Project ETE Report on

Health Seer: An AI based disease predictor

Submitted in partial fulfillment of the
requirement for the award of the degree of

B.Tech CSE

Under The Supervision
of Name of Supervisor
: Dr. Aanjey Mani
Tripathi

Submitted By

Nishant Kumar Pandey (18SCSE1010366)
Nishant Sinha (18SCSE1010381)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
OCTOBER, 2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"HEALTH SEER: AI ML BASED DISEASES PREDICTOR"** in partial fulfillment of the requirements for the award of Bachelor of Technology submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of August 2021 to December 2021, under the supervision of Dr. Aanjey Mani Tripathi, Department of Computer Science and Engineering, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Nishant Sinha (18SCSE1010381)
Nishant Kumar Pandey (18SCSE1010366)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor Name
Dr. Aanjey Mani Tripathi

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of Nishant Sinha (18SCSE1010381) and Nishant Kumar Pandey(18SCSE1010366) has been held on 20/12/2021 and their work is recommended for the award of Bachelor of Technology.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: 20, December, 2021
Place: Greater Noida

Table of Contents

List of Figures

Microsoft Word - SCSE Project Review-2 Format.docx

Nishant Kumar Pandey (18SCSE1010366)

Nishant Sinha(18SCSE1010381)

GALGOTIAS UNIVERSITY

Under the guidance of

Dr.AanjeyManiTripathi

Abstract

Artificial intelligence has witnessed an extensive growth in the medical field in the recent decades. It has proved its mettle in prediction of some of the most fatal diseases such as various types of cancer, heart disease, diabetes and now also has a growing popularity in the mental health domain. Due to the wide availability of patient data, it has become easier to analyse the patterns through it to reach certain clinical outcomes such as providing insights to the doctors regarding a particular health condition in order to make treatment easier. In addition, AI can reduce inevitable human errors and draw inferences from large chunks of data which aids in prediction of the health risks and health outcome.

This application aims to use various datasets of different diseases and by using ML algorithms, predict certain outcomes related to them. The user can feed their data to the app based on a questionnaire after which the app will use the pre-existing dataset to match and predict an outcome. The health risks and the info about possible diseases that the user may contract in the near future will be displayed using easily understandable graphs. Patients will also get some health advisory based on their report.

Keywords: AI: artificial intelligence, ML: machine learning

Microsoft Word - SCSE Project Review-1 Format.docx

Introduction

Artificial intelligence has come a long way when it comes to predicting diseases in the past few decades. It has proved beneficial in reducing human error to a significant level by making more precise predictions and by finding previously unknown patterns in the dataset. Due to the availability of large patient databases, it has become easier to find complex patterns in the datasets by using machine learning algorithms such as K-NN, Naïve Bayes, SVM etc. with an accuracy averaging beyond 95%.

The datasets for our present application is gathered from online repositories of health data such as Kaggle Competition, Amazon public datasets and Microsoft research. This dataset will be used for further predictive analysis. Our application will provide the user easily understandable graphical representation of their health analysis along with some suggestions based on the outcome.

Types of classification algorithms:

a. Naïve Bayes

- In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features.
- They are among the simplest Bayesian network models, but coupled with kernel density estimation, they can achieve higher accuracy levels.
- Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.
- For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum

likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

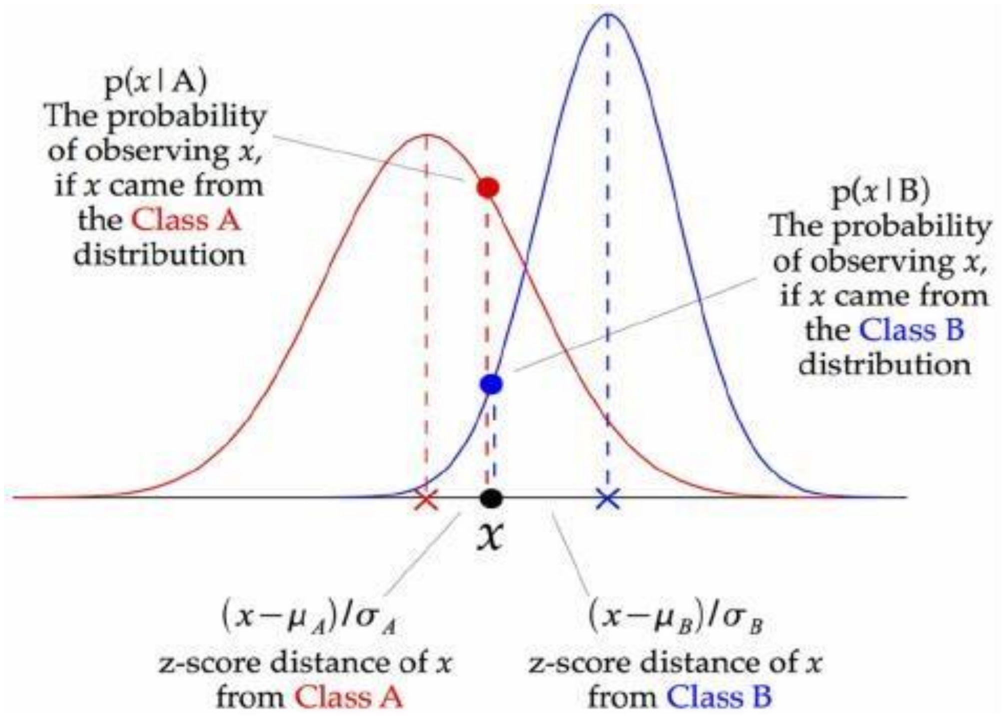
- Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers.[6] Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

b. <u>Random forest</u>

- Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.
- For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned.
- For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set.
- Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.
- Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

c. **Gaussian naïve bayes**

- Gaussian Naive Bayes is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data.
- Naive Bayes are a group of supervised machine learning classification algorithms based on the Bayes theorem. It is a simple classification technique, but has high functionality. They find use when the dimensionality of the inputs is high. Complex classification problems can also be implemented by using Naive Bayes Classifier.
- Naive Bayes Classifiers are based on the Bayes Theorem. One assumption taken is the strong independence assumptions between the features. These classifiers assume that the value of a particular feature is independent of the value of any other feature.
- In a supervised learning situation, Naive Bayes Classifiers are trained very efficiently. Naive Bayed classifiers need a small training data to estimate the parameters needed for classification.
- Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.
- Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.
- An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.
- An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.


p(x | A)
The probability of observing x, if x came from the Class A distribution
p(x | B)
The probability of observing x, if x came from the Class B distribution
x
$(x-\mu_A)/\sigma_A$
z-score distance of x from Class A
$(x-\mu_B)/\sigma_B$
z-score distance of x from Class B

- The above illustration indicates how a Gaussian Naive Bayes (GNB) classifier works. At every data point, the z-score distance between that point and each class-mean is calculated, namely the distance from the class mean divided by the standard deviation of that class.

Literature Review

a. Med - BERT

A disease predictor called Med – BERT was a disease predictor was made by University of Texas. This model was based on deep learning algorithms and proved to be beneficial in many clinical tasks. Using BERT (bidirectional encoder representations from transformers) and related models achieved high accuracy rates. The pre training of BERT helped in producing contextualized embeddings which could be applied to smaller finely tuned datasets in order to boost their performance.
Source - https://arxiv.org/ftp/arxiv/papers/2005/2005.12833.pdf

b. Prediction of heart disease and classifiers' sensitivity analysis

According to this article, using algorithms such as K- NN, Naive Bayes, Decision tree J48, JRip, SVM, Adaboost, Stochastic Gradient Decent (SGD) and Decision Table (DT) proved to be highly efficient in predictive analysis gave results with accuracy reaching well above 95%. It was shown that using different classifications helped in achieving high accuracy with few number of attributes.

Source - https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-03626-y

c. PMC' s predictive analysis in health care

In this study published by PMC, various measures which should be taken in algorithm selection for predictive analysis of diseases is taken into light. Different factors such as differences between academic hospital patients and regional hospital patients is also considered to be important in selection of the algorithms. It was concluded in this study that for complex algorithms, there was a need for alternative and innovative solutions.

Source - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6857503/

d. Top AI algorithms in healthcare

According to Analytics India magazine, some of the most widely used algorithms for disease prediction are – Support Vector Machines (SVM), Artificial Neural Networks (CNN & RNN), Logistic Regression, Random Forest, Discriminant Analysis and Naïve Bayes. These algorithms were mostly disease specific, for instance- Random forest is used in predicting risk of disease from ECG and MRI analysis and Discriminant Analysis proved useful in early diagnosis of diabetic Peripheral Neuropathy.

Source - https://analyticsindiamag.com/top-6-ai-algorithms-in-healthcare/

REQUIRED TOOLS

A. TensorFlow

- TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

- TensorFlow was developed by the Google Brain team for internal Google use in research and production. The initial version was released under the Apache License 2.0 in 2015.

- TensorFlow can be used in a wide variety of programming languages, most notably Python, as well as Javascript, C++, and Java.This flexibility lends itself to a range of applications in many different sectors.

- TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

Features:

a. **Auto Differentiation** – Auto Differentiation is the process of automatically calculating the gradient vector of a model with respect to each of its parameters. With this feature, TensorFlow can automatically compute the gradients for the parameters in a model, which is useful to algorithms such as backpropagation which require gradients to optimize performance. To do so, the framework must keep track of the order of operations done to the input Tensors in a model, and then compute the gradients with respect to the appropriate parameters.

b. **Eager execution -** TensorFlow includes an "eager execution" mode, which means that operations are evaluated immediately as opposed to being added to a computational graph which is executed later.Code executed eagerly can be examined step-by step-through a debugger, since data is augmented at each line of code rather than later in a computational graph. This execution paradigm is considered to be easier to debug because of its step- by- step transparency.

c. **Distribute** - In both eager and graph executions, TensorFlow provides an API for distributing computation across multiple devices with various distribution strategies. This distributed computing can often speed up the execution of training and evaluating of TensorFlow models and is a common practice in the field of AI.

d. **Losses -** To train and assess models, TensorFlow provides a set of loss functions (also known as cost functions). Some popular examples include mean squared error (MSE) and binary cross entropy (BCE).These loss functions compute the "error" or "difference" between a model's output and the expected output (more broadly, the difference between two tensors). For different datasets and models, different losses are used to prioritize certain aspects of performance.

e. **Metrics -** In order to assess the performance of machine learning models, TensorFlow gives API access to commonly used metrics. Examples include various accuracy metrics (binary, categorical, sparse categorical) along with other metrics such as Precision, Recall, and Intersection-over-Union (IoU).

f. **TF.nn** - TensorFlow.nn is a module for executing primitive neural network operations on models. Some of these operations include variations of convolutions (1/2/3D, Atrous, depthwise), activation functions (Softmax, RELU, GELU, Sigmoid, etc.) and their variations, and other Tensor operations (max-pooling, bias-add, etc.).

g. **Optimizers -** TensorFlow offers a set of optimizers for training neural networks, including ADAM, ADAGRAD, and Stochastic Gradient Descent (SGD). When training a model, different optimizers offer different modes of parameter tuning, often affecting a model's convergence and performance.

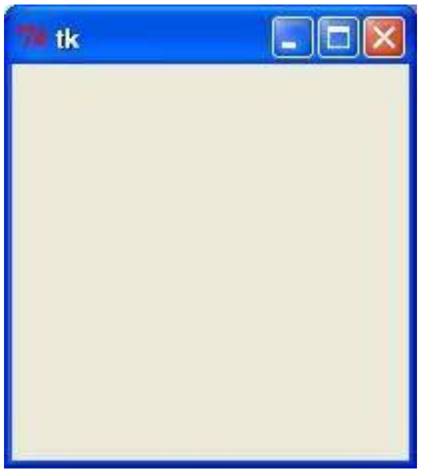
B. Tkinter

- Tkinter is the Python interface to the Tk GUI toolkit shipped with Python.
- Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.
- Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –
  a. Import the Tkinter module.
  b. Create the GUI application main window.
  c. Add one or more of the above-mentioned widgets to the GUI application.
  d. Enter the main event loop to take action against each event triggered by the user.

- Example:

```
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
    top.mainloop()
```

This would create a following window –



Tkinter Widgets

- Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.
- There are currently 15 types of widgets in Tkinter :

a. **Button** - The Button widget is used to display buttons in your application.

b. **Canvas -** The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.

c. **Checkbutton -** The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time.

d. **Entry -** The Entry widget is used to display a single-line text field for accepting values from a user.

e. **Frame -** The Frame widget is used as a container widget to organize other widgets.

f. **Label** - The Label widget is used to provide a single-line caption for other widgets. It can also contain images.

g. **Listbox** - The Listbox widget is used to provide a list of options to a user.

h. **Menubutton -** The Menubutton widget is used to display menus in your application.

i. **Menu -** The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.

j. **Message -** The Message widget is used to display multiline text fields for accepting values from a user.

k. **Radiobutton -** The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.

l. **Scale -** The Scale widget is used to provide a slider widget.

m. **Scrollbar** - The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes.

n. **Text** - The Text widget is used to display text in multiple lines.

o. **Toplevel -** The Toplevel widget is used to provide a separate window container.

Geometry Management

- All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area.
- Tkinter exposes the following geometry manager classes: pack, grid, and place.
- The pack() Method − This geometry manager organizes widgets in blocks before placing them in the parent widget.
- The grid() Method − This geometry manager organizes widgets in a table-like structure in the parent widget.
- The place() Method − This geometry manager organizes widgets by placing them in a specific position in the parent widget.

C. **Pandas**

- pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

- It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.Its name is a play on the phrase "Python data analysis" itself.

Library Features:

- DataFrame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation[6] and frequency conversions, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

Dataframes:

Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features.

D. **Numpy:**

- NumPy is a Python library used for working with arrays.
- It also has functions for working in domain of linear algebra, fourier transform, and matrices.
- In Python we have lists that serve the purpose of arrays, but they are slow to process.
- NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
- The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
- Arrays are very frequently used in data science, where speed and resources are very important.
- NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.
- This behavior is called locality of reference in computer science.
- This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.
- NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

E. **Seaborn;**

- Seaborn is an open- source Python library built on top of matplotlib.

- It is used for data visualization and exploratory data analysis.

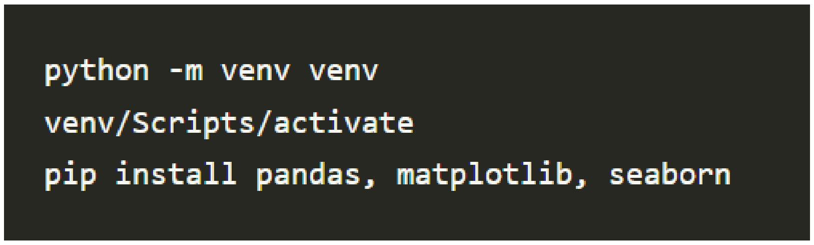
- Seaborn works easily with data frames and the Pandas library.

- The graphs created can also be customized easily.

- Graphs can help us find data trends that are useful in any machine learning or forecasting project.

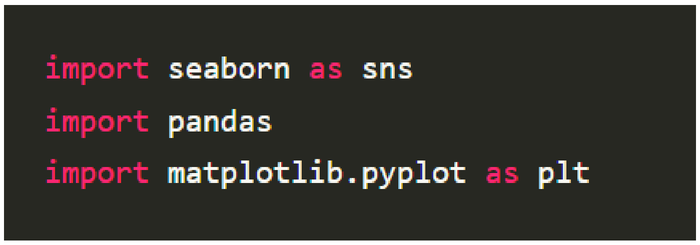
    - Graphs make it easier to explain your data to non-technical people.
    - Visually attractive graphs can make presentations and reports much more appealing to the reader.
    - A good understanding of python.
    - Some experience working with the Pandas Library.
    - Some experience working with the Matplotlib Library.
    - A basic understanding of data analysis.

- Installing Seaborn:



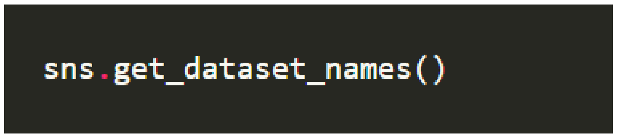
```
python -m venv venv
venv/Scripts/activate
pip install pandas, matplotlib, seaborn
```

- Import Seaborn and loading dataset:



```
import seaborn as sns
import pandas
import matplotlib.pyplot as plt
```

- Seaborn has 18 in-built datasets, that can be found using the following command.



```
sns.get_dataset_names()
```

- We will be using the Titanic dataset for this tutorial.

- Count Plot:

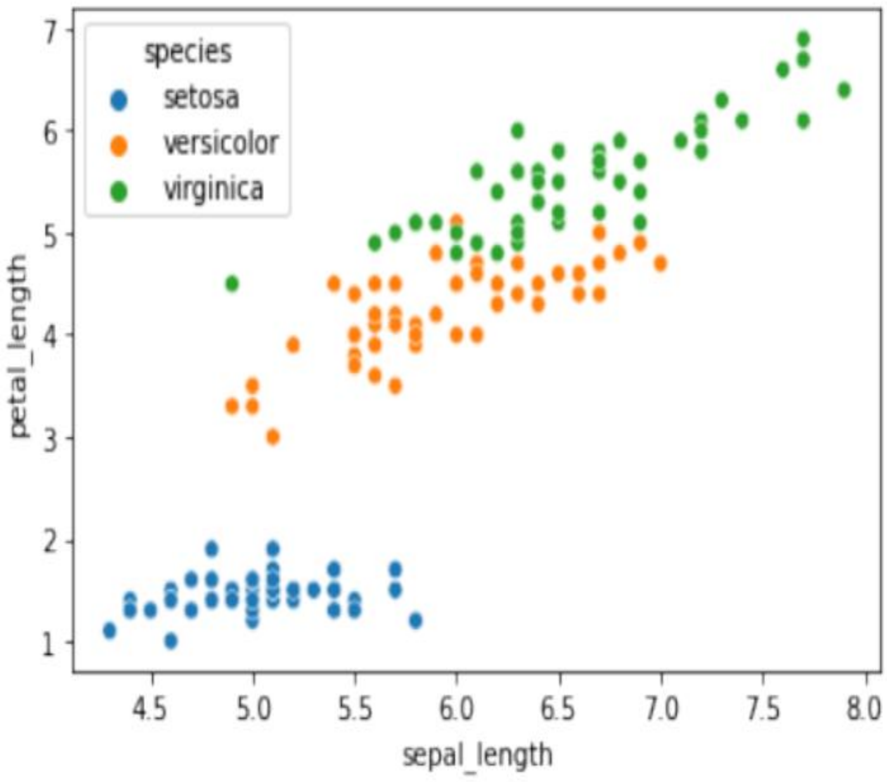
    - A count plot is helpful when dealing with categorical values.
    - It is used to plot the frequency of the different categories.
    - **data** - The dataframe.
    - **x** - The name of the column.
    - We can observe from the graph that the number of male passengers is significantly higher than the number of female passengers.
    - We can further break up the bars in the count plot based on another categorical variable. The color palette of the plot can also be customized.
    - hue - The name of the categorical column to split the bars.
    - palette - The color palette to be used. For a list of color palettes, check out matplotlib’s documentation.

- Scatter Plot:
  - For this plot and the plots below, we will be working with the iris dataset. The iris dataset contains data related to flower's petal size (petal length and petal width) and sepal size (sepal length and sepal width).

```
df = sns.load_dataset('iris')
df.head()
```

  - For this plot and the plots below, we will be working with the iris dataset. The iris dataset contains data related to flower's petal size (petal length and petal width) and sepal size (sepal length and sepal width).
  - First, we will need to load the iris dataset.
  - Scatter plots help understand co-relation between data,

```
sns.scatterplot(x='sepal_length', y ='petal_length' ,
data = df , hue = 'species')
```

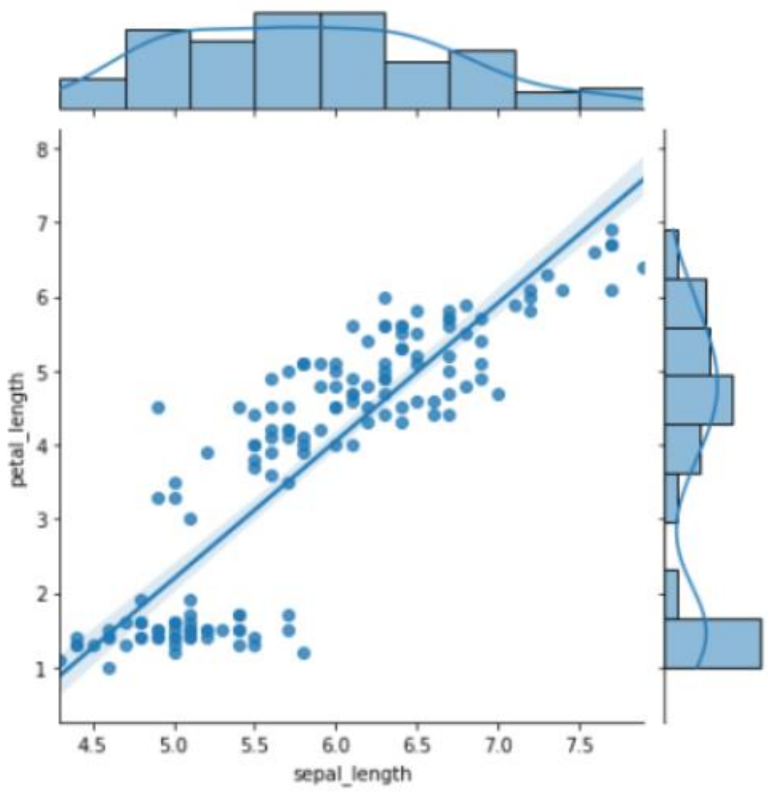


- A scatterplot requires data for its **x-axis** and **y-axis**. We can also pass a value for the **hue** parameter to color the dots based on a categorical column.
- In the plot above we can observe that an iris flower with a sepal length < 6cm and petal length > 2cm is most likely of type **setosa**.

- Although there is no distinct boundary present between the **versicolor** dots and **virginica** dots, an iris flower with petal length between 2cm and 5cm is most likely of type **versicolor**, while iris flowers with petal length > 5cm are most likely of type **virginica**.

- Joint Plot:
    - A Joint Plot is also used to plot the correlation between data.

```
sns.jointplot(x='sepal_length' , y ='petal_length',
data = df , kind = 'reg')
```

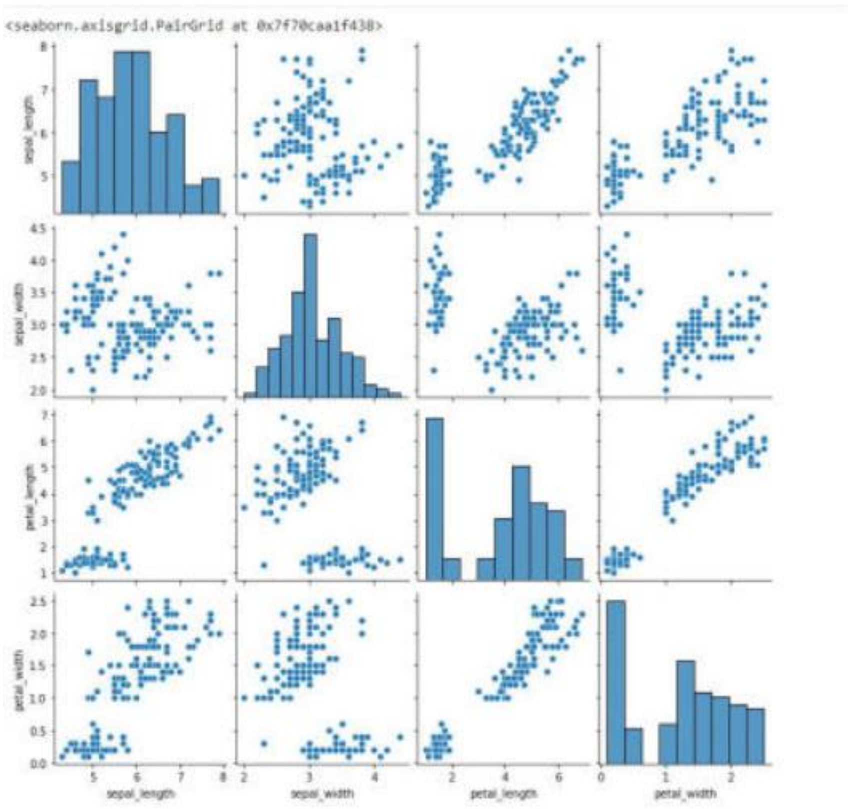


- kind - The kind of plot to be plotted. It can be one of the following.
- 'scatter', 'hist', 'hex', 'kde', 'reg', 'resid'

- Pair Plots:

    - Seaborn lets us plot multiple scatter plots. It’s a good option when you want to get a quick overview of your data.

```
sns.pairplot(df)
```



```
<seaborn.axisgrid.PairGrid at 0x7f70caa1f438>
```

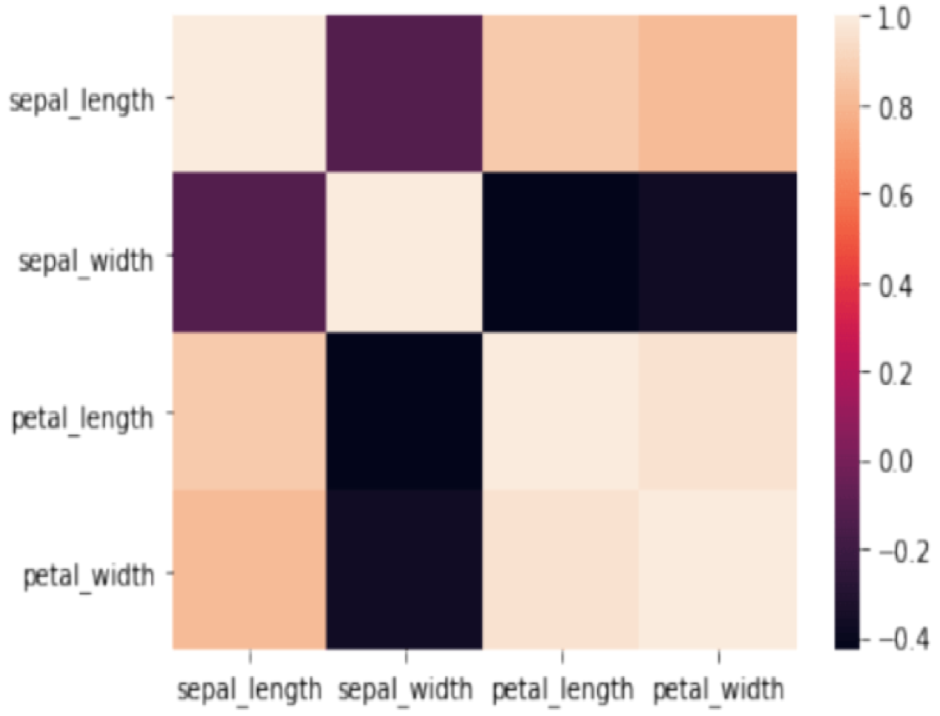
- It pairs all the continuous data and plots their correlation. It also plots the distribution of the data.

- If you do not wish to pair all the columns, you can pass in two more parameters x_vars and y_vars.

- HeatMap:

    - A heat map can be used to visualize confusion, matrices, and correlation.

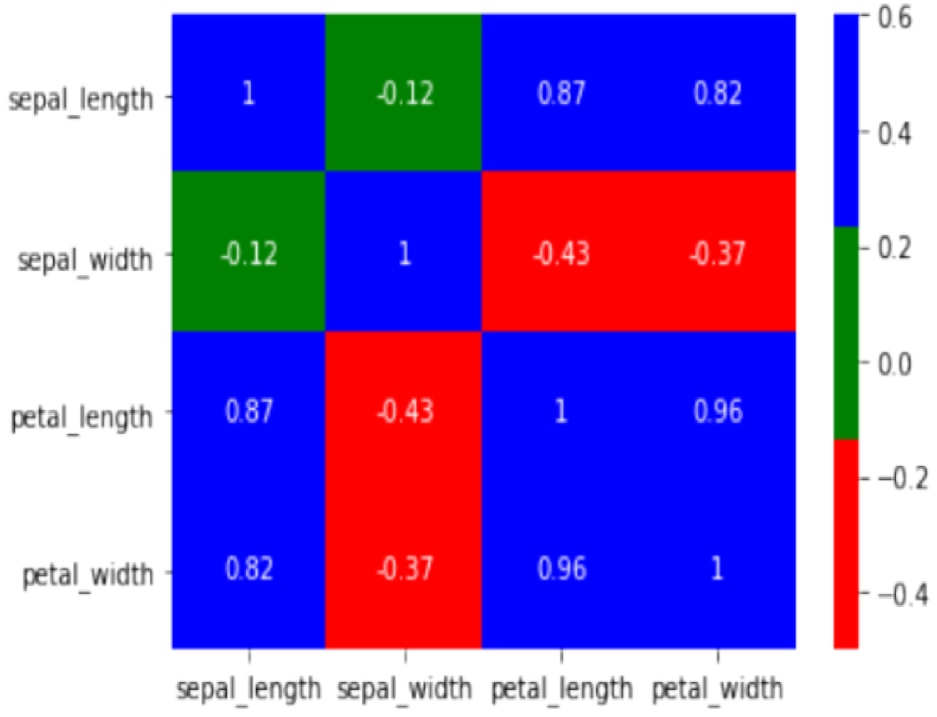
```
corr = df.corr()
sns.heatmap(corr)
```



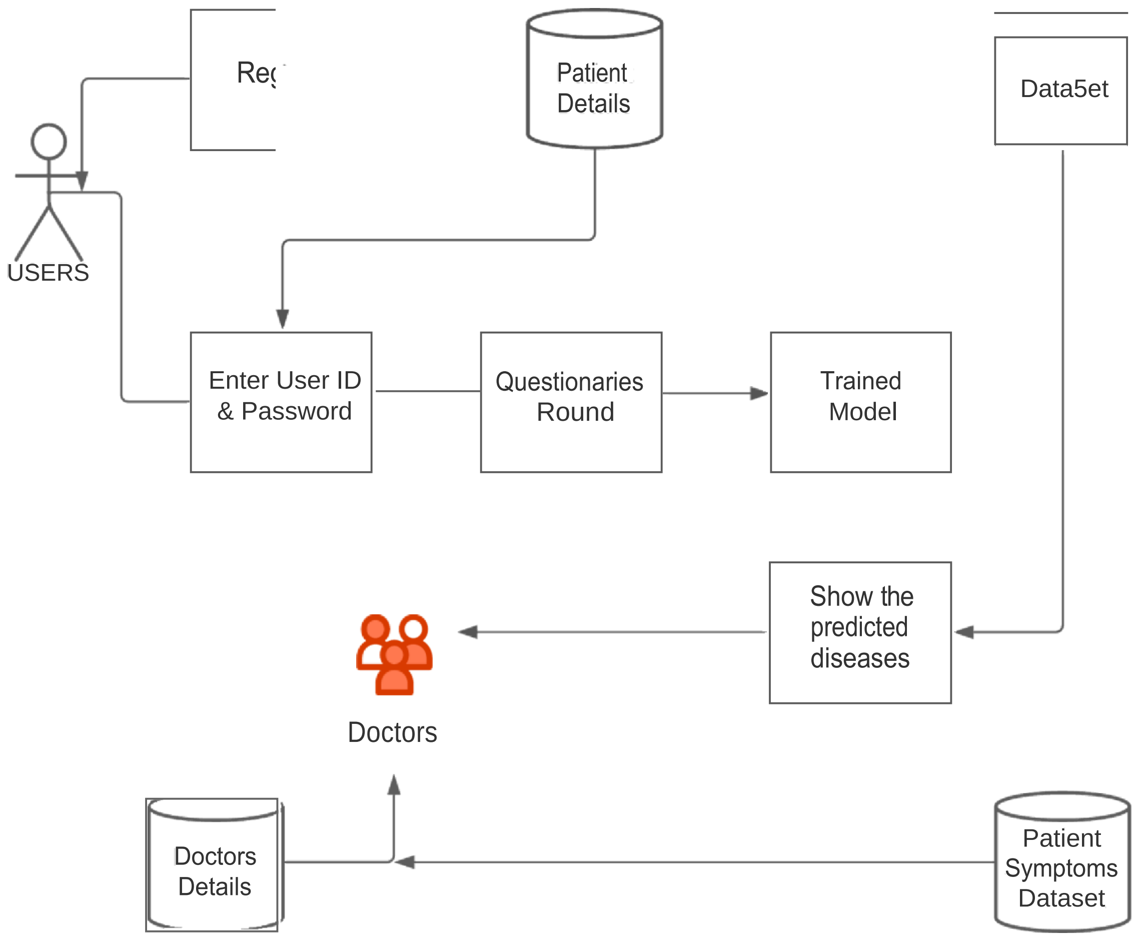
- We can customize the color scheme, the minimum/maximum values, and annotations.

- Based on the heatmap we can conclude that sepal length has a high positive correlation with petal length and petal width while sepal width has negative correlation with petal length and petal width.

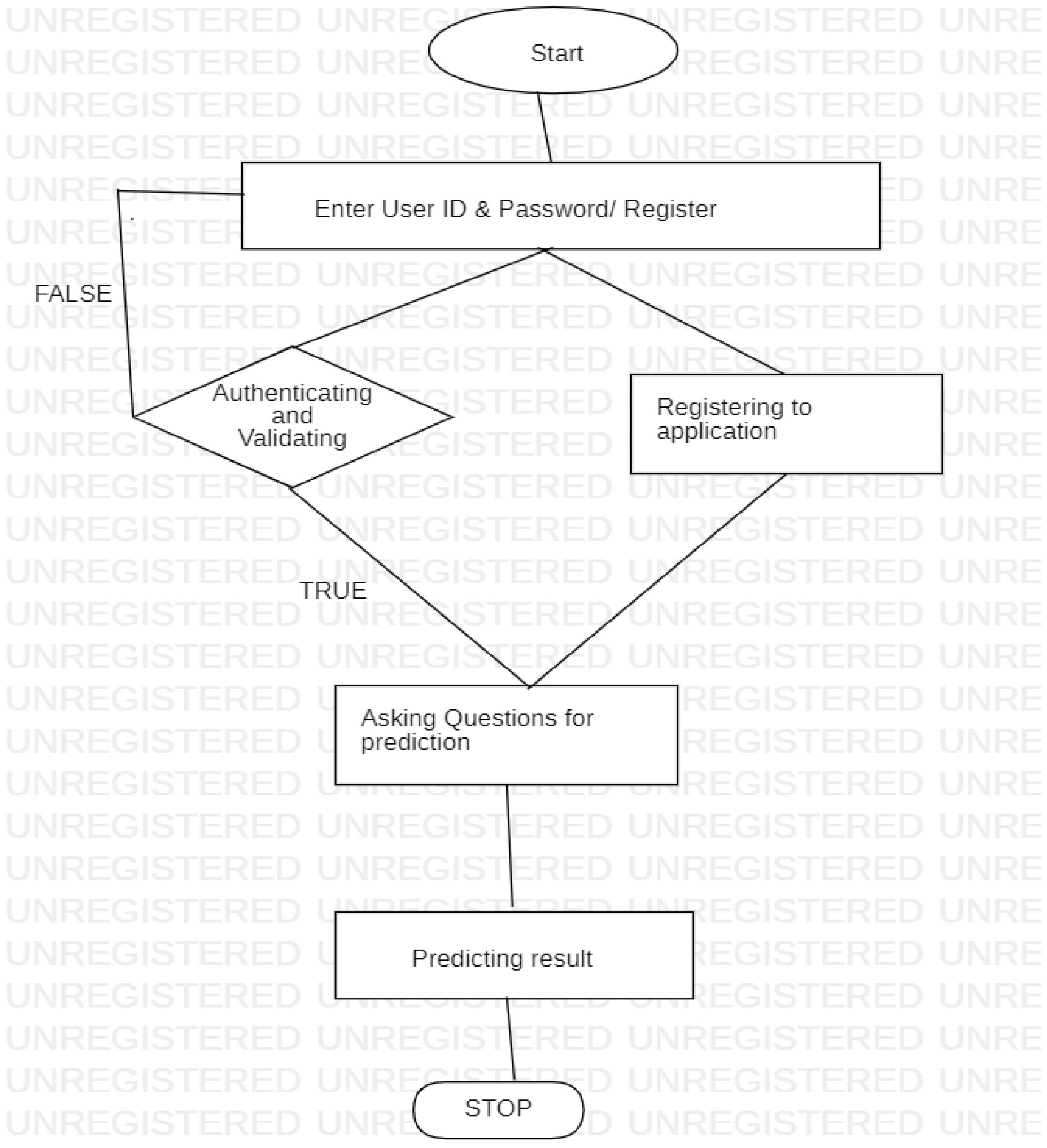
```
sns.heatmap(corr, cmap=['red','green','blue'],
vmin = -.5 , vmax = 0.6,annot = True)
```



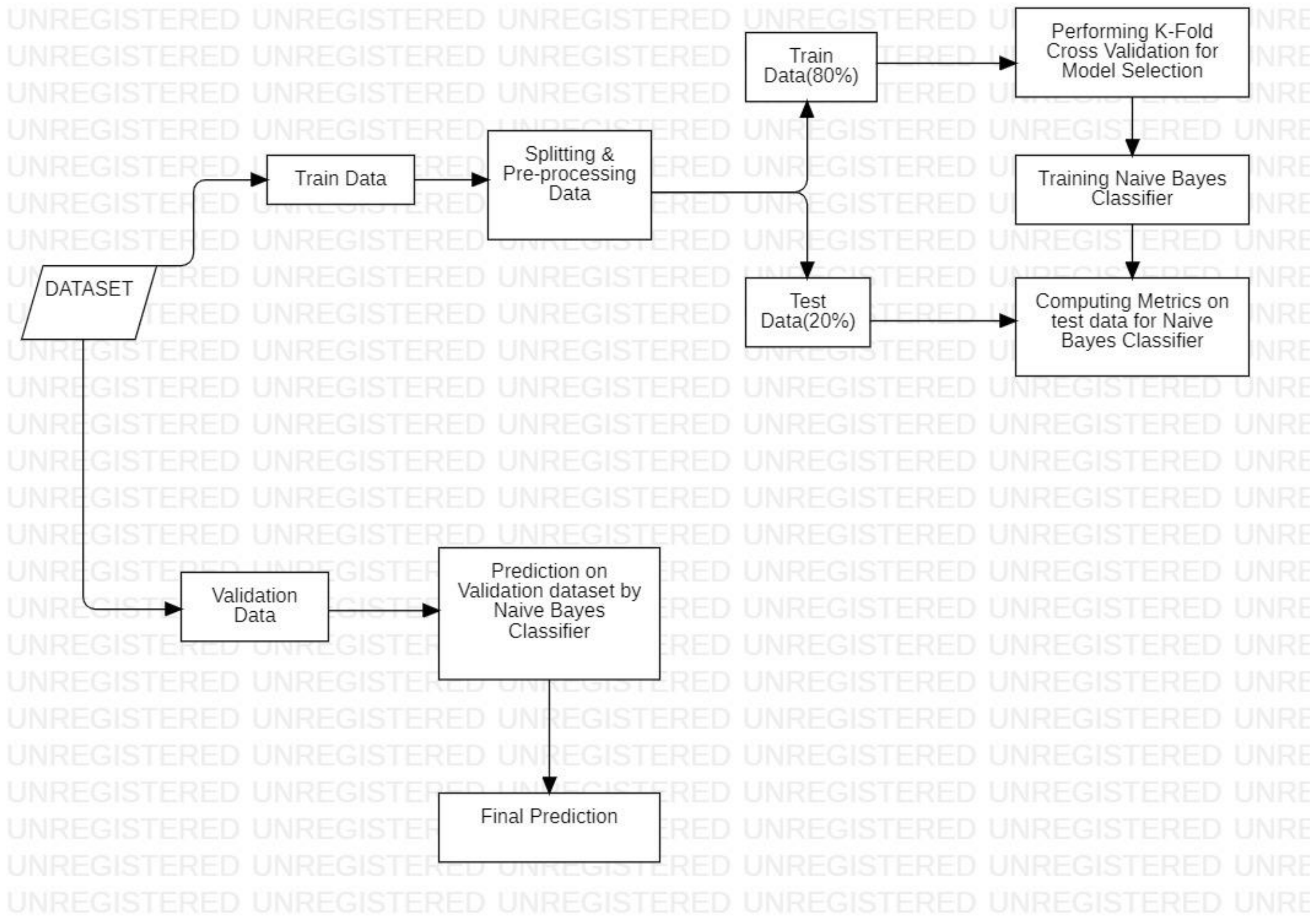
PROJECT DESIGN



Application Architecture


Start
Enter User ID & Password/ Register
FALSE
Authenticating and Validating
Registering to application
TRUE
Asking Questions for prediction
Predicting result
STOP

Application FlowChart Diagram


DATASET
Train Data
Splitting & Pre-processing Data
Train Data(80%)
Performing K-Fold Cross Validation for Model Selection
Training Naive Bayes Classifier
Test Data(20%)
Computing Metrics on test data for Naive Bayes Classifier
Validation Data
Prediction on Validation dataset by Naive Bayes Classifier
Final Prediction

WorkFlow (USE-CASE) Diagram

GUI SCREENSHOTS

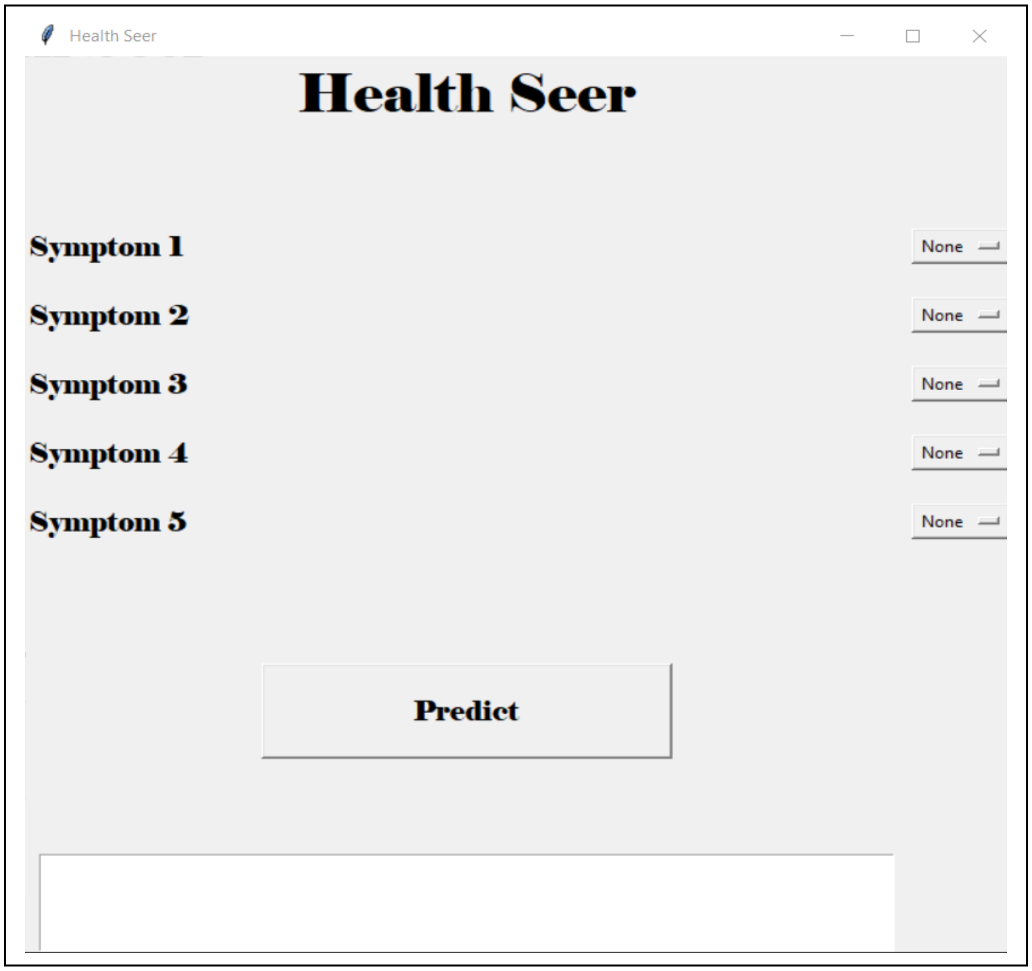
- First of all, let focus on the point i.e., What is GUI?
- GUI stands for Graphical User Interface.
- A graphics-based operating system interface that uses icons, menus and a mouse (to click on the icon or pull down the menus) to manage interaction with the system.
- Developed by Xerox, the GUI was popularized by the Apple Macintosh in the 1980s.
- At the time, Microsoft's operating system, MS-DOS, required the user to type specific commands, but the company's GUI, Microsoft Windows, is now the dominant user interface for personal computers (PCs).
- A comprehensive GUI environment includes four components: a graphics library, a user interface toolkit, a user interface style guide and consistent applications.
- The graphics library provides a high-level graphics programming interface.
- The user interface toolkit, built on top of the graphics library, provides application programs with mechanisms for creating and managing the dialogue elements of the windows, icons, menus, pointers and scroll bars (WIMPS) interface.
- The user interface style guide specifies how applications should employ the dialogue elements to present a consistent, easy-to-use environment (i.e., "look and feel") to the user.
- Application program conformance with a single user interface style is the primary determinant of ease of learning and use, and thus, of application effectiveness and user productivity.
- Elements of the GUI are:

    - Button - A graphical representation of a button that performs an action in a program when pressed.
    - Dialog box - A type of window that displays additional information, and asks a user for input.

- Icon - Small graphical representation of a program, feature, or file.
- Menu - List of commands or choices offered to the user through the menu bar.
- Menu bar - Thin, horizontal bar containing the labels of menus.
- Ribbon - Replacement for the file menu and toolbar that groups programs activities together.
- Tab - Clickable area at the top of a window that shows another page or area.
- Toolbar - Row of buttons, often near the top of an application window, that controls software functions.
- Window - Rectangular section of the computer's display that shows the program currently being used.

- How GUI Works?

  - On the basis of our application screenshots, We are giving you the brief procedure of the working.
  - In First Screen, you will see the User Interface of our application. In other words, you can say that how our application looks?



In this picture, It will show the structure of our application or you can say that the UI of the application.

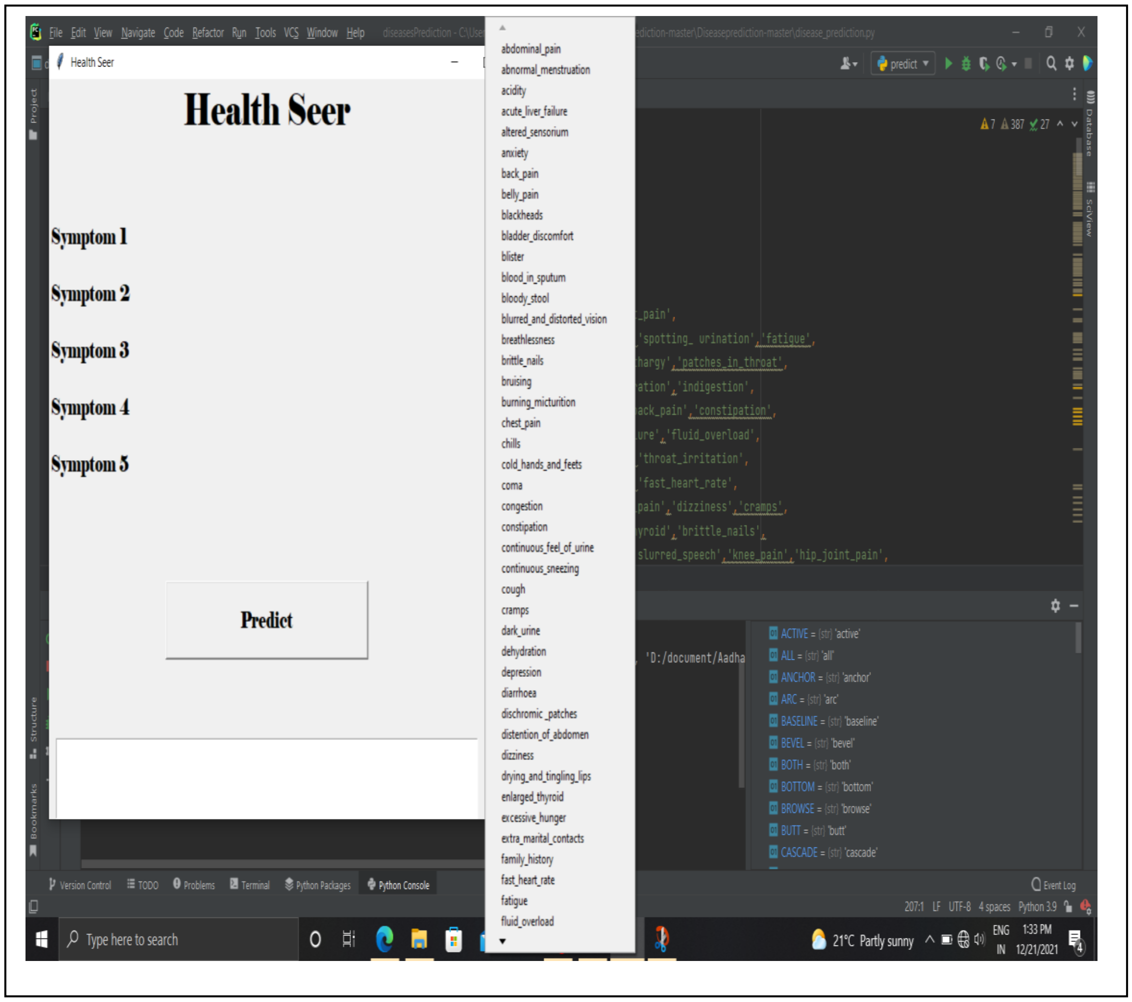
The title of our application will show at the center of the UI.

There are five Symptoms with the option menu section that are designed in a way of row and column.

In each symptom's Option Menu Section, we provide all symptoms in it. From where, Users can select any of them.

This process will show in the next figure.

- In Second you will see the working of Option Menu.



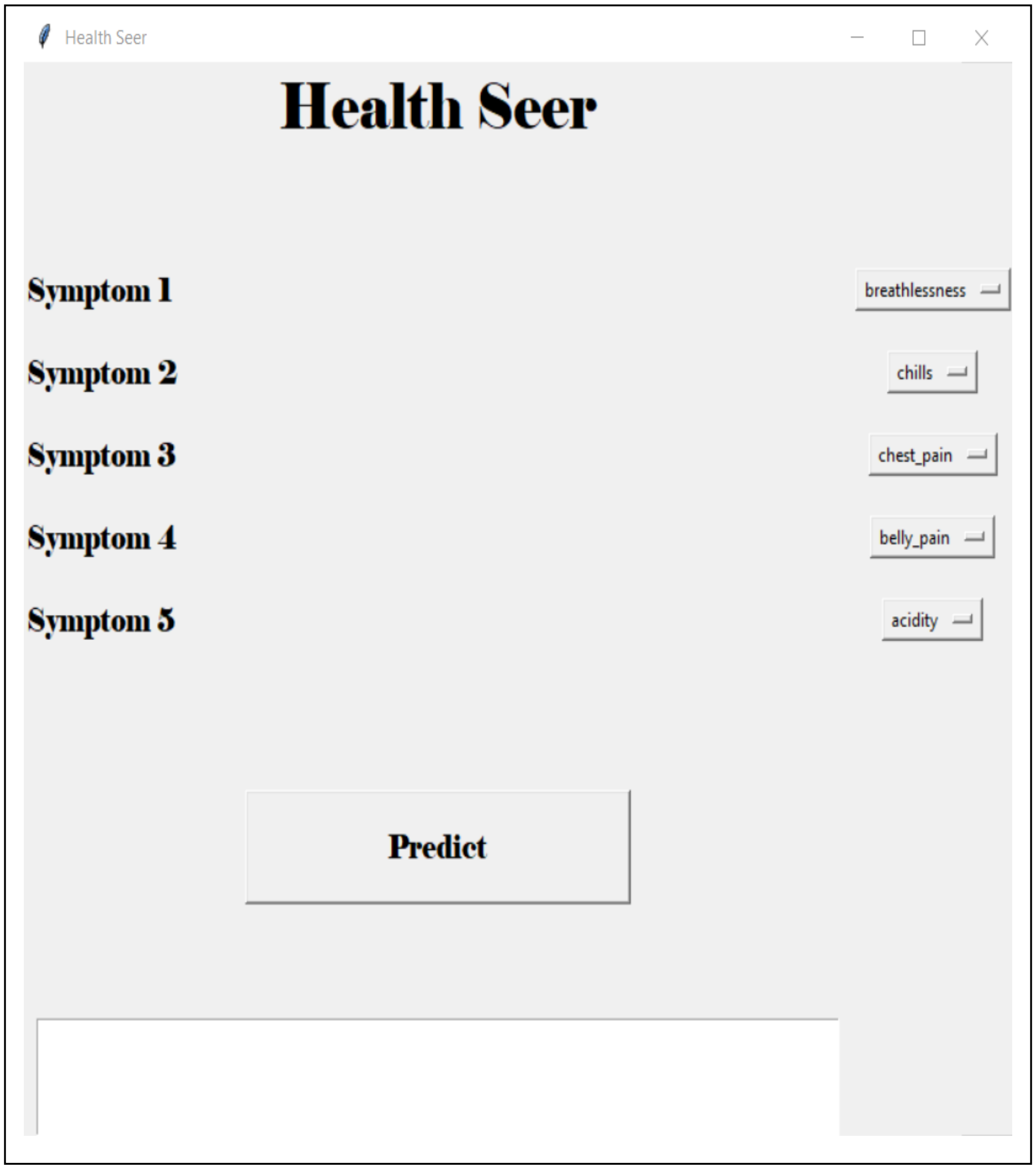
```
_pain',
'spotting_ urination','fatigue',
hargy','patches_in_throat',
ation','indigestion',
ack_pain','constipation',
ure','fluid_overload',
'throat_irritation',
'fast_heart_rate',
pain','dizziness','cramps',
yroid','brittle_nails',
slurred_speech','knee_pain','hip_joint_pain',
```

```
'D:/document/Aadha
ACTIVE = {str} 'active'
ALL = {str} 'all'
ANCHOR = {str} 'anchor'
ARC = {str} 'arc'
BASELINE = {str} 'baseline'
BEVEL = {str} 'bevel'
BOTH = {str} 'both'
BOTTOM = {str} 'bottom'
BROWSE = {str} 'browse'
BUTT = {str} 'butt'
CASCADE = {str} 'cascade'
```

In this screen, It will show the options of all symptoms that are placed on the option menu section.

In each and every Symptom's sections, there is option menu section is provided for selecting the symptom that user going through.

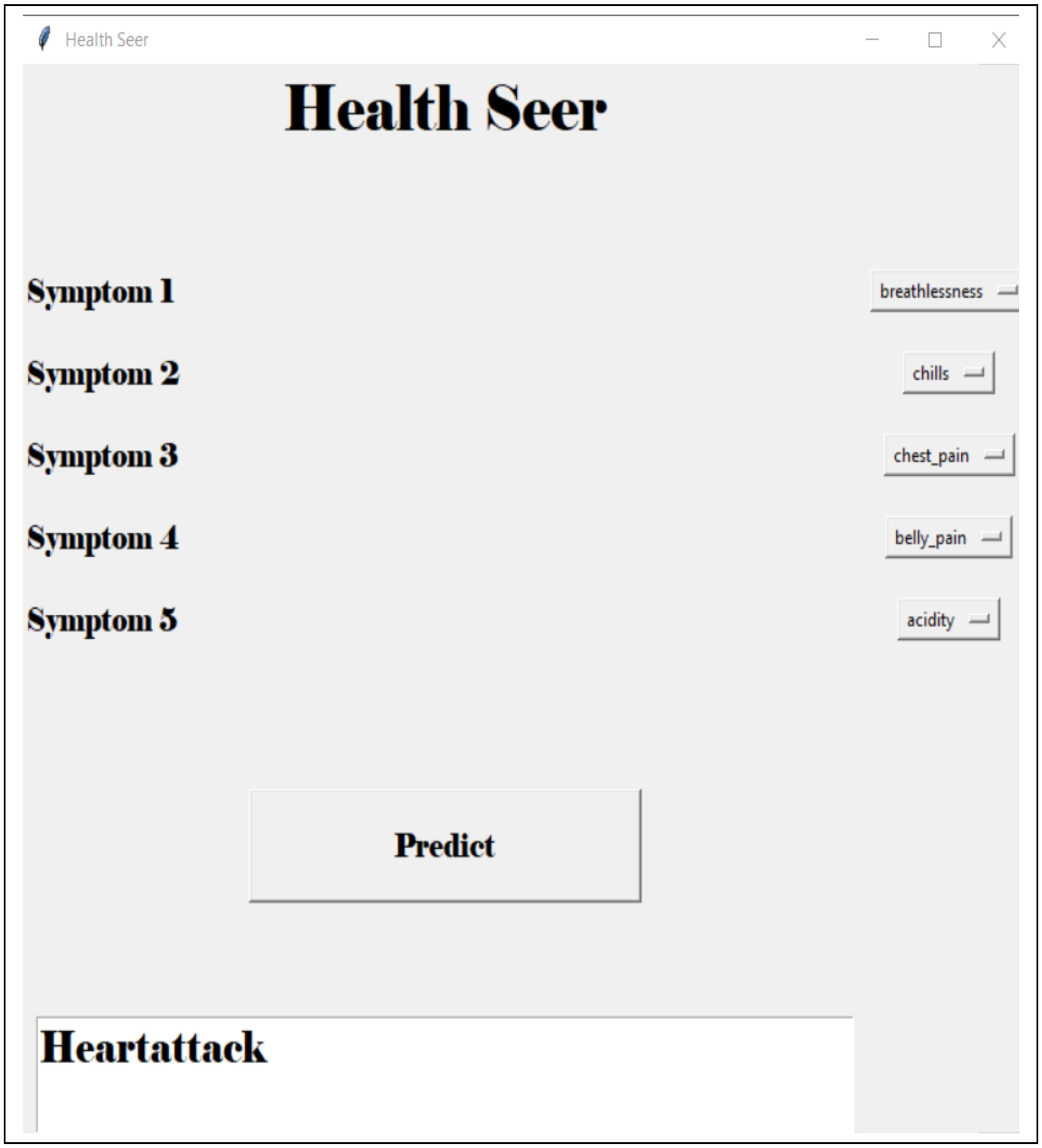
- In Third Screenshot, the selected symptoms shown.



In this picture, It shows the selected symptoms of the user.

From that selected Symptoms, the machine or model will predict the diseases that user being suffering through.

- In Fourth Screenshot, the predicted result will show in the message box section.



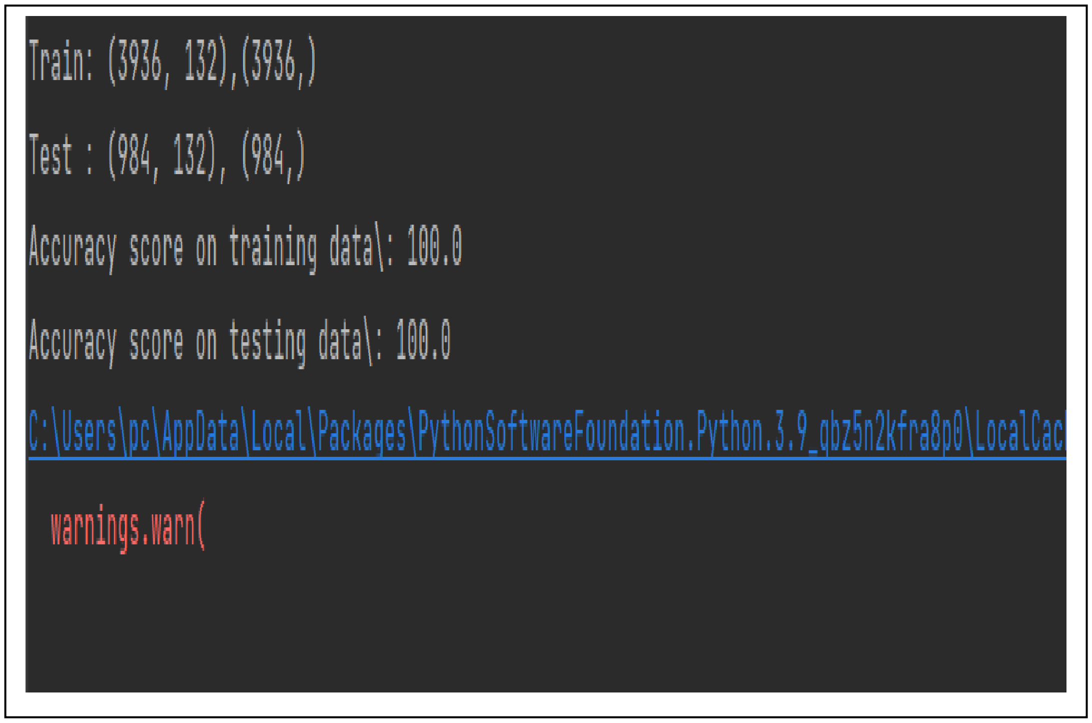
Heartattack

In this Screenshot, It shows the predicted result.

On the basis of selected symptoms, the model will train and test the data according to the dataset (which we provided).

After that, the preprocessing starts and data analyzation on the basis of the symptoms and then the model will predict the diseases the user is going suffering from.

- In Fifth Screenshot, it will show the training and testing data of the datasets.



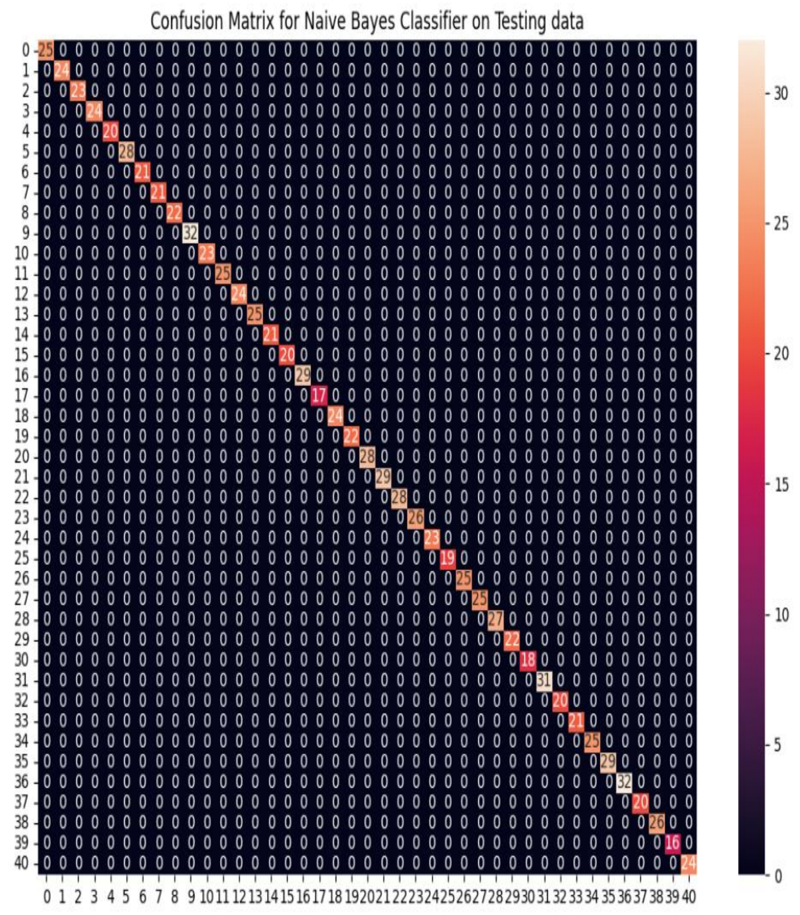
```
Train: (3936, 132),(3936,)

Test : (984, 132), (984,)

Accuracy score on training data\: 100.0

Accuracy score on testing data\: 100.0

C:\Users\pc\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2kfra8p0\LocalCac

  warnings.warn(
```

In this picture, you will see the training and test data row's and column's element.

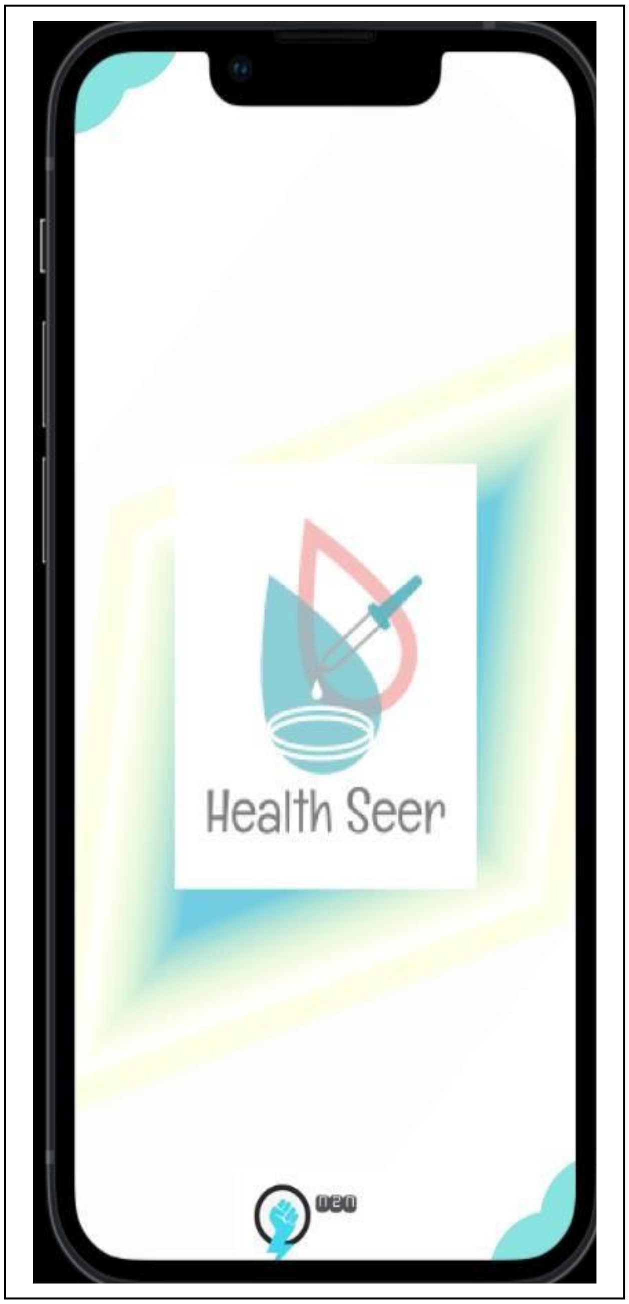
The Accuracy Score model used to predict the correct score according to the dataset and the symptoms that user selects.

The training and testing data of the symptoms and the diseases will check the accuracy score.

- In the Sixth Screenshot, it will show the confusion matrix on the basis of the testing data.

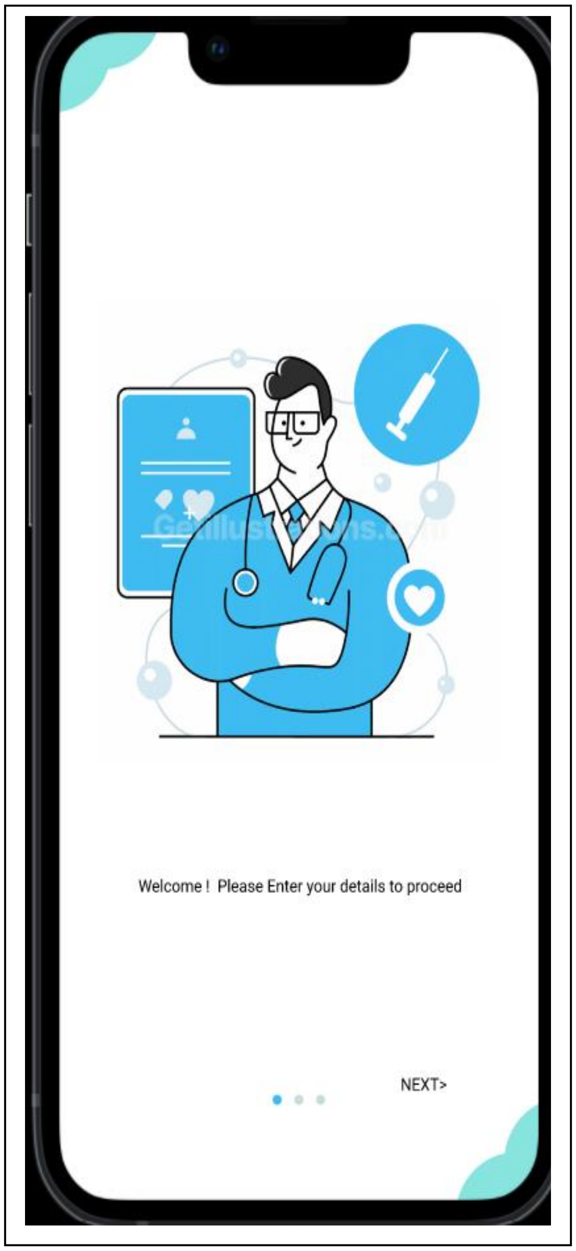


Future Implementation

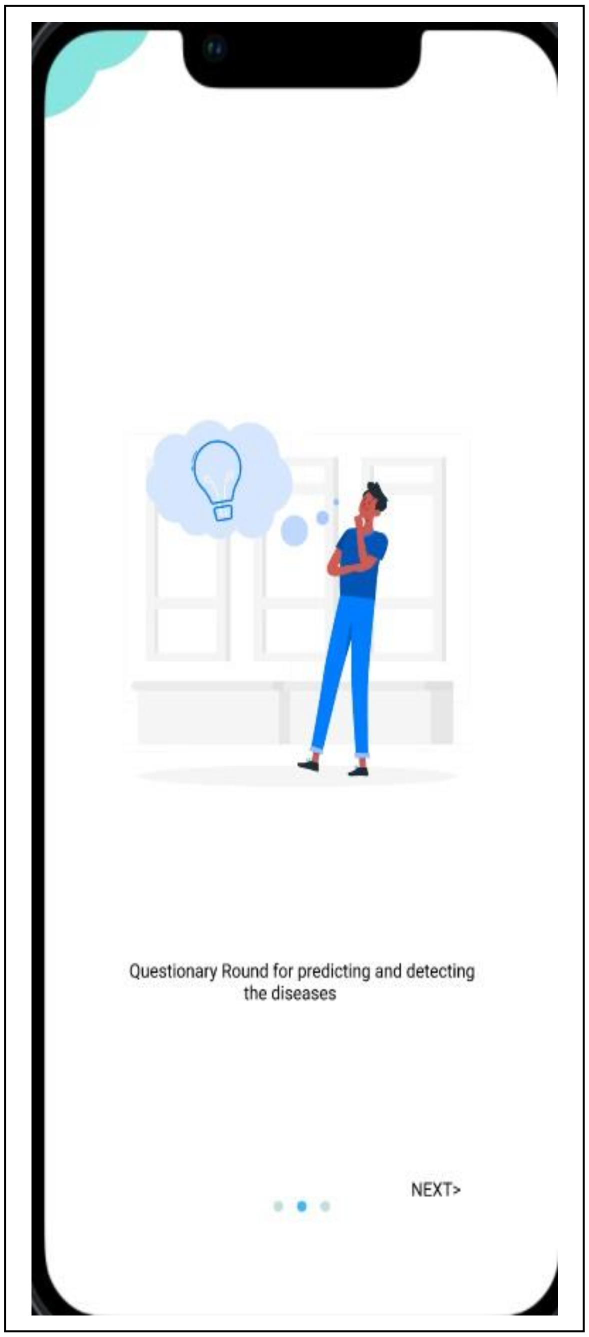


Splash Screen:

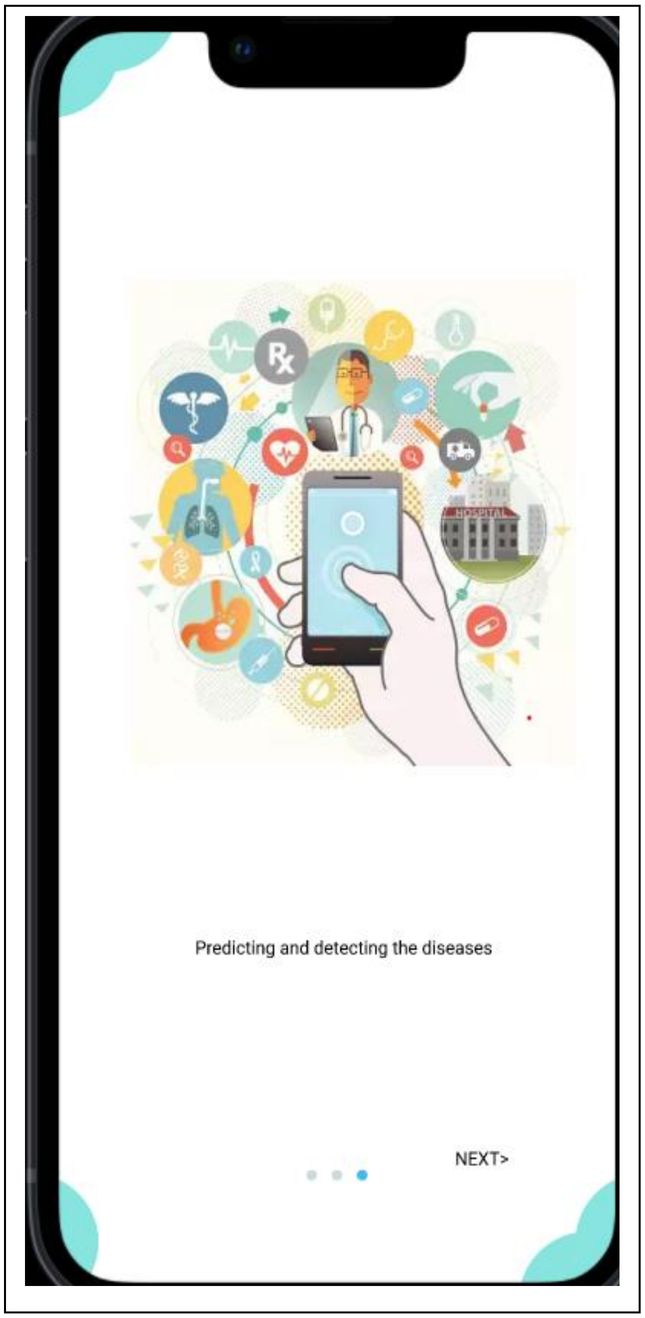
In this page, the application logo and the company logo is going to displayed on the screen.



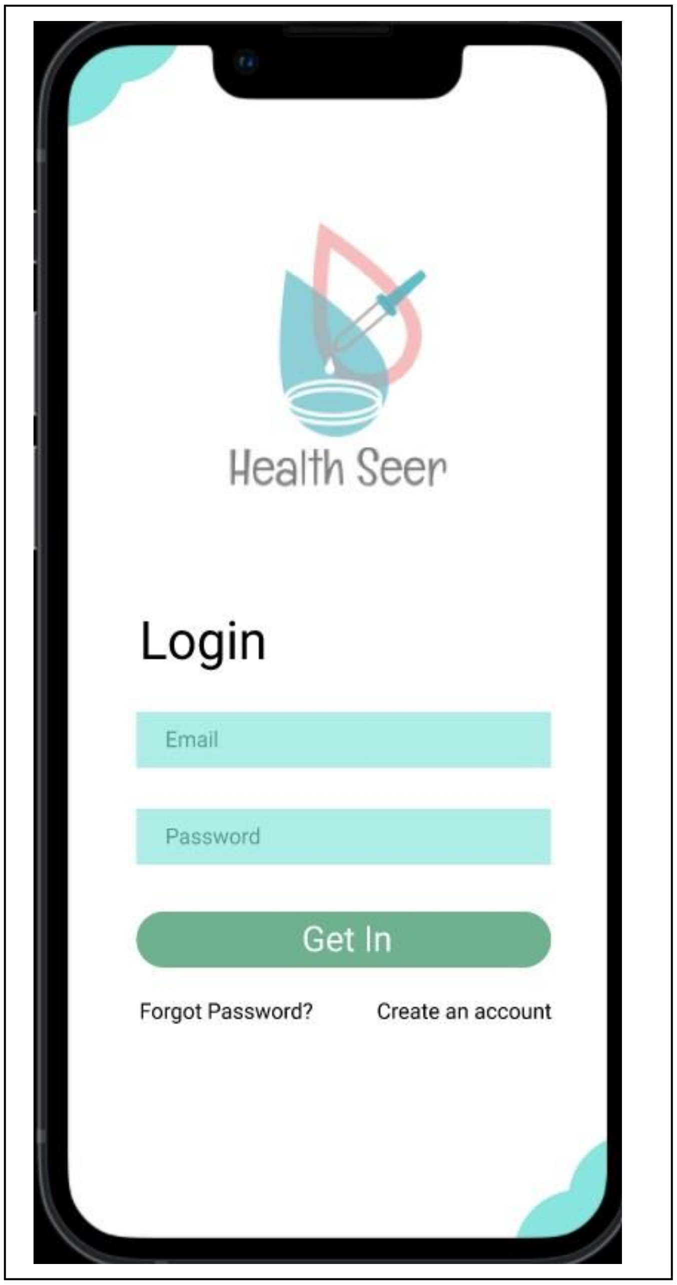
Onboarding Screen:
An onboarding screen is like a walkthrough, aimed to introduce what an app does to a user and of course how to use it.



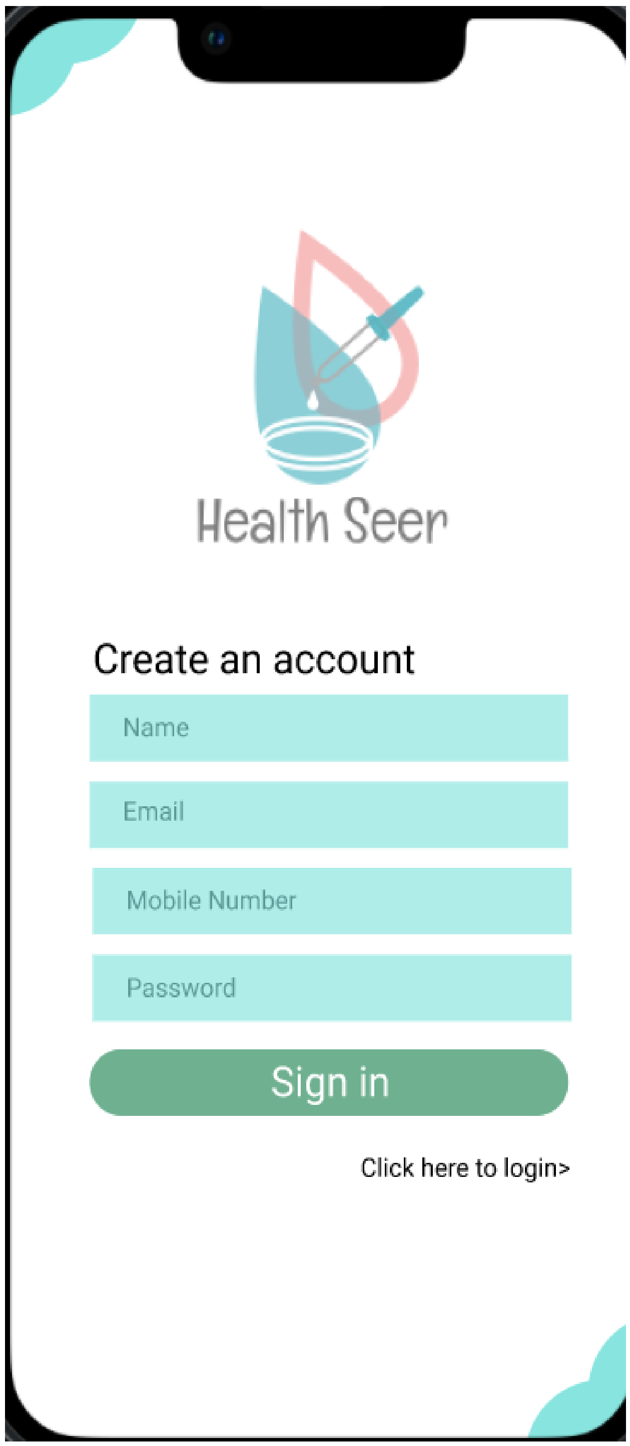
Onboarding Screen 2: In this screen, we illustrate the user regarding our questioner round.



Onboarding Screen 3:
In this screen, we illustrate the user regarding the prediction and detection.

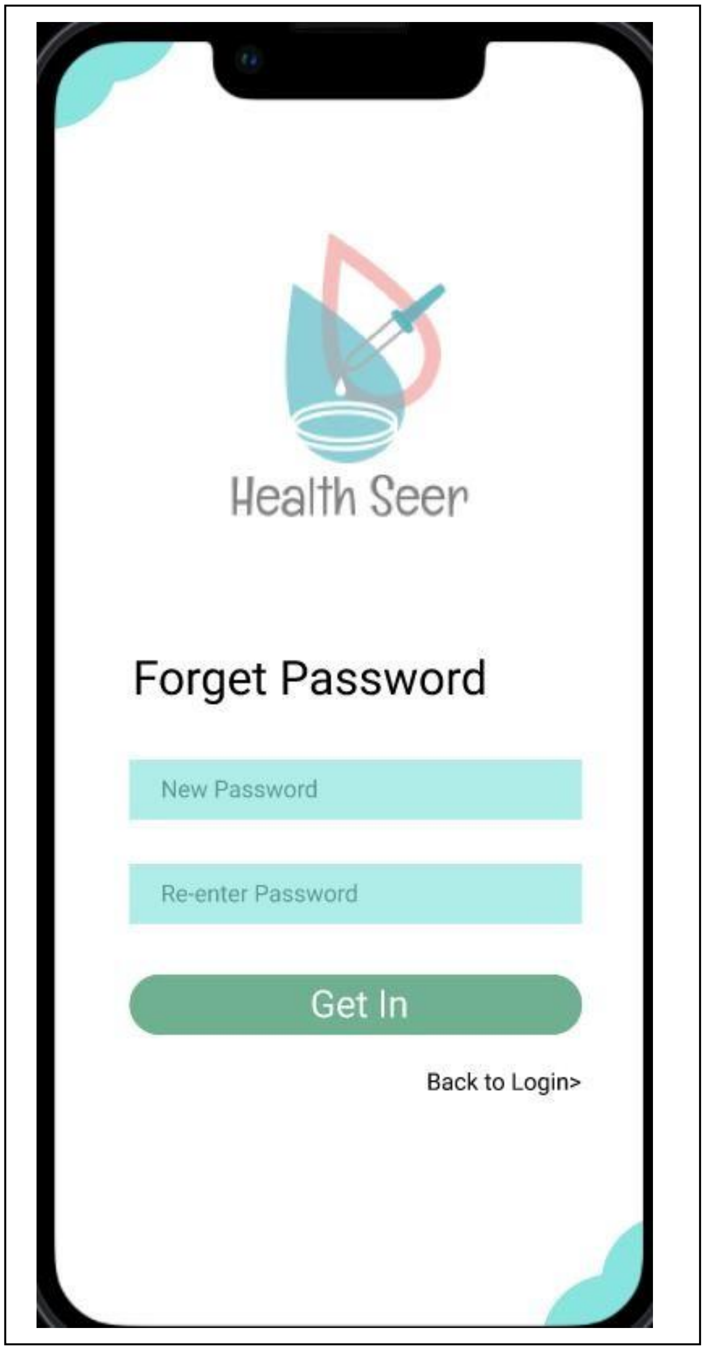


Login Screen:
In this screen, the user logged into the homepage of our health seer application.


Health Seer

Create an account

Sign In Screen:
In this screen, the users provide their details to enroll into this application.



Forget Password Screen: In this screen, whenever the user didn't remember the password. Then, this page is helpful to them.

CONCLUSION:

We were able to complete our project successfully with a high accuracy rate which lied beyond 90 percent on every run. We learnt a great deal about the importance of artificial intelligence and machine learning in the sector of healthcare. We also have been left with the future scope of our application which is good and quite interesting to us as we will be designing a mobile application with a more interactive and attractive user interface. The mobile app will have some added features such as a highly detailed report generation and will contain some suggestions based on the same report. Given the growth of machine learning and artificial intelligence in the healthcare systems, it is pretty much evident that with further development in the technology, it will be easier to carry out better predictions and reduce the error caused by humans and not to forget all the time that will be saved. People living in remote areas will benefit a lot with it as they will always have to travel for longer distances in order to get some basic check- up. We will also add doctor consultation option in further updates for benefit of such population. Dataset will also be increased based on new scientific updates and improvements in the model will be made time to time to enhance the performance of the training.