# Chapter-1: Introduction

## 1.1 Introduction

"Medcare" is a cutting-edge disease recommendation and prediction system that harnesses the power of advanced technology to revolutionize healthcare and improve patient outcomes. Our platform integrates a range of algorithms and data analysis techniques to offer personalized insights and predictive recommendations for a wide spectrum of medical conditions. With a strong focus on patient well-being and proactive healthcare management, "Medcare" serves as a beacon of innovation in the medical field. There are some key points related to our project:

* Timely disease Identification and prediction
* Improved patient care
* Real-time updates
* Personalized Recommendation

## 1.2 Problem Statement

With the rise in number of patient and disease every year medical system is overloaded and with time have become overpriced in many countries. Most of the disease involves a consultation with doctors to get treated. In today's fast-paced world, timely and accurate diagnosis of health issues is of paramount importance. However, individuals often face challenges in promptly identifying and addressing potential health concerns. This project aims to develop a machine learning-based system that predicts potential health problems based on user-provided symptoms. The existing healthcare infrastructure often experiences high patient volumes, leading to extended waiting times and delayed diagnoses. This solution seeks to alleviate this strain by providing a preliminary assessment of health conditions based on user-reported symptoms.

The primary objective of this project is to create an intuitive and user-friendly interface where individuals can input their symptoms, leading to the generation of a probabilistic prediction regarding the most likely health problems they might be facing. This predictive model will be trained on a comprehensive dataset encompassing a wide range of medical conditions and their associated symptoms. Additionally, the model will continuously learn and adapt based on user feedback and emerging medical research.

The successful implementation of this system will revolutionize the way individuals engage with their own health, empowering them to make informed decisions and seek appropriate medical attention in a timely manner. Furthermore, it has the potential to act as a valuable tool for healthcare providers, enabling them to prioritize and optimize patient care based on initial symptom assessments.

## 1.3 Objective

### 1.3.1 Major Objectives

The main objectives of this project are as follows:

* Identify potential health issues at an early stage by analyzing symptoms.
* Improve diagnostic accuracy through predictive analytics.
* Provide recommendations for Drugs.
* Appointment with the Doctor.

### 1.3.2 Project as a solution

Our project aims to accomplish the above-mentioned goal by effectively implementing a module on the unique web-based application to solve the issue.

## 1.4 Scope and Limitation

### 1.4.1 Project Scope

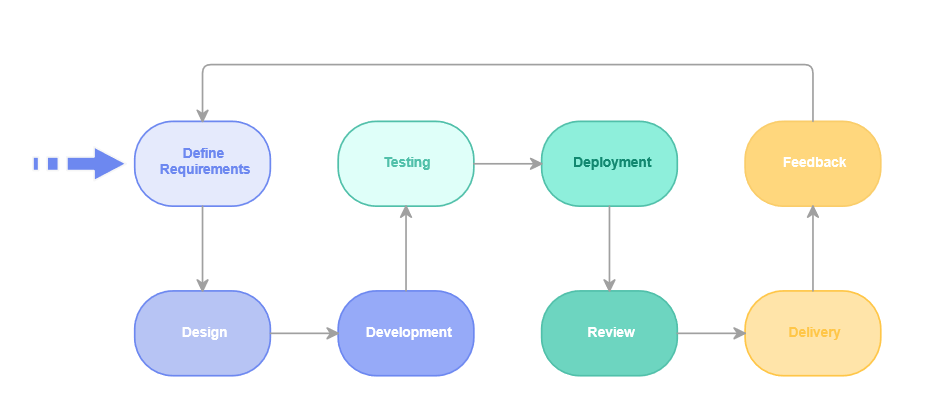
This portion of our project describes the project's objectives, delivery strategy, and level of effectiveness**.** Create algorithms to recommend appropriate medications based on individual patient profiles, disease predictions, and known drug interactions.

### 1.4.2 Project Limitation

Although this initiative achieves some of its objectives, there is still room for improvement. The primary limitation is the system's size, which was intended to accommodate a limited number of users. Furthermore, the models' accuracy could differ because so little data was utilized to train them. It's possible that the model won't adjust effectively to emerging illnesses or shifting epidemiological trends. Resource-intensive models may be more difficult to use in settings with limited resources since they demand a large amount of infrastructure and processing power.

## 1.5 Methodology

The selection of the Scrum framework for Agile development plays a pivotal role. Scrum is chosen due to its iterative and incremental approach, which is particularly well-suited for projects like "Medcare” Scrum divides the project into manageable iterations called sprints, typically 2 to 4 weeks long, during which specific tasks and features are developed and tested. This iterative and incremental process ensures that at the end of each sprint, there's a working portion of the project, allowing for continuous feedback from stakeholders, healthcare professionals, and patients. This feedback is used to adapt and refine the project, ensuring it aligns closely with user needs and the evolving healthcare landscape. Scrum's focus on flexibility, adaptability, and collaboration with users makes it ideal for "Medcare” enabling a dynamic development process that can respond effectively to changing requirements and incorporate valuable feedback.

**

# Chapter 2: Background Study and Literature Review

## 2.1 Fundamental Theories

Medcare is a website that provides facility to the general public who is sick/ill. It provides help to the people digitally without having to go in physical facility. The users can simply use the website to login and input the symptoms that they are facing and then have the drugs recommended and they can also take appointment of the doctor for further checkup. Our facility will be provided in form of mobile application and website.

### 2.1.1 Description of ‘Random Forest’ Algorithm

A Random Forest is an ensemble learning algorithm that is used for both classification and regression tasks. It is a versatile and powerful machine learning method that operates by constructing a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

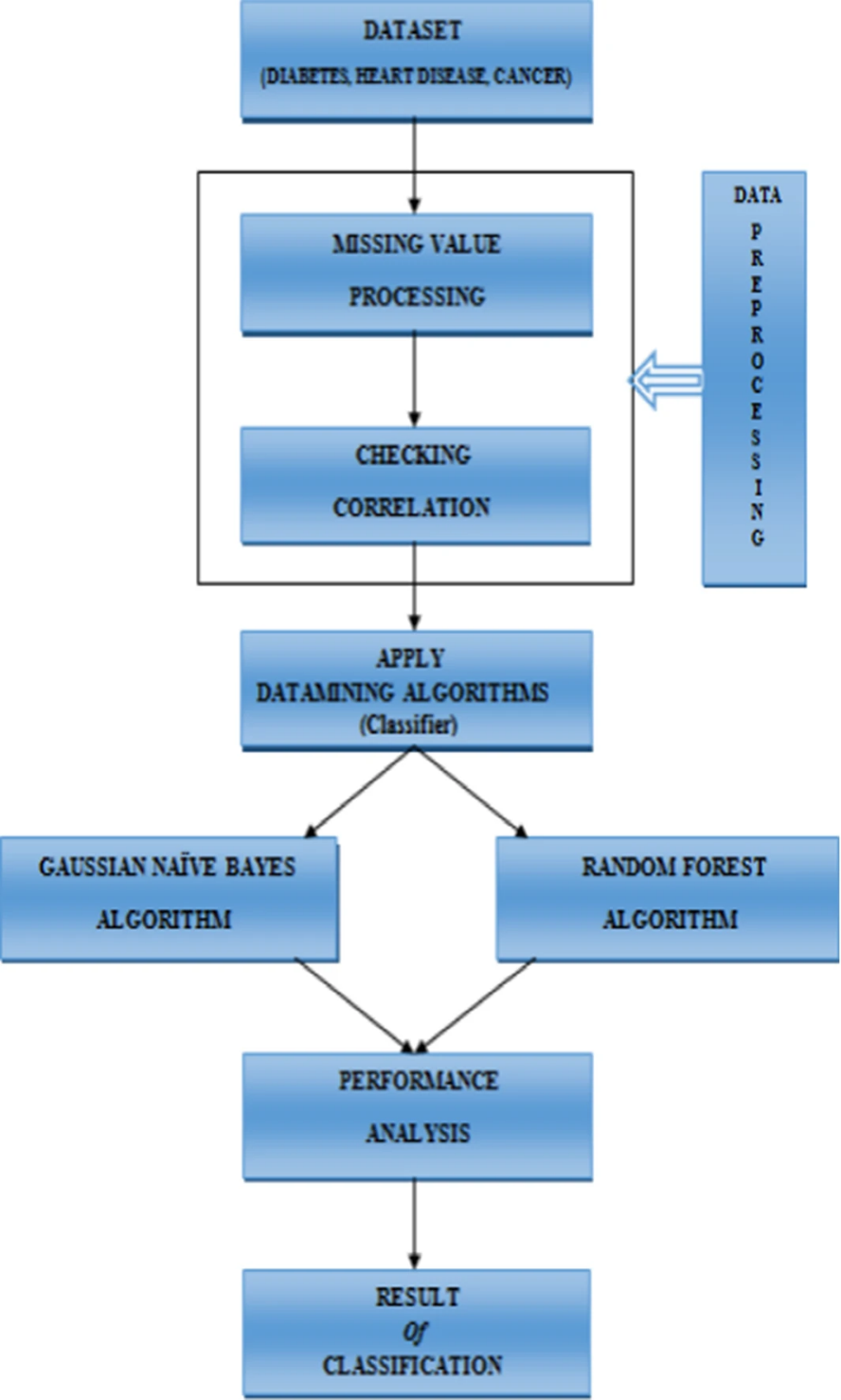
Here's a step-by-step description of how the Random Forest algorithm works:

1. **Data Collection**: Gather a dataset that includes information on symptoms, diseases, and corresponding drug recommendations.
2. **Feature Extraction**: Identify relevant features from the dataset, such as various symptoms and other relevant patient information.
3. **Training the Random Forest**: Utilize the Random Forest algorithm to train a model on the dataset. During training, the algorithm will create an ensemble of decision trees based on subsets of the data and symptoms.
4. **Symptom-Based Decision Trees**: Each decision tree in the Random Forest focuses on subsets of symptoms and their relationships with diseases. The randomness introduced during tree construction helps capture diverse patterns in symptom-disease associations.
5. **Prediction Process**: When a new set of symptoms is input into the system, the ensemble of decision trees collectively predicts the likely disease. For drug recommendation, this prediction can be further used to suggest appropriate medications associated with the predicted disease.
6. **Ensemble Voting for Robust Predictions**: The Random Forest ensemble provides robustness by aggregating predictions from multiple decision trees. The final disease prediction is often determined by majority voting among the trees.
7. **Drug Recommendation**: Based on the predicted disease, the system can recommend drugs commonly prescribed or associated with the identified condition. This recommendation can be further refined based on additional patient-specific factors if available in your dataset.

### 2.1.2 Description of ‘Naïve Bayes’ Algorithm

Naive Bayes is particularly suitable for text classification tasks, making it effective for symptom-based disease prediction where symptoms can be treated as features. It is computationally efficient and can provide quick predictions, making it a valuable component in a symptom-based disease prediction system with drug recommendation

1. **Data Representation**: Collect a dataset with symptom-disease pairs, where each instance consists of a set of symptoms associated with a particular disease. Include drug information corresponding to each disease in the dataset.
2. **Preprocessing:** Preprocess the data by representing symptoms and diseases in a format suitable for Naive Bayes. This may involve converting categorical data into a numerical format.
3. **Training:** Use the preprocessed data to train a Naive Bayes classifier. The algorithm estimates the probability of a disease given a set of symptoms by calculating the conditional probabilities of each symptom given the presence of a disease.
4. **Classification:** Given a set of symptoms for a patient, use the trained Naive Bayes model to predict the probability of each disease. The model assigns the patient to the disease with the highest probability.
5. **Drug Recommendation**: Once the disease is predicted, associate it with the corresponding drug information from the dataset. Provide drug recommendations based on the predicted disease.
6. **Handling Independence Assumption**: Naive Bayes assumes that symptoms are conditionally independent given the disease, which may not be entirely accurate in real-world scenarios. However, this simplifying assumption often works well in practice.
7. **Feedback Loop:** Implement a feedback loop where user feedback on the accuracy of predictions and drug recommendations can be used to update and improve the model over time.



## 2.2 Related Works

There has been a significant amount on disease prediction system in last decade. Some of them are listed below:

1. "A Survey on Deep Learning in Medical Image Analysis" (2017)

This survey provides insights into the applications of deep learning, including neural networks, in medical image analysis. While it may not focus explicitly on symptoms, understanding image-based disease diagnosis is crucial in the broader context of healthcare prediction systems.

1. "Prediction of Heart Disease Using Classification Algorithms" (2018)

Some studies focus on specific diseases, such as heart disease. This paper discusses the use of classification algorithms for predicting heart disease based on symptoms and patient data.

1. "Predictive Modeling of Hospital Readmission Rates Using Electronic Medical Record-Wide Machine Learning: A Case-Study Using Mount Sinai Heart Failure Cohort" (2018)

This work explores predictive modeling of hospital readmission rates using machine learning. While not specifically symptom-based, it demonstrates the use of machine learning in predicting healthcare outcomes.

1. "Disease Prediction by Machine Learning Over Big Data from Healthcare Communities" (2017)

This paper discusses the use of machine learning over large healthcare datasets for disease prediction. It may provide insights into handling diverse data sources and predicting diseases based on symptoms.

1. "A Review on Heart Disease Prediction using Data Mining Techniques" (2018)

Reviews like this one summarize existing literature on disease prediction using data mining techniques, including methods that use symptoms as input features.

## 2.3 Existing Systems

Several systems and platforms have been developed for symptom-based disease prediction. These systems often leverage machine learning, artificial intelligence, and data analytics to enhance accuracy in disease prediction.

**Ada Health:**

Ada Health is an AI-powered health platform that uses a symptom-checker to help users assess their symptoms and understand potential health issues. The system considers various symptoms to provide personalized health information and advice.

**Buoy Health:**

Buoy Health is an AI-driven platform that assists users in understanding their symptoms and guides them to appropriate healthcare resources. The system uses a chat-based interface to gather information about symptoms and offers potential diagnoses.

**Google's Symptom Search:**

Google has implemented a symptom search feature that provides information about possible conditions based on entered symptoms. It aims to help users understand potential health issues and encourage informed discussions with healthcare professionals.

**Adastra Health:**

Adastra Health offers a symptom-checking tool that uses AI to analyze symptoms and provide information on potential health conditions. It aims to assist individuals in understanding their symptoms and deciding when to seek medical attention.

## 2.4 Existing Research Works

In medical domains, artificial intelligence (AI) primarily focuses on developing the algorithms and techniques to determine whether a system’s behavior is correct in disease diagnosis. Medical diagnosis identifies the disease or conditions that explain a person’s symptoms and signs. Typically, diagnostic information is gathered from the patient’s history and physical examination [1]. According to the National Academics of Science, Engineering, and Medicine report of 2015, the majority of people will encounter at least one diagnostic mistake during their lifespan [[2](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8950225/#B4-healthcare-10-00541)]. The appropriate application of ML to these data promises to transform patient risk stratification broadly in the field of medicine and especially in infectious diseases. This, in turn, could lead to targeted interventions that reduce the spread of healthcare-associated pathogens. In this review, we begin with an introduction to the basics of ML [3].

There has been research on Efficient Heart Disease Prediction Using Hybrid Deep Learning Classification Models. To further enhance this process, a newly developed GSA (Genetic Sine Algorithm) is proposed as it is capable of selecting optimal features and avoid getting trapped in local optima. The selected features are subjected to the classification technique by RNN (Recurrent Neural Network) integrated with LSTM (Long Short-Term Memory) algorithm. To filter out all the [invalid information](https://www.sciencedirect.com/topics/engineering/invalid-information) and emphasize only on critical information, DPA-RNN+LSTM (Deep Progressive Attention-RNN+LSTM) has been developed so as to improve the classification rate.[4]

Machine learning (ML) is used practically everywhere, from cutting-edge technology (such as mobile phones, computers, and robotics) to health care (i.e., disease diagnosis, safety). ML is gaining popularity in various fields, including disease diagnosis in health care. Many researchers and practitioners illustrate the promise of machine-learning-based disease diagnosis (MLBDD), which is inexpensive and time-efficient [5]. Traditional diagnosis processes are costly, time-consuming, and often require human intervention. While the individual’s ability restricts traditional diagnosis techniques, ML-based systems have no such limitations, and machines do not get exhausted as humans do. As a result, a method to diagnose disease with outnumbered patients’ unexpected presence in health care may be developed.

# Chapter 3: System Analysis

## 3.1. System Analysis

### 3.1.1. Requirement Analysis

Requirement analysis is the essential phase of systematically gathering and examining the essential elements for a system or project. This process involves comprehending the core issue or opportunity that the system intends to address, along with identifying both functional and non-functional prerequisites crucial for achieving the desired objectives. The requirement analysis for our healthcare platform entails two primary sections: Functional Requirements and Non-Functional Requirements.

### i. Functional Requirement:

Functional requirements describe what the system needs to do. They show the tasks and actions it should perform. For our project, the system will be a website that works well on different devices. It's important that anyone using the website won't have trouble because of the device they're using. Our focus is to make sure the website works smoothly on computers.

|  |  |  |
| --- | --- | --- |
| Req no. | Req. name | Req. Description |
| FR1 | Symptom Input | Users can input their symptoms into the system. |
| FR2 | Disease Prediction | The system should predict potential diseases based on symptoms. |
| FR3 | User Profile Management | Users can create, view, and manage their profiles. |
| FR4 | User Interface | Ensure an interactive interface for smooth user experience. |

### ii. Non-Functional Requirements:

|  |  |  |
| --- | --- | --- |
| Req. No. | Req. name | Req. Description |
| NFR1 | Scalability | The application will expand beyond the scope of a college project. Necessary measures, such as server upgrades and team expansion, will be implemented accordingly. |
| NFR2 | Maintainability | A comprehensive documentation of all system components will ensure easy maintenance, allowing future developers to uphold application quality effortlessly. |
| NFR3 | Usability | The website will prioritize simplicity, making features easily accessible and navigable. User interface components will be thoughtfully organized for effective usage. |
| NFR4 | Security | System administration privileges, like assigning doctors and managing permissions, will be restricted to authorized team members. Additionally, the system will be secured against SQL Injection attacks. |

## 3.2 Feasibility Study:

### 3.2.1 Technical Feasibility:

The proposed medical alert system necessitates intricate ICT solutions for continuous patient monitoring. Successful operation relies on seamless integration of hardware and software components. Once established and automated, it will efficiently gather and process patient data, utilizing specialized algorithms for timely alerts.

### 3.2.2 Economic Feasibility:

This is a web-based facility which is developed without using any physical hardware. To gather the necessary tools and implement them with the proper algorithm for hospitals, it costs the least possible. Thus, developing and maintaining Medcare is economically feasible.

### 3.2.3 Operational Feasibility:

Ensuring practicality and usability is crucial for successful adoption. Reliable models and algorithms promptly notify healthcare professionals in case of patient distress. The intuitive user interface design facilitates efficient operation, ensuring high operational feasibility.

### 3.2.4 Schedule Feasibility:

The project plan includes well-defined milestones, allowing completion within the stipulated timeline. Allocated time accounts for system development, testing, and refinement stages, ensuring adherence to the schedule.

# Chapter 5: Implementation and Testing

## 5.1 Implementation

### 5.1.1 Tools Used

|  |  |
| --- | --- |
| **Category** | **Tools Used** |
| Diagram | Draw.io, Photoshop |
| Communication | Discord, Google Meet, Viber |
| UI/UX | Figma |
| Code Editors | Jupyter, Sublime Text, VSCode |
| Documentation | Microsoft Office |
| Presentation | Microsoft PowerPoint |

### 5.1.2 Technology Used

|  |  |
| --- | --- |
| **Category** | **Technologies** |
| Frontend | Html, CSS, JS, Bootstrap |
| Backend | Django, MySQL |
| Machine Learning Libraries | SckitLearn, Pandas, Numpy, Joblib |
| Version Control | Github |

## 5.2 Description of disease prediction model

### 5.2.1 Data gathering

Data gathering is a critical phase in the development of a Disease Prediction System using Artificial Intelligence (AI). The quality and diversity of the data collected directly influence the accuracy and reliability of the predictions made by the system.

Here, in this project we used datasets from online sources like Kaggle as well we will also provide manual data for the operation of our project.

### 5.2.2 Data Preparation for Classification Modeling

The combined dataset was splitted into training and testing datasets. The dataset was splitted into around 70% training data and 30% testing data.

### 5.2.3 Building the Gaussian Naive Bayes Model

When Developing the Gaussian Navi Bayes Model, our first plan was to ensure it’s well organized with feature and target. We Split the data into features (X) and the target variable(y). To train the model, we used a smaller dataset with 10 columns. After fitting the model for 10 columns, the following loss and accuracy scores were obtained.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Columns | Train Dataset | Test Dataset | Train Accuracy | Test Accuracy |
|  |  |  |  |  |

The model performance was not that great, but was a good starting point. Therefore, we worked on improving the model’s performance by increasing the number of datasets and increasing the number of epochs to 20. After the implementation of these strategies, following results were observed.

### 5.2.4 Building Random Forest Model

## 5.3 Testing

### 5.3.1 Unit Testing

Modules are tested using unit testing in comparison to the comprehensive design. The design and coding of the project have been tested at every stage. To ensure that data is correctly entering and leaving the program unit, we verify the module interface during the testing process. We verify that the data that is temporarily stored is intact while the algorithm is running by looking at the local data structure. Every path that deals with errors is tested at the end.

|  |  |
| --- | --- |
| Objective | Predict disease |
| Action | User input symptoms |
| Expected result | The output should be an expected disease |
| Actual Result | The application gave the expected result |
| Conclusion | The test was successful. |

………………………………….pic………………………

When the user inputs the given symptoms, the module gave an expected output( disease predicted) .

|  |  |
| --- | --- |
| Objective | To make appointment. |
| Action | User request for an appointment. |
| Expected Result | The appointment is requested successfully. |
| Actual Result | The application gave a positive notification |
| Conclusion | The test was successful. |

………………………...pic………………………….

As soon as the user request for an appointment the appointment notification is sent to doctor panel which is ready to be approved or declined.

### 5.3.2 System Testing

System testing is conducted to verify the overall functionality, integration, and performance of a software system. It aims to detect and rectify defects, ensuring the software meets specified requirements. This phase evaluates the system's reliability, usability, and compatibility across various environments. Security measures are validated, and stakeholders gain confidence in the software's readiness for deployment. System testing is a crucial step to deliver a reliable and effective product to end-users.

|  |  |
| --- | --- |
| Objective | Opening the application |
| Action | The application was run through terminal command. |
| Expected Result | The application should load properly. |
| Actual Result | The application loaded properly. |
| Conclusion | The test was successful |