        - Heart disease is the leading cause of death for people of most racial and ethnic groups.
        - This model will predict heart disease based on different parameters like age, gender, cholesterol level, blood pressure level, etc.
    - **Liver disease prediction:**
        - Patients with Liver disease have been continuously increasing because of excessive consumption of alcohol, inhale of harmful gases, intake of contaminated food, pickles and drugs.
        - This model will predict liver disease based on different parameters like age, gender, total bilirubin, direct bilirubin, etc.
    - **Brain stroke prediction:**
        - According to the World Health Organization (WHO) stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths.
        - This model will predict whether a patient is likely to get stroke based on the input parameters like gender, age, various diseases, and smoking status.
- **Symptom/ Preventions for diseases:**
    - It will show information about symptoms for the disease as well as the preventive measures that can be followed.

8.1.2 Features of Project:

- Machine Learning model will predict if the user has disease or not based on the inputs given by the user.
- View analytical graphs based on the disease.
- View symptoms for any disease.
- View preventive measures for diseases.

8.2 CODE

Code for Machine learning Models:

Heart_Disease Model:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import pickle

from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_recall_fscore_support,
mean_squared_error, roc_auc_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

# Reading dataset
df_train = pd.read_csv(r'..\Datasets\heart.csv')

# Rename columns for better readability
df_train.columns = ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'cholesterol',
'fasting_blood_pressure',
                    'rest_ecg', 'max_heart_rate_achieved', 'exercise_induced_angina', 'st_depression',
'slope',
                    'num_major_vessels', 'thal', 'target']

# One hot encoding
df_train = pd.get_dummies(data = df_train, columns = ['sex', 'chest_pain_type',
'fasting_blood_pressure', 'rest_ecg',
                                        'exercise_induced_angina', 'slope', 'num_major_vessels', 'thal'],
dtype= 'int')

# Training model
X = df_train.drop('target', axis = 1)
y = df_train['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 5)


# Logistic Regression
logistic = LogisticRegression(max_iter=10000)
logistic.fit(X_train, y_train)
pred = logistic.predict(X_test)

#Calculating accuracy
score = np.around(accuracy_score(y_test, pred), decimals = 3) *100


# Tuple of model and score
model_score = (logistic, score)

# Dumping model and score
pickle.dump(model_score, open('Heart_Disease.sav', 'wb'))
```

Liver_Disease Model:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
from imblearn.combine import SMOTETomek
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_recall_fscore_support,
mean_squared_error, roc_auc_score
from sklearn.tree import DecisionTreeClassifier


# Reading Dataset
df_train = pd.read_csv(r'..\Datasets\indian_liver_patient.csv')

# There are 4 null values in one column of the dataset.
# Adding mean value to the null values
mean_ratio = df_train['Albumin_and_Globulin_Ratio'].mean()

def fill_ratio(df_train, mean_ratio):
    if math.isnan(df_train['Albumin_and_Globulin_Ratio']):
        return mean_ratio
    else:
        return df_train['Albumin_and_Globulin_Ratio']

df_train['Albumin_and_Globulin_Ratio'] = df_train.apply(fill_ratio, axis = 1, args =
(mean_ratio, ))

# Renaming  column for better readability
df_train.rename(columns = {'Dataset': 'Response'}, inplace = True)

# converting categorical values to numerical values
df_train = pd.get_dummies(df_train, columns = ['Gender'], dtype= 'int')

# Model
X = df_train.drop('Response', axis = 1)
y = df_train['Response']

# Oversampling the dataset
smk = SMOTETomek(random_state = 1, sampling_strategy = 'minority')
x_res, y_res = smk.fit_resample(X, y)

# Scaling
standardScaler = StandardScaler()
scale_columns = ['Age', 'Total_Bilirubin', 'Direct_Bilirubin', 'Alkaline_Phosphotase',
'Alamine_Aminotransferase',
                 'Aspartate_Aminotransferase', 'Total_Protiens', 'Albumin',
'Albumin_and_Globulin_Ratio']
```

```
df_train[scale_columns] = standardScaler.fit_transform(df_train[scale_columns])


X_train, X_test, y_train, y_test = train_test_split(x_res, y_res, test_size = 0.3, random_state =
50)

# Decision Tree
dt_clf = DecisionTreeClassifier()
dt_clf.fit(X_train, y_train)
pred = dt_clf.predict(X_test)

#Calculating accuracy
score = np.around(accuracy_score(y_test, pred), decimals = 3) *100


# Tuple of model and score
model_score = (dt_clf, score)

# Dumping model and score
pickle.dump(model_score, open('Liver_Disease.sav', 'wb'))
```

Brain_Stroke Model:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_recall_fscore_support,
mean_squared_error, roc_auc_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

# Read dataset
df_train = pd.read_csv(r'..\Datasets\healthcare-dataset-stroke-data.csv')

# Removing irrelevant columns
df_train.drop(columns = 'id', axis = 1, inplace = True)

# Handling null values
mean_bmi = df_train.groupby(['gender', 'age']).mean()['bmi']
mean_bmi = np.around(mean_bmi, decimals = 3)

def fill_bmi(df_train, mean_bmi):
    if math.isnan(df_train['bmi']):
        return mean_bmi[df_train['gender']][df_train['age']]
```

```
    else:
        return df_train['bmi']

df_train['bmi'] = df_train.apply(fill_bmi, axis = 1, args = (mean_bmi, ))
df_train['bmi'].iloc[2030] = mean_bmi['Female'][0.48]
df_train.drop(index = 3116, inplace = True)

# One hot encoding
df_objects = df_train.select_dtypes(include = 'object')
df_train = pd.get_dummies(df_train, columns= df_objects.columns, dtype = 'int')

# Model
X = df_train.drop('stroke', axis = 1)
y = df_train['stroke']

# Oversampling
smk = SMOTETomek(random_state = 2, sampling_strategy = 'minority')
x_res, y_res = smk.fit_resample(X, y)

# train test split
X_train, X_test, y_train, y_test = train_test_split(x_res, y_res, test_size = 0.3, random_state =
17)

#Logistic Regression

logistic = LogisticRegression(max_iter=10000)
logistic.fit(X_train, y_train)
pred = logistic.predict(X_test)

#Calculating accuracy
score = np.around(accuracy_score(y_test, pred), decimals = 3) *100


# Tuple of model and score
model_score = (logistic, score)

# Dumping model and score
pickle.dump(model_score, open('Stroke_Prediction.sav', 'wb'))
```

8.3 FUNCTIONAL DESIGN (SCREEN SHOTS)

HOME PAGE

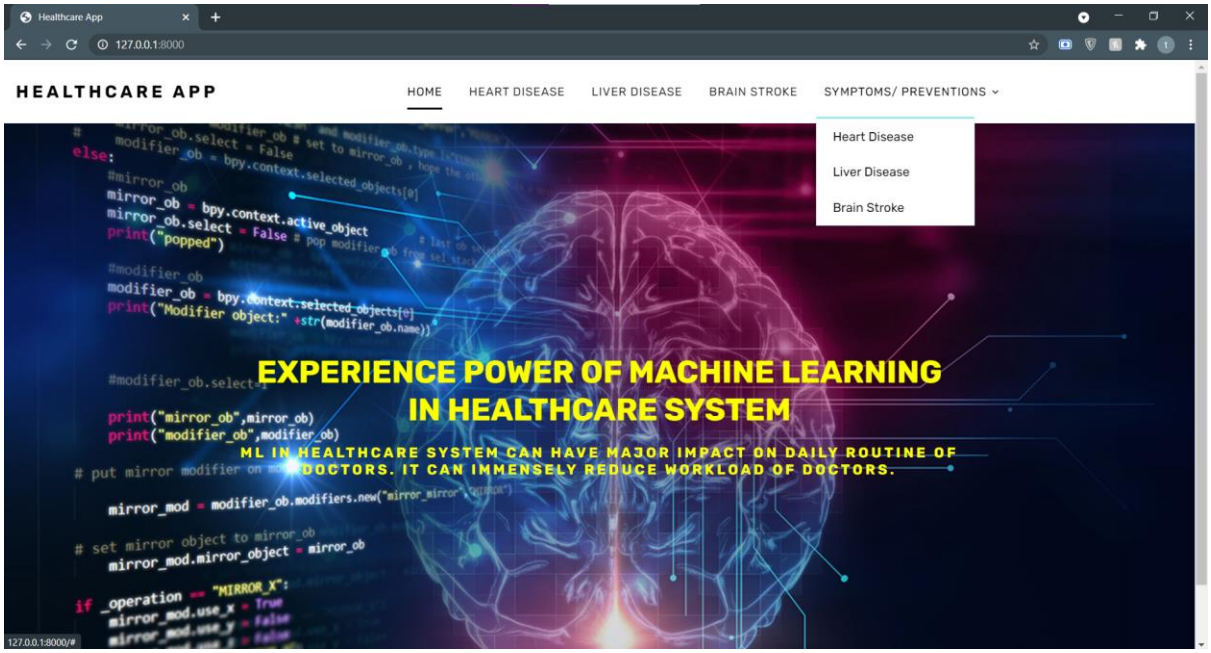

Figure 6: Healthcare app home page

HEART DISEASE:

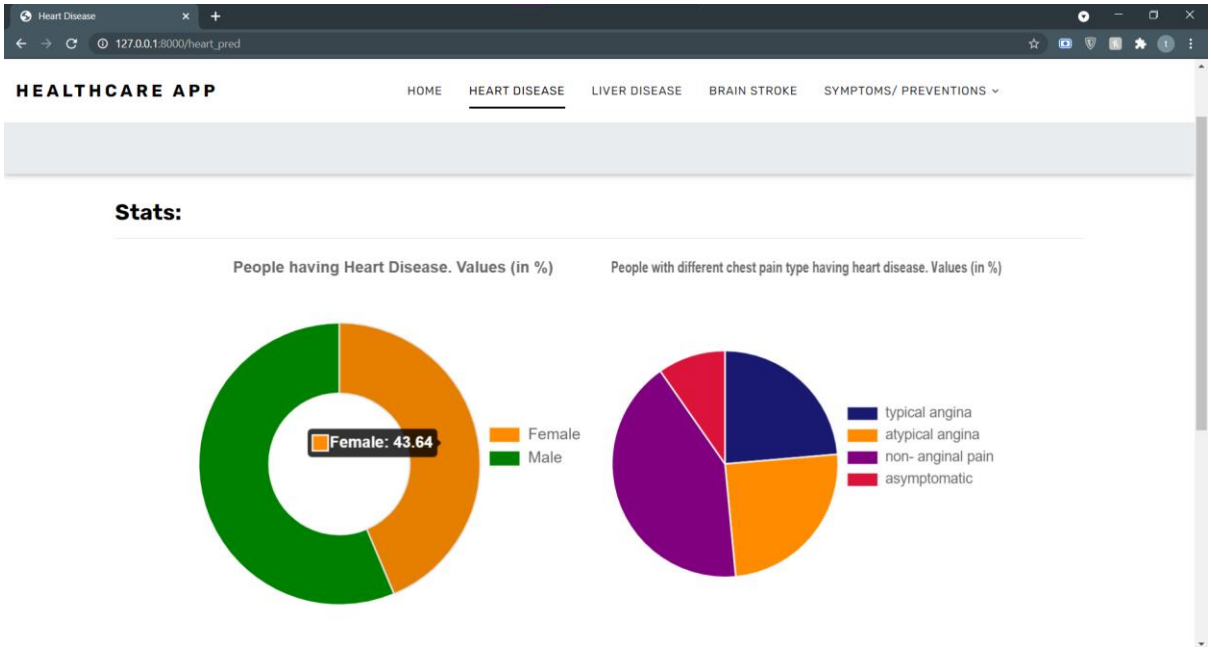

Figure 7.a: Heart disease graphs

Figure 7.b: Line graph

HEART DISEASE PREDICTION:

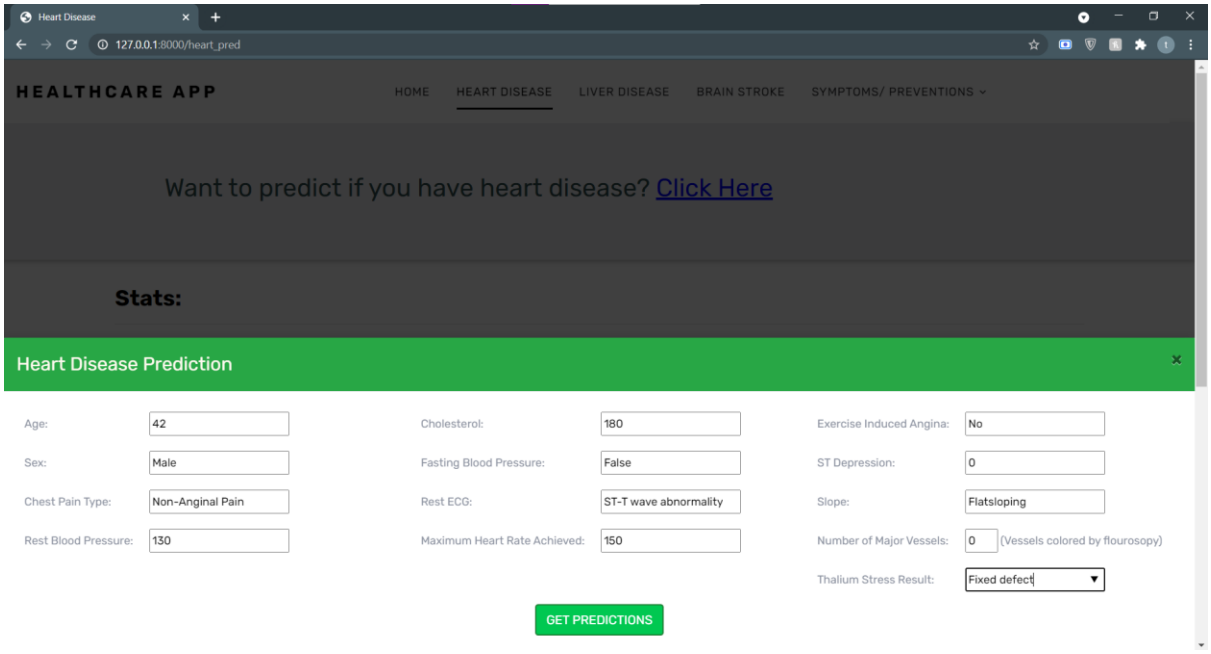

Figure 7.c: Heart Disease Prediction Modal

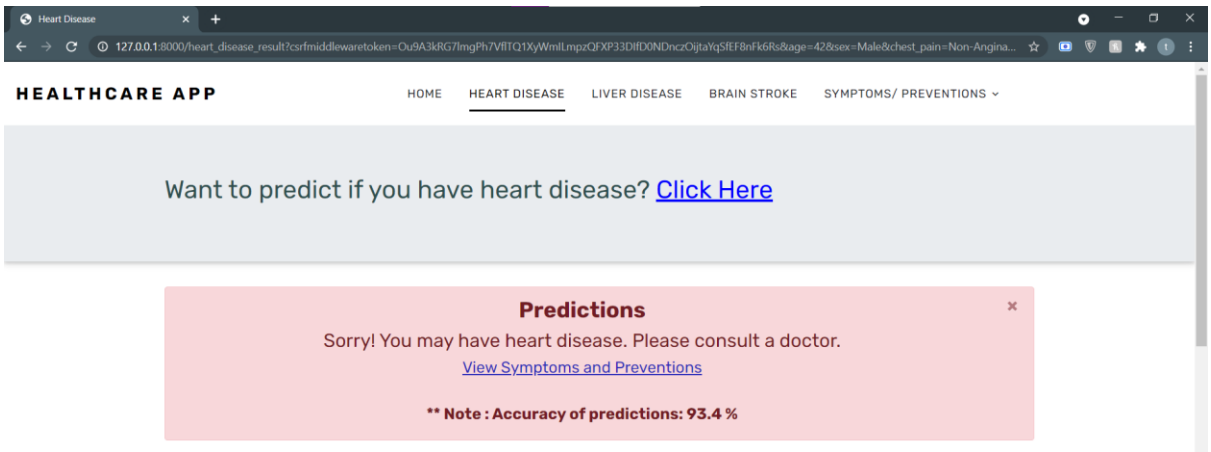

Figure 7.d: Predictions result

LIVER DISEASE:

Figure 8.a: Liver Disease page

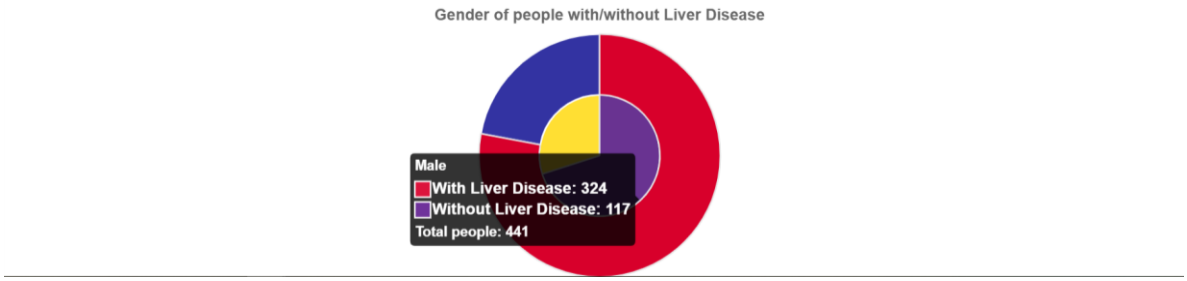

Figure 8.b: Pie Chart

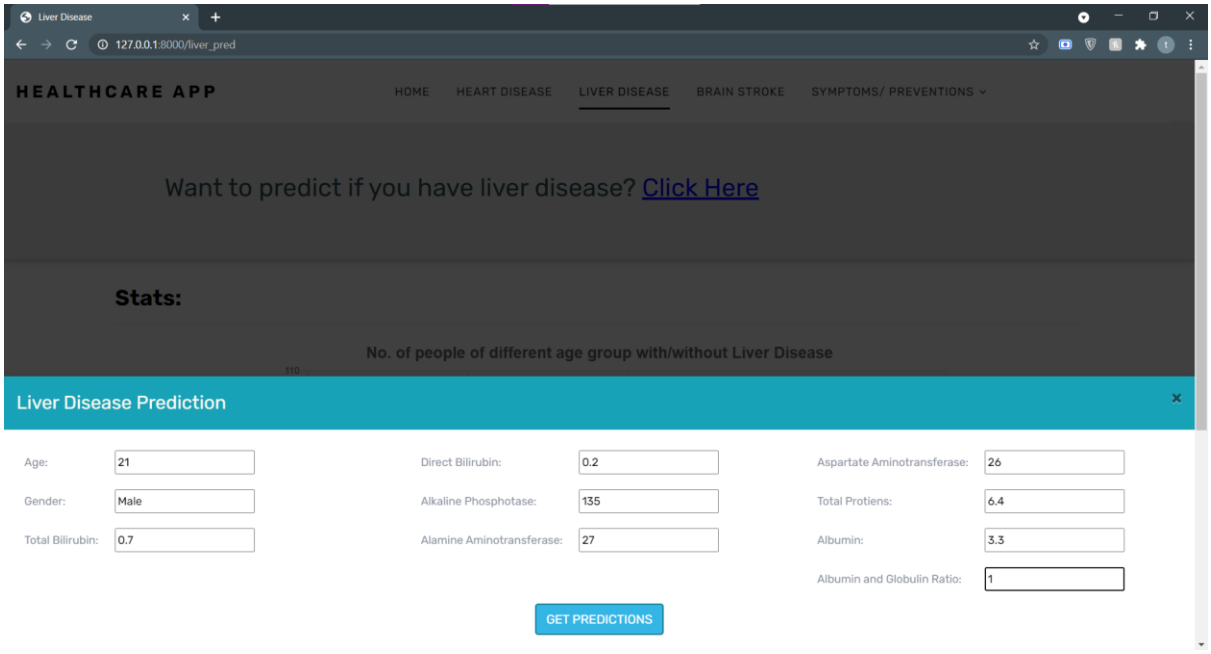

Figure 8.c: Liver Disease Prediction Modal

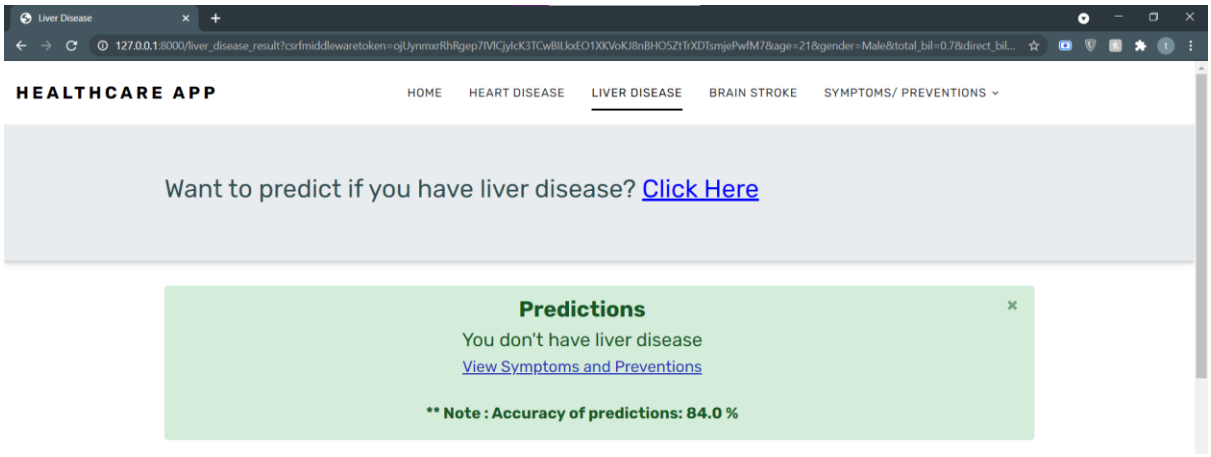

Figure 8.d: Predictions result

BRAIN STROKE:

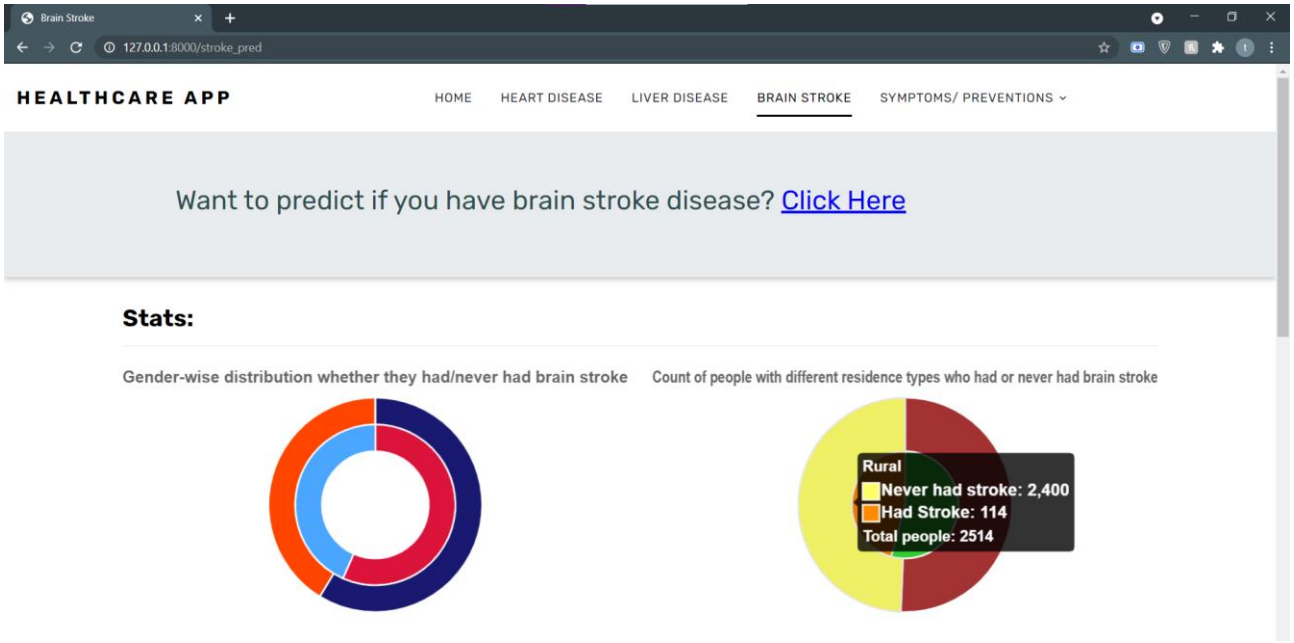

Figure 9.a: Brain Stroke page

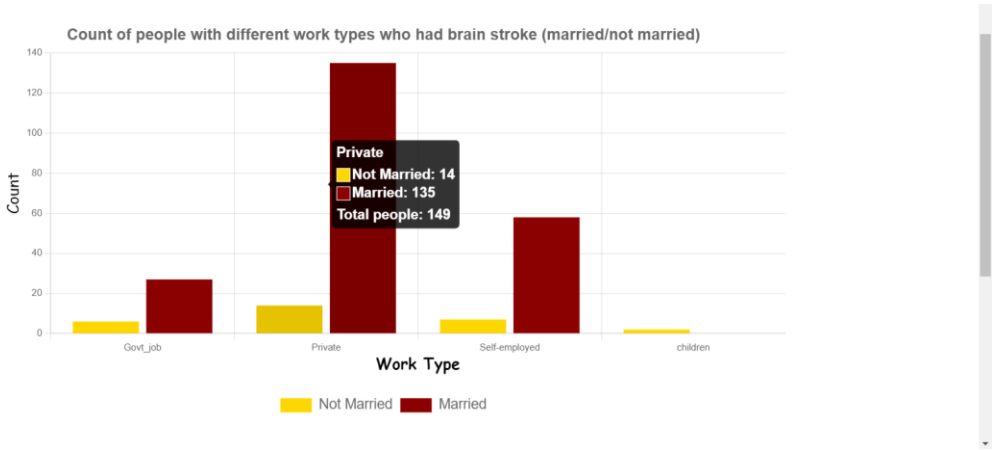

Figure 9.b: Bar graph for work type

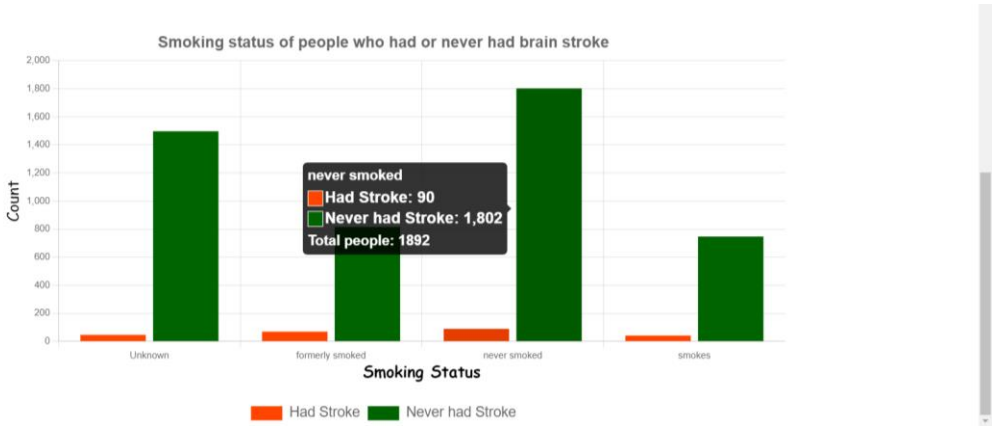

Figure 9.c: Bar graph for smoking status

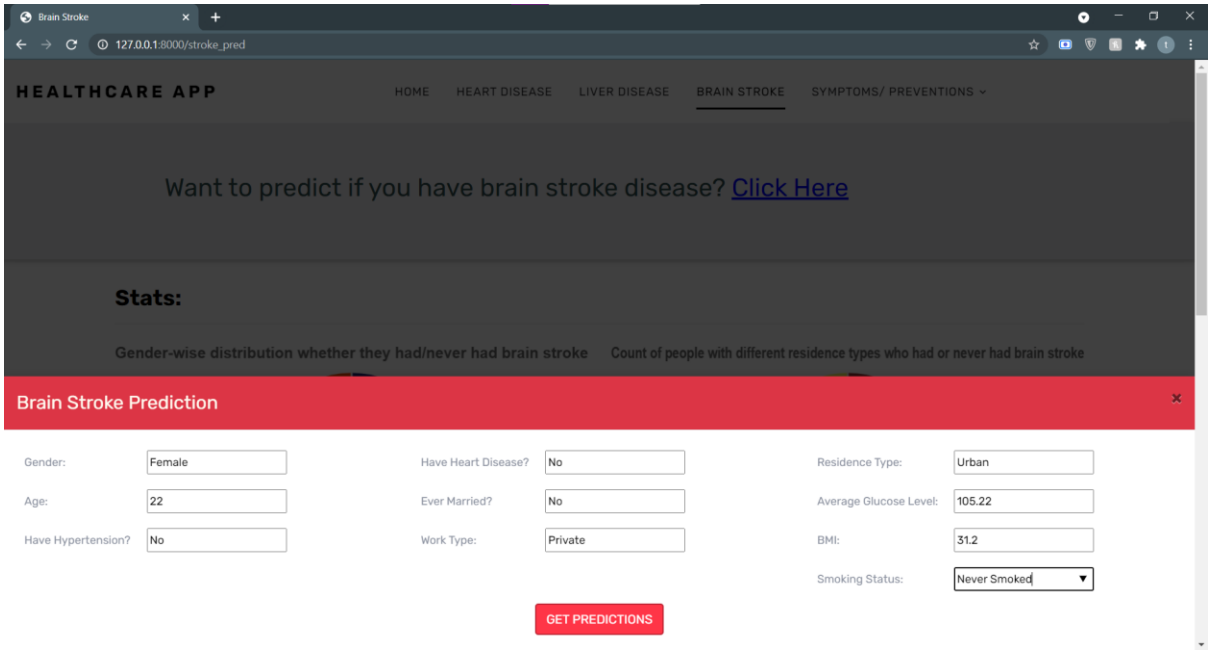

Figure 9.d: Brain Stroke Prediction Modal

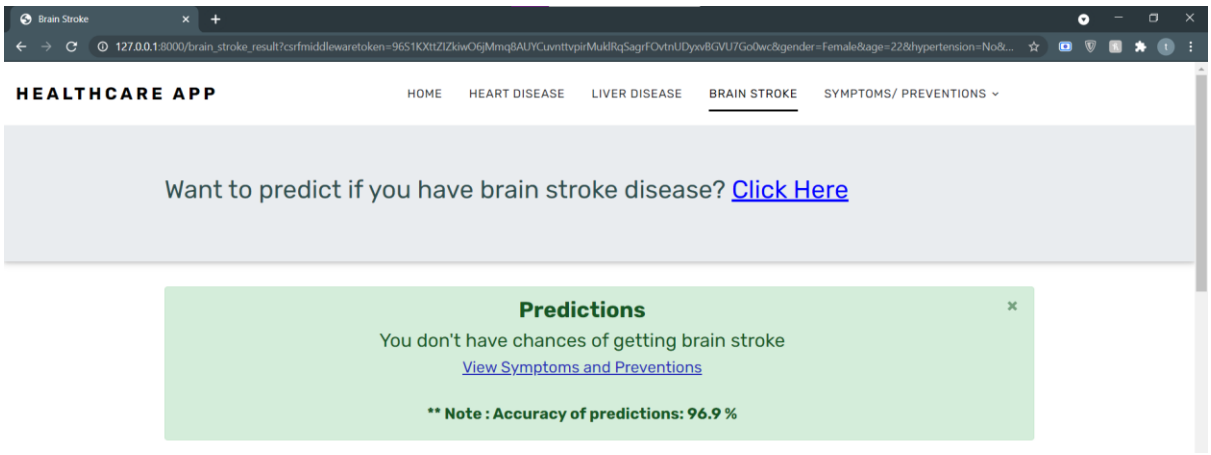

Figure 9.e: Predictions result

SYMPTOMS/ PREVENTIVE MEASURES:

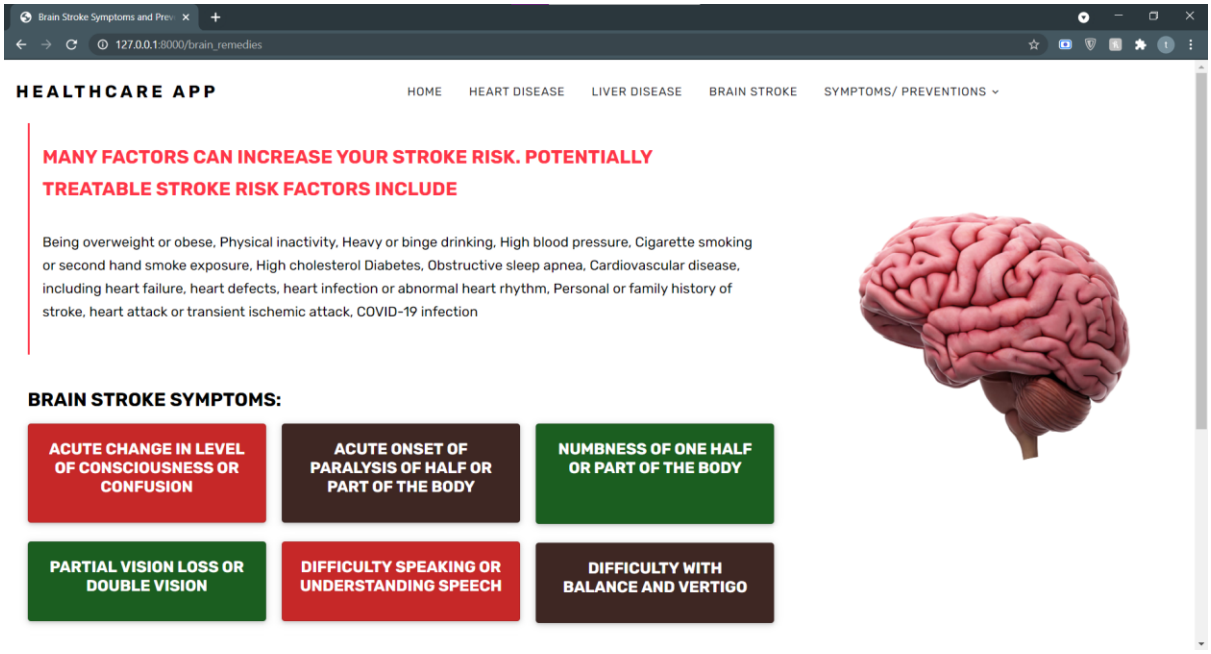

Figure 10.a: Brain Stroke Symptoms

Figure 10.b: Brain Stroke preventive measures

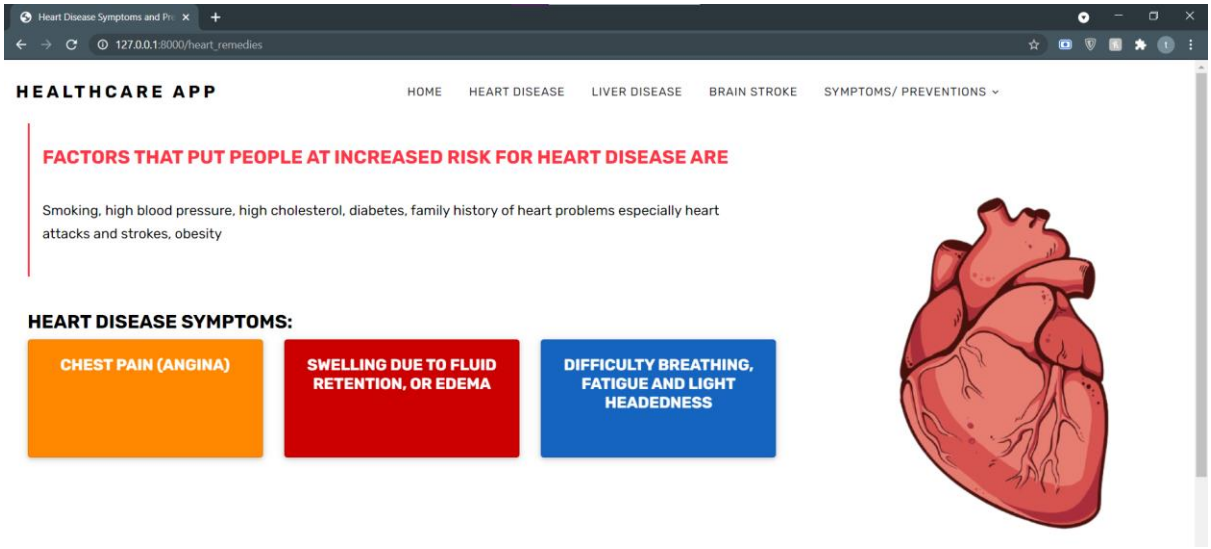

Figure 11.a: Heart Disease symptoms

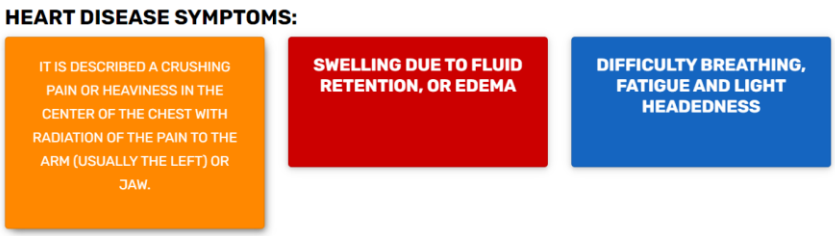

Figure 11.b: On Hover flip effect of symptom

Figure 11.c: Heart Disease preventive measures

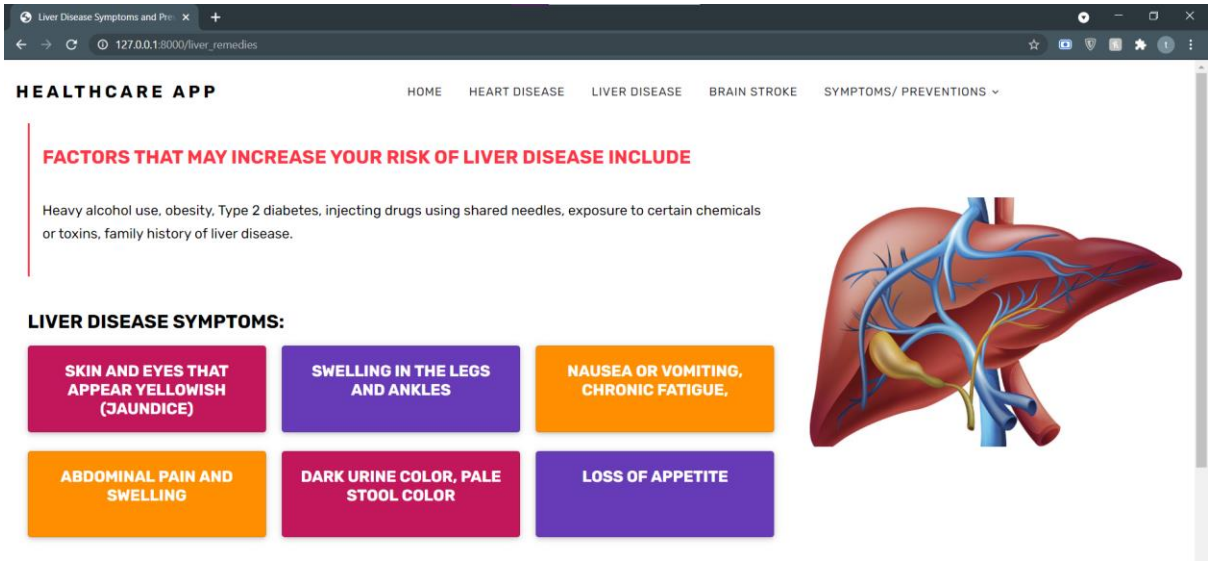

Figure 12.a: Liver Disease Symptoms

Figure 12.b: Liver Disease preventive measures

8.4 WORK DONE ON PROJECT

1. Search and download of datasets.
2. Data preprocessing.
3. Exploratory Data Analysis.
4. Selecting model based on score
5. Making user interface for the website
6. Integrating model into website
7. Adding charts using chart.js
8. Add Symptoms/ Preventions

9. Key Learnings

- Learning and implementing various machine learning models, EDA, how to optimize a model.
- Learning Django as web framework for python.
- Integration of model with User interface.
- Add chart on website using chart.js and how to change its configurations

10.Challenges

- Balancing the dataset and optimize the model.
- Add graphs on the website using chart.js
- Learning and implementing Django

11.Conclusion and Future Expansion

Conclusion:

- Healthcare project is very crucial as it is related to medical. One wrong prediction can mislead a patient. It is very important to optimize the model.
- While making this project, I have learnt about different machine learning models and to calculate their scores and how to optimize the model. Also, I have about different python libraries like imblearn, sklearn, seaborn, etc.
- Also, I have learnt how to implement and integrate machine learning model with user interface.
- I have learnt various new technologies in my internship period and while making my college project like Django, codeigniter, flutter.

Future Expansion:

I would like to integrate concept of image processing in healthcare project based on which various other disease could be predicted.

12.References

1. Datasets:
    a. https://www.kaggle.com/uciml/indian-liver-patient-records
    Upload Date: 2017-09-20
    b. https://www.kaggle.com/fedesoriano/stroke-prediction-dataset
    Upload Date: 2021-01-27
    c. https://www.kaggle.com/ronitf/heart-disease-uci
    Upload Date: 2018-06-25
2. Courses completed for this project:
    a. Python and Django Bootcamp:
    https://www.udemy.com/certificate/UC-15549f73-e326-4d86-a325-da196f784ed9/
    b. Data Science and Machine Learning Bootcamp:
    https://www.udemy.com/certificate/UC-f6d4fa22-680e-4e67-8496-6e35fe077058/
3. https://towardsdatascience.com/creating-a-machine-learning-based-web-application-using-django-5444e0053a09
4. https://www.chartjs.org/docs/latest/