



**NAALAIYA THIRAN PROJECT - 2022  
19ECI01-PROFESSIONAL READINESS FOR  
INNOVATION, EMPLOYABILITY AND  
ENTREPRENEURSHIP**



**EARLY DETECTION OF CHRONIC KIDNEY DISEASE  
USING MACHINE LEARNING**

**A PROJECT REPORT**

*Submitted by*

***TEAM ID-PNT2022TMID52837***

**KAVEYA S**

**CITC1904084**

**MAHITHA S**

**CITC1904087**

**NAVEENA R**

**CITC1904096**

**SWATHI D**

**CITC1904117**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION  
ENGINEERING**

**COIMBATORE INSTITUTE OF TECHNOLOGY, COIMBATORE – 641 014**

**(Government Aided Autonomous Institution affiliated to Anna University)**

**ANNA UNIVERSITY: CHENNAI 600025**

**NOVEMBER 2022**

## PROJECT CALENDAR

Phase	Phase Description	Week	Dates	Activity Details
1	Preparation Phase (Pre- requisites, Registrations, Environment Set-up, etc.)	2	22 - 27 Aug 2022	Creation GitHub account & collaborate with Project repository in project workspace
2	Ideation Phase (Literature Survey, Empathize, Defining Problem Statement, Ideation)	2	29 Aug – 3rd Sept 2022	Literature survey (Aim, objective, problem statement and need for the project)
		3	5 - 10th Sept 2022	Preparing Empathy Map Canvas to capture the user Pains & Gains
		4	12 - 17 Sept 2022	Listing of the ideas using brainstorming session
3	Project Design Phase -I (Proposed Solution, Problem- Solution Fit, Solution Architecture)	5	19 - 24 Sept 2022	Preparing the proposed solution document
		6	26 Sept - 01 Oct 2022	Preparing problem - solution fit document & Solution Architecture
4	Project Design Phase -II (Requirement Analysis, Customer Journey, Data Flow Diagrams, Technology Architecture)	7	3 - 8 Oct 2022	Preparing the customer journey maps
		8	10 - 15 Oct 2022	Preparing the Functional Requirement Document & Data- Flow Diagrams and Technology Architecture
5	Project Planning Phase (Milestones & Tasks, Sprint Schedules )	9	17 - 22 Oct 2022	Preparing Milestone & Activity List, Sprint Delivery Plan
6	Project Development Phase (Coding & Solutioning, acceptance Testing, Performance Testing)	10	24 - 29 Oct 2022	Preparing Project Development - Delivery of Sprint-1
		11	31 Oct - 5 Nov 2022	Preparing Project Development - Delivery of Sprint-2
		12	7 - 12 Nov 2022	Preparing Project Development - Delivery of Sprint-3
		13	14 - 19 Nov 2022	Preparing Project Development - Delivery of Sprint-4

## **TABLE OF CONTENTS**

<b>CHAPTER NO</b>	<b>CONTENTS</b>	<b>PAGE NO</b>
	<b>ABSTRACT</b>	
	<b>LIST OF FIGURES</b>	
	<b>LIST OF TABLES</b>	
<b>1</b>	<b>INTRODUCTION</b>  <b>1.1 PROJECT OVERVIEW</b>  <b>1.2 PURPOSE</b>	<b>1</b>
<b>2</b>	<b>LITERATURE SURVEY</b>  <b>2.1 EXISTING SOLUTION</b>  <b>2.2 PROBLEM STATEMENT DEFINITION</b>	<b>3</b>
<b>3</b>	<b>IDEATION &amp; PROPOSED SOLUTION</b>  <b>3.1 EMPATHY MAPCANVAS</b>  <b>3.2 IDEATION &amp; BRAINSTORMING</b>  <b>3.3 PROPOSED SOLUTION</b>  <b>3.4 PROBLEM SOLUTION FIT</b>	<b>4</b>
<b>4</b>	<b>REQUIREMENT ANALYSIS</b>  <b>4.1 FUNCTIONAL REQUIREMENT</b>  <b>4.2 NON-FUNCTIONAL REQUIREMENT</b>	<b>11</b>
<b>5</b>	<b>PROJECT DESIGN</b>  <b>5.1 DATA FLOW DIAGRAMS</b>  <b>5.2 SOLUTION &amp; TECHNICAL ARCHITECTURE</b>	<b>14</b>
<b>6</b>	<b>PROJECT PLANNING &amp; SCHEDULING</b>  <b>6.1 SPRINT PLANNING &amp; ESTIMATION</b>	<b>17</b>

	<b>6.2 SPRINT DELIVERY SCHEDULE</b>	
<b>7</b>	<b>CODING &amp; SOLUTIONING</b> <b>7.1 FEATURE</b>	<b>20</b>
<b>8</b>	<b>TESTING AND RESULTS</b> <b>8.1 TEST CASES AND RESULTS</b>	<b>37</b>
<b>9</b>	<b>PERFORMANCE RESULTS</b> <b>9.1 PERFORMANCE METRICES</b>	<b>39</b>
<b>10</b>	<b>ADVANTAGES &amp; DISADVANTAGES</b>	<b>40</b>
<b>11</b>	<b>CONCLUSION</b>	<b>44</b>
<b>12</b>	<b>FUTURE SCOPE</b>	<b>45</b>

**SOURCE CODE**

**GITHUB & PROJECT DEMO LINK**

## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
<b>3.1</b>	<b>EMPATHY MAP</b>	<b>4</b>
<b>3.2</b>	<b>BRAINSTORMING</b>	<b>5</b>
<b>3.3</b>	<b>CLUSTERING OF IDEAS</b>	<b>6</b>
<b>3.4</b>	<b>PRIORITIZATION OF IDEAS</b>	<b>6</b>
<b>3.5</b>	<b>PROBLEM SOLUTION FIT</b>	<b>9</b>
<b>5.1</b>	<b>DATA FLOW DIAGRAM</b>	<b>13</b>
<b>5.2</b>	<b>SOLUTION ARCHITECTURE</b>	<b>14</b>
<b>7.1</b>	<b>LIBRARIES REQUIRED</b>	<b>19</b>
<b>7.2</b>	<b>LOADING THE DATASET</b>	<b>20</b>
<b>7.3</b>	<b>FEATURES OF DATASET</b>	<b>21</b>
<b>7.4</b>	<b>DATA PREPROCESSING</b>	<b>22</b>
<b>7.5</b>	<b>HANDLING MISSING VALUES</b>	<b>23</b>
<b>7.6</b>	<b>LABEL ENCODING</b>	<b>24</b>
<b>7.7</b>	<b>TRAIN-TEST SPLIT</b>	<b>24</b>
<b>7.8</b>	<b>PERFORMANCE ANALYSIS ON TRAINED DATA</b>	<b>25</b>
<b>7.9</b>	<b>FEATURE IMPORTANCE</b>	<b>26</b>
<b>7.10</b>	<b>SAVE THE MODEL</b>	<b>26</b>
<b>7.11</b>	<b>FLASK IMPLEMENTATION</b>	<b>27</b>
<b>8.1</b>	<b>LOGIN PAGE</b>	<b>29</b>
<b>8.2</b>	<b>DETAILS OF THE PATIENT</b>	<b>29</b>
<b>8.3</b>	<b>OUTPUT OF CKD AFFECTED PATIENT</b>	<b>30</b>
<b>8.4</b>	<b>OUTPUT OF NON-CKD PATIENT</b>	<b>30</b>
<b>9.1</b>	<b>ROC CURVE</b>	<b>31</b>
<b>9.2</b>	<b>ROC CURVE AFTER MODEL TRAINING</b>	<b>32</b>

### **LIST OF TABLES**

<b>TABLE NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
<b>3.1</b>	<b>PROPOSED SOLUTION</b>	<b>7</b>
<b>4.1</b>	<b>FUNCTIONAL REQUIREMENTS</b>	<b>10</b>
<b>4.2</b>	<b>NON-FUNCTIONAL REQUIREMENT</b>	<b>11</b>
<b>6.1</b>	<b>SPRINT PLANNING AND ESTIMATION</b>	<b>15</b>
<b>6.2</b>	<b>SPRINT DELIVERY PLAN</b>	<b>17</b>

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 PROJECT OVERVIEW**

Chronic kidney disease (CKD) is a global health problem with high morbidity and mortality rate as it induces other diseases. It is a severe lifelong condition caused either by renal disease or by impaired functions of the kidneys. CKD is classified as a condition that results in decreased kidney function over time, as indicated by Glomerular filtration rate(GFR). Since there are no obvious symptoms during the early stages of chronic kidney disease, patients often fail to notice the disease. As people with CKDs are likely to die at early age, early detection of this disease enables patients to receive timely treatment to ameliorate the progression of this disease.

In many hospitals and clinics, there is a shortage of nephrologists or general physicians who diagnose the disease. This has resulted in patients waiting longer to get a diagnosis. Therefore, developing a system to classify a patient into classes of ‘CKD’ or ‘Non-CKD’ can help the doctors to deal with multiple patients and provide diagnosis faster. In time, hospitals and clinics can implement the proposed machine learning framework that have lower medical expert retention which can provide early diagnosis to patients in regional areas. Machine learning models can effectively aid clinicians achieve this goal due to their fast and accurate recognition performance. Those who are with Chronic Kidney Disease (CKD) are not aware that the medical tests they take for other purposes sometimes contain useful information about CKD disease. This information is sometimes not used effectively to tackle the identification of the disease. Therefore, attributes of different medical tests are investigated to identify what attributes contain useful information about CKD.

A database with several attributes of healthy subjects and subjects with CKD are analyzed with different techniques. Common Spatial Pattern

(CSP) filter and Linear Discriminant Analysis (LDA) are first used to identify the dominant attributes that could contribute in detecting CKD. Here CSP filter is applied to optimize separation between CKD and non-CKD. Then, classification methods are also used to identify the dominant attributes. The primary aim of the proposed system is to implement and compare the performance of various supervised and unsupervised algorithms and identify the best possible combinations that can provide better accuracy and detection rate.

## **1.2 PURPOSE**

There is a shortage of nephrologists or general physicians who diagnose this disease in many hospitals. This has resulted in patients waiting longer to get a diagnosis. Therefore, developing a system to classify a patient into classes of 'CKD' or 'Non-CKD' can help the doctors to deal with multiple patients and provide diagnosis faster. In time, hospitals and clinics can implement the proposed machine learning framework that have lower medical expert retention which can provide early diagnosis to patients in regional areas.



## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 EXISTING SOLUTION**

In attention and unawareness of the symptoms of Chronic Kidney Disease has resulted in many patients being affected adversely by CKD. There is no early and efficient detection of Chronic Kidney Disease currently. This existing problem of shortage of nephrologists in regional areas is overcome with the use of machine learning for early detection of chronic kidney disease.

#### **2.3 PROBLEM STATEMENT DEFINITION**

Early detection of Chronic kidney disease using machine learning aims to prevent the high morbidity rate and high mortality rate.

#### **SUMMARY**

This is a binary classification as there are only two classes i.e CKD class or healthy class. The classifier needs the training to identify how the given samples are associated with the particular class. After training the model, the performance of the designed network is examined by evaluating the performance parameters. For comparing the performance of the proposed model, performance evaluation can be done using parameters precision, sensitivity, specificity, False Negative Rate (FNR), False Discovery Rate (FDR), Matthews Correlation Coefficient (MCC), Negative Predictive Value (NPV), F1 score, False Positive Rate (FPR) and Misclassification Rate (MR). Ten-fold cross-validation can be done further to estimate the model performance.

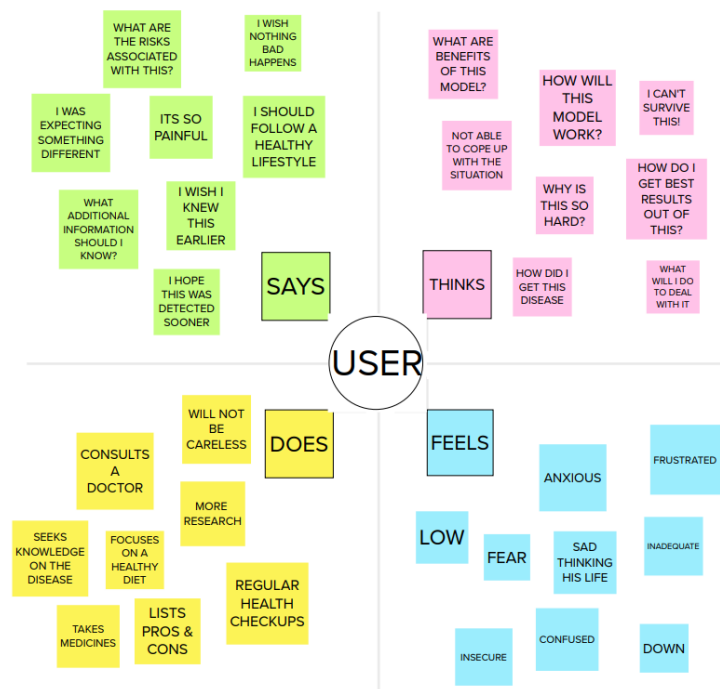
## CHAPTER 3

### IDEATION AND PROPOSED SOLUTION

#### 3.1 EMPATHY MAP CANVAS

An empathy map is a collaborative visualization used to express clearly what one knows about a particular type of user. It externalizes knowledge about users in order to create a shared understanding of user needs, and aid in decision making.

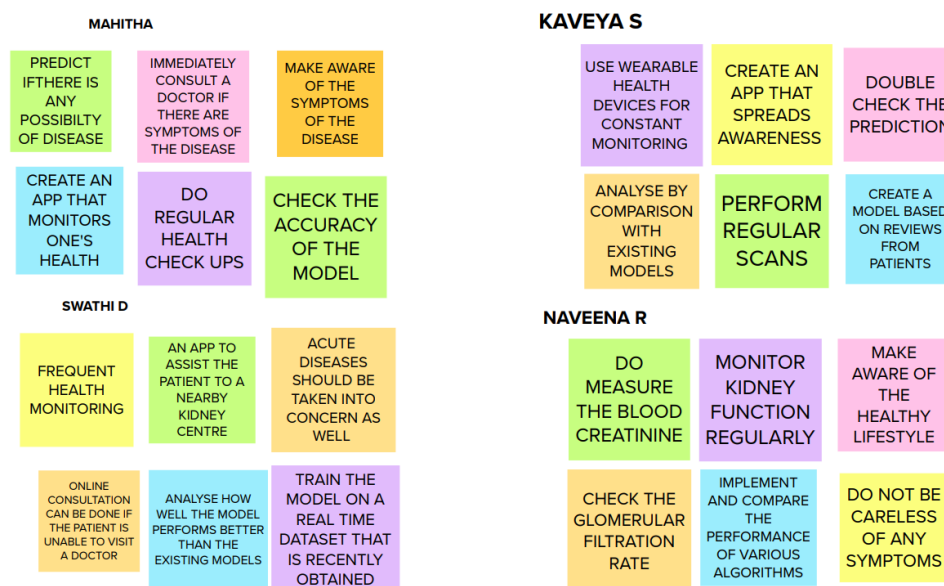
Empathy maps are split into 4 quadrants (Says, Thinks, Does, and Feels), with the user in the middle. Empathy maps provide a glance into who a user is as a whole. The *Says* quadrant contains what the user says or what he needs. The *Thinks* quadrant captures what the user is thinking throughout the experience. The *Does* quadrant encloses the actions the user takes. The *Feels* quadrant is the user's emotional state. The empathy map for Industry Specific Intelligent Fire Management System is shown in Fig 3.1.



**Fig 3.1 Empathy map**

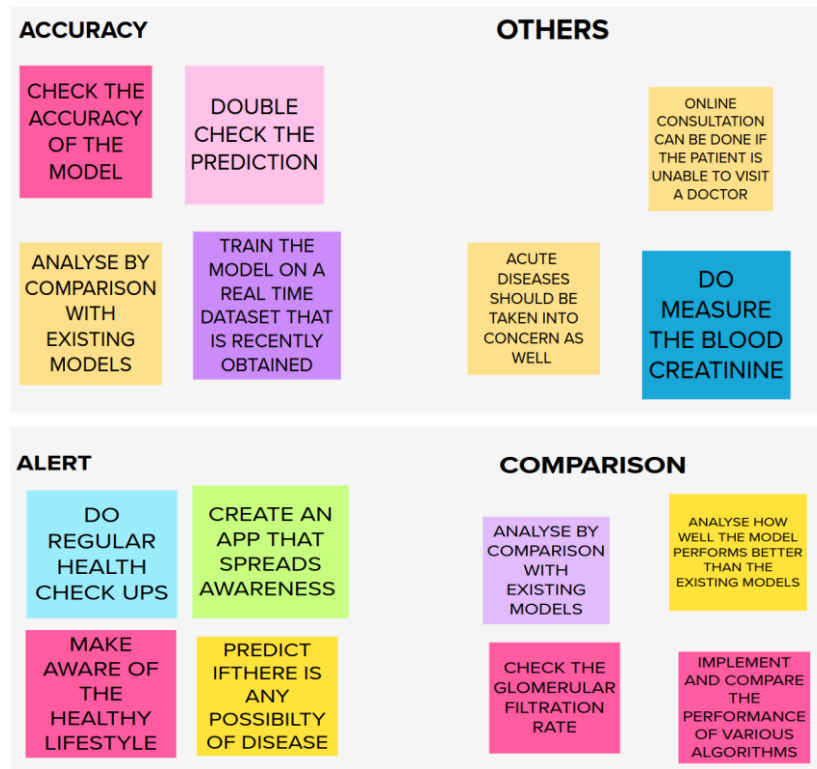
### **3.2 IDEATION AND BRAINSTORMING**

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity. Both brainstorming and ideation are processes invented to create new valuable ideas, perspectives, concepts and insights, and both are methods for envisioning new frameworks and systemic problem solving. The brainstorming chart for early detection of chronic kidney disease using machine learning is shown in Fig 3.2.



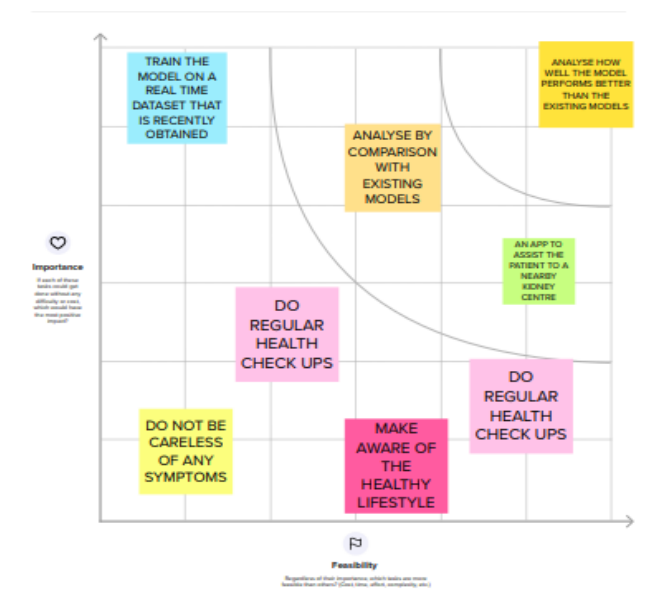
**Fig 3.2 Brainstorming**

The ideas are clusters based on similarity and each group is given a label. The clustering of ideas are done as shown in Fig 3.3.



**Fig 3.3 Clustering of ideas**

The shared ideas are grouped and given a label. Each idea is prioritised based on what is important and feasible. Prioritization of ideas is shown in Fig 3.4.



**Fig 3.4 Prioritization of ideas**

### **3.3 PROPOSED SOLUTION**

The proposed solution for Industry Specific Intelligent Fire Management System is shown in Table 3.1.

Table 3.1

<b>S.No.</b>	<b>Parameter</b>	<b>Description</b>
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none"><li>• Chronic Kidney Disease can lead to other crucial diseases which makes the person affected less immune.</li><li>• These diseases can be fatal as well.</li></ul>
2.	Idea / Solution description	<ul style="list-style-type: none"><li>• Machine learning technique is used to detect the chronic kidney disease sooner, thus enabling the doctors for earlier treatment</li></ul>
3.	Novelty / Uniqueness	<ul style="list-style-type: none"><li>• Accurate prediction of the disease is sooner</li><li>• This helps the doctors to treat the patients efficiently(early treatment of the disease)</li></ul>
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"><li>• Machine learning techniques enable the patients to know if they will be affected by chronic disease by predicting accurately</li></ul>
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"><li>• The model can be used by doctors for efficient treatment.(prior prediction of the disease)</li></ul>
6.	Scalability of the Solution	<ul style="list-style-type: none"><li>• The proposed solution helps the patients to know if they will be affected by chronic kidney diseases and the doctors in early detection of the disease, thus ensuring effective treatment</li></ul>

### **3.1 PROBLEM SOLUTION FIT**

The Problem-Solution Fit simply means that one have found a problem with the customer and that the solution one have realized for it actually solves the customer's problem. The Problem-Solution Fit is an important step towards the Product-Market Fit. The structure of Problem Solution Fit is given below:

1. **Customer State fit:** to make sure one understand the target group, their limitations and their currently available solutions, against which one is going to compete.
2. **Problem-Behavior fit:** to help one to identify the most urgent and frequent problems, understand the real reasons behind them and see which behavior supports it.
3. **Communication-Channel fit:** to help one to sharpen the communication with strong triggers, emotional messaging and reaching customers via the right channels.
4. **Solution guess:** translate all the validated data one have gathered into a solution that fits the customer state and his/her limitations, solves a real problem and taps into the common behavior of the target group.

The problem solution fit for Early detection of chronic kidney disease using machine learning is shown in Fig 3.5.

<b>1. CUSTOMER SEGMENT(S)</b> Who is your customer? <b>CS</b>  Patients prone to kidney diseases and those who have symptoms of it.	<b>6. CUSTOMER CONSTRAINTS</b> What constraints prevent your customers from taking action or limit their choices of solutions? <b>CC</b>  budget, no cash, access to medical facilities, lack of awareness, carelessness, insufficient knowledge on the seriousness of the disease	<b>5. AVAILABLE SOLUTIONS</b> Tests are taken to know the presence of the chronic kidney disease <b>AS</b>  Fatal deaths due to late diagnosis of the disease  Lack of mechanisms for early detection
<b>2. JOBS-TO-BE-DONE / PROBLEMS</b>  Create a model which predicts the disease with higher accuracy.  Analyse and compare with various existing models and choose the best model which fits the problem.	<b>9. PROBLEM ROOT CAUSE</b> <b>RC</b>  Unhealthy lifestyle of people leads to cause of the disease  Food habits which contains chemicals stimulating the deterioration of kidney health	<b>7. BEHAVIOUR</b> <b>BE</b>  Find the right things to do about the disease.  Infer about the symptoms that causes the chronic disease.  Calculate the benefits of predicting the disease in early stages
<b>3. TRIGGERS</b> <b>TR</b> Any sudden changes in health conditions like fainting etc Talking with other patients and knowing their plans of curing the disease	<b>10. YOUR SOLUTION</b> <b>SL</b> Create a model which predicts the disease with higher accuracy.	<b>8. CHANNELS of BEHAVIOUR</b> <b>CH</b> 8.1 ONLINE Take online tests now and then to know the current health status Check the lifestyle tips with respect to the disease in the
<b>4. EMOTIONS: BEFORE / AFTER</b> <b>EM</b> Lost Insecure Frustrated Feeling low Depressed	Analyze and compare with various existing models and choose the best model which fits the problem.  Survey in hospitals and patients to know the right way to predict the disease with absolute accuracy	Internet Build self awareness about the disease and symptoms by surfing through the web 8.2 OFFLINE Visit the doctor periodically Take regular health checkups and medications

**Fig 3.5 Problem Solution Fit**

## **CHAPTER 4**

### **REQUIREMENT ANALYSIS**

Requirements analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and Non-functional requirements.

#### **4.1 FUNCTIONAL REQUIREMENTS**

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

The following table 4.1 shows the functional requirements of the proposed solution.

<b>FR No.</b>	<b>Functional Requirement (Epic)</b>	<b>Sub Requirement (Story / Sub-Task)</b>
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Feature Selection	Blood test, Glomerular Filtration Rate (GFR)
FR-4	Feature Vectorization	Training and testing dataset should be developed
FR-5	Prediction	Model predicts the CKD using machine learning algorithms
FR-6	Classifier	Send all the output to the classifier and produce the final output results
FR-7	Detection	The model should be capable to retrieve and display with correct accuracy.



#### **4.1 NON-FUNCTIONAL REQUIREMENTS**

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like Portability, Security, Maintainability, Reliability, Scalability, Performance, Reusability, Flexibility.

The following table 4.2 shows the Non-Functional Requirements of the proposed solution.

<b>FR No.</b>	<b>Non-Functional Requirement</b>	<b>Description</b>
NFR-1	Usability	The user should have full access to the model to test himself/herself
NFR-2	Security	The user information given to the model should be secure.
NFR-3	Reliability	The prediction of the model should be more trustworthy to the user.
NFR-4	Performance	The accuracy of the prediction must be maximum.
NFR-5	Availability	It must be easily and cheaply available to the user.
NFR-6	Scalability	The model should be able to handle large number of users at the same time.

## **CHAPTER 5**

### **PROJECT DESIGN**

#### **5.1 DATA FLOW DIAGRAMS**

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

There are four main elements of a DFD — external entity, process, data store, and data flow.

##### **External entity**

An external entity, which are also known as terminators, sources, sinks, or actors, are an outside system or process that sends or receives data to and from the diagrammed system. They’re either the sources or destinations of information, so they’re usually placed on the diagram’s edges. External entity symbols are similar across models except for Unified, which uses a stick-figure drawing instead of a rectangle, circle, or square.

## Process

Process is a procedure that manipulates the data and its flow by taking incoming data, changing it, and producing an output with it. A process can do this by performing computations and using logic to sort the data, or change its flow of direction. Processes usually start from the top left of the DFD and finish on the bottom right of the diagram.

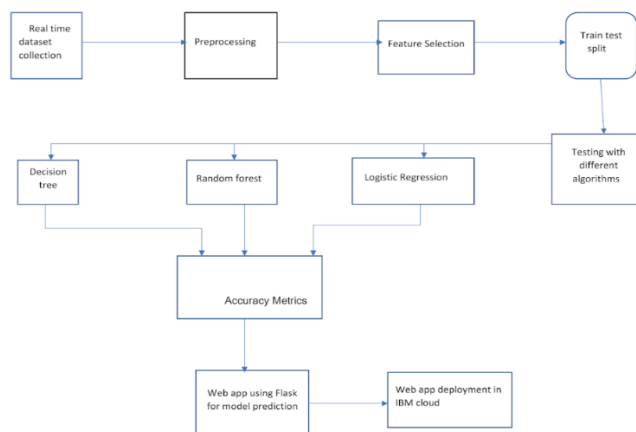
## Data store

Data stores hold information for later use, like a file of documents that's waiting to be processed. Data inputs flow through a process and then through a data store while data outputs flow out of a data store and then through a process.

## Data flow

Data flow is the path the system's information takes from external entities through processes and data stores. With arrows and succinct labels, the DFD can show the direction of the data flow.

The data flow diagram for early detection of chronic kidney disease using machine learning is shown in following figure 5.1

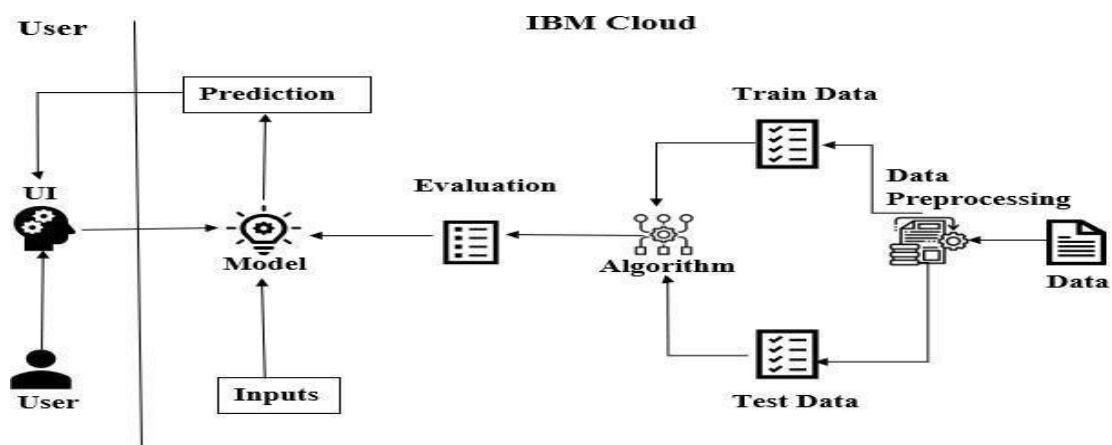


**Fig 5.1 Data flow diagram of Early detection of CKD using ML**

## **5.2 SOLUTION AND TECHNICAL ARCHITECTURE**

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
  - Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
  - Define features, development phases, and solution requirements.
  - Provide specifications according to which the solution is defined, managed, and delivered.
- The figure 5.2 shows the solution architecture of CKD detection



**Fig 5.2 Solution Architecture of CKD detection**

## **CHAPTER 6**

### **PROJECT PLANNING & SCHEDULING**

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

Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. Sprint planning is done in collaboration with the whole scrum team.

The sprint is a set period of time where all the work is done. However, before leap into action it is necessary to set up the sprint. It need to decide on how long the time box is going to be, the sprint goal, and where it is going to start. The sprint planning session kicks off the sprint by setting the agenda and focus. If done correctly, it also creates an environment where the team is motivated, challenged, and can be successful. The Table 6.1 shows the sprint planning and estimation of CKD detection

<b>Sprint</b>	<b>Functional Requirement (Epic)</b>	<b>User Story Number</b>	<b>User Story / Task</b>	<b>Story Points</b>	<b>Priority</b>
Sprint-1	Registration	USN-1	As a user, I can register for the diagnosis using my email and password	7	High
Sprint-1		USN-2	As a user, I will receive confirmation email on registering for the diagnosis tool	6	High
Sprint-4		USN-3	As a user, I can register for the application through my Gmail	6	Low

Sprint-1	Login	USN-4	As a user, I can log into the application by entering my credentials	6	High
Sprint-3	Dashboard	USN-5	As a user, I can see my past records and activities	6	High
Sprint-2	Entry form	USN-6	As a user, I must enter my pre-diagnostic test results	7	High
Sprint-3	Report	USN-7	As a user, I can view the report generated by the tool	7	High
Sprint-3	Remedies	USN-8	As a user, I will receive remedies to treat my symptoms	6	Medium
Sprint-4	Queries	USN-9	As a customer care executive, I must assist users that face problems through Q&A	6	Low
Sprint-4	Feedback	USN-10	As a customer care executive, I should get input for the tool's enhancement from users	7	Low
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
Sprint-2	Train model	USN-12	As an administrator, I must use the most suitable ML model for detection of CKD	6	High

## 6.1 **SPRINT DELIVERY SCHEDULE**

The sprint delivery plan is scheduled accordingly as shown in the below table 6.2 which consists of the sprints with respective to their duration, sprint start and end date and the releasing data.

<b>TITLE</b>	<b>DESCRIPTION</b>	<b>DATE</b>
<b>Literature Survey &amp; Information Gathering</b>	Literature survey on the project & gathering information by referring technical papers, research publications, journals etc.	2 SEPTEMBER 2022
<b>Prepare Empathy Map</b>	Prepare Empathy Map Canvas to capture the user Pains & Gains. Prepare list of problem Statements that are to be solved by this project.	9 SEPTEMBER 2022
<b>Ideation</b>	List the ideas by organizing a brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	16 SEPTEMBER 2022
<b>Proposed Solution</b>	Prepare the proposed solution document, which includes novelty, feasibility of idea, revenue model, social impact, scalability of solution, etc.	23 SEPTEMBER 2022
<b>Problem Solution Fit</b>	Prepare problem - solution fit document.	23 SEPTEMBER 2022

<b>Solution Architecture</b>	Prepare solution architecture document.	30 SEPTEMBER 2022
<b>Customer Journey</b>	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	7 OCTOBER 2022
<b>Functional Requirement</b>	Prepare the functional requirement document.	7 OCTOBER 2022
<b>Data Flow Diagrams</b>	Draw the data flow diagrams.	14 OCTOBER 2022
<b>Technology Architecture</b>	Prepare the technology architecture diagram.	14 OCTOBER 2022
<b>Prepare Milestone &amp; Activity List</b>	Prepare the milestones & activity list of the project.	21 OCTOBER 2022
<b>Project Development - Delivery of Sprint-1, 2, 3 &amp; 4</b>	Develop & submit the developed code by testing it.	11 NOVEMBER 2022



## **CHAPTER 7**

### **CODING AND SOLUTIONING**

#### **7.1 FEATURES**

- Message to the patient if he is affected by CKD
- Using mail for authentication
- After the prediction of the disease, the patient is notified so that he or she can take precautions.

#### **Basic Explanation:**

#### **Libraries required:**

```
In [2]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

-

### **Fig 7.1 Libraries Required**

#### **Pandas:**

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays

#### **Loading the dataset :**

The dataset for kidney disease prediction is a csv file and is loaded using read command. The dataset is viewed using head command.

```

In [4]: df=pd.read_csv(r'/kidney_disease.csv')

In [5]: print("Dataset shape is {}".format(df.shape))
Dataset shape is (400, 26)

In [6]: df.head()

```

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

5 rows x 26 columns

**Fig 7.2 Loading the dataset**

### Analyzing the features of dataset:

The features of the dataset is analysed by viewing the columns and getting information of each column. This helps in data pre-processing

```
In [7]: df.columns
```

```
Out[7]: Index(['id', 'age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr',  
             'bu', 'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',  
             'appet', 'pe', 'ane', 'classification'],  
            dtype='object')
```

```
In [8]: df.info()
```

```
RangeIndex: 400 entries, 0 to 399  
Data columns (total 26 columns):  
#   Column                Non-Null Count  Dtype  
---  ---  
0   id                     400 non-null    int64  
1   age                   391 non-null    float64  
2   bp                     388 non-null    float64  
3   sg                     353 non-null    float64  
4   al                     354 non-null    float64  
5   su                     351 non-null    float64  
6   rbc                    248 non-null    object  
7   pc                     335 non-null    object  
8   pcc                    396 non-null    object  
9   ba                     396 non-null    object  
10  bgr                    356 non-null    float64  
11  bu                     381 non-null    float64  
12  sc                     383 non-null    float64  
13  sod                    313 non-null    float64  
14  pot                    312 non-null    float64  
15  hemo                   348 non-null    float64  
16  pcv                    330 non-null    object  
17  wc                     295 non-null    object  
18  rc                     270 non-null    object  
19  htn                    398 non-null    object  
20  dm                     398 non-null    object  
21  cad                    398 non-null    object  
22  appet                  399 non-null    object  
23  pe                     399 non-null    object  
24  ane                    399 non-null    object  
25  classification         400 non-null    object  
dtypes: float64(11), int64(1), object(14)  
memory usage: 81.4+ KB
```

**Fig 7.3 Features of dataset**

### **Data pre-processing:**

Data pre-processing is a significant step in machine learning. This is begun with analyzing the value counts of each column so as to decide whether entropy encoding is needed or not. Continuous values are converted to categorical values because the output should be a classified one.

```

In [9]: df['pcv'].value_counts()

Out[9]: 41      21
        52      21
        44      19
        48      19
        40      16
        43      14
        42      13
        45      13
        32      12
        36      12
        33      12
        50      12
        28      12
        34      11
        37      11
        30       9
        29       9
        35       9

In [15]: df['pcv']=df['pcv'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t43','43').replace('\t?', 'NaN'))

# cleaning "WC"
df['wc']=df['wc'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t?', 'NaN').replace('\t6200','6200').replace('\t8400','8400'))

# cleaning "RC"
df['rc']=df['rc'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t?', 'NaN'))

# cleaning "dm"
df['dm']=df['dm'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\tno','no').replace('\tyes','yes').replace(' yes','yes'))

# cleaning "CAD"
df['cad']=df['cad'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\tno','no'))

# cleaning "Classification"
df['classification']=df['classification'].apply(lambda x:x if type(x)==type(3.5) else x.replace('ckd\t','ckd'))

In [16]: #explicitly converting numerical columns
mistyped=['pcv','rc','wc']
for i in mistyped:
    df[i]=df[i].astype('float')

In [17]: #categorical columns
cat_cols=list(df.select_dtypes('object'))
cat_cols

In [19]: #handling missing data
df.isnull().sum().sort_values(ascending=False)

```

**Fig 7.4 Data pre-processing**

### Handling missing values:

The dataset may have missing values which affects the performance of the model significantly. The missing values are hence preprocessed before training the model.

```

In [28]: df['pe'].mode()

Out[28]: 0    no
dtype: object

In [29]: df['ane'].mode()

Out[29]: 0    no
dtype: object

In [30]: #filling missing values with mode for categorical attributes
df['rbc'].fillna('normal',inplace=True)
df['pc'].fillna('normal',inplace=True)
df['pcc'].fillna('notpresent',inplace=True)
df['ba'].fillna('notpresent',inplace=True)
df['htn'].fillna('no',inplace=True)
df['dm'].fillna('no',inplace=True)
df['cad'].fillna('no',inplace=True)
df['appet'].fillna('good',inplace=True)
df['pe'].fillna('no',inplace=True)
df['ane'].fillna('no',inplace=True)

In [31]: #filling missing values with median for numerical attributes
for col in num_cols:
    df[col]=df[col].fillna(df[col].median())

In [32]: df.isna().sum().sort_values(ascending=False)

```

**Fig 7.5 Handling missing values**

### Label Encoding:

In machine learning, **label encoding** is the process of converting the values of a categorical variable into integer values. Label Encoding cites the transmogrification of the labels into the numeric form to change it into a form that can be read by the machine. Machine learning algorithms can thereafter determine in a correct way as to how these labels must be managed. It is a crucial pre-processing measure during the integrated dataset in supervised learning.

```

In [35]: df['classification'].unique()

Out[35]: array([1, 0])

In [36]: #label encoding for categorical attributes
df['rbc']=df['rbc'].map({'normal':0,'abnormal':1})
df['pc']=df['pc'].map({'normal':0,'abnormal':1})
df['pcc']=df['pcc'].map({'notpresent':0,'present':1})
df['ba']=df['ba'].map({'notpresent':0,'present':1})
df['htn']=df['htn'].map({'no':0,'yes':1})
df['dm']=df['dm'].map({'no':0,'yes':1})
df['cad']=df['cad'].map({'no':0,'yes':1})
df['pe']=df['pe'].map({'no':0,'yes':1})
df['ane']=df['ane'].map({'no':0,'yes':1})
df['appet']=df['appet'].map({'good':0,'poor':1})

```

**Fig 7.6 Label Encoding**

### **Splitting the dataset into training and testing dataset:**

The dataset is split into training and testing dataset as a next step after processing the data. This is to make sure the model is trained on a dataset and when given a new dataset, the model predicts it correctly.

```

In [37]: from sklearn.model_selection import train_test_split
x=df.drop('classification',axis=1)#independent
y=df['classification']#dependent

X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
print("X_train size {} , X_test size {}".format(X_train.shape,X_test.shape))

X_train size (320, 25) , X_test size (80, 25)

```

**Fig 7.7 Train- Test Split**

### **Performance analysis of the trained model:**

The performance of the model is evaluated using confusion metrics which gives true positive, true negative, false positive and false negative values in prediction on the trained dataset.

```
In [19]: tuned_parameters = [{'n_estimators':[7,8,9,10,11,12,13,14,15,16], 'max_depth':[2,3,4,5,6,None],
                             'class_weight':[None,{0: 0.33,1:0.67},'balanced'], 'random_state':[42]}]
clf = GridSearchCV(RandomForestClassifier(), tuned_parameters, cv=10, scoring='f1')
clf.fit(X_train, y_train)

print("Detailed classification report:")
y_true, lr_pred = y_test, clf.predict(X_test)
print(classification_report(y_true, lr_pred))

confusion = confusion_matrix(y_test, lr_pred)
print('Confusion Matrix:')
print(confusion)

# Determine the false positive and true positive rates
fpr,tpr,roc_auc = auc_scorer(clf, X_test, y_test, 'RF')

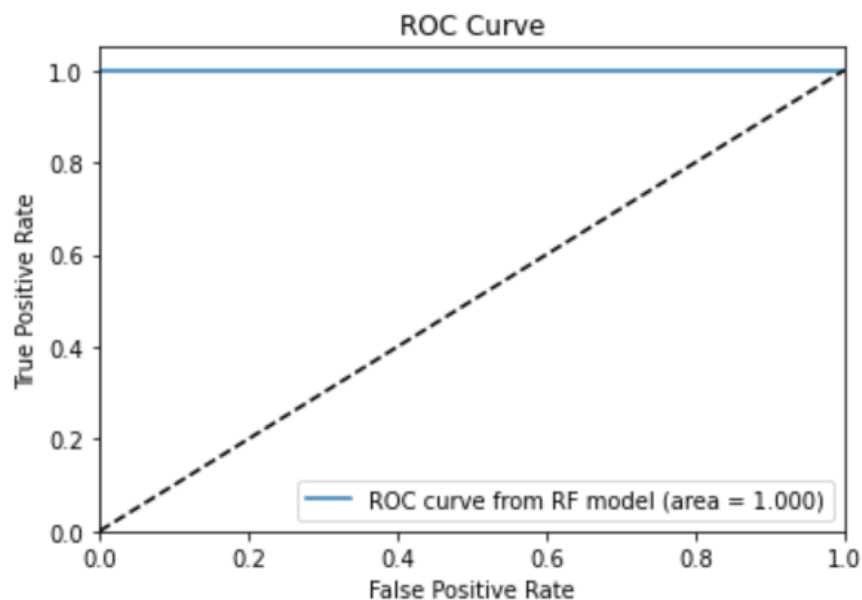
print('Best parameters:')
print(clf.best_params_)
clf_best = clf.best_estimator_
```

Detailed classification report:

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	39
1.0	1.00	1.00	1.00	14
accuracy			1.00	53
macro avg	1.00	1.00	1.00	53
weighted avg	1.00	1.00	1.00	53

Confusion Matrix:

```
[[39  0]
 [ 0 14]]
```



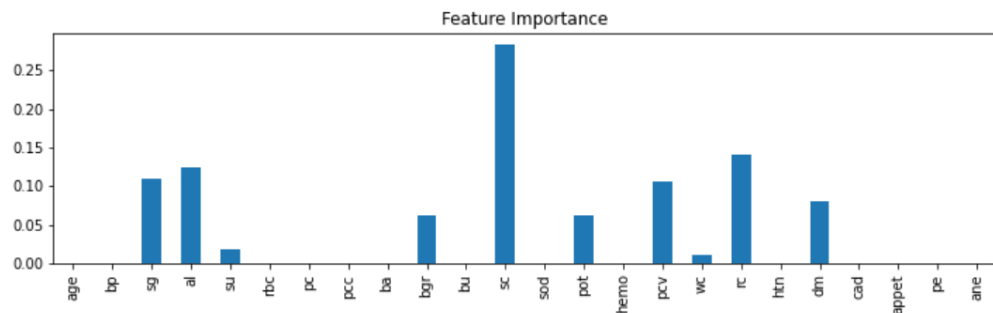
**Fig 7.8 Performance analysis on trained data**

## Feature Importance:

The important feature in determining the presence of chronic kidney disease is calculated.

```
In [20]: plt.figure(figsize=(12,3))
features = X_test.columns.values.tolist()
importance = clf_best.feature_importances_.tolist()
feature_series = pd.Series(data=importance,index=features)
feature_series.plot.bar()
plt.title('Feature Importance')
```

Out[20]: Text(0.5, 1.0, 'Feature Importance')



**Fig 7.9 Feature Importance**

## Save the model:

The trained model is saved in the form of a pkl file so that it can be accessed easily.

```
In [24]: import pickle
pickle.dump(clf_best, open('randomclass_chronic', 'wb'))
```

With proper tuning of parameters using cross-validation in the training set, the Random Forest Classifier achieves an accuracy of 88.8% and an ROC AUC of 99.2%.

**Fig 7.10 Save the model**

## Flask Implementation:

Flask is a micro web framework written in Python . It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.



```
In [ ]: from flask import Flask, request, redirect, render_template
app = Flask(__name__)
@app.route("/",methods=['GET', 'POST'])
def index():
    return render_template('index.html')
@app.route("/val",methods=['POST'])

def val():
    test=[]
    if request.method == 'POST':
        test.append(request.form.get("age"))
        test.append(request.form.get("bp"))
        test.append(request.form.get("sg"))
        test.append(request.form.get("al"))
        test.append(request.form.get("su"))
        rb=request.form.get("rbc")
        if rb=='abnormal':
            test.append(1)
        else:
            test.append(0)
        pc=request.form.get("pc")
        if pc=='abnormal':
            test.append(1)
        else:
            test.append(0)
        pcc=request.form.get("pcc")
        if pcc=='present':
            test.append(1)
        else:
            test.append(0)
        ba=request.form.get("ba")
        if ba=='present':
            test.append(1)
        else:
            test.append(0)
        test.append(request.form.get("bgr"))
        test.append(request.form.get("bu"))
        test.append(request.form.get("sc"))
        test.append(request.form.get("sod"))
```

**Fig 7.11 Flask Implementation**

```

test.append(request.form.get("pot"))
test.append(request.form.get("hemo"))
test.append(request.form.get("pcv"))
test.append(request.form.get("wc"))
test.append(request.form.get("rc"))
ht=request.form.get("htn")
if ht=='yes':
    test.append(1)
else:
    test.append(0)
d=request.form.get("dm")
if d=='yes':
    test.append(1)
else:
    test.append(0)
ca=request.form.get("cad")
if ca=='yes':
    test.append(1)
else:
    test.append(0)
ap=request.form.get("appet")
if ap=='good':
    test.append(1)
elif ap=='poor':
    test.append(0)
else:
    test.append(np.nan)
p=request.form.get("pe")
if p=='yes':
    test.append(1)
else:
    test.append(0)
an=request.form.get("ane")
if an=='yes':
    test.append(1)
else:
    test.append(0)
print(test)
test_df=pd.DataFrame(test)
test_df=np.array(test_df).reshape(1, -1)

```

```

ans1=loaded_class.predict(test_df)
ans2=loaded_reg.predict(test_df)
if int(ans1)==1:
    answer1="Sorry to say!! You have CHRONIC DISEASE!!!"
    return render_template('rename.html',answer1=answer1,answer2=ans2)
else:
    answer1="Happy to say that you don't have CHRONIC DISEASE"
    return render_template('rename2.html',answer1=answer1,answer2=ans2)

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

```

```

* Serving Flask app "__main__" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

```

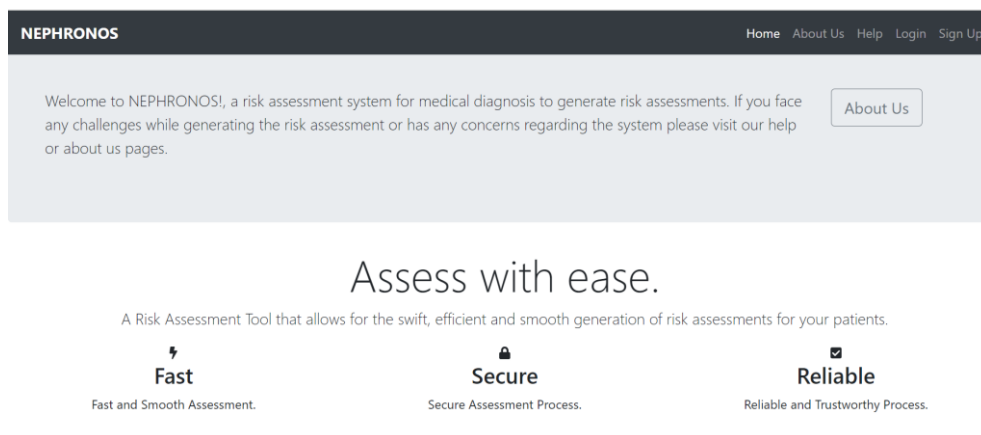
**Fig 7.12 Flask Implementation**

## **CHAPTER 8**

### **TESTING AND RESULTS**

#### **8.1 TEST CASES**

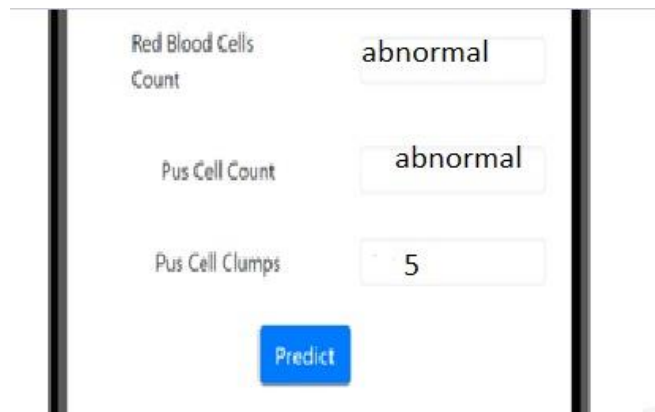
From the below figure 8.1 shows that the login page of the NEPHRONOS website, where the login details are available. The website provides some options like login, signup, home and about the page. The Fig 8.2 shows that the Kidney disease prediction page where the patient can enter the details of blood pressure, blood sugar level, albumin and specific gravity.



**Fig 8.1 Login page of NEPHRONOS website**

The screenshot shows the 'Kidney Disease Prediction' page on the NEPHRONOS website. The top navigation bar is dark with the logo 'NEPHRONOS' on the left and links for 'Get Started', 'Kidney Disease', and 'Logout' on the right. The main heading is 'Kidney Disease Prediction' with a subtext 'Please enter the patient details'. Below this, a form is displayed with four input fields: 'Blood Pressure' with the value '80', 'Specific Gravity' with the value '1.025', 'Albumin' with the value '0', and 'Blood Sugar Level' with the value 'normal'.

**Fig 8.2 Patient details**



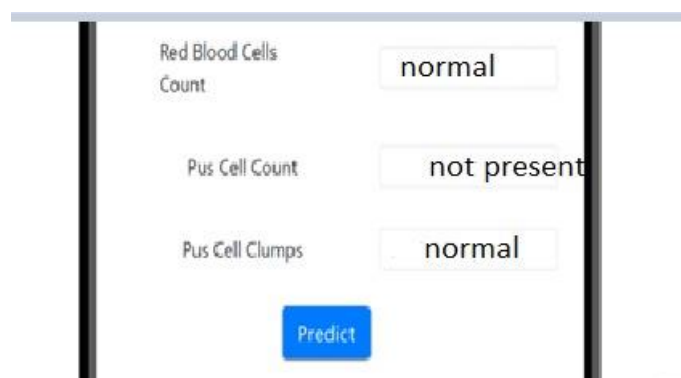
Red Blood Cells Count	abnormal
Pus Cell Count	abnormal
Pus Cell Clumps	5
<input type="button" value="Predict"/>	

You are affected by Chronic Kidney Disease.Consult a doctor as soon as possible

---

**Fig 8.3 Output of CKD affected patient**

Fig 8.3 shows the output of chronic kidney disease affected patient. The website calculate the red blood cells, puss cell, puss cell clumps after the patient enters the details and predict whether the patient is affected or not. If affected means, it will print “You are affected by Chronic kidney disease. Consult a doctor as soon as possible”. Fig 8.4 shows the output of Chronic kidney disease not affected patient.



Red Blood Cells Count	normal
Pus Cell Count	not present
Pus Cell Clumps	normal
<input type="button" value="Predict"/>	

➡ Woah!!!You are not affected by Chronic Kidney Disease.

---

**Fig 8.4 Output of CKD not affected patient**

## CHAPTER 9

### PERFORMANCE METRICS

Fig 9.1 shows the confusion matrix of the model before training and Fig 9.2 shows the performance of the model after training.

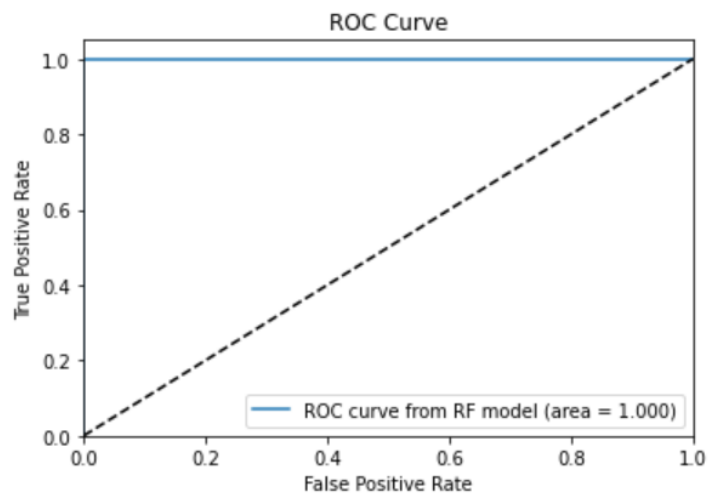
```
Detailed classification report:
              precision    recall  f1-score   support

     0.0         1.00      1.00      1.00        39
     1.0         1.00      1.00      1.00        14

 accuracy          1.00
 macro avg         1.00      1.00      1.00        53
 weighted avg      1.00      1.00      1.00        53
```

Confusion Matrix:

```
[[39  0]
 [ 0 14]]
```



Best parameters:

```
{'class_weight': None, 'max_depth': 2, 'n_estimators': 8, 'random_state': 42}
```

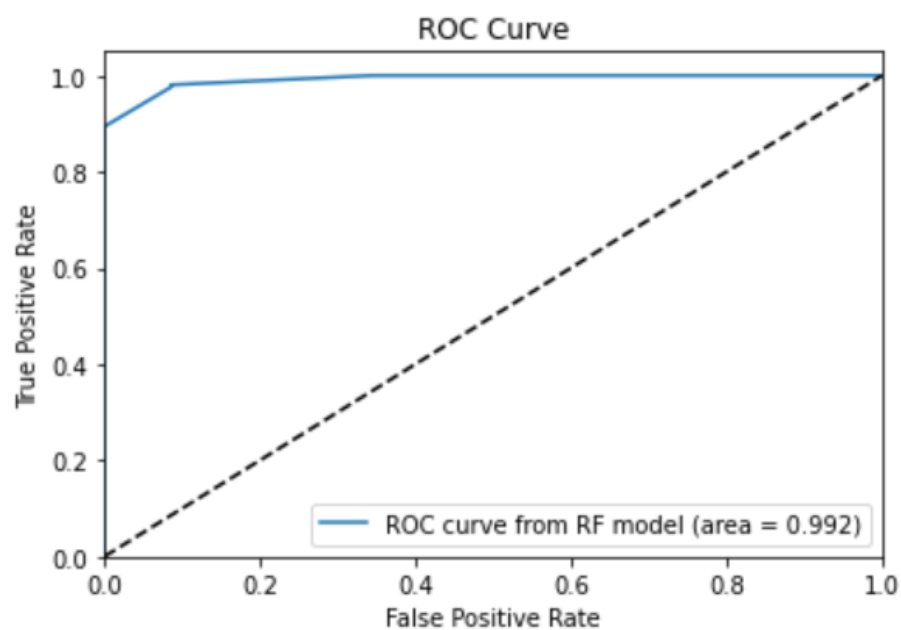
**Fig 9.1 ROC Curve**

	precision	recall	f1-score	support
0.0	0.56	1.00	0.72	35
1.0	1.00	0.87	0.93	207
accuracy			0.89	242
macro avg	0.78	0.93	0.83	242
weighted avg	0.94	0.89	0.90	242

Confusion Matrix:

```
[[ 35  0]
 [ 27 180]]
```

Accuracy: 0.888430



**Fig 9.2 ROC Curve after model training**

## **CHAPTER 10**

### **ADVANTAGES AND DISADVANTAGES**

Chronic kidney disease is a major burden on the healthcare system because of its increasing prevalence, high risk of progression and mortality prognosis. It is rapidly becoming a global health crisis.

Reliable prediction is one of the most important and essential outcomes of prediction using machine learning techniques. It is critical to have reliable techniques at predicting CKD in its early stages.

Machine learning techniques are excellent in predicting CKD by offering a methodology for predicting CKD status using clinical data, which incorporates data preprocessing, a technique for managing missing values, data aggregation, and feature extraction.

A number of physiological variables, as well as ML techniques such as logistic regression (LR), k- means clustering, decision tree (DT) classification, and -nearest neighbor (KNN), are being used to train the models for reliable prediction.

Accuracy is also the important factors which is inevitable while prediction and we can achieve upto 97% accuracy by using machine learning models in prediction. Compared to prior research and existing models, the accuracy rate of the models employed in this study is considerably greater, implying that they are more trustworthy.

Diagnosis of kidney impairment early may help in rectification, which is not always possible. But we can assure the avoidance of serious damage, by having got a better understanding of a few indicators caused by kidney disease.

Early diagnosis and treatment of CKD is a serious challenge for the medical community. The treating physician (nephrologist) is called on the one hand to slow down the progression of the disease to more advanced stages, and

if possible, to suspend it, and on the other hand, to treat the above-mentioned systemic manifestations.

The advances in sensor networks, communication technologies, data science and statistical processing have rendered ML techniques as important tools in various health-oriented applications, such as in the early diagnosis of several chronic conditions.

The early detection of CKD has been facilitated by the implementation of routine reporting of estimated glomerular filtration rates (eGFRs) and by education of primary care physicians on the implications of detecting a decreased eGFR with respect to patient safety as well as to cardiovascular and renal outcomes.

An  $\text{eGFR} < 60$  is very common in older people. An  $\text{eGFR} < 45$  identifies a smaller sub-group of older people with significant comorbidity, impaired functional state and a high risk of potentially reversible consequences such as anaemia. The benefits of identifying older people with an  $\text{eGFR} > 45$  need to be determined.

The CKD risk groups are individuals with diabetes, hypertension and a family history of renal disease. Therapy must emphasize the maximal reduction of cardiovascular risk. The complications of CKD such as anemia and renal osteodystrophy can be identified and treated on time. Most patients with chronic kidney disease are detected in the community, therefore their initial care must be organized at the level of primary care, along with programs for hypertension and diabetes.



## **CHAPTER 11**

### **CONCLUSION**

Chronic kidney disease (CKD) is rapidly becoming a global health crisis. Chronic kidney disease (CKD) is a major public health concern around the world, with negative outcomes such as renal failure, cardiovascular disease, and early death . Unhealthy dietary habits and insufficient water consumption are significant contributors to this disease. Without kidneys, a person can only live for 18 days on average, requiring kidney transplantation and dialysis. It is critical to have reliable techniques at predicting CKD in its early stages.

Chronic kidney disease is a condition characterized by progressive loss of kidney function over time. It is a silent disease, as most sufferers have no symptoms. Early diagnosis and treatment of CKD is a serious task for the medical community that resorts to ML theory to design an efficient solution to this challenge. In the present work, a methodology based on supervised learning is described, which aims to create efficient models for predicting the risk of CKD occurrence by mainly focusing on probabilistic, tree-based and ensemble learning-based models

According to a 2010 study by the Global Burden of Disease Study (GBDS), chronic kidney disease (CKD) was listed as the 18th leading cause of mortality worldwide, up from 27th in 1990 . Chronic kidney disease affects over 500 million people worldwide with a disproportionately high burden in developing countries, particularly South Asia and sub-Saharan Africa . According to a 2015 study, there were 110 million people with CKD in high-income nations (men 48.3 million, women 61.7 million), but 387.5 million in low- and middle-income countries .

According to the study, their precision was 96.25 percent, and their accuracy was 97 percent. Compared to prior research, the accuracy percent of the models used in this investigation is considerably higher, indicating that the

models used in this study are more reliable than those used in previous studies. When cross validation measurements are used in the prediction of chronic kidney disease, the LR method outperforms the other processes.

## **CHAPTER 12**

### **FUTURE SCOPE**

Future research may build on this work by developing a web application that incorporates these algorithms and using a bigger dataset than the one utilized in this study. This will aid in the achievement of improved outcomes as well as the accuracy and efficiency with which healthcare practitioners can anticipate kidney issues. This will enhance the dependability of the framework as well as the framework's presentation. The hope is that it would encourage people to seek early treatment for chronic renal disease and to make improvements in their lives.

In future work, we aim to direct our research on Deep Learning methods by applying the Long-Short-term-Memory (LSTM) and CNN and investigate the performance boost that these models may provide. To exploit the capabilities of these models, we aim to follow two directions. The former will apply a data augmentation method to enhance the limited-size dataset before feeding it to the ML models, such as an SVR-based additive input doubling method. In the latter, we will experiment from the beginning with a large-scale non-synthetic database.

## **SOURCE CODE**

### **PYTHON SCRIPT**

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

%matplotlib inline

def auc_scorer(clf, X, y, model): # Helper function to plot the ROC curve
    if model=='RF':
        fpr, tpr, _ = roc_curve(y, clf.predict_proba(X)[:,:1])
    elif model=='SVM':
        fpr, tpr, _ = roc_curve(y, clf.decision_function(X))
    roc_auc = auc(fpr, tpr)

    plt.figure() # Plot the ROC curve
    plt.plot(fpr, tpr, label='ROC curve from '+model+' model (area = %0.3f)' %
roc_auc)
    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC Curve')
    plt.legend(loc="lower right")
    plt.show()

    return fpr,tpr,roc_auc

df = pd.read_csv("/kidney_disease.csv")

df.head()
```

```

df[['htn','dm','cad','pe','ane']] =
df[['htn','dm','cad','pe','ane']].replace(to_replace={'yes':1,'no':0})
df[['rbc','pc']] = df[['rbc','pc']].replace(to_replace={'abnormal':1,'normal':0})
df[['pcc','ba']] = df[['pcc','ba']].replace(to_replace={'present':1,'notpresent':0})
df[['appet']] = df[['appet']].replace(to_replace={'good':1,'poor':0,'no':np.nan})
df['classification'] =
df['classification'].replace(to_replace={'ckd':1.0,'ckd\t':1.0,'notckd':0.0,'no':0.0})
df.rename(columns={'classification':'class'},inplace=True)

df['pe'] = df['pe'].replace(to_replace='good',value=0) # Not having pedal edema is
good
df['appet'] = df['appet'].replace(to_replace='no',value=0)
df['cad'] = df['cad'].replace(to_replace='tno',value=0)
df['dm'] = df['dm'].replace(to_replace={'\tno':0,'\tyes':1,' yes':1, ':np.nan'})
df.drop('id',axis=1,inplace=True)

df.head()

corr_df = df2.corr()

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

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

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

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

tuned_parameters =
[{'n_estimators':[7,8,9,10,11,12,13,14,15,16],'max_depth':[2,3,4,5,6,None],
    'class_weight':[None,{0: 0.33,1:0.67}],'balanced'],'random_state':[42]]
clf = GridSearchCV(RandomForestClassifier(), tuned_parameters,
cv=10,scoring='f1')
clf.fit(X_train, y_train)

```

```

print("Detailed classification report:")
y_true, lr_pred = y_test, clf.predict(X_test)
print(classification_report(y_true, lr_pred))

confusion = confusion_matrix(y_test, lr_pred)
print('Confusion Matrix:')
print(confusion)

# Determine the false positive and true positive rates
fpr, tpr, roc_auc = auc_scorer(clf, X_test, y_test, 'RF')

print('Best parameters:')
print(clf.best_params_)
clf_best = clf.best_estimator_

plt.figure(figsize=(12,3))
features = X_test.columns.values.tolist()
importance = clf_best.feature_importances_.tolist()
feature_series = pd.Series(data=importance, index=features)
feature_series.plot.bar()
plt.title('Feature Importance')

# Are there correlation in missing values?
corr_df = pd.isnull(df).corr()

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

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

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

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.show()

```

```

df2 = df.dropna(axis=0)
no_na = df2.index.tolist()
some_na = df.drop(no_na).apply(lambda x: pd.to_numeric(x,errors='coerce'))
some_na = some_na.fillna(0) # Fill up all Nan by zero.

X_test = some_na.iloc[:, :-1]
y_test = some_na['class']
y_true = y_test
lr_pred = clf_best.predict(X_test)
print(classification_report(y_true, lr_pred))

confusion = confusion_matrix(y_test, lr_pred)
print('Confusion Matrix:')
print(confusion)

print('Accuracy: %3f' % accuracy_score(y_true, lr_pred))
# Determine the false positive and true positive rates
fpr,tpr,roc_auc = auc_scorer(clf_best, X_test, y_test, 'RF')

import pickle
pickle.dump(clf_best, open('randomclass_chronic', 'wb'))

```

## **HTML SCRIPT:**

```
<!DOCTYPE  
html>
```

```
<html>  
<head>  
<meta charset="UTF-8">  
<title>Login Page</title>  
<style><!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<title>Login Page</title>  
<style>
```

```
input[type=text], input[type=password] {  
    width: 100%;  
    padding: 12px 20px;  
    margin: 8px 0;  
    display: inline-block;  
    border: 1px solid #ccc;  
    box-sizing: border-box;  
}
```

```
button {  
    background-color: #04AA6D;  
    color: white;  
    padding: 14px 20px;  
    margin: 8px 0;  
    border: none;  
    cursor: pointer;  
    width: 100%;  
}
```

```
button:hover {  
    opacity: 0.8;  
}
```

```
.cancelbtn {  
    width: auto;
```



```
padding: 10px 18px;
background-color: #f44336;
}
```

```
.imgcontainer {
text-align: center;
margin: 24px 0 12px 0;
}
```

```
img.avatar {
width: 40%;
border-radius: 50%;
}
```

```
.container {
padding: 16px;
}
```

```
span.psw {
float: right;
padding-top: 16px;
}
```

```
/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
span.psw {
display: block;
float: none;
}
.cancelbtn {
width: 100%;
}
}
```

```
.header {
overflow: hidden;
background-color: #f1f1f1;
padding: 20px 10px;
}
```

```
/* Style the header links */
.header a {
float: left;
color: black;
```

```
text-align: center;
padding: 12px;
text-decoration: none;
font-size: 18px;
line-height: 25px;
border-radius: 4px;
}
```

/\* Style the logo link (notice that we set the same value of line-height and font-size to prevent the header to increase when the font gets bigger \*/

```
.header a.logo {
  font-size: 75px;
  font-weight: bold;
  align: center;
}
```

/\* Change the background color on mouse-over \*/

```
.header a:hover {
  background-color: #ddd;
  color: black;
}
```

/\* Style the active/current link\*/

```
.header a.active {
  background-color: dodgerblue;
  color: white;
}
```

/\* Float the link section to the right \*/

```
.header-right {
  float: right;
}
```

/\* Add media queries for responsiveness - when the screen is 500px wide or less, stack the links on top of each other \*/

```
@media screen and (max-width: 500px) {
  .header a {
    float: none;
    display: block;
    text-align: left;
  }
  .header-right {
    float: none;
  }
}
```

```

    }
}
    * {box-sizing: border-box}

/* Add padding to containers */
.container {
    padding: 16px;
}

/* Full-width input fields */
input[type=text], input[type=password] {
    width: 100%;
    padding: 15px;
    margin: 5px 0 22px 0;
    display: inline-block;
    border: none;
    background: #f1f1f1;
}

input[type=text]:focus, input[type=password]:focus {
    background-color: #ddd;
    outline: none;
}

/* Overwrite default styles of hr */
hr {
    border: 1px solid #f1f1f1;
    margin-bottom: 25px;
}

/* Set a style for the submit/register button */
.registerbtn {
    background-color: #04AA6D;
    color: white;
    padding: 16px 20px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: 100%;
    opacity: 0.9;
}

.registerbtn:hover {
    opacity: 1;
}

```

```

}

/* Add a blue text color to links */
a {
    color: dodgerblue;
}

/* Set a grey background color and center the text of the "sign in"
section */
.signin {
    background-color: #f1f1f1;
    text-align: center;
}
</style>
</head>
<body>

<form action="/action_page.php" method="post">
    <div class="header">

        <a href="#default" class="logo">Chronic Kidney Disease Prediction</a>
        <div class="header-right">
            <a class="active" href="#home">Home</a>
            <a href="#contact">Contact</a>
            <a href="#about">About</a>
        </div>
    </div>
    <h2>Login Form</h2>
    <div class="container">
        <label for="uname"><b>Username</b></label>
        <input type="text" placeholder="Enter Username" name="uname"
required>

        <label for="psw"><b>Password</b></label>
        <input type="password" placeholder="Enter Password"
name="psw" required>

        <button type="submit">Login</button>
        <label>
            <input type="checkbox" checked="checked" name="remember">
Remember me

```

```

    </label>
</div>

<div class="container" style="background-color:#f1f1f1">
    <button type="button" class="cancelbtn">Cancel</button>
    <span class="psw">Forgot <a href="#">password?</a></span>
</div>
</form>

</body>
</html>

```

```

input[type=text], input[type=password] {
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    border: 1px solid #ccc;
    box-sizing: border-box;
}

```

```

button {
    background-color: #04AA6D;
    color: white;
    padding: 14px 20px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: 100%;
}

```

```

button:hover {
    opacity: 0.8;
}

```

```

.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

```

```

.imgcontainer {
    text-align: center;
}

```

```

    margin: 24px 0 12px 0;
}

img.avatar {
    width: 40%;
    border-radius: 50%;
}

.container {
    padding: 16px;
}

span.psw {
    float: right;
    padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
    span.psw {
        display: block;
        float: none;
    }
    .cancelbtn {
        width: 100%;
    }
}

.header {
    overflow: hidden;
    background-color: #f1f1f1;
    padding: 20px 10px;
}

/* Style the header links */
.header a {
    float: left;
    color: black;
    text-align: center;
    padding: 12px;
    text-decoration: none;
    font-size: 18px;
    line-height: 25px;
    border-radius: 4px;

```

```
}
```

```
/* Style the logo link (notice that we set the same value of line-height  
and font-size to prevent the header to increase when the font gets  
bigger */
```

```
.header a.logo {  
    font-size: 75px;  
    font-weight: bold;  
    align: center;  
}
```

```
/* Change the background color on mouse-over */
```

```
.header a:hover {  
    background-color: #ddd;  
    color: black;  
}
```

```
/* Style the active/current link*/
```

```
.header a.active {  
    background-color: dodgerblue;  
    color: white;  
}
```

```
/* Float the link section to the right */
```

```
.header-right {  
    float: right;  
}
```

```
/* Add media queries for responsiveness - when the screen is 500px  
wide or less, stack the links on top of each other */
```

```
@media screen and (max-width: 500px) {  
    .header a {  
        float: none;  
        display: block;  
        text-align: left;  
    }  
    .header-right {  
        float: none;  
    }  
}  
* {box-sizing: border-box}
```

```
/* Add padding to containers */
```

```
.container {
```

```

padding: 16px;
}

/* Full-width input fields */
input[type=text], input[type=password] {
width: 100%;
padding: 15px;
margin: 5px 0 22px 0;
display: inline-block;
border: none;
background: #f1f1f1;
}

input[type=text]:focus, input[type=password]:focus {
background-color: #ddd;
outline: none;
}

/* Overwrite default styles of hr */
hr {
border: 1px solid #f1f1f1;
margin-bottom: 25px;
}

/* Set a style for the submit/register button */
.registerbtn {
background-color: #04AA6D;
color: white;
padding: 16px 20px;
margin: 8px 0;
border: none;
cursor: pointer;
width: 100%;
opacity: 0.9;
}

.registerbtn:hover {
opacity: 1;
}

/* Add a blue text color to links */
a {
color: dodgerblue;
}

```



```

/* Set a grey background color and center the text of the "sign in"
section */
.signin {
    background-color: #f1f1f1;
    text-align: center;
}
</style>
</head>
<body>

<form action="/action_page.php" method="post">
    <div class="header">

        <a href="#default" class="logo">Chronic Kidney Disease Prediction</a>
        <div class="header-right">
            <a class="active" href="#home">Home</a>
            <a href="#contact">Contact</a>
            <a href="#about">About</a>
        </div>
    </div>
    <h2>Login Form</h2>
    <div class="container">
        <label for="uname"><b>Username</b></label>
        <input type="text" placeholder="Enter Username" name="uname"
required>

        <label for="psw"><b>Password</b></label>
        <input type="password" placeholder="Enter Password"
name="psw" required>

        <button type="submit">Login</button>
        <label>
            <input type="checkbox" checked="checked" name="remember">
Remember me
        </label>
    </div>

    <div class="container" style="background-color:#f1f1f1">
        <button type="button" class="cancelbtn">Cancel</button>
        <span class="psw">Forgot <a href="#">password?</a></span>

```

```
</div>  
</form>
```

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

**GITHUB LINK :**

<https://github.com/IBM-EPBL/IBM-Project-9625-1659030297>

**DEMO LINK:**

<https://drive.google.com/file/d/1RIrRnSV6BF4srpuj6OugQ8bKBZb-uDAR/view?usp=drivesdk>

**REFERENCES**

1. Antony, L., Azam, S., Ignatious, E., Quadir, R., Beeravolu, A. R., Jonkman, M., & De Boer, F. (2021). A comprehensive unsupervised framework for chronic kidney disease prediction. *IEEE Access*, 9, 126481-126501.
2. Qin, J., Chen, L., Liu, Y., Liu, C., Feng, C. and Chen, B., (2019). A machine learning methodology for diagnosing chronic kidney disease. *IEEE Access*, 8, pp.20991-21002.
3. Chen, G., Ding, C., Li, Y., Hu, X., Li, X., Ren, L., Ding, X., Tian, P. and Xue, W., (2020). Prediction of chronic kidney disease using adaptive hybridized deep convolutional neural network on the internet of medical things platform. *IEEE Access*, 8, pp.100497-100508.
4. Bhaskar, N. and Manikandan, S., (2019). A deep-learning-based system for automated sensing of chronic kidney disease. *IEEE Sensors Letters*, 3(10), pp.1-4.
5. Nishanth, A. and Thiruvaran, T., (2017). Identifying important attributes for early detection of chronic kidney disease. *IEEE reviews in biomedical engineering*, 11, pp.208-216.