

Chronic Kidney Disease Prediction



By:

Muhammad Tayyab

10583

Muhammad Haroon Imtiaz

10959

Supervised by:

Yasir Ijaz

Co-Supervised by:

Salman Khizer

Faculty of Computing

Riphaah International University, Faisalabad

Spring 2022

A Thesis Report Submitted to

Faculty of Computing,
Riphah International University, Faisalabad

As a Partial Fulfillment of the Requirement for the Award of
the Degree of
Bachelor of Science in Computer Science

Faculty of Computing
Riphah International University, Faisalabad

Final Approval

This is to certify that we have read the report submitted by ***Muhammad Tayyab (10583)*** and ***Muhammad Haroon Imtiaz (10959)*** for the partial fulfillment of the requirements for the degree of the Bachelors of Science in Computer Science (BSCS). It is our judgment that this report is of sufficient standard to warrant its acceptance by Riphah International University, Faisalabad Campus for the degree of Bachelors of Science in Computer Science (BSCS).

Committee:

1

Yasir Ijaz
Supervisor

2

Salman Khizer
Co-Supervisor

3

Dr. Ijaz Ali Shaukat
Principal
Riphah College of Computing

Declaration

We hereby declare that this document “**Chronic Kidney Disease Prediction**” neither as a whole nor as a part has been copied out from any source. It is further declared that we have done this project with the accompanied report entirely on the basis of our personal efforts, under the proficient guidance of our teachers especially our supervisor **Yasir Ijaz** and co-supervisor **Salman Khizer**. If any part of the system is proved to be copied out from any source or found to be reproduction of any project from anywhere else, we shall stand by the consequences.

Muhammad Tayyab
10583

Muhammad Haroon Imtiaz
10959

Dedication

We dedicate this work to our beloved teachers. Also, to our family. They never failed to give us with guidance and encouragement, as well as to fulfil all of our necessities throughout time, we develop our system, and for teaching us that even the most difficult work can be accomplished when one step is taken at a time. We dedicate this work to all of those who have assisted us in whatever manner in completing it.

Acknowledgement

To begin, we owe it to Allah, the Almighty, the Compassionate, the Merciful, and the source of all of our accomplishments, to provide us with the bravery and our necessary abilities to fulfil this task. We want to show our gratitude to our Supervisor, Sir Yasir Ijaz, and Co-Supervisor, Sir Salman Khizer for their guidance and support. We are extremely grateful to them for showing their faith in us.

Muhammad Tayyab
10583

Muhammad Haroon Imtiaz
10959

Abstract

Machine learning based chronic kidney disease prediction is an application that predicts the illness establish on the data given by the end-user and provides an accurate result based on that data. Now since the health business plays such an essential part in healing patient's diseases, this is a great app for the health industry to tell the patient, and also depends on the patient if he/she want to go to a hospital. This web app based on machine learning that determine whether the patient has chronic kidney disease or not. It includes different machine learning and data science techniques to get insights from the data to analyze the data and build a model that predict whether the patient has chronic kidney disease based on the features. For this purpose, the data have been collected from two resources, Kaggle and University of California, Irvine Machine Learning Repository. This data contains 26 health-related features. The model has been using the principles of Supervised Machine Learning and Deep Learning. The model has been created using classification technique to predict the values and evaluate the risk for chronic kidney disease. This app has been trained using different machine learning models, Random Forest Tree and Gradient Boosting and an Artificial-Neural-Network. Different 8 features from 26 features are chosen. These are at the center of attention in terms of how much they influence the creatinine level. They are Packed Cell Volume, Red blood cells, Diabetes mellitus, Hemoglobin, Albumin, Appetite, Hypertension, and Specific gravity. So, applying frequently available robustness specification this model can evaluate the possibility of chronic kidney disease for people's health. In this way, it can decrease the impact of this disease and facilitate at this disease early detection.

Table of Contents

Chapter 1:.....	2
1. Introduction.....	2
1.1 Background	2
1.2 Motivations and Challenges	3
1.2.1 Motivation.....	3
1.2.2 Challenges.....	4
1.3 Goal and Purpose	4
1.3.1 Goals	4
1.3.2 Purpose.....	4
1.4 Solution Overview.....	5
1.5 Report Overviews.....	5
Chapter 2:.....	8
2. Literature Survey	8
2.1 Introduction	8
2.2 Literature Review	8
2.2.1 Data Pre-Processing and Cleaning.....	11
2.2.2 Data Visualization and Analysis	12
2.2.3 Model Building	12
2.3 Summary	12
Chapter 3:.....	14
3. Requirement Analysis.....	14
3.1 Introduction	14
3.1.1 Purpose.....	14
3.1.2 Intended Use	14
3.2 Problem Scenarios.....	14
3.2.1 User Interface.....	14
3.2.2 Hardware Supporting	15
3.3 Functional Requirements.....	15
3.4 Non-Functional Requirements	15

3.4.1	Usability	15
3.4.2	Efficiency	16
3.4.3	Interface	16
3.4.4	Reliability.....	16
3.4.5	Availability	16
3.4.6	Platform constraints:	16
Chapter 4:	18
4.	System Design	18
4.1	Introduction	18
4.1.1	Entering Symptoms.....	18
4.1.2	Disease Prediction.....	18
4.1.3	Generating Report.....	18
4.1.4	Clean Data.....	18
4.1.5	Train and Check models Accuracy	19
4.1.6	Data Analysis	19
4.2	Architectural Design	19
4.3	Detailed Design	20
4.4	Process Flow Diagrams.....	22
4.5	Class Diagram	22
4.6	Use Cases	23
4.6.1	Use Case Description for Web Application.....	24
4.7	Activity Diagram.....	26
4.8	Sequence Diagram.....	27
4.9	State Chart Diagram	27
4.10	Deployment Diagram	28
Chapter 5:	31
5.	Implementation	31
5.1	Overview	31
5.2	Algorithms.....	32
5.2.1	Random Forest Tree.....	32
5.2.2	Gradient Boosting	33

5.2.3	Artificial neural network	35
5.2.4	Accuracy	37
5.3	Flow Control	38
5.3.1	Web Application	38
5.4	Components, Libraries, Web Services and stubs	39
5.4.1	Web Application	39
5.5	Interfaces	40
5.6	Best Practices / Coding Standards.....	44
5.6.1	Code Formatting	44
5.6.2	Naming Conventions	45
5.6.3	Indentation	45
5.6.4	Commenting Standards	45
5.6.5	Deployment Environment.....	45
5.6.6	Summary	45
Chapter 6:	47
6.	Testing and Evaluation	47
6.1	Introduction	47
6.2	Types of Testing.....	47
6.2.1	UNIT TESTING	47
6.2.2	INTEGRATION TESTING.....	48
6.3	SYSTEM TESTING	48
6.4	List of Test Scenarios	48
6.4.1	Test Case Scenarios for Web Application	48
6.5	Performance and Evaluation	50
6.5.1	Accomplishment	50
6.6	Summary	50
Chapter 7:	52
7.	Conclusion and Outlook	52
7.1	Introduction	52
7.2	Achievements and Improvements	52
7.2.1	Achievements.....	52

7.2.2	Improvements	53
7.3	Critical Review.....	53
7.4	Future Recommendations/Outlook	53
7.5	Summary	54
Appendix A.....		54
	Work Breakdown Structure	54
Appendix B		56
	Roles and Responsibility Matrix.....	56

List of Figures

Figure 4.1 Architecture Methodology.....	19
Figure 4.2 Model Architecture.....	20
Figure 4.3 Models Detail Design	21
Figure 4.4 Data Flow Diagram	22
Figure 4.5 Model Class Diagram	23
Figure 4.9 User Use Case.....	23
Figure 4.10 Activity Diagram	26
Figure 4.11 Sequence diagram.....	27
Figure 4.12 Models State Chart Diagram	28
Figure 4.13 Deployment Diagram	28
Figure 5.1 ANN Implementation	36
Figure 5.2 HomePage.....	40
Figure 5.3 Report Generate	41
Figure 5.4 Clean Data	41
Figure 5.5 Models	42
Figure 5.6 Prediction.....	43
Figure 5.7 Visualization.....	44

List of Tables

Table 4.1 UC 01 Visit Website	24
Table 4.2 UC 02 Generate Report.....	24
Table 4.3 UC 03 Clean Data	25
Table 4.4 UC 04 Train Models	25
Table 4.5 UC 05 Visualize Data	25
Table 4.6 UC 06 Prediction	26
Table 5.1 BNN vs ANN.....	36
Table 5.2 Accuracy	37
Table 5.3 Libraries and Framework.....	40
Table 6.1 Test case scenarios.....	49
Table 7.1 Appendix.....	61

LIST OF ABBREIVATIONS

ANN	Artificial neural network
CKD	Chronic Kidney Disease
CKF	Chronic Kidney Failure
CM	Correlation Matrix
CRD	Chronic Renal Disease
ML	Machine learning
RFT	Random Forest Tree
DT	Decision Tree
GB	Gradient Boosting
UCI	University of California Irvine

Chapter 1:

Introduction

Chapter 1:

1. Introduction

Machine learning-based illness prediction is an application that detect the chronic kidney disease based on the data supplied by the user. It may also forecast a patient's or user's ailment depending on the information entered into the system and deliver an accurate result based on that information. Based on the values of the characteristics supplied by the user, we construct a system that predicts the user's chronic kidney illness. This is also a form of healthcare industry help to inform the patient and also useful for the patient if the patient wants to go to a hospital or not. The user can find out if you have the disease or not using this app. If someone is currently suffering from an illness, they should visit a doctor, which is time-consuming and expensive. It can also be a challenge for the user if they are away from doctors and hospitals because the condition is not available. Therefore, if the above procedure is performed using an automated application that saves time and money. It may be easier for the patient, making the procedure more efficient. There are a few additional kidney-related disease predictive systems that use data mining to assess a patient's level of risk. CKD is a web-based program that protects the user's kidney disease based on the features provided by the user. It contains data sets gathered from the Kaggle and UCI ML repository. With the aid of illness predict, the user able to determine the likelihood of a disease based on the feature values provided.

1.1 Background

The kidney is an important organ in our body. Urine is formed by the removal of all lavish from us. CKD also known as Chronic renal disease (CRD) or Chronic kidney failure (CKF), is an existence alarming condition caused by the kidneys' inability to perform their functions properly. It is soon becoming one of the most crucial aspects of public health concerns in world. In this form of illness, the Glomerular Filtration Rate (GFR) steadily drops for more than three months [1]. It is characterized as a silent killer since this disease has no physical signs in the beginning days of the disease. CKD is an intensifying disease that develops over time and eventually quit operating. This disease

has become one of the world's most critical public health problems. It's a sneaky disease that's usually only diagnosed after it's too late due to test abnormalities. If CKD has proceeded to this stage, the only options for saving a patient's life are dialysis or a kidney transplant. On the other hand, early detection can help prevent renal failure. The most effective approach to evaluate monitoring the GFR on a systematic basis can help forecast working of kidney and level of the disease. The GFR is measured using the individual's age, sex, race, and blood creatinine level. In 2016, 753 million people worldwide were diagnosed with CKD, including 417 million women and 336 million men [2]. Each year, approximately one Million of people in 112 impoverished nations died because of kidney disease as a result of a lack of financial resources to pay for dialysis or kidney transplantation [3]. Our study attempts to predict chronic kidney disease in patients by analyzing patients' data and using machine learning algorithms to determine whether they have chronic kidney disease or not. Because we have so much data in today's environment, we can use to assess a different type of machine learning and deep learning methods it and extract hidden patterns or insights. These insights may then be used for medical diagnosis.

1.2 Motivations and Challenges

1.2.1 Motivation

Machine learning algorithms and techniques have been used to analyses and evaluate data from a range of data science applications and applications over a long period of time. The primary goal of this research was to look at a number of data attributes and options in pre-data processing, such as data purification also known as data cleaning and training model using various machine learning algorithms. We chose this project to learn about data science and machine learning techniques because we can apply such approaches to the data and construct a model that can predict whether or not a patient has chronic kidney disease based on the features he would provide. We investigate several machine learning and data science methodologies before using a classification and deep learning model to train the data we collected. Using these techniques, we created a web-based

application that uses machine learning and deep learning to determine if a patient has chronic kidney disease or not based on some features.

1.2.2 Challenges

Renal illness is often clinically hidden until it has progressed to an advanced level, and diabetes, hypertension, and cardiovascular disease are the most ordinary foundation of CKD. The biggest challenge in this project is predicting the illness based on the patient's characteristics. There are several medical tests that can diagnose the condition, but they are quite costly. The fatality rate and overall effects can be reduced if the condition is detected early. Because we have so much data in today's environment, to analyse information and uncover hidden patterns or insights, we may utilize a variety of machine learning and deep learning methods. These insights may then be used for medical diagnosis.

1.3 Goal and Purpose

1.3.1 Goals

Chronic Kidney Disease is often clinically hidden and also known as silent killer until it has progressed to an advanced level. Medical tests to anticipate the illness are available, but they are highly costly. Our major objective is to create a web-based program in which the user inputs information such as the number of red blood cells, appetite, etc., and the system determines whether or not the user has chronic kidney disease based on those inputs. Early detection of CKD can aid with therapy and avoid expensive treatments like dialysis and transplants.

1.3.2 Purpose

The following are the major objectives of this project's development:

- Our main objective is to develop a web app that predict the based on the features provided by the user.
- This project predicts the kidney disease of the patient based on the features provided by the patient.

- The disease is predicted using algorithms, and in order to obtain the result, the user must input the values of the attributes or variables given in the menu.
- Our primary goal is to use various data science and machine learning approaches.
- This app can also be used to generate a detail report about the data and clean the raw data into a clean format and visualize the data to get more information about the data.
- We'll employ a variety of machine learning methods, with the most accurate algorithm being selected for model training and model development.

1.4 Solution Overview

CKD also known as Chronic Renal Disease (CRD) or Chronic Kidney Failure (CKF), is an existence alarming condition caused by the kidneys' inability to perform their functions properly. We aimed to develop a model that could predict based on chronic kidney disease data provided by users. Our Project based on three major parts: Pre-processing, Exploratory Data Analysis and Prediction/model training. As a result, we'll begin by gathering data from the Kaggle and UCI repository. This dataset has 700 plus rows and 26 columns. In the dataset, there are some missing variables as well as some needless values. We clean up the data set first by removing any foreign values, and then we fill in the missing values with the median value. The model is then trained machine learning and deep learning. We employ a range of machine learning models, such as the random forest tree, gradient boosting, and artificial neural network (ANN). We develop a web-app where the user can input the features and then model predict whether the user has the chronic kidney disease or not. The user enters the features, and the model predict whether or not the person has chronic kidney disease based on those features.

1.5 Report Overviews

This report is divided into different chapters to explain the process and methodologies applied to carry out this project. It includes market surveys in which we take into account all similar apps. The report also consists of the requirement analysis phase, the design

phase, the implementation phase and the testing phase which shows a complete picture of the application we have made.

Chapter 2:

Literature Survey

Chapter 2:

2. Literature Survey

2.1 Introduction

It is an existence alarming condition caused by the kidneys' inability to perform their functions properly. It is soon becoming one of the most important public health concerns in the globe. In this form of illness, the Glomerular Filtration Rate (GFR) steadily drops for more than three months. It is characterized as a silent killer since there are no physical symptoms in early days of the disease. This disease is a slowly intensifying disease that develops over time and eventually quit operating. This disease has become one of the world's most critical public health problems. It's a sneaky disease that's usually only diagnosed after it's too late due to test abnormalities. If CKD has proceeded to this stage, the only options for saving a patient's life are dialysis or a kidney transplant. On the other hand, early detection can help prevent renal failure. The most effective approach to evaluate monitoring of GFR on a customary basis that can help forecast the function and stages of this disease. It's detection in early levels can be reduced with therapy and avoid expensive treatments like dialysis and transplants. It's feasible for the examiner of data and other information on patients using machine learning techniques in order to diagnose CKD early [4]. CKD has recently identified as a major people's health concern. Each year, a lot of people lost their lives as a result of poor healthcare, and health education is lacking [5] and the exorbitant expense of CKD treatment. Globally, 13.4% of the population is predicted to be afflicted by kidney disease, according to worldwide data about kidney illnesses [6].

2.2 Literature Review

Different tests and studies have been done in recent years as medical research and machine learning have advanced, culminating in the release of significant major articles. Many research investigates and assess chronic illnesses utilizing a variety of early detection strategies.

- This section contains evaluations of a variety of technical and review publications for predicting kidney disease. Among the data mining algorithms tested for detecting accuracy are Logistic-Regression, multilayer perception, ANN, decision table, radical basic function, naive Bayes, k-nearest neighbor, and sequential minimum optimization. The level of accuracy of such procedures varies depending on the kind of dataset, and there is no one guideline for the optimal outcome [7].
- Using classification methods, look into renal function failure. They use random forest, radial basic function, and back-propagation neural networks to categorise cases into different phases of renal disease based on case severity. They test multiple strategies for performance measures such as specificity, kappa, sensitivity, and accuracy, using a dataset from the state of Coimbatore with roughly 1000 patients and 15 variables, and find that the radical basic function is the best classifier with 85.3 percent detection accuracy [8].
- Having a dataset two classes with 400 cases, and having 24 variables. Classifiers for CKD detection include support vector machine (SVM), logistic regression, and k-nearest neighbor (KNN). They utilize the machine learning repositories for data of chronic kidney disease. The findings suggest that the SVM approach is the most accurate and sensitive detection technique [9].
- ANN, Decision Trees, and Logistic regression are utilized to analyze performance of ANN, Decision Trees, and Logistic Regression for Kidney treatment “survivability”. The accuracy measurements used to evaluate the data mining approaches were classification accuracy, sensitivity, and specificity. For each approach, they used 10-fold cross-validation and a confusion matrix to get results. They discovered that ANN produces superior outcomes. As a consequence, ANN displays tangible findings from patient records including kidney dialysis [10].
- The study compares the accuracy, time to create the model, and size of the training data set of different neural networks [11].
- Supervised approaches to forecast the early likelihood of AVF failure in patients, as outlined in their research. They employed categorization algorithms to predict

the likelihood of complications in new hemodialysis patients recommended to AVF surgery [12].

- To discriminate between normal and renal disease patients, apply approaches for texture analysis to examine pictures dataset of the disease of the kidney. They utilize MATLAB to determine “the value of the root mean square (RMS), the value of homogeneity, the values of average, and the values of a correlation matrix (CM)” using mathematical techniques such as Fourier analysis. They used Risk Management Solutions and cortical-region-values of 0.3 and 0.0049 to discriminate between normal and renal disease patients in pictures dataset of 32 individuals [13].
- Control CKD with a suitable diet plan and, using their categorization system, offer diet programs to various individuals. They suggest a diet depending on the patient's potassium levels in the blood. They employ different multicast algorithms method to achieve ninety nine percent accuracy [14].
- Using the data from the machine learning repository including values on roughly “400 individuals”, use hybrid classification algorithms to examine renal illness. They use a support vector machine and a k nearest neighbors’ classifier to choose the dataset's most important characteristic, then use the relief and gain ratio approach to select it. They get to the conclusion that the k nearest neighbor method over performs than other algorithms on the basis of “f-measure, precision, and the contrast matrix” for certain characteristics [15].
- On a dataset with 400 records and 24 characteristics, 12 different classification algorithms were evaluated. They calculated the accuracy of prediction findings by comparing their estimated results to real results. Precision, responsiveness, and uniqueness were utilized as assessment criteria. The decision tree approach gives us result of up to 98.6%, responsiveness of 0.9720, precision of 1 and uniqueness of 1 [19].
- When the data is being trained, utilizing the creation of an algorithm using neural-network for making prediction on CKD, we look for the influence of class imbalance. Comparison research was conducted utilizing a sampling algorithm in this suggested work [20].

Machine learning, based on previous studies, can help to categorize data into different categories and provide important details to the data. The findings suggest that when machine learning methods are combined with character selection methods, they may provide reliable classification results. This work uses a set of the most common methods of machine learning by combining a selection and patients with common renal disease should be identified using this technique, and maintain benefits of differentiating effects of machine learning strategies.

So, we found out that there are three basic steps to build this in the model using data science techniques. There are many ML algorithms that was tried before to predict this disease. So first we need to Preprocess the data and clean the data. Then we have to visualize or analyze the data. And the model is build using different algorithms and the algorithm which give us the best accuracy and is implemented in this project.

2.2.1 Data Pre-Processing and Cleaning

You can gather the dataset from Kaggle or UCI machine learning repository it utilized in this project. So, after gathering the data, we must do a few steps to convert it into a usable format, as the data is in raw form and contains numerous missing values and null values. So, first and foremost, we must complete the data preprocessing and cleaning procedure.

2.2.1.1 Data Pre-Processing

As a result, the initial stage in constructing a machine learning model is data processing. Typically, data is erroneous and inconsistent at first, and it lacks numerous aspects. So, the first step is to obtain the data set from which the model is built. You may obtain the data set from a variety of sources, including Kaggle and UCI machine learning repository. Different Python APIs can also be used to collect data. Following the data collection, we must load several libraries based on our requirements. The essential libraries are NumPy, Pandas, and Matplotlib.

2.2.1.2 Data Cleaning

So, after pre-processing the data, it's time to clean it up. It eliminates any useless and insignificant values from the data and transform it into a clean form during data cleaning. As a result, it first checks for duplicate values. If there are any duplicate values or

columns, they must be removed. The following step is to check for any missing or null values. If the values are numeric, the median of the column values is used to fill in for the missing values.

2.2.2 Data Visualization and Analysis

In data analysis, we apply different logical and Statistical Techniques to show, describe and visualize the data. We use different graphs to show the relation between the values of the data. As they said your visualization of the data is a bridge that removes the distance between the numbers and the words. Seaborn and Matplotlib are the two most famous libraries that are used for the visualization of the data. Data visualization is commonly used to get access to the inner workings of a system and to examine how values relate to one another.

2.2.3 Model Building

Following data cleansing and data analysis, a machine learning system's next step is to develop a model. Scikit-Learn is the famous library used for ML model building. To populate the model, many approaches may be utilized. It uses three algorithms, and the one with the best accuracy and is integrated into the system. This model is built using a random forest tree, gradient boosting, and a neural-network ANN.

2.3 Summary

The value of our proposed Final year project in the market was studied in this section. After reading the literature, which provided us with good basis for developing this strategy. It covers the processes and mechanisms that went into making this model. We'll go into the reasoning behind choosing a certain technology for this project in more detail.

Chapter 3:

Requirement Analysis

Chapter 3:

3. Requirement Analysis

3.1 Introduction

The requirement of the project is analyzed briefly. It does how the system works and how the function is performed in this application. It also analyses the scope of the project. This covers the effort done to discover the needs of various users in order to design better and more accessible software. Both functional and non-functional requirements are critical. Additional information is provided below.

3.1.1 Purpose

The goal of our CKD prediction project is to determine whether or not the user has the condition. The user enter values for a few characteristics, and the machine forecast the sickness.

3.1.2 Intended Use

Based on the features that the user enters, this program can only determine if the user has this condition or not. The user can not only predict the disease but the user can also perform other actions like clean the dataset, generate a detail report about the dataset, analyze the data and train different models.

3.2 Problem Scenarios

The problem is address by developing a web application based on machine learning model that predicts the kidney illness depends on the information supplied by the user.

3.2.1 User Interface

In the web application form, a user interface should be provided so that the user can immediately understand how to traverse the application and its features. The features are then provided and based on those features; the chronic renal disease can be predicted.

3.2.2 Hardware Supporting

This application not necessitate the use of any hardware. However, to evaluate the application, the user have a computer and an internet connection. To predict the disease based on such data, the user needs also to know the feature values, such as the number of red blood cells.

3.3 Functional Requirements

Following are some functional requirements that should meet:

- The user shall be able tom upload and clean the data set.
- The user shall be able to generate a detailed report about the data set.
- The user shall be able to analyze the data using different type of charts.
- The user shall be able to train three different models and check their accuracy.
- The user shall be able to predict the disease bye by giving the features value.
- The user shall have the data set to perform all these actions.

3.4 Non-Functional Requirements

The term "non-functional requirements" referred to the system's constraint or restriction. For the selection of language platform implementation techniques and tools, they may be related to emergent system features such as reliability reaction time and storage openings. Some of them are given below.

3.4.1 Usability

Usability is the essential cause for the success of the system.[16] That means the system can be used by many users efficiently and effectively at the same time. The user can easily understand our application. There are not any Complex features that affect the User experience. Our application is providing a friendly user interface to increase the usability of the system.

3.4.2 Efficiency

The system should be efficient enough to clean, train, predict, analyze and generate report about the data. When the user provides the feature values, the system must be efficient enough to provide the appropriate output.

3.4.3 Interface

The user interface should be simple to use and adaptable to a variety of devices. The software may be compressed to fit different screen sizes and provide a better user experience.

3.4.4 Reliability

The model must be trustworthy to both the user and the system expectation can handle easily and error grade then it is reliable for the user.[17] The user should be able to get the intended result without interacting too much with the software.

3.4.5 Availability

When a user wants to utilize a feature, it must be available to them. And all the ride had their vehicles and proper mapping to all the address.[18]

3.4.6 Platform constraints:

The important thing is to train the model and predict the disease based on the feature values provided by the user. For easy approach to the system the web application should be online.

Chapter 4:

System Design

Chapter 4:

4. System Design

4.1 Introduction

These objectives are a set of concepts that utilized in our system to predict renal illness using machine learning. Data flow diagrams were used in the creation of this system. Diagrams of sequences Class diagrams, component diagrams, activity diagrams, status chart diagrams, and deployment diagrams are all examples of class diagrams. We completed our job after finishing the diagrams. The following are the functions that the system is capable of.

4.1.1 Entering Symptoms

After the user has been properly loaded into the system, he or she must offer some of the features listed on the menu.

4.1.2 Disease Prediction

The model predicts the disease of the person that he might have chronic kidney disease or not.

4.1.3 Generating Report

The user can generate a detail report about the data to get more information about the data. The user upload the file and the system generates a detail report about it. The user can also download that report.

4.1.4 Clean Data

The user can also clean the dataset using this web app. The user needs to upload the raw data file of CKD disease and the system clean the data. The data cleaning includes replacing null values, changing wrong data types, replacing categorical values etc. The user can then download the clean form of the data.

4.1.5 Train and Check models Accuracy

The user can upload the clean data set and can train and check accuracy of the algorithms. There are three algorithms Random Forest tree, Gradient boosting and ANN. The user can train these models and can check the accuracy.

4.1.6 Data Analysis

The user can also analyze the data. The user can select a feature and see different type of charts/graphs to analyze the data and get more info about the data.

4.2 Architectural Design

Let's talk about the system flow before we talk about the software design model, we utilized in this software system. When the user first opens the program, the user have to select that which task he/she wants to perform.

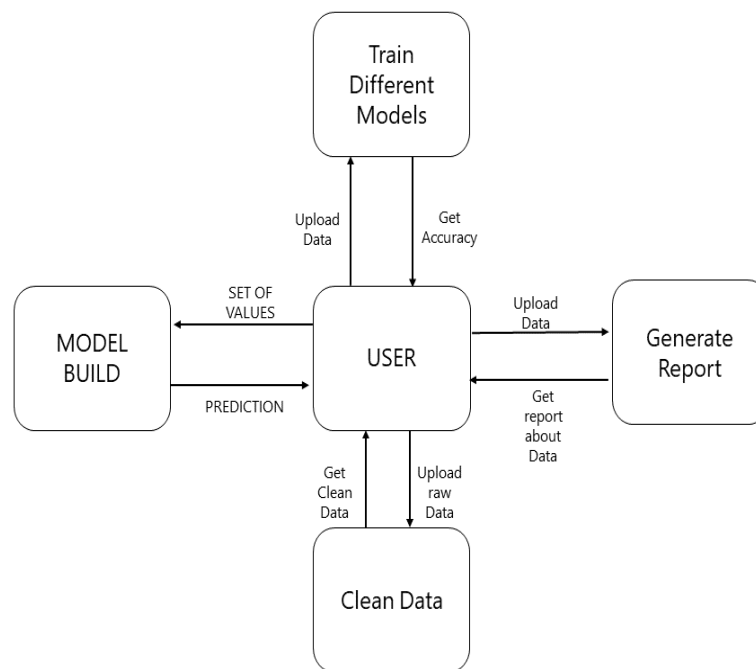


Figure 4.1: Architecture Methodology

To forecast the disease, the user must enter the values of the features that have been requested, and our algorithm predicts the disease based on those features. Architectural styles are used to provide a framework for all of the system's components.

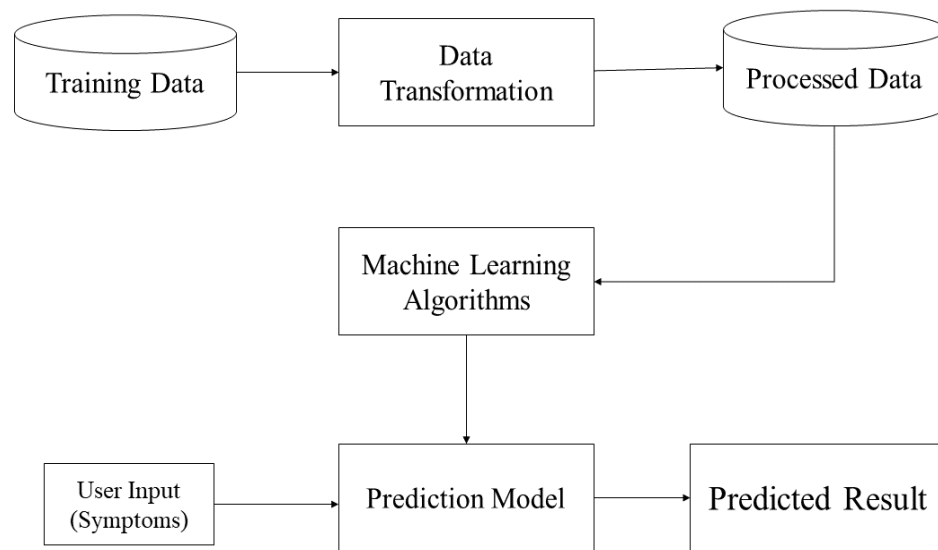


Figure 4.2: Model Architecture

4.3 Detailed Design

This portion predefined all the required details to begin a project. To begin our project, we'll need a fundamental pattern. Such as the design of a building, which a builder would use to construct a structure according to specifications. We'll go through the basics of our project's structure. We established basic UML diagrams and ER diagrams for our project in this section. This section predefined all the necessary detail to start project. We need a basic pattern to start our project like architecture of a building that a builder used to build a building according to requirement.

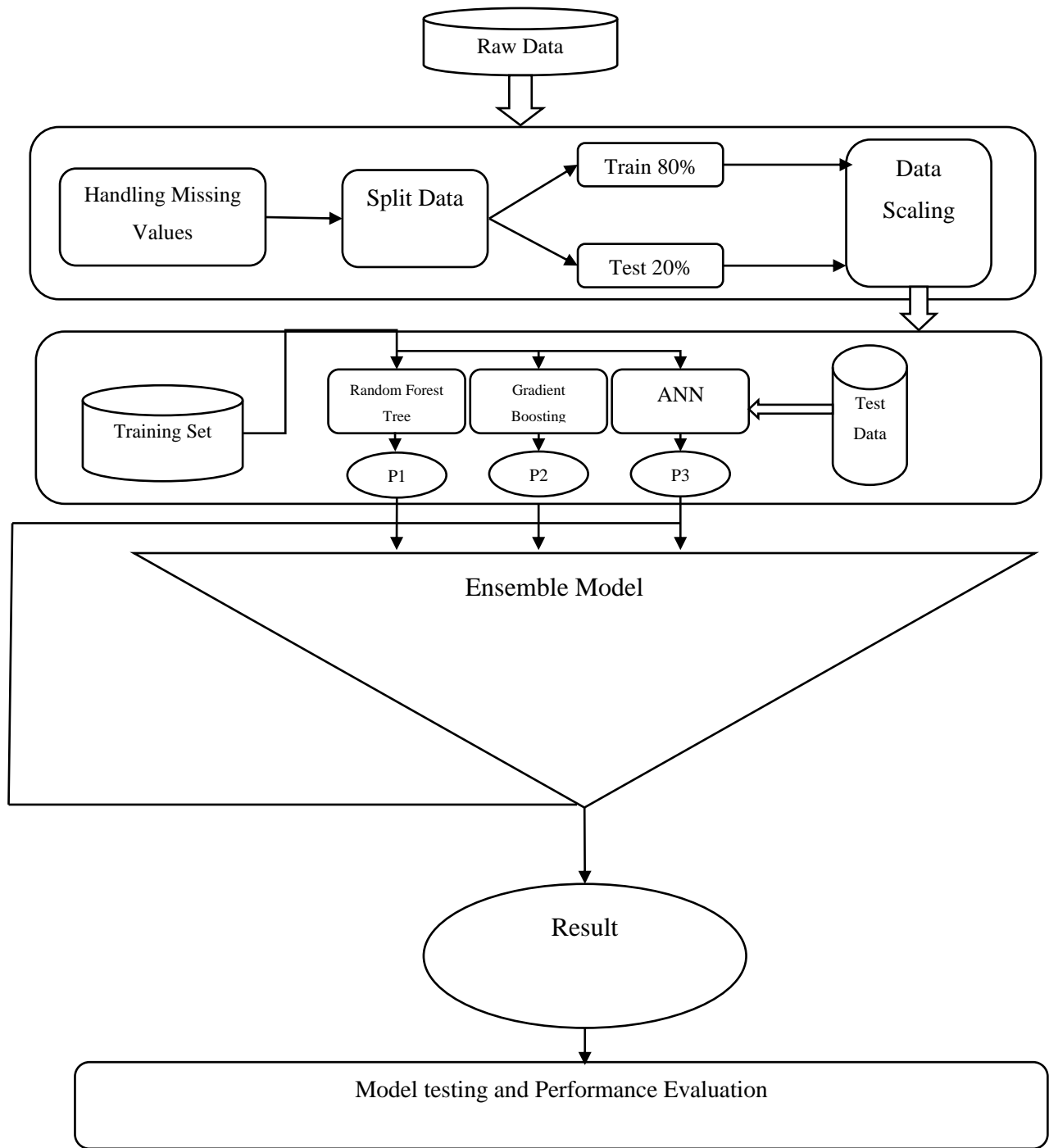


Figure 4.3: Models Detail Design

4.4 Process Flow Diagrams

Most of the elements required for a conventional flow diagram are included in the data flow.

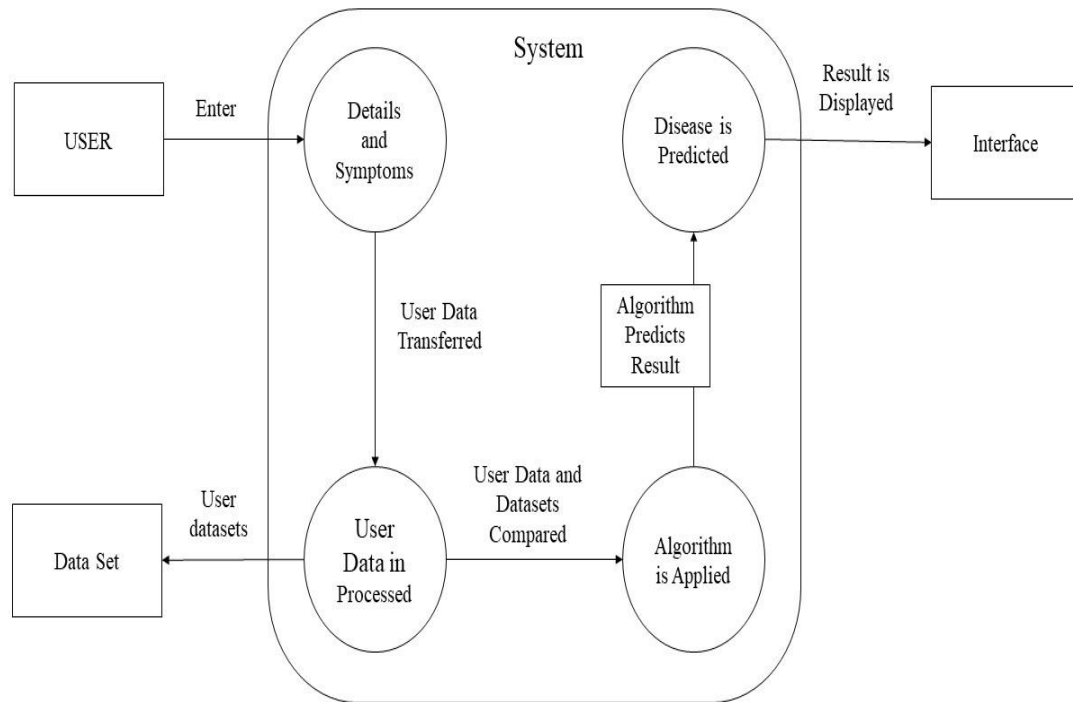


Figure 4.4: Data Flow Diagram

This data flow diagram shows how the model transitions from one state to the next as the user enters all of the general information, as well as the symptoms that are entered into the system, compares to the prediction model, and if true, predicts the appropriate result; otherwise, it shows the detail of where the user went wrong while entering the information.

4.5 Class Diagram

CKD prediction model consists of the same Class-diagram as all the other application that consists of the basic Class diagram, here the Class diagram is the basic entity that is required to carry on with the project. A class diagram has the details about other classes which handle the related data and all of the other necessary properties, as well as their

interactions with other entities, are required to use the concept of this model. The user has to enter all the necessary information to predict the disease.

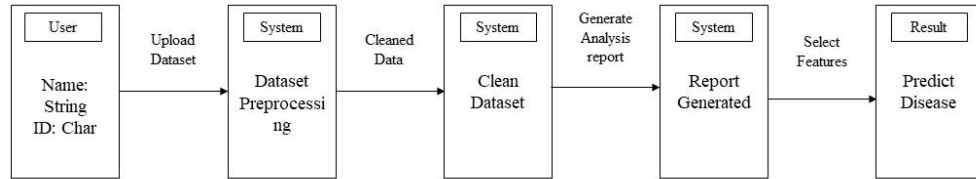


Figure 4.5: Model Class Diagram

4.6 Use Cases

The use case diagram for the machine learning project consists most of the elements that a typical this type diagram requires. This use case diagram depicts how the model progresses from start to finish. From one step to the next, he inputs all the details and some other information, and also the symptoms, in the system, which the system compares to our model and, if true, protects the appropriate reserves at the device. The system's use case diagram is shown below.

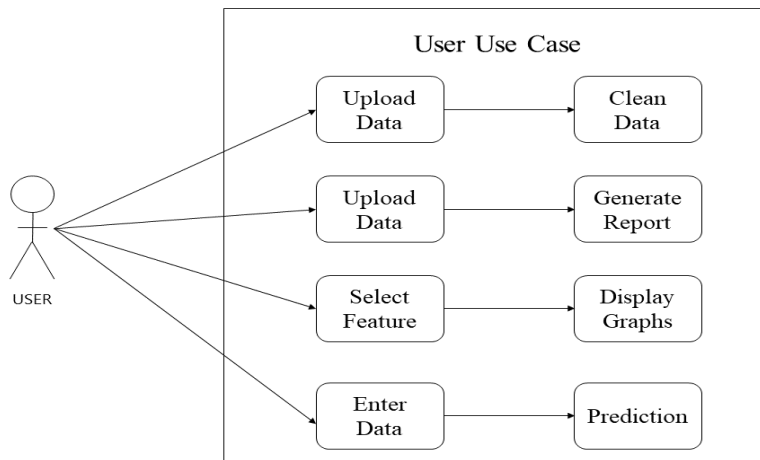


Figure 4.6: User Use Case

It also shows the detail that while inputting the feature values where the user goes wrong, as well as the appropriate equation measure for the user to follow.

4.6.1 Use Case Description for Web Application

4.6.1.1 UC 01 Visit Website

Use Case ID:	UC01
Use Case:	Visit the website
Description:	Open the web browser and search for the web link: https://tayyab885-chronic-kidney-disease-prediction-app-s4cofm.streamlitapp.com/
Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	Select Activity
Main Success Scenario:	Website should have been opened

Table 4.1: UC 01 Visit Website

4.6.1.2 UC 02 Generate Report

Use Case ID:	UC02
Use Case:	Generate Report
Description:	Upload the dataset
Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	NO
Main Success Scenario:	Report should have been generated

Table 4.2: UC 02 Generate Report

4.6.1.3 UC 03 Clean Data

Use Case ID:	UC03
Use Case:	Clean Dataset
Description:	Upload the raw dataset

Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	NO
Main Success Scenario:	Clean Data

Table 4.3: UC 03 Clean Data

4.6.1.4 UC 04 Train Models

Use Case ID:	UC04
Use Case:	Train Models
Description:	Upload the clean dataset
Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	NO
Main Success Scenario:	Train Models and check their Accuracy

Table 4.4: UC 04 Train Models

4.6.1.5 UC 05 Visualize Data

Use Case ID:	UC05
Use Case:	Visualize Data
Description:	Select the Feature
Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	NO
Main Success Scenario:	Graph should have been displayed.

Table 4.5: UC 05 Visualize Data

4.6.1.6 UC 06 Prediction

Use Case ID:	UC06
--------------	------

Use Case:	Chronic Kidney Disease Prediction
Description:	Input the feature values
Actor:	Doctor/Patient
Precondition:	Web Application should be open
Postcondition:	NO
Main Success Scenario:	Disease Predicted

Table 4.6: UC 06 Prediction

4.7 Activity Diagram

There's another diagram in the UML to describe the vital features of this model is the function representation. This diagram is actually a flow chart that shows the move of data from one point to other. The action is system performance. From action to action, a flow of control is expressed.

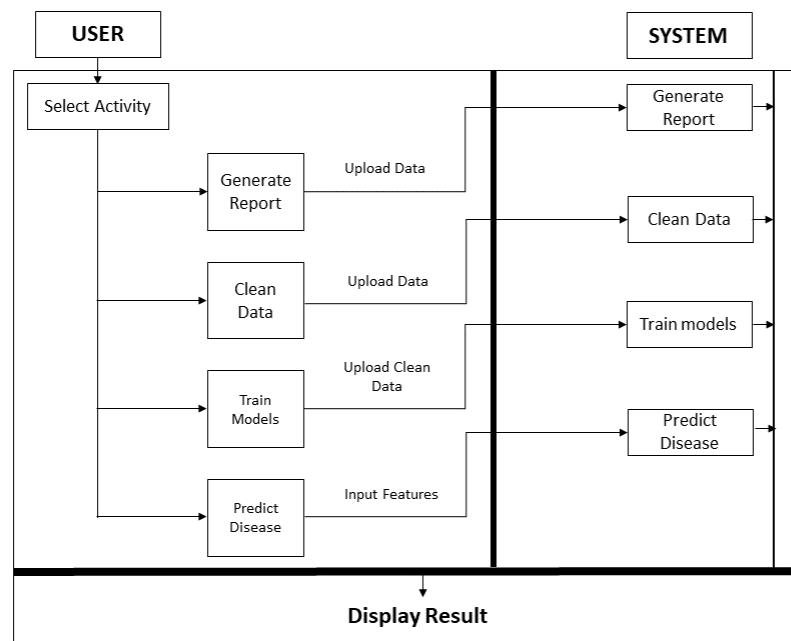


Figure 4.7: Activity Diagram

The action in this calculation starts with the user. The user can generate our report by uploading the data set and can clean the data set and you the user can upload the clean dataset and train the model using different algorithms to check their accuracy and in the

end the user can also visualize the data by selecting the features from the data and can also predict the disease by inputting the features value. Finally, once the data from the data set has been processed, an analysis is taken place, and the appropriate result is displayed on the interface.

4.8 Sequence Diagram

This diagram for the project kidney disease prediction using ML have all the elements that a quality sequence diagram would have. The sequence diagram explains how it works from one state to the next state when the user logged into the application, then inputs the necessary details and some other data, as well as the values of the features, which are compared to the prediction model, and if true, the appropriate result is predicted. The model predicts the outcomes and show it on user interface.

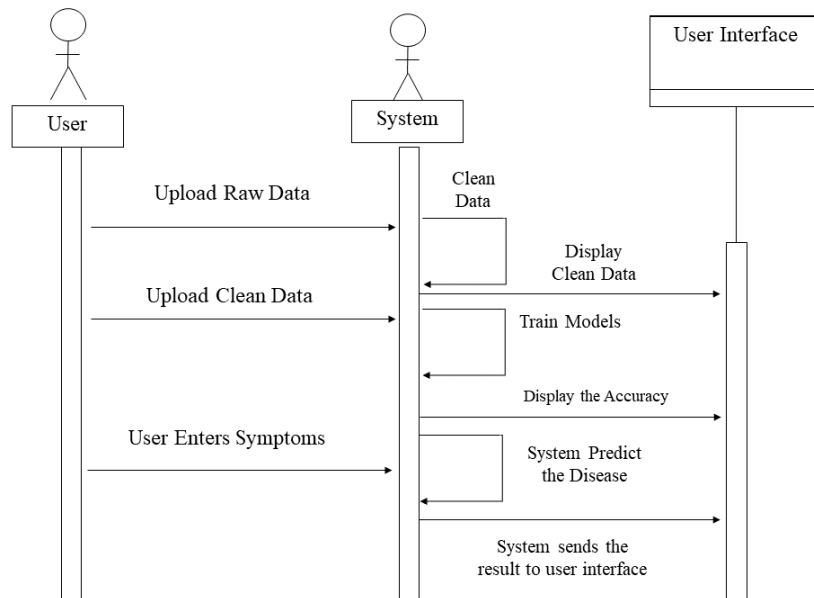


Figure 4.8: Sequence diagram

4.9 State Chart Diagram

In this diagram models the flow from one state to the state of a specific object in the system. So, the user can generate a report about the dataset, clean the dataset, visualize the data, train and check accuracy of different algorithms and have to input some features

for symptoms and based on the symptoms the model works. The model takes our decisions whether the user has chronic kidney disease or not and show the result on the interface.

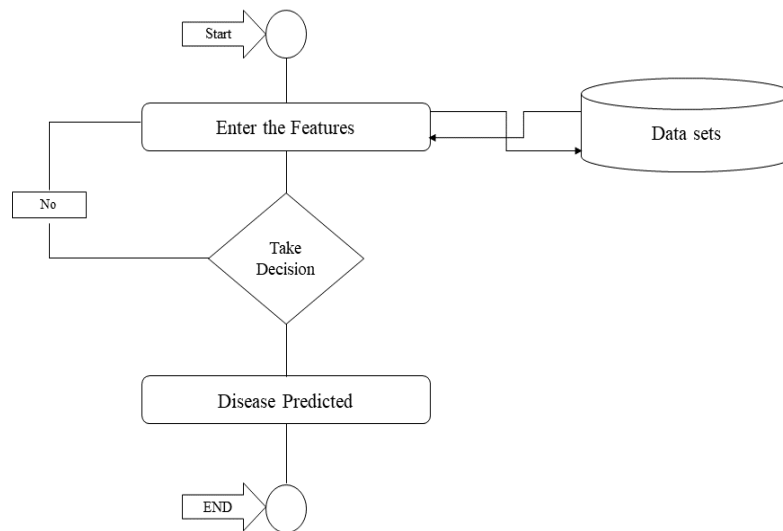


Figure 4.9: Models State Chart Diagram

4.10 Deployment Diagram

The deployment diagram depicts how runtime processing nodes and the components that reside on them are configured.

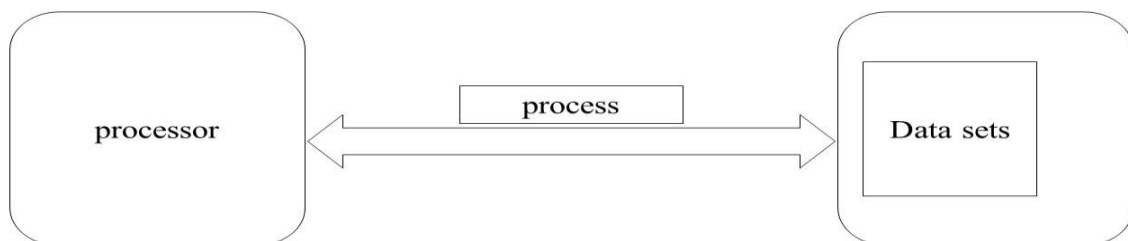


Figure 4.10: Deployment Diagram

It's a type of structural diagram that depicts the system's physical features. Here the deployment diagram depicts the project's ultimate step, as well as how the system changes after change in the machine. Starting with the system's processing of the user's input data, it compares the data with the help of data sets, then trains and tests the data using various machine learning techniques. The result is then displayed on the user's display or interface.

Chapter 5: Implementation

Chapter 5:

5. Implementation

5.1 Overview

The project Chronic Kidney Disease Prediction using ML and deep learning is created to solve basic illness in early stages. The project is done with the help of Python completely. Even the interface of the project is done using Python library Streamlit. Here the user first see the homepage of the web app. The home page contains the general information about the application and the disease. We develop a simple and user-friendly app so that anyone can access and use it. The user can access and use multiple sections of the web application like generate reports, clean data, data analysis, models and prediction. Let me explain the complete implementation and working of the project below.

- On the homepage of the app, the user gives some basic information about the project as well as chronic kidney disease.
- The user can access and use multiple sections of the web application like generate reports, clean data, data analysis, models and prediction.
- The user can submit/upload data and get a detailed report about the data set in the generate report section. Missing or null data, a range of values, and a chart/visual representation of each column are all included in the report.
- The user can also download the report by clicking the download button.
- The user can upload and clean the data set in the next stage. The user can upload the raw data set and receive it in a clean format. It cleans the dataset by removing all missing values and replacing them with the column's median value. The model can then be trained or tested using a clean dataset.
- The user can also download the clean format of the dataset by clicking the download button.

- The most critical element of every machine learning project is cleaning the data set. After cleaning the data set, the user can proceed to the following stage, where they can submit the cleaned data set file and train, test and get accuracy of three different algorithms.
- To analyse the data and gain understanding from it, the user can see many sorts of graphs such as a bar chart, a pie chart, a heatmap, and so on.
- The user can predict the disease. To protect the disease the user has to enter different symptoms values and based on those values, the model with highest accuracy predicts whether or not the user has that disease.

5.2 Algorithms

In this project 3 different algorithms are used. From which two are machine-learning classification algorithms named as Random-Forest-Tree and Gradient-Boosting and one is a deep learning neural network known as Artificial neural network (ANN).

5.2.1 Random Forest Tree

This is a supervised learning environment. A machine learning phase that incorporates push-ups to increase the efficiency of the Decision Tree. It includes tree predictions, and the tree relies on a randomly selected vector. All trees have the same distribution. Instead of dividing nodes based on flexibility, Random Forest distributes them using the best use among a group of predictions taken randomly from the node itself. The worst time complexity for a random forest is $O(Mn \log n)$, where M is the number of trees growing, n is the number of occurrences, and d is the size of the data. It's a straightforward classifier that employs a variety of decision tree models. In machine learning, it may be used for both regression and classification. It's a simple algorithm to utilize. Forest is made by the trees. A forest is thought to be stronger the more trees there are.

For categorization and establishing the final class in each tree, it employs the Gini index. To create the final classifier, the final class of each tree is aggregated and voted on by the weighted values. A random forest is a system that combines the number of decision trees in different sub-sets of a data set and measures the results in order to increase the predicted accuracy of the data set. Instead of relying on a single decision-making tree, the

random forest monitors predictions from each tree and predicts the final outcome based on multiple predictable votes.

For our model this algorithm gives us about 100 percent accuracy.

5.2.1.1 Steps

For its implementation there are four steps:

- From the dataset it chooses random samples.
- Create a DT for every sample and then receive the result for each of them.
- For every expected result make a decision.
- As the for the last result, choose result with most votes.

5.2.1.2 Advantages

- It can be used to perform the classification and regression tasks.
- Can handle large, high-dimensional datasets.
- It makes the performance of the model increase and decrease the overfitting.

5.2.1.3 Disadvantages

- The big disadvantage of this algorithm is that it might become too lethargic and unserviceable for instantaneous calculations if it has a lot of trees. In wide-ranging, these systems are quick to learn yet take a long time to make predictions after they've been taught.
- Despite the fact that Random Forest may be used for both classification and regression problems, it is not better suited to regression.

5.2.2 Gradient Boosting

Gradient boost is a well-known way to improve. Each prediction in the growth gradient corrects the error of the previous forecast. Unlike Adaboost, the weight of the training model has not changed; instead, each prediction is trained using errors before predicting as a label. CART is an important student in a form known as the Gradient Development Tree. The gradient magnification method can be used to predict not only continuous but also target variable variables (such as Classifier). The Mean Square Error (MSE) is a costly function when used as a router, while a Loss Loss is a cost function when used as a

separator. Gradient-enhancing dividers are a collection of machine learning algorithms that integrate multiple weak learning models to produce a powerful speculative model. When made to increase gradient, pruning trees are often used. Gradient-enhancing models are gaining popularity due to their ability to differentiate complex data sets, and have recently won a number of Kaggle data science challenges. Scikit-Learn, a Python machine learning package, includes a variety of implementations to improve class variability. Providing a feature of machine learning algorithm and making the system separate conditions / data points in one of several different classes is called segregation. Because classes are classified by nature, the model cannot be labeled as part of a class and part of another. It gives us an accuracy of 99 percent.

5.2.2.1 Steps

In order to create a gradient boosting classifier, we'll need to go through a few phases. We'll need to do the following:

- Complement the model.
- Parameters and Hyperparameters of the model should be fine-tuned.
- Make forecasts.
- Analyze the outcomes.

5.2.2.2 Advantages

- Frequently gives exceptional forecasting accuracy.
- Lots of versatility - can optimize on a variety of loss functions and includes various hyper parameter tweaking possibilities, making the function fit extremely adaptable.
- There is no need to pre-process data; it typically works well with category and numerical values as is.
- Imputation is not necessary to handle missing data.

5.2.2.3 Disadvantages

- Boosting the gradient models continue to be improved in order to reduce any faults. This can lead to overfitting by exaggerating outliers.

- Computationally intensive - frequently necessitates a large number of trees (>1000), which can be time and memory consuming.
- Because of the approach's considerable flexibility, there are a lot of variables that interact and have a big impact on how it behaves (number of iterations, tree depth, regularization parameters, etc.). During tuning, this necessitates a huge grid search.
- Less interpretive in nature, yet this can be easily remedied using a variety of methods.

5.2.3 Artificial neural network

The term "artificial neural network" refers to the sub-field of artificial intelligence that is influenced by biology and is shaped like the brain. A computer network based on the biological sensory networks that make up the human brain structure is known as the artificial neural network. Synthetic sensory networks, such as the human brain, deploy neurons connected to each other at various levels of networks. These neurons are called nodes. Synthetic sensory networks, such as the human brain, attach neurons to each other at various levels of networks. Nodes are the name of these senses. The human brain contains about 1,000 billion neurons. Each neuron contains a number of connections from 1,000 to 100,000. Data is deposited in the mind of the human in such a way that it can be conveyed, and it can extract many types of data from mind at one time if needed. The mind of the person, we might say, is completed up of marvelous corresponding mainframes. In nerve networks, dendrites from biological neural networks represent input, cell nuclei represent nodes, synapses represent weights, and axons represent output. Computers are designed to act as brain cells connected to a sensory network. The standard Artificial Neural Network resembles the illustration below. In our model we use two hidden layers for the implementation of ANN. We train it for 200 epochs and it gives us an average accuracy of 90 percent.

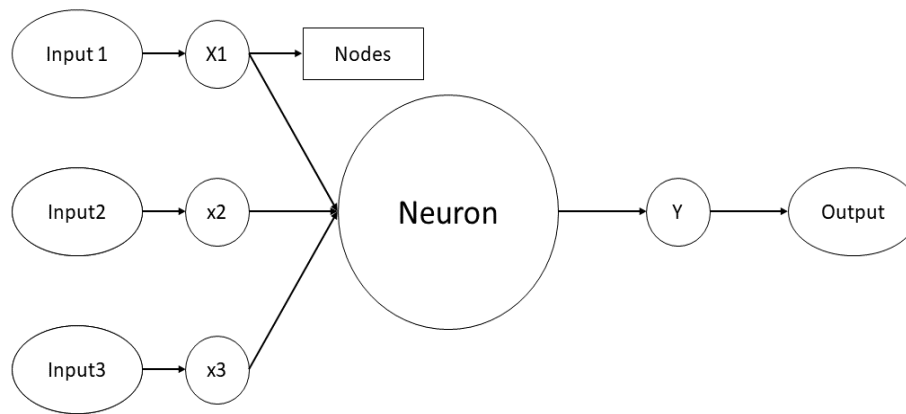


Figure 5.1: ANN Implementation

The connection among a genetic and a non-natural neural network is as follows:

Biological Neural Network	Artificial Neural Network
Dendrites	Inputs
Cell Nucleus	Nodes
Synapse	Weights
Axon	Output

Table 5.1: BNN vs ANN

ANN model consists of three different layers.

- The Input Layer
- The Hidden Layers
- The Output Layer

5.2.3.1 Steps

- In the first stage, input units are transferred to the hidden layer, i.e., data with certain weights attached.

- Neurons make up each buried layer. All the neurons get the independent values.
- All calculation is done in the hidden layer after the inputs have been passed on.
- Each concealed layer goes through the whole procedure stated in point 3. We move to the end layer which is our output layer, after passing through all of the hidden layers. This is where we get our result.
- The error is determined after collecting the predictions from the output layer, which is the difference between the actual and expected output.

5.2.3.2 Advantages

- Parallel processing is possible.
- Data storage over the whole network.
- Capacity to work with a limited amount of information.
- Having a memory distribution is a good thing to have.
- Having the ability to tolerate mistakes.

5.2.3.3 Disadvantages

- Ensure that the network structure is correct.
- The network's unnoticed activity.
- Hardware is required.
- Difficulty in communicating the problem to the network.
- The network's duration is unclear.

5.2.4 Accuracy

Algorithms	Accuracy
Random Forest Tree	100 Percent
Gradient Boosting	99 Percent
Artificial Neural Network	90 Percent

Table 5.2: Accuracy

5.3 Flow Control

To implement this proposed solution of our project, we used Python. For the implementation of building the model, we used Scikit-Learn and TensorFlow libraries. For the project of web application, we had to make a virtual environment first, so that the packages installed in that project do not overwrite other existing packages and libraries. After making a virtual environment we install the required libraries namely Scikit-Learn, Matplotlib, Seaborn, NumPy, Pandas, TensorFlow 2.0, Keras and Streamlit. We use Streamlit to develop the web application to deploy our model. We use Jupyter-Notebook and Visual Studio Code as the IDEs.

To create a web project, we followed the following steps:

1. Create a virtual environment in Anaconda using the command: “conda create --name env-name”.
2. Installed the required packages and libraries using the pip utility of python by running the command: “pip install library-name”.

5.3.1 Web Application

To make our application accessible for anyone, we developed a simple application. After opening this application, the user can see the homepage with the general information about our application and the disease. The user can access multiple pages like generate reports, clean data, data analysis, models and prediction.

Pseudocode for the features in web application is following.

- Get Dataset
- Read the Dataset
- If dataset have missing values (Nan or Null values)
- Fill the Values with Median
- Change other features of Dataset
- Return the Clean Dataset
- Get Clean Dataset
- Read the Clean Dataset
- Implement three Algorithms

- Return the Accuracy of Algorithms
- Get the feature values
- Read the pickle file of model with highest accuracy
- SET features values to model (np. array(features_values))
- Return whether the user have chronic kidney disease or not

5.4 Components, Libraries, Web Services and stubs

5.4.1 Web Application

The UI components in our web application are created with Streamlit. In our application, we employ a variety of Streamlit tags. To build and launch our web application, we employ a number of libraries. The following packages and libraries are included:

Libraries	Purpose
NumPy	NumPy is a Python package that allows you to interact with frameworks. NumPy is a Python module to trick large arrays and matrices. As a result, we needed to install this library to perform various calculations, especially during the data cleaning phase.
Pandas	Pandas is a Python library of open source widely used for data science, data analysis, and machine learning activities. This library is used to filter data and perform various tasks.
Seaborn	Seaborn is a Python package based on open-source software. Used to analyze test data and view data. The graphs produced can be easily modified. In order to make it visible, we needed to install this library.
Scikit-Learn	Scikit-Learn is a Python machine learning library. It was formerly used to run classification algorithms.
TensorFlow	It is an open source computer software kit that speeds up and simplifies machine learning and neural network development. It was used to create the ANN model.
Streamlit	Streamlit is an Python based open source software framework. It helps us to develop web applications for data science and machine learning. Used

	in the development and implementation of web application.
--	---

Table 5.3: Libraries and Framework

5.5 Interfaces

Interfaces of our system defines the flow of a system.

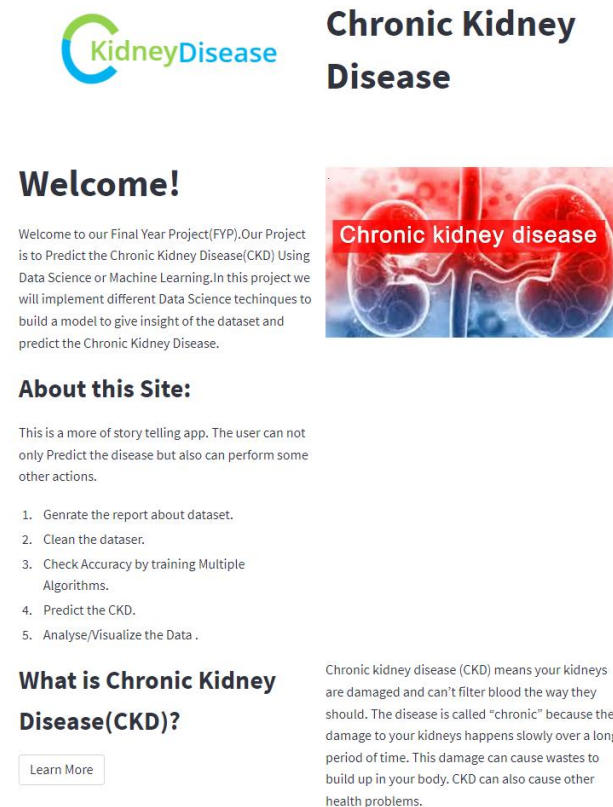


Figure 5.2: HomePage



Figure 5.3: Report Generate

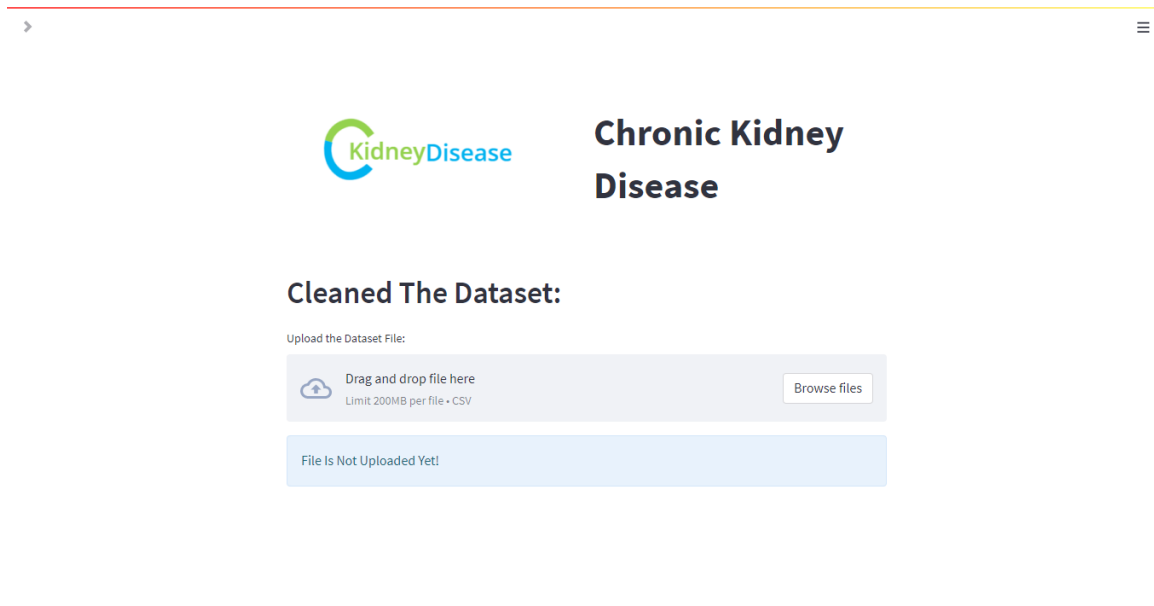


Figure 5.4: Clean Data

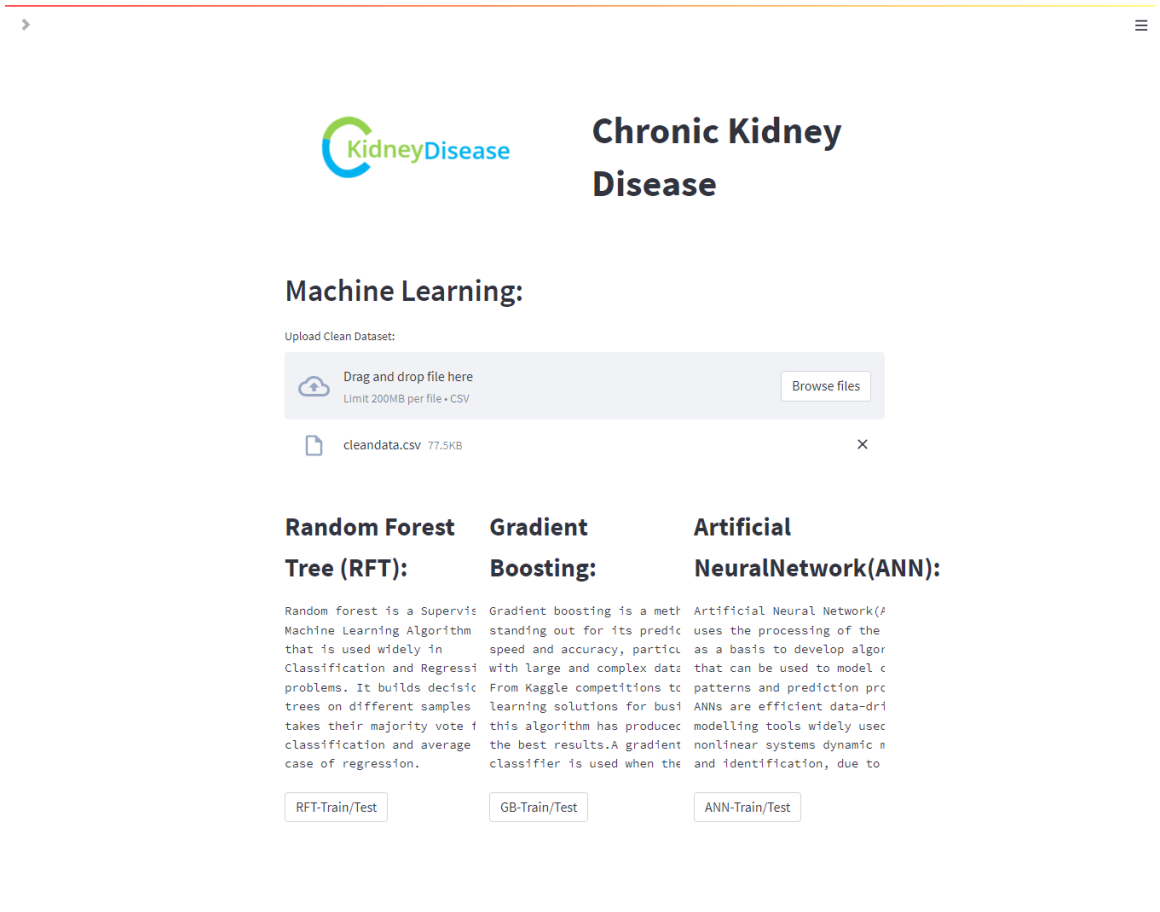



Figure 5.5: Models



Chronic Kidney Disease

Predict the Chronic Kidney Disease(CKD):

Diabetes Mellitus:

Yes = 1 & No = 0

Specific Gravity:

Ex: (1.005,1.010,1.015,1.020,1.025)

Hypertension:

Yes = 1 , No = 0

Hemoglobin (gms):

In gms

Albumin:

(0,1,2,3,4,5)

Packed Cell Volume:

(1,....,60)

Appetite:

Good = 1 , Poor = 0

Red Blood Cells (millions/cmm):

In Millions/cmm

CKD Prediction

Figure 5.6: Prediction

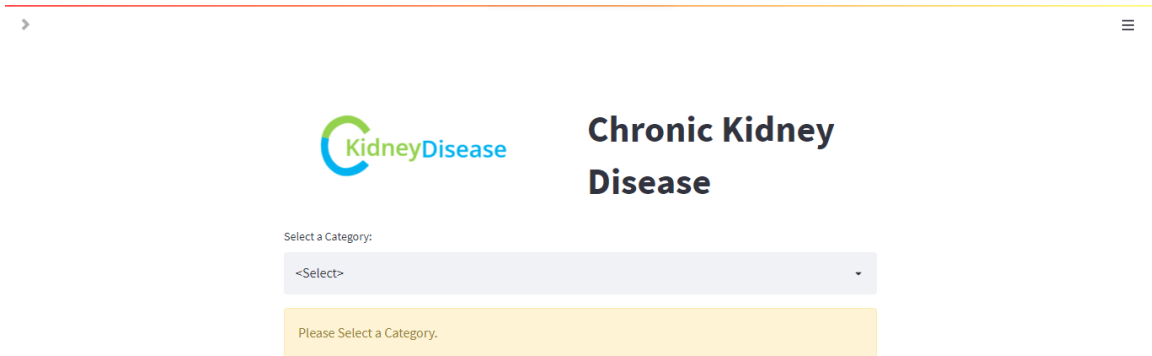


Figure 5.7: Visualization

5.6 Best Practices / Coding Standards

We followed coding standards when designing our application so that any other developer may simply understand and adjust it if necessary. We ensure that we fulfil the criteria of writing code in the right manner since we constructed a web application.

5.6.1 Code Formatting

One of the most important jobs in improving the readability of code is code formatting. Most of the time, we format code such that it is easy to read, comprehend, and modify by other developers. To structure our web application's code according to PEP-8 guidelines [22]. PEP-8 is a document that outlines the formatting requirements for Python code. We use Pascal case for the names of our classes, and snake case for the names of our files and variables. We followed the PEP-8 document's requirement of writing a Python statement in under 80 characters. We carefully followed PEP-8's blank line guideline, which stated that there should be exactly two empty lines following a function or method declaration and two empty lines after importing declarations. PEP-8 additionally requires that all files be imported in their own full statement, such as not importing two or more files in the same line.

5.6.2 Naming Conventions

We utilize the Pascal case naming convention to write the names of our classes and filenames, as described in the prior explanation. Furthermore, interfaces are built using Pascal case to name our variables, and we follow the PEP-8 requirements in terms of format.

5.6.3 Indentation

Indentations are not necessary in other programming languages. To define the block, they employ curly brackets or semicolons at the conclusion. Because Python does not utilize curly brackets to establish a code block or semicolons at the end of the line, we must indent our code. Indentation in Python indicates which code belongs to which block. Every intent should include four spaces, according to the PEP-8 coding standard.

5.6.4 Commenting Standards

After a time, it's not uncommon to forget about the code you wrote yourself. In our code, comments are used to define the intent of statements. We may also use comments to pause the execution of undesired code without eliminating it, allowing us to reuse the code that we write later.

5.6.5 Deployment Environment

For our web application to be deployed, a web browser is required. We also have hosted our web application on Streamlit cloud service so that anyone can access this application easily.

5.6.6 Summary

To complete several tasks throughout the development of our product, we used pre-built components as well as third-party libraries. We follow programming standards and always format our code accurately according to the standard guidelines.

Chapter 6: Testing and Evaluation

Chapter 6:

6. Testing and Evaluation

6.1 Introduction

The process of comparing a system's components to requirements and specifications is tested and evaluated. The findings are analyzed to determine how far the design has progressed in terms of performance and supportability, among other things. We examine that a software product's outcome fits the real need and is error-free throughout software testing.

We carefully tested the web application and attempted to test each test scenario. However, after running each test case, we discovered that our web application had no technical issues.

A doctor was also involved in the examination of this technology. In addition to the doctor, our suggested method for illness identification delivers reliable findings based on the values of the feature provided by the user.

6.2 Types of Testing

6.2.1 UNIT TESTING

Unit testing involves the creation of test scenarios to ensure that the core concept of the system is effective and that the input of the system results in significant results. Alderson branch authentication and online code flow are required. Testing of software components for each application. It is done after each unit has been completed but before it is integrated. This is an interventional building assessment based on prior knowledge of the building structure. Unit testing is used to evaluate the application of a particular business process and system configuration at the partial level.

6.2.2 INTEGRATION TESTING

Integration testing is used to determine if two or more software components can work together as a single application. Experiments run the event, focusing on the basic output of screens and fields. The composite test shows that, while the components are individually satisfying, the component combinations are correct and consistent, as shown by the successful unit test. Integration testing is a type of physical examination that aims to identify any problems that may develop as a result of component integration.

6.3 SYSTEM TESTING

Software or hardware testing is performed in a fully integrated system to test compliance with the system and its standard terms. System testing is a form of black box checking that does not require understanding the internal functioning of the code or understanding. System testing, in general, uses all integrated software components that pass the integration test, as well as the software system itself, integrated with any relevant hardware system. System testing is a very limited type of testing that appears to reveal errors in both inter-assemblage and the system as a whole. In the context of building operational requirements, system testing is performed in a complete system.

6.4 List of Test Scenarios

The following is a list of situations that were used to test the inputs and outputs for the results:

6.4.1 Test Case Scenarios for Web Application

Test Case Id	Test Scenario	Test Steps	Expected Result	Actual Result	Pass/Fail
1	Generate Report	Go to the application and Generate report section and upload data.	Generate the report that give insights of the data.	Report Generated	Pass
2	Clean	Go to the Clean	Clean the	Data Cleaned	Pass

	Dataset	data section and upload the dataset	data		
3	Implement Different Algorithms	Go to the Machine Learning part and upload the clean dataset and can train and check accuracy of those algorithms	Train and Check Accuracy	Display Accuracy	Pass
4	Submit the features values	Go to prediction part and enter features values to predict the disease using the algorithm with highest accuracy	Have Disease or not	Predict the disease	Pass
5	Analyze the data	Go to Data Analysis part and see different charts to get more information about the data	Show the graphs/charts about the specific column that user choose	Graphs/Charts Displayed	Pass

Table 6.1: Test case scenarios

6.5 Performance and Evaluation

- The aforementioned test cases are used to assess the application.
- The application's performance is consistent.
- The application is separated into two sections, one for technical users and the other for non-technical users.
- The user interface is simple and intuitive.
- The application may be used without any prior knowledge.

6.5.1 Accomplishment

We accomplished the following goals:

- For the system's online application, we created a user-friendly interface.
- We create an app that does more than simply forecast disease; it also provides a story about the data.
- Our system is as precise as it possibly can be.

6.6 Summary

This chapter discusses the testing conditions used to evaluate an application during test costs. Specifies the work to be performed, as well as the predicted and actual output when the task is evaluated. It then indicates whether the performance of the task was successful or failed based on the result. The entire application policy is briefly described.

Chapter 7:

Conclusion and Outlook

Chapter 7:

7. Conclusion and Outlook

7.1 Introduction

The goal of this project is to create a web application that can assist doctors, patients, and other medical personnel in detecting or predicting whether or not a user has chronic kidney disease. In today's age of modern technology and our fast-paced, tech-oriented lifestyle, a web application allows users to communicate with computers all over the world and anticipate chronic renal disease as well as execute other activities to have a better understanding of the data. The goal of disease prediction is to foresee diseases that, if left untreated and often disregarded, might turn deadly and create a slew of problems for the patient and their family members. The user interacts with the Prediction application by filling out a form that contains the parameter set that the trained models utilize as input. In comparison to other state-of-the-art methodologies, the Prediction app delivers the best results. The project is set up in such a manner that the system takes the user's symptoms as input and creates an output, which is illness prediction. A forecast accuracy probability of 95% is attained on average. And comparing all the Random Forest Tree gives us accuracy of 100%.

7.2 Achievements and Improvements

7.2.1 Achievements

The initiative is being created to ensure that patients, doctors, and other medical personnel use machine learning and deep learning to direct the condition. The objectives are met by offering a web application for the patient or doctor to enter feature data and determine if they have chronic kidney disease or not. Furthermore, the user may write a report on the data set, clean the data set, train three algorithms to improve their accuracy, and visualize data to learn more about it.

The features that are achieved are following:

- The user can use the web application.
- A user may submit a data set and produce a report from it.
- By uploading the raw data set file, the user can clean the data set.
- After that, the user may train three alternative algorithms and see how accurate they are.
- Furthermore, by providing values for the characteristics, the user may anticipate the illness.
- Finally, the user may examine the data by looking at the charts or graphs to have a better understanding of it.

This app may be used not only by doctors, but also by patients and medical professionals to forecast disease.

7.2.2 Improvements

In any advancement, there is always space for improvement. The user's medicine suggestions have not been incorporated in this project. As a result, drug suggestions can be incorporated into the project. A user's sickness history can be stored in a log, and pharmaceutical recommendations can be applied.

7.3 Critical Review

The application ensures that fundamental functions such as uploading files, creating reports, and cleaning data may be performed properly. There are some dangers that cannot be totally eradicated in some cases. Similarly, if the patient or doctor does not fill in all of the feature values, but the application begins to forecast the disease, or if the data set is not uploaded, but the application begins to train the algorithms.

7.4 Future Recommendations/Outlook

The aim reached in this application may be utilized to improve the application through additional research and development. To improve accuracy, deep learning and other machine learning methods can be applied.

The following are recommendations that can be implemented in the future:

- The user interface might be improved.

- More features might be added to improve the application's usefulness.
- It is possible to improve accuracy.
- Deep learning may be used to obtain the highest level of precision.
- Keeping risk mistake to a minimum.
- It's possible to implement this as a mobile application.

7.5 Summary

The chapter recounts the accomplishments of the girls in our initiative. It gives an overview of the features and functionalities that we have successfully developed in the application. It makes recommendations for future improvements to the project since there is always space for improvement. We ensured that the application is available from wherever in this project. The web application is designed to accomplish the job of allowing the user to not only anticipate sickness but also to execute additional actions in order to understand more about the data and get insight from it.

Appendix A

Work Breakdown Structure

A work breakdown structure (WBS) is deliverable based decomposition of project scope. The WBS includes 100% of the work defined by the project scope and captures all deliverables – internal, external, and interim – in terms of the work to be completed, including project management.

1) Project Management

1.1 Work Breakdown Structure (WBS)

1.2 Role and Responsibility matrix

2) Requirement Summary

2.1 Requirement Elicitation

2.1.1 Brainstorming

2.1.2 Document Analysis

2.1.3 Interview of Stakeholders

2.1.4 Questionnaires

- 2.1.5 Internet Searching and Research Papers
- 2.1.6 Similar System Analysis
- 2.1.7 Prototyping
- 2.1.8 Literature / Market Survey
- 2.2 Document
 - 2.2.1 Use Case Diagrams
 - 2.2.2 Fully Dressed Use Cases
 - 2.2.3 FRS
 - 2.2.4 NFRS
- 3) Design**
 - 3.1 High Level Design
 - 3.1.1 Use Case Diagram
 - 3.1.2 Basic Architecture Diagram
 - 3.1.3 Data Flow Diagrams
 - 3.1.4 Class Diagram
 - 3.1.5 System Sequence Diagram
 - 3.1.6 Detailed Design
 - 3.1.7 Activity Diagram
 - 3.1.8 State Chart Diagram
 - 3.1.8 Deployment Diagram
- 4) System Implementation**
 - 4.1 Tools/IDE
 - 4.1.1 Visual Studio Code
 - 4.1.2 Jupyter Notebook
 - 4.2 Technology / Language
 - 4.2.1 Python
 - 4.2.2 Machine Learning
 - 4.2.3 Deep Learning
 - 4.3 Libraries / Framework / Components
 - 4.3.1 Pandas
 - 4.3.2 NumPy

- 4.3.4 Streamlit
- 4.3.5 TensorFlow
- 4.3.6 Seaborn
- 4.3.7 Scikit-learn

5) System Testing

5.1 Static Testing

5.1.1 Structured Group Examination

5.1.1.1 Informal review

5.1.1.2 Walkthroughs

5.1.1.3 Inspection

5.1.2 Static Analyzer

5.2 Unit Testing

5.2.1 Integration Testing

5.3 Generate Report

Appendix B

Roles and Responsibility Matrix

The purpose of roles & responsibility matrix is to identify assigned task to the team member.

Table 7. 1 Roles

WBS	WBS Deliverable	Activity	Activity to Complete the Deliverable	Duration (Of Day)	Responsible Team Member(s) & Role(s)
1	Project Management	1.1	Work Breakdown Structure	5	Muhammad Tayyab, Haroon Imtiaz
		1.2	Roles and Responsibility	2	Muhammad Tayyab,

			matrix		Haroon Imtiaz
2	Requirement Summary	2.1	Requirement Elicitation	4	Muhammad Tayyab, Haroon Imtiaz
		2.1.1	Brainstorming	2	Muhammad Tayyab, Haroon Imtiaz
		2.1.2	Document Analysis	2	Muhammad Tayyab, Haroon Imtiaz
		2.13	Interview of Stakeholders	3	Muhammad Tayyab, Haroon Imtiaz
		2.14	Questionnaires	3	Muhammad Tayyab, Haroon Imtiaz
		2.15	Internet Searching and Research	7	Muhammad Tayyab, Haroon Imtiaz
		2.16	Similar System Analysis	2	Muhammad Tayyab, Haroon Imtiaz
		2.17	Prototyping	2	Muhammad Tayyab, Haroon Imtiaz
		2.18	Literature / Market Survey	4	Muhammad Tayyab, Haroon Imtiaz
		2.2	Document	2	Muhammad

					Tayyab, Haroon Imtiaz
		2.2.1	Use case Diagram	2	Muhammad Tayyab, Haroon Imtiaz
		2.2.2	Fully Dressed Use case	3	Muhammad Tayyab, Haroon Imtiaz
		2.2.3	FRS	3	Muhammad Tayyab, Haroon Imtiaz
		2.2.4	NFRS	2	Muhammad Tayyab, Haroon Imtiaz
3	Design	3.1	High Level Design	4	Muhammad Tayyab, Haroon Imtiaz
		3.1.1	Use Case Diagram	5	Muhammad Tayyab, Haroon Imtiaz
		3.1.2	Basic Architecture Diagram	4	Muhammad Tayyab, Haroon Imtiaz
		3.1.3	Data Flow Diagrams	3	Muhammad Tayyab, Haroon Imtiaz
		3.1.4	Class Diagram	3	Muhammad Tayyab, Haroon Imtiaz
		3.1.5	System	2	Muhammad

			Sequence Diagram		Tayyab, Haroon Imtiaz
		3.1.6	Activity Diagram	2	Muhammad Tayyab, Haroon Imtiaz
4	System Implementation	4.1	Tools/IDE	4	Muhammad Tayyab, Haroon Imtiaz
		4.1.1	IDE visual studio code	2	Muhammad Tayyab, Haroon Imtiaz
		4.1.2	Jupyter Notebook	2	Muhammad Tayyab, Haroon Imtiaz
		4.3	Technology Language	5	Muhammad Tayyab, Haroon Imtiaz
		4.3.1	Python	8	Muhammad Tayyab, Haroon Imtiaz
		4.3.2	Machine Learning	4	Muhammad Tayyab, Haroon Imtiaz
		4.3.3	Deep Learning	3	Muhammad Tayyab, Haroon Imtiaz
		4.4	Library/Framework/Components	3	Muhammad Tayyab, Haroon Imtiaz
		4.4.1	Pandas	2	Muhammad

					Tayyab, Haroon Imtiaz
		4.4.2	Numpy	4	Muhammad Tayyab, Haroon Imtiaz
		4.4.3	Streamlit	4	Muhammad Tayyab, Haroon Imtiaz
		4.4.4	TensorFlow	4	Muhammad Tayyab, Haroon Imtiaz
		4.4.5	Seaborn	2	Muhammad Tayyab, Haroon Imtiaz
		4.4.6	Scikit-learn	2	Muhammad Tayyab, Haroon Imtiaz
5	System Testing	5.1	Static Testing	2	Muhammad Tayyab, Haroon Imtiaz
		5.1.1	Structured Group Examination	2	Muhammad Tayyab, Haroon Imtiaz
		5.1.1.1	Informal review	2	Muhammad Tayyab, Haroon Imtiaz
		5.1.1.2	Walkthroughs	2	Muhammad Tayyab, Haroon Imtiaz
		5.1.13	Inspection	2	Muhammad

					Tayyab, Haroon Imtiaz
		5.1.2	Static Analyzer	2	Muhammad Tayyab, Haroon Imtiaz
		5.3	Unit Testing	1	Muhammad Tayyab, Haroon Imtiaz
		5.3.1	Integration Testing	2	Muhammad Tayyab, Haroon Imtiaz
		5.3.2	System Testing	2	Muhammad Tayyab, Haroon Imtiaz
		5.4	Generate Report	4	Muhammad Tayyab, Haroon Imtiaz

Table 7.1: Appendix

Reference and Bibliography

- [1] P. E. a. L. A. Stevens, "Evaluation and management of chronic kidney disease: synopsis of the kidney disease: improving global outcomes 2012 clinical practice guideline," *Annals of internal medicine*, vol. 158, pp. 825--830, 2013.
- [2] B. a. P. N. a. R. G. a. o. Bikbov, "Disparities in chronic kidney disease prevalence among males and females in 195 countries: analysis of the global burden of disease 2016 study," *Nephron*, vol. 139, pp. 313--318, 2018.
- [3] W. G. a. R. G. a. M. S. a. T. M. Couser, "The contribution of chronic kidney disease to the global burden of major noncommunicable diseases," *Kidney international*, vol. 80, pp. 1258--1270, 2011.
- [4] G. a. K. K. a. S. M. Kumar, "The use of artificial intelligence based techniques for intrusion detection: a review," *Artificial Intelligence Review*, vol. 34, pp. 369--387, 2010.
- [5] M. a. H. O. a. K. S. a. o. Ponum, "EasyDetectDisease: An Android App for Early Symptom Detection and Prevention of Childhood Infectious Diseases," *Interactive journal of medical research*, vol. 8, p. e12664, 2019.
- [6] N. R. a. F. S. T. a. O. J. L. a. H. J. A. a. O. C. A. a. L. D. S. a. H. F. R. Hill, "Global prevalence of chronic kidney disease--a systematic review and meta-analysis," *PloS one*, vol. 11, p. e0158765, 2016.
- [7] P. M. Patil, "Review on prediction of chronic kidney disease using data mining techniques," *International Journal of Computer Science and Mobile Computing*, vol. 5, pp. 135--141, 2016.
- [8] R. S. & R. N, "Diagnosis of chronic kidney disease using machine learning algorithms," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, pp. 812--820, 2016.
- [9] F. T. N. T. C. Charleonnann A, "Predictive analytics for chronic kidney disease using machine learning techniques," *IEEE International Conference on Management and Innovation Technology (MITicon)*, p. 2016.

- [10] K. a. N. Y. a. K. M. V. Lakshmi, "Performance comparison of three data mining techniques for predicting kidney dialysis survivability," *International Journal of Advances in Engineering \& Technology*, vol. 7, p. 242, 2014.
- [11] K. a. A. B. Kumar, "Artificial neural networks for diagnosis of kidney stones disease," vol. 10, 2012.
- [12] M. a. R. M. a. S. M. M. Khavanin Zadeh, "Data mining performance in identifying the Risk Factors of early arteriovenous fistula failure in Hemodialysis Patients," *International journal of hospital research*, vol. 2, pp. 49--54, 2013.
- [13] R. a. M. B. K. Ahmad, "Chronic kidney disease stage identification using texture analysis of ultrasound images," *Biomedical Signal Processing and Control*, vol. 69, p. 102695, 2021.
- [14] W. a. P. K. a. K. K. Gunarathne, "Performance evaluation on machine learning classification techniques for disease classification and forecasting through data analytics for chronic kidney disease (CKD)," in *2017 IEEE 17th international conference on bioinformatics and bioengineering (BIBE)*, 2017, pp. 291--296.
- [15] F. a. B. M. S. a. P. K. Kayaalp, "A hybrid classification example in describing chronic kidney disease," in *2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT)*, IEEE, 2018, pp. 1--4.
- [16] M. B. a. N. D. M. a. L. C. P. Twidale, "Everyone everywhere: A distributed and embedded paradigm for usability," *Journal of the Association for Information Science and Technology*, 2021.
- [17] M. D. a. R. J. T. a. K. J. A. Ekstrand, "Collaborative filtering recommender systems," 2011.
- [18] T. L. a. C. R. L. Lei, "Mapping transit-based access: integrating GIS, routes and schedules," *International Journal of Geographical Information Science*, vol. 24, pp. 283--304, 2010.
- [19] S. a. S. V. a. S. A. Sharma, "Performance based evaluation of various machine learning classification techniques for chronic kidney disease diagnosis," *arXiv preprint arXiv:1606.09581*, 2016.

- [20] P. Yildirim, "Chronic kidney disease prediction on imbalanced data by multilayer perceptron: Chronic kidney disease prediction," in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, 2017, pp. 193--198.
- [21] T. a. C. R. a. B. P. a. E. P. Tilley, "A survey of formal concept analysis support for software engineering activities," in *Formal concept analysis*, Springer, 2005, pp. 250--271.
- [22] <https://peps.python.org/pep-0008/>.