

**PROJECT TITLE :**

EARLY DETECTION OF CHRONIC KIDNEY  
DISEASE USING MACHINE LEARNING

**TEAM ID :**

PNT2022TMID27851

**TEAM MEMBERS :**

SWAATHI C M

JEEVITHA R

NIVETHA R

MONISHA D

# **1. INTRODUCTION**

## **1.1 PROJECT OVERVIEW**

In this project the chronic kidney disease is predicted in early stages. There will be a take test button for the patients to click if they are

## **1.2 PURPOSE**

The purpose of this project is to predict the chronic kidney disease earlier because it is better to diagnose the disease in early stages. Whenever the patients finds out that they are symptomatic towards the chronic kidney disease they can use this application. Also there is no consultation fee. If the result is positive then it is adviced for the patients to consult the doctor. This also reduces the consultation fee.

## **2. LITERATURE SURVEY**

### **2.1 EXISTING PROBLEM**

Early Detection of Kidney Disease Using ECG Signals Through Machine Learning Based Modelling

### **2.2 REFERENCES**

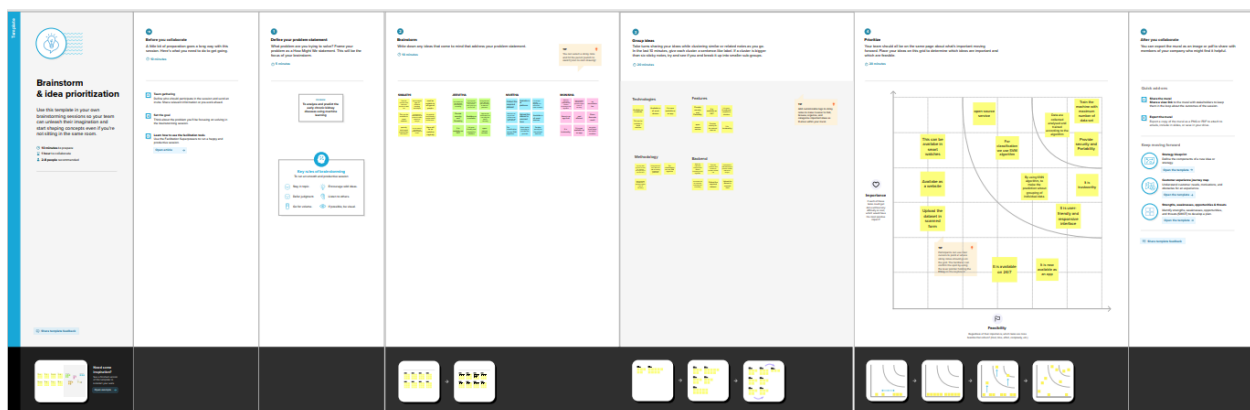
1. "About Chronic Kidney Disease", National Kidney Foundation, 2018, [online] Available: <https://www.kidney.org/atoz/content/about-chronic-kidney-disease>.
2. B. Franczyk-Skóra, A. Gluba, M. Banach, D. Kozłowski, J. Małyszko and J. Rysz, "Prevention of sudden cardiac death in patients with chronic kidney disease", BMC Nephrology, vol. 13, no. 1, 2012.
3. "Global Facts: About Kidney Disease", 2018, [online] Available: <https://www.kidney.org/kidneydisease/global-facts-about-kidney-disease>.
4. "Kidney Disease in Bangladesh", 2018, [online] Available: <http://www.worldlifeexpectancy.com/bangladesh-kidney-disease>.

### **2.3 PROBLEM STATEMENT DEFENITION**



<b>Problem Statement (PS)</b>	<b>I am (Customer)</b>	<b>I'm trying to</b>	<b>But</b>	<b>Because</b>	<b>Which makes me feel</b>
To predict the early chronic kidney diseases using machine learning by analyzing the dataset collected from the patient.	A Patient	know my health condition	it gives inappropriate results	the data is not analyzed properly	worried

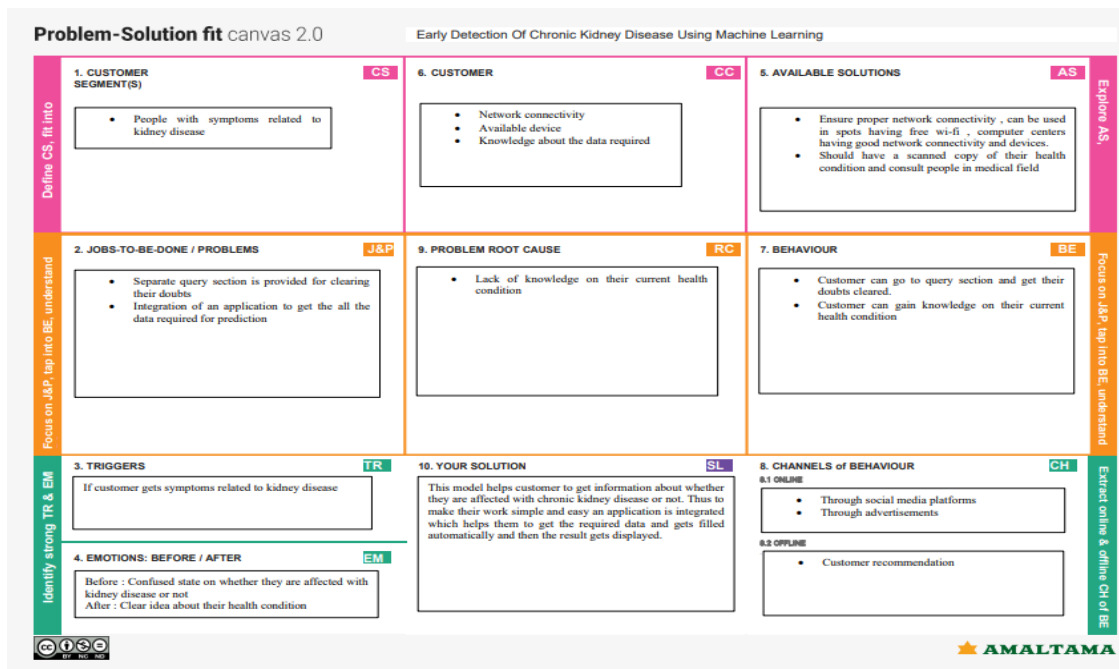
### **3.1 EMPATHY MAP CANVAS**



### 3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To predict the chronic kidney diseases early using machine learning by analyzing the dataset collected from the patient
2.	Idea / Solution description	Early detection of disease is necessary. As there is currently no cure. Thus, by using this application, the disease can be detected early so that the patient can change their lifestyle to control the effects of the disease.
3.	Novelty / Uniqueness	It helps the people to detect the disease earlier.
4.	Social Impact / Customer Satisfaction	It helps people to know about the disease earlier and it is cost efficient
5.	Business Model (Revenue Model)	It is user-friendly. This application is accessible to everyone.
6.	Scalability of the Solution	This application can be used anytime in convenience to the user.

### 3.4 PROBLEM SOLUTION FIT



# 4.REQUIREMENT ANALYSIS

## 4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home page	<ul style="list-style-type: none"><li>• Chronic Kidney disease description</li><li>• Symptoms of the disease</li><li>• A button that takes user to the next page</li></ul>
FR-2	User Entry	<ul style="list-style-type: none"><li>• Input form for pre-diagnostic test results</li><li>• The user enter the required data for testing purpose</li></ul>
FR-3	Analyze	<ul style="list-style-type: none"><li>• The data entered by the user has to be analyzed with the existing data set.</li></ul>
FR-4	Result	<ul style="list-style-type: none"><li>• If Positive – Test Result along with the Information about what is to be done next will be displayed.</li><li>• If Negative – Test result along with preventive measures to prevent themselves from getting Chronic Kidney disease will be displayed.</li></ul>
FR-5	Feed back	<ul style="list-style-type: none"><li>• Allows users to submit feedback through a form</li></ul>

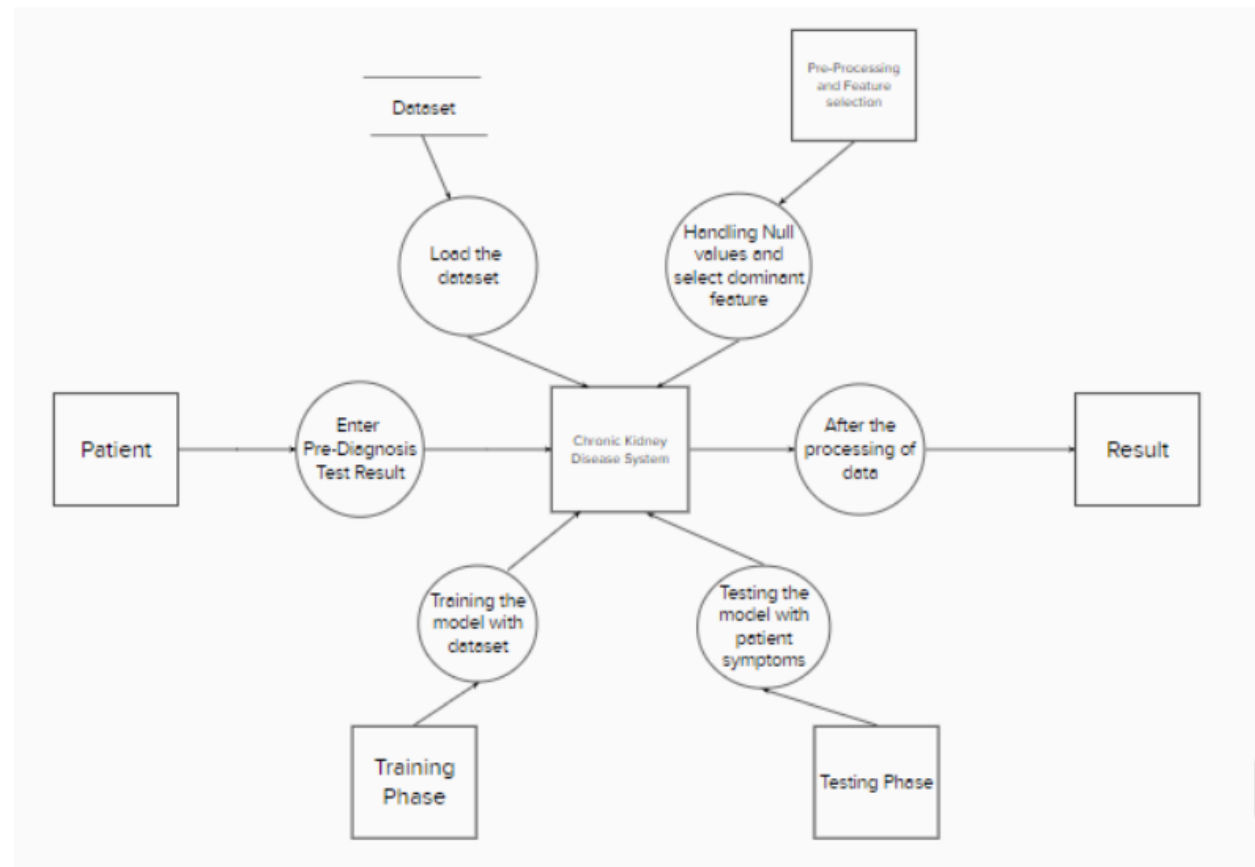
## 4.2 NON-FUNCTIONAL REQUIREMENT

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	<ul style="list-style-type: none"><li>• Its is user friendly for illiterates and people with no understanding of computer, mobile should be able to use the product.</li></ul>
NFR-2	Security	<ul style="list-style-type: none"><li>• the details shared by users and maintain confidentiality</li><li>• the information not shared to others</li></ul>
NFR-3	Reliability	<ul style="list-style-type: none"><li>• The database update process must roll back all related updates when any updates fails.</li></ul>
NFR-4	Performance	<ul style="list-style-type: none"><li>• Reduction in overall time taken for diagnosis</li><li>• The Home-page load time must be no more than 2 seconds</li></ul>
NFR-5	Availability	<ul style="list-style-type: none"><li>• Available at any time to users from various places without any restriction</li></ul>
NFR-6	Scalability	<ul style="list-style-type: none"><li>• Multiple users can use the website at the same time without traffic limit.</li></ul>

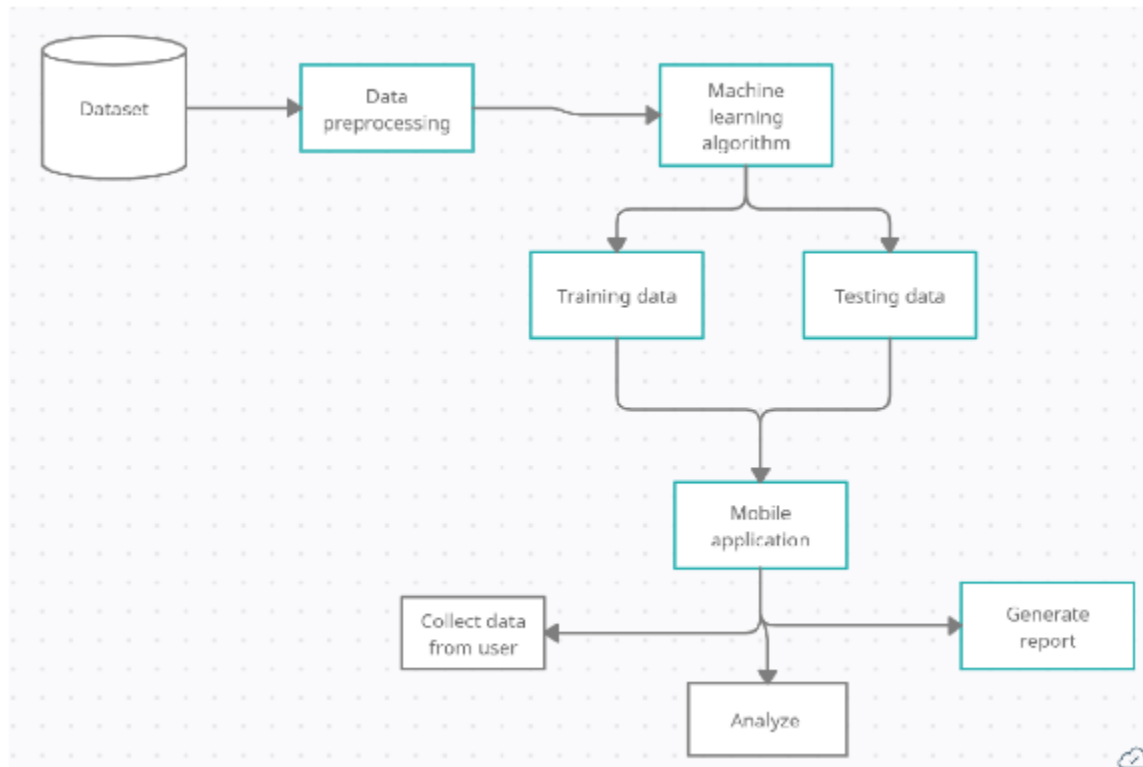


# 5.PROJECT DESIGN

## 5.1 DATA FLOW DIAGRAMS



## 5.2 Solution & Technical Architecture



### 5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for diagnosis tool	I will receive confirmation email	High	Sprint-1
		USN-3	As a user, I can register for the application through the Gmail	I can register & access the dashboard with Gmail Login	Low	Sprint-4
	Login	USN-4	As a user, I can log into the application by entering my credentials	I can login and access past records	Medium	Sprint-1
	Dashboard	USN-5	As a user, I can see my past records and activities	I can access the functionalities diagnosing tool	High	Sprint-3
	Entry form	USN-6	As a user, I must enter my pre-diagnostic test results	I can use the form to input test results	High	Sprint-2
	Report	USN-7	As a user, I can view the report generated by the tool	I can view negative/ positive results produced after diagnosis	High	Sprint-3
Customer Care Executive	Queries	USN-9	As a customer care executive, I must assist users that face problems through Q&A	I will provide 24/7 support for the tool	Low	Sprint-4
	Feedback	USN-10	As a customer care executive, I should get input for the tool's enhancement from users	I must work on improving tool's performance	Low	Sprint-4
Administrator	Feature importance	USN-11	As an administrator, I should identify the most significant factors that lead to CKD based on the present trend	I must identify important features	High	Sprint-2
	Train model	USN-12	As an administrator, I must use the most suitable ML model for detection of CKD	I should efficiently train the ML model	High	Sprint-2

## **6 . PROJECT PLANNING AND SCHEDULING**

### **6.1 SPRINT PLANNING AND ESTIMATION**

<b>TITLE</b>	<b>DESCRIPTION</b>
<b>Sprint 1</b> <b>(24th October to 29th October)</b>	Organize a demonstration session for each sprint, review the code and share your inputs. Evaluate the Code & Features as per the evaluation metrics and save the score for individual students.
<b>Sprint 2</b> <b>(31st October to 5th November)</b>	Organize a demonstration session for each sprint, review the code and share your inputs. Evaluate the Code & Features as per the evaluation metrics and save the score for individual students.
<b>Sprint 3</b> <b>(7th November to 12th November)</b>	Organize a demonstration session for each sprint, review the code and share your inputs. Evaluate the Code & Features as per the evaluation metrics and save the score for individual students.
<b>Sprint 4</b> <b>(14th November to 19th November)</b>	Organize Final evaluation to review the Code, Project report, Project Demonstration and submit the entire score card for each students. Final evaluation of the Code, Project report, Project Demonstration and submit the entire score card for the projects.

## 6.2 SPRINT DELIVERY SCHEDULE

### Product Backlog, Sprint Schedule, and Estimation (4 Marks)

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	Registration	USN-1	As a user, I can register for the diagnosis tool using my email and password	7	High	Team Lead Member 1
Sprint-1		USN-2	As a user, I will receive confirmation email on registering for the diagnosis tool	6	High	Member 1 Member 4
Sprint-4		USN-3	As a user, I can register for the application through my Gmail	6	Low	Member 2 Member 3
Sprint-1	Login	USN-4	As a user, I can log into the application by entering my credentials	6	High	Team Lead Member 1
Sprint-3	Dashboard	USN-5	As a user, I can see my past records and activities	6	High	Team Lead Member 3
Sprint-2	Entry form	USN-6	As a user, I must enter my pre-diagnostic test results	7	High	Team Lead Member 1
Sprint-3	Report	USN-7	As a user, I can view the report generated by the tool	7	High	Member 2 Team Lead

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Remedies	USN-8	As a user, I will receive remedies to treat my symptoms	6	Medium	Member 1 Member 2
Sprint-4	Queries	USN-9	As a customer care executive, I must assist users that face problems through Q&A	6	Low	Member 2 Member 3
Sprint-4	Feedback	USN-10	As a customer care executive, I should get input for the tool's enhancement from users	7	Low	Member 3 Member 4
Sprint-2	Feature importance	USN-11	As an administrator, I should identify the most significant factors that lead to CKD based on the present trend	6	High	Member 2 Member 4
Sprint-2	Train model	USN-12	As an administrator, I must use the most suitable ML model for detection of CKD	6	High	Team Lead Member 4

### Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

## 6.3 REPORTS FROM JIRA

The screenshot shows the 'All sprints' view in Jira. The top navigation bar includes 'Jira Software', 'Your work', 'Projects', 'Filters', 'Dashboards', 'People', 'Apps', and a 'Create' button. A search bar is on the right. The main content area is titled 'All sprints' and shows a Kanban board with four columns: 'TO DO', 'IN PROGRESS', 'IN REVIEW 1 ISSUE', and 'DONE 10 ISSUES'. The 'IN REVIEW 1 ISSUE' column contains two issues: 'API KEY GENERATION' (EDOKDUML-9) and 'As an administrator the project is to be deployed in the cloud platform and api key is to be generated'. The 'DONE 10 ISSUES' column contains three issues: 'As a user, I can view the report generated by the tool' (EDOKDUML-8), 'As a user, I can enter the required data for prediction purpose' (EDOKDUML-4), and 'As a user, I will be able to view the symptoms of chronic kidney disease' (EDOKDUML-10). A 'Quickstart' sidebar on the right provides instructions on how to use Jira.

The screenshot shows the 'Roadmap' view in Jira. The top navigation bar is the same as the previous screenshot. The main content area is titled 'Roadmap' and shows a Gantt chart for the project 'Early Detection Of Chronic Kidney Disease Using Machine Learning'. The chart displays the timeline of the project, with columns for 'TO DO', 'IN PROGRESS', 'IN REVIEW 1 ISSUE', and 'DONE 10 ISSUES'. The 'IN PROGRESS' column contains a list of issues: 'EDOKDUML-12 Data Set', 'EDOKDUML-13 Model Building', 'EDOKDUML-14 Home Page', 'EDOKDUML-15 User Entry', 'EDOKDUML-16 Analysis', 'EDOKDUML-17 Report', 'EDOKDUML-18 API Key Generation', 'EDOKDUML-19 Feature Importance', and 'EDOKDUML-20 Train Model'. A 'Quickstart' sidebar on the right provides instructions on how to use Jira.

Early Detection Of Ch...  
Software project

Back to project

Reports

Overview

Burnup report

Sprint burndown chart

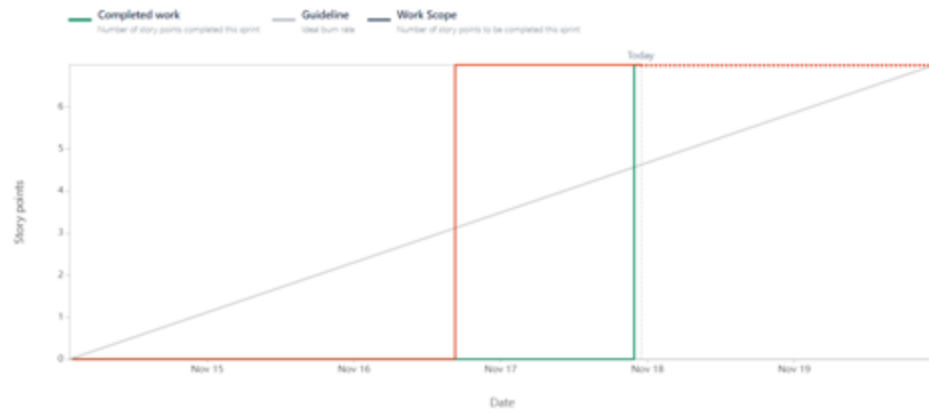
Cumulative flow diagram

Cycle time report

Deployment frequency report

You're in a team-managed project  
[Learn more](#)

Date - November 14th, 2022 - November 19th, 2022



Date Event Issue Completed Scope

# 7. CODING AND SOLUTIONING

## 7.1 FEATURE 1:

HOMEPAGE.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="homepage.css">
  <title>Chronic Kidney Disease</title>
</head>
<body>
  <div class="front">
    
    <h1>CHRONIC KIDNEY DISEASE DETECTION</h1>
    <p>Kidney disease does not tend to cause symptoms when it's at an early stage.
      This is because the body is usually able to cope with a significant reduction in kidney
      function.
      Kidney disease is often only diagnosed at this stage if a routine test for another condition,
      such as a blood or urine test, detects a possible problem.

      If it's found at an early stage, medicine and regular tests to monitor it may help stop it
      becoming more advanced.</p>
      <p><b>SYMPTOMS</b><b>-Nausea and vomiting, muscle cramps, loss of appetite, swelling via
      feet and ankles, dry, itchy skin, shortness of breath, trouble sleeping, urinating either too much
      or too little</b></p>
      <a href="form.html">
        <button type="submit" >TAKE TEST</button>
      </a>
    </div>
  </body>
</html>
```

HOMEPAGE.CSS

```
body{
```



```

        overflow: hidden;
    }
    .background{
        position: absolute;
        left: 0px;
        top: 0px;
        z-index: -1;
        width: 100%;
        height: 100%;
        -webkit-filter: blur(5px);
        filter: blur(5px);
    }
    .front{
        height:100vh;
        width: 100%;
        text-align: center;
        font-size: 200%;
        color: rgb(11, 12, 12);
    }
    button{
        background-color: rgb(192, 171, 171);
        height: 50px;
        width: 200px;
        font-size: large;
        margin-block-end: auto;
        bottom: 50px;
    }

```

**The Homepage consists of the explanation of chronic kidney disease and the symptoms for chronic kidney disease. It also contains a button name take test. The patient has to click the button to take the test.**

## **7.2 FEATURE 2**

```

FORM.HTML
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">

```

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="stylesheet" href="form.css">
<title>Document</title>
</head>
<body>
  <div class="container">
    <h1>Enter the required details</h1>
    <div class="formbox">
      <video autoplay loop muted plays-inline id="myvedio">
        <source src="images/testing.mp4" type="video/mp4">
      </video>
      <form action="result.html" method="GET">
<!-- <section class="image-1">
</section>
<section class="image-2">
</section>
<section class="image-3">
</section>
<section class="image-4">
</section> -->
<!-- Name -->
<div>
  <h3>Name</h3>
  <input type="text" name="fullname" id="fullname">
</div>
<!-- Age -->
<div>
  <h3>Age</h3>
  <input type="number" name="age" id="age" >
</div>
<!-- Blood Pressure -->
<div>
  <h3>Blood Pressure</h3>
  <input type="number" name="bp" id="bp">
</div>
<!-- Urinary Specific gravity -->
<div>
  <h3>Urinary Specific gravity</h3>
  <input type="text" name="sg" id="sg">
</div>
```

```
<!-- Acute leukaemia -->
<div>
  <h3>Acute leukaemia</h3>
  <input type="number" name="al" id="al">
</div>
<!-- Secretory unit -->
<div>
  <h3>Secretory unit</h3>
  <input type="number" name="su" id="su">
</div>
<!-- Red Blood Cells -->
<div>
  <h3>Red Blood Cells</h3>
  <input type="radio" name="rbc" id="rbcn" value="normal">
  <label for="rbcn">Normal</label>
  <input type="radio" name="rbc" id="rbca" value="abnormal">
  <label for="rbca">Abnormal</label>
</div>
<!-- Platelet Count -->
<div>
  <h3>Platelet Count</h3>
  <input type="radio" name="pc" id="pcn" value="normal">
  <label for="pcn">Normal</label>
  <input type="radio" name="pc" id="pca" value="abnormal">
  <label for="pca">Abnormal</label>
</div>
<!-- prothrombin Complex concentrate -->
<div>
  <h3>Prothrombin Complex Concentrate</h3>
  <input type="radio" name="pcc" id="pccp" value="Present">
  <label for="pccp">Present</label>
  <input type="radio" name="pcc" id="pccn" value="Not Present">
  <label for="pccn">Not Present</label>
</div>
<!-- Bronchical Asthma -->
<div>
  <h3>Bronchical Asthma</h3>
  <input type="radio" name="ba" id="bap" value="Present">
  <label for="bap">Present</label>
  <input type="radio" name="ba" id="ban" value="Not Present">
  <label for="ban">Not Present</label>
```

```
</div>
<!-- Blood glucose regulator -->
<div>
  <h3>Blood glucose regulator</h3>
  <input type="number" name="bgr" id="bgr">
</div>
<!-- Bandelette terms -->
<div>
  <h3>Bandelette terms</h3>
  <input type="number" name="bt" id="bt">
</div>
<!-- subcutaneous -->
<div>
  <h3>Subcutaneous</h3>
  <input type="number" name="sub" id="sub">
</div>
<!-- Segregation of duties -->
<div>
  <h3>Segregation of duties</h3>
  <input type="number" name="seg" id="seg">
</div>
<!-- Pot -->
<div>
  <h3>POT</h3>
  <input type="number" name="pot" id="pot">
</div>
<!-- haemoglobin -->
<div>
  <h3>Haemoglobin</h3>
  <input type="number" name="hemo" id="hemo">
</div>
<!-- Pneumococcal conjugate vaccine -->
<div>
  <h3>Pneumococcal conjugate vaccine</h3>
  <input type="number" name="pcv" id="pcv">
</div>
<!-- white blood cell count -->
<div>
  <h3>White blood cell count </h3>
  <input type="number" name="wc" id="wc">
</div>
```

```
<!-- red blood cell count -->
<div>
  <h3>RC</h3>
  <input type="number" name="rc" id="rc">
</div>
<!-- hypertension -->
<div>
  <h3>Hypertension</h3>
  <input type="radio" name="htn" id="htny" value="yes">
  <label for="htny">Yes</label>
  <input type="radio" name="htn" id="htnn" value="no">
  <label for="htnn">No</label>
</div>
<!-- diabetes mellitus -->
<div>
  <h3>DM</h3>
  <input type="radio" name="dm" id="dmy" value="yes">
  <label for="dmy">Yes</label>
  <input type="radio" name="dm" id="dmn" value="no">
  <label for="dmn">No</label>
</div>
<!-- coronary artery disease -->
<div>
  <h3>Coronary artery disease</h3>
  <input type="radio" name="cad" id="cady" value="yes">
  <label for="cady">Yes</label>
  <input type="radio" name="cad" id="cadn" value="no">
  <label for="cadn">No</label>
</div>
<!-- appetite -->
<div>
  <h3>Appetite</h3>
  <input type="radio" name="appet" id="appetg" value="good">
  <label for="appetg">Good</label>
  <input type="radio" name="appet" id="appetp" value="poor">
  <label for="appetp">Poor</label>
</div>
<!-- pulmonary embolism -->
<div>
  <h3>Pulmonary embolism</h3>
  <input type="radio" name="pe" id="pey" value="yes">
```

```

    <label for="cady">Yes</label>
    <input type="radio" name="pe" id="pen" value="no">
    <label for="pen">No</label>
  </div>
  <!-- anemia -->
  <div>
    <h3>Anemia</h3>
    <input type="radio" name="ane" id="aney" value="yes">
    <label for="aney">Yes</label>
    <input type="radio" name="ane" id="anen" value="no">
    <label for="anen">No</label>
  </div>
  <!-- Classification -->
  <div class="btn">
    <button type="submit">Submit</button>
  </div>
</form>
</div>
</div>
</body>
</html>

```

## FORM .CSS

```

input,label {
  cursor: pointer;
  margin: .3rem;
  padding-right: 5rem;
}
input[type=text],[type=number],[type=submit] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border-radius: 4px;
  box-sizing: border-box;
  font-size: 20px;
}
input[type="radio"]:focus {
  height: 1rem;
  width: 1rem;
}

```

```
margin-right: 0.2rem;
margin-left: 10px;
font-size: large;
}
button {
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 10px 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
}
.btn{
  display: flex;
  width: 300px;
  text-align: center;
  justify-content: center;
  padding-top: 20px;
  padding-left: 250px;
  margin-right: 20px;
}
h1{
  text-align: center;
}
form {
  box-sizing: border-box;
  padding: 2rem;
  border-radius: 15rem;
}
.formbox{
  max-width: 800px;
  margin: auto;
  border-radius: 10px;
  display: flex;
  align-items: center;
  box-shadow: 0 0 20px rgba(0, 0, 0, 7);
  color: rgb(14, 1, 1);
  font-size: larger;
  font-weight: 500;
```

```
background:rgba(0, 0, 0, 0.1)
}
#myvedio {
width: 100vw;
height: 100vh;
object-fit: cover;
position: fixed;
left: 0;
right: 0;
top: 0;
bottom: 0;
z-index: -1;
}
```

**In Feature 1 the patient has to enter the details such as red blood cells count, blood pressure etc in the form available. The details has to be entered from the recently scanned report. Entering all the fields in the form is mandatory. After filling the information submit the form.**

### **7.3 FEATURE 3**

positive.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="page3.css">
  <title>Positive</title>
</head>
<body>
  <div class="gws"></div>
  <div class="positive">
    <h1>The result is</h1>
    
    <h2>"ONCE YOU CHOOSE HOPE, ANYTHING'S POSSIBLE."</h2>
    <h2>GET WELL SOON &#129310;</h2>
  </div>
```



```
</body>
</html>
```

positive.css

```
.gws{
  background-image: url("images/BG.jpeg");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  background-attachment: fixed;
  /* background-size: 100% 100%; */
  height: 680px;
  width: 100%;
  filter: blur(8px);
  -webkit-filter: blur(8px);
  filter: blur(8px);
  -webkit-filter: blur(8px);
}

.positive{
  /* display: flex;
  justify-content: center; */
  background-color: rgb(0,0,0); /* Fallback color */
  background-color: rgba(0,0,0, 0.4); /* Black w/opacity/see-through */
  color: rgb(255, 255, 255);
  font-weight: bold;
  border: 3px solid #f0d800;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 2;
  width: 80%;
  padding: 20px;
  text-align: center;
}

.pos{
  height: 100%;
  width: 25%;
}
```

negative.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="page4.css">
  <title>Negative</title>
</head>
<body>
  <div class="gws"></div>
  <div class="positive">
    <h1>The result is</h1>
    
    <h2>"THE GREATEST HEALTH IS WEALTH"</h2>
    <h2>STAY HEALTHY &#128522;</h2>
  </div>
</body>
</html>
```

negative.css

```
.gws{
  background-image: url("images/emojis.gif");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  background-attachment: fixed;
  /* background-size: 100% 100%; */
  height: 680px;
  width: 100%;
  filter: blur(8px);
  -webkit-filter: blur(8px);
  filter: blur(8px);
  -webkit-filter: blur(8px);
}
.positive{
  /* display: flex;
  justify-content: center; */
```

```

background-color: rgb(0,0,0); /* Fallback color */
background-color: rgba(0,0,0, 0.4); /* Black w/opacity/see-through */
color: rgb(255, 255, 255);
font-weight: bold;
border: 3px solid #f1f1f1;
position: absolute;
top: 50%;
left: 50%;
transform: translate(-50%, -50%);
z-index: 2;
width: 80%;
padding: 20px;
text-align: center;
}
.neg{
height: 100%;
width: 25%;
}

```

**From the details entered by the patient in the form the results will be displayed. The result will be either positive or negative.**

#### **7.4 FEATURE 4**

```

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv('chronickidneydisease.csv')
df
df.describe()
print("The size of the dataset is ",df.shape)
print("Datatype of each column : ")
print("-----")
df.info()
df.corr()
sns.heatmap(df.corr())
print("Drop unnecessary column : ")
print("-----")

```

```

df = df.drop(columns = ['id'],axis = 1)
df
df['age'].value_counts()
df['bp'].value_counts()
df['sg'].value_counts()
df['al'].value_counts()
df['su'].value_counts()
df['rbc'].value_counts()
df['pc'].value_counts()
df['pcc'].value_counts()
df['ba'].value_counts()
df['bgr'].value_counts()
df['bu'].value_counts()
df['sc'].value_counts()
df['sod'].value_counts()
df['pot'].value_counts()
df['hemo'].value_counts()
df['pcv'].value_counts()
df['pcv'] = df['pcv'].replace('\t43','43')
df=df[df['pcv'] != '\t?']
df['wc'].value_counts()
df=df[df['wc'] != '\t?']
df['rc'].value_counts()
df=df[df['rc'] != '\t?']
df['htn'].value_counts()
df['dm'].value_counts()
df['dm'].replace(['\tyes','\tno'],['yes','no'],inplace = True)
df['cad'].value_counts()
df['cad'].replace('\tno','no',inplace = True)
df['appet'].value_counts()
df['pe'].value_counts()
df['ane'].value_counts()
df['classification'].value_counts()
df['classification'].replace('ckd\t','ckd',inplace = True)
numerical = ['age','bp','sg','al','sod','pot','hemo','pcv','wc','rc','su','bgr','bu','sc']
df['rc'] = df['rc'].astype(float)
df['wc'] = df['wc'].astype(float)
for x in numerical:
    sns.boxplot(df[x])
    plt.show()
for x in numerical:

```

```

q1 = df[x].quantile(0.25)
q3 = df[x].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df[x] = np.where(df[x]>=upper_limit,df[x].median(),df[x])
df[x] = np.where(df[x]<=lower_limit,df[x].median(),df[x])
for x in numerical:
    sns.boxplot(df[x])
    plt.show()
q1 = df['bgr'].quantile(0.25)
q3 = df['bgr'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['bgr'] = np.where(df['bgr']>=upper_limit,df['bgr'].median(),df['bgr'])
df['bgr'] = np.where(df['bgr']<=lower_limit,df['bgr'].median(),df['bgr'])
sns.boxplot(df['bgr'])
q1 = df['bu'].quantile(0.25)
q3 = df['bu'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['bu'] = np.where(df['bu']>=upper_limit,df['bu'].median(),df['bu'])
df['bu'] = np.where(df['bu']<=lower_limit,df['bu'].median(),df['bu'])
sns.boxplot(df['bu'])
q1 = df['sc'].quantile(0.25)
q3 = df['sc'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['sc'] = np.where(df['sc']>=upper_limit,df['sc'].median(),df['sc'])
df['sc'] = np.where(df['sc']<=lower_limit,df['sc'].median(),df['sc'])
sns.boxplot(df['sc'])
print('CHECKING FOR NULL VALUES : ')
print("-----")
df.isnull().sum()
for x in df.columns:
    if(df[x].dtypes=='float64'):
        df[x].fillna(df[x].median(),inplace = True)
    else:

```

```

df[x].fillna(df[x].mode()[0],inplace = True)
df.isnull().sum()
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['rbc'] = le.fit_transform(df['rbc'])
df['rbc'].value_counts()
df['pc'] = le.fit_transform(df['pc'])
df['pc'].value_counts()
df['pcc'] = le.fit_transform(df['pcc'])
df['pcc'].value_counts()
df['ba'] = le.fit_transform(df['ba'])
df['ba'].value_counts()
df['htn'] = le.fit_transform(df['htn'])
df['htn'].value_counts()
df['dm'] = le.fit_transform(df['dm'])
df['dm'].value_counts()
df['cad'] = le.fit_transform(df['cad'])
df['cad'].value_counts()
df['appet'] = le.fit_transform(df['appet'])
df['appet'].value_counts()
df['pe'] = le.fit_transform(df['pe'])
df['pe'].value_counts()
df['ane'] = le.fit_transform(df['ane'])
df['ane'].value_counts()
df['classification'] = le.fit_transform(df['classification'])
df['classification'].value_counts()
sns.distplot(df['bp'])
sns.distplot(df['sg'])
sns.distplot(df['al'])
sns.barplot(df['appet'].value_counts().index,df['appet'].value_counts())
sns.barplot(df['rbc'].value_counts().index,df['rbc'].value_counts())
sns.lineplot(df['age'],df['sod'])
def countplot_of_2(x,hue,title=None,figsize=(18,5)):
    plt.figure(figsize=figsize)
    sns.countplot(data=df[[x,hue]],x=x,hue=hue)
    plt.title(title)
    plt.show()
countplot_of_2('rc','htn','rc/htn')
sns.pairplot(df)
X = df.drop(columns = ['classification'],axis = 1)
X.head()

```

```

y = df['classification']
y.tail()
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
X_scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
X_scaled
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_scaled,y,test_size=0.25,random_state = 50)
print('X_train: ',X_train.shape)
print('X_test: ',X_test.shape)
print('y_train: ',y_train.shape)
print('y_test: ',y_test.shape)
from sklearn.tree import DecisionTreeClassifier
model1 = DecisionTreeClassifier()
model1.fit(X_train,y_train)
train_pred = model1.predict(X_train)
test_pred = model1.predict(X_test)
from sklearn.metrics import accuracy_score
print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred))
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,test_pred)
pd.crosstab(y_test,test_pred)
from sklearn.metrics import classification_report
classification_report(y_test,test_pred)
from sklearn.metrics import roc_curve,roc_auc_score
probability = model1.predict_proba(X_test)[:,-1]
fpr,tpr,thresholds = roc_curve(y_test,probability)
plt.plot(fpr,tpr)
plt.title('ROC CURVE')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
roc_auc_score(y_test,probability)
parameters = {'criterion':['gini','entropy'],'max_depth':range(1,10),'random_state':[50,60]}
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(model1, parameters,cv = 10,verbose = 1)
grid.fit(X_train,y_train)
grid.best_params_
grid.best_estimator_
grid.best_score_

```

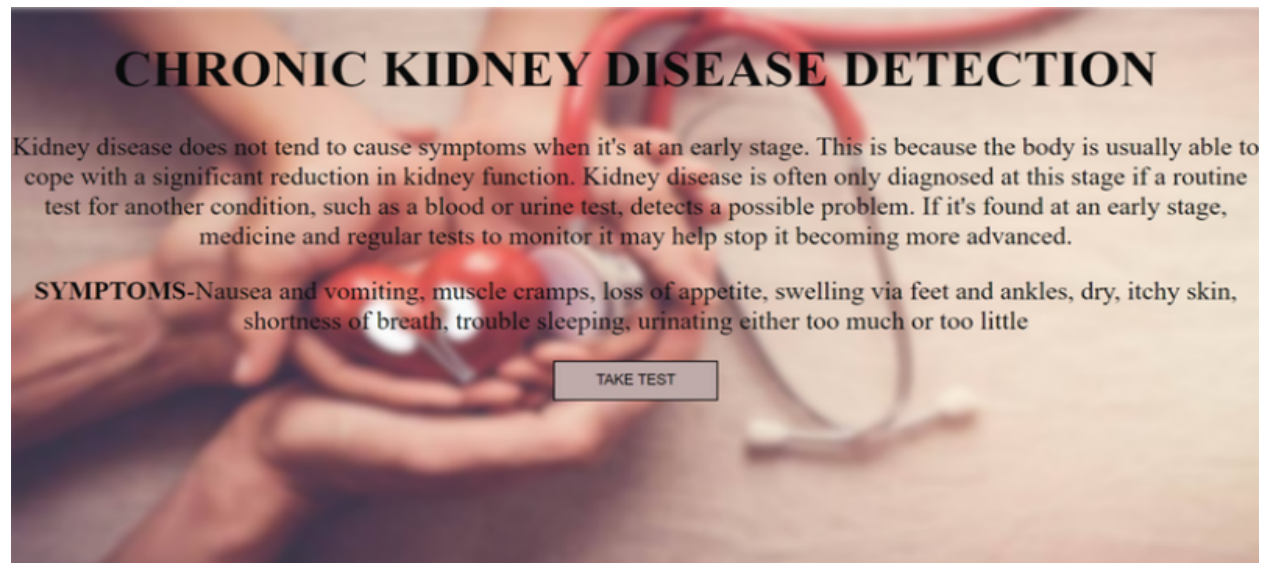
```

model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
model2.fit(X_train,y_train)
train_pred2 = model1.predict(X_train)
test_pred2 = model1.predict(X_test)
print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred2))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred2))
pd.crosstab(y_test,test_pred2)
model1.predict([[80,70,1.02,0,0,1,1,0,0,104,28,1.1,135,4.1,15.3,48,6300,6.1,0,0,0,0,0]])
model2.predict([[80,70,1.02,0,0,1,1,0,0,104,28,1.1,135,4.1,15.3,48,6300,6.1,0,0,0,0,0]])
model1.predict([[48,70,1.005,4,0,1,0,1,0,117,56,3.8,111,2.5,11.2,32,6700,3.9,1,0,0,1,1]])
model2.predict([[48,70,1.005,4,0,1,0,1,0,117,56,3.8,111,2.5,11.2,32,6700,3.9,1,0,0,1,1]])
model1.predict([[47,60,1.02,0,0,1,1,0,0,137,17,0.5,150,3.5,13.6,44,7900,4.5,0,0,0,0,0]])
model2.predict([[47,60,1.02,0,0,1,1,0,0,137,17,0.5,150,3.5,13.6,44,7900,4.5,0,0,0,0,0]])
model1.predict([[73,100,1.01,3,2,0,0,1,0,295,90,5.6,140,2.9,9.2,30,7000,3.2,1,1,1,1,0]])
model2.predict([[73,100,1.01,3,2,0,0,1,0,295,90,5.6,140,2.9,9.2,30,7000,3.2,1,1,1,1,0]])
import joblib
joblib.dump(model1,'ckd_prediction.pkl')

```

**Here the required libraries are imported . Dataset is loaded . Data preprocessing is performed by applying techniques such as removing unnecessary values, outlier replacement , replacing null values , encoding categorical columns. Vizualization is performed . Dependent independent split, scaling , train test split are done. Model building , training of the model , testing of the model is done . Metrics is evaluated . Tuning of the model is done .Using the model the testing for some values is done . Pickle file is generated.**

## OUTPUT:



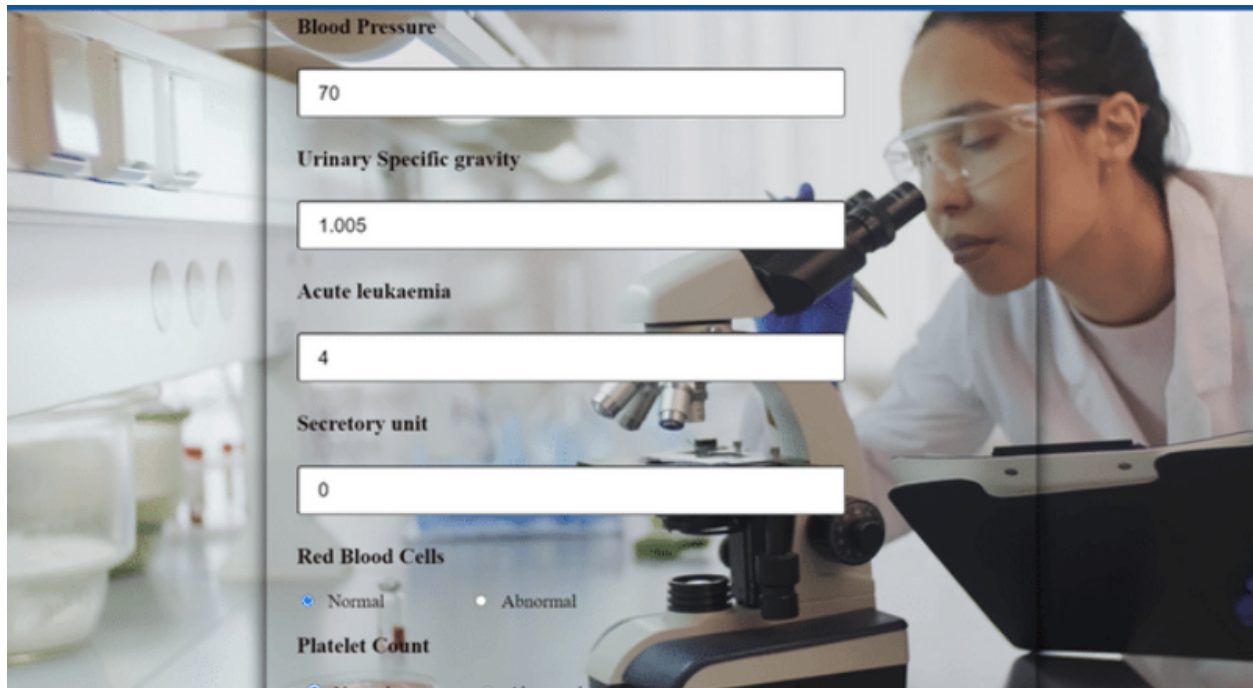
# CHRONIC KIDNEY DISEASE DETECTION

Kidney disease does not tend to cause symptoms when it's at an early stage. This is because the body is usually able to cope with a significant reduction in kidney function. Kidney disease is often only diagnosed at this stage if a routine test for another condition, such as a blood or urine test, detects a possible problem. If it's found at an early stage, medicine and regular tests to monitor it may help stop it becoming more advanced.

**SYMPTOMS**-Nausea and vomiting, muscle cramps, loss of appetite, swelling via feet and ankles, dry, itchy skin, shortness of breath, trouble sleeping, urinating either too much or too little

[TAKE TEST](#)





**Blood Pressure**

70

**Urinary Specific gravity**

1.005

**Acute leukaemia**

4

**Secretory unit**

0

**Red Blood Cells**

☒ Normal ☐ Abnormal

**Platelet Count**

The result is  
**POSITIVE**  
ONCE YOU CHOOSE HOPE, ANYTHING IS POSSIBLE.  
GET WELL SOON 🍀



# 8. TESTING

## 8.1 TEST CASES

	Test Scenarios
1	Verify if user is able to view the homepage.
2	Verify if user is able to navigate to the page which contains form when take test button is clicked.
3	Verify if user is able to enter the all the required details in the form.
4	Verify user is able to view the result when the form is submitted.

## 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	8	3	2	4	17
Duplicate	2	0	1	0	3
External	1	3	0	1	5
Fixed	9	3	2	11	25
Not Reproduced	0	0	1	0	1
Skipped	0	1	0	1	2
Won't Fix	0	4	3	0	7
Totals	20	14	9	17	60

### 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	8	0	0	8
Client Application	42	0	0	42
Security	3	0	0	3
Outsource Shipping	2	0	0	2
Exception Reporting	8	0	0	8
Final Report Output	3	0	0	3
Version Control	2	0	0	2

	A	B	C	D	E	F
1					Date	15-Nov-22
2					Team ID	PNT2022TMD27851
3					Project Name	Early Detection of Chronic Kidney Disease using Machine Learning
4					Maximum Marks	4 marks
5	Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute
6	Homepage_TC_001	UI	Home Page	Verify if the user is able to view the UI of the homepage properly	Internet connection	1.Enter URL and click go 2.Verify if the UI of the homepage is displayed properly.
7	Homepage_TC_002	Functional	Home Page	Verify if user is able to view the form when the take test button is clicked.	Internet connection	1.Enter URL and click go 2.Verify if the UI of the homepage is displayed properly. 3.Verify if the form is displayed when the take test button is clicked.
8	Form_TC_001	Functional	Form page	Verify if the user is able to enter all the required details in the form	Patient's Scan Report	1.Enter URL and click go 2.Verify if the UI of the homepage is displayed properly. 3.Verify if the form is displayed when the take test button is clicked. 4. Check if the user is able to fill all the required details in the form.
9	Form_TC_002	Functional	Form page	Verify if the result is displayed when the form is submitted.		1. Enter URL and click go. 2. Verify if the UI of the form is displayed perfectly. 3. Verify if the user is able to enter all the required details in the form. 4. Verify if the result is displayed when the form is submitted.

Test Data	Expected Result	Actual Result	Status
<a href="http://127.0.0.1:5000">http://127.0.0.1:5000</a>	User should navigate to the homepage.	Working as expected	Pass
	User should navigate to the page where the form will be displayed.	Working as expected	Pass
Values in the scan report that are to be filled in the form for prediction	All the fields in the form has to be filled.	Working as expected	Pass
	The user should be navigated to the result page either positive or negative based on the result.	Working as expected	Pass

# 9.RESULTS

## 9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Metrics	<b>Classification Model:</b> Confusion Matrix – [ [68, 2], [1,29] ]  Accuracy Score- Training accuracy score : 1.0 Testing accuracy score : 0.97  Classification Report – 0.97 , 0.97 , 0.97 , 100	
2.	Tune the Model	Hyperparameter Tuning – <u>GridSearchCV</u>  Validation Method – Training accuracy score : 1.0 Testing accuracy score : 0.97	

### CONFUSION MATRIX

```
In [118]: from sklearn.metrics import confusion_matrix  
confusion_matrix(y_test,test_pred)
```

```
Out[118]: array([[68,  2],  
                [ 1, 29]], dtype=int64)
```

```
In [119]: pd.crosstab(y_test,test_pred)
```

```
Out[119]:
```

	col_0	0	1
classification			
0	68	2	
1	1	29	

## ACCURACY SCORE

```
In [95]: from sklearn.metrics import accuracy_score
print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred))
```

```
TRAINING ACCURACY : 1.0
TESTING ACCURACY : 0.97
```

## CLASSIFICATION REPORT :

```
In [97]: from sklearn.metrics import classification_report
classification_report(y_test,test_pred)
```

```
Out[97]: *           precision    recall  f1-score   support\n\n      0           0.99      0.97      0.98       70\n      1           0.94      0.97      0.95       30\n accuracy               0.97      0.97      0.97      100\nweighted avg           0.96      0.97
```

## HYPER PARAMETER TUNING

```
In [99]: parameters = {"criterion":["gini","entropy"],"max_depth":range(1,10),"random_state":[50,60]}
```

```
In [100]: from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(model1, parameters,cv = 10,verbose = 1)
grid.fit(X_train,y_train)
```

Fitting 10 folds for each of 36 candidates, totalling 360 fits

```
Out[100]: GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
                      param_grid={'criterion': ['gini', 'entropy'],
                                   'max_depth': range(1, 10), 'random_state': [50, 60]},
                      verbose=1)
```

```
In [101]: grid.best_params_
```

```
Out[101]: {'criterion': 'gini', 'max_depth': 6, 'random_state': 60}
```

```
In [102]: grid.best_estimator_
```

```
Out[102]: DecisionTreeClassifier(max_depth=6, random_state=60)
```

```
In [103]: grid.best_score_
```

```
Out[103]: 0.9833333333333334
```

```
In [104]: model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
```

## VALIDATION :

```
In [104]: model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
```

```
In [105]: model2.fit(X_train,y_train)
```

```
Out[105]: DecisionTreeClassifier(max_depth=6, random_state=60)
```

```
In [106]: train_pred2 = model1.predict(X_train)
test_pred2 = model1.predict(X_test)
```

```
In [107]: print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred2))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred2))
```

```
TRAINING ACCURACY : 1.0
TESTING ACCURACY : 0.97
```

```
In [108]: pd.crosstab(y_test,test_pred2)
```

```
Out[108]:
```

	col_0	0	1
classification			
0	68	2	
1	1	29	



## **10. ADVANTAGES AND DISADVANTAGES**

### **ADVANTAGES:**

1. The consultation fee is reduced.
2. It does not require a doctors appointment.
3. If they have access to the internet it is easy to take test.
4. Since the application predicts the chronic kidney disease earlier it will be easy to diagnose.
5. It is more efficient and it is not time consuming application.

### **DISADVANTAGES**

1. The user has to enter all the details required to predict in the form manually.

## **11. CONCLUSION**

This project has been designed to predict the chronic kidney disease in early stages so that it will be easy to diagnose, without any consultation fee and also without any doctors appointment. The result will be positive if the patient is affected by the chronic kidney disease, otherwise the result will be negative.

## **12. FUTURE SCOPE**

This application will be used by everyone in the future to know if they are affected by the chronic kidney disease in early stages. As it requires no consultation fee and doctors appointment, there is no need for going to the hospital to take test, so this is an easy way know to about their health condition.

# 13. APPENDIX

## SOURCE CODE:

Homepage.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="homepage.css">
  <title>Chronic Kidney Disease</title>
</head>
<body>
  <div class="front">
    
    <h1>CHRONIC KIDNEY DISEASE DETECTION</h1>
    <p>Kidney disease does not tend to cause symptoms when it's at an early stage.
      This is because the body is usually able to cope with a significant reduction in kidney
      function.
      Kidney disease is often only diagnosed at this stage if a routine test for another condition,
      such as a blood or urine test, detects a possible problem.

      If it's found at an early stage, medicine and regular tests to monitor it may help stop it
      becoming more advanced.</p>
    <p><b>SYMPTOMS</b>-Nausea and vomiting, muscle cramps, loss of appetite, swelling via
      feet and ankles, dry, itchy skin, shortness of breath, trouble sleeping, urinating either too much
      or too little</p>
    <a href="form.html">
      <button type="submit" >TAKE TEST</button>
    </a>
  </div>
</body>
</html>
```

Homepage.css

```
body{
  overflow: hidden;
}
.background{
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
  width: 100%;
  height: 100%;
  -webkit-filter: blur(5px);
  filter: blur(5px);
}
.front{
  height:100vh;
  width: 100%;
  text-align: center;
  font-size: 200%;
  color: rgb(11, 12, 12);
}
button{
  background-color: rgb(192, 171, 171);
  height: 50px;
  width: 200px;
  font-size: large;
  margin-block-end: auto;
  bottom: 50px;
}
```

FORM.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="form.css">
  <title>Document</title>
</head>
<body>
```

```

<div class="container">
  <h1>Enter the required details</h1>
  <div class="formbox">
    <video autoplay loop muted plays-inline id="myvedio">
      <source src="images/testing.mp4" type="video/mp4">
    </video>
    <form action="result.html" method="GET">
<!-- <section class="image-1">
</section>
<section class="image-2">
</section>
<section class="image-3">
</section>
<section class="image-4">
</section> -->
<!-- Name -->
<div>
  <h3>Name</h3>
  <input type="text" name="fullname" id="fullname">
</div>
<!-- Age -->
<div>
  <h3>Age</h3>
  <input type="number" name="age" id="age" >
</div>
<!-- Blood Pressure -->
<div>
  <h3>Blood Pressure</h3>
  <input type="number" name="bp" id="bp">
</div>
<!-- Urinary Specific gravity -->
<div>
  <h3>Urinary Specific gravity</h3>
  <input type="text" name="sg" id="sg">
</div>
<!-- Acute leukaemia -->
<div>
  <h3>Acute leukaemia</h3>
  <input type="number" name="al" id="al">
</div>
<!-- Secretory unit -->

```

```
<div>
  <h3>Secretary unit</h3>
  <input type="number" name="su" id="su">
</div>
<!-- Red Blood Cells -->
<div>
  <h3>Red Blood Cells</h3>
  <input type="radio" name="rbc" id="rbcn" value="normal">
  <label for="rbcn">Normal</label>
  <input type="radio" name="rbc" id="rbca" value="abnormal">
  <label for="rbca">Abnormal</label>
</div>
<!-- Platelet Count -->
<div>
  <h3>Platelet Count</h3>
  <input type="radio" name="pc" id="pcn" value="normal">
  <label for="pcn">Normal</label>
  <input type="radio" name="pc" id="pca" value="abnormal">
  <label for="pca">Abnormal</label>
</div>
<!-- prothrombin Complex concentrate -->
<div>
  <h3>Prothrombin Complex Concentrate</h3>
  <input type="radio" name="pcc" id="pccp" value="Present">
  <label for="pccp">Present</label>
  <input type="radio" name="pcc" id="pccn" value="Not Present">
  <label for="pccn">Not Present</label>
</div>
<!-- Bronchical Asthma -->
<div>
  <h3>Bronchical Asthma</h3>
  <input type="radio" name="ba" id="bap" value="Present">
  <label for="bap">Present</label>
  <input type="radio" name="ba" id="ban" value="Not Present">
  <label for="ban">Not Present</label>
</div>
<!-- Blood glucose regulator -->
<div>
  <h3>Blood glucose regulator</h3>
  <input type="number" name="bgr" id="bgr">
</div>
```

```
<!-- Bandelette terms -->
<div>
  <h3>Bandelette terms</h3>
  <input type="number" name="bt" id="bt">
</div>
<!-- subcutaneous -->
<div>
  <h3>Subcutaneous</h3>
  <input type="number" name="sub" id="sub">
</div>
<!-- Segregation of duties -->
<div>
  <h3>Segregation of duties</h3>
  <input type="number" name="seg" id="seg">
</div>
<!-- Pot -->
<div>
  <h3>POT</h3>
  <input type="number" name="pot" id="pot">
</div>
<!-- haemoglobin -->
<div>
  <h3>Haemoglobin</h3>
  <input type="number" name="hemo" id="hemo">
</div>
<!-- Pneumococcal conjugate vaccine -->
<div>
  <h3>Pneumococcal conjugate vaccine</h3>
  <input type="number" name="pcv" id="pcv">
</div>
<!-- white blood cell count -->
<div>
  <h3>White blood cell count </h3>
  <input type="number" name="wc" id="wc">
</div>
<!-- red blood cell count -->
<div>
  <h3>RC</h3>
  <input type="number" name="rc" id="rc">
</div>
<!-- hypertension -->
```



```
<div>
  <h3>Hypertension</h3>
  <input type="radio" name="htn" id="htny" value="yes">
  <label for="htny">Yes</label>
  <input type="radio" name="htn" id="htnn" value="no">
  <label for="htnn">No</label>
</div>
<!-- diabetes mellitus -->
<div>
  <h3>DM</h3>
  <input type="radio" name="dm" id="dmy" value="yes">
  <label for="dmy">Yes</label>
  <input type="radio" name="dm" id="dmn" value="no">
  <label for="dmn">No</label>
</div>
<!-- coronary artery disease -->
<div>
  <h3>Coronary artery disease</h3>
  <input type="radio" name="cad" id="cady" value="yes">
  <label for="cady">Yes</label>
  <input type="radio" name="cad" id="cadn" value="no">
  <label for="cadn">No</label>
</div>
<!-- appetite -->
<div>
  <h3>Appetite</h3>
  <input type="radio" name="appet" id="appetg" value="good">
  <label for="appetg">Good</label>
  <input type="radio" name="appet" id="appetp" value="poor">
  <label for="appetp">Poor</label>
</div>
<!-- pulmonary embolism -->
<div>
  <h3>Pulmonary embolism</h3>
  <input type="radio" name="pe" id="pey" value="yes">
  <label for="cady">Yes</label>
  <input type="radio" name="pe" id="pen" value="no">
  <label for="pen">No</label>
</div>
<!-- anemia -->
<div>
```

```

    <h3>Anemia</h3>
    <input type="radio" name="ane" id="aney" value="yes">
    <label for="aney">Yes</label>
    <input type="radio" name="ane" id="anen" value="no">
    <label for="anen">No</label>
  </div>
  <!-- Classification -->
  <div class="btn">
    <button type="submit">Submit</button>
  </div>
</form>
</div>
</div>
</body>
</html>

```

#### FORM .CSS

```

input,label {
  cursor: pointer;
  margin: .3rem;
  padding-right: 5rem;
}
input[type=text],[type=number],[type=submit] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border-radius: 4px;
  box-sizing: border-box;
  font-size: 20px;
}
input[type="radio"]:focus {
  height: 1rem;
  width: 1rem;
  margin-right: 0.2rem;
  margin-left: 10px;
  font-size: large;
}
button {
  background-color: #4CAF50;

```

```
border: none;
color: white;
padding: 10px 20px;
text-align: center;
text-decoration: none;
display: inline-block;
font-size: 16px;
}
.btn{
display: flex;
width: 300px;
text-align: center;
justify-content: center;
padding-top: 20px;
padding-left: 250px;
margin-right: 20px;
}
h1{
text-align: center;
}
form {
box-sizing: border-box;
padding: 2rem;
border-radius: 15rem;
}
.formbox{
max-width: 800px;
margin: auto;
border-radius: 10px;
display: flex;
align-items: center;
box-shadow: 0 0 20px rgba(0, 0, 0, 7);
color: rgb(14, 1, 1);
font-size: larger;
font-weight: 500;
background:rgba(0, 0, 0, 0.1)
}
#myvedio {
width: 100vw;
height: 100vh;
object-fit: cover;
```

```
position: fixed;
left: 0;
right: 0;
top: 0;
bottom: 0;
z-index: -1;
}
```

positive.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="page3.css">
  <title>Positive</title>
</head>
<body>
  <div class="gws"></div>
  <div class="positive">
    <h1>The result is</h1>
    
    <h2>"ONCE YOU CHOOSE HOPE, ANYTHING'S POSSIBLE."</h2>
    <h2>GET WELL SOON &#129310;</h2>
  </div>
</body>
</html>
```

positive.css

```
.gws{
  background-image: url("images/BG.jpeg");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  background-attachment: fixed;
  /* background-size: 100% 100%; */
  height: 680px;
  width: 100%;
```

```

    filter: blur(8px);
-webkit-filter: blur(8px);
    filter: blur(8px);
-webkit-filter: blur(8px);
}
.positive{
    /* display: flex;
    justify-content: center; */
    background-color: rgb(0,0,0); /* Fallback color */
    background-color: rgba(0,0,0, 0.4); /* Black w/opacity/see-through */
    color: rgb(255, 255, 255);
    font-weight: bold;
    border: 3px solid #f0d800;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    z-index: 2;
    width: 80%;
    padding: 20px;
    text-align: center;
}
.pos{
    height: 100%;
    width: 25%;
}

```

negative.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="page4.css">
    <title>Negative</title>
</head>
<body>
    <div class="gws"></div>
    <div class="positive">

```

```
<h1>The result is</h1>

<h2>"THE GREATEST HEALTH IS WEALTH"</h2>
<h2>STAY HEALTHY &#128522;</h2>
</div>
</body>
</html>
```

negative.css

```
.gws{
  background-image: url("images/emojis.gif");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  background-attachment: fixed;
  /* background-size: 100% 100%; */
  height: 680px;
  width: 100%;
  filter: blur(8px);
  -webkit-filter: blur(8px);
  filter: blur(8px);
  -webkit-filter: blur(8px);
}
.positive{
  /* display: flex;
  justify-content: center; */
  background-color: rgb(0,0,0); /* Fallback color */
  background-color: rgba(0,0,0, 0.4); /* Black w/opacity/see-through */
  color: rgb(255, 255, 255);
  font-weight: bold;
  border: 3px solid #f1f1f1;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 2;
  width: 80%;
  padding: 20px;
  text-align: center;
}
```

```
.neg{  
    height: 100%;  
    width: 25%;  
}
```

CKD PREDICTION.ipynb:

```
import pandas as pd  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
df = pd.read_csv('chronickidneydisease.csv')  
df  
df.describe()  
print("The size of the dataset is ",df.shape)  
print("Datatype of each column : ")  
print("-----")  
df.info()  
df.corr()  
sns.heatmap(df.corr())  
print("Drop unnecessary column : ")  
print("-----")  
df = df.drop(columns = ['id'],axis = 1)  
df  
df['age'].value_counts()  
df['bp'].value_counts()  
df['sg'].value_counts()  
df['al'].value_counts()  
df['su'].value_counts()  
df['rbc'].value_counts()  
df['pc'].value_counts()  
df['pcc'].value_counts()  
df['ba'].value_counts()  
df['bgr'].value_counts()  
df['bu'].value_counts()  
df['sc'].value_counts()  
df['sod'].value_counts()  
df['pot'].value_counts()  
df['hemo'].value_counts()  
df['pcv'].value_counts()
```

```

df['pcv'] = df['pcv'].replace('\t43','43')
df=df[df['pcv'] != '\t?']
df['wc'].value_counts()
df=df[df['wc'] != '\t?']
df['rc'].value_counts()
df=df[df['rc'] != '\t?']
df['htn'].value_counts()
df['dm'].value_counts()
df['dm'].replace(['\tyes','\tno'],['yes','no'],inplace = True)
df['cad'].value_counts()
df['cad'].replace('\tno','no',inplace = True)
df['appet'].value_counts()
df['pe'].value_counts()
df['ane'].value_counts()
df['classification'].value_counts()
df['classification'].replace('ckd\t','ckd',inplace = True)
numerical = ['age','bp','sg','al','sod','pot','hemo','pcv','wc','rc','su','bgr','bu','sc']
df['rc'] = df['rc'].astype(float)
df['wc'] = df['wc'].astype(float)
for x in numerical:
    sns.boxplot(df[x])
    plt.show()
for x in numerical:
    q1 = df[x].quantile(0.25)
    q3 = df[x].quantile(0.75)
    IQR = q3 - q1
    upper_limit = q3 + (1.5 * IQR)
    lower_limit = q1 - (1.5 * IQR)
    df[x] = np.where(df[x]>=upper_limit,df[x].median(),df[x])
    df[x] = np.where(df[x]<=lower_limit,df[x].median(),df[x])
for x in numerical:
    sns.boxplot(df[x])
    plt.show()
q1 = df['bgr'].quantile(0.25)
q3 = df['bgr'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['bgr'] = np.where(df['bgr']>=upper_limit,df['bgr'].median(),df['bgr'])
df['bgr'] = np.where(df['bgr']<=lower_limit,df['bgr'].median(),df['bgr'])
sns.boxplot(df['bgr'])

```



```

q1 = df['bu'].quantile(0.25)
q3 = df['bu'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['bu'] = np.where(df['bu']>=upper_limit,df['bu'].median(),df['bu'])
df['bu'] = np.where(df['bu']<=lower_limit,df['bu'].median(),df['bu'])
sns.boxplot(df['bu'])
q1 = df['sc'].quantile(0.25)
q3 = df['sc'].quantile(0.75)
IQR = q3 - q1
upper_limit = q3 + (1.5 * IQR)
lower_limit = q1 - (1.5 * IQR)
df['sc'] = np.where(df['sc']>=upper_limit,df['sc'].median(),df['sc'])
df['sc'] = np.where(df['sc']<=lower_limit,df['sc'].median(),df['sc'])
sns.boxplot(df['sc'])
print('CHECKING FOR NULL VALUES : ')
print("-----")
df.isnull().sum()
for x in df.columns:
    if(df[x].dtypes=='float64'):
        df[x].fillna(df[x].median(),inplace = True)
    else:
        df[x].fillna(df[x].mode()[0],inplace = True)
df.isnull().sum()
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['rbc'] = le.fit_transform(df['rbc'])
df['rbc'].value_counts()
df['pc'] = le.fit_transform(df['pc'])
df['pc'].value_counts()
df['pcc'] = le.fit_transform(df['pcc'])
df['pcc'].value_counts()
df['ba'] = le.fit_transform(df['ba'])
df['ba'].value_counts()
df['htn'] = le.fit_transform(df['htn'])
df['htn'].value_counts()
df['dm'] = le.fit_transform(df['dm'])
df['dm'].value_counts()
df['cad'] = le.fit_transform(df['cad'])
df['cad'].value_counts()

```

```

df['appet'] = le.fit_transform(df['appet'])
df['appet'].value_counts()
df['pe'] = le.fit_transform(df['pe'])
df['pe'].value_counts()
df['ane'] = le.fit_transform(df['ane'])
df['ane'].value_counts()
df['classification'] = le.fit_transform(df['classification'])
df['classification'].value_counts()
sns.distplot(df['bp'])
sns.distplot(df['sg'])
sns.distplot(df['al'])
sns.barplot(df['appet'].value_counts().index,df['appet'].value_counts())
sns.barplot(df['rbc'].value_counts().index,df['rbc'].value_counts())
sns.lineplot(df['age'],df['sod'])
def countplot_of_2(x,hue,title=None,figsize=(18,5)):
    plt.figure(figsize=figsize)
    sns.countplot(data=df[[x,hue]],x=x,hue=hue)
    plt.title(title)
    plt.show()
countplot_of_2('rc','htn',rc/htn)
sns.pairplot(df)
X = df.drop(columns = ['classification'],axis = 1)
X.head()
y = df['classification']
y.tail()
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
X_scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
X_scaled
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_scaled,y,test_size=0.25,random_state = 50)
print('X_train: ',X_train.shape)
print('X_test: ',X_test.shape)
print('y_train: ',y_train.shape)
print('y_test: ',y_test.shape)
from sklearn.tree import DecisionTreeClassifier
model1 = DecisionTreeClassifier()
model1.fit(X_train,y_train)
train_pred = model1.predict(X_train)
test_pred = model1.predict(X_test)
from sklearn.metrics import accuracy_score

```

```

print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred))
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,test_pred)
pd.crosstab(y_test,test_pred)
from sklearn.metrics import classification_report
classification_report(y_test,test_pred)
from sklearn.metrics import roc_curve,roc_auc_score
probability = model1.predict_proba(X_test)[:,-1]
fpr,tpr,thresholds = roc_curve(y_test,probability)
plt.plot(fpr,tpr)
plt.title('ROC CURVE')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
roc_auc_score(y_test,probability)
parameters = {"criterion":["gini','entropy'],'max_depth':range(1,10),'random_state':[50,60]}
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(model1, parameters,cv = 10,verbose = 1)
grid.fit(X_train,y_train)
grid.best_params_
grid.best_estimator_
grid.best_score_
model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
model2.fit(X_train,y_train)
train_pred2 = model1.predict(X_train)
test_pred2 = model1.predict(X_test)
print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred2))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred2))
pd.crosstab(y_test,test_pred2)
model1.predict([[80,70,1.02,0,0,1,1,0,0,104,28,1.1,135,4.1,15.3,48,6300,6.1,0,0,0,0,0]])
model2.predict([[80,70,1.02,0,0,1,1,0,0,104,28,1.1,135,4.1,15.3,48,6300,6.1,0,0,0,0,0]])
model1.predict([[48,70,1.005,4,0,1,0,1,0,117,56,3.8,111,2.5,11.2,32,6700,3.9,1,0,0,1,1]])
model2.predict([[48,70,1.005,4,0,1,0,1,0,117,56,3.8,111,2.5,11.2,32,6700,3.9,1,0,0,1,1]])
model1.predict([[47,60,1.02,0,0,1,1,0,0,137,17,0.5,150,3.5,13.6,44,7900,4.5,0,0,0,0,0]])
model2.predict([[47,60,1.02,0,0,1,1,0,0,137,17,0.5,150,3.5,13.6,44,7900,4.5,0,0,0,0,0]])
model1.predict([[73,100,1.01,3,2,0,0,1,0,295,90,5.6,140,2.9,9.2,30,7000,3.2,1,1,1,1,0]])
model2.predict([[73,100,1.01,3,2,0,0,1,0,295,90,5.6,140,2.9,9.2,30,7000,3.2,1,1,1,1,0]])
import joblib
joblib.dump(model1,'ckd_prediction.pkl')

```

**GITHUB LINK:**

<https://github.com/IBM-EPBL/IBM-Project-18395-1659684735>

**PROJECT DEMO LINK:**

<https://www.youtube.com/embed/xcgE5bKtqXc>