# Early Detection of Chronic Kidney Disease using Machine Learning

**IBM PROJECT REPORT**

***Submitted by***

TEAM ID: PNT2022TMID40421

**TEAM MEMBERS**


| | |
|---|---|
| THANIGAIVEL H P | 513419104046 |
| SHANMUGAN S | 513419104039 |
| SUDARSON PRAVINTH G | 513419104044 |
| KARTHI KEYAN KALATHI S | 513419104701 |


***in partial fulfilment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**



**UNIVERSITY COLLEGE OF ENGINEERING KANCHEEPURAM**

(A Constituent College of Anna University, Chennai)

ANNA UNIVERSITY: CHENNAI – 600 025

# INDEX

# CHAPTER 1

# INTRODUCTION

## 1.1 Project Overview:

Chronic Kidney Disease (CKD) is a major medical problem which can be diagonised by using this system. It can be cured if treated in early stages. Chronic kidney disease can be detected with regular laboratory tests, and some treatments are present which can prevent development, slow disease progression, and risk of cardiovascular disease, and improve survival and quality of life. Out of the 11 machine learning methods considered, XG BOOST classifier are shown to result in the highest accuracy and minimal bias to the attributes. The research also considers the practical aspects of data collection and highlights the importance of incorporating domain knowledge when using machine learning for CKD status prediction**.**

## 1.2 Project Overview:

The primary goal of the project is to detect the Chronic Kidney Disease . People with Chronic Kidney Disease suffer from Leg swelling, exhaustion, nausea, loss of appetite, and confusion are some of symptoms. Anemia, bone disease, and high blood pressure (often linked to the activation of the Renin-Angiotensin-Aldosterone system) are complications that can be related to hormonal dysfunction of the kidneys. In this project CKD patients and healthy subjects will be analyzed to predict the presence of Chronic Kidney Disease.

# CHAPTER 2

# LITERATURE SURVEY

## 2.1 Existing problem

At the initial stage user will gather the dataset required to make a diagnosis of CKD. Once the data has been loaded, system will preprocess the data and extraction of features will be done. Once the features required for the prediction have been extracted system will compare the features with model and prediction will be given as final result in the end.

### 2.1.1 References

| S.No | PAPER TITLE | TECHNOLOGIES USED | DESCRIPTION |
|---|---|---|---|
| 1 | Intelligent systems on the cloud for the early detection of chronic kidney disease | Back-propagation networks,Generalized Feed Forward Neural Networks, and Modular Neural Networks. | Utilizing Google Application Engine, the system created in accordance with the best model is uploaded to the Google cloud platform. The end solution can more effectively give CKD. |
| 2 | Performance Analysis of Machine Learning Classifier for Predicting Chronic Kidney Disease | Regression and classification, decision tree classifier, random forest | This proposed system detects CKD- Chronic Kidney Disease using machine learning; they have attained an accuracy of 100% for decision tree classifier, 95.12% for random forest and 98.82% in logistic regression. |

| | | | |
|---|---|---|---|
| 3 | Prediction of chronic kidney disease (CKD) using Data Science | Support Vector Machine, Random Forest, XGBoost, Logistic Regression, Neural networks, Naive Bayes Classifier. | This research work is primarily concentrated on finding the best suitable classification algorithm which can be used for the diagnosis of CKD based on the classification report and performance factors. |
| 4 | A Neural Network based Model for Predicting Chronic Kidney Diseases | Artificial Neural Network algorithms | The 14 different properties are analyzed and linked to chronic kidney disorder victims and foretold accuracy for a machine learning algorithm named Artificial Neural Network. A er analyzing the outcomes, it is recognized that the algorithm gives correctness of 96. |
| 5 | A Machine Learning Methodology for Diagnosing Chronic Kidney Disease | Logistic regression, Random forest, Support vector machine, knearest neighbor, Naive Bayes classifier, and Feed Forward Neural | A machine learning approach for diagnosing CKD was suggested in this study. An integrated model that combines logistic regression and random forest with the aid of perceptron was utilized and it was able to attain an average accuracy of 99.83% a er ten times of simulation. |
| 6 | Early Diagnosis of Chronic Kidney Disease Using Machine Learning Algorithms with Least Parameters by RFE and Feature Importance Techniques | Linear, Logistic, Decision tree, CART, and Random forest classifier | The primary goal of this research project is to enhance the diagnostic precision by assessing the optimum feature selection and developing a prediction model using machine learning methods. By using different classifier methods, the model achieved a diagnosis accuracy of 0.925. |

| | | | |
|---|---|---|---|
| 7 | Chronic kidney disease Diagnosis using Multilayer perceptron classifier | Multilayer Perceptron Classifier | The Experimental results show that the proposed model can perform classification with the testing accuracy of 92.5% surpassing the scores achieved by SVM and naive bayes classifier. |
| 8 | Detection of Chronic Kidney Disease Using Machine Learning Algorithms with Least Number of Predictors | Logistic regression, SVM, Random forest, and Gradient boosting | The link between variables has been researched in order to decrease the number of features and eliminate redundancy. Tenfold cross-validation has been used to train, test, and validate the classifiers. |
| 9 | Chronic Kidney Disease Prediction and Recommendation of Suitable Diet plan by using Machine Learning | Machine Learning Algorithms,MDRD equation | The proposed system which detects chronic kidney disease using machine learning defines 3 zones(Safe zone,Caution zone,Danger zone) on the basis of blood potassium level. doctors a different native technique to detect chronic renal illnesses in a patient's early stages. |

| | | | |
|---|---|---|---|
| 10 | Optimization of Prediction Method of Chronic Kidney Disease Using Machine Learning Algorithm. | Support Vector Machine, AdaBoost, Linear Discriminant Analysis, and Gradient Boosting. | These algorithms are used using a dataset from the UCI machine learning repository that is available online. Gradient Boosting (GB) Classifiers produce results with a predictably high accuracy of roughly 99.80%. Based on these benchmarks, the most effective and optimized algorithms for the requested job can be chosen. |

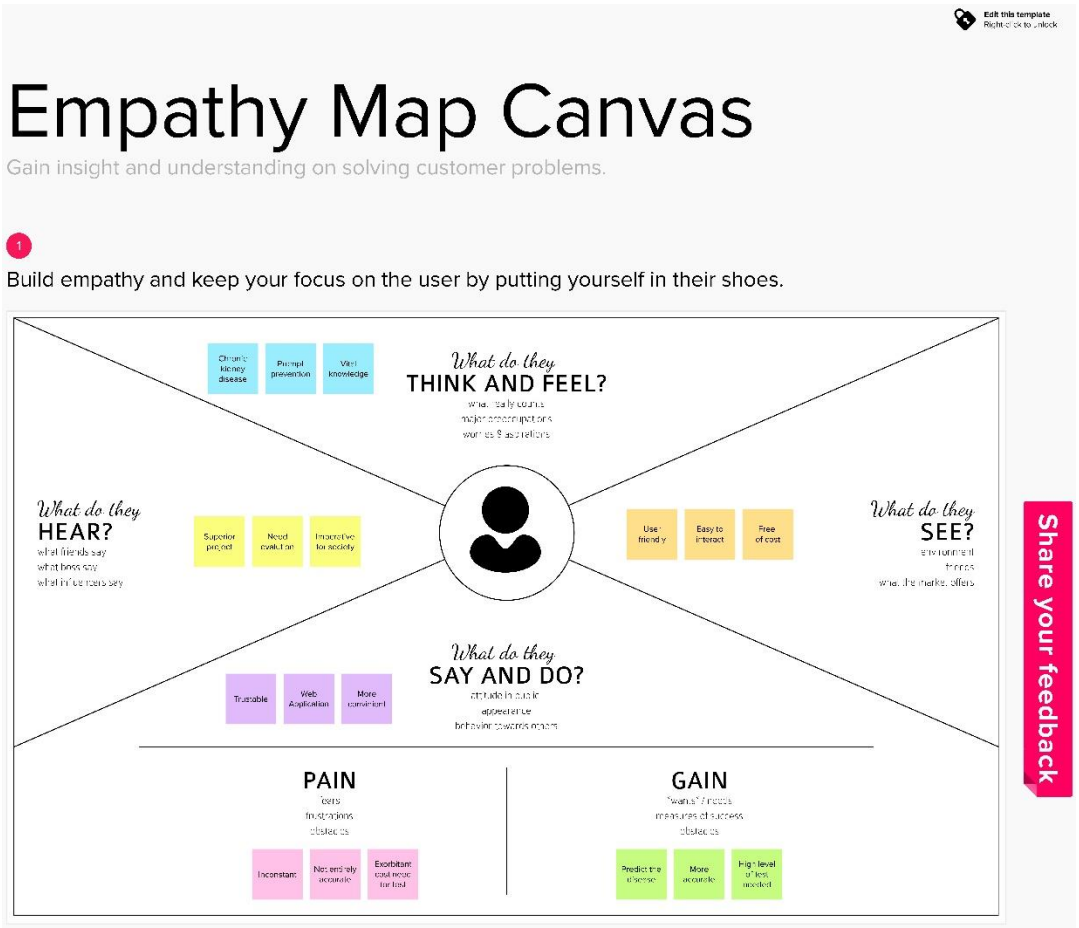
### 2.1.2 Problem Statement Definition

- Chronic Kidney Disease (CKD) is a major medical problem which can be diagonised by using this system. It can be cured if treated in early stages.
- Chronic kidney disease can be detected with regular laboratory tests, and some treatments are present which can prevent development, slow disease progression, and risk of cardiovascular disease, and improve survival and quality of life.
- CKD also recognized as Chronic Renal Disease (CRD) which is an uncharacteristic functioning of kidney or a failure of renal function expanding over a period of months or years.
- Final output predicts whether the person is having CKD or not by using minimum number of features.

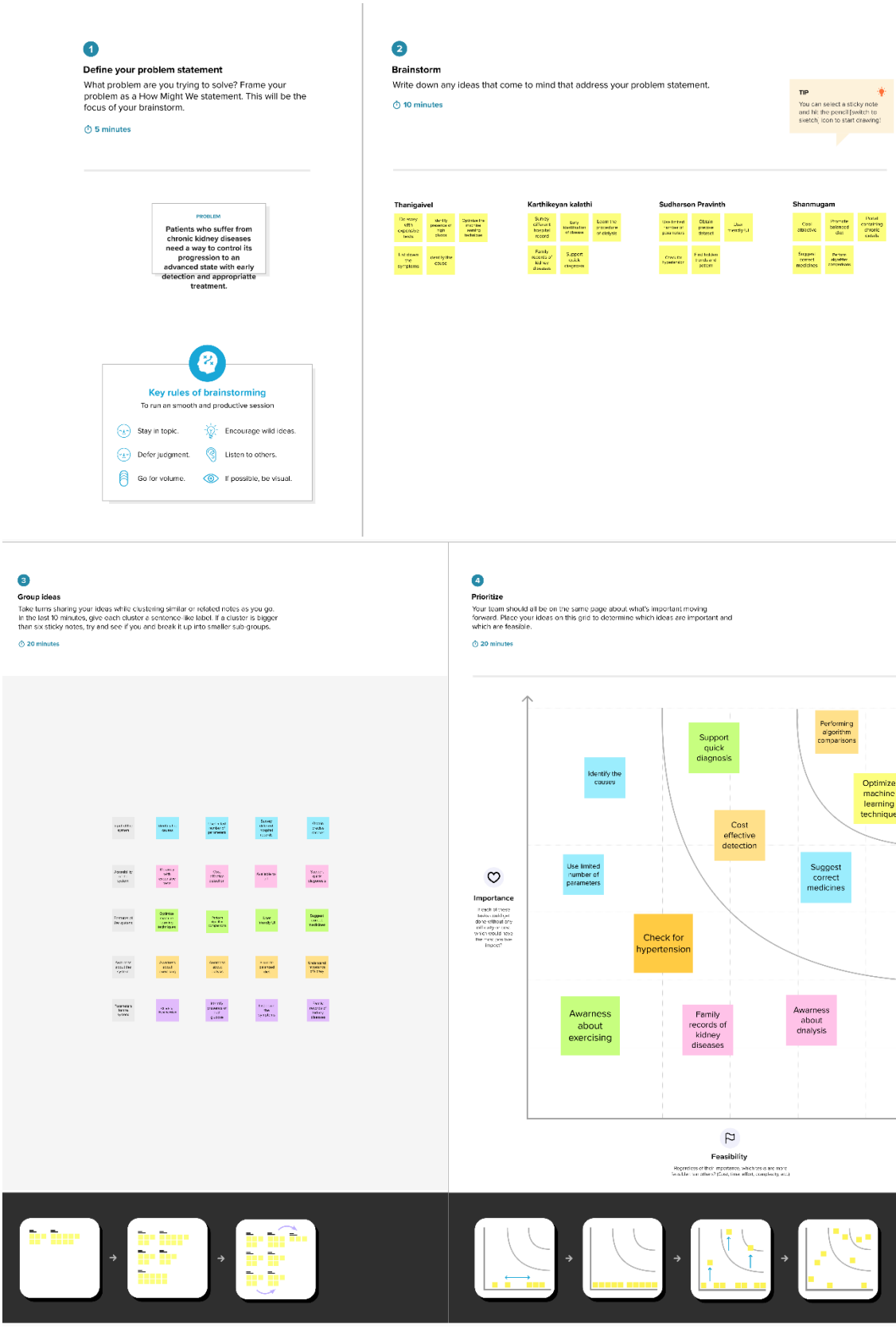
# CHAPTER 3

# IDEATION & PROPOSED SOLUTION

## 3.1 Empathy Map Canvas



# 3.2 Ideation & Brainstorming

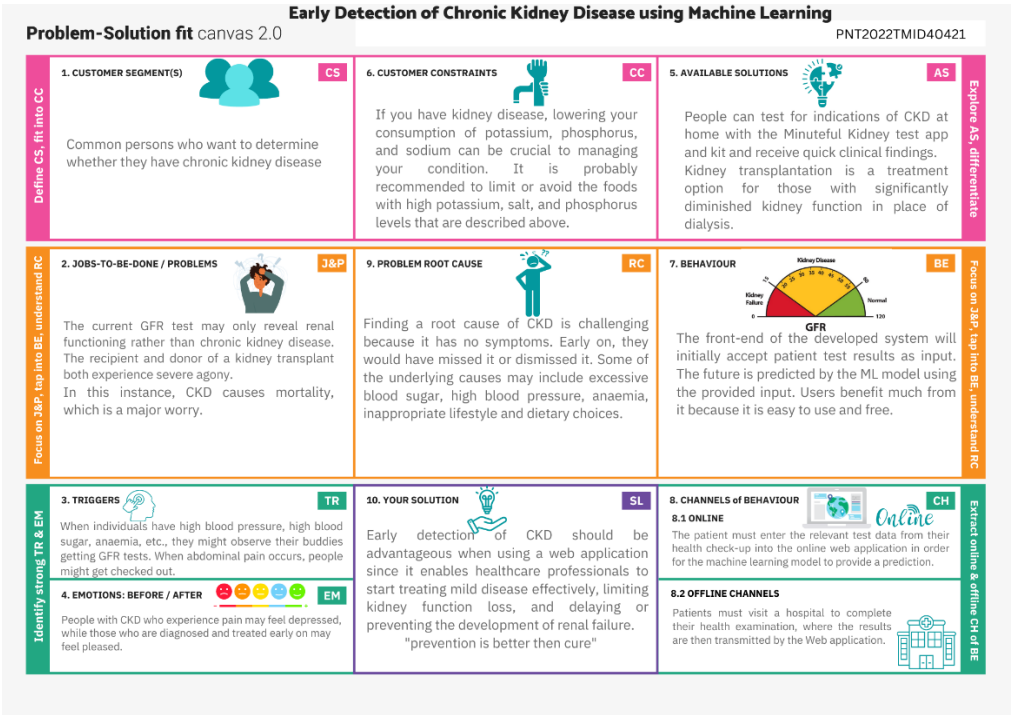


### 3.3 Proposed Solution

| SNO | Parameter | Description |
|---|---|---|
| 1 | Problem Statement (Problem to be solved) | Detection of Kidney disease at an early stage |
| 2 | Idea / Solution description | The feasibility and importance of these ideas are evaluated using the brainstorming and idea prioritising template in mural. All those are:<br>1. Examine urine pus cells, anaemia, urea, edema, packed cell volume, and diabetes. If it is higher than the cutoff then mention CKD.<br>2. Regular blood tests for Glomerular Filtration Rate (GFR) and Creatinine |
| | | For the standard techniques of early detection and avoidance of chronic kidney disease, all the suggestions above are to be followed. Analyze each of them separately.<br>Regarding point no. 1,<br>It is possible to check the anaemia, urea, packed cell volume, edoema in the leg, diabetes, and urine pus cells of many people, roughly 1000 people, and then build an excel sheet to determine whether or not they have chronic kidney disease. This excel spreadsheet can then be used to train and test an AI model. Later, the model can determine whether or not chronic renal disease is present if only these parameters are fed into it.<br>Regarding point no. 2,<br>One thousand people's blood samples for creatinine and glomerular filtration rate (GFR) can be examined to determine whether they have chronic kidney disease. Based on the data, an AI model can be created, trained, and tested to forecast chronic kidney disease. A ML model can be built to analyze these data and further used to predict the kidney disease from it. |

| | | |
|---|---|---|
| 3 | Novelty / Uniqueness | According to numerous specialists, the best conventional strategy is to evaluate multiple blood parameters and spot chronic kidney disease at an early stage. User is able to identify Chronic Kidney Disease in around the time of (15 minutes). |
| 4 | Social Impact / Customer Satisfaction | Early detection of CKD which reduce fatalities related to this disease and it also help in facilitation of appropriate dosing of medications and allow timely preparation for kidney replacement, which may improve outcome. |
| 5 | Business Model (Revenue Model) | Capable of bringing revenue from directly clients. Can collaborate with health care sector and generate revenue from their customers. |
| 6 | Scalability of the Solution | ML models are easily resized and updated. The same machine learning model can therefore be used to identify other fatal diseases when fed with different sets of data |

## 3.4 Problem Solution Fit



# CHAPTER-4

REQUIREMENT ANALYSIS

## 4.1 Functional requirement

**Following are the functional requirements of the proposed solution.**

| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task) |
|---|---|---|
| FR-1 | User Registration | Registration through Login page |
| FR-2 | User Confirmation | Confirmation via Email<br>Confirmation via OTP |
| FR-3 | User Requirements | Create an account, see certain content, recalculate data using a particular algorithm, and perform other actions. |
| FR-4 | Business Requirements | An Application allowing patient to identify Chronic kidney disease in around the time of 15 minutes. |
| FR-5 | User Authentication | Challenges the user to validate credentials(for example, through Password) |
| FR-6 | User Authorization | Once the server receives a request with authorization, it can verify your credentials and grant you access to the resources. |

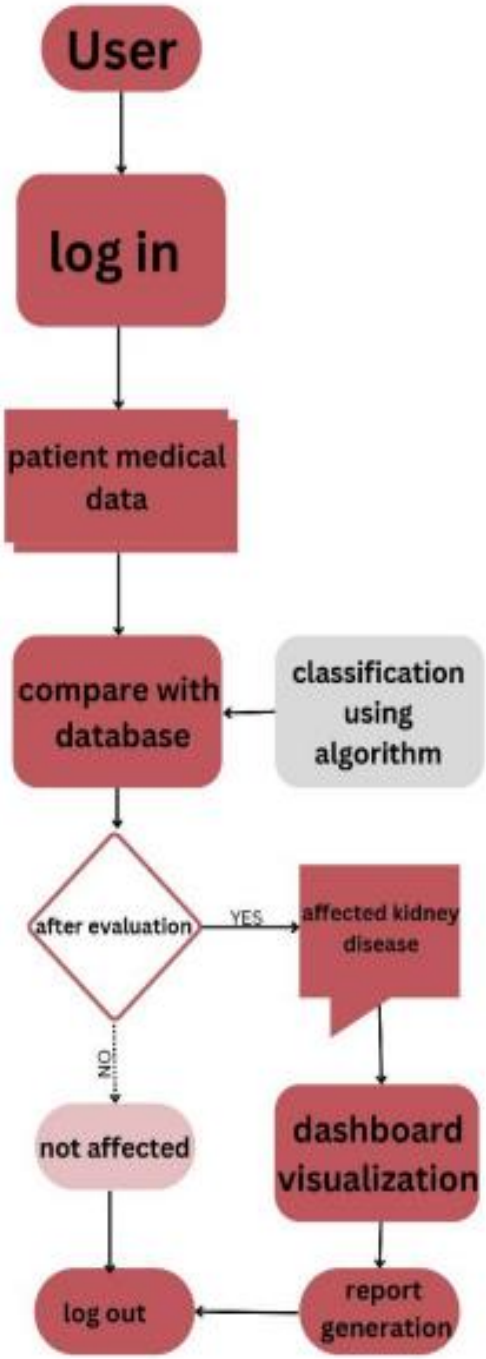
## 4.2 Non-Functional Requirement:

Following are the non-functional requirements of the proposed solution.

| FR No. | Non-Functional Requirement | Description |
|---|---|---|
| NFR-1 | **Usability** | Users can easily traverse an application's interface when it is useful, and applications with high usability allow users to quickly understand what a feature is and what it can achieve. |
| NFR-2 | **Security** | Safeguard sensitive information and secure databases to store patients' medical records. |
| NFR-3 | **Reliability** | You can check the percentage of the probability of failure, or failure rate, to determine the reliability of a system. |
| NFR-4 | **Performance** | Reducing overall load time of prediction and interactivity with user |
| NFR-5 | **Availability** | Indicates the user's ability to access the system at some point |
| NFR-6 | **Scalability** | Describes the ability to properly handle upbringing (and downgrade) workload |

# CHAPTER-5
# PROJECT DESIGN

## 5.1 Data Flow Diagrams



## 5.2 Solution & Technical Architecture

### Technical Architecture:

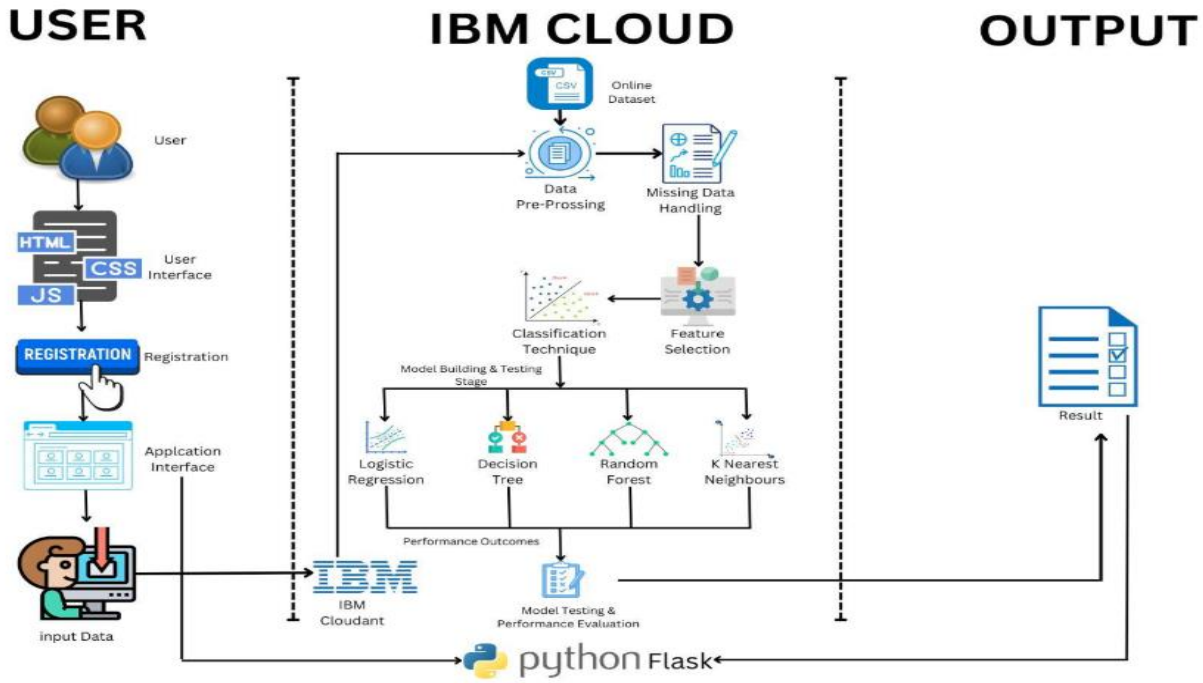



Table: Components & Technologies:

| S.No | Component | Description | Technology |
|---|---|---|---|
| 1. | User Interface | An Interface for the user to interact with the prediction model. | HTML, CSS, JavaScript |
| 2. | User Registration | User can register in the web application | HTML forms |
| 3. | Disease Prediction | The user enters the data which is given as input to model to predict the disease. | Machine Learning with Python. |
| 4. | Update Prediction result | The result of disease prediction is updated in the Web UI for the user to know the output. | Python. |
| 5. | Database | Relational database structure to store the user data | MYSQL. |
| 6. | Cloud Database | Database services on IBM cloud. | IBM Cloudant. |
| 7. | Machine Learning Model | To predict he chronic kidney disease (CKD) with various input parameters. | Random Forest, KNN, Decision tree, Logistic Registration. |
| 8. | Infrastructure (Server / Cloud) | Application Deployment on Cloud | IBM Cloud. |

## 5.3 User Stories

**User Stories**

Use the below template to list all the user stories for the product.

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|---|---|---|---|---|---|---|
| Customer (Mobile user) | Registration | USN-1 | As a user can register for the application by entering my email, password, and confirming password. | user can access account / dashboard | High | Sprint-1 |
| | | USN-2 | As a user will receive confirmation email once have registered for the application | User can receive confirmation email & click confirm | High | Sprint-1 |
| | | USN-3 | As a user can register for the application through Login page | User can register & access the dashboard with Login page | Low | Sprint-2 |
| | | USN-4 | As a user can verify the signup credentials of the application through mail | User can receive the registration conformation email | Medium | Sprint-1 |
| | Login | USN-5 | As a user can log into the application by entering email & password | dashboard was immediately directed | High | Sprint-1 |
| | Dashboard | USN-6 | User can access my dashboard whenever want as a user by utilising user login information | User can access all the details in my dashboard | Medium | Sprint-1 |
| Customer (Web user) | Sign-in | USN-7 | User can use the application as a user at any time and from any location by using user login credentials | User can utilize my time & access the application | High | Sprint-2 |
| Customer Care Executive | Clarification | USN-8 | As a user, they need to clarify some related problems while using the application | Web application give detailed explanation to user 24/7 | Medium | Sprint-2 |
| Administrator | Quality assurance | USN-9 | As a user they have some credibility issues while using application | We can give 100% assurance to that application | High | Sprint-3 |

# CHAPTER 6
# PROJECT PLANNING & SCHEDULING:

## 6.1 Sprint Planning & Estimation

Use the below template to create product backlog and sprint schedule

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
|---|---|---|---|---|---|---|
| Sprint-1 | Collection of Dataset | USN-1 | Collect dataset from Google and clean the dataset | 5 | High | Shanmugan S |
| Sprint-1 | Model | USN-2 | Create, test and save the model | 5 | High | Karthi Keyan Kalathi S ,Thanigaivel HP |
| Sprint-2 | Home page | USN-3 | The user can enter into the home page | 6 | | Suderson Pravinth G |
| Sprint-2 | Prediction button | USN-4 | User can use the prediction button to enter into the prediction page | 4 | High | Shanmugan S |
| Sprint-3 | Prediction page | USN-5 | The user will be presented with the prediction page where he can enter the values of report | 3 | Low | Suderson Pravinth G |

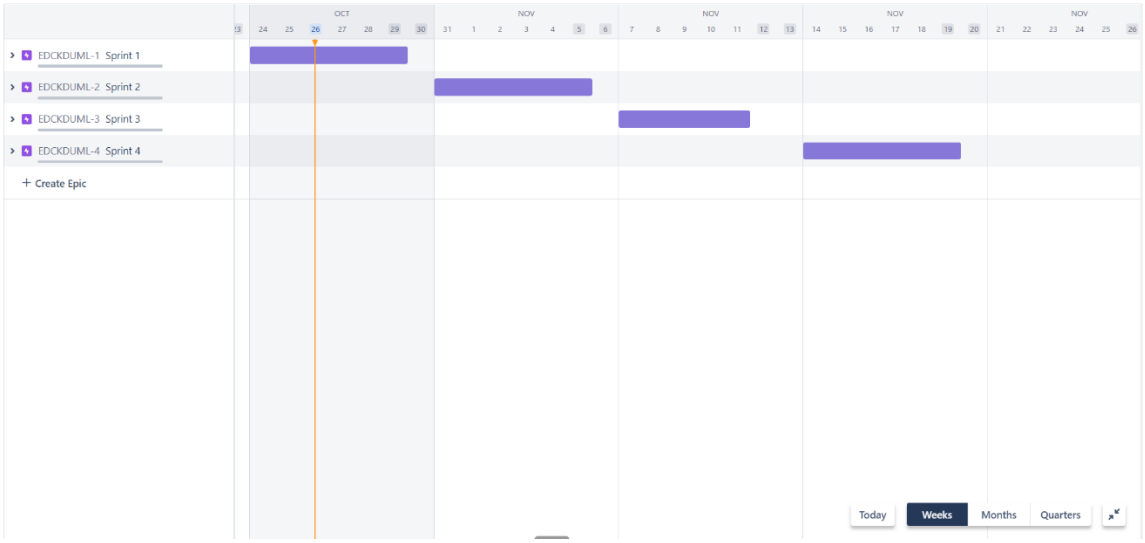
| | | | | | | |
|---|---|---|---|---|---|---|
| Sprint-3 | | USN-6 | User should enter the blood glucose parameters | 7 | Medium | Thanigaivel HP |
| Sprint-4 | Result | USN-7 | The user will get the output | 4 | High | Karthi Keyan Kalathi S |
| Sprint-4 | Deployment | USN-8 | Deploy into IBM CLOUD | 6 | High | Karthi Keyan Kalathi S ,Thanigaivel HP, Suderson Pravinth G,Shanmugan S |

## 6.2 Sprint Delivery Schedule

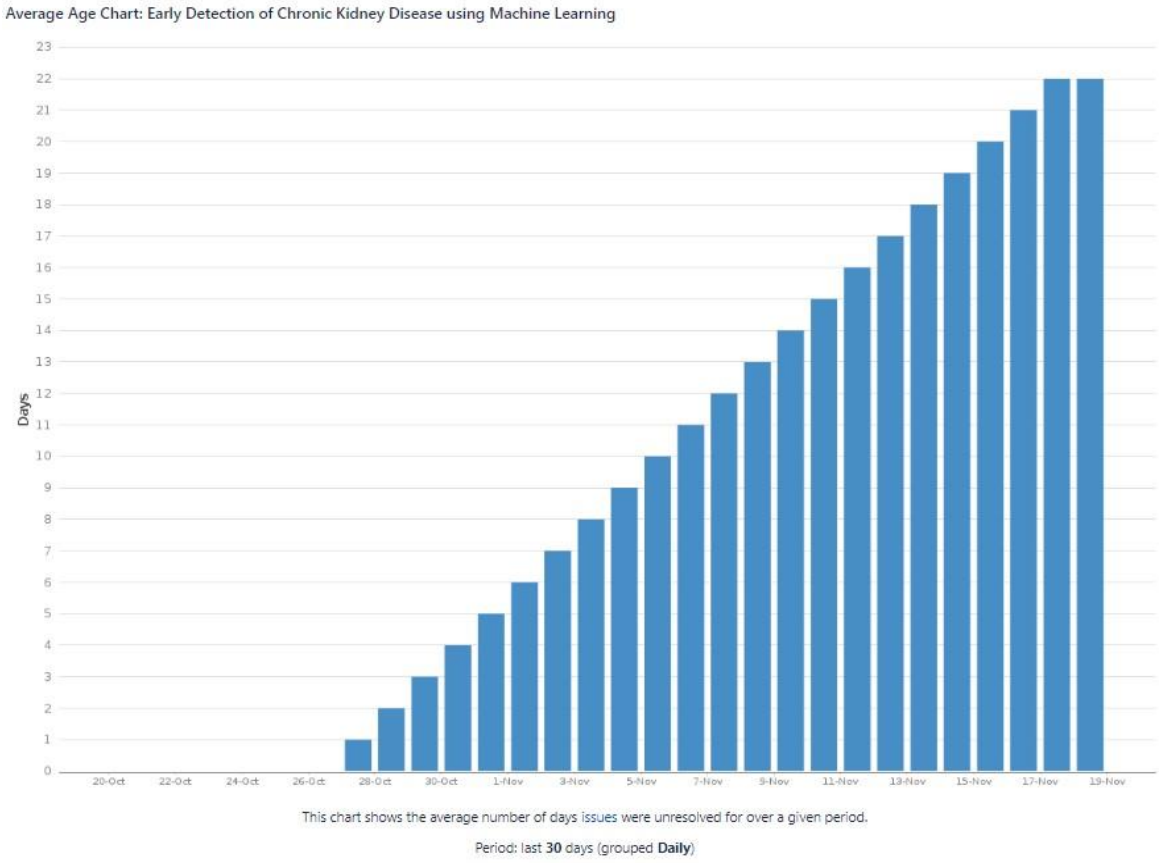
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date (Actual)** |
|---|---|---|---|---|---|---|
| Sprint-1 | 2 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 20 | 29 Oct 2022 |
| Sprint-2 | 2 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 31 Oct 2022 |
| Sprint-3 | 4 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 07 Nov 2022 |
| Sprint-4 | 1 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 14 Nov 2022 |

## 6.3 Reports from JIRA

ROADMAP:




BURNDOWN CHART:



# CHAPTER-7

# CODING & SOLUTIONING

## Install & Importing the Dependencies

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

## Extract the Data-Set ( Kidney_Disease.csv )

```
kidney=pd.read_csv('kidney_disease.csv')
```

# Performing Exploitory Data Analysis ( EDA )

- Modifying the Column Names as per our requirements

```
columns=pd.read_csv("data_description.txt",sep=' - ')
columns=columns.reset_index()
columns = columns.drop('index',axis=1)
```

# Performing Data cleaning

```
def extract_cat_num(kidney):
    cat_col=[col for col in kidney.columns if kidney[col].dtype=='O']
    num_col=[col for col in kidney.columns if kidney[col].dtype!='O']
    return cat_col,num_col
cat_col,num_col=extract_cat_num(kidney)

# dirtiness in categorical data
for col in cat_col:
    print('{} has {} values'.format(col,kidney[col].unique()))
    print("\n")


kidney['diabetes mellitus'].replace(to_replace={'\tno':'no','\tyes':'yes'},inplace=True)

kidney['coronary artery disease'].replace(to_replace={'\tno':'no'},inplace=True)

kidney['class'].replace(to_replace={'ckd\t':'ckd'},inplace=True)

# no dirtiness

for col in cat_col:

    print('{} has {} values'.format(col,kidney[col].unique()))

    print("\n")
```

# Analysing distribution of each and every column

```
len(num_col)

plt.figure(figsize=(30,30))

for i,feature in enumerate(num_col):

   plt.subplot(5,3,i+1)   # 5 rows and 3 columns

   kidney[feature].hist()

   plt.title(feature)
```

## Check Label distribution of categorical Data

```
plt.figure(figsize=(20,20))


for i,feature in enumerate(cat_col):

   plt.subplot(4,3,i+1)

   sns.countplot(kidney[feature])


plt.figure(figsize=(20,20))


for i,feature in enumerate(cat_col):

   plt.subplot(4,3,i+1)

   sns.countplot(kidney[feature],hue=kidney['class'])

sns.countplot(kidney['class'])
```

## Correlation between features

```
plt.figure(figsize=(12,12))

sns.heatmap(kidney.corr(method='pearson'),cbar=True,cmap='BuPu',annot=True)
```

Analyse distribution of red blood cell count chronic as well as non chronic

```
grid=sns.FacetGrid(kidney,hue='class',aspect=2)

grid.map(sns.kdeplot,'red blood cell count')

grid.add_legend()

grid=sns.FacetGrid(kidney,hue='class',aspect=2)

grid.map(sns.kdeplot,'haemoglobin')

grid.add_legend()
```

```
plt.figure(figsize=(12,10))
sns.scatterplot(x=kidney['red blood cell count'],y=kidney['packed cell volume'],hue=kidney['class'])
plt.xlabel('red blood cell count')
plt.ylabel('packed cell volume')
plt.title('Relationship between red blood cell count and packed cell volume')

plt.figure(figsize=(12,10))
sns.scatterplot(x=kidney['red blood cell count'],y=kidney['haemoglobin'],hue=kidney['class'])
plt.xlabel('red blood cell count')
plt.ylabel('haemoglobin')
plt.title('Relationship between haemoglobin and red blood cell count')
```

# Handling Missing Values

```
kidney.isnull().sum()
plt.subplot(1,2,1)
sns.boxplot(x=kidney['class'],y=kidney['age'])
list(enumerate(cat_col))
plt.figure(figsize=(15,15))
for i in enumerate(num_col):
    plt.subplot(4,4,i[0]+1)
    sns.boxplot(x=kidney['class'],y=i[1],data=kidney.reset_index())
np.mean(kidney)
kidney.isnull().sum()
for i in num_col:
    kidney[i].fillna(kidney[i].median(),inplace=True)
kidney.isnull().sum()
kidney.describe()
```

# Filling missing values in categorical columns using random values

```
kidney['red blood cells'].isnull().sum()

random_sample=kidney['red blood cells'].dropna().sample(152)

random_sample

kidney[kidney['red blood cells'].isnull()].index

random_sample.index

random_sample.index=kidney[kidney['red blood cells'].isnull()].index    #in this way index will be equal

random_sample.index

kidney.loc[kidney['red blood cells'].isnull(),'red blood cells']=random_sample

kidney['red blood cells'].isnull().sum()

sns.countplot(kidney['red blood cells'])       # checking that ratio didnt change after filling missing values

#filling random values in all categorical columns

def Random_value_Imputation(feature):
    random_sample=kidney[feature].dropna().sample(kidney[feature].isnull().sum())
    random_sample.index=kidney[kidney[feature].isnull()].index
    kidney.loc[kidney[feature].isnull(),feature]=random_sample

Random_value_Imputation('pus cell')     #only this column because it has higher no. of missing value

kidney.isnull().sum()

def impute_mode(feature):
    mode=kidney[feature].mode()[0]
    kidney[feature]=kidney[feature].fillna(mode)

for col in cat_col:
    impute_mode(col)

kidney[cat_col].isnull().sum()

kidney.isnull().sum()
```

## Performing the Feature Encoding

```
for col in cat_col:
    print('{} has {} categories'.format(col,kidney[col].nunique()))
```

## Label Encoding ---> Because there are less no. of categories in each column

LabelEncoder can be used to normalize labels. It can also be used to transform non-numerical labels (as long as they are hashable and comparable) to numerical labels. Fit label encoder.

- normal -- 0
- abnormal --1

```
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

for col in cat_col:
    kidney[col]=le.fit_transform(kidney[col])
```

# XGBoost Classifier - For our Model

## Since we are using XGBoost , feature scaling is not required

```
from xgboost import XGBClassifier

params={'learning-rate':[0,0.5,0.20,0.25],
        'max_depth':[5,8,10],
       'min_child_weight':[1,3,5,7],
       'gamma':[0.0,0.1,0.2,0.4],
       'colsample_bytree':[0.3,0.4,0.7]}

from sklearn.model_selection import RandomizedSearchCV

classifier=XGBClassifier()

random_search=RandomizedSearchCV(classifier,param_distributions=params,n_iter=5,scoring='roc_auc',n_jobs=-1,cv=5,verbose=3)

random_search.fit(X_train,y_train)

random_search.best_estimator_    #Checking for best model

random_search.best_params_ classifier.fit(X_train,y_train)
```

# CHAPTER-8

# TESTING

## 8.1 Test Cases

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Date | 17.11.2022 | | | | | | | | |
| | | | | Team ID | PNT2022TMID40421 | | | | | | | | |
| | | | | Project Name | Early Detection of Chronic Kidney | | | | | | | | |
| | | | | Maximum Marks | 4 marks | | | | | | | | |
| Test case ID | Feature Type | Component | Test Scenario | Pre-Requisite | Steps To Execute | Test Data | Expected Result | Actual Result | Status | Commnets | TC for Automation(Y/N) | BUG ID | Executed By |
| LoginPage_TC_OO1 | Functional | Kaggle | Verify User can collect from Hospitals /Download from Kaggle for further purpose. | Kaggle | 1.Enter into kaggle website<br>2.Download the dataset | https://www.kaggle.com/ | Download the Dataset | Working as expected | Pass | Downloaded dataset from kaggle | NO | | SUDARSON PRAVINTH G |
| LoginPage_TC_OO2 | Functional | Data Pre-processing | verify Data pre-processing by using machine learning algorithm in Jupiter Notebook | Anaconda prompt , Jupyter Notebook | 1.Enter Anaconda prompt<br>2.Enter Jupyter Notebook & do Data pre-processing | | Pre-processing the dataset using machine learning Algorithm | Working as expected | pass | pre-processed and handeled the data successfully | NO | | KARTHI KEYAN KALATHI S |
| LoginPage_TC_OO3 | Functional | Build a Model | Verify user can Build a Machine learning model using Logistic Regression & Save the model in Pickle form | Anaconda prompt , Jupyter Notebook | 1.Enter Anaconda prompt<br>2.Enter Jupyter Notebook & do Model Building | Model building using logistic regression | Build a Machine Learning Model | Working as expected | pass | | NO | | THANIGAIVEL H p |
| LoginPage_TC_OO4 | UI | Flask Deployment | Verify user can Create Html pages home.html,kidney.html, main.html & predict.html Run both pages in app.py | Visual Studio Code | 1.Click on VS code ,create html pages . Run html pages on app.py by using live server . | Run a website in localhost server http://127.0.0.1:5000/ | Appears a Prediction page on local host server | Working as expected | pass | using VS code created home.html,kidney.html,main.html & predict.html | NO | | SHANMUGAN S |
| LoginPage_TC_OO5 | UI | Local host | Verify user can Run in localhost server Entering home page by home.html then kidney.html gives prediction page & predict.html gives Result | Visual Studio Code | Click on the http link<br>Enter the values as in the dataset<br>Click on submit | Gives prediction result as patient have CKD or NOT http://127.0.0.1:5000/predict | Predict the Result | Working as expected | Pass | Entering data of patient click Sumbit it shows a person affected or not | NO | | SUDARSON PRAVINTH G |
| LoginPage_TC_OO6 | Functional | IBM Deployment | Verify user can Deploy using Jupiter notebook in IBM with cloud object storage, Watson studio Integration with flask using API key & Scoring Endpoint | IBM CLOUD | 1.Enter IBM Cloud using login credentials<br>2.Use jupyter notebook in IBM<br>3.create project & deployment sapce<br>4.Create watson studio, cloud object storage.<br>5.deploy on IBM. | Deploy the project in IBM CLOUD | Application should show same resut as vs code flask integration | Working as expected | Pass | Deployed in IBM Watson studio scikit_model.pkl | NO | | KARTHI KEYAN KALATHI S |
| LoginPage_TC_OO6 | Functional | Flask Integration | Verify Using addign API key & Scoring End point Run in new.py | IBM CLOUD | Click on VS code & execute new.py | scoring point | Finding an Scoring point | Working as expected | Pass | Run new.py to check scoring point | NO | | SHANMUGAN S |
| LoginPage_TC_OO6 | Functional | sk Integrati | Verify Using adding model scikit_model.pkl and adding API key & Scoring End point Run home.html,kidney.html, main.html & predict.html pages in app.py in local host server | IBM CLOUD | /S code & execute same html pages | in localhost server http://12 | Appears a Prediction page on local h | rking as expe | pass | Run app.py | NO | | THANIGAIVEL H p |

## 8.2 User Acceptance Testing

### 1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

### 2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

| Resolution | Severity 1 | Severity 2 | Severity 3 | Severity 4 | Subtotal |
|---|---|---|---|---|---|
| By Design | 11 | 4 | 3 | 3 | 21 |
| Duplicate | 1 | 0 | 3 | 0 | 4 |
| External | 2 | 3 | 0 | 1 | 6 |
| Fixed | 12 | 2 | 4 | 20 | 38 |
| Not Reproduced | 0 | 0 | 1 | 0 | 1 |
| Skipped | 0 | 0 | 1 | 1 | 2 |
| Won't Fix | 0 | 0 | 0 | 0 | 0 |
| Totals | 26 | 9 | 12 | 25 | 72 |

## 3.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

| Section | Total Cases | Not Tested | Fail | Pass |
|---|---|---|---|---|
| Print Engine | 7 | 0 | 0 | 7 |
| Client Application | 50 | 0 | 0 | 50 |
| Security | 0 | 0 | 0 | 0 |
| Outsource Shipping | 3 | 0 | 0 | 3 |
| Exception Reporting | 9 | 0 | 0 | 9 |
| Final Report Output | 4 | 0 | 0 | 4 |
| Version Control | 2 | 0 | 0 | 2 |

# CHAPTER-9

# RESULTS

## 9.1 Performance Metrics

### Let's Predict our model Accuracy.

```
In [ ]: y_pred=classifier.predict(X_test)

In [ ]: y_pred

Out[125]: array([0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
                 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
                 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
                 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1,
                 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1,
                 0, 0, 1, 0, 1, 0, 1, 0, 0, 1])
```

### Evaluation of the model

```
In [ ]: from sklearn.metrics import confusion_matrix,accuracy_score

In [ ]: confusion_matrix(y_test,y_pred)

Out[127]: array([[70,  2],
                 [ 0, 48]], dtype=int64)

In [ ]: accuracy_score(y_test,y_pred)

Out[128]: 0.9833333333333333
```

> As we Performed all the Methods and Trained our Model using different Menthods

**We Got Very Good Accuracy Using XGBoost - 98% Accuracy**

# CHAPTER-10

# ADVANTAGES & DISADVANTAGES

## 10.1 Advantages

1. Increased awareness of CKD may make it easier to take therapeutic measures to slow the deterioration in renal function or avoid CKD-related metabolic problems.
2. In addition, regardless of the need for or type of renal replacement therapy (such as dialysis or transplantation), a uniform disease classification and action plan encompassing all patients may improve patient care continuity.
3. Early detection of chronic kidney disease is the advantage because we can cure in first stage.

## 10.2 Disadvantages

1. CKD has a significant negative impact on quality of life, increased risks of cardiovascular illness, and premature mortality (QoL). According to estimates, CKD patients' mortality from cardiovascular disease (CVD) is at least 8 to 10 times higher than that of non-CKD patients.
2. Your heart and blood vessels may experience problems as a result of chronic kidney disease. Anemia (low red blood cell count) (low red blood cell count) bone issues.

# CHAPTER-11
# CONCLUSION

The advantage of this strategy is that because the prediction process goes much more quickly, doctors can start treating CKD patients as soon as possible and classify a larger population of patients in a shorter amount of time. We would prefer to work with larger datasets in the future or compare the results of this dataset with another dataset that contains the same information because the dataset used in this paper only has 400 examples. Additionally, using the appropriate dataset, we attempt to determine whether a person with this syndrome has a higher chance of developing chronic risk factors like hypertension, a family history of kidney failure, and diabetes in order to reduce the incidence of CKD. Early prediction is very crucial for both the experts and the patients to prevent and slow down the progress of chronic kidney disease to kidney failure.

# CHAPTER-12
# FUTURE SCOPE

This effort will serve as the foundation for the CKD patient healthcare system. This work's extension is that the use of machine learning results in high-caliber performance. It is hoped that this will motivate people to make positive changes in their life and seek early treatment for chronic renal illness. In future an application can be developed with large dataset processing capabilities and with login credentials

# CHAPTER-13

# APPENDIX

## 7.1 Home.html:

```
{% extends 'main.html' %}
{% block content %}
{% if message %}
        <div class="alert alert-danger">{{ message }}</div>
    {% endif %}

<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Kidney Disease Prediction</title>

    <link rel="canonical"
href="https://getbootstrap.com/docs/4.0/examples/carousel/">

    <!-- Bootstrap core CSS -->
    <link href="../../dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
      .jumbotron{
       background-color: bisque;

      }
      .btn{
        color: white;
        background-color: #6cc1c3;
        border-color: #6cc1c3;
      }
      .lead{
        background-image: url(./kidney.jpg);
      }
    </style>

  </head>
```

```
  <body>


    <main role="main">

      <section class="jumbotron p-3 p-md-5 text-black rounded text-center">
        <div class="container">
          <h1 style="font-family: sanserif; " class="jumbotron-heading">Chronic
Kidney Disease Prediction</h1>
          <p class="lead">Chronic kidney disease (CKD) is one of the most
critical health problems due to its increasing prevalence. In this
project, we aim to test the ability of machine learning algorithms
for the prediction of chronic kidney disease using the smallest
subset of features</p>

        </div>
      </section>




        <!-- START THE FEATURETTES -->

        <hr class="featurette-divider">

        <div class="row featurette">
          <div class="col-md-12">
            <h2 style="font-family: sanserif;" class="featurette-heading">How it
will works</h2>
            <p class="lead">This prediction will be used in healthcare
Applications. As it was very important to predict weather the patient was having
any chances of getting this Kidney Disease. </p>
          </div>
        </div>

        <hr class="featurette-divider">

        <div class="row featurette">
          <div class="col-md-12">
            <h2 style="font-family: sanserif;"  class="featurette-heading">Stages
of Disease</h2>
            <p class="lead"><iframe src="https://www.miskawaanhealth.com/wp-
content/uploads/2021/05/chronic-kidney-disease-stages.jpg" title="W3Schools Free
Online Web Tutorials" width="100%"  height="600px"></iframe></p>
          </div>
```

```
        </div>

        <hr class="featurette-divider">

        <div class="row featurette">
          <div class="col-md-12">
            <h2 style="font-family: sanserif;"  class="featurette-heading">Future
implementations</h2>
            <p class="lead">In future an application can be developed with large
dataset processing and capabilities
              more accurate result can produce using this dataset</p>
          </div>
        </div>

        <hr class="featurette-divider">

        <!-- /END THE FEATURETTES -->

      </div><!-- /.container -->
      <section class="jumbotron p-3 p-md-5 text-black rounded text-center">
        <div class="container">
          <h1 class="jumbotron-heading">Chronic Kidney Disease Prediction</h1>
          <p class="lead">Chronic kidney disease (CKD) is one of the most
critical health problems due to its increasing prevalence. In this
paper, we aim to test the ability of machine learning algorithms
for the prediction of chronic kidney disease using the smallest
subset of features</p>
          <p>
            <a href="{{ url_for('kidneyPage') }}" class="btn btn-primary my-
2">Check out the Project</a>
          </p>
        </div>
      </section>


    </main>

    <!-- Bootstrap core JavaScript
    ================================================== -->
    <!-- Placed at the end of the document so the pages load faster -->
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
```

```
    <script>window.jQuery || document.write('<script
src="../../assets/js/vendor/jquery-slim.min.js"><\/script>')</script>
    <script src="../../assets/js/vendor/popper.min.js"></script>
    <script src="../../dist/js/bootstrap.min.js"></script>
    <!-- Just to make our placeholder images work. Don't actually copy the next
line! -->
    <script src="../../assets/js/vendor/holder.min.js"></script>
  </body>
</html>

{% endblock %}
```

## 7.2 Kidney.html

```
{% extends 'main.html' %}
{% block content %}


<div class="row"  style="margin-bottom: 125px;" >
    <div class="col-md-2"></div>
    <div class="col-md-8">

        <center><h1 style="font-family: fantasy;">KIDNEY DISEASE
PREDICTOR</h1></center>
           <center><h3 style="color:red ; text-align: left; font-family:
sansserif; position: relative;" >
                NOTE: <br><span style="color:black ; " >Normal-0,Abnormal-
1</span></h3></center>

        <div class="card card-body" style="border: 1px solid black; ; background-
size: cover; background-color:bisque;">
            <form class="form-horizontal" action="{{ url_for('predictPage') }}"
method="POST">
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                                <h6 style="color: rgb(1, 1, 1);">AGE</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="age" >
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">BLOOD PRESSURE</h6>
```

```
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="bp" >
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="form-group">
                        <h6 style="color: rgb(0, 0, 0);">ALBUMIN</h6>
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="al" >
                    </div>
                </div>
            </div>
            <div class="row">
                <div class="col-md-4">
                    <div class="form-group">
                        <h6 style="color: rgb(0, 0, 0);">SUGAR</h6>
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="su" >
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="form-group">
                        <h6 style="color: rgb(0, 0, 0);">RED BLOOD CELLS</h6>
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="rbc" >
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="form-group">
                        <h6 style="color: rgb(0, 0, 0);">PUS CELL</h6>
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="pc" >
                    </div>
                </div>
            </div>
            <div class="row">
                <div class="col-md-4">
                    <div class="form-group">
                        <h6 style="color: rgb(0, 0, 0);">PUS CELL CLUMPS</h6>
                        <input style="border: 1px solid black;" class="form-
control" type="text" name="pcc" >
                    </div>
                </div>
                <div class="col-md-4">
                    <div class="form-group">
```

```
                            <h6 style="color: rgb(0, 0, 0);">BACTERIA</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="ba" >
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">BLOOD GLUCOSE
RANDOM</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="bgr" >
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">BLOOD UREA</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="bu">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">SERUM
CREATININE</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="sc" >
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">POTASSIUM</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="pot" >
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">WHITE BLOOD CELL
COUNT</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="wc" >
```

```
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">HYPERTENSION</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="htn">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">DIABETIES
MELLITUS</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="dm" >
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">CORONARY ARTERY
DISEASE</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="cad" >
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">PEDAL EDEMA</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="pe" >
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <h6 style="color: rgb(0, 0, 0);">ANEMIA</h6>
                            <input style="border: 1px solid black;" class="form-
control" type="text" name="ane" >
                        </div>
                    </div>
                </div>
                <input type="submit" class="btn btn-info btn-block"
value="Predict" >
            </form>
```

```
        </div>
    </div>
    <div class="col-md-2"></div>
</div> <br>
<h1 style="text-align: center" > Sample-Inputs in the Data Set</h1> <br>

<table class="table" >
  <thead>
    <tr>
      <th scope="col">age</th>
      <th scope="col">bp</th>
      <th scope="col">al</th>
      <th scope="col">su</th>
      <th scope="col">rbc</th>
      <th scope="col">pc</th>
      <th scope="col">pcc</th>
      <th scope="col">ba</th>
      <th scope="col">bgr</th>
      <th scope="col">bu</th>
      <th scope="col">sc</th>
      <th scope="col">pot</th>
      <th scope="col">wc</th>
      <th scope="col">htn</th>
      <th scope="col">dm</th>
      <th scope="col">cad</th>
      <th scope="col">pe</th>
      <th scope="col">ane</th>
      <th scope="col">Disease</th>

    </tr>
  </thead>
  <tbody>
    <tr>

      <td>24</td>
      <td>100</td>
      <td>2</td>
      <td>0</td>
      <td>1</td>
      <td>0</td>
      <td>1</td>
      <td>0</td>
      <td>136</td>
      <td>60</td>
      <td>1.9</td>
```

```
    <td>3.7</td>
    <td>9600</td>
    <td>1</td>
    <td>1</td>
    <td>0</td>
    <td>0</td>
    <td>1</td>
    <td>Present</td>
  </tr>
  <tr>
    <td>68</td>
    <td>80</td>
    <td>3</td>
    <td>0</td>
    <td>0</td>
    <td>1</td>
    <td>0</td>
    <td>0</td>
    <td>157</td>
    <td>162</td>
    <td>9.6</td>
    <td>4.9</td>
    <td>11000</td>
    <td>0</td>
    <td>1</td>
    <td>0</td>
    <td>0</td>
    <td>1</td>
    <td>Present</td>
  </tr>
  <tr>
    <td>51</td>
    <td>0</td>
    <td>0</td>
    <td>0</td>
    <td>1</td>
    <td>0</td>
    <td>0</td>
    <td>0</td>
    <td>121</td>
    <td>27</td>
    <td>0.8</td>
    <td>3.7</td>
    <td>8300</td>
    <td>0</td>
```

```
      <td>0</td>
      <td>0</td>
      <td>0</td>
      <td>0</td>
      <td>Healthy</td>
    </tr>
  </tbody>
</table>

{% endblock %}
```

## 7.3 Main.html

```
<!DOCTYPE html>
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="og:title" content="Kidney-Disease Prediction">
  <meta name="author" content="Venkata Sreeram">
  <meta name="og:image" content="static/logo1.png">
  <meta name="Keywords" content="Flask, Machine Learning, Deep Learning,
Artificial Intelligence, AI, ML,DL, Web Development">
  <meta name="description" content="A Machine Learning and Deep Learning based
webapp for Multiple Disease Prediction.">
  <title>Kidney Disease Predictor</title>
  <link rel="icon" href="{{ url_for('static', filename = 'logo1.png') }}"
type="image/icon type">
  <p> Kidney Disease </p>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet"/>
    <link rel="canonical"
href="https://getbootstrap.com/docs/4.0/examples/sticky-footer/">
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
```

```
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

  <style>
    html, body{ height:100%; margin:0; }
header{ height:50px;}
footer{ height:75px; background:black; }

/* Trick */
body{
  display:flex;
  flex-direction:column;
}

footer{
  padding:10px;
  margin-top:auto;
  margin-bottom: auto;
  background-color:#6cc1c3 ;
}
button{
  border-radius: 13px;
  background-color: bisque;
  border-color: bisque;
  border-style: hidden;

}
button:hover {
  background-color: white;
}

  </style>

</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-dark fixed-top bg-dark"
style="background-color: #6cc1c3 !important; ">
        <a style="text-decoration: none; font-family: fantasy;   color:  bisque"
href="{{ url_for('home') }}"><h1> KIDNEY DISEASE PREDICTION</h1></a>
```

```
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item active">
             <button> <a href="{{ url_for('home') }}" class="nav-link"><h3
style="color: black; font-family:  sansserif  ">Home</h3></a></button>
            </li>
             
            <br>
            <li class="nav-item active">
               <button><a class="nav-link" href="{{ url_for('kidneyPage')
}}"><h3  style=" color: black; font-family: sansserif;
">Predictor</h3></a></button>
             </li>
          </ul>
        </div>


  </nav>
  <br>
<br>
<br>
<br>
<main>
  <div class="container-fluid" style="margin-bottom: 20px;">
  {% block content %}

  {% endblock %}
    </div>
  </main>
  <footer style="color: #6cc1c3;">
    <center>
      <ul style="list-style-type:none;">
      <li style="display: inline;"><a style="color: white;" target="blank">
PNT2022TMID40421 </a></li>
    </ul>
  </center>
  </footer>
</body>
</html>
```

7.4 Predict.html

```
{% extends 'main.html' %}
{% block content %}
    <div class="row"  style="margin-bottom: 477px;">
        <div class="col-md-3"></div>
        <div class="col-md-6">
            {% if pred == 1 %}
                <div class="jumbotron ">
  <h1 class="display-4">You have a Kidney Disease !</h1>
  <p class="lead">Please Consult the Doctor Immideately. It was too risky without
consultation. Make sure of health in your diet.</p>
  <hr class="my-4">
  <p>Proper Doctor Consultation Needed.</p>
  <p class="lead">
    <a class="btn btn-primary btn-lg" href="https://www.who.int/"
role="button">Learn more</a>
  </p>
</div>
            {% else %}
                <div class="jumbotron">
  <h1 class="display-4">Great! You are Healthy</h1>
  <p class="lead">You are Absolutely Alright ! There is no Marks for Kidney
Disease. Enjot=y you life with full of Happiness.</p>
  <hr class="my-4">
  <p>Be careful at your health. Nothing is important than your health.</p>
  <p class="lead">
    <a class="btn btn-primary btn-lg" href="https://www.who.int/"
role="button">Learn more</a>
  </p>
</div>
            {% endif %}
            <div class="row">
                <div class="col-md-4"></div>
                <div class="col-md-4"><a href="{{ url_for('home') }}" class="btn
btn-block btn-primary">Back to Home</a></div>
                <div class="col-md-4"></div>
            </div>
        </div>
        <div class="col-md-3"></div>
```

```
    </div>
{% endblock %}
```

App.py

```
from flask import Flask, render_template, request, flash, redirect
import pickle
import numpy as np
from PIL import Image
from tensorflow.keras.models import load_model

import requests

import json
# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = "9G6L7fYYvmvpB0mI2kqJFm5LyfDvSdIakixARdY-LpZ7"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}



app = Flask(__name__)

def predict(values, dic):
    if len(values) == 18:
        model = pickle.load(open('models/scikit_model.pkl','rb'))
        values = np.asarray(values)
        return model.predict(values.reshape(1, -1))[0]

@app.route("/")
def home():
    return render_template('home.html')



@app.route("/kidney", methods=['GET', 'POST'])
def kidneyPage():
```

```
    return render_template('kidney.html')



@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()
            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list, to_predict_dict)
    except:
        message = "Please enter valid Data"
        return render_template("home.html", message = message)

    return render_template('predict.html', pred = pred)



if __name__ == '__main__':
    app.run(debug = True)

    # NOTE: manually define and pass the array(s) of values to be scored in the
next line
#payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values":
[array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}
payload_scoring = {"input_data": [{"field": [['white blood cell count', 'blood
urea', 'blood glucose random','serum creatinine', 'packed cell volume',
'albumin', 'haemoglobin','age', 'sugar', 'ypertension']],
  "values": [[7800.0,36.0,121.0,1.2,44.0,1.0,15.4,48.0,0.0,1]]}]}
response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/b2d8555c-0595-43d8-abda-
40bf8a6796d5/predictions?version=2022-11-09', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
predictions = response_scoring.json()
print(predictions)
```

New.py

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = "9G6L7fYYvmvpB0mI2kqJFm5LyfDvSdIakixARdY-LpZ7"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next
line
#payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values":
[array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}
payload_scoring = {"input_data": [{"field": [['white blood cell count', 'blood
urea', 'blood glucose random','serum creatinine', 'packed cell volume',
'albumin', 'haemoglobin','age', 'sugar', 'ypertension']],
  "values": [[7800.0,36.0,121.0,1.2,44.0,1.0,15.4,48.0,0.0,1]]}]}
response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/b2d8555c-0595-43d8-abda-
40bf8a6796d5/predictions?version=2022-11-09', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
predictions = response_scoring.json()
print(predictions)
```

Github https://github.com/IBM-EPBL/IBM-Project-1699-1658410055

Demo Link: