MEDICAL ASSISTANCE APPLICATION

Major Project Report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology in

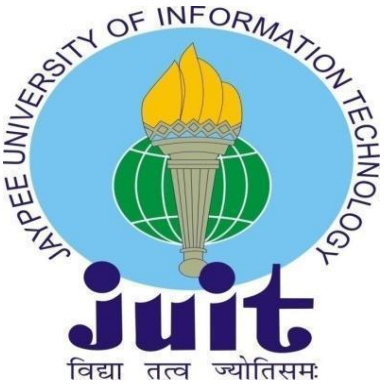
Computer Science and Engineering

By

Tarankit Chadha 181239

UNDER THE SUPERVISION OF

Dr. Jagpreet Sidhu



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, 173234, Himachal Pradesh, INDIA

CERTIFICATE

I hereby declare that the work presented in this report entitled **"Medical Assistance Application"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering and Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2022 to May 2022 under the supervision of **(Dr. Jagpreet Sidhu)** (Assistant Professor S.G of Department of Computer Science & Engineering/Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Tarankit Chadha
181239

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Jagpreet Sidhu
Assistant Professor S.G.
Department of Computer Science & Engineering and Information Technology
Dated:

ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to Almighty God for his divine blessing that makes it possible to complete the project work successfully.

I am grateful and wish my profound indebtedness to supervisor Dr. Jagpreet Sidhu, Assistant Professor S.G., Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of "Visualization, Machine Learning, and Neural Networking" helped me to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to Dr. Jagpreet Sidhu**,** Department of CSE, for his kind help to finish my project.

I would also generously welcome each of those individuals who have helped me straightforwardly or in a roundabout way to make this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Tarankit Chadha
181239

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Al & ML Based Ailment Divination is a system that anticipates disease-related facts or symptoms that are fed into the system and provides accurate findings based on that information. It works well when the patient isn't seriously unwell and the user merely has to know the type of ailment. It's an app that gives users guidance and techniques for staying on track with their health plans, as well as a way to diagnose disease via divination. Nowadays, the healthcare enterprise performs a critical function in treating patients' illnesses, so that is additionally a shape of help withinside the healthcare enterprise to inform the consumer and additionally beneficial to the consumer in case he does not wish to visit the medical institution or different clinics, so via way of means of including signs and all different beneficial information, they may be already in control, and the healthcare enterprise can benefit via way of means of virtually including this application into the traditional approach.

Chapter 01:- INTRODUCTION

1.1 Introduction

AI & ML Based Ailment Divination is a system that uses user information at the time of registration to forecast sickness. It also anticipates a patient's or user's disease based on the information or symptoms entered into the system and returns suitable findings. It was over if the patient was not very ill and the user only needed to know the illness kind. It's a tool that gives users advice and methods for keeping their health plan in check.

This is a valuable tool in the healthcare business to inform the user, and also helps the user if they do not want to travel to the hospital or any other clinic since the healthcare industry has played such an essential part in treating patients' ailments. This approach is beneficial to those who apply it. With the aid of Machine Learning Algorithms, Python Programming Languages, and previously available datasets in hospitals, we were successfully able to implement the project and get the required results.

Doctors nowadays utilize a variety of scientific technologies and methods to diagnose and treat not only common but also deadly disorders. Accurate and precise diagnosis is always required for effective therapy Physicians sometimes make mistakes while diagnosing patients therefore Ailment Divination systems that employ machine learning approaches can assist achieve the proper findings in these instances.

According to a study, the AI & ML Based Ailment Divination project was created to combat common illnesses in their early phases. As we all know, mankind has been so engrossed in the competitive arena of economic development that we have forgotten about our health. The primary reason for not knowing about one's health is the laziness of visiting a doctor and the worry of the time involved as they do not have time to schedule an appointment and see a doctor, which ultimately leads to a deadly disease.

1.2 Problem Statement

Now a day's there are a variety of issues in the health industry that are connected to the equipment or gadgets that will supply wrong or unaccepted results, so to avoid those results and get the correct and desired results we are building a program or project which will give the accurate Divinations based on information provided by the user and also based on the datasets that are available in that machine. The health industry is quite an information and knowledge poor and this industry is a very vast industry that has a lot of work to be done. So, with the help of all those algorithms, techniques, and methodologies we have done this project which will help the people who are in need. So the problem here is that many people go to hospitals or clinics to know how is their health and how much they are improving in the upcoming days, but they need to travel to get to know their responses, and sometimes patients may or may not get the results based on various factors such as doctor might be on leave or some whether problem so he might not have come to the hospital and many more reasons will be there so to avoid all those reasons and confusion we are making a project which will help all those people and all the patients who are in need to know the condition of their health, and at sometimes if the person has been observing few symptoms and he/she is not sure about the Ailment he/she is encountered with so this will lead to various Ailments in future, So, to avoid that and get to know the Ailment in early stages of the symptoms this Ailment Divination will help a lot of people ranging from children to teenagers to adults and also the senior citizens.

1.3 Objectives

The purpose of making this project called "Medical Assistance App" is to predict the accurate Ailment of the patient using all their general information and also the symptoms. Using this information, there we will compare with previous datasets of the patients and predict the Ailment of the patient he/she has been through. If this Divination is done at the early stages of the Ailment with the help of this project and all other necessary measures the Ailment can be cured and in general this Divination system can also be very useful in the health industry. If the healthcare sector approves this initiative, doctors' workload will be decreased, and they will be able to anticipate a patient's condition more readily. The basic goal of this illness categorization is to offer a diagnosis for numerous and recurrent disorders that, if left untreated or neglected, can develop into a lethal condition that causes tremendous harm to the patient and their family members. This system will predict the most possible Ailment based on the symptoms. The health industry information and knowledge are quite poor and this industry is a very vast industry that

has a lot of work to be done. So, with the help of all those algorithms, techniques, and methodologies we have done this project which will help the people who are in need.

1.4 Methodology

- This Project will predict the Ailments of the patients based on the symptoms and other general information using the datasets.
- This is achieved primarily based totally on the preceding datasets of the hospitals so after evaluation it is able to offer as much as 80% of correct results, and the mission continues to be growing in addition to getting the 100% accurate results.
- With the assistance of Med AI, it is able to expect the Ailment of the affected person and might remedy diverse issues and save you diverse aspects.
- It provides security for the system so that no one can break into it and no one can make any changes in the system.
- The user must enter the symptoms from the given menu, to induce correct accuracy, the user has to enter all the symptoms.
- Here we can easily prepare the data and transform that data into the algorithm, which will reduce the overall work of the project.
- It provides the necessary options to choose from the types and attributes.
- Here the user has to register first, to use the Divination, and then log in to the system using the credentials such as username and password.
- After opening the program to log in, the user must first register by selecting the register/register button.
- The user must then enter basic registration information, which is saved in the system.

1.5 Organization

- **Chapter 1:**Contains the Introduction, Problem statement, scope, and Objectives of the System or Project.
- **Chapter 2:** The literature survey discusses an abstract survey of the published papers and if any disadvantages are identified in the paper.
- **Chapter 3:** Discusses the detailed requirement of the problem identified for the major project, system architecture, and implementation details

- **Chapter 4:** Discusses the Performance Analysis of the model
- **Chapter 5:** Concludes the Report
- **Chapter 6:** Discusses any Future Scope

Chapter 02:- LITERATURE SURVEY

2.1 Survey

Machine learning, according to Tors Mitchell, is “a computer program that is claimed to learn from experience and other actions and to perform anything that builds knowledge as measured”. Many existing machine learning algorithms are focused on identifying and/or exploiting links across databases, as machine learning involves a combination of interaction and relationships. When Machine Learning Algorithms discover a particular combination, the model might utilize this connection to forecast future observations or data performance, resulting in intriguing patterns. In Machine Learning there are a variety of algorithms such as Naive Bayes Classifier, Bayes theorem, KNN (K-Nearest Neighbor Classifier), Decision Trees, Entropy, lD3, SVM (Support Vector Machines), Random Forest, and so on are some of the algorithms used in Machine Learning.

In 1959, Arthur Samuel created the phrase " learning machine." The study and development of algorithms that can interpret and separate data are what machine learning all exists of. lt has a strong connection to mathematical brilliance, which brings to the field methodologies, teaching, and application backgrounds. Machine learning is frequently confused with data mining, a subset of data mining that focuses on analytical data analysis and is referred to as uncontrolled reading.

Machine learning is a method used to create complicated models and algorithms that borrow from Bulela in the field of data mathematics; these are known as prediction predictions in a commercial application. By learning from past correlations and data styles, researchers, data scientists, engineers, and analysts may "create dependable, repeated judgments and outcomes" and discover "hidden insight".

2.2 Existing System

Traditional divination techniques and fashions include numerous chance variables, in addition to numerous algorithms, consisting of databases, programs, and plenty of more. High-chance and Low-chance affected person type is achieved primarily based totally on the assessments which are achieved in a group. But those fashions are most effectively treasured in medical conditions and now no longer withinside the huge enterprise sector. So, to encompass the Ailment Divinations in diverse health-associated industries, we've used the standards of device studying and supervised studying techniques to construct the Divinations system.

After researching and comparing all algorithms and machine learning theorems, we have concluded that algorithms such as Logistic Regression, KNN, Naive Bayes, Regression, and Random Forest Algorithm are important in building a system of - Ailment Divination that predicts patients sickness. After doing research and confirming findings, we concluded that it can predict an accuracy level of up to 90% after employing various approaches such as neural networks to diagnose disease. Patient statistical data, outcomes, and disease history are all stored in, allowing us to find a data solution that lowers the cost of medical case studies. The current method can forecast the disease but not the subtype of the disease, and it can't predict the human state. The disease's diagnosis is unclear and unexplained.

2.3 Proposed System

The proposed Al & ML Based Ailment Divination system would employ a variety of methodologies, algorithms, and other tools to develop a patient-centered diagnostic system based on symptoms and compare those symptoms to previously existing system data. We can estimate the correct percentage of the patient's sickness by combining those data sets and comparing them to the patient's condition. The divination model of a system in which the data is pre-processed for future reference and the user selects the feature where he will put the different symbols includes a database and symbols. Following that, the data was separated using different algorithms and techniques such as Logistic Regression, KNN, Naive Bayes, Random Forest, and others. The data is then sent to the reconsideration model, which displays the system's risk

analysis and also provides visible systemic estimates in such a way that it reflects various possibilities, such as how the system behaves when a large number of divorces are filed and makes patient reconsiderations based on the outcome. For the entire risk analysis necessary to execute Disease Prediction, we have integrated the whole structure and informal data from here. We can detect chronic illnesses in a certain location and community through systematic analysis. With the aid of algorithms and methods, we pick characteristics automatically in informal analysis. This system recognizes the user's symptoms and predicts the disease based on the symptoms and previous databases, assisting in ongoing diagnostic tests for viral infections, heart rate, blood pressure, diabetes, and a variety of other internal and external symptoms, predicting appropriate and specific illnesses.

2.4 Feasibility Study

Any key step in the software development process has been acquired. Allows developers to receive a working product that has been tested. Refers to product research that might be done in terms of product outcomes, application performance, and technical assistance needed to use it. A possible investigation should be carried out based on a variety of circumstances and conditions.

2.4.1 Economic Feasibility

Economic recovery is the difference between the advantages or results we get from a product and the overall cost we spend to enhance it. The creation of a new product enhances system correctness and speeds up application and reporting processing in the present system.

2.4.2 Probability Feasibility

The performance of a product to make it work is referred to as availability. Some items may perform admirably during manufacture and usage, yet they may malfunction in real life. lt entails researching the needed personalities as well as their technical knowledge. The contained data, updated information, and reports for generations are accurate and quick in the present system

2.4.3 Technical Feasibility

The term "technical performance" relates to whether or not the software currently available on the market is capable of completely supporting the present system.

lt investigates the benefits and drawbacks of utilizing specific development software, as well as its viability. lt also learns how much more time customers will need to make the app function. The user interface of the present system is usable and does not need a great deal of knowledge or training.

lt just takes a few mouse clicks to complete activities and generate reports. Because consumers want rapid access to websites with a high level of security, the software used to upgrade is best suited for current applications. This is accomplished by combining a web server and a data server in the same physical location.

Chapter 03:- SYSTEM DEVELOPMENT

3.1 Tools & Technologies Used

3.1.1 Python

Python is a robust programming language with a wide range of capabilities. Its broad features make working with targeted programs (including meta-programming and meta-objects) simple, and it fully supports object-oriented programming. Many additional paradigms, such as contract generation and logic programming, are supposed through extensions. Python takes advantage of power typing as well as the integration of reference computation and waste management waste collecting. lt also supports advanced word processing (late binding), which binds the way the words change during the process.

Patches to less essential sections of C Python that can give a minor improvement in performance at an obvious price are rejected by Python developers who try to prevent premature execution. When speed is crucial, the Python program developer can use mod-written modules in C languages or PyPy, a timely compiler, to submit time-sensitive jobs. Cython is a Python interpreter that converts Python scripts to C and invokes the Python interpreter directly from the C-1evel API. Python developers strive to make the language enjoyable to use. Python's architecture supports Lisp culture in terms of functionality. Filters, maps, and job reduction, as well as a list comprehension, dictionaries, sets, and generator expressions, are all included.

Two modules (itertools and functools) in the standard library use realistic Haskell and Standard ML tools.

3.1.2 Why Python?

We are utilizing the python language since Python can be utilized on the vast majority of the stages like Windows, Mac, Linux, Raspberry Pi, etc. Python is a stage-free language. Python is comparative and as straightforward as the English language. Python bolsters bunches of libraries

and it has a straightforward linguistic structure like English while java and C++ have complex codes. Python programs have fewer lines than some other programming languages. That is the reason we use the Python language for man-made consciousness, AI, Dealing with large amounts of information. Python is an article arranged language.

3.1.3 Django

Django is a sophisticated Python web framework that encourages quick development and ease of use. It was built by professional developers to handle a variety of Web development issues, allowing you to focus on building your app instead of reinventing the wheel. It's a free and open-source project.

- Django is designed to make it easier for developers to move projects from concept to completion.
- Django takes security very seriously and assists developers in avoiding many typical security blunders.
- Django's ability to grow rapidly and consistently is used by some of the busiest websites on the internet.

3.1.4 Joblib

In Python, Joblib is a set of utilities for delivering lightweight pipelines Disk caching for visual tasks and slow re-testing (memo mime pattern) are the main features of this basic computing system. Using joblib, create a file with a model submerged in saltwater Joblib is a cake substitute because it performs well on things with a huge numpy list. Instead of file names, these functions accept anything like a file.

Joblib is a Python operations pipeline resource that is part of the SciPy environment system. Well, for Python objects that employ NumPy data structures, it provides storing and loading services. Everyone uses tasks because they are the simplest extraction method. In Joblib, pipeline works (or works) are done with decorated works. The data model must be specified to specify logical compliance parameters. Joblib forgoes this and instead uses hashing to work longer and

harder.

3.1.5 Scikit-Learn

Scikit -learn is perhaps Python's most helpful machine learning package. The learning library includes several useful machine learning and mathematical modeling techniques, such as division, slowing, merging, and size reduction.

It's one of Scikit-most Learn's popular APIs. The Estimator API is used with all of Scikit-machine Learn's learning algorithms because it provides a uniform interface and a wide range of ML applications. The scale is the object read by the data (which contains data).

TensorFlow is a more advanced version of the standard library. Scikit-Learn is a cutting-edge package that allows you to create learning algorithms for a variety of machines thus you may construct an object in one or a few lines of code and use it to measure an Iron point or forecast a result.

3.1.6 Psycopg 2

Psycopg is a widely used PostgreSQL data adaptor for Python programming. The comprehensive usage of Python DB API 2.0 data and cable security are two of its key characteristics (several threads can share the same connections). It's made for multi-threaded programs that create and deploy many directories and a high number of inserts or UPDATEs.

Psycopg 2 is extensively used as a lib wrapper in C, resulting in improved speed and security. Because of the system's versatility, several Python types are supported out of the box and are meant to match PostgreSQL data types; familiarity may be expanded and modified. Unique and Python 3 are both supported by Psycopg 2. Psycopg2 is a Postgresql driver with DB API 2.0 that has been painting iron points. lt is built with multithreaded applications in mind and has its connection pool to manage.

3.1.7 HTML

Hypertext Markup Language (Html) stands for Hypertext Markup Language. HTML is a markup language used to specify the format of a document that will be displayed on a computer screen. HTML pages may contain audio, moving graphics, lists, genuine data, and java documents and can be generated as basic text or sophisticated multimedia. Web browsers, programs that may go across the network and show a range of information, display HTML pages. HTML is the most widely used web publication format. Allows the author to insert not just text but also text titles, lists, and tables, as well as still pictures, video, and audio in the text. Details can be retrieved from the student's computer. HTML sites may also be used to enter data and as a business transaction's front end.

3.1.8 CSS

HTML is used to show a document produced in the tag language. Cascading style sheets save time and effort by allowing you to change the appearance of numerous web pages at once. External style sheets are stacked one on top of the other. CSS style sheet files are used to specify the layouts, textures, and differences in the presentation of different devices and screen sizes on websites. External.css files are generally used to hold the style description. CSS is divided into three categories:

- Inline CSS is a type of CSS that is used in the body section and is related to an element called inline CSS. Style property is used to provide this sort of style within the HTML tag.
- Internal CSS (also known as featured CSS) is used when a single HTML document requires a unique style. The CSS rules must be given in the header section of the HTML file, which is CSS embedded within the HTML page.
- External CSS: This contains a separate CSS file that solely uses tag symbols. to hold style assets. The CSS assets are written in a separate file. A link tag must be used to link a CSS extension to an HTML text It indicates that the style may only be specified once for each item and that it will be applied to web pages

3.1.9 Javascript

JavaScript is a portable scripting language. JavaScript is a scripting language for building in-app apps. JavaScript is a platform that is open source. A tiny server connection is JavaScript. Because no such functionality is available. JavaScript cannot be utilized for communication. When the user submits the form, the JavaScript HTML code integration is completed, and all entries are enabled, they are posted to the Web server. Curly-bracket syntax, strong typing, and object-oriented orientation are all features. The web application's language is JavaScript. JavaScript is the most widely used programming language on the planet. Event-driven, functional, and essential programming paradigms are used in JavaScript.

3.1.10 Why Javascript?

We are utilizing Javascript language since Javascript can be utilized on the vast majority of the stages like Windows, Mac, Linux, Raspberry Pi, etc. There is a low threshold to get started. JavaScript mixes or works well with other languages and can be used in a wide range of applications. It comes with a no. of libraries which make it easy for users to work around. Server load is less using JS, as, On the client-side, the load on the website server is reduced. The basic idea in javascript is API routing, fast-paced growth, internal usage, growing numerous domains, ease of implementation, regulation, tools, etc. It holds the power to create rich interfaces.

3.1.11 Nodejs

The term "Node.js" does not relate to a specific file in this context; it is an extension of the normal JavaScript code file extension. Node.js is a cross-platform, open-source JavaScript environment that runs behind the V8 engine, allowing you to use JavaScript code without having to use a web browser. Node.js allows programmers to use JavaScript to construct command-line tools and server-side scripts that generate dynamic web page content before sending it to a user's browser. Node.js allows programmers to use JavaScript to construct command-line tools and server-side scripts that generate dynamic web page content before sending it to a user's browser.

Node.js allows programmers to use JavaScript to construct command-line tools and server-side scripts that generate dynamic web page content before sending it to a user's browser. As a result, Node.js represents the "JavaScript Everywhere paradigm, which encompasses web application development using a single programming language. multiple server-side languages, and client-side text using a non-blocking I/O model, making it an excellent choice for developing real-time data storage applications. Node.js is a fast and powerful programming language. It's an excellent choice for creating messaging or chat apps. It also aids in the development of large apps and e-commerce sites that rely on processing speed. Node.js is better suited for nonrestrictive, event-driven servers due to its single-threaded nature. Node.js is a programming language that is used for standard web pages and background API services. It is built for real-time, Push-based thinking. As a result, it's critical to communicate with members of the development team as well as stakeholders. In general, operational requirements describe how effectively a system performs under specific situations.

3.2 Hardware and Software Requirements

- Pentium 4. Intel Core i3, 5, i7. and 2 GHz processor RAM must be at 512MB.
- Hard disc with a capacity of at least 10 GB
- Input Keyboard and Mouse are the devices that are used.
- Monitor or PC as an output device
- Versions of Windows 7, 10, and above are supported.
- Jupyter Notebook as a platform
- Python Django (previous)
- Python, PostgreSQL. and Files as a Background
- Python is the programming language.

3.3 Machine Learning Algorithms

3.3.1 Logistic Regression

Logistic regression is a supervised learning classification approach for predicting a target variable's probability. The goal or variable quantity has a dichotomous character, which means there are only two possible classes. In plain English, the variable quantity is doubled, with

information coded as either 1 (represents success/yes) or 0 (represents disappointment/no). P(Y=1) is predicted as an element of X by a computed relapse model. It's one of the greatest ML calculations, which will be used for a variety of concerns such as spam detection, diabetes prediction, malignant growth detection, and so on.

Assumptions for Logistic Regression

Before we dive into the implementation of logistic regression, it's important to remember the following assumptions regarding the same:

- Because the target variables in binary logistic regression must always be binary, the desired outcome is represented by factor level 1.
- There should be no multicollinearity in the model, which means that the unbiased variables must be independent of one another.
- For logistic regression, we need to choose an excessive sample size.

Logistic Regression's Benefits

- On linearly separable datasets, logistic regression works well.
- While logistic regression is less prone to overfitting, it can still overfit in high-dimensional datasets. In these cases, regularisation (L1 and L2) approaches should be used to minimise over-fitting.
- Logistic regression not only provides a proportion of how important an indicator (coefficient size) is, but also the direction in which it is related (positive or negative).

3.3.2 K-Nearest Neighbours

This algorithm is one of the best Machine learning techniques. It's a lazy learning algorithm used for regression and classification. It classifies the objects using their “k” nearest neighbors. k-NN only considers the neighbors around the object, not the underlying data distribution. If k = 1, it assigns the unknown to the category of the closest neighbor. If k > 1, the classification is determined by a majority vote supported by the k nearest neighbor prediction result. Both for classification and regression, a helpful procedure will be wont to relegate weight to the commitments of the neighbors, all together that the closer neighbors offer more to the normal

than the more far off ones. As an example, a standard weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.

When KNN is used to group events, the result is determined by the class with the absolute best recurrence from the K-most similar events. In essence, each instance votes for his or her class, and the class with the highest votes is chosen as the winner of the prediction. If you're using K and have a large number of classes (e.g. 2), choosing a K value with an odd number to avoid a tie is a good idea. As a result, when you have an odd number of classes, select a decent number for K. Ties can be consistently broken by increasing K by one and looking at the class of the most comparable occurrence in the preparation dataset. Here are a few things to remember:

As we decrease the worth of K to 1, our predictions abate stable.

Conversely, as we increment the value of K, our expectations become more steady on account of larger part casting a ballot/averaging, and accordingly, bound to make more precise forecasts (up to a specific degree) In the end, we begin to observe an expanding number of mistakes. It's now we all know we've pushed the worth of K too far.

In circumstances where we are taking a larger part vote (in favor of model: picking the mode during a characterization issue) among marks, we normally make K an odd number to have a tiebreaker.

Advantages:

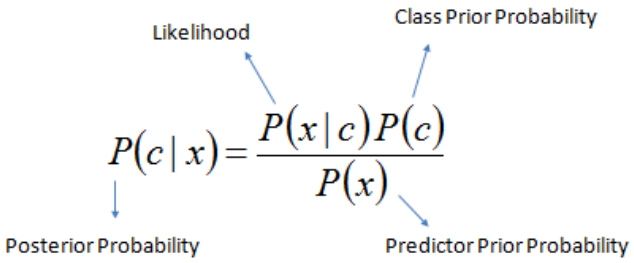
- The algorithm is simple and straightforward to implement.
- There's no need to create a model, tune a few parameters, or make more assumptions.
- The algorithm can be changed as needed. It has applications in categorization, regression, and search.

Disadvantage:

- The calculation gets essentially slower because the number of models as well as indicators/free factors increases.

3.3.3 Naive Bayes

Naive Bayes classifier is based on Bayes' theorem and is one of the oldest approaches for classification problems. The formula is:



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Fig 1 . Naive Bayes

The purpose is to determine the likelihood of an event. An occurrence with the letter B occurs. The naive Bayes classifier combines Bayes' model with decision rules such as the most likely result hypothesis. Naive Bayes methods are a collection of directed learning computations that rely on Bayes' hypothesis and the "naive" assumption of contingent freedom between each pair of highlights given the class variable's worth. It was created as a benchmark for text categorization tasks and is still used today.

Pros:

- Predicting the category of a test data set is simple and quick. It's also good at multi-class prediction.
- When the concept of independence is valid, a Naive Bayes classifier outperforms alternative models such as logistic regression, and you want less training data. When compared to a numerical variable, it performs well with categorical input variables (s). The typical distribution is assumed for numerical variables (bell curve, which could be a strong assumption).

Applications of Naive Bayes Algorithms

- **Real-time Prediction**: Naive Bayes is a fast and eager learning classifier. As a result, it can be utilized to make real-time forecasts.
- **Multi-class Prediction:** This algorithm is also well-known for its ability to predict multiple classes.
- **Message classification/Spam filtering/Sentiment analysis:** Naive Bayes classifiers are commonly utilized in message arrangement (due to increased prompts, multi-class concerns, and freedom rules)]. As a result, it's commonly used in Spam sifting (identifying spam email) and Sentiment Analysis (in web-based media examination, to spot positive and negative client feelings).
- **Recommendation System:** A Recommendation System combines the Naive Bayes Classifier and Collaborative Filtering to channel inconspicuous data and predict whether or not a client would actually enjoy a given item.

Building a basic model using Naive Bayes in Python

- Building a Naive Bayes model in Python is made easier with Scikit Learn (a Python module). The sci-kit learn package has three versions of the Naive Bayes model:
- Gaussian: it's a classification method that assumes features have a conventional distribution.
- Multinomial: It is a type of count that is used for discrete counts. Let's imagine we're dealing with a text classification issue. We can now examine Bernoulli trials, which are a step further than "word occurring within the record," and instead of "number of times result in number x_i is seen over the n preliminaries," you can think of it as "number of times result in number x_i is seen over the n preliminaries."

3.3.4 Random Forest

Random forest is a self-administered learning algorithm that can be used for classification and regression. Regardless, it is primarily used for classification. We know that a tree is made up of trees, and that having more trees means having more powerful woods. Similarly, a random backwoods calculation computation resolves on decision trees based on data testing, receives the

assumption from all of them, and then chooses the best course of action by projecting a polling form. It is a better technique than a single option tree since it reduces overfitting by averaging the results.

Working of Random Forest Algorithm

With the help of the stages below, we can learn how the Random Forest algorithm works:

- Begin by determining irregular cases from a given dataset.
- Following that, for each case, this calculation will generate a choice tree. Then it will get the desired outcome from each decision tree at that point.
- For each expected outcome, a polling form will be projected in this movement. Finally, as the last forecast outcome, choose the most casted ballot expectation result.

Working Example of Algorithms

We started with the Train Test split, but made the switch to the K Fold method because the Train Test split frequently contributes to the formation of models with high bias and the possibility of selecting test data with similar values, i.e. non-random values, resulting in an inaccurate assessment of model performance. We designed a function to execute numerous algorithms with different K values of KFold, such as Logistic Regression, Random Classifier, KNN, and Multinomial Naive Bayes.

```
from scipy.stats import chi2_contingency
```

```
chi2_contingency(pd.crosstab(df['cold_hands_and_feets'],df['weight_gain']))
```

```
(4362.40173527245,
 0.0,
 1,
 array([[4.69464146e+03, 1.11358537e+02],
        [1.11358537e+02, 2.64146341e+00]]))
```

```
x = df.drop(['prognosis'],axis =1)
y = df['prognosis']
```

```
from sklearn.model_selection import train_test_split
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)
```

```
from sklearn.naive_bayes import MultinomialNB
mnb = MultinomialNB()
mnb = mnb.fit(x_train, y_train)
from sklearn.metrics import accuracy_score
y_pred = mnb.predict(x_test)
accuracy_score(y_pred,y_test)
```

```
1.0
```

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(mnb, x_test, y_test, cv=3)
print (scores)
print (scores.mean())
```

```
[1. 1. 1.]
1.0
```

Fig 2. Train Test Split

```
from sklearn.ensemble import GradientBoostingClassifier
gbm = GradientBoostingClassifier()
from sklearn.linear_model import LogisticRegression
log = LogisticRegression()
mnb = MultinomialNB()
from sklearn.ensemble import RandomForestClassifier
ran = RandomForestClassifier(n_estimators =5)
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
```

Fig 3. Algorithms

```
def evaluate(train_data,kmax,algo):
    test_scores = {}
    train_scores = {}
    for i in range(2,kmax,2):
        kf = KFold(n_splits = i)
        sum_train = 0
        sum_test = 0
        data = df
        for train,test in kf.split(data):
            train_data = data.iloc[train,:]
            test_data = data.iloc[test,:]
            x_train = train_data.drop(["prognosis"],axis=1)
            y_train = train_data['prognosis']
            x_test = test_data.drop(["prognosis"],axis=1)
            y_test = test_data["prognosis"]
            algo_model = algo.fit(x_train,y_train)
            sum_train += algo_model.score(x_train,y_train)
            y_pred = algo_model.predict(x_test)
            sum_test += accuracy_score(y_test,y_pred)
        average_test = sum_test/i
        average_train = sum_train/i
        test_scores[i] = average_test
        train_scores[i] = average_train
        print("kvalue: ",i)
    return(train_scores,test_scores)
```

Fig 4. K Fold Method

3.4 Use Case Model

Figure 5 illustrates the Use Case diagram between the User to the Ailment divination model. Initially, the user enters the System by Signup and Signing in and input the Symptoms occurring so that they can predict the ailment. This Data is Stored in the database and processed for the applied algorithms of Machine Learning Such as KNN, Naive-Bayes, Logistic Regression, etc. and the result is shown through the Django Interface.

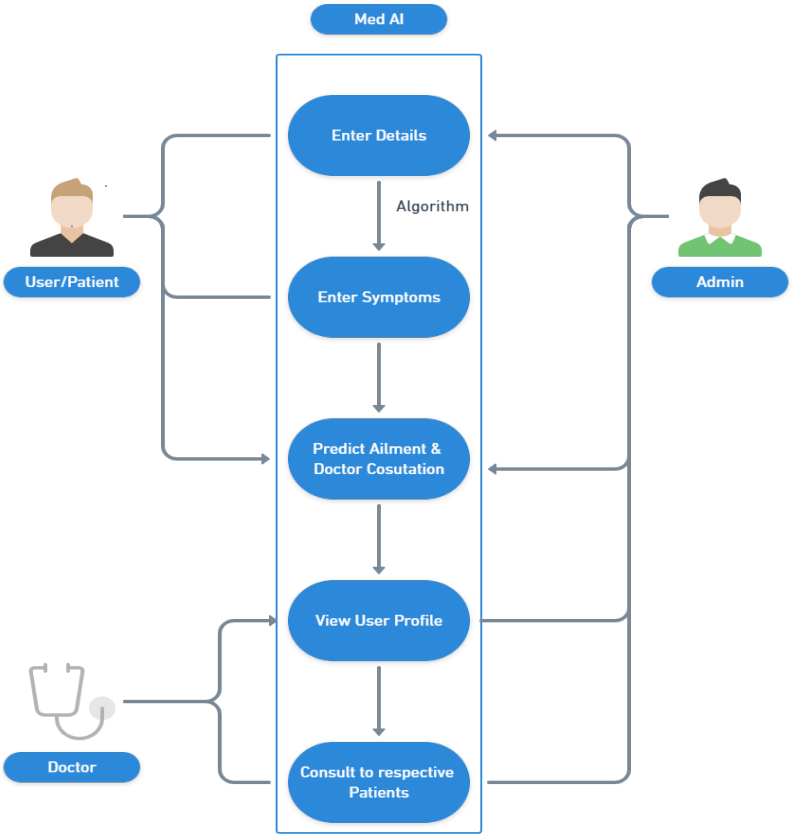


Fig 5 . Use Case Diagram.

3.5 Flow-Chain of Model

A disease project employing machine learning has all of the components necessary for a flow diagram. This data flow diagram illustrates how to transfer a model from one stage to the next, such as signing in and filling out all of the required information, as well as any other common features and symbols in the system. it's compared to a divination model, and if it's correct, it forecasts important outcomes; if not, it displays where the user went during data entering.

Figure 6 illustrates the Data flow diagram between the User to the Ailment divination model. Initially the user enters the System by Signup and Sign in and input the Symptoms occurring so that they can predict the ailment. This Data is Stored in the database and processed for the applied algorithms of Machine Learning Such as KNN, Naive-Bayes, Logistic Regression, MNB, etc. and Applying these algorithms ailment is divinated and shown through the Django

Interface using Flask.

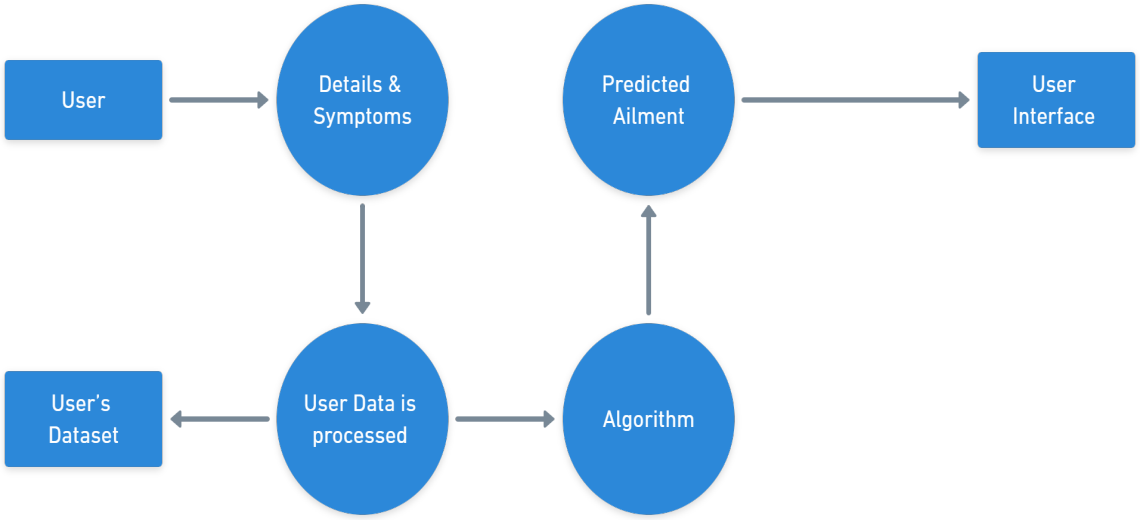


Fig 6. Data Flow Diagram

3.6 Modules

3.6.1 Admin

- An Admin(Administration) who manages and monitors the System Workflow.
- Admin is the one who actually Monitors and Manages the Website and Schedules the Appointments for the User to their respective doctors or specialists.
- Admin can view and modify (delete, update and Insert) all the Users or Patients who had their appointments with their respective doctors.
- Admin can view the names of the doctors of their respective fields and can search for the specialists of the particular field.
- Admin will approve the registration of the doctors based on their experience and knowledge by conducting offline/Online Consultations.
- Admin can approve/reject or delete the account of doctor request for the registration on the website.
- Admin can view the schedules and appointments of doctors with their patients.

3.6.2 Doctor

- ➢ Doctor Module is the one that takes care of their patients.
- ➢ Doctors can schedule the appointments for their patients and can update their profile and also they can consult their Patients regarding health issues in the following emergency.
- ➢ Doctors can view and edit their profiles.
- ➢ Doctors can view and Update the Schedules and Consulting hours for their patients.
- ➢ Doctors can Approve or cancel the requests of the Booking or appointments.
- ➢ Doctors will be notified on the arrival of the requests.
- ➢ Doctors will receive messages and chats from their patients and admins.

3.6.3 Patient

- ➢ The patient is the core module of the project. The major scope of this project is to automate the health services, consult and book appointments.
- ➢ Patients can view and edit their profiles.
- ➢ Patients Or Users can divine their ailment by inputting the symptoms that occurred.
- ➢ Patients can view the suggested doctors for their respective symptoms like Fever, Cough, Headache, Abdominal pain, Yellow-eye, Vomiting, Constipation, Loose-motion, Burning chest, Itching, and so on.
- ➢ Patients can view the suggested specialists, surgeons, and doctors in their respective fields like Cardiologists, and gynecologists. Orthopedic, Dentist, and so on.
- ➢ Patients can book their appointments and view the status of their appointments.
- ➢ Patients can consult their doctors via message during consultation hours.
- ➢ Patients can view the profile and details of their doctors.

3.7 Functional Requirements

Operating requirements are the features or services that product developers should employ to enable users to accomplish their jobs. As a result, it's critical to communicate with members of the team. In general, operational requirements describe how effectively a system performs under specific situations.

3.8 Non Functional Requirements

Inaccurate requirements refer to system flaws or limits. They can be linked to new system structures like dependability, reaction time, and storage stay, as well as platform, implementation techniques, and tools.

Non-Functional requirements might be determined on a variety of factors, including user demands, financial constraints, and corporate regulations.

3.9 User Interface Functionalities Requirements

A user interface specification (UI specification) is a document that details the user interface. There is generally some basic software that requires UI when it comes to case design and case management. The goal of developing use cases is for the designer UI to have a better knowledge of product features.

3.10 Login Functionality

The sign-in page allows registered users to log in to the site and have access to all of the capabilities that their account grants them. Users' information is validated when they put in their username and password and then click to submit, and they are logged in when they log in. If they are incorrect, an error notice will appear. If a user forgets his password, he or she should click "Forgot Password?" to get to the password recovery page. If the user does not already have an account, click the register button to take them to the checkout.

3.10.1 The Procedure for Registration

Adding Patients & Doctors: Patients and Doctors can register themselves on the website after inputting the required information asked on the signup page and after signing up they can sign in and visit their respective profiles.

3.10.2 Double-Check

All the information that is provided by the user, be it a Doctor or a Patient, their personal details and important information is stored in the database.

This information is safe and can only be accessed by the administrator. This information could be retrieved anytime from the database.

3.10.3 Generating Reports

Patient and Physician Information: The System provides data to each patient with information such as the patient's name, phone number,address, doctor's name, ward name, and other pertinent information.

3.10.4 Database

Authorized Patient and Physician Data: Each patient must provide specific information, including his or her phone number, surname, and last name, personal health number, postal code, nation, address, city, patient, and medical ID number. and so on. Update Patient and Physician Information: The system allows users to update patient information in accordance with mandatory criteria.

3.10.5 Login Page

The Login Page is a page that will be accessed by everyone who uses the site, including administrators, patients, and doctors. The sign-in page allows registered users to log in and use all of the services that their account entitles them to. If they enter their username and password and click to submit the verified user’s information, they will be logged in. If they are incorrect, an error notice will appear.

3.10.6 Consulting Doctor

Patients or Users can interact with their doctors via messaging them in the respective consultation hours and clarify their doubts and questions regarding their Ailment. Doctors will be notified when the Patients try to communicate with the doctors during the respective consultation hours. Doctors can view the reply to the Patients concerning their medicines, daily routines based on the symptoms like Fever, Cough, Headache, Abdominal pain, Yellow-eye, Vomiting, Constipation, Loose- motion, Burning chest, Itching, and so on.

3.10.7 Interface and framework

The Inner Interface of this system consist of Python‘s library interface called Tkinter. Then it goes into the framework model where all the actions and services are combined and then the result is processed. It also consists of a file system where all the user-related information is stored such as username, password, age, phone, and email. After the User Interface, it consists of the framework in which the system works accordingly using all the technologies, algorithms, and various tools in which the project works accordingly. The framework consists of all the modules starting from the data preparation, data building, and assessment stage.

All these three factors are then going into the data collection phase, where the data is classified accordingly using the appropriate models and algorithms such as Logistic Regression, naive Bayes, and random forest.

Then all those algorithms use the datasets and it forms the sets where all the previous data is stored, then using that data it compares with the new data, and the result is generated.

3.11 Detailed Design

We designed our system such that every time a user logs in, he or she must register, and a new user cannot use the system without first registering. After registering, the user will need to provide basic information such as their username, age, email address, phone number, and password. After that, the user must use the same username and password to log in to the system.

Here are some of the capabilities of this system:

- Inputting Symptoms: Once a consumer efficiently logs in to the device then he/she has to pick out the signs and symptoms from the given menu.
- Ailment Divination: The prediction model predicts the Ailment of a person he might have, based on the user entered symptoms.

3.12 Sequence Diagram

A Ailment divination challenge utilising system studying includes a chain diagram with all the factors expected in a typical collection diagram. This collection diagram depicts how the version progresses from one level to the next, just like the way you would construct a machine and consist of all of the details, other details, and not unusual place images. Unlike a divination version, this collection diagram resembles wherein the consumer made a mistake whilst coming into the statistics and shows the appropriate monogram. The chronology of all of the corporations is related to wherein the consumer opens the program.

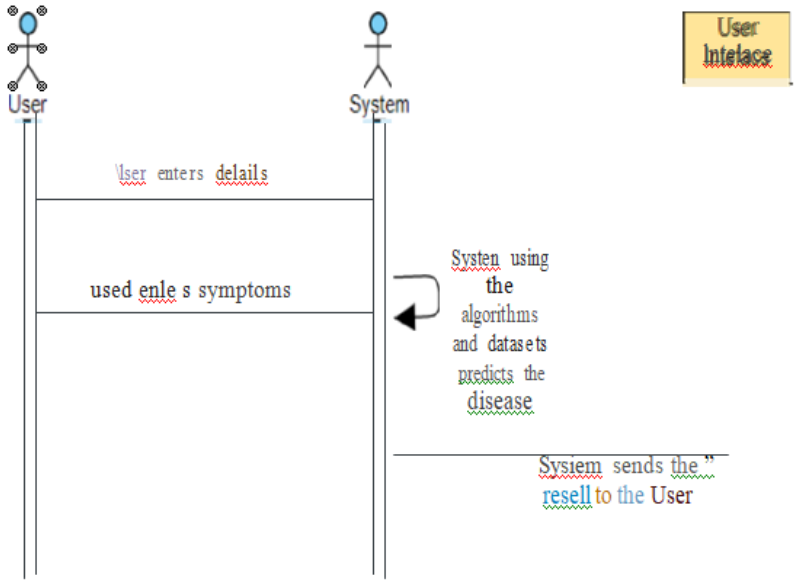


Fig 7. Sequence Diagram

3.13 Collaboration Diagram

These diagrams can be used to outline the operation of every object and to reveal how a certain use case works dynamically. The diagram depicts how all the fashions are related to show applicable outcomes from the consumer, which includes establishing the utility and the usage of the programme, registering, and storing the registration statistics withinside the report device, after which the usage of that consumer data to log into the device and provide all of the required data.

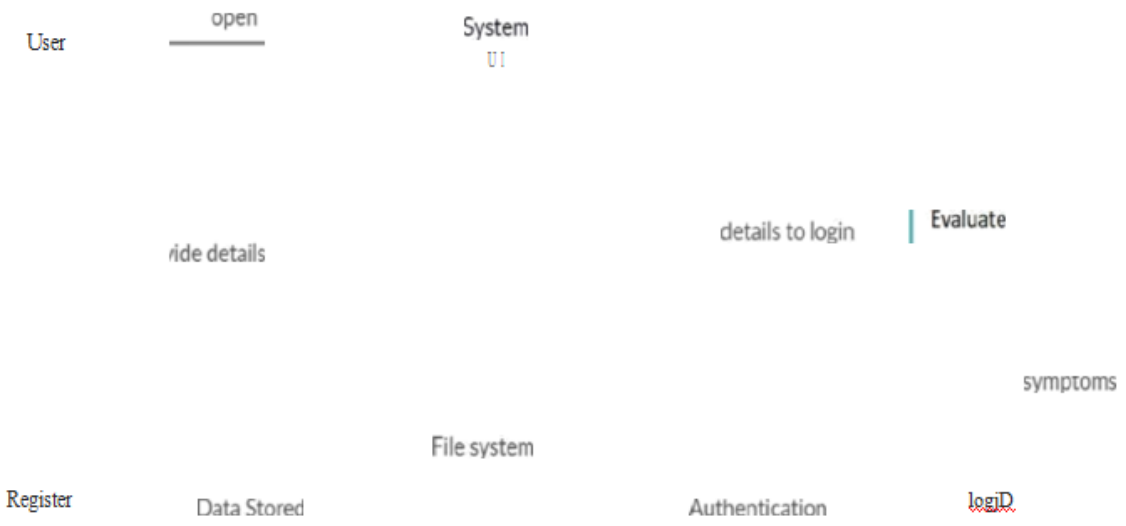


Fig 8. Collaboration Diagram

The Fig illustrates the Collaboration diagram among the User(Admin, Patient and Doctor) to the Ailment divination model. All the Users such as Admin, Patient and Doctor initially have to sign up their data to the database and Proceed for the Stability login the Authentication might be Provided through the Login and Evaluated within the Database. The Dataset and Algorithms are implied for incoming inputs so one can be saved in the Database i.e.PostgreSQL.

3.14 Activity Diagram

A job drawing is another essential diagram in the U ML for describing a program's dynamic characteristics. A work diagram is a flowchart that depicts the progression of work from one task to the next. A type of function is a system function. A control flow is shown from one action to the next. The user in this picture begins the process by registering with the system and logging in with their credentials. The symbols are compared to the system, and if they match, the user advances to the divination phase, where divination takes place. Only once the data from the data sets has been processed will the analysis begin, and only then will the appropriate findings be shown without selection.

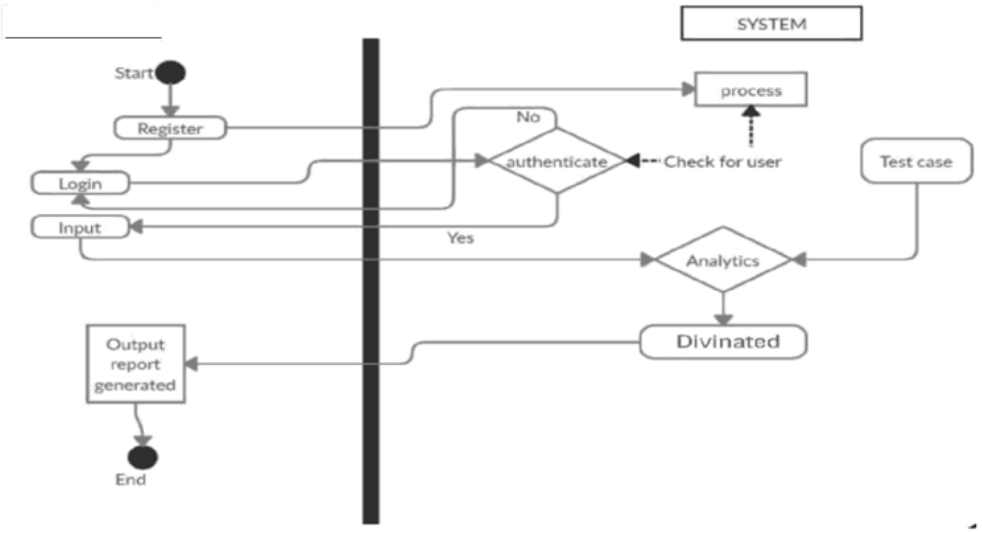


Fig 9. Activity Diagram

3.15 Class Diagram

Class drawing is the fundamental business required to continue the work in machine learning diagnosis. Every other software has basic classroom drawing, and class drawing is the basic business needed to continue the work in machine learning diagnosis. The class diagram contains information on all applicable categories and related data sets, as well as all other necessary attributes and relationships with other organisations, all of which is required for the use of divination, in which the user will enter all required information such as username, email, phone number, and many other attributes required to sign in to the system and use the concept of divination, in which the user will enter all required information such as username, email, phone number, and many other attributes required to sign in to the system and use the concept of divination

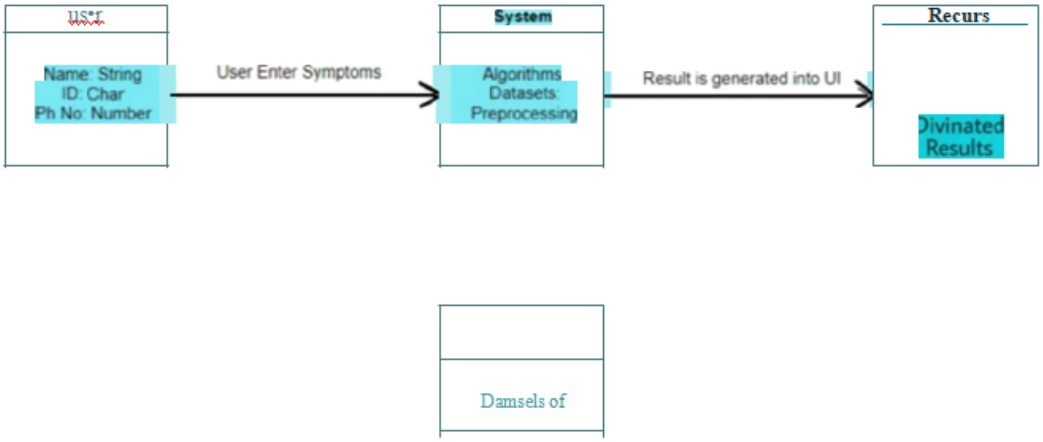


Fig 10. Class Diagram

3.16 Component Diagram

The arrangement and connections of the material components of a system are described in an object diagram, also known as a UML object di agrarn. Component drawings are frequently used to help the data implementation model and double-check that the planned development covers all areas of the needed system function. The object diagram depicts all of the major corn}xinents that make up the system. Design, Algorithm, File System, and Datasets are all linked in this way. The algorithm is used to process the results and offer accurate accuracy, while the U1 is used to show the results appropriately in the system and the file system is used to store user data. Therefore, as all components are connected.

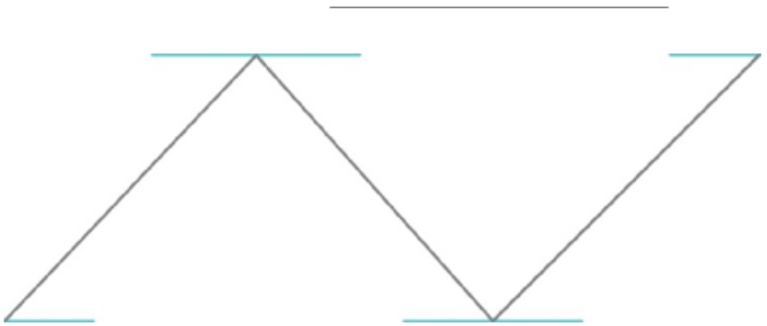


Fig 11. Component Diagram

3.17 State Diagram

The efficacy of one object in responding to a sequence of events in a system is depicted in a diagram of the Government chart. Harel's status chart or state machine diagram are other names for it. The dynamic flow of control is moved from the status quo to the state of something within the system in this UML diagram. lt is similar to the job diagram, but there are a few more rules, such as how to begin and terminate the programme. The programme begins with registration and login, and if login is successful, it will go to the next phase and. The user must then enter the symbols and hit the divination button after successfully logging in. Simultaneously, the retrieval process will complete its task and anticipate the relevant outcomes.

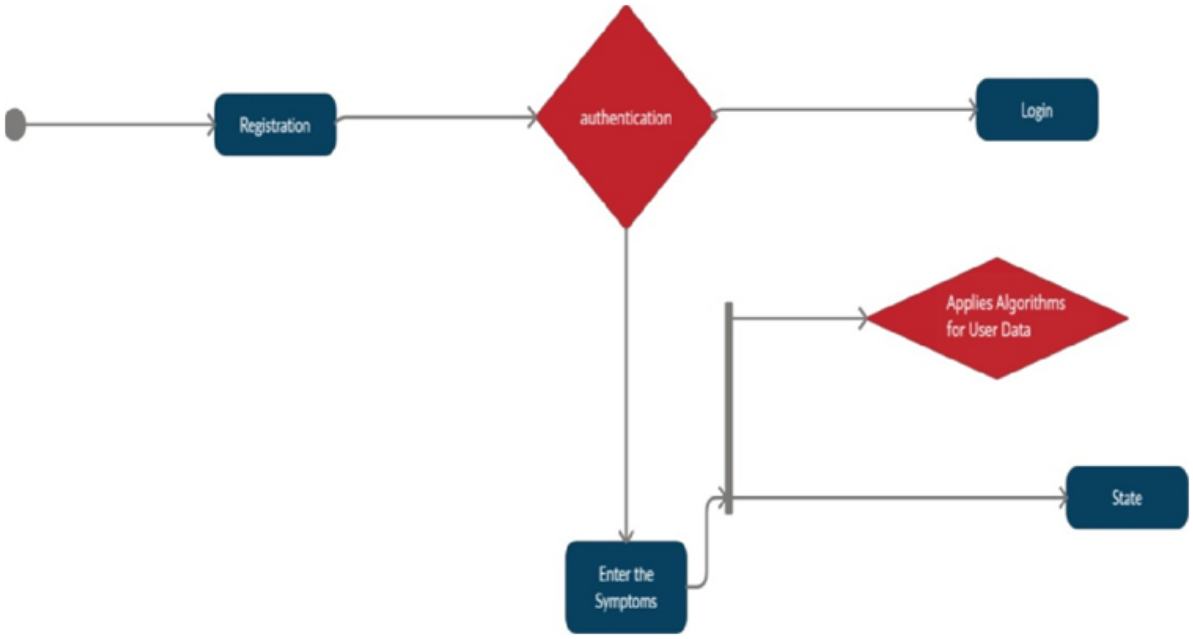


Fig 12. State Diagram

3.18 Deployment Diagram

The deployment diagram depicts the processing time per pixel in addition to the components that make up the time frame. Display diagrams are a kind of creation diagram which are used to symbolise the cloth of a goal item.

The Datasets for the Ailment Divination Model is referred from the Health Websites Health Department Officials and the corresponding Data's withinside the kaggle.corn. These datasets are saved withinside the database and the symptoms inputs from the Patients are skilled collectively through applying numerous Machine gaining knowledge of algorithms which incorporates numerous strategies referred as naive bayes classifier algorithms.

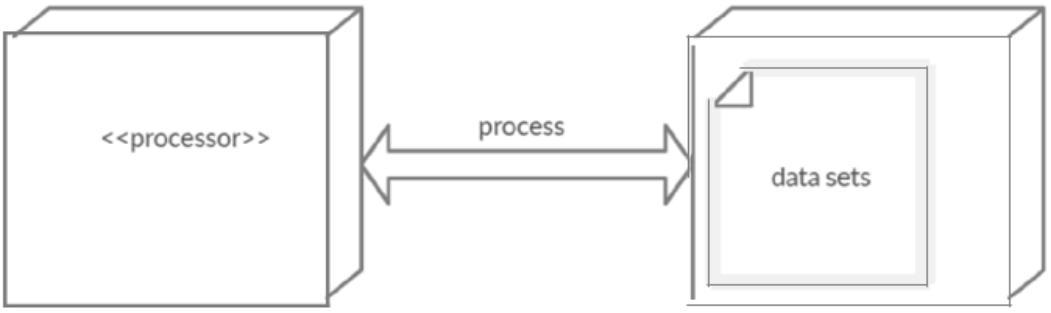


Fig 13. Deployment Diagram

Chapter 04:- PERFORMANCE ANALYSIS

4.1 Dataset

The disease's symptoms were gathered from kaggle.com and several health-related websites. The CSV file contains 5000 rows of patient information, including symptoms (132 distinct types of symptoms) and diseases (40 classes of general disease).

4.2 Performance Evaluation

For our model, we opted for the K Fold since it allows the model to be trained on multiple train-test splits. This offers a better idea of how well the model will perform on data that hasn't been seen before. The hold-out strategy, on the other hand, is based on a single train-test split. As a result, a function was written to run various algorithms with varied K values of KFold.

```
algo_dict = {'l_o_g':log,'m_n_b':mnb,'r_a_n':ran,'k_n_n':knn}
algo_train_scores={}
algo_test_scores={}
max_kfold = 11
for algo_name in algo_dict.keys():
    print(algo_name)
    tr_score,tst_score = evaluate(df,max_kfold,algo_dict[algo_name])
    algo_train_scores[algo_name] = tr_score
    algo_test_scores[algo_name] = tst_score
print(algo_train_scores)
print(algo_test_scores)
```

```
l_o_g
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
m_n_b
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
r_a_n
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
k_n_n
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
{'l_o_g': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'm_n_b': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'r_a_n': {2: 1.0, 4:
1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'k_n_n': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}}
{'l_o_g': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'm_n_b': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'r_a_n': {2: 0.9914634
146341463, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'k_n_n': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}}
```

Fig 14 . Performance Evaluating Function

In the k fold cross-validation, Random Forest was found to be the best match, with a training

score of 1.0 and a testing score of 0.98 and a k value of 2. The other algorithms all appear to be overfit.

```
df_test = pd.DataFrame(algo_test_scores)
df_train = pd.DataFrame(algo_train_scores)
df_test.plot(grid = 1)
plt.show()
```

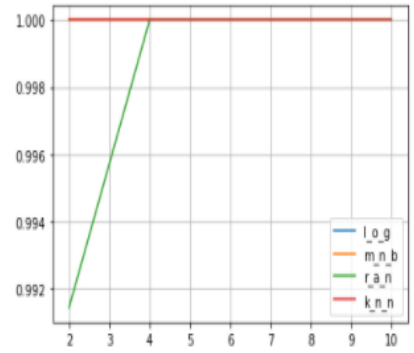


Fig 15. .Visual Comparison of different Algorithms

4.3 Prediction Of Disease

The accuracy of the predicted disease is directly proportional to the number of input symptoms.

```
psymptoms = []
psymptoms = input("Enter symptoms:- ")
psymptoms = psymptoms.split(",")


testingsymptoms = []
#append zero in all coloumn fields...
for x in range(0, len(symptomslist)):
  testingsymptoms.append(0)


#update 1 where symptoms gets matched...
for k in range(0, len(symptomslist)):

  for z in psymptoms:
      if (z == symptomslist[k]):
          testingsymptoms[k] = 1


inputtest = [testingsymptoms]

predicted = ran.predict(inputtest)
print("predicted disease is : ")
print(predicted)

y_pred_2 = ran.predict_proba(inputtest)
confidencescore=y_pred_2.max() * 100
print(" confidence score of : = {0} ".format(confidencescore))

confidencescore = format(confidencescore, '.0f')
predicted_disease = predicted[0]
```

```
Enter symptoms:-  headache,muscle_pain,vomiting
predicted disease is :
['Malaria']
 confidence score of : = 40.0
```

```
import joblib
joblib.dump(ran,'trained_model')
```

```
['trained_model']
```

Fig 16. Disease Prediction

4.4 Website

4.4.1 Users UI

Fig 17. Homepage for all Users

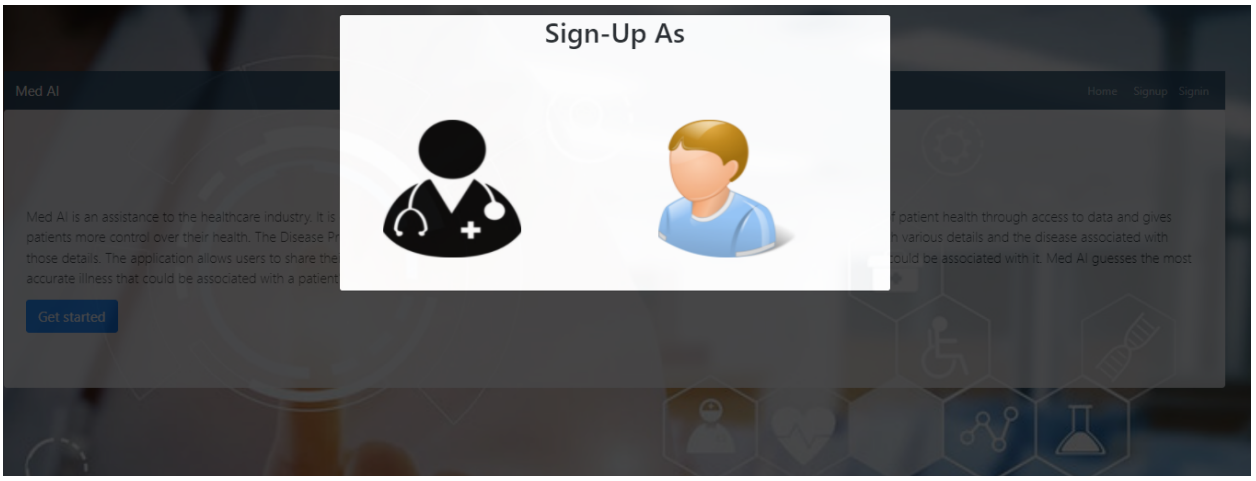


Fig 18. Signup Page

4.4.2 Patient UI

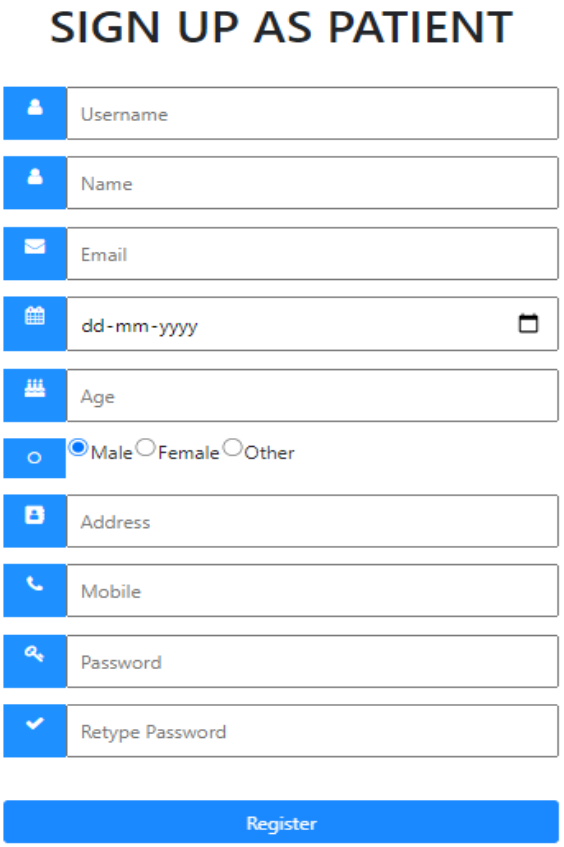


Fig 19. Signup Page for Patient

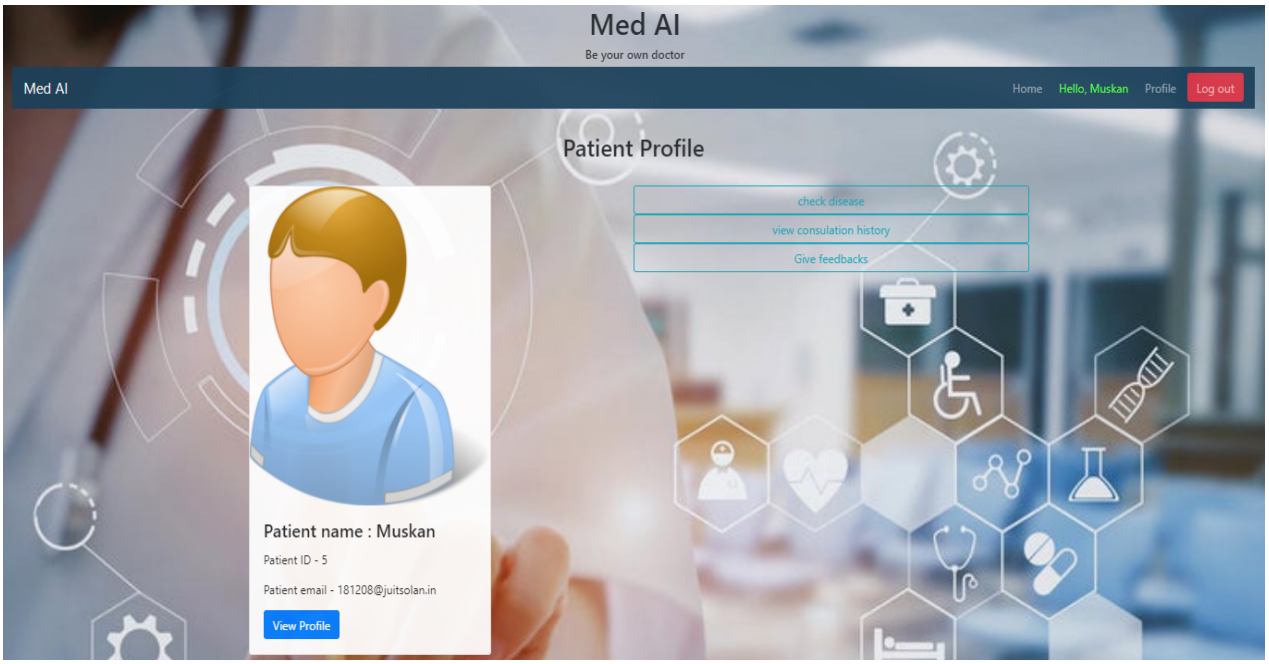


Fig 20. Patient Profile

Fig 21. Ailment Predictor

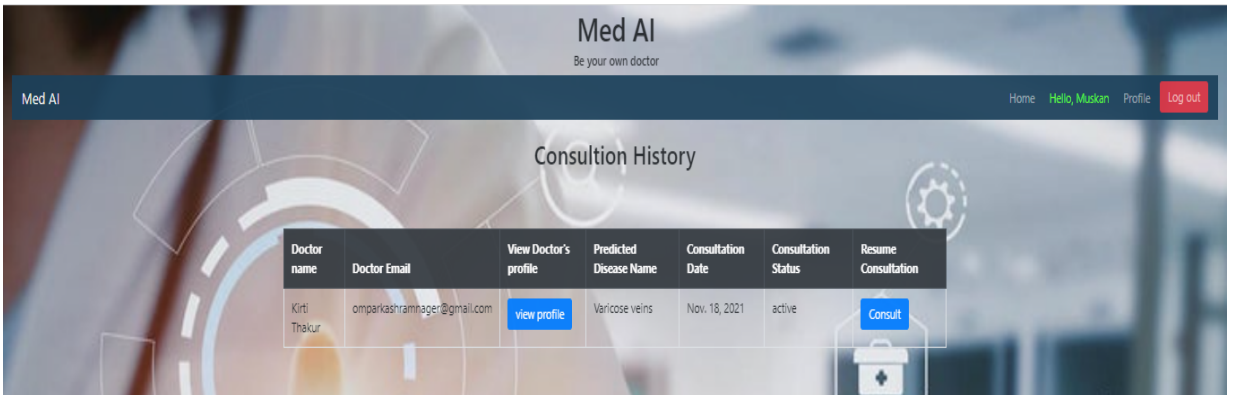


Fig 22. Consult a Doctor

4.4.3 Doctor UI

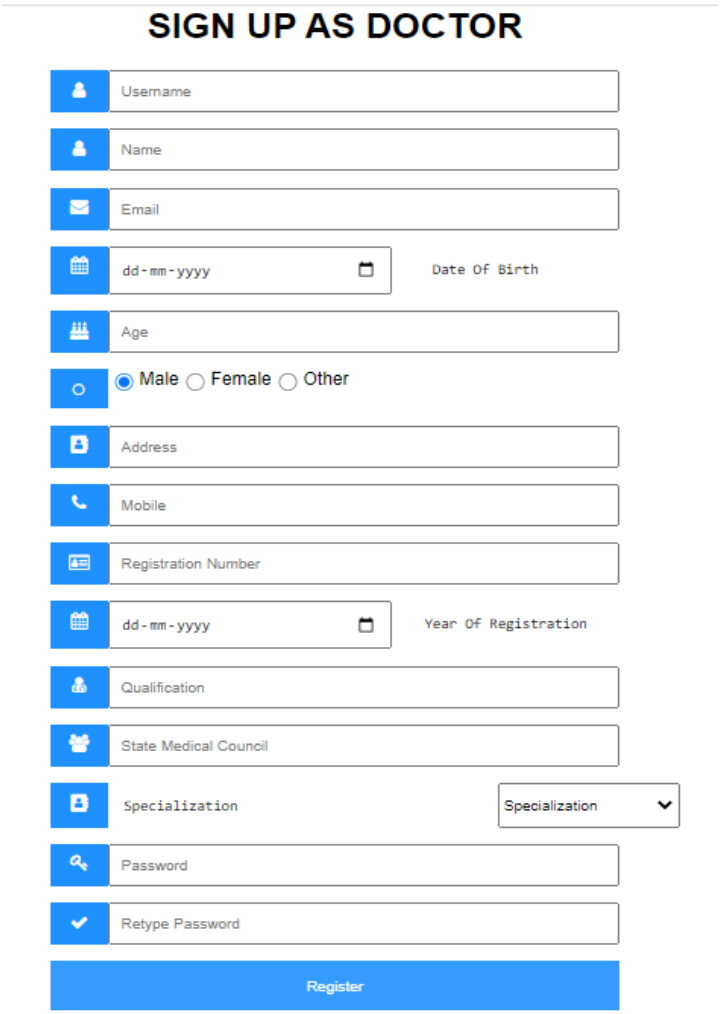


Fig 23. Signup Page for Doctor

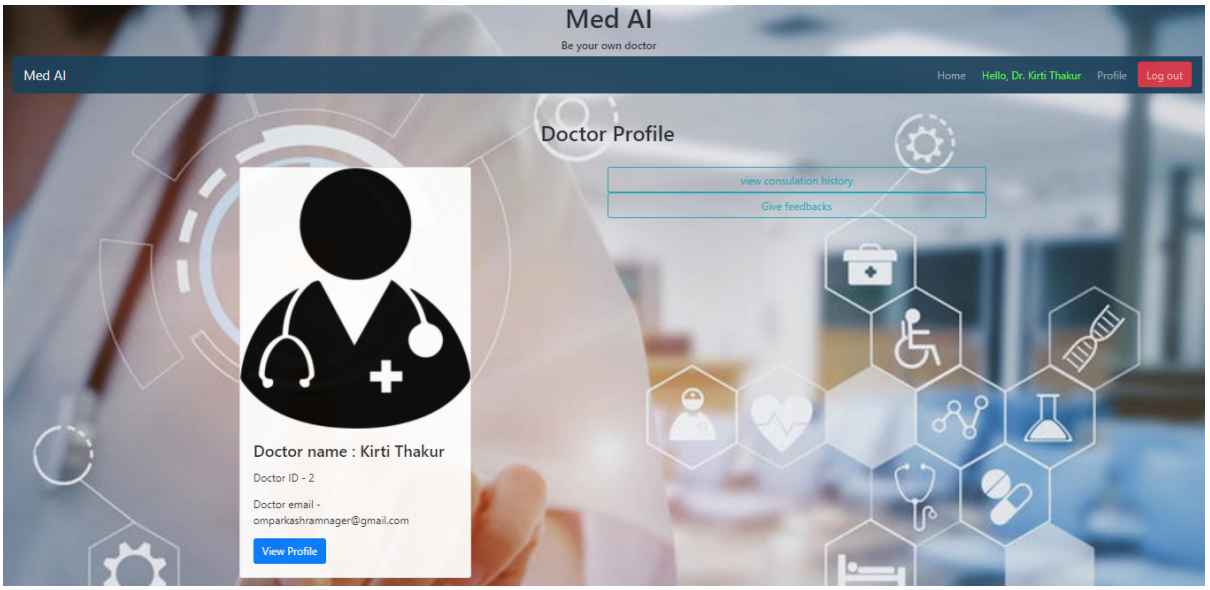


Fig 24. Doctor Profile

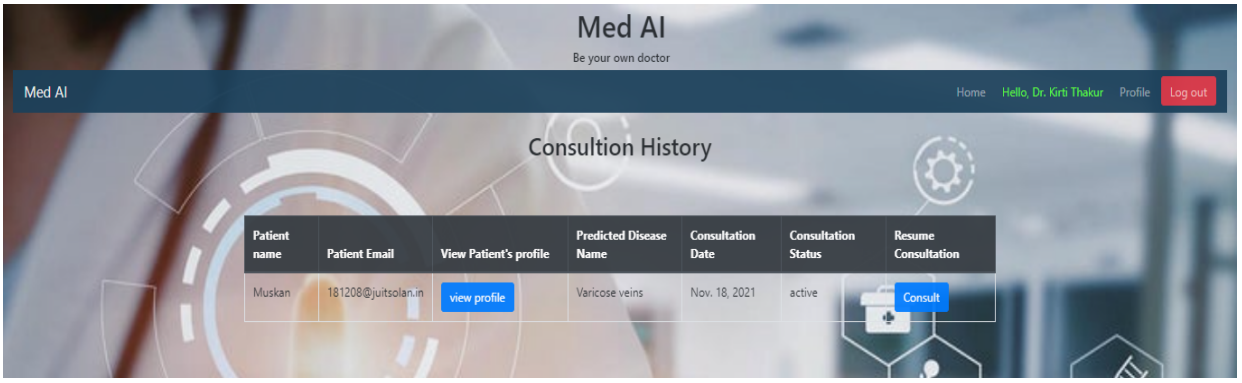


Fig 25. Consultation to the Doctor

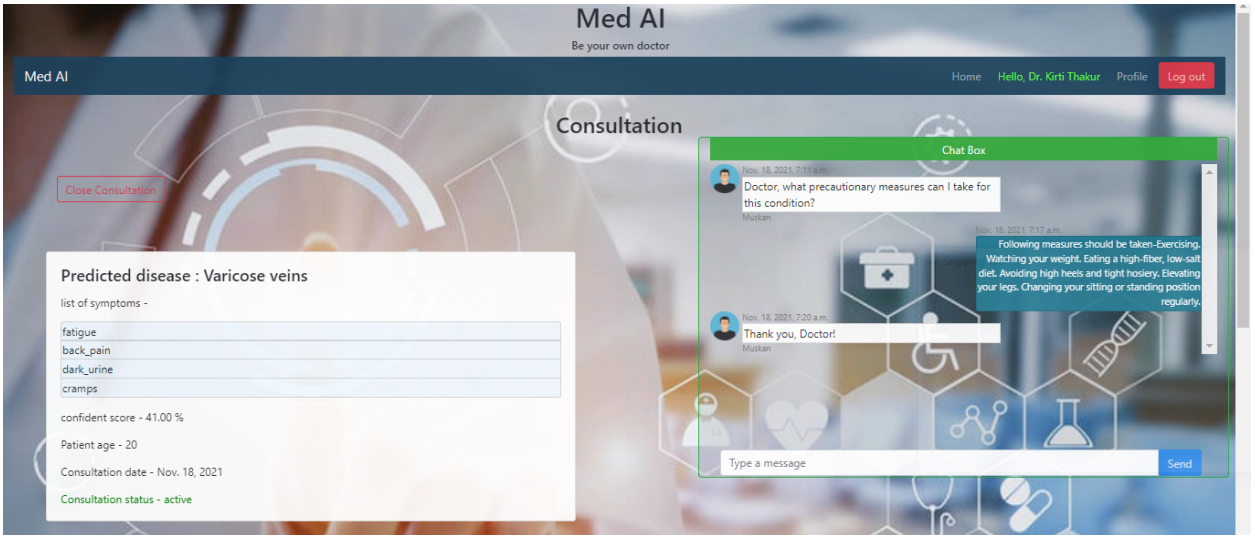


Fig 26. Consultation

4.4.4 Admin UI

Fig 27. Sign In Page for Admin

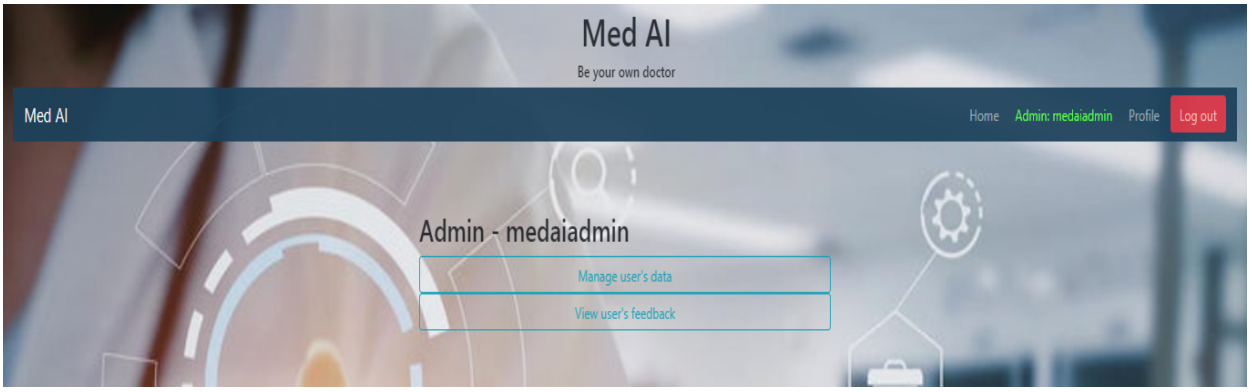


Fig 28. Admin Profile

Chapter 05:- CONCLUSION

5.1 Conclusion

Finally, we'll remark that this project Diagnostics is extremely useful in everyday life and is very essential in the field of health care because health care professionals are the ones who utilize these programs regularly to forecast patient diseases based on their general data and symptoms. Nowadays, the healthcare industry plays an important role in treating patients' illnesses, so this is also a form of assistance in the healthcare industry to inform the user and also useful to the user in case he does not want to go to the hospital or other clinics, so by adding signs and other useful information, they are already in control, and the healthcare industry can benefit from the program by simply implementing it. By just asking the user for symptoms and entering into the system, they may determine directly and to a degree accurately what diseases they have in just a few seconds. If the healthcare sector adopts this initiative, doctors will have less work to perform and will be able to anticipate a patient's condition more readily. Divination is used to treat a variety of common diseases that, if left untreated, can develop into a fatal condition that can affect the sufferer and his or her family.

5.2 Future Scope

Today, the majority of data in the digital world is disseminated and misused. We can apply the existing data to unknown patterns by studying them. The future goal of this research is to develop a method for accurately predicting cardiac disease. A logistic regression technique, Naive-Bayes, and SK-learn in machine learning can be used to predict cardiac disease. The future enhancement also involves making the application more user-friendly to better the user engagement and using improved techniques and algorithms, to better the outcomes and their accuracy.

REFERENCES

1. Dhanashri Gujar, Rashmi Biyani, Tejaswini Brahmani, Snehal Bhosale, Tejaswita P. Vaidya, “Ailment Divination and Doctor Recommendation System” www.ir.jet.net
2. Sambe, Ngutor, Abanyam, Noah Lumun, Iorkyaa, Philip Terwase “Ailment Divination Based on Prior Knowledge” by www. cup- us JSP
3. Shratik J. Mishra, Albar M. Vasi, Vinay Menon, K. Jayamalini - “GDPS General Ailment Divination System”
4. M. Shyamala Devi, Shermin Shamsudheen, Rincy Merlin Mathew “Ailment Divination Using Machine Learning by International Research I journal of Engineering and Technology (IRJET)”.
5. Kaveeshwar, S.A., and Cornwall, J., 2014, “The current state of ailment Mellitus in India”. AMI, 7(I), pp. 45-4S.
6. Dean, L., McEntyre, J., 2004, “The Genetic Landscape of Ailment | Internet j. Bethesda (MD): National Center for Biotechnology Information (US); Chapter I, Introduction to Ailment. 2004 I ul
7. IEEE “Machine Learning Methods Used in Ailment” by www.Wikipedia.com
8. Sapna Sharma , P.Sharma “Predictive Risk Factors of Heart Disease using an Efficient Classification based Approach”
9. Kerenalli, Sudarshana and Ko, Harshitha and TS, Pooja and Ramdas, Pooja and Hegde, Sam “A Novel Disease Divination System Using Field Data”
10. Thomas Bavo Azongo, Abdulai Abubakari “Role of diviners in healthcare delivery in Ghana: Implications and consequences”

APPENDICES

manage.py > main

```python
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'medi.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()

```

calc > urls.py > ...

```python
from django.urls import path , re_path
from . import views

urlpatterns = [
    path("", views.home, name="home"),

    path('admin_ui', views.admin_ui , name='admin_ui'),

    path('patient_ui', views.patient_ui , name='patient_ui'),
    path('checkdisease', views.checkdisease, name="checkdisease"),
    path('pviewprofile/<str:patientusername>', views.pviewprofile , name='pviewprofile'),
    path('pconsultation_history', views.pconsultation_history , name='pconsultation_history'),
    path('consult_a_doctor', views.consult_a_doctor , name='consult_a_doctor'),
    path('make_consultation/<str:doctorusername>', views.make_consultation , name='make_consultation'),
    path('rate_review/<int:consultation_id>', views.rate_review , name='rate_review'),


    path('dconsultation_history', views.dconsultation_history , name='dconsultation_history'),
    path('dviewprofile/<str:doctorusername>', views.dviewprofile , name='dviewprofile'),
    path('doctor_ui', views.doctor_ui , name='doctor_ui'),



    path('consultationview/<int:consultation_id>', views.consultationview , name='consultationview'),
    path('close_consultation/<int:consultation_id>', views.close_consultation , name='close_consultation'),


    path('post', views.post, name='post'),
    path('chat_messages', views.chat_messages, name='chat_messages'),



]
```

calc > models.py > ...

```python
from django.db import models
from django.contrib.auth.models import User
from django.contrib.postgres.fields import ArrayField
from datetime import date
# Create your models here.
#user = models.OneToOneField(settings.AUTH_USER_MODEL)
class patient(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, primary_key=True)
    is_patient = models.BooleanField(default=True)
    is_doctor = models.BooleanField(default=False)
    name = models.CharField(max_length = 50)
    dob = models.DateField()
    address = models.CharField(max_length = 100)
    mobile_no = models.CharField(max_length = 15)
    gender = models.CharField(max_length = 10)
    @property
    def age(self):
        today = date.today()
        db = self.dob
        age = today.year - db.year
        if today.month < db.month or today.month == db.month and today.day < db.day:
            age -= 1
        return age
class doctor(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, primary_key=True)
    is_patient = models.BooleanField(default=False)
    is_doctor = models.BooleanField(default=True)
    name = models.CharField(max_length = 50)
    dob = models.DateField()
    address = models.CharField(max_length = 100)
    mobile_no = models.CharField(max_length = 15)
    gender = models.CharField(max_length = 10)
    registration_no = models.CharField(max_length = 20)
```

calc > models.py > ...

```python
    year_of_registration = models.DateField()
    qualification = models.CharField(max_length = 20)
    State_Medical_Council = models.CharField(max_length = 30)
    specialization = models.CharField(max_length = 30)
    rating = models.IntegerField(default=0)
class diseaseinfo(models.Model):
    patient = models.ForeignKey(patient , null=True, on_delete=models.SET_NULL)
    diseasename = models.CharField(max_length = 200)
    no_of_symp = models.IntegerField()
    symptomsname = ArrayField(models.CharField(max_length=200))
    confidence = models.DecimalField(max_digits=5, decimal_places=2)
    consultdoctor = models.CharField(max_length = 200)
class consultation(models.Model):
    patient = models.ForeignKey(patient ,null=True, on_delete=models.SET_NULL)
    doctor = models.ForeignKey(doctor ,null=True, on_delete=models.SET_NULL)
    diseaseinfo = models.OneToOneField(diseaseinfo, null=True, on_delete=models.SET_NULL)
    consultation_date = models.DateField()
    status = models.CharField(max_length = 20)
class rating_review(models.Model):
    patient = models.ForeignKey(patient ,null=True, on_delete=models.SET_NULL)
    doctor = models.ForeignKey(doctor ,null=True, on_delete=models.SET_NULL)
    rating = models.IntegerField(default=0)
    review = models.TextField( blank=True )
    @property
    def rating_is(self):
        new_rating = 0
        rating_obj = rating_review.objects.filter(doctor=self.doctor)
        for i in rating_obj:
            new_rating += i.rating
        new_rating = new_rating/len(rating_obj)
        new_rating = int(new_rating)
        return new_rating
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

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

```
df.head()
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	blackheads	scurrin
0	1	1	1	0	0	0	0	0	0	0	...	0	
1	0	1	1	0	0	0	0	0	0	0	...	0	
2	1	0	1	0	0	0	0	0	0	0	...	0	
3	1	1	0	0	0	0	0	0	0	0	...	0	
4	1	1	1	0	0	0	0	0	0	0	...	0	

5 rows × 133 columns

```
df.columns
```

```
Index(['itching', 'skin_rash', 'nodal_skin_eruptions', 'continuous_sneezing',
       'shivering', 'chills', 'joint_pain', 'stomach_pain', 'acidity',
```

```
from scipy.stats import chi2_contingency
```

```
chi2_contingency(pd.crosstab(df['cold_hands_and_feets'],df['weight_gain']))
```

```
(4362.40173527245,
 0.0,
 1,
 array([[4.69464146e+03, 1.11358537e+02],
        [1.11358537e+02, 2.64146341e+00]]))
```

```
x = df.drop(['prognosis'],axis =1)
y = df['prognosis']
```

```
from sklearn.model_selection import train_test_split
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)
```

```
from sklearn.naive_bayes import MultinomialNB
mnb = MultinomialNB()
mnb = mnb.fit(x_train, y_train)
from sklearn.metrics import accuracy_score
y_pred = mnb.predict(x_test)
accuracy_score(y_pred,y_test)
```

```
1.0
```

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(mnb, x_test, y_test, cv=3)
print (scores)
print (scores.mean())
```

```
[1. 1. 1.]
1.0
```

```
from sklearn.model_selection import KFold
```

```
def evaluate(train_data,kmax,algo):
    test_scores = {}
    train_scores = {}
    for i in range(2,kmax,2):
        kf = KFold(n_splits = i)
        sum_train = 0
        sum_test = 0
        data = df
        for train,test in kf.split(data):
            train_data = data.iloc[train,:]
            test_data = data.iloc[test,:]
            x_train = train_data.drop(["prognosis"],axis=1)
            y_train = train_data['prognosis']
            x_test = test_data.drop(["prognosis"],axis=1)
            y_test = test_data["prognosis"]
            algo_model = algo.fit(x_train,y_train)
            sum_train += algo_model.score(x_train,y_train)
            y_pred = algo_model.predict(x_test)
            sum_test += accuracy_score(y_test,y_pred)
        average_test = sum_test/i
        average_train = sum_train/i
        test_scores[i] = average_test
        train_scores[i] = average_train
        print("kvalue: ",i)
    return(train_scores,test_scores)
```

```
from sklearn.ensemble import GradientBoostingClassifier
gbm = GradientBoostingClassifier()
from sklearn.linear_model import LogisticRegression
log = LogisticRegression()
mnb = MultinomialNB()
from sklearn.ensemble import RandomForestClassifier
ran = RandomForestClassifier(n_estimators =5)
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
```

```
algo_dict = {'l_o_g':log,'m_n_b':mnb,'r_a_n':ran,'k_n_n':knn}
algo_train_scores={}
algo_test_scores={}
max_kfold = 11
for algo_name in algo_dict.keys():
    print(algo_name)
    tr_score,tst_score = evaluate(df,max_kfold,algo_dict[algo_name])
    algo_train_scores[algo_name] = tr_score
    algo_test_scores[algo_name] = tst_score
print(algo_train_scores)
print(algo_test_scores)
```

```
l_o_g
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
m_n_b
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
r_a_n
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
k_n_n
kvalue:  2
kvalue:  4
kvalue:  6
kvalue:  8
kvalue:  10
{'l_o_g': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'm_n_b': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'r_a_n': {2: 1.0, 4:
1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'k_n_n': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}}
{'l_o_g': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'm_n_b': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'r_a_n': {2: 0.9914634
146341463, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}, 'k_n_n': {2: 1.0, 4: 1.0, 6: 1.0, 8: 1.0, 10: 1.0}}
```

```
test_scores={}
train_scores={}
for i in range(2,4,2):
    kf = KFold(n_splits = i)
    sum_train = 0
    sum_test = 0
    data = df
    for train,test in kf.split(data):
        train_data = data.iloc[train,:]
        test_data = data.iloc[test,:]
        x_train = train_data.drop(["prognosis"],axis=1)
        y_train = train_data['prognosis']
        x_test = test_data.drop(["prognosis"],axis=1)
        y_test = test_data["prognosis"]
        algo_model = ran.fit(x_train,y_train)
        sum_train += ran.score(x_train,y_train)
        y_pred = ran.predict(x_test)
        sum_test += accuracy_score(y_test,y_pred)
    average_test = sum_test/i
    average_train = sum_train/i
    test_scores[i] = average_test
    train_scores[i] = average_train
    print("kvalue: ",i)
print(train_scores)
print(test_scores)
```

```
kvalue:  2
{2: 1.0}
{2: 0.9975609756097561}
```

```
diseaselist=['Fungal infection','Allergy','GERD','Chronic cholestasis','Drug Reaction','Peptic ulcer diseae','AIDS','Diabetes ',
  'Gastroenteritis','Bronchial Asthma','Hypertension ','Migraine','Cervical spondylosis','Paralysis (brain hemorrhage)',
  'Jaundice','Malaria','Chicken pox','Dengue','Typhoid','hepatitis A', 'Hepatitis B', 'Hepatitis C', 'Hepatitis D',
  'Hepatitis E', 'Alcoholic hepatitis','Tuberculosis', 'Common Cold', 'Pneumonia', 'Dimorphic hemmorhoids(piles)',
  'Heart attack', 'Varicose veins','Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia', 'Osteoarthristis',
  'Arthritis', '(vertigo) Paroymsal  Positional Vertigo','Acne', 'Urinary tract infection', 'Psoriasis', 'Impetigo']
```

```
symptomslist=['itching','skin_rash','nodal_skin_eruptions','continuous_sneezing','shivering','chills','joint_pain',
  'stomach_pain','acidity','ulcers_on_tongue','muscle_wasting','vomiting','burning_micturition','spotting_ urination',
  'fatigue','weight_gain','anxiety','cold_hands_and_feets','mood_swings','weight_loss','restlessness','lethargy',
  'patches_in_throat','irregular_sugar_level','cough','high_fever','sunken_eyes','breathlessness','sweating',
  'dehydration','indigestion','headache','yellowish_skin','dark_urine','nausea','loss_of_appetite','pain_behind_the_eyes',
  'back_pain','constipation','abdominal_pain','diarrhoea','mild_fever','yellow_urine',
```

```
        psymptoms = []
        psymptoms = input("Enter symptoms:- ")
        psymptoms = psymptoms.split(",")


        testingsymptoms = []
        #append zero in all coloumn fields...
        for x in range(0, len(symptomslist)):
          testingsymptoms.append(0)


        #update 1 where symptoms gets matched...
        for k in range(0, len(symptomslist)):

          for z in psymptoms:
              if (z == symptomslist[k]):
                  testingsymptoms[k] = 1


        inputtest = [testingsymptoms]

        predicted = ran.predict(inputtest)
        print("predicted disease is : ")
        print(predicted)

        y_pred_2 = ran.predict_proba(inputtest)
        confidencescore=y_pred_2.max() * 100
        print(" confidence score of : = {0} ".format(confidencescore))

        confidencescore = format(confidencescore, '.0f')
        predicted_disease = predicted[0]
```

```
Enter symptoms:-  headache,muscle_pain,vomiting
predicted disease is :
['Malaria']
 confidence score of : = 40.0
```

```
import joblib
joblib.dump(ran,'trained_model')
```

```
['trained_model']
```