

EARLY DETECTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING

PROJECT REPORT

TEAM ID: PNT2022TMID01939

**BRUNDHA B , ABINAYA V , ABINAYA K ,
PRATHEEKSHA Y**

1. INTRODUCTION

1.1 PROJECT OVERVIEW :

Chronic kidney disease prediction is one of the most important issues in healthcare- analytics. The most interesting and challenging tasks in day-to-day lives as one third of the adult population is affected by chronic kidney disease (CKD), and millions die each year because they do not have access to affordable treatment. Chronic Kidney Disease can be cured, if treated in the early stages. The main aim of the project is to predict whether the patient have chronic kidney disease or not in a painless, accurate and faster way based on certain diagnostic measurement like Blood Pressure (BP), Albumin (Al) etc., and then appropriate treatment can be given based on the details provided by the model.

1.2 PURPOSE

The purpose of the project is to alert doctors for an early detection of kidney disease and hence ensure speedy recovery or prevention of kidney disease. This Project aims at creating a model for early detection of Chronic Kidney Disease using Machine Learning technology. The output is integrated with Flask. The front end developed in html is used to receive user input on various parameters needed to decide on the early detection of kidney disease. The same model is deployed into IBM cloud.

2. LITERATURE SURVEY:

2.1 EXISTING SYSTEM

Presently kidney disease is detected at late stages in many countries leading to loss of precious lives. There are very few means to identify them at an early stage. Most of the user details remain unverified and it's difficult to track the fake users. The user interface

of the application is not user friendly and the user must have a device with an android operating system with an active internet connection to interact with this application.

2.2 REFERENCES

1. "What Is Chronic Kidney Disease?". National Institute of Diabetes and Digestive and Kidney Diseases. June 2017. Retrieved 19 December 2017.
2. "Kidney Failure". MedlinePlus. Retrieved 11 November 2017.
3. "Chronic Kidney Disease Tests & Diagnosis". National Institute of Diabetes and Digestive and Kidney Diseases. October 2016. Retrieved 19 December 2017.
4. "Kidney Failure". National Institute of Diabetes and Digestive and Kidney Diseases. Retrieved 11 November 2017.
5. "Managing Chronic Kidney Disease". National Institute of Diabetes and Digestive and Kidney Diseases. October 2016.
6. "Eating Right for Chronic Kidney Disease | NIDDK". National Institute of Diabetes and Digestive and Kidney Diseases. Retrieved 5 September 2019.
7. "Mineral & Bone Disorder in Chronic Kidney Disease". National Institute of Diabetes and Digestive and Kidney Diseases. November 2015. Retrieved 19 December 2018

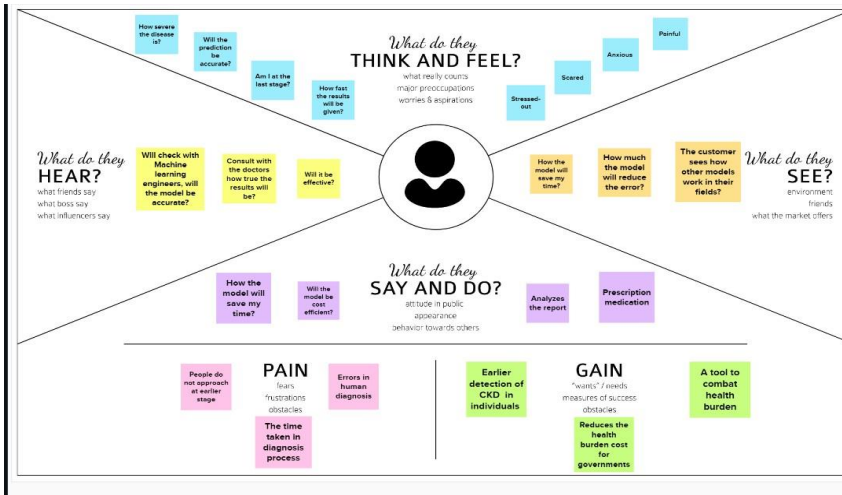
2.3 PROBLEM STATEMENT

Chronic Kidney Disease (CKD) is a serious medical condition that, if diagnosed early enough, is curable. Most individuals are unaware that the various medical tests we undergo for various reasons may provide important information about kidney disorders. As a result, characteristics of numerous medical tests are examined to see which characteristics might contain useful information about the disease. According to the information, doing so enables us to assess the problem's severity, and we utilize this data to create a machine learning model that forecasts chronic kidney disease.

If chronic kidney disease is addressed early on, it may be cured. This project's primary goal is to more accurately and quickly identify whether a patient has chronic kidney disease using diagnostic data including Blood Pressure (Bp), Albumin, and other parameters (Al).

3. IDEATION & PROPOSED SOLUTION:

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND BRAINSTORMING

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Akshaya B

What is CKD?

Machine should be user friendly

Developed in such a way that even the layman can identify. Reliability

Time used in analysis is reduced

Early and fast detection

Using available data in the report to predict accurately

Amrutha A

Easy method to diagnose and implement

precise at analyzing and detection

Healthy kidney to avoid future diseases

Better treatment

Healthy life

Highly reliable

Bhavedharani S

High accuracy

Prevent pain by early detection

No need for separate test reports

Human errors can be eliminated

Time efficient

Darshana C R

Non expensive

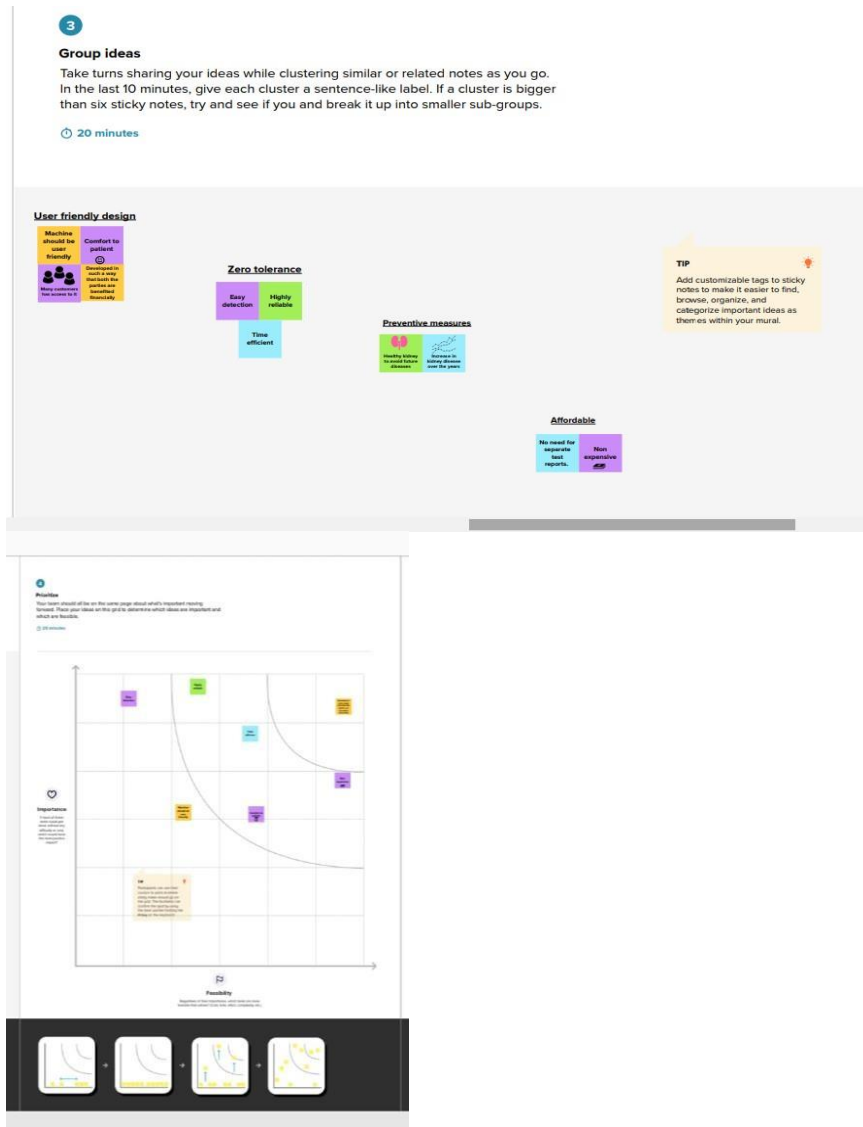
Easy detection

Comfort to patient

Many symptoms that require little

Easily accessible

Healthcare can be accessed easily by anyone at any time



3.3 PROPOSED SOLUTION

S.no	Parameter	Description
1.	Problem Statement(Problem to be solved)	<ul style="list-style-type: none"> Chronic Kidney Disease (CKD) is a major medical problem hence Chronic kidney disease prediction is one of the most important issues in healthcare analytics. 10% of the population worldwide is

		<p>affected by chronic kidney disease (CKD),and millions die each year because they do not have access to affordable treatment.</p> <ul style="list-style-type: none"> ● Chronic kidney Disease can be cured, if treated in the early stages.
2.	Solution Description	<ul style="list-style-type: none"> ● The idea is to detect the presence of kidney disease through machine learning based classification models. ● Early detection of chronic kidney disease is identified through various ML Algorithms such as Logistics Regression, Random Forest, Decision Tree, Support Vector Machines and KNN. ● Using these techniques, each algorithm's effectiveness is evaluated. ● A web app is developed that tasks basic user details about kidney details and result is produced
3.	Novelty	<ul style="list-style-type: none"> ● Aims to find the best machine learning model for the early prediction of chronic kidney disease by analyzing the essential parameters and comparing their predictive accuracies. ● Then collaborate the best machine learning model to an interactive user- interface which helps in the early detection of CKD and provide cure.
4.	Social Impact	<ul style="list-style-type: none"> ● The main aim of this application is early prediction of chronic kidney

		disease that can possibly stop or slow the progression of this disease to the end stage.
5.	Business Model	<ul style="list-style-type: none"> ● The widespread use of Machine Learning of predicting the CKD in the Medical Industries promotes medical innovation, lowers medical expenses, and improves medical quality. To cure the CKD patients at early stages. ● We can generate revenue through direct customers or can also collaborate with the health care sector and generate revenue from their customers.
6.	Scalability of solution	<ul style="list-style-type: none"> ● Early prediction of CKD using Machine Learning that is more efficient to analyze the disease so that it can be cured on time. ● We can also use image data and apply Deep Learning techniques such as Multilayer Perceptron (MLP) etc., which will provide an improved accuracy than machine learning.

3.4 Problem Solution Fit

Problem-Solution Fit			Project Title: Early Chronic Kidney Disease using Machine Learning Team ID: PNT2022TMID24563		
Define CS, fit into CL	1. CUSTOMER SEGMENT(S) CS All age people and Doctors.	6. CUSTOMER LIMITATIONS CL <small>EG. BUDGET, DEVICES</small> <ul style="list-style-type: none"> Costs and human errors associated with lab testing Separate expense for taking tests for CKD detection. 	5. AVAILABLE SOLUTIONS AS <small>PLUSES & MINUSES</small> <ul style="list-style-type: none"> Separate tests for CKD and analyzing the reports for CKD detection. Tests are taken only when the symptoms occur, but the symptoms show only when the patient is in critical stage. 		
	2. PROBLEMS / PAINS PR <small>+ ITS FREQUENCY</small> <ul style="list-style-type: none"> Early detection prevents severe damage of the kidney and lives of the patient. It also helps doctors to provide better treatment and to avoid critical conditions. 	9. PROBLEM ROOT / CAUSE RC <ul style="list-style-type: none"> Increase in kidney diseases among people due to modern culture. Early stages of kidney disease do not show symptoms. Time wasted in the detection can be used in the treatment. 	7. BEHAVIOR BE <small>+ ITS INTENSITY</small> Earlier detection and high accuracy directs patients to correct and proper medication which helps people to live long with healthy kidney.		
Focus on PR, tap into BE, understand RC	3. TRIGGERS TO ACT TR Knowledge about CKD and its seriousness, awareness among people about kidney diseases and early prevention through social media and news channels.	10. YOUR SOLUTION SL Our model makes use of available reports and says whether the patient's kidney is affected by any diseases or not, if yes, then says in which stage is the patient now and helps doctor to provide better treatment to cure the patient.	8. CHANNELS of BEHAVIOR CH ONLINE Deploying as a web app it can be used in online from anywhere. OFFLINE Make use of cost efficient and proper testing laboratories		
Identify strong TR & PR	4. EMOTIONS EM <small>BEFORE / AFTER</small> Before- Fear, Anxiety, confusion. After - relaxed and tension free.		Expect online & offline CH of BE		

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form
FR-2	User Confirmation	Confirmation via password.
FR-3	Obtain Information	The system should be able to get the information for predicting the disease.
FR-4	Displaying Result	The system must be able to display whether the user is affected or not

4.2 NON FUNCTIONAL REQUIREMENT

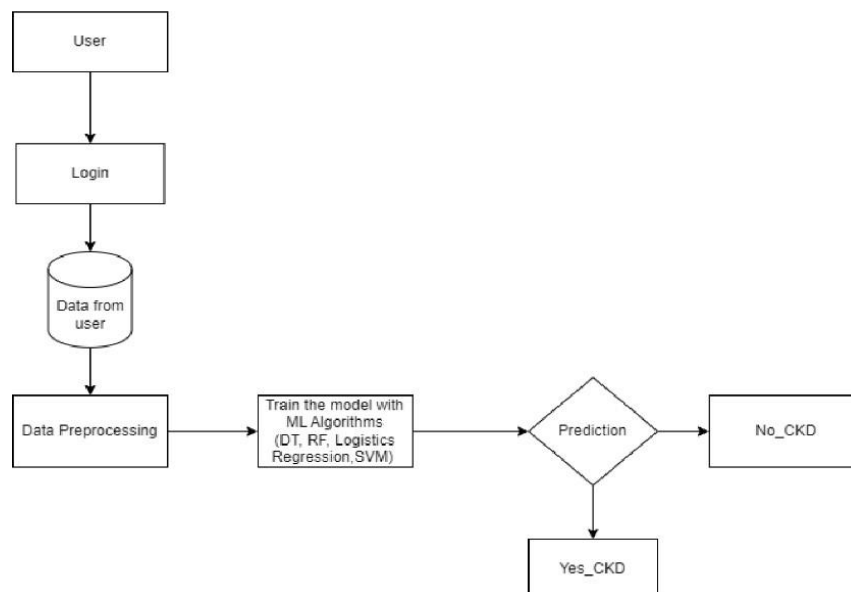
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Access to use the application is permitted only to the registered users

NFR-2	Security	Authentication is done for the security process.
NFR-3	Reliability	The user gets the correct and predicted value and standard results.
NFR-4	Performance	Lowering the total load time of prediction and user interaction
NFR-5	Availability	Easily available to everyone.
NFR-6	Scalability	To be able to change things.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored



5.2 SOLUTION AND TECHNICAL ARCHITECTURE

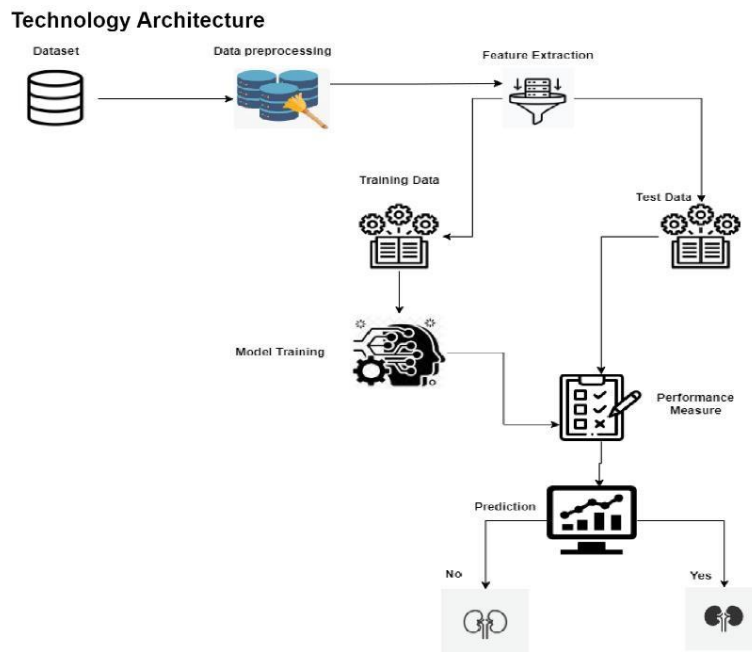


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1	User Interface	How user interacts with application	HTML, CSS,Python Flask
2	Application Logic-1	Get input from the user	HTML,CSS,Python Flask
3	Application Logic-2	Predicts based on the provided input	Python
4	Application Logic-3	Displays the predicted Result	Python,HTML,CSS,Flask
5.	Machine Learning Model	Random Forest,Regression techniques,Decision tree and SVM	Classification Algorithms

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
------	-----------------	-------------	------------

1.	Open-Source Frameworks	List the open-source frameworks used	Google colab,Jupyter notebook,IBM cloud and Flask.
2.	Scalable Architecture	Model can be scalable	Python
3.	Availability	It is used as a website(UI) or available in cloud	IBM cloud
4.	Performance	High accuracy	Machine Learning Classification techniques

5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	can access account	High	Sprint-1
	Login	USN-2	As a user, I can log into the application	Login into account	High	Sprint-1
Data Entry	Enter data	USN-3	Enter symptoms and clinical data	Enter ClinicalData	High	Sprint-2
Customer (View Result)	View Result	USN-4	Result can be viewed by the user	View Result	Medium	Sprint-3
Administrator	Predict Disease	USN-5	As the admin, I build ML models to predict CKD	I deploy models within websites	High	Sprint-1

6. PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND SCHEDULING

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

Use the below template to create product backlog and sprint schedule

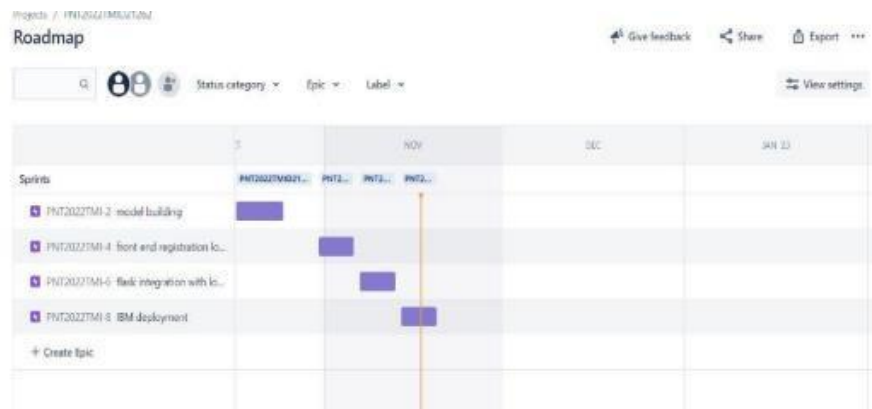
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collect the appropriate dataset for detection of Chronic Kidney Disease.	10	High	Amrutha A
Sprint-1	Data Pre-processing	USN-2	Transform the collected data into useful readable format through data pre-processing	10	High	Bhavadharani S
Sprint-2	Model Building	USN-3	Identifying the variations in the test reports of a diseased person and calculating the severeness of damage	10	Medium	Akshaya B
Sprint-2	Model Building	USN-4	Splitting the available dataset as training and testing datasets for the model	10	Medium	Darshana C R
Sprint-3	Training and Testing	USN-5	Training the model using classification algorithm and testing the performance of the model.	10	High	Akshaya B
Sprint-3	Application Building	USN-6	Develop HTML code for user interface and Python code for connection of model.	10	High	Darshana C R
Sprint-4	Application Building	USN-7	Running of Flask with connection of model to get the detection result from the model	10	Medium	Bhavadharani S
Sprint-4	Implementation of Application	USN-8	Completion of Application and Deployment in IBM Cloud	10	Medium	Amrutha A

6.2 SPRINT DELIVERY SCHEDULE

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



7. CODING AND SOLUTIONING

7.1 FLASK DEPLOYMENT

Using Flask we are locally deploying our machine Learning model. Flask acts as a web Framework. Additionally we have app.py file to locally deploy the model

Home.html

```

Flask > templates > home.html > html > body > div.form-wrapper.signup-form > form.form > div.inputContainer > input.input
1  <html>
2  <head>
3      <link rel="stylesheet" href="/static/style.css">
4      <title>KIDNEY DISEASE PREDICTION</title>
5  </head>
6  <body>
7      <div class="form-wrapper signup-form">
8          <form action="/login" method="POST" class="form">
9              <h1>KIDNEY DISEASE PREDICTOR</h1>
10             <h4>ENTER THE VALUES BELOW</h4>
11             <p>Prediction Result: <br>
12                 0: Chronic Kidney Disease has been found. <br>
13                 1: Chronic Kidney Disease has not been found. </p>
14             <div class="inputContainer">
15                 <input type="text" class="input" name="age" placeholder="Age" required>
16                 <label for="" class="label">Age</label>
17             </div>
18             <div class="inputContainer">
19                 <input type="text" class="input" name="bp" placeholder="Blood Pressure" required>
20                 <label for="" class="label">Blood Pressure</label>
21             </div>
22             <div class="inputContainer">
23                 <input type="text" class="input" name="sg" placeholder="Specific Gravity" required>
24                 <label for="" class="label">Specific Gravity</label>
25             </div>
26             <div class="inputContainer">
27                 <input type="text" class="input" name="alb" placeholder="Albumin" required>
28                 <label for="" class="label">Albumin</label>
29             </div>
30             <div class="inputContainer">
31                 <input type="text" class="input" name="sugar" placeholder="Sugar" required>
32                 <label for="" class="label">Sugar</label>
33             </div>
34             <div class="inputContainer">
35                 <input type="text" class="input" name="RBC" placeholder="RBC" required>

```

```

> templates > home.html > html > body > div.form-wrapper.signup-form > form.form > div.inputContainer > input.input
36                 <input type="text" class="input" name="Bacteria" placeholder="Bacteria" required>
37                 <label for="" class="label">Bacteria</label>
38             </div>
39             <div class="inputContainer">
40                 <input type="text" class="input" name="bgr" placeholder="BGR" required>
41                 <label for="" class="label">Blood Glucose Random</label>
42             </div>
43             <div class="inputContainer">
44                 <input type="text" class="input" name="bu" placeholder="BU" required>
45                 <label for="" class="label">Blood Urea</label>
46             </div>
47             <div class="inputContainer">
48                 <input type="text" class="input" name="sc" placeholder="SC" required>
49                 <label for="" class="label">Serum Creatinine</label>
50             </div>
51             <div class="inputContainer">
52                 <input type="text" class="input" name="sodium" placeholder="Sodium" required>
53                 <label for="" class="label">Sodium</label>
54             </div>
55             <div class="inputContainer">
56                 <input type="text" class="input" name="haemo" placeholder="Haemo" required>
57                 <label for="" class="label">Haemoglobin</label>
58             </div>
59             <div class="inputContainer">
60                 <input type="text" class="input" name="pcv" placeholder="PCV" required>
61                 <label for="" class="label">Packed Cell Volume</label>
62             </div>
63             <div class="inputContainer">

```

```
Go Run Terminal Help • home.html - ML - Visual Studio Code
main.html chronic_kidney_disease.csv app.py Model_one.pkl # style.css pic.jpeg home.html
Flask > templates > home.html > html > body > div.form-wrapper.signup-form > form.form > div.inputContainer > input.input
59 <input type="text" class="input" name="haemo" placeholder="Haemo" required>
60 <label for="" class="label">Haemoglobin</label>
61 </div>
62 <div class="inputContainer">
63 <input type="text" class="input" name="pcv" placeholder="PCV" required>
64 <label for="" class="label">Packed Cell Volume</label>
65 </div>
66 <div class="inputContainer">
67 <input type="text" class="input" name="rbc-count" placeholder="RBC Count" required>
68 <label for="" class="label">RBC Count</label>
69 </div>
70 <div class="inputContainer">
71 <input type="text" class="input" name="hypertension" placeholder="Hypertension" required>
72 <label for="" class="label">Hypertension</label>
73 </div>
74 <div class="inputContainer">
75 <input type="text" class="input" name="pe" placeholder="PE" required>
76 <label for="" class="label">Peda Edema</label>
77 </div>
78
79 <div class="result">
80 <b>{{y}}</b>
81 </div>
82 <input type="submit" class="submitBtn" value="Predict">
83 <a href="/loginForm" style="text-decoration: none;"></a>
84 </form>
85 </div>
86
87 </body>
88 </html>
```

Home.css

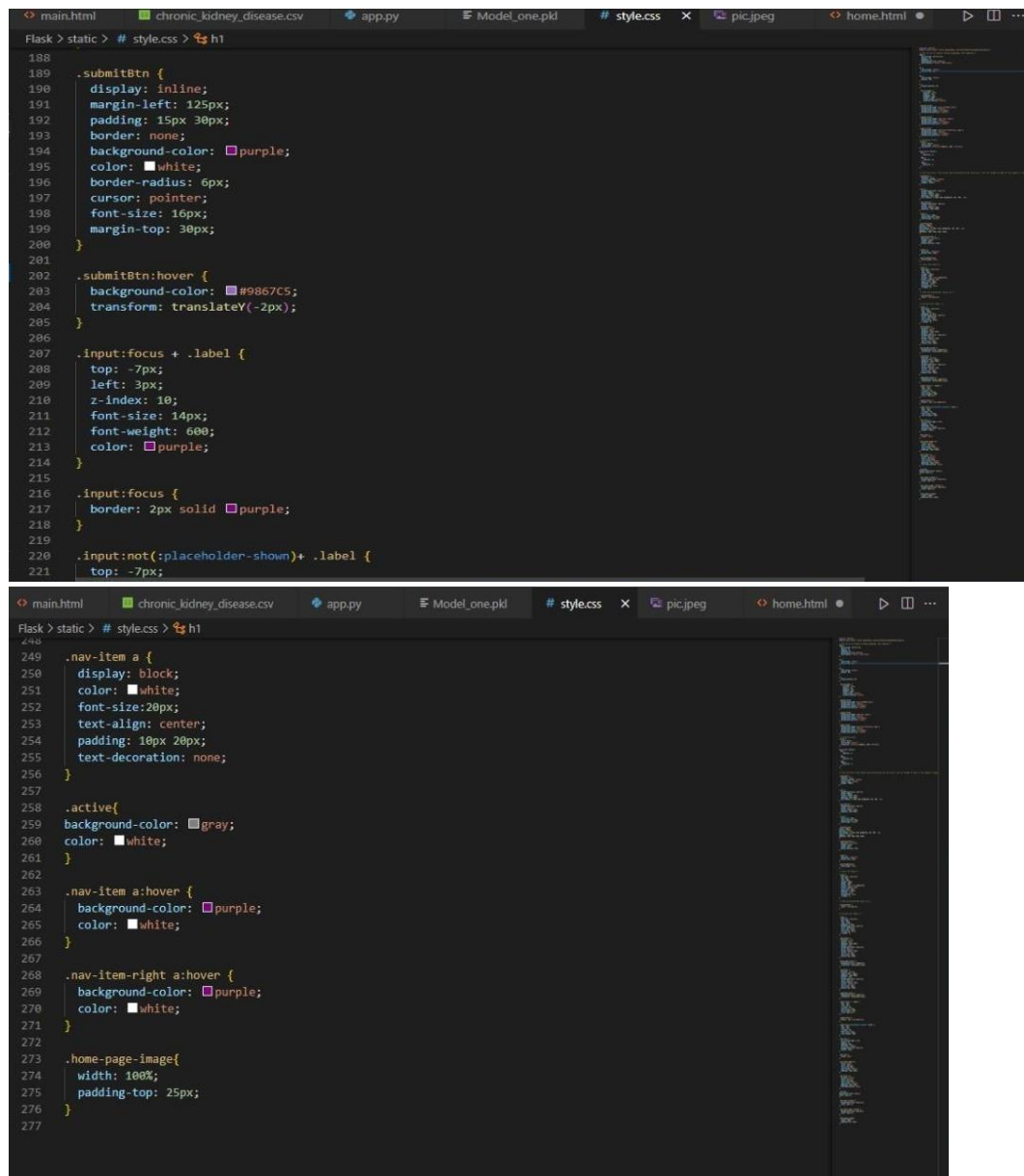
```
Go Run Terminal Help style.css - ML - Visual Studio Code
main.html chronic_kidney_disease.csv app.py Model_one.pkl # style.css pic.jpeg home.html
Flask > static > # style.css > h1
1 @charset "UTF-8";
2 @import url('https://fonts.googleapis.com/css2?family=Lato&display=swap');
3
4 /* Get rid of all default margins/paddings. Set typeface */
5 body {
6   box-sizing: border-box;
7   margin: 0;
8   padding: 0;
9   background-color: white;
10  font-family: "lato", sans-serif;
11 }
12
13 h1 {
14   text-align: center;
15   margin-top: 20;
16 }
17
18 h4 {
19   text-align: center;
20   margin: 40;
21 }
22
23 p {
24   margin-bottom: 35;
25 }
26
27 .form-wrapper {
28   display: flex;
29   margin: 175;
30   padding: 175;
31   height: 100%;
32   align-items: center;
33   justify-content: center;
34 }
```

```
Run Terminal Help style.css - ML - Visual Studio Code
main.html chronic_kidney_disease.csv app.py Model_one.pkl # style.css x pic.jpeg home.html
ask> static> # style.css > h1
62 }
63
64 @keyframes blink {
65   0% {
66     opacity: 1;
67   }
68   50% {
69     opacity: 0;
70   }
71   100% {
72     opacity: 1;
73   }
74 }
75
76
77 /* Puts the form in the center both horizontally and vertically. Sets its height to 100% of the viewport's height
78
79 .signupFrm {
80   display: flex;
81   justify-content: center;
82   align-items: center;
83   height: 100vh;
84 }
85
86
87 .form {
88   background-color: white;
89   width: 400px;
90   border-radius: 8px;
91   padding: 20px 90px;
92   box-shadow: 0 10px 25px rgba(92, 99, 105, .2);
93 }
94
95 .searchForm {
```

```
Flask> static> # style.css > h1
93 }
94
95 .searchForm {
96   background-color: white;
97   width: inherit;
98   border-radius: 8px;
99   padding: 20px 90px;
100 }
101
102 .title {
103   font-size: 50px;
104   margin-bottom: 50px;
105   text-align: center;
106 }
107
108 .userProfile{
109   width: 400px;
110   display: block;
111   box-shadow: 0 10px 25px rgba(92, 99, 105, .2);
112   margin: 0 auto;
113   padding: 10px 20px 10px 20px;
114 }
115
116 .inputContainer {
117   position: relative;
118   height: 45px;
119   width: 90%;
120   margin-bottom: 17px;
121 }
122
123
124 .result {
125   position: relative;
126   margin-top: 45px;
127 }
```

```
Go Run Terminal Help style.css - ML - Visual Studio Code
main.html chronic_kidney_disease.csv app.py Model_one.pkl style.css x pic.jpeg home.html
Flask > static > # style.css > h1
35
36 .upload-form{
37   background-image: url('3125018.jpg');
38   background-size: contain;
39   background-repeat: no-repeat;
40   background-position: center;
41 }
42
43 .signup-form{
44   background-image: url('pic.jpeg');
45   background-size: contain;
46   background-repeat: no-repeat;
47   background-position: center;
48 }
49
50 .login-form{
51   background-image: url('nutritionist_4.jpg');
52   background-size: contain;
53   background-repeat: no-repeat;
54   background-position: center;
55 }
56
57 /* Blinking style*/
58 .blink {
59   color: red;
60   text-align: center;
61   animation: blink 1s steps(1, end) infinite;
62 }
63
64 @keyframes blink {
65   0% {
66     opacity: 1;
67   }
68   50% {
69     opacity: 0;
70   }
71 }
```

```
Go Run Terminal Help style.css - ML - Visual Studio Code
main.html chronic_kidney_disease.csv app.py Model_one.pkl style.css x pic.jpeg home.html
Flask > static > # style.css > h1
122
123
124 .result {
125   position: relative;
126   margin-top: 45px;
127 }
128
129 .giftCardDetails{
130   text-align: left;
131 }
132
133 /* Style the inputs */
134
135 .input {
136   position: absolute;
137   top: 0px;
138   left: 0px;
139   height: 100%;
140   width: 100%;
141   border: 1px solid #DADCE0;
142   border-radius: 7px;
143   font-size: 16px;
144   padding: 0 20px;
145   outline: none;
146   background: none;
147   z-index: 1;
148 }
149
150 /* Hide the placeholder texts (a) */
151
152 ::placeholder {
153   color: transparent;
154 }
155
156
```

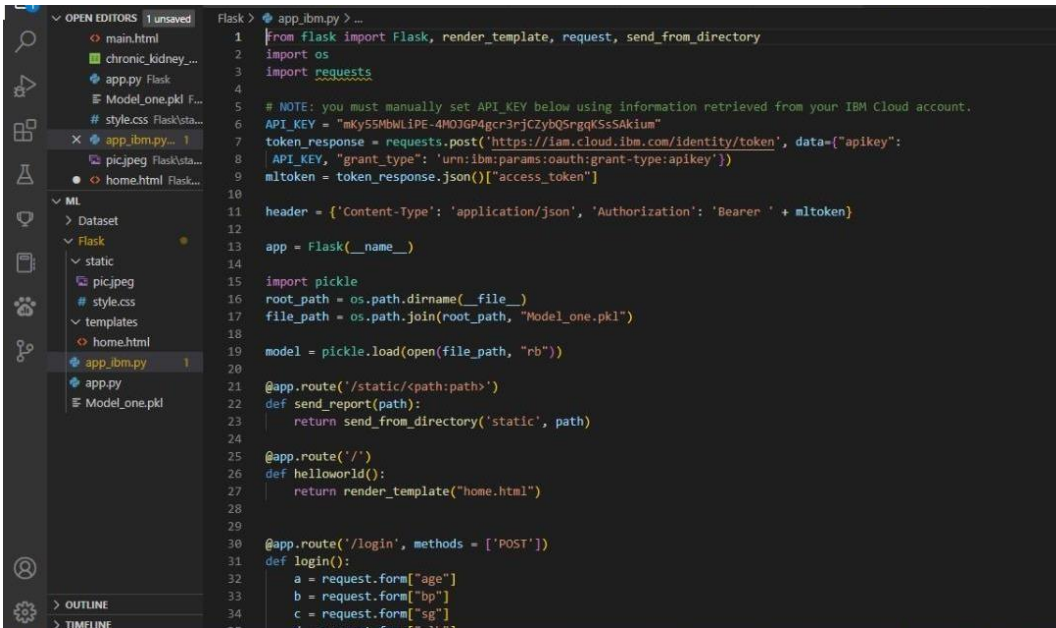


app.py

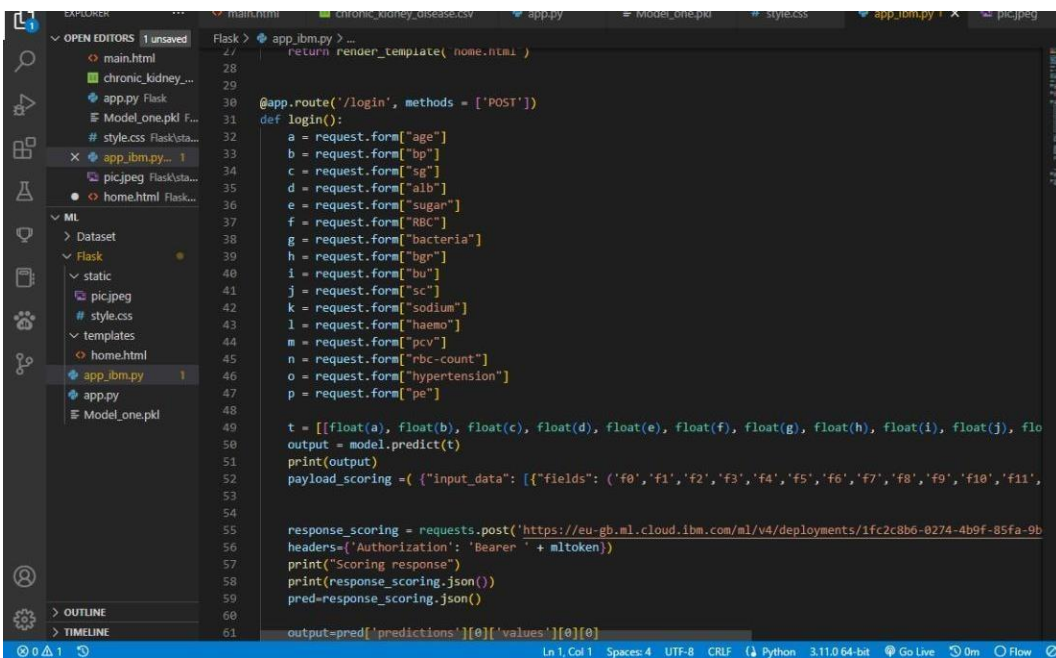

```
Flask > app.py
1  from flask import Flask, render_template, request, send_from_directory
2
3
4  app = Flask(__name__)
5
6  import pickle
7  model = pickle.load(open(r'Model_one.pkl', 'rb'))
8
9  @app.route('/static/<path:path>')
10 def send_report(path):
11     return send_from_directory('static', path)
12
13
14 @app.route('/')
15 def helloworld():
16     return render_template("home.html")
17
18
19 @app.route('/login', methods = ['POST'])
20 def login():
21     a = request.form["age"]
22     b = request.form["bp"]
23     c = request.form["sg"]
24     d = request.form["alb"]
25     e = request.form["sugar"]
26     f = request.form["RBC"]
27     g = request.form["bacteria"]
28     h = request.form["bgr"]
29     i = request.form["bu"]
30     j = request.form["sc"]
31     k = request.form["sodium"]
32     l = request.form["haemo"]
33     m = request.form["pcv"]
34     n = request.form["rbc-count"]
35     o = request.form["hypertension"]
36     p = request.form["pe"]
37
38     t = [[float(a), float(b), float(c), float(d), float(e), float(f), float(g), float(h), float(i), float(j), float(k), float(l), float(m), float(n), float(o), float(p)]]
39     output = model.predict(t)
40     print(output)
41
42     return render_template("home.html", y = "The predicted result is: " + str(output[0]))
43
44 @app.route('/admin')
45 def admin():
46     return "Hey Admin How are you?"
47
48 app.run(host='localhost', port=5000)
49 if __name__ == '__main__':
50     app.run(debug = True)
```

7.2 IBM DEPLOYMENT

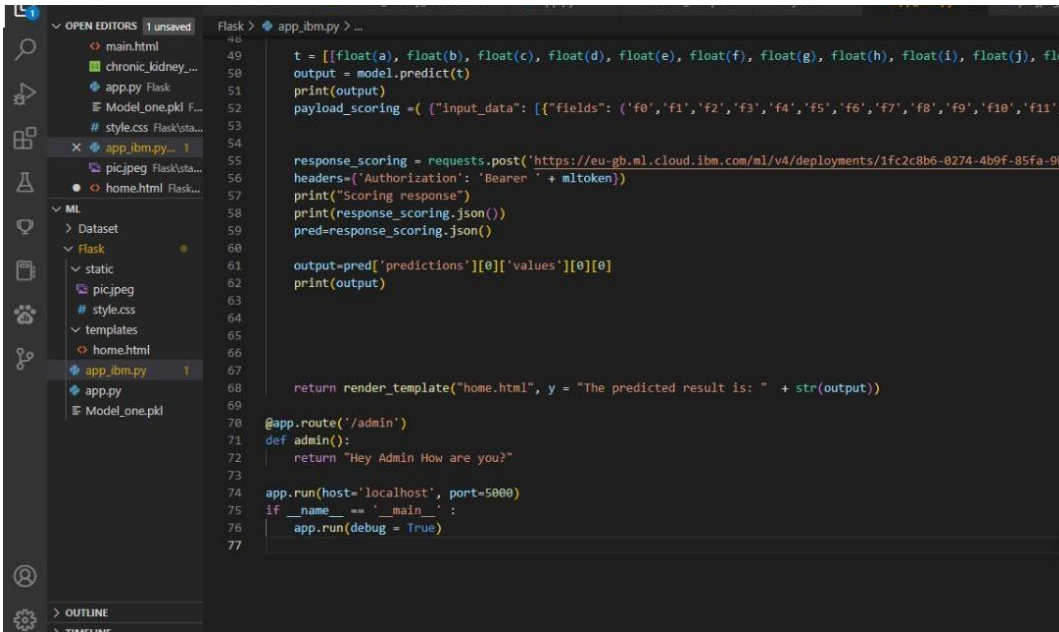
Now after Locally deploying our machine Learning model we deployed our model into IBM deployment



```
1 from flask import Flask, render_template, request, send_from_directory
2 import os
3 import requests
4
5 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
6 API_KEY = "mky55MBwLiPE-4MOJGP4gcr3rjCZybQ5rgqK5sSakium"
7 token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
8 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
9 mltoken = token_response.json()["access_token"]
10
11 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
12
13 app = Flask(__name__)
14
15 import pickle
16 root_path = os.path.dirname(__file__)
17 file_path = os.path.join(root_path, "Model_one.pkl")
18
19 model = pickle.load(open(file_path, "rb"))
20
21 @app.route('/static/<path:path>')
22 def send_report(path):
23     return send_from_directory('static', path)
24
25 @app.route('/')
26 def helloworld():
27     return render_template("home.html")
28
29
30 @app.route('/login', methods = ['POST'])
31 def login():
32     a = request.form["age"]
33     b = request.form["bp"]
34     c = request.form["sg"]
```



```
47     return render_template("home.html")
48
49
50 @app.route('/login', methods = ['POST'])
51 def login():
52     a = request.form["age"]
53     b = request.form["bp"]
54     c = request.form["sg"]
55     d = request.form["alb"]
56     e = request.form["sugar"]
57     f = request.form["RBC"]
58     g = request.form["bacteria"]
59     h = request.form["bgr"]
60     i = request.form["bu"]
61     j = request.form["sc"]
62     k = request.form["sodium"]
63     l = request.form["haemo"]
64     m = request.form["pcv"]
65     n = request.form["rbc-count"]
66     o = request.form["hypertension"]
67     p = request.form["pe"]
68
69     t = [[float(a), float(b), float(c), float(d), float(e), float(f), float(g), float(h), float(i), float(j), flo
70 output = model.predict(t)
71 print(output)
72 payload_scoring = ({ "input_data": [{"fields": ('f0','f1','f2','f3','f4','f5','f6','f7','f8','f9','f10','f11',
73
74
75 response_scoring = requests.post('https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/1fc2c8b6-0274-4b9f-85fa-9b
76 headers={'Authorization': 'Bearer ' + mltoken})
77 print("Scoring response")
78 print(response_scoring.json())
79 pred=response_scoring.json()
80
81 output=pred['predictions'][0]['values'][0][0]
```

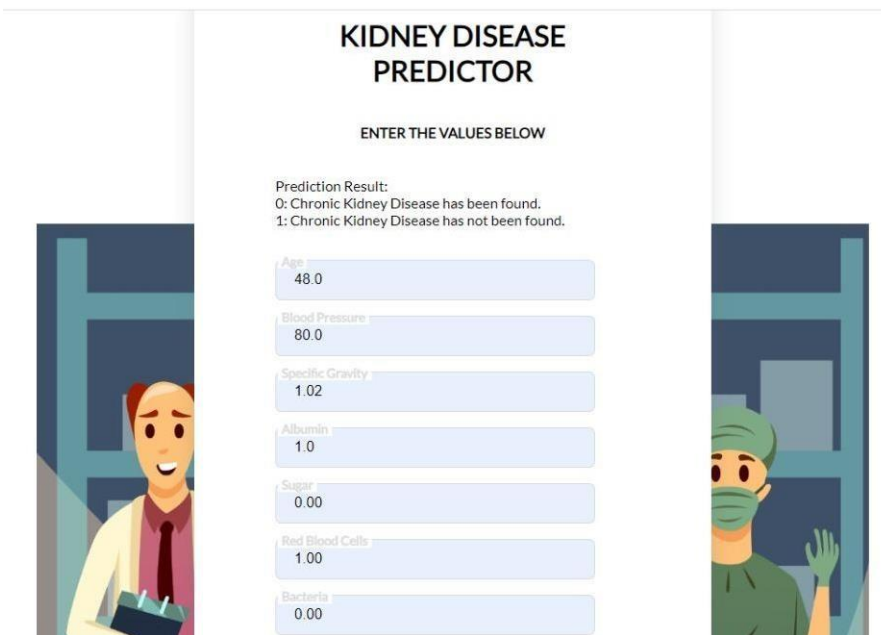


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with files like main.html, chronic_kidney..., app.py, Model_one.pkl, style.css, picjpeg, templates, home.html, app_ibm.py, app.py, and Model_one.pkl. The code editor shows the following Python code:

```
49 t = [[float(a), float(b), float(c), float(d), float(e), float(f), float(g), float(h), float(i), float(j), flo
50 output = model.predict(t)
51 print(output)
52 payload_scoring = { "input_data": [{"fields": ('f0','f1','f2','f3','f4','f5','f6','f7','f8','f9','f10','f11',
53
54
55 response_scoring = requests.post('https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/1fc2c8b6-0274-4b9f-85fa-9b
56 headers={'Authorization': 'Bearer ' + mltoken})
57 print("Scoring response")
58 print(response_scoring.json())
59 pred=response_scoring.json()
60
61 output=pred['predictions'][0]['values'][0][0]
62 print(output)
63
64
65
66
67
68 return render_template("home.html", y = "The predicted result is: " + str(output))
69
70 @app.route('/admin')
71 def admin():
72     return "Hey Admin How are you?"
73
74 app.run(host='localhost', port=5000)
75 if __name__ == '__main__':
76     app.run(debug = True)
77
```

8. TESTING

8.1 TESTCASES



The screenshot shows a web application titled "KIDNEY DISEASE PREDICTOR". Below the title, it says "ENTER THE VALUES BELOW". There are seven input fields with the following labels and values:



- Age: 48.0
- Blood Pressure: 80.0
- Specific Gravity: 1.02
- Albumin: 1.0
- Sugar: 0.00
- Red Blood Cells: 1.00
- Bacteria: 0.00

Below the input fields, it says "Prediction Result:" followed by two lines of text:

- 0: Chronic Kidney Disease has been found.
- 1: Chronic Kidney Disease has not been found.

The application is flanked by two illustrations: a doctor on the left and a nurse on the right.

http://localhost:5000/login



Age	121.0
Blood Urea	36
Serum Creatinine	1.20
Sodium	138.00
Haemoglobin	15.40
Packed Cell Volume	32
RBC Count	34.0
Hypertension	1.00
Peds Edema	0.00



The predicted result is: 1

Predict

KIDNEY DISEASE PREDICTOR

ENTER THE VALUES BELOW

Prediction Result:
0: Chronic Kidney Disease has been found.
1: Chronic Kidney Disease has not been found.



Age	51
Blood Pressure	60
Specific Gravity	1.00
Albumin	3.0
Sugar	4.0
Red Blood Cells	1.00
Bacteria	0.00

http://localhost:5000/login

106.00

Blood Urea 36

Serum Creatinine 23

Sodium 32

Haemoglobin 11.60

Packed Cell Volume 32.0

RBC Count 23

Hypertension 1

Peda Edema 0

The predicted result is: 0

Predict

8.2 USER ACCEPCANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Early Detection of Chronic Kidney Disease] 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	3	1	1	1	6
Duplicate	4	0	2	0	6
External	2	2	0	1	5
Fixed	1	1	1	1	4
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	10	4	4	3	21

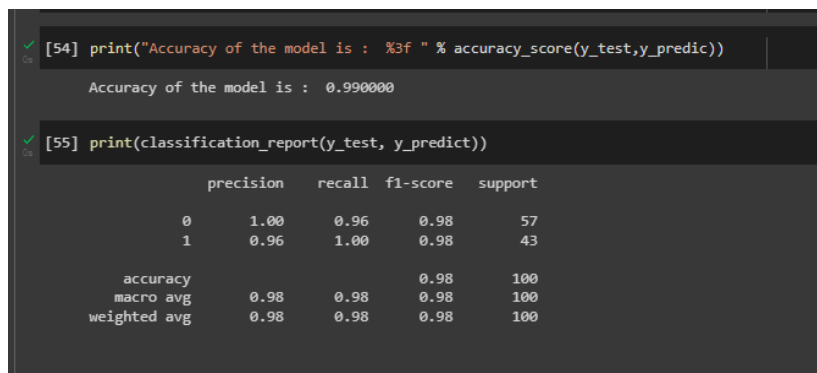
3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Home Screen	1	0	0	1
User Input	3	0	0	3
Chronic Kidney Disease testing	2	0	0	2
No Chronic Kidney Disease testing	2	0	0	2
Version Control	2	0	0	2

9. RESULTS

9.1 PERFORMANCE METRICS



The screenshot shows a Jupyter Notebook interface with two code cells. The first cell, labeled [54], contains the code `print("Accuracy of the model is : %3f " % accuracy_score(y_test,y_predic))` and its output is "Accuracy of the model is : 0.990000". The second cell, labeled [55], contains the code `print(classification_report(y_test, y_predict))` and its output is a classification report table.

	precision	recall	f1-score	support
0	1.00	0.96	0.98	57
1	0.96	1.00	0.98	43
accuracy			0.98	100
macro avg	0.98	0.98	0.98	100
weighted avg	0.98	0.98	0.98	100

10. ADVANTAGES & DISADVANTAGES

Advantages:

Chronic kidney disease (CKD) is one of the most critical health problems due to its increasing prevalence. It is also known as chronic renal disease which is a condition characterized by a gradual loss of kidney function over time. A better testing method which could possibly detect CKD in the early stages would be much more useful using machine learning algorithm

- Greater cost reduction in hospitals for testing
- Helps in early diagnosis of the disease
- Chances of recovery is higher

Disadvantages:

Even Though the CKD prediction model web application consists of a lot of advantages but it comes with certain disadvantages here are some of them .

- Chances of prediction to be wrong for least number of time which can cause problems
- Vast feature in dataset on discovery of time for the disease making the model inefficient to keep up the metrics
- Since it's a web application it requires scaling of web applications to handle concurrent requests after a certain threshold.

11. CONCLUSION

The benefit of this approach is that the prediction process takes far less time doctors to initiate treatment at the earliest for patients with CKD and further to classify larger populations of patients within a shorter span. Because the dataset used in this paper is tiny with 400 examples, we prefer to work with larger datasets in the future or compare the results of this dataset with a different dataset with the same. In addition, to help minimize the incidence of CKD, we try to predict if a person with this syndrome chances chronic risk factors such as hypertension, family history of kidney failure and diabetes using the appropriate dataset. Early prediction is very crucial for both the experts and the patients to prevent and slow down the progress of chronic kidney disease to kidney failure.

12. FUTURE SCOPE

This work will be considered as the basement for the healthcare system for CKD patients. Also, extension to this work is that implementation of Machine learning provides high- quality performance. The hope is that it would encourage people to seek early treatment for chronic renal disease and to make improvements in their lives.

13. APPENDIX

Chronic Kidney Disease (CKD) or chronic renal disease has become a major issue with a steady growth rate. A person can only survive without kidneys for an average time of 18 days, which makes a huge demand for a kidney transplant and Dialysis. It is important to have effective methods for early prediction of CKD. Machine learning methods are effective in CKD prediction. This work proposes a workflow to predict CKD status based on clinical data, incorporating data prepossessing, a missing value handling method with collaborative filtering and attributes selection. Out of the 11 machine learning methods considered, the extra tree classifier and random forest classifier are shown to result in the highest accuracy and minimal bias to the attributes. The research also considers the practical aspects of data collection and highlights the importance of incorporating CKD status prediction.

GITHUB LINK :

<https://github.com/IBM-EPBL/IBM-Project-7593-1658893656>