

# **Project Report**

## **1. INTRODUCTION**

### **1.1 Project Overview**

Chronic kidney disease is detected during the screening of people who are known to be in threat by kidney problems, such as those with high blood pressure or diabetes and those with a blood relative Chronic Kidney Disease(CKD) patients. So the early prediction is necessary in combating the disease and to provide good treatment.

### **1.2 Purpose**

Chronic Kidney Disease (CKD) is considered as an important threat for the society with respect to the health in the present era. This project predicts whether the person is having CKD or not by using minimum number of features.

## **2. LITERATURE SURVEY**

### **2.1 Existing problem**

The kidney prediction is a costly and important clinical event with substantial morbidity, it appears less frequently in aging people compared with cardiovascular mortality. The measurement of kidney function and management of kidney disease in older individuals remain challenging now a days. So this project will help the person to predict the disease in online and it can be predicted as early as possible.

### **2.2 References**

#### **1. Prediction of chronic kidney disease using different classification algorithms.**

**Khaled Mohamad Almustafa.**

**Informatics in Medicine Unlocked 24 (2021) 100631.**

In this study, different classifiers were applied for the classification of a CKD dataset. The algorithms were applied using random tree, decision table (DT), K-nearest neighbor (K-NN), J48, stochastic gradient descent (SGD) and Naïve Bayes classifiers, and a prediction model was proposed based on feature selection to efficiently predict CKD cases. Results showed that the J48 and decision table classifiers outperformed the other classifiers with accuracies of 99%, ROCs equal to 0.999 and 0.992, MAEs of 0.0225 and 0.1815, and RMSEs of 0.0807 and 0.2507, respectively. A sensitivity analysis of selected classifiers was implemented to evaluate the performance of these classifiers with changes in their parameters. Additionally, the results showed an enhanced classification performance for K-NN (K = 1). Naïve Bayes and decision table classification were enhanced to 99.75%, 98.25% and 99.25%, respectively.

#### **2. Analysis of the performance of feature optimization techniques for the diagnosis of machine learning-based chronic kidney disease.**

**Muhammad Minoar Hossain a, Reshma Ahmed Swarna a, Rafid Mostafiz b, Pabon Shaha a, Lubna Yasmin Pinky a, Mohammad Motiur Rahman a, Wahidur Rahman c, Md. Selim Hossain d, Md. Elias Hossain e, Md. Sadiq Iqbal f.**  
**Machine Learning with Applications 9 (2022) 100330**

In this paper, research analysis of several feature optimization approaches along with a max voting ensemble model to establish a highly accurate CKD diagnosis system by using an appropriate feature set. The ensemble model of this research is structured with five existing classifiers. Three types of feature optimization namely feature importance, feature reduction, and feature selection where for each approach two most proficient techniques are analyzed with the mentioned ensemble model. Based on all analysis the research gets a feature optimization technique called Linear discriminant analysis belonging to the feature selection approach provides the most outstanding result of 99.5% accuracy by using 10-fold cross validation.

### **3. Neural network and support vector machine for the prediction of chronic kidney disease: A comparative study.**

**Njoud Abdullah Almansour, Hajra Fahim Syed, Nuha Radwan Khayat, Rawan Kanaan Altheeb, Renad Emad Juri, Jamal Alhiyafi, Saleh Alrashed, Sunday O. Olatunji.**  
**Computers in Biology and Medicine 109 (2019) 101–111**

In this paper, A dataset of 400 patients and 24 attributes related to diagnosis of chronic kidney disease was used. The classification techniques used in this study include Artificial Neural Network (ANN) and Support Vector Machine (SVM). To perform experiments, all missing values in the dataset were replaced by the mean of the corresponding attributes. Then, the optimized parameters for the Artificial Neural Network (ANN) and Support Vector Machine (SVM) techniques were determined by tuning the parameters and performing several experiments. The empirical results from the experiments indicated that ANN performed better than SVM, with accuracies of 99.75% and 97.75%, respectively, indicating that the outcome of this study is very promising.

### **4. Explainable prediction of chronic renal disease in the colombian population using neural networks and case-based reasoning.**

**Vásquez-Morales, Gabriel R., et al.**  
**Ieee Access 7 (2019): 152900-152910**

In this paper, demographic data of two different populations: people diagnosed with CKD and others without CKD is used. A neural network-based classifier is used to predict whether a person is at a risk of developing CKD. The model achieved an accuracy of 95%. A CaseBased Reasoning(CBR) is used as a twin system for the proposed paradigm for the

explanation of CKD predictions. The system was also used to test on population, where 7% of the total population in Colombia were identified as being at risk of developing CKD.

#### **5. Detection and diagnosis of chronic kidney disease using deep learning-based heterogeneous modified artificial neural network.**

**Fuzhe Ma, Tao Sun , Lingyun Liu , Hongyu Jing.**

**Future Generation Computer Systems 111 (2020) 17-26**

In this paper, Heterogeneous Modified Artificial Neural Network (HMANN) has been proposed for the early detection, segmentation, and diagnosis of chronic renal failure on the Internet of Medical Things (IoMT) platform. Furthermore, the proposed HMANN is classified as a Support Vector Machine and Multilayer Perceptron (MLP) with a Backpropagation (BP) algorithm. The proposed algorithm works based on an ultrasound image which is denoted as a preprocessing step and the region of kidney interest is segmented in the ultrasound image. In kidney segmentation, the proposed HMANN method achieves high accuracy and significantly reducing the time to delineate the contour.

#### **6. A machine learning methodology for diagnosing chronic kidney disease.**

**Qin, Jiongming, et al.**

**IEEE Access 8 (2019): 20991-21002.**

In this paper, the CKD dataset was from the UCI which has a large number of missing values was used. KNN imputation was used to fill the missing values. Six machine learning algorithms (logistic regression, random forest, support vector machine, k-nearest neighbour, Naive Bayes classifier and feed forward neural network) was developed for the analysis of prediction. The random forest model achieved the best performance with 99.75% accuracy. An integrated model by combining both logistic regression and random forest was also built. This model achieved an accuracy of 99.83% after ten times of simulation.

#### **7. Prediction of chronic kidney disease using machine learning algorithm.**

**Tekale, Siddheshwar, et al.**

**International Journal of Advanced Research in Computer and Communication Engineering 7.10 (2018): 92-96.**

In this paper, the data of 400 patients with 24 attributes related to CKD was used. Only the 14 optimal attributes for the prediction of CKD were considered. The various machine learning models like Decision Tree, and SVM was built and compared. The SVM achieved an accuracy of 96.75% while the Decision Tree algorithms with 91.75%. SVM performs better than the Decision Tree algorithms but it is time-consuming the Decision Tree algorithms.

**8. Generating comparative analysis of early Stage prediction of Chronic Kidney Disease. Rubini, L. Jerlin, and P. Eswaran.**

**International Journal of Modern Engineering Research (IJMER) 5.7 (2015): 49-55.**

In this paper, a new chronic kidney disease dataset with three classifiers such as radial basis function network, multilayer perceptron, and logistic regression was proposed. Highest accuracy is achieved by MLP(99.75%) RBF Network followed by RBK Network(98.5%) and then Logistic Regression(97.5%). Error rate, sensitivity, specificity, F-score and kappa values are also predicted using these classifiers. Among all these classifiers Multilayer perceptron classifier gave good accuracy.

**9. A soft computing approach to kidney diseases evaluation.**

**Neves, José, et al.**

**Journal of medical systems 39.10 (2015): 1-9.**

In this paper, the dataset consists 24 attributes, forming five main categories. A hybrid decision support system is developed by Neves and his team, allowed to consider incomplete, unknown, and even contradictory information. This is complemented with an approach to computing centered on Artificial Neural Networks, in order to weigh the Degree-of Confidence in terms of reasoning procedures and knowledge representation based on Logic Programming. Their study involved 558 patients with an age average of 51.7 years and the chronic kidney disease was observed in 175 cases.

**10. A classification of ckd cases using multivariate kmeans clustering.**

**Dubey, Abhinandan.**

**International Journal of Scientific and Research Publications 5.8 (2015): 1-5.**

In this paper, an adopted K-means Clustering algorithm with a single mean vector of centroids, to classify and make clusters of varying probability of likeliness of suspect being prone to CKD. They observed and stated that the suspects falling in clusters K1 or K3 are surely suffering from CKD. The probability of a suspect lying in K2 cluster to fall in the class of CKD is 0.50545, which implies that the suspect cannot be classified by their L-factor classifier. However, suspects from clusters K1 & K3 were found to be falling in CKD class with full probability.

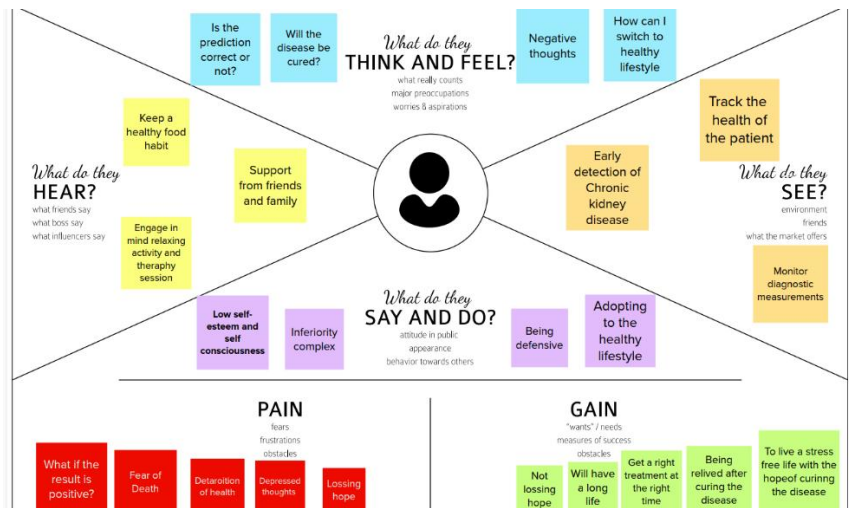
## **2.3 Problem Statement Definition**

Kidney diseases affect the normal function of the kidney. Early prediction of kidney disease using classification and regression algorithms are effective that can help the doctors to diagnose the disease within a short duration of time as it is a complex task for the doctors to predict early. The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best classifier for determining the kidney disease. This Project examines data from kidney patients concentrating on relationships kidney enzymes, proteins, age and gender using

them to try and predict the likeliness of kidney disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask-based web application.

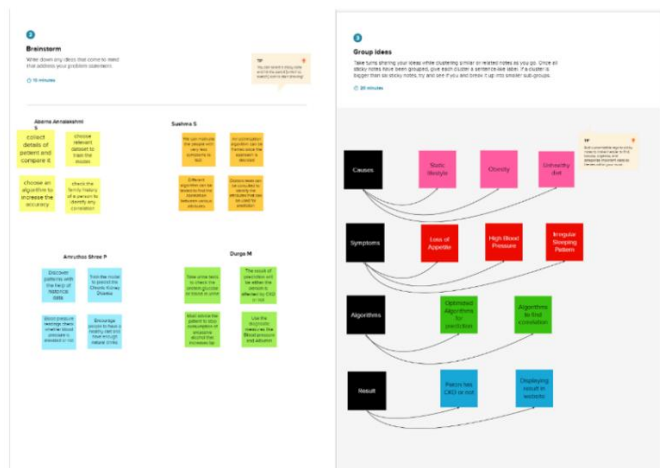
### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas



#### 3.2 Ideation & Brainstorming

##### Brainstorm, Idea Listing and Grouping



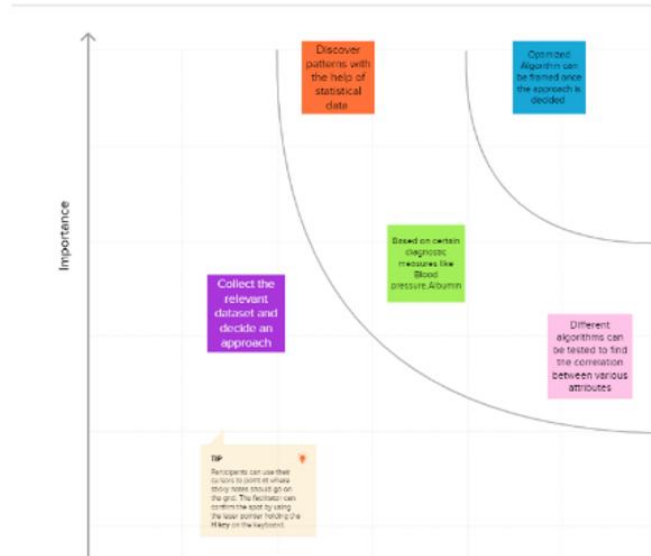
## Idea Prioritization

4

### Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes



## 3.3 Proposed Solution

### Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

| S.No. | Parameter                                | Description   |
|-------|--|---|
| 1.    | Problem Statement (Problem to be solved) | Predicting whether the person is affected by early chronic kidney disease due to issues like high blood pressure, diabetes, high sugar level etc. This may even lead to the death of person if it's found in later stage. So there is need to predict in advance. |
| 2.    | Idea / Solution description              | The parameters that are collected from the user can be analysed with the trained models to predict the chronic kidney disease in advance. To solve this problem, models are developed using machine learning algorithms to make prediction with higher accuracy.  |
| 3.    | Novelty / Uniqueness                     | The model is focused on various parameters which helps to predict the person is affected or not to ensure the high accuracy.  |
| 4.    | Social Impact / Customer Satisfaction    | The main goal to help the person to know whether they are affected by chronic kidney disease or not. Thus, it is necessary to provide higher accuracy results.  |
| 5.    | Business Model (Revenue Model)           | The customers get attracted to the application as they can know their condition from their place instead of travel over a distance to get the results. As all the parameters are analysed, they provide accurate results.   |
| 6.    | Scalability of the Solution              | Through the analysis of all the data and solutions, our solution is scalable. We could optimize it by changing the parameters to predict the disease.   |

### 3.4 Problem Solution fit

|                        |  |   |   |                           |
|------------------------|--|---|---|---------------------------|
| Define CS, fit into CC | 1. CUSTOMER SEGMENT(S)<br>✓ Patients. <b>CS</b>  | 6. CUSTOMER CONSTRAINTS<br>✓ Health details<br>✓ cash <b>CC</b>   | 5. AVAILABLE SOLUTIONS<br>✓ With available health and family health background like blood pressure, age, cardiovascular disorder, family disease, etc., <b>AS</b>     | Explore AS, differentiate |
|                        | 2. JOBS-TO-BE-DONE / PROBLEMS<br>✓ To predict the kidney disease from the available test samples. <b>J&amp;P</b> | 9. PROBLEM ROOT CAUSE<br>✓ Due to the delay in prediction of kidney disease in the available solution, we tend to predict the kidney disease. <b>RC</b> | 7. BEHAVIOUR<br>✓ Directly related: find the kidney disease at the right time.<br>✓ Indirectly associated: patients will have a relaxation and inner peace. <b>BE</b> |                           |

|                        |  |  |   |  |
|------------------------|--|--|---|--|
| Identify strong & weak | 3. TRIGGERS<br>✓ Fast and efficient application<br>✓ User-friendly<br>✓ Early prediction compared to other application leads to faster recovery. <b>TR</b> | 10. YOUR SOLUTION<br>In our application, we get the patient details like blood pressure, diabetes, as inputs and as output we predict whether the kidney disease is detected or not. <b>SL</b> | 8. CHANNELS of BEHAVIOUR<br>8.1 ONLINE<br>They can search the website for early kidney disease prediction.<br>8.2 OFFLINE<br>What kind of actions do customers take offline? They bring their test samples. <b>CH</b> |  |
|                        | 4. EMOTIONS: BEFORE / AFTER<br>✓ Fear, insecure > confident, in control, satisfactory. <b>EM</b>   |  |   |  |

## 4. REQUIREMENT ANALYSIS

### 4.1 Functional requirement

#### Functional Requirements:

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 Form.   |
| FR-2   | User Confirmation             | Confirmation via retyping password.  |
| FR-3   | Obtain Information            | The system should be able to get the information for predicting the disease from the user. |
| FR-4   | Displaying Result             | The system must be able to display whether the user is affected or not.                    |

## 4.2 Non-Functional requirements

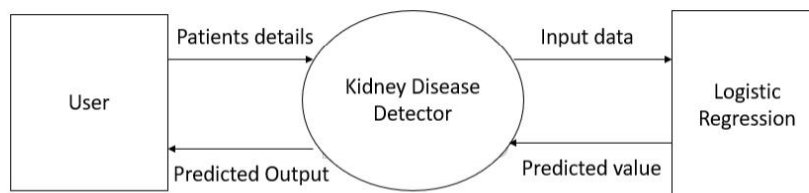
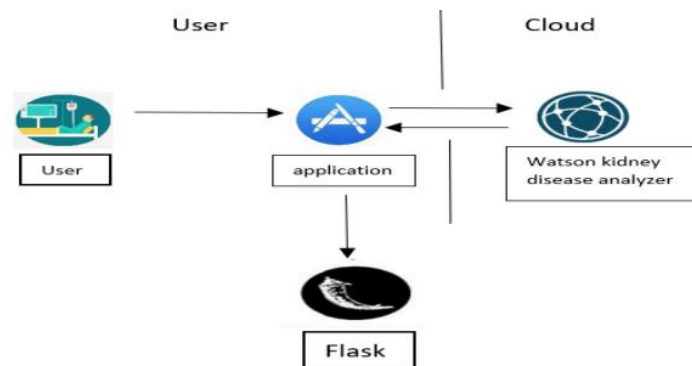
### Non-functional Requirements:

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

| FR No. | Non-Functional Requirement | Description  |
|--------|----------------------------|--|
| NFR-1  | <b>Usability</b>           | Access to use the application is permitted only to the registered users.         |
| NFR-2  | <b>Security</b>            | Authentication is done for security process.                                     |
| NFR-3  | <b>Reliability</b>         | The user gets the correct and predicted value and standard results.              |
| NFR-4  | <b>Performance</b>         | The user gets the results faster accessing the application from remote location. |
| NFR-5  | <b>Availability</b>        | The application is accessible only when the user is online.                      |
| NFR-6  | <b>Scalability</b>         | This application can be used anywhere as it is portable(ie.computer,laptop etc). |

## 5. PROJECT DESIGN

### 5.1 Data Flow Diagrams





## 5.2 Solution & Technical Architecture

Table-1 : 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 fed as input to model to predict the disease | Machine Learning with python |
| 4.   | Update prediction result        | The result of prediction is updated in web UI  | Python                       |
| 5.   | Database                        | Data Type-Numeric  | MySQL                        |
| 6.   | Cloud Database                  | Database Service on Cloud  | IBM Cloudant                 |
| 7.   | Machine Learning Model          | To predict the early chronic kidney disease using various input parameters           | Logistic Regression          |
| 8.   | Infrastructure (Server / Cloud) | Application Deployment Cloud   | IBM Cloud                    |

## 5.3 User Stories

### User Stories

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

| Functional Requirement (Epic) | User Story Number | User Story / Task                                    | Acceptance criteria  | Priority | Release  |
|-------------------------------|-------------------|--|--|----------|----------|
| Accepts/Enable user's input   | USN-1             | As a user, I can enter into the application.         | I can access the application.  | High     | Sprint-1 |
|                               | USN-2             | As a user, I can give the inputs to the application. | The system must accept the values (if all the entered values are correct). | High     | Sprint-1 |

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Delivery Schedule

|         |   |             |
|---------|---|-------------|
| Sprint1 | Data collection and preprocessing               | 30 oct 2022 |
| Sprint2 | Data preprocessing and visualization            | 5 nov 2022  |
| Sprint3 | Model Building and dumping pkl file             | 12 nov 2022 |
| Sprint4 | Web page creation using flask and do prediction | 19 nov 22   |

## 7. CODING & SOLUTIONING (Explain the features added in the project along with code)

Prediction.ipynb

```

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import roc_curve, auc, confusion_matrix, classification_report, accuracy_score
from sklearn.linear_model import LogisticRegression
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv("C:/Users/Widnows/Desktop/ibm/ibm deployment/chronickidneydisease.csv")
df.head()
df[df.duplicated()]
df.isna().sum()
df2 = df.dropna(axis=0)
df2
df[['htn','dm','cad','pe','ane']] = df[['htn','dm','cad','pe','ane']].replace(to_replace={'yes':1,'no':0})
df[['rbc','pc']] = df[['rbc','pc']].replace(to_replace={'abnormal':1,'normal':0})
df[['pcc','ba']] = df[['pcc','ba']].replace(to_replace={'present':1,'notpresent':0})
df[['appet']] = df[['appet']].replace(to_replace={'good':1,'poor':0,'no':np.nan})
df['classification'] = df['classification'].replace(to_replace={'ckd':1.0,'ckd\t':1.0,'notckd':0.0,'no':0.0})
df.rename(columns={'classification':'class'},inplace=True)
df['pe'] = df['pe'].replace(to_replace='good',value=0) # Not having pedal edema is good
df['appet'] = df['appet'].replace(to_replace='no',value=0)
df['cad'] = df['cad'].replace(to_replace='tno',value=0)
df['dm'] = df['dm'].replace(to_replace={'\tno':0,'\tyes':1,' yes':1,":np.nan"})
corr_df = df2.corr()

# Generate a mask for the upper triangle
mask = np.zeros_like(corr_df, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.title('Correlations between different predictors')
plt.show()
catcols=set(df2.dtypes[df2.dtypes=='O'].index.values)

```

Catcols

for i in catcols:

```
print("LABEL ENCODING OF:",i)
LEi=LabelEncoder()
print(df2[i])
df2[i]=LEi.fit_transform(df2[i])
print(df2[i])
print("*"*100)
```

```
X_train, X_test, y_train, y_test = train_test_split(df2.iloc[:, :-1], df2['classification'], test_size = 0.33,
random_state=44, stratify= df2['classification'] )
```

```
X_train.head()
```

```
X_train.info()
```

```
from sklearn.linear_model import LogisticRegression
```

```
lgr=LogisticRegression()
```

```
lgr.fit(X_train,y_train)
```

```
y_pred=lgr.predict(X_test)
```

## 8. TESTING

### 8.1 Test Cases

| Test case id | Feature Type | Component | Test scenario                            | Steps to execute   | Test Data              | Expected result                       |
|--------------|--------------|-----------|--|--|------------------------|---------------------------------------|
| 1            | Functional   | Home Page | Verify user is able to see the home page | 1.Enter URL and click go<br>2.Check whether the home page is visible to user                                       | https://shopenzer.com/ | Home page is visible                  |
| 2            | UI           | Home Page | Verify all the inputs are provided       | 1.Enter URL and click go<br>2.Check whether the home page is visible to user<br>3.Enter all the datas are provided | https://shopenzer.com/ | All the fields are filled with inputs |
| 3            | Functional   | Home Page | Verify whether the                       | 1.Enter URL and click go   | Age:55<br>bp:122       | User should                           |

|   |            |              |  |   |                               |                                   |
|---|------------|--------------|--|---|-------------------------------|-----------------------------------|
|   |            |              | valid inputs are provided                                      | 2.Check whether the home page is visible to user<br>3.Enter all the datas are provided<br>4.Check whether valid inputs are provided | sugar:134                     | navigate to user account homepage |
| 4 | Functional | predict Page | Verify whether the prediction page is navigated from home page | 1.Enter all the fields in home page<br>2.Click Enter to predict   | Age:55<br>bp:122<br>sugar:134 | Navigate and show prediction      |

## 8.2 .User Acceptance Testing

### 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      | 10         | 4          | 2          | 3          | 20       |
| Duplicate      | 1          | 0          | 3          | 0          | 4        |
| External       | 2          | 3          | 0          | 1          | 6        |
| Fixed          | 11         | 2          | 4          | 20         | 37       |
| Not Reproduced | 0          | 0          | 1          | 0          | 1        |
| Skipped        | 0          | 0          | 1          | 1          | 2        |
| Won't Fix      | 0          | 5          | 2          | 1          | 8        |
| Totals         | 24         | 14         | 13         | 26         | 77       |

## 4. 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  | 51          | 0          | 0    | 51   |
| Security            | 2           | 0          | 0    | 2    |
| 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    |

## 9. RESULTS

### 9.1 Performance Metrics

The Early chronic kidney disease is predicted with good accuracy using the attributes like blood pressure, hemoglobin, artery disease, age, albumin, rbc count, wbc count etc.

## 10. ADVANTAGES & DISADVANTAGES

The advantage is the early chronic kidney is predicted with few features with good accuracy.

## 11. CONCLUSION

Thus the website implementation using flask and python was implemented and used for getting the inputs. The prediction of the disease is made successfully.

## 12. FUTURE SCOPE

The suggestions can be made available.

## 13. APPENDIX

Source Code

### HTML files

#### Index.html

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>kidneydisease</title>
<style>
body {
background: linear-gradient(
```

```
    rgba(209, 12, 12, 0.75),
    rgba(217, 255, 3, 0.75)),
```

```
url('https://pharmanewsintel.com/images/site/article_headers/_normal/Chronic_Disease_Manage.png');
    background-repeat: no-repeat;
```

```
}.container {
    border: 2px solid #ccc;
    padding: 10px;
    width: 20em;
    height: 21em;
    background-color: white;
}
```

```
table.center {
    margin-left: auto;
    margin-right: auto;
}
```

```
.font{
    font-weight: bold;
    font-size: larger;
}
```

```
.hello{
    opacity: 0.5;
}
```

```
</style>
</head>
```

```
<body>
```

```
<center><p style="font-size:60px;color:aliceblue;">Chronic Disease Prediction</p></center>
```

```
<table width="500" height="100" class="center">
```

```
<form action="/val" method="post"><center>
```

```
<tr class="font">
```

```
<td><label for="age">Age:</label><br></td>
```

```
<td><input type="number" id="age" name="age">
```

```
</tr>
```

```
<tr class="font">
```

```
<td><label for="bp">Blood Pressure:</label><br>
```

```
<td><input type="number" id="bp" name="bp">
</tr>
<tr class="font">
  <td><label for="sg">Urinary Specific Gravity:</label><br></td>
  <td><input type="number" id="sg" name="sg"></td>
</tr>
<tr class="font">
  <td><label for="al">albumin:</label><br></td>
  <td><input type="number" id="al" name="al"></td>
</tr>
<tr class="font">
  <td> <label for="su">sugar:</label><br></td>
  <td><input type="number" id="su" name="su"></td>
</tr>
<tr class="font">
  <td>
    <label for="rbc">Red Blood Cells:</label><br></td>
  <td> <input type="text" id="rbc" name="rbc"></td>
</tr>
<tr class="font">
  <td> <label for="pc">Pus cell:</label><br></td>
  <td> <input type="text" id="pc" name="pc"></td>
</tr>
<tr class="font">
  <td><label for="pcc">Pus cell clumps:</label><br></td>
  <td><input type="text" id="pcc" name="pcc"></td>
</tr>
<tr class="font">
  <td><label for="ba">Bacteria:</label><br></td>
  <td> <input type="text" id="ba" name="ba"></td>
</tr>
<tr class="font">
  <td><label for="bgr">Blood Glucose Random:</label><br></td>
  <td><input type="number" id="bgr" name="bgr"></td>
</tr>
<tr class="font">
  <td><label for="bu">Blood urea:</label><br></td>
  <td><input type="number" id="bu" name="bu"></td>
</tr>
<tr class="font">
  <td><label for="sc">Serum creatinine:</label><br></td>
  <td><input type="number" id="sc" name="sc"></td>
</tr>
<tr class="font">
```

```

        <td><label for="sod">Sodium:</label><br></td>
        <td><input type="number" id="sod" name="sod"></td>
    </tr>
    <tr class="font">
        <td> <label for="pot">Potassium:</label><br></td>
        <td><input type="number" id="pot" name="pot"></td>
    </tr>
    <tr class="font">
        <td><label for="hemo">Hemoglobin:</label><br></td>
        <td><input type="number" id="hemo" name="hemo"></td>
    </tr>
    <tr class="font"> <td><label for="pcv">PackedCellVolume:</label><br></td>
        <td><input type="text" id="pcv" name="pcv"></td>
    </tr>
    <tr class="font">
        <td><label for="wc">Whitebloodcellcount:</label><br></td>
        <td><input type="text" id="wc" name="wc"></td>
    </tr>
    <tr class="font">
        <td><label for="rc">Redbloodcellcount:</label><br></td>
        <td><input type="text" id="rc" name="rc"></td>
    </tr>
    <tr class="font"> <td><label for="htn">Hypertension:</label><br></td>
        <td><input type="text" id="htn" name="htn"></td>
    </tr>
    <tr class="font">
        <td><label for="dm">DiabetesMellitus:</label><br></td>
        <td><input type="text" id="dm" name="dm"></td>
    </tr>
    <tr class="font">
        <td><label for="cad">Coronaryarterydisease:</label><br></td>
        <td><input type="text" id="cad" name="cad"></td>
    </tr>
    <tr class="font"> <td><label for="appet">Appetite:</label><br></td>
        <td><input type="text" id="appet" name="appet"></td>
    </tr>
    <tr class="font"> <td><label for="pe">PedalEdema:</label><br></td>
        <td><input type="text" id="pe" name="pe"></td>
    </tr>
    <tr class="font"> <td> <label for="ane">Aneamia:</label><br></td>
        <td><input type="text" id="ane" name="ane"></td>
    </tr></center>
    <tr class="font">
        <td><center><button type="submit">Check</button></center></td>

```



```
</tr>
</form>
</table>
</body>
</html>
```

### **Rename.html**

```
<html>
<head>
<style>
body {
  background-color: #E6E6FA;
}
</style>
</head>
<body >
<br>
<br>
<br>
<center><h1>{{answer1}}</h1></center>
<br>

</body>
</html>
```

### **Rename2.html**

```
<html>
<head>
<style>
body {
  background-color: #E6E6FA;
}
</style>
</head>
<body >
<br>
<br>
<br>
<center><h1>{{answer1}}</h1></center>
<br>
```

</body>  
</html>

### **Prediction.ipynb**

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import roc_curve, auc, confusion_matrix, classification_report, accuracy_score
from sklearn.linear_model import LogisticRegression
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv("C:/Users/Widnows/Desktop/ibm/ibm deployment/chronickidneydisease.csv")
df.head()
df[df.duplicated()]
df.isna().sum()
df2 = df.dropna(axis=0)
df2
df[['htn','dm','cad','pe','ane']] = df[['htn','dm','cad','pe','ane']].replace(to_replace={'yes':1,'no':0})
df[['rbc','pc']] = df[['rbc','pc']].replace(to_replace={'abnormal':1,'normal':0})
df[['pcc','ba']] = df[['pcc','ba']].replace(to_replace={'present':1,'notpresent':0})
df[['appet']] = df[['appet']].replace(to_replace={'good':1,'poor':0,'no':np.nan})
df['classification'] = df['classification'].replace(to_replace={'ckd':1.0,'ckd\t':1.0,'notckd':0.0,'no':0.0})
df.rename(columns={'classification':'class'},inplace=True)
df['pe'] = df['pe'].replace(to_replace='good',value=0) # Not having pedal edema is good
df['appet'] = df['appet'].replace(to_replace='no',value=0)
df['cad'] = df['cad'].replace(to_replace='tno',value=0)
df['dm'] = df['dm'].replace(to_replace={'\tno':0,'\tyes':1,' yes':1,":np.nan"})
corr_df = df2.corr()

# Generate a mask for the upper triangle
mask = np.zeros_like(corr_df, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)
```

```

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,
             square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.title('Correlations between different predictors')
plt.show()
catcols=set(df2.dtypes[df2.dtypes=='O'].index.values)
Catcols
for i in catcols:
    print("LABEL ENCODING OF:",i)
    LEi=LabelEncoder()
    print(df2[i])
    df2[i]=LEi.fit_transform(df2[i])
    print(df2[i])
    print("***100)

X_train, X_test, y_train, y_test = train_test_split(df2.iloc[:, :-1], df2['classification'], test_size = 0.33,
random_state=44, stratify= df2['classification'] )
X_train.head()
X_train.info()
from sklearn.linear_model import LogisticRegression
lgr=LogisticRegression()
lgr.fit(X_train,y_train)
y_pred=lgr.predict(X_test)

```

### **Flaskfile.ipynb**

```

import numpy as np
import pandas as pd
import flask

from flask import Flask, request, redirect, render_template
import pickle

loaded_reg = pickle.load(open('randomreg_chronic.pkl', 'rb'))
app = Flask(__name__, static_url_path='')
@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')
@app.route('/val', methods=['POST'])

def val():
    test=[]
    if request.method == 'POST':

```

```
test.append(float(request.form["age"]))
test.append(float(request.form["bp"]))
test.append(float(request.form["sg"]))
test.append(float(request.form["al"]))
test.append(float(request.form["su"]))
rb=request.form["rbc"]
if rb==0:
    test.append(1)
else:
    test.append(0)
pc=request.form["pc"]
if pc==0:
    test.append(1)
else:
    test.append(0)
pcc=request.form["pcc"]
if pcc==1:
    test.append(1)
else:
    test.append(0)
ba=request.form["ba"]
if ba==1:
    test.append(1)
else:
    test.append(0)
test.append(float(request.form["bgr"]))
test.append(float(request.form["bu"]))
test.append(float(request.form["sc"]))
test.append(float(request.form["sod"]))
test.append(float(request.form["pot"]))
test.append(float(request.form["hemo"]))
test.append(float(request.form["pcv"]))
test.append(float(request.form["wc"]))
test.append(float(request.form["rc"]))
ht=request.form["htn"]
if ht==1:
    test.append(1)
else:
    test.append(0)
d=request.form["dm"]
if d==1:
    test.append(1)
else:
    test.append(0)
```

```

ca=request.form["cad"]
if ca==1:
    test.append(1)
else:
    test.append(0)
ap=request.form["appet"]
if ap==1:
    test.append(1)
else:
    test.append(0)

p=request.form["pe"]
if p==1:
    test.append(1)
else:
    test.append(0)
an=request.form["ane"]
if an==1:
    test.append(1)
else:
    test.append(0)
print(test)
test_df=pd.DataFrame(test)
test_df=np.array(test_df).reshape(1, -1)

ans2=loaded_reg.predict(test_df)
if int(ans2)==1:
    answer1="You have CHRONIC DISEASE!!!"
    return render_template('rename.html',answer1=answer1)
else:
    answer1="You don't have CHRONIC DISEASE"

    return render_template('rename2.html',answer1=answer1)

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

```

### GitHub & Project Demo Link

<https://drive.google.com/file/d/1XVWApSJ0ltiLZcf14TYzrN5izIPABNZ2/view?usp=sharing>

<https://github.com/IBM-EPBL/IBM-Project-2302-1658469423>

