

# ***Early Detection of Chronic Kidney Disease using Machine Learning***

***Team ID-PNT2022TMID53897***

## **1.INTRODUCTION**

Chronic kidney disease (CKD) is a worldwide under diagnosed public health problem with increasing incidence and prevalence that has high costs and poor outcomes . The number of patients on renal replacement therapy has doubled every decade since 1980, and prevalence of CKD in the early stages has also markedly increased . Unfortunately, CKD in its earliest stages is usually an asymptomatic condition which progresses to its end stage over a period of several years and is diagnosed late in its course. Therefore, strategies to reduce the incidence of end-stage renal disease require effective methods of screening early in the disease process. Intervention in the early stages of CKD seems to be more effective to prevent or delay the progression of CKD. Moreover, reduced kidney function was found to be an independent risk factor for cardiovascular events and/or all-cause mortality . Therefore, early detection of CKD and treatment of its complications seems to be important to improve outcome in cardiovascular diseases. However, the screening methods most suitable to identify for further diagnosis individuals with CKD remain to be settled. Albumin was used as a screening tool but by itself is not sufficient. Garg et al found that albumin and renal insufficiency measured on a single occasion identified different segments of the population. More than one third of people with an EGFR below 30 ml/min/1.73 m<sup>2</sup> demonstrated no albumin. Moreover, one third of diabetic patients and almost two thirds of non-diabetic hypertensive patients demonstrated no albumin. There is no simple correlation between the progress of glomerular disease and kidney function . Both glomerular and tubule interstitial damage can mediate impairment of renal function. Furthermore, inulin clearance is better correlated with tubule interstitial damage than glomerular disease . In many forms of renal diseases due to primary glomerular lesions, there is a significant inverse correlation between the extent of tubule interstitial damage and the GFR . Therefore, supplementing screening methods such as albumin and EGFR with other diagnostic tools seems to be necessary for early detection of kidney disease. It is worth mentioning at this point that both albumin and decrease in eGFR were used as 2 basic markers for the new classification of CKD proposed by the Kidney Disease Outcomes Quality

Initiative of the National Kidney Foundation and introduced widely after slight modification by the KDIGO (Kidney Disease: Improving Global Outcomes) international community . Damage to the interstitial compartment, especially to more distal segments of tubules, is accompanied by an inability to maximally concentrate the urine, and therefore may result in nocturia. We believe that asking simple questions about nocturia could help distinguish the missing segment of the population with CKD who demonstrate no albumin (after exclusion of the main reasons for nocturia, such as urinary tract infection, poorly controlled diabetes, severe heart failure and evening dosing of diuretics). The definition of nocturia was accepted as the necessity to urinate more than once during the night, which disturbs sleep and which had lasted at least 3 months.

## **1.1 Project Overview**

Chronic kidney disease (CKD) is one of the most critical health problems due to its increasing prevalence. In this , we aim to test the ability of machine learning algorithms for the prediction of chronic kidney disease using the smallest subset of features. Several statistical tests have been done to remove redundant features such as the ANOVA test, the Pearson's correlation, and the Cramer's V test. Logistic regression, support vector machines, random forest, and gradient boosting algorithms have been trained and tested using 10-fold cross-validation. We achieve an accuracy of 99.1 according to F1- measure from Gradient Boosting classifier and by using Random forest method. Also, we found that hemoglobin has higher importance for both random forest and Gradient boosting in detecting CKD. Finally, our results are among the highest compared to previous studies but with less number of features reached so far. Hence, we can detect CKD at only \$26.65 by performing three simple tests.

## **1.2 Purpose**

This project helps everyone to easily detect the Kidney disease which is present in our body . It is very user friendly

## 2. LITERATURE SURVEY

S.No	PAPER TITLE	TECHNOLOGIES USED	DESCRIPTION
1	Chronic Kidney Disease Prediction and Recommendation of Suitable Diet plan by using Machine Learning	Machine Learning Algorithms ,MDRD equation	The proposed system which detects chronic kidney disease using machine learning defines 3 zones(Safe zone,Caution zone,Danger zone) on the basis of blood potassium level.
2	Performance Analysis of Machine Learning Classifier for Predicting Chronic Kidney Disease	Regression and classification, decision tree classifier, random forest	This proposed system detects CKD- Chronic Kidney Disease using machine learning; they have attained an accuracy of 100% for decision tree classifier, 95.12% for random forest and 98.82% in logistic regression.
3	Prediction of chronic kidney disease (CKD) using Data Science	Support Vector Machine, Random Forest, XGBoost, Logistic Regression, Neural networks, Naive Bayes Classifier.	This research work is primarily concentrated on finding the best suitable classification algorithm which can be used for the diagnosis of CKD based on the classification report and performance factors.

4	Chronic kidney disease Diagnosis using Multilayer perceptron classifier	Multilayer Perceptron Classifier	The Experimental results show that the proposed model can perform classification with the testing accuracy of 92.5% surpassing the scores achieved by SVM and naive bayes classifier.
5	A Neural Network based Model for Predicting Chronic Kidney Diseases	Artificial Neural Network algorithms	The 14 different properties are analyzed and linked to chronic kidney disorder victims and foretold accuracy for a machine learning algorithm named Artificial Neural Network. After analyzing the outcomes, it is recognized that the algorithm gives correctness of 96.
6	Early Diagnosis of Chronic Kidney Disease Using Machine Learning Algorithms with Least Parameters by RFE and Feature Importance Techniques	Linear, Logistic, Decision tree, CART, and Random forest classifier	The primary goal of this research project is to enhance the diagnostic precision by assessing the optimum feature selection and developing a prediction model using machine learning methods. By using different classifier methods, the model achieved a diagnosis accuracy of 0.925.
7	A Machine Learning Methodology for Diagnosing Chronic Kidney Disease	Logistic regression, Random forest, Support vector machine, knearest neighbor, Naive Bayes classifier, and Feed Forward Neural Network	A machine learning approach for diagnosing CKD was suggested in this study. An integrated model that combines logistic regression and random forest with the aid

			of perceptron was utilized and it was able to attain an average accuracy of 99.83% after ten times of simulation.
<b>8</b>	Detection of Chronic Kidney Disease Using Machine Learning Algorithms with Least Number of Predictors	Logistic regression, SVM, Random forest, and Gradient boosting	The link between variables has been researched in order to decrease the number of features and eliminate redundancy. Tenfold cross-validation has been used to train, test, and validate the classifiers.
<b>9</b>	Intelligent systems on the cloud for the early detection of chronic kidney disease	Back-propagation networks, Generalized Feed Forward Neural Networks, and Modular Neural Networks	Utilizing Google Application Engine, the system created in accordance with the best model is uploaded to the Google cloud platform. The end solution can more effectively give CKD
<b>10</b>	Optimization of Prediction Method of Chronic Kidney Disease Using Machine Learning Algorithm.	Support Vector Machine, AdaBoost, Linear Discriminant Analysis, and Gradient Boosting	These algorithms are used using a dataset from the UCI machine learning repository that is available online. Gradient Boosting (GB) Classifiers produce results with a predictably high accuracy of roughly 99.80%. Based on these benchmarks, the most effective and optimized algorithms for the requested job can be chosen

## 2.1 Existing problem

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

SNO	LITERATURE PAPER	AUTHOR	PROPOSED METHOD	ACCURACY	YEAR
1	Computer-Aided Diagnosis of Chronic Kidney Disease in Developing Countries: A Comparative Analysis of Machine Learning Techniques	Andressa C. M. Da S. Queiroz, Alvaro Sobrinho, Leandro Dias Da Silva, Evandro De Barros Costa, Maria Eliete Pinheiro, Angelo	J48 decision tree is a suitable machine learning technique for such screening in developing countries, due to the easy interpretation of its classification results	95.00%	2020

Perkusich



2	Chronic Kidney Disease Prediction using Machine Learning Models	S.Revathy, B.Bharathi, P.Jeyanthi, M.Ramesh	Decision tree, Random Forest and Support Vector Machine learning models are constructed to carry out the diagnosis of CKD	99.16%	2019
3	Preemptive Diagnosis of Chronic Kidney Disease Using	Reem A. Alassaf, Khawla A. Alsulaim, Noura Y. Alroomi,	ANN, SVM, Naïve Bayes along with k-NN comparison approach	ANN, SVM, Naïve Bayes - 98% k-NN - 93.9%	2018

	Machine	Nouf S.			
	Learning	Alsharif,			
	Techniques	Mishael F.			
		Aljubeir,			
		Sunday O.			
		Olatunji,			
		Alaa Y.			
		Alahmadi,			
		Mohammed			
		Imran, Rahma A.			
		Alzahrani,			
		Nora S.			
		AlturayEIF			

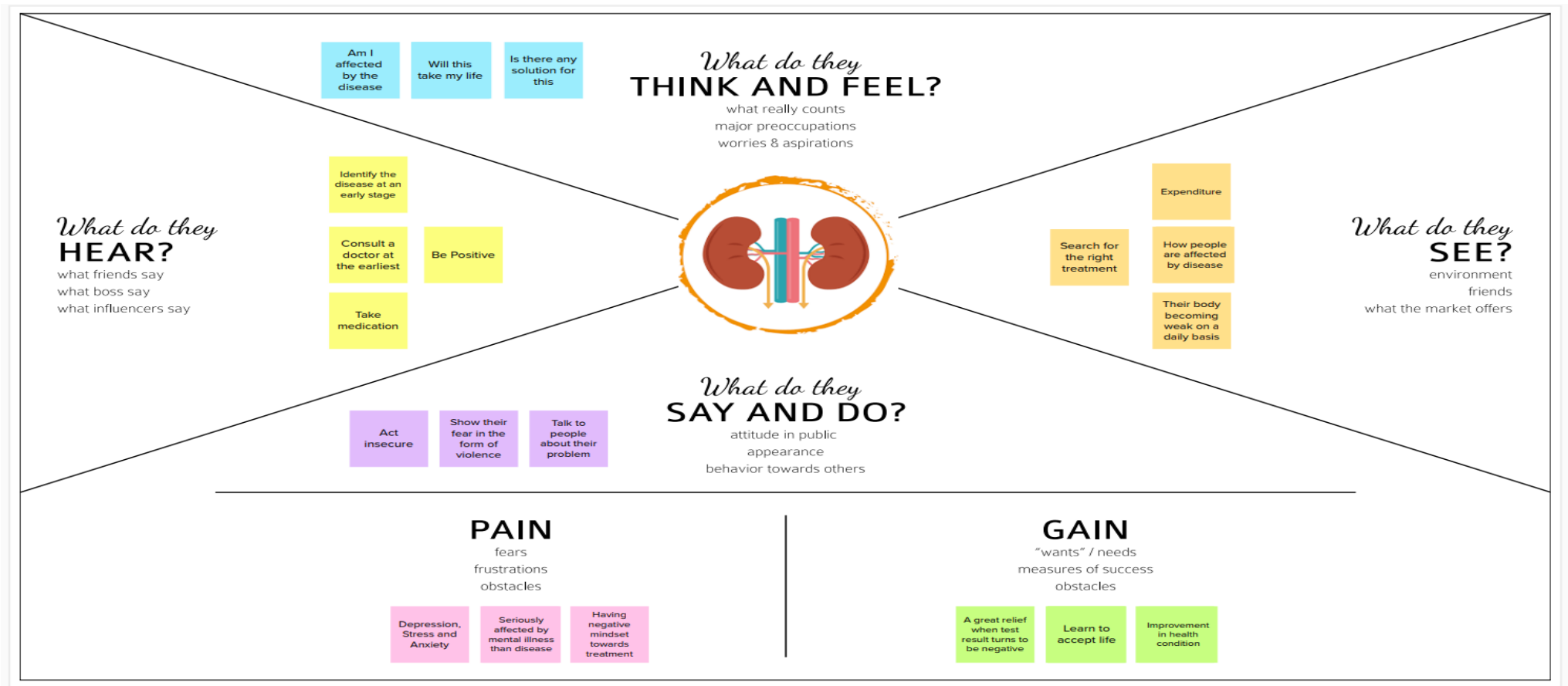
4	Prediction of Chronic Kidney Disease Using Machine Learning Algorithm	Siddheshwar Tekale, Pranjal Shingavi, Sukanya Wandhekar, Ankit Chatorikar	Decision tree algorithms along comparison with SVM	Decision tree – 91.75% SVM-96.75%	2018
5	Neural network and support vector machine for the prediction of chronic kidney disease: A comparative study	Njoud Abdullah Almansour, Hajra Fahim Syed, Nuha Radwan Khayat, Rawan Kanaan Altheeb, Renad Emad Juri, Jamal Alhiyafi, Saleh Alrashed, Sunday O. Olatunji	Comparative analysis was carried out on the two models-ANN and SVM	ANN - 99.75% SVM - 97.75%	2019

## **2.3 Problem Statement Definition**


Non-communicable illnesses are the leading cause of early death, and CKD is the leading non-communicable disease. Chronic Kidney Disease is a major concern for the global health care system. People with CKD must focus on implementing proven, cost-effective therapies to as many people as possible while taking into consideration restricted needs, human and financial resources. Chronic kidney disease (CKD) is now wreaking havoc on society and is spreading at an alarming rate. Various efforts have been undertaken to advance early therapy to prevent the condition from progressing to chronic disease. Recent research suggests that some of the negative outcomes can be avoided with early identification and treatment.

# 3. IDEATION & PROPOSED SOLUTION

## 3.1 Empathy Map Canvas



## 3.2 Ideation & Brainstorming



## Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare  
1 hour to collaborate  
20 people recommended

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

1. **Team gathering**  
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

2. **Set the goal**  
Think about the problem you'll be focusing on solving in the brainstorming session.

3. **Learn how to use the facilitation tools**  
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

**PROBLEM**

Patients who suffer from chronic kidney disease need a way to control the progression to an advanced state with early detection and appropriate treatment.

**5 minutes**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

10 minutes

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

10 minutes

Take turns sharing your ideas while clustering or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

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

20 minutes

### 3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<p>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.</p> <p>A better testing method which could possibly detect CKD in the early stages would be much more useful using machine learning algorithm</p>
2.	Idea / Solution description	<p>The idea of approaching the problem is by creating a suitable machine learning model which involves deep understanding of the data which needs to be collected from real time , handle the missing data and standardizing the data by preprocessing technique which makes it suitable for ml model training and prediction using different approach of model creation depending on the dataset and output</p>
3.	Novelty / Uniqueness	<ul style="list-style-type: none"><li>• Easy to use User interface (UI)</li><li>• accurate accuracy by comparing the performance of different ml model technique</li></ul>

4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> <li>• Greater cost reduction in hospitals for testing</li> <li>• Helps in early diagnosis of the disease</li> <li>• Chances of recovery is higher</li> </ul>
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> <li>• subscription based model with initial trial basis</li> <li>• charges/commission for the actual prediction and recovery of a person</li> </ul>
6.	Scalability of the Solution	<ul style="list-style-type: none"> <li>• The server in which the app is deployed</li> <li>• containing the ml model must be capable of handling concurrent request and handle multiple request</li> <li>• maintaining the ml model by tweaking the parameter which doesn't play vital role in prediction by seeing the next set of dataset.</li> <li>• regular maintenance and changes in model with new features included in it</li> </ul>



### 3.4 Problem Solution fit

Problem-Solution fit canvas 2.0			Early Detection of Chronic Kidney Disease Using Machine Learning		
Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Patients who are facing issues related to kidneys. Elderly people, are more prone to get kidney disease. Diabetic Patients Alcoholic addicted Patients	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> Patients are afraid about risk of using new technology They are limiting themselves as they are not aware of the test accuracies	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Currently in the Medical field, the tests that are performed to detect chronic kidney disease are: 1. Ultra Sound Scan 2. MRI Scan 3. CT Scan	Explore AS, differentiate	
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> Problems related to identifying the chronic kidney disease  Accuracy of patients test results  Time taken to produce test results	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> The root cause of the problem is inaccurate results.  The test takes much time to evaluate the results.	<b>7. BEHAVIOUR</b> <span>BE</span> They take costly Scans because they had no other choice.  They blindly trust the inaccurate test results and become more anxious and sad.	Focus on J&P, tip into BE, understand RC	
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> Their dilemma or confusion of whether they really have chronic kidney disease or not!	<b>10. YOUR SOLUTION</b> <span>SL</span> Predicts Faster and accurately.  Time and Cost of Test is drastically reduced  Helps to take treatment at right time.	<b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>They consider taking tests costing lower from any of the online labs.</b>	Extract online & offline CH of BE	
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> <b>BEFORE:</b> Anxious about their medical condition. <b>AFTER:</b> Determined and able to follow doctor's advice on hat to do next to improve their condition		<b>8.2 OFFLINE</b> They take many tests in offline labs and wait for enormous time to gets results		

## 4.REQUIREMENT ANALYSIS

### 4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	In order to become a new user, you will need to register through a form
FR-2	User Login	Users who already have credentials can log in with those credentials
FR-3	User Requirements	Past records can be stored in a database Create a report to indicate whether or not there is chronic kidney disease present A diagnostic remedy for the symptoms that you are experiencing
FR-4	User Entry	A form to be filled out in order to enter the results of pre-diagnostic tests
FR-5	Business Requirements	Diagnose CKD quickly with a quick blood test
FR-6	User Feedback	The user can submit feedback through a form on the website

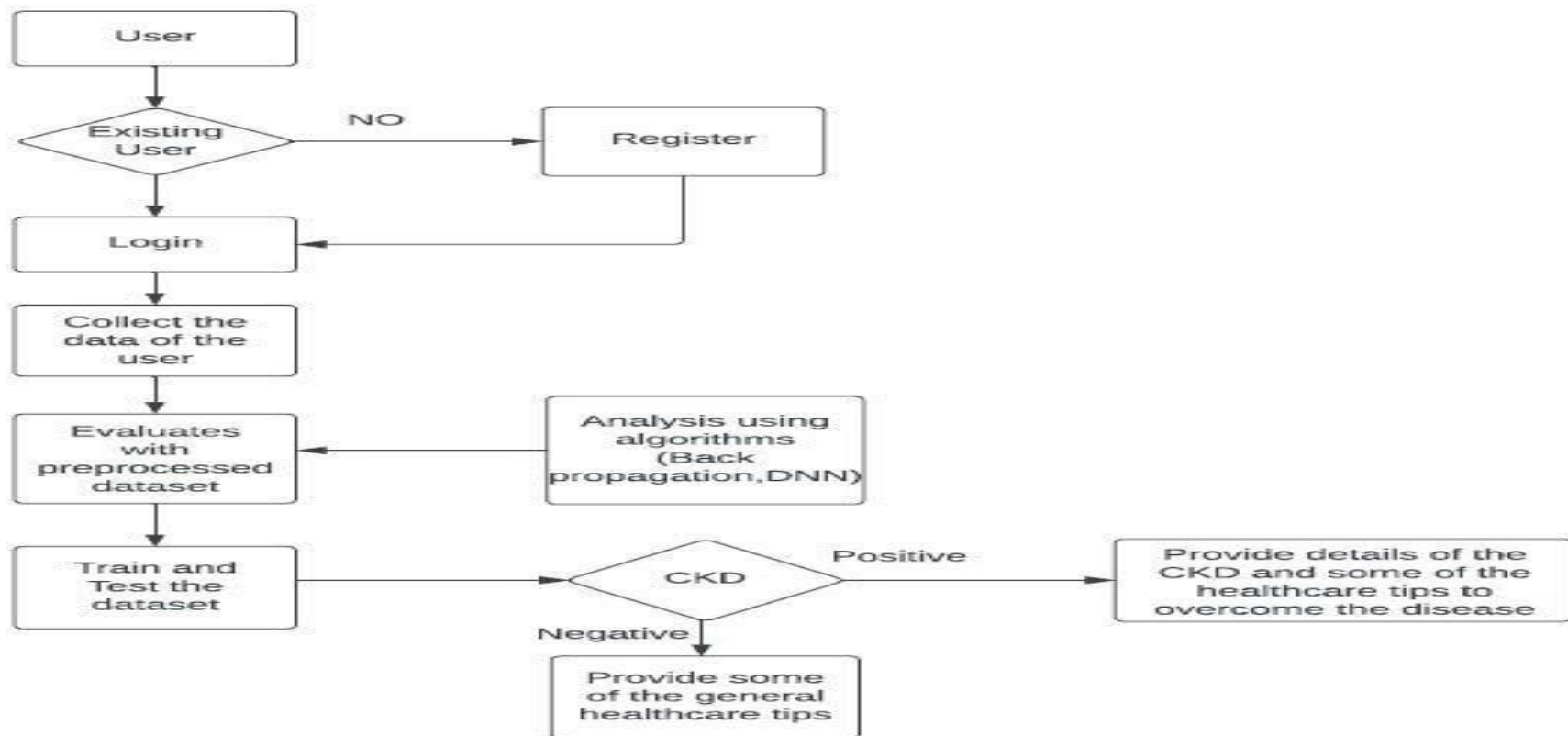
## 4.2 Non-Functional requirements

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

FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	Easy-to-use interface for communication that is user-friendly
NFR-2	<b>Security</b>	Maintain the confidentiality of the details that users share with you
NFR-3	<b>Reliability</b>	A ML model must be able to predict probabilities with sufficient accuracy to provide a reliable diagnosis
NFR-4	<b>Performance</b>	A reduction in the overall time it takes for a diagnosis to be completed
NFR-5	<b>Availability</b>	The service is available to users from various locations at any time
NFR-6	<b>Scalability</b>	A large number of users need to be supported at the same time

## 5. PROJECT DESIGN

**5.1 Data Flow Diagrams** 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 store





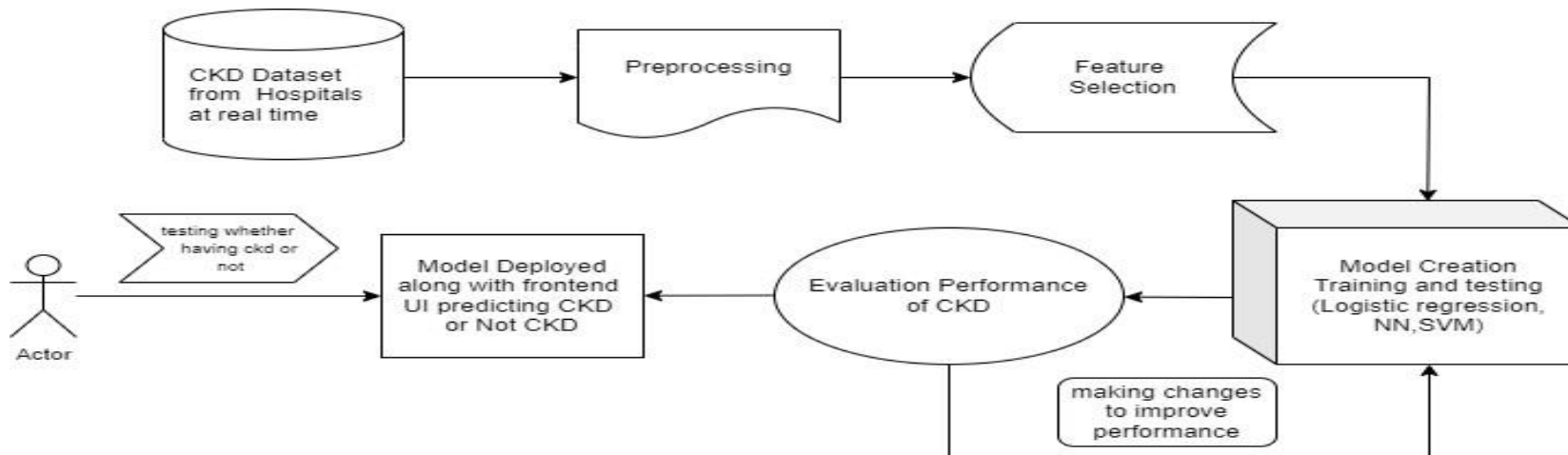
## 5.2 Solution & Technical Architecture

### Solution Architecture

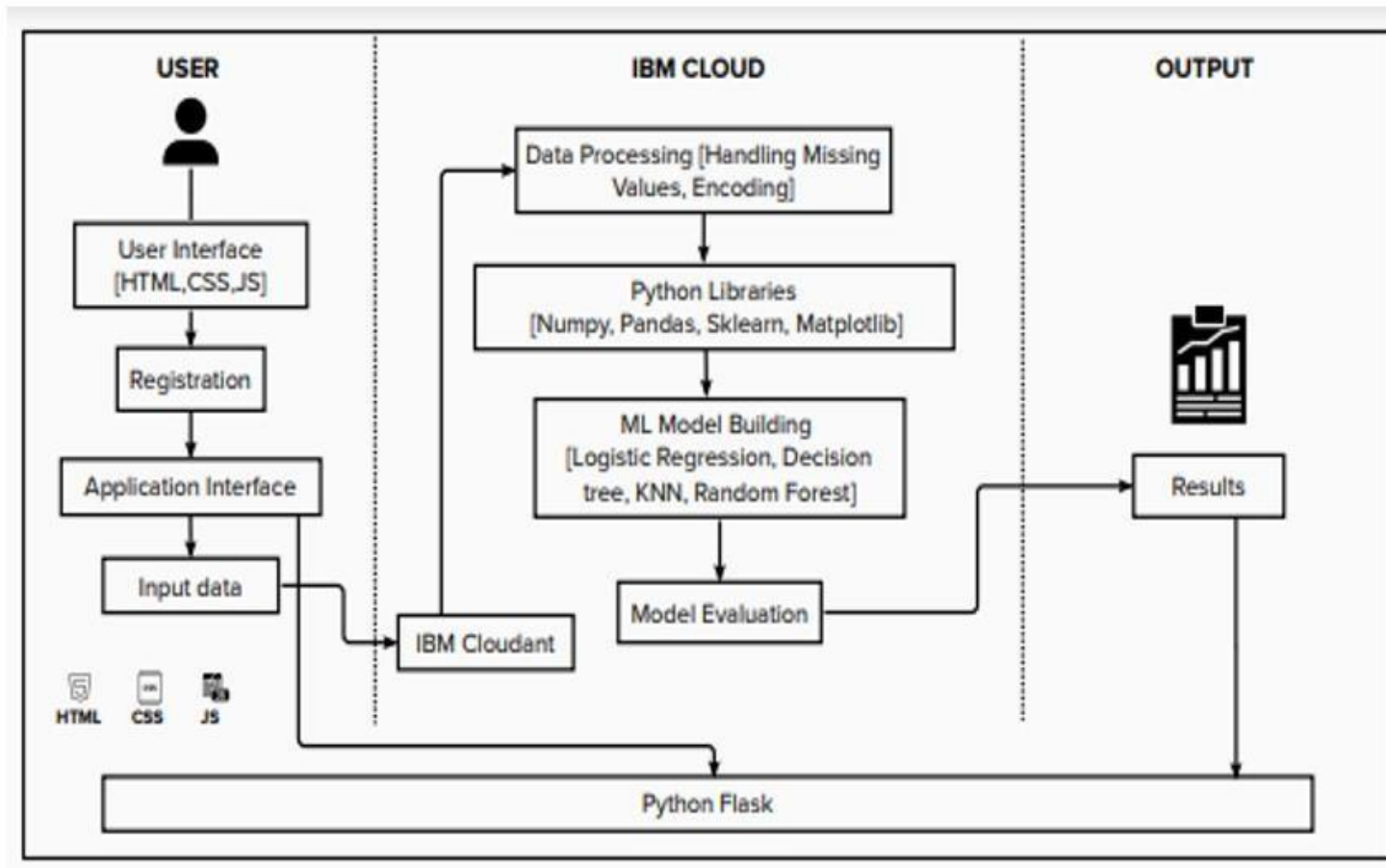
The Solution Architecture consist of the following elements/process for creation of accurate machine learning model they are as follows:-

- Dataset - The need of data is important to analyze the features for prediction
- Preprocessing Techniques -To handle missing data as well as standardizing the data for the model to process
- Feature Selection - Splitting the data into independent and dependent variables and dropping unwanted features using dimensionality reduction or dropping categorical variable with no importance
- Model creation and evaluation - model is created with various ml algorithm aiming for greater accuracy and evaluation is done repeatedly to improve accuracy
- Deploying model - model is deployed for the access of the service over the internet by customer
- User/customer-main actors for accessing the service

### Solution Architecture Diagram:



## Technical Architecture:



## 5.3 User Stories

### User Stories

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

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Verification	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I am authorized user to avail the web service	High	Sprint-1
	Dashboard	USN-4	As a user, I can navigate and interact with the web app to provide inputs for prediction and testing	I am entitled to enter only valid input for prediction	High	Sprint-1
Customer Care Executive	Assist	USN-5	Collecting the issues and reports from the user through various method of communication	The report or issue must be valid and fully verified	High	Sprint-2
Administrator	Manage	USN-6	Management head controlling all the web services as well as assigning task to improve the service	Complete proper working of web service including security aspect	High	Sprint-3



## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

Sprints are the backbone of any good Agile development team. And the better prepared you are before a sprint, the more likely you are to hit your goals. Spring planning helps to refocus attention, minimize surprises, and (hopefully) guarantee better code gets shipped. The main event during agile methodology is the sprint, the stage where ideas turn into innovation and valuable products come to life. On one hand, agile sprints can be highly effective and collaborative. At the same time, they can be chaotic and inefficient if they lack proper planning and guidance. And for this reason, making a sprint schedule is one of the most important things you can do to ensure that your efforts are successful

### 6.2 Sprint Delivery Schedule

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the diagnosis tool using my email and password	7	High	Team Lead Member 1

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

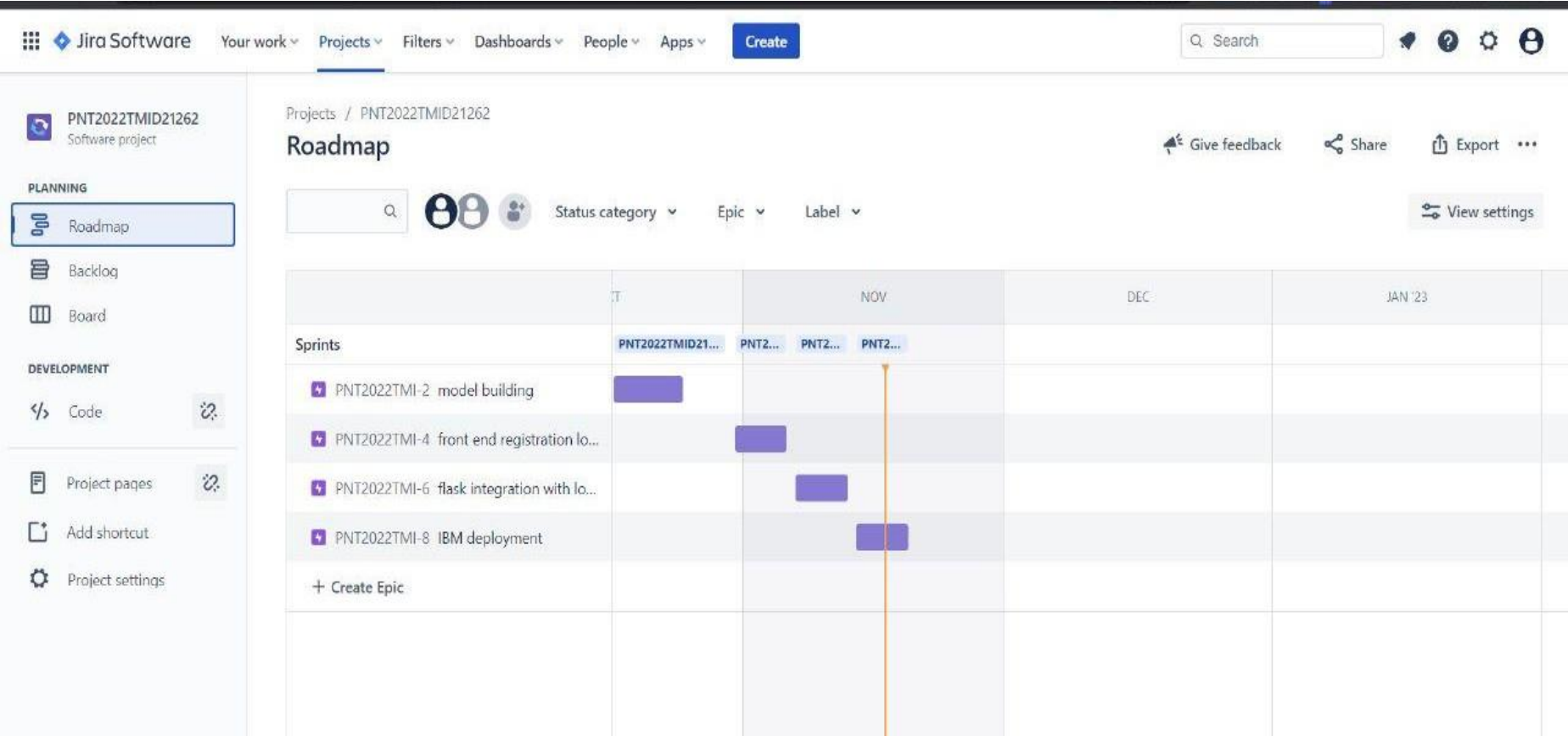
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
--------	-------------------------------	-------------------	-------------------	--------------	----------	--------------

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

**Project Tracker, Velocity & Burndown Chart:**

<b>Sprint</b>	<b>Total Story Points</b>	<b>Duration</b>	<b>Sprint Start Date</b>	<b>Sprint End Date (Planned)</b>	<b>Story Points Completed (as on Planned End Date)</b>	<b>Sprint Release Date (Actual)</b>
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

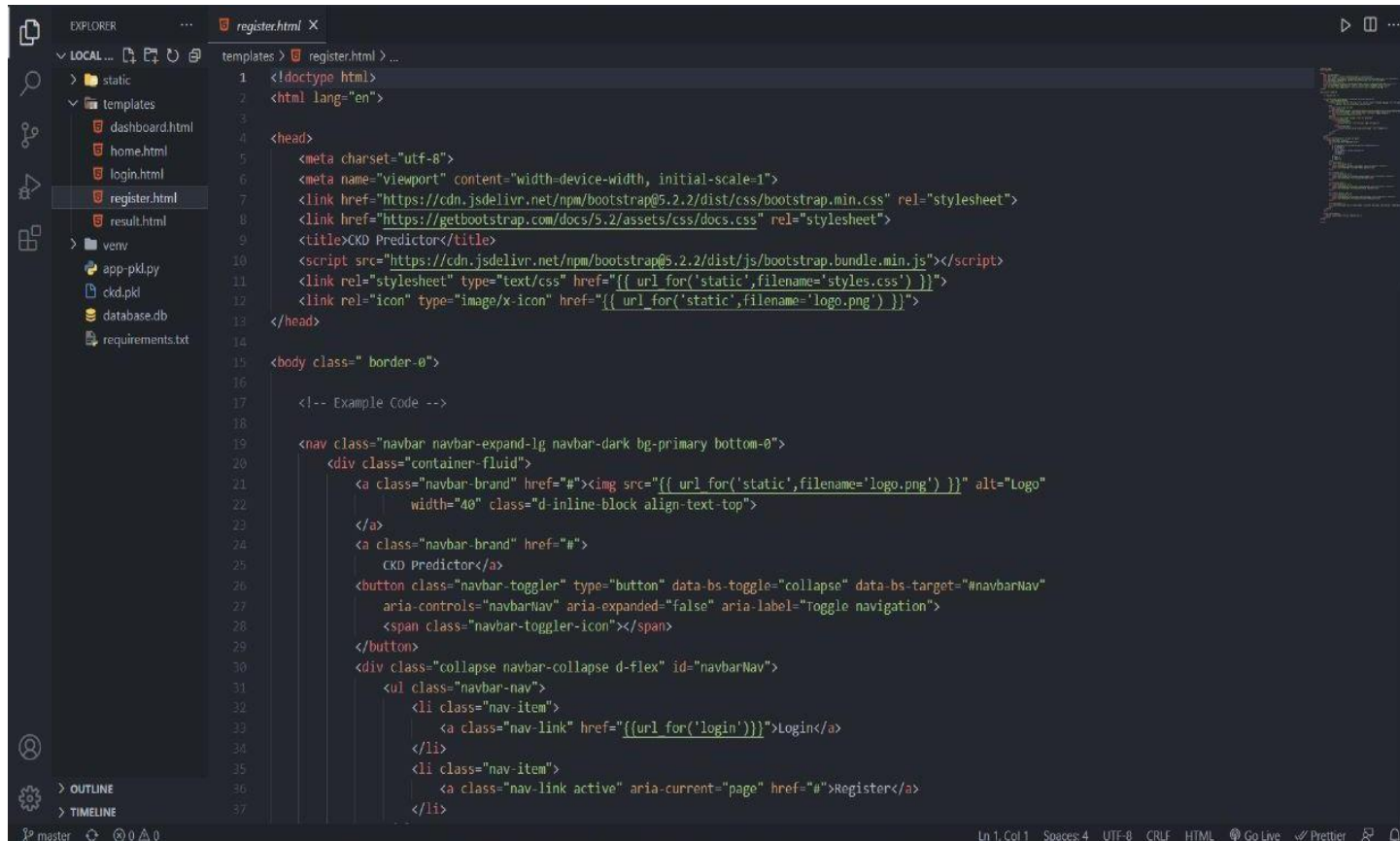
## 6.3 Reports from JIRA



## 7. CODING & SOLUTIONING

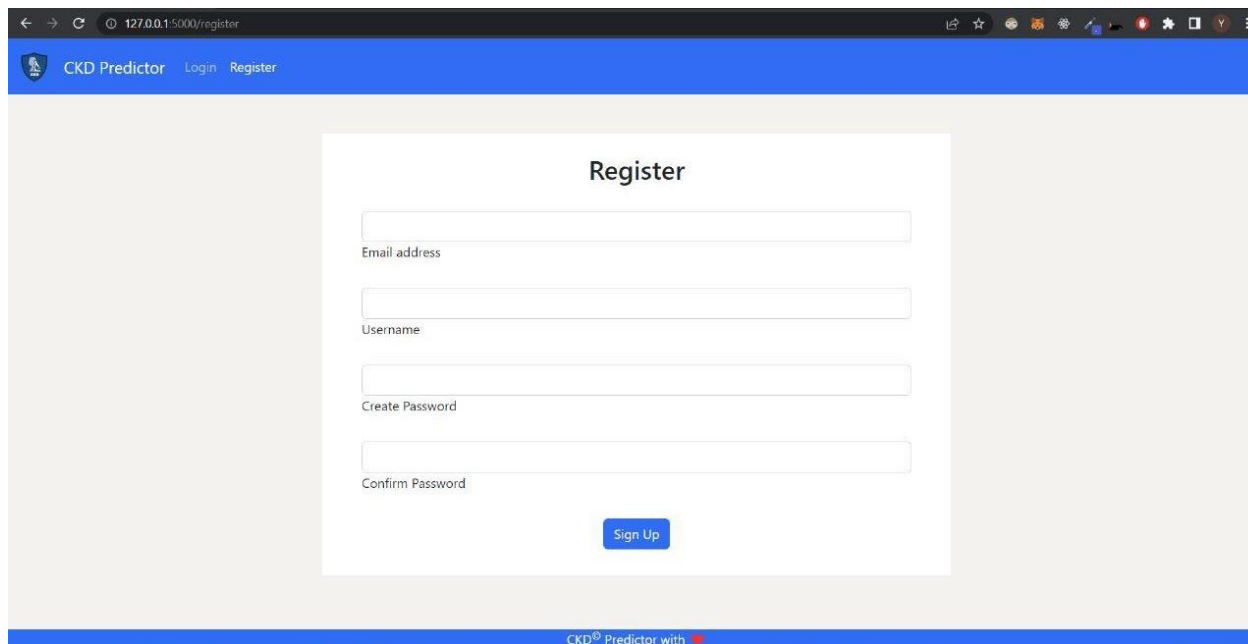
### User Registration and login :

Register.html:



```
1 <!doctype html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
8   <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
9   <title>CKD Predictor</title>
10  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
11  <link rel="stylesheet" type="text/css" href="{{ url_for('static',filename='styles.css') }}">
12  <link rel="icon" type="image/x-icon" href="{{ url_for('static',filename='logo.png') }}">
13 </head>
14
15 <body class=" border-0">
16
17   <!-- Example Code -->
18
19   <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
20     <div class="container-fluid">
21       <a class="navbar-brand" href="#">
23       </a>
24       <a class="navbar-brand" href="#">
25         CKD Predictor</a>
26       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
27         aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
28         <span class="navbar-toggler-icon"></span>
29       </button>
30       <div class="collapse navbar-collapse d-flex id="navbarNav">
31         <ul class="navbar-nav">
32           <li class="nav-item">
33             <a class="nav-link" href="{{ url_for('login') }}">Login</a>
34           </li>
35           <li class="nav-item">
36             <a class="nav-link active" aria-current="page" href="#">Register</a>
37           </li>
38         </ul>
39       </div>
40     </div>
41   </nav>
```





## Login.html

```
EXPLORER
  LOCAL ...
    static
    templates
      dashboard.html
      home.html
      login.html
      register.html
      result.html
    venv
    app-pkl.py
    ckd.pkl
    database.db
    requirements.txt

login.html
1 <!doctype html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
8   <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
9   <title>CKD Predictor</title>
10  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
11  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
12  <link rel="icon" type="image/x-icon" href="{{ url_for('static', filename='logo.png') }}">
13 </head>
14
15 <body class="border-0">
16
17   <!-- Example Code -->
18
19   <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
20     <div class="container-fluid">
21       <a class="navbar-brand" href="#">
23       </a>
24       <a class="navbar-brand" href="#">
25         CKD Predictor</a>
26       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbartnav"
27         aria-controls="navbartnav" aria-expanded="false" aria-label="Toggle navigation">
28         <span class="navbar-toggler-icon"></span>
29       </button>
30       <div class="collapse navbar-collapse d-flex id="navbartnav">
31         <ul class="navbar-nav">
32           <li class="nav-item">
33             <a class="nav-link active" aria-current="page" href="#">Login</a>
34           </li>
35           <li class="nav-item">
36             <a class="nav-link" href="{{ url_for('register') }}">Register</a>
37           </li>
38         </ul>
39       </div>
40     </div>
41   </nav>
```



```
36 <a class="nav-link" href="{{url_for('register')}}">Register</a>
37 </li>
38 </ul>
39 </div>
40 </div>
41 </nav>
42 <div class="w-50 mx-auto mt-5 bg-white px-5 py-4">
43 <form method="post" action="">
44 <h2 class="text-center">Login</h2><br>
45
46 <!-- Username input -->
47 <div class="form-outline mb-4">
48 <input type="username" id="username" name="username" class="form-control" required />
49 <label class="form-label" for="username">Username</label>
50 </div>
51
52 <!-- Password input -->
53 <div class="form-outline mb-4">
54 <input type="password" id="password" name="password" class="form-control" required />
55 <label class="form-label" for="password">Password</label>
56 <div>
57 <!-- with messages = get_flashed_messages(with_categories=true) %>
58 <!-- if messages %>
59 <p class="flash">
60 <!-- for category, message in messages %>
61 <!-- message %>
62 <!-- endfor %>
63 </p>
64 <!-- endif %>
65 <!-- endwith %>
66 </div>
67 </div>
68
69 <!-- Submit button -->
70 <div class="text-center"><button type="submit" class="btn btn-primary btn-block mb-2">Sign in</button>
71 </div>
72 <div class="text-center mt-3">Not yet registered? <a href="{{url_for('register')}}">Register</a></div>
```

```
73 </form>
74 </div>
75
76 <footer class="footer"><small><sup>©</sup> Predictor with ❤</small></footer>
77 </body>
78
79 </html>
```

← → ↻ 127.0.0.1:5000/login

CKD Predictor Login Register

# Login

yash33

Username

.....

Password

Sign in

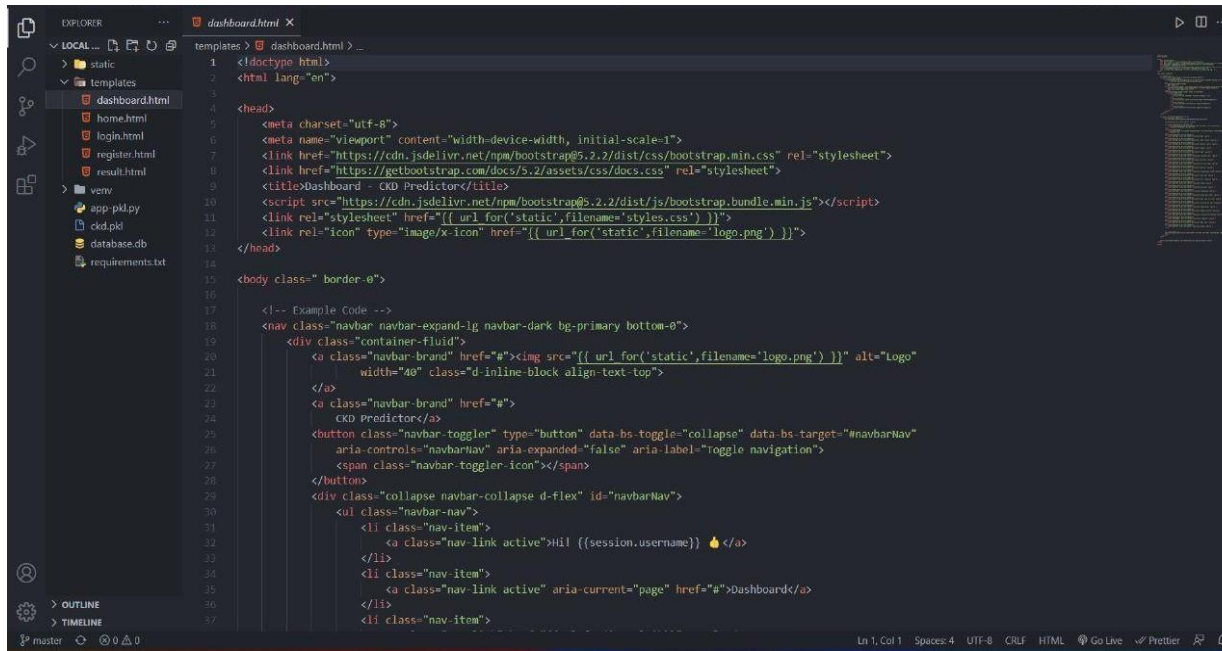
Not yet registered? [Register](#)

CKD® Predictor with ❤️

# Dashboard and Result

## Dashboard.html

2.



```
1 <!doctype html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
8   <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
9   <title>Dashboard - CKD Predictor</title>
10  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
11  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
12  <link rel="icon" type="image/x-icon" href="{{ url_for('static', filename='logo.png') }}">
13 </head>
14
15 <body class="border-0">
16
17   <!-- Example Code -->
18   <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
19     <div class="container-fluid">
20       <a class="navbar-brand" href="#">
22       </a>
23       <a class="navbar-brand" href="#">
24         CKD Predictor</a>
25       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
26         aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
27         <span class="navbar-toggler-icon"></span>
28       </button>
29       <div class="collapse navbar-collapse d-flex id="navbarNav">
30         <ul class="navbar-nav">
31           <li class="nav-item">
32             <a class="nav-link active">Hi! {{ session.username }} 🙌</a>
33           </li>
34           <li class="nav-item">
35             <a class="nav-link active" aria-current="page" href="#">Dashboard</a>
36           </li>
37           <li class="nav-item">
```

3.

```
75 <a class="nav-link active" aria-current="page" href="#">Dashboard</a>
76 </li>
77 <li class="nav-item">
78 <a class="nav-link" href="{{url_for('result')}}">Result</a>
79 </li>
80 <li class="nav-item">
81 <a class="nav-link" href="{{url_for('logout')}}">Logout</a>
82 </li>
83 </ul>
84 </div>
85 </nav>
86
87 <div class="w-75 mx-auto bg-white px-5 py-4">
88 <form method="post" action="/">
89 <h2 class="text-center">Chronic Kidney Disease Predictions</h2><br>
90 <h3>Provide your test details below </h3><br>
91
92 <div class="form-outline mb-2 mx-auto w-25">
93 <input type="datetime-local" id="timestamp" name="timestamp" class="form-control"
94 placeholder="timestamp" required /></div>
95 </div>
96 <div class="form-outline">
97 <input type="hidden" id="username" name="username" class="form-control" value="{{session.username}}"
98 required /></div>
99 <div class="form-outline mb-4 col-6 details">
100 <input name="age" class="form-control" placeholder="Age" required />
101 </div>
102 <div class="form-outline mb-4 col-6 details">
103 <input name="bp" class="form-control" placeholder="Blood Pressure" required />
104 </div>
105 <div class="form-outline mb-4 col-6 details">
106 <input name="sg" class="form-control" placeholder="Specific gravity" required />
107 </div>
108 <div class="form-outline mb-4 col-6 details">
109 <input name="al" class="form-control" placeholder="Albumin" required />
110 </div>
111 <div class="form-outline mb-4 col-6 details">
112 <input name="su" class="form-control" placeholder="sugar" required />
113 </div>
114 <div class="form-outline mb-4 col-6 details">
115 <input name="rbc" class="form-control" placeholder="red blood cells" required />
116 </div>
117 <div class="form-outline mb-4 col-6 details">
118 <input name="pc" class="form-control" placeholder="pus cells" required />
119 </div>
120 <div class="form-outline mb-4 col-6 details">
121 <input name="pcc" class="form-control" placeholder="pus cell clumps" required />
122 </div>
123 <div class="form-outline mb-4 col-6 details">
124 <input name="ba" class="form-control" placeholder="bacteria" required />
125 </div>
126 <div class="form-outline mb-4 col-6 details">
127 <input name="bgr" class="form-control" placeholder="blood glucose random" required />
128 </div>
129 <div class="form-outline mb-4 col-6 details">
130 <input name="bu" class="form-control" placeholder="blood urea" required />
131 </div>
132 <div class="form-outline mb-4 col-6 details">
133 <input name="sc" class="form-control" placeholder="serum creatinine" required />
134 </div>
135 <div class="form-outline mb-4 col-6 details">
```

```
62 <div class="form-outline mb-4 col-6 details">
63 <input name="age" class="form-control" placeholder="Age" required />
64 </div>
65 <div class="form-outline mb-4 col-6 details">
66 <input name="bp" class="form-control" placeholder="Blood Pressure" required />
67 </div>
68 <div class="form-outline mb-4 col-6 details">
69 <input name="sg" class="form-control" placeholder="Specific Gravity" required />
70 </div>
71 <div class="form-outline mb-4 col-6 details">
72 <input name="al" class="form-control" placeholder="Albumin" required />
73 </div>
74 <div class="form-outline mb-4 col-6 details">
75 <input name="su" class="form-control" placeholder="sugar" required />
76 </div>
77 <div class="form-outline mb-4 col-6 details">
78 <input name="rbc" class="form-control" placeholder="red blood cells" required />
79 </div>
80 <div class="form-outline mb-4 col-6 details">
81 <input name="pc" class="form-control" placeholder="pus cells" required />
82 </div>
83 <div class="form-outline mb-4 col-6 details">
84 <input name="pcc" class="form-control" placeholder="pus cell clumps" required />
85 </div>
86 <div class="form-outline mb-4 col-6 details">
87 <input name="ba" class="form-control" placeholder="bacteria" required />
88 </div>
89 <div class="form-outline mb-4 col-6 details">
90 <input name="bgr" class="form-control" placeholder="blood glucose random" required />
91 </div>
92 <div class="form-outline mb-4 col-6 details">
93 <input name="bu" class="form-control" placeholder="blood urea" required />
94 </div>
95 <div class="form-outline mb-4 col-6 details">
96 <input name="sc" class="form-control" placeholder="serum creatinine" required />
97 </div>
98 <div class="form-outline mb-4 col-6 details">
```

The image shows a VS Code editor window with a file explorer on the left and a code editor in the center. The file explorer shows a project structure with a 'templates' folder containing 'dashboard.html', 'home.html', 'login.html', 'register.html', and 'result.html'. The 'dashboard.html' file is selected and its content is displayed in the code editor. The code is an HTML template for a CKD predictor dashboard, featuring two form outlines for 'pedal edema' and 'anemia', a submit button, and a footer.

```
128 <div class="form-outline mb-4 col-6 details">
129   <input name="pe" class="form-control" placeholder="pedal edema" required />
130 </div>
131 <div class="form-outline mb-4 col-6 details">
132   <input name="ane" class="form-control" placeholder="anemia" required />
133 </div>
134
135
136
137 <!-- Submit button -->
138 <div class="text-center"><button type="submit" class="btn btn-primary btn-block mb-4 ">Submit</button>
139 </div>
140 </form>
141
142
143 <footer class="footer">CKD<sup class="footerc">©</sup> Predictor with ❤️</footer>
144 </body>
145
146 </html>
```

The status bar at the bottom indicates the current file is 'Ln 146, Col 8', the encoding is 'UTF-8', and the language is 'HTML'. The editor is using the 'Go Live' extension and the 'Prettier' formatter.

Dashboard - CKD Predictor

127.0.0.1:5000/dashboard

CKD PredictorHi! yash33DashboardResultLogout

## Chronic Kidney Disease Prediction

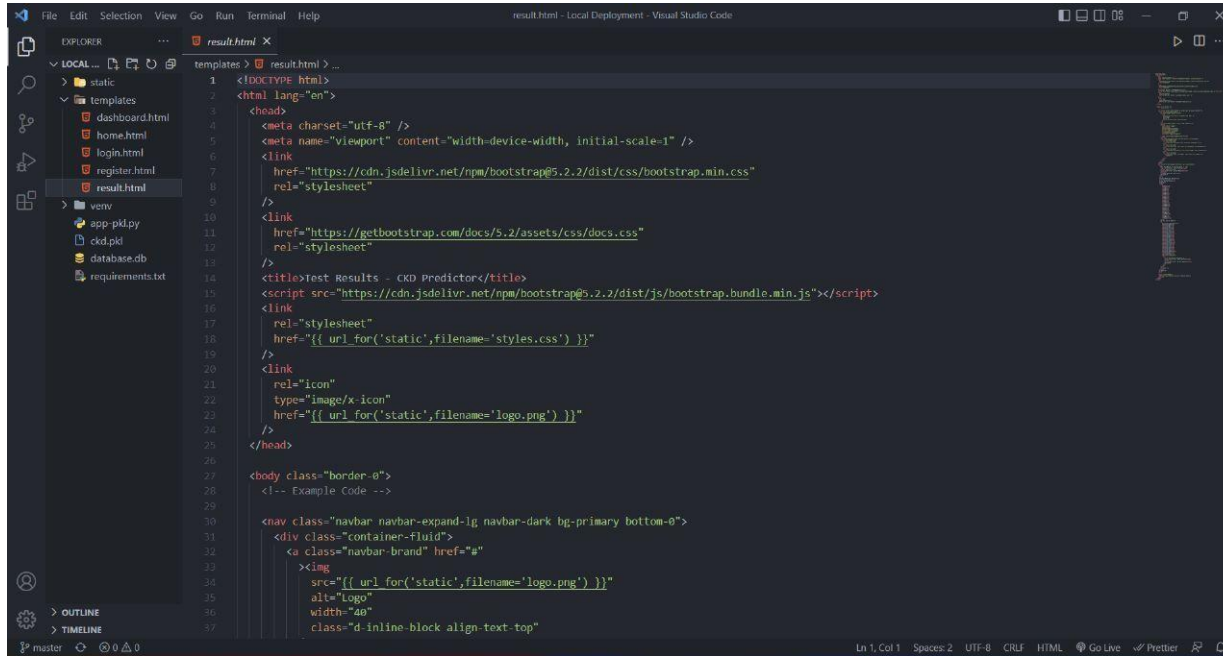
Provide your test details below 🗨️ :

dd-mm-yyyy --:--

Age	Blood Pressure
Specific Gravity	Albumin
Sugar	Red Blood Cells
Pus Cells	Pus Cell Clumps
Bacteria	Blood Glucose Random
Blood Urea	Serum Creatinine
Sodium	Potassium

CKD<sup>®</sup> Predictor with ❤️

## Result.html



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1" />
6   <link
7     href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
8     rel="stylesheet"
9   />
10  <link
11    href="https://getbootstrap.com/docs/5.2/assets/css/docs.css"
12    rel="stylesheet"
13  />
14  <title>Test Results - CKD Predictor</title>
15  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
16  <link
17    rel="stylesheet"
18    href="{{ url_for('static',filename='styles.css') }}"
19  />
20  <link
21    rel="icon"
22    type="image/x-icon"
23    href="{{ url_for('static',filename='logo.png') }}"
24  />
25 </head>
26
27 <body class="border-0">
28   <!-- Example Code -->
29
30   <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
31     <div class="container-fluid">
32       <a class="navbar-brand" href="#"
33         >
```

```
168 templates > result.html > html > body.border-0
169 width="40"
170 class="d-inline-block align-text-top"
171 />
172 </a>
173 <a class="navbar-brand" href="#"> CKD Predictor</a>
174 <button
175   class="navbar-toggler"
176   type="button"
177   data-bs-toggle="collapse"
178   data-bs-target="#navbarNav"
179   aria-controls="navbarNav"
180   aria-expanded="false"
181   aria-label="toggle navigation"
182 >
183   <span class="navbar-toggler-icon"></span>
184 </button>
185 <div class="collapse navbar-collapse d-flex id="navbarNav">
186   <ul class="navbar-nav">
187     <li class="nav-item">
188       <a class="nav-link active">HI! {{session.username}} 🍌 </a>
189     </li>
190     <li class="nav-item">
191       <a class="nav-link" href="{{url_for('dashboard')}}">Dashboard</a>
192     </li>
193     <li class="nav-item">
194       <a class="nav-link active" aria-current="page" href="#">Result</a>
195     </li>
196     <li class="nav-item">
197       <a class="nav-link id="logout" href="{{url_for('logout')}}">
198         >Logout</a>
199     </li>
200   </ul>
201 </div>
202 </nav>
203 <div class="mx-auto bg-white mt-5 px-1 py-4 text-center">
```

```
153 templates > result.html > html > body.border-0 > nav.navbar.navbar-expand-lg.navbar-dark.bg-primary.bottom-0
154 </nav>
155 <div class="mx-auto bg-white mt-5 px-1 py-4 text-center">
156   <div>
157     Your Test Result is: {% if predict == 1 %}
158     <span style="color: red">Positive</span>
159     {% elif predict == 0 %}
160     <span style="color: green">Negative</span>
161     {% else %}
162     <span>Not Available Now!</span>
163     {% endif %}
164   </div>
165   <br />
166   <div>Your Past Test Results</div>
167   {% if table.size == 0 %}
168   <div>No Records found</div>
169   {% else %}
170   <table>
171     <tr>
172       <th>Date</th>
173       <th>Age</th>
174       <th>BP</th>
175       <th>Sex</th>
176       <th>AL</th>
177       <th>SMC</th>
178       <th>BIC</th>
179       <th>PC</th>
180       <th>PCC</th>
181       <th>BAC</th>
182       <th>BGR</th>
183       <th>BUC</th>
184       <th>SCC</th>
185       <th>SOD</th>
186       <th>POT</th>
187       <th>HbA1c</th>
188       <th>PCV</th>
189       <th>WBC</th>
190       <th>HbA1c</th>
```



File Edit Selection View Go Run Terminal Help result.html - Local Deployment - Visual Studio Code

EXPLORER

LOCAL DEPLOYMENT

- static
- templates
  - dashboard.html
  - home.html
  - login.html
  - register.html
  - result.html
- venv
- app-pkl.py
- cdk.pd
- database.db
- requirements.txt

templates > result.html X

```
105 <th>PCV</th>
106 <th>WC</th>
107 <th>HC</th>
108 <th>HMC</th>
109 <th>MC</th>
110 <th>CAC</th>
111 <th>APPET</th>
112 <th>PE</th>
113 <th>AME</th>
114 <th>Result</th>
115 </tr>
116 {% for row in table %}
117 <tr>
118 <td>{{row.timestamp}}</td>
119 <td>{{row.age}}</td>
120 <td>{{row.bp}}</td>
121 <td>{{row.sg}}</td>
122 <td>{{row.al}}</td>
123 <td>{{row.su}}</td>
124 <td>{{row.rbc}}</td>
125 <td>{{row.pc}}</td>
126 <td>{{row.pcc}}</td>
127 <td>{{row.ba}}</td>
128 <td>{{row.bgr}}</td>
129 <td>{{row.bu}}</td>
130 <td>{{row.sc}}</td>
131 <td>{{row.sod}}</td>
132 <td>{{row.pot}}</td>
133 <td>{{row.hemo}}</td>
134 <td>{{row.pcv}}</td>
135 <td>{{row.wc}}</td>
136 <td>{{row.rc}}</td>
137 <td>{{row.htn}}</td>
138 <td>{{row.dm}}</td>
139 <td>{{row.cad}}</td>
140 <td>{{row.appet}}</td>
141 <td>{{row.pe}}</td>
```

Ln 94, Col 22 Spaces: 2 UTF-8 CRLF HTML Go Live Prettier

result.html - Local Deployment - Visual Studio Code

EXPLORER

- LOCAL DEPLOYMENT
  - static
  - templates
    - dashboard.html
    - home.html
    - login.html
    - register.html
    - result.html
  - verv
    - app.pkl.py
    - ckd.pkl
    - database.db
    - requirements.txt

templates > result.html > ...

```
131 <td>{{row.sod}}</td>
132 <td>{{row.pot}}</td>
133 <td>{{row.hemo}}</td>
134 <td>{{row.pcv}}</td>
135 <td>{{row.wc}}</td>
136 <td>{{row.rc}}</td>
137 <td>{{row.htn}}</td>
138 <td>{{row.dm}}</td>
139 <td>{{row.cad}}</td>
140 <td>{{row.appet}}</td>
141 <td>{{row.pe}}</td>
142 <td>{{row.anc}}</td>
143 <td class="resultcol">
144 <div>
145     {% if row.result=='positive' %}
146     <span style="color: red">Positive</span>
147     {% else %}
148     <span style="color: green">Negative</span>
149     {% endif %}
150 </div>
151 </td>
152 </tr>
153 {% endfor -%}
154 </table>
155 {% endif %}
156 </div>
157
158 <footer class="footer">
159 <sup class="footerc"></sup> Predictor with ❤️
160 </footer>
161 </body>
162 </html>
163
```

Ln 162, Col 1 Spaces: 2 UTF-8 CRLF HTML Go Live Prettier

Test Results - CKD Predictor

127.0.0.1:5000/result

CKD Predictor

Hi! yash33

Dashboard

Result

Logout

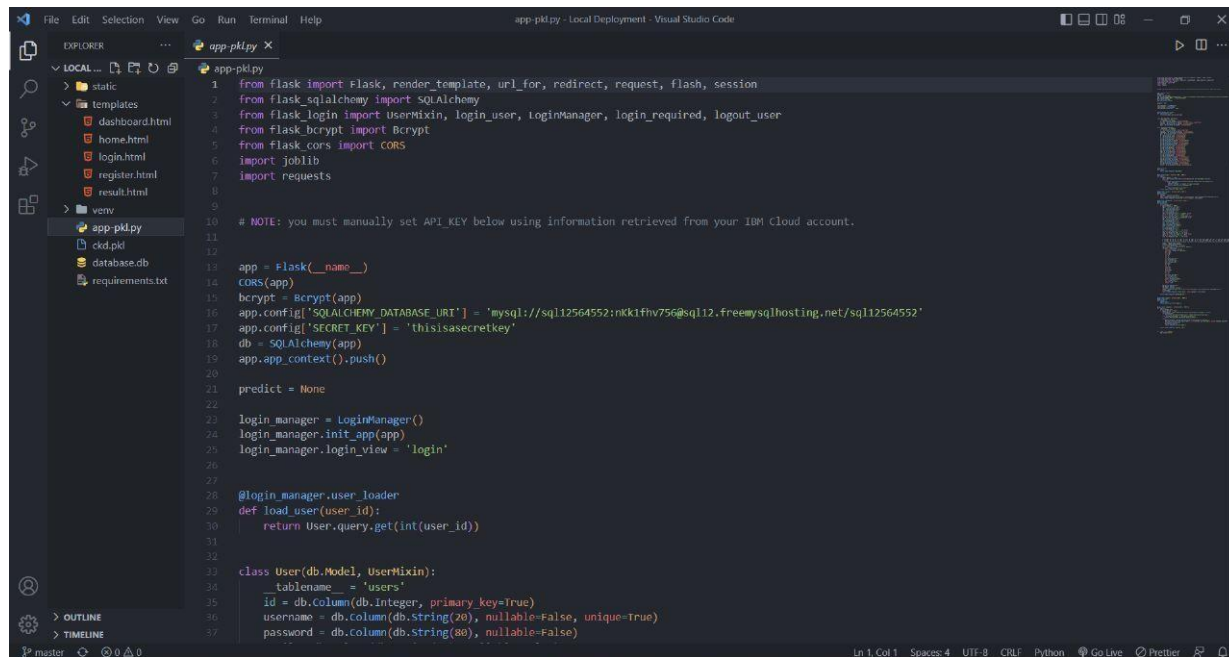
Your Test Result is: Not Available Now!

Your Past Test Results

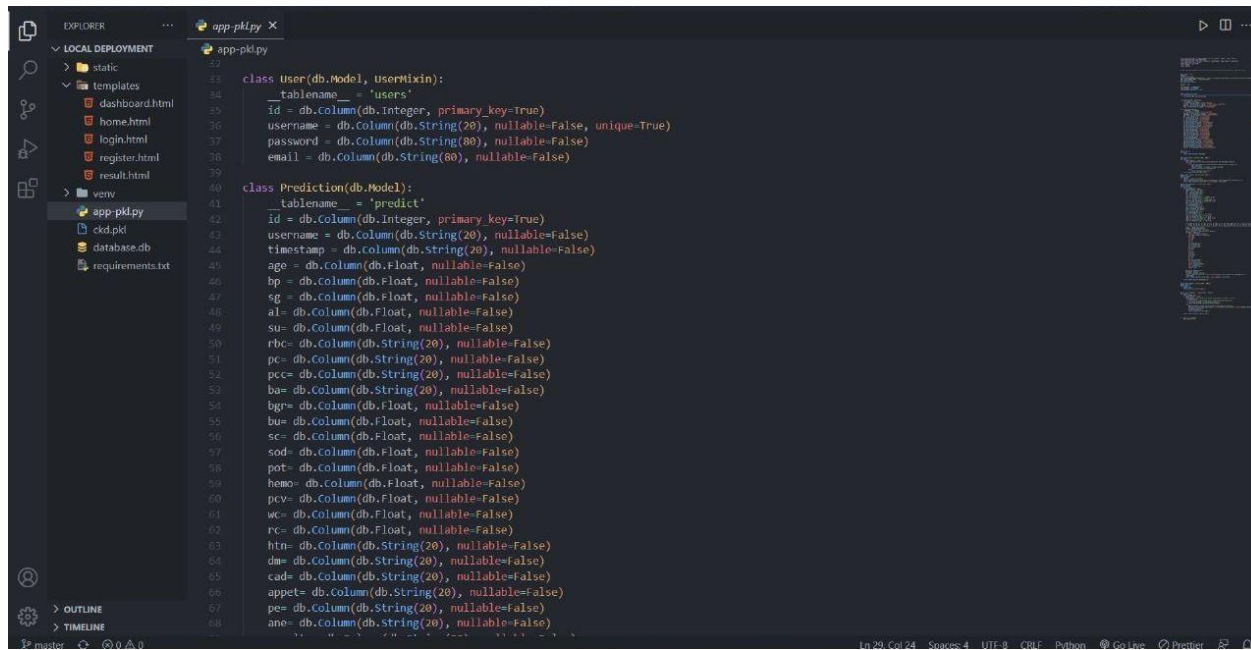
Date	Age	BP	SG	AL	SU	RBC	PC	PCC	BA	BGR	BU	SC	SOD	POT	HEMO	PCV	WC	RC	HTN	DM	CAD	APPET	PE	ANE	Result
2022-11-13T19:53	41.0	80.0	1.02	0.0	0.0	normal	normal	notpresent	notpresent	122.0	25.0	0.8	138.0	5.0	17.1	41.0	9100.0	5.2	no	no	no	good	no	no	Negative

CKD<sup>®</sup> Predictor with

# Flask Integration and Deployment

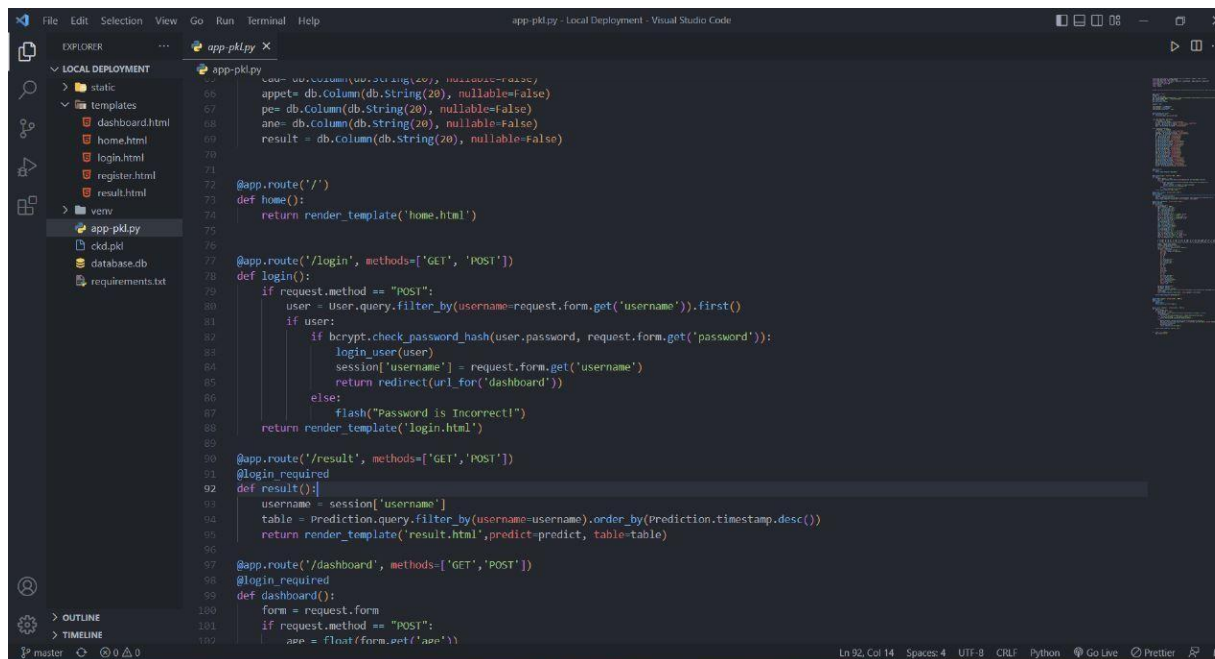


```
1 from flask import Flask, render_template, url_for, redirect, request, flash, session
2 from flask_sqlalchemy import SQLAlchemy
3 from flask_login import UserMixin, login_user, login_manager, login_required, logout_user
4 from flask_bcrypt import Bcrypt
5 from flask_cors import CORS
6 import joblib
7 import requests
8
9
10 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
11
12
13 app = Flask(__name__)
14 CORS(app)
15 bcrypt = Bcrypt(app)
16 app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://sql12564552:nkkl1fv756@sql12.freemysqlhosting.net/sql12564552'
17 app.config['SECRET_KEY'] = 'thisisasecretkey'
18 db = SQLAlchemy(app)
19 app.app_context().push()
20
21 predict = None
22
23 login_manager = LoginManager()
24 login_manager.init_app(app)
25 login_manager.login_view = 'login'
26
27
28 @login_manager.user_loader
29 def load_user(user_id):
30     return User.query.get(int(user_id))
31
32
33 class User(db.Model, UserMixin):
34     tablename = 'users'
35     id = db.Column(db.Integer, primary_key=True)
36     username = db.Column(db.String(20), nullable=False, unique=True)
37     password = db.Column(db.String(80), nullable=False)
```



This screenshot shows the VS Code editor with the `app.py` file open. The Explorer sidebar on the left shows the project structure under 'LOCAL DEPLOYMENT', including `static`, `templates` (with `dashboard.html`, `home.html`, `login.html`, `register.html`, `result.html`), `venv`, `app-pkl.py`, `ckd.pkl`, `database.db`, and `requirements.txt`. The main editor displays the database models:

```
32
33 class User(db.Model, UserMixin):
34     __tablename__ = 'users'
35     id = db.Column(db.Integer, primary_key=True)
36     username = db.Column(db.String(20), nullable=False, unique=True)
37     password = db.Column(db.String(80), nullable=False)
38     email = db.Column(db.String(80), nullable=False)
39
40 class Prediction(db.Model):
41     __tablename__ = 'predict'
42     id = db.Column(db.Integer, primary_key=True)
43     username = db.Column(db.String(20), nullable=False)
44     timestamp = db.Column(db.String(20), nullable=False)
45     age = db.Column(db.Float, nullable=False)
46     bp = db.Column(db.Float, nullable=False)
47     sg = db.Column(db.Float, nullable=False)
48     al = db.Column(db.Float, nullable=False)
49     su = db.Column(db.Float, nullable=False)
50     rbc = db.Column(db.String(20), nullable=False)
51     pc = db.Column(db.String(20), nullable=False)
52     pcc = db.Column(db.String(20), nullable=False)
53     ba = db.Column(db.String(20), nullable=False)
54     bgr = db.Column(db.Float, nullable=False)
55     bu = db.Column(db.Float, nullable=False)
56     sc = db.Column(db.Float, nullable=False)
57     sod = db.Column(db.Float, nullable=False)
58     pot = db.Column(db.Float, nullable=False)
59     hemo = db.Column(db.Float, nullable=False)
60     pcv = db.Column(db.Float, nullable=False)
61     wc = db.Column(db.Float, nullable=False)
62     rc = db.Column(db.Float, nullable=False)
63     htn = db.Column(db.String(20), nullable=False)
64     dm = db.Column(db.String(20), nullable=False)
65     cad = db.Column(db.String(20), nullable=False)
66     appet = db.Column(db.String(20), nullable=False)
67     pe = db.Column(db.String(20), nullable=False)
68     ane = db.Column(db.String(20), nullable=False)
```



This screenshot shows the VS Code editor with the `app.py` file open, displaying the Flask routes. The Explorer sidebar is the same as in the first image. The main editor displays the following code:

```
69     result = db.Column(db.String(20), nullable=False)
70
71 @app.route('/')
72 def home():
73     return render_template('home.html')
74
75 @app.route('/login', methods=['GET', 'POST'])
76 def login():
77     if request.method == "POST":
78         user = User.query.filter_by(username=request.form.get('username')).first()
79         if user:
80             if bcrypt.check_password_hash(user.password, request.form.get('password')):
81                 login_user(user)
82                 session['username'] = request.form.get('username')
83                 return redirect(url_for("dashboard"))
84             else:
85                 flash("Password is Incorrect!")
86         return render_template('login.html')
87
88 @app.route('/result', methods=['GET', 'POST'])
89 @login_required
90 def result():
91     username = session['username']
92     table = Prediction.query.filter_by(username=username).order_by(Prediction.timestamp.desc())
93     return render_template('result.html', predict=predict, table=table)
94
95 @app.route('/dashboard', methods=['GET', 'POST'])
96 @login_required
97 def dashboard():
98     form = request.form
99     if request.method == "POST":
100         age = float(form.get('age'))
```

```
100 form = request.form
101 if request.method == "POST":
102     age = float(form.get('age'))
103     bp = float(form.get('bp'))
104     sg = float(form.get('sg'))
105     al= float(form.get('al'))
106     su= float(form.get('su'))
107     rbc= 0 if form.get('rbc') == 'normal' else 1
108     pc= 0 if form.get('pc') == 'normal' else 1
109     pcc= 0 if form.get('pcc') == 'notpresent' else 1
110     ba= 0 if form.get('ba') == 'notpresent' else 1
111     bgr= float(form.get('bgr'))
112     bu= float(form.get('bu'))
113     sc= float(form.get('sc'))
114     sod= float(form.get('sod'))
115     pot= float(form.get('pot'))
116     hemo= float(form.get('hemo'))
117     pcv= float(form.get('pcv'))
118     wc= float(form.get('wc'))
119     rc= float(form.get('rc'))
120     htn= 0 if form.get('htn') == 'no' else 1
121     dm= 0 if form.get('dm') == 'no' else 1
122     cad= 0 if form.get('cad') == 'no' else 1
123     appet= 0 if form.get('appet') == 'good' else 1
124     pe= 0 if form.get('pe') == 'no' else 1
125     ane= 0 if form.get('ane') == 'no' else 1
126
127     print(age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemo, pcv, wc, rc, htn, dm, cad, appet, pe, ane)
128     X = [[age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemo, pcv, wc, rc, htn, dm, cad, appet, pe, ane]]
129     # *****using the pickle file for prediction*****
130     model = joblib.load('ckd.pkl')
131     predict = model.predict(X)[0]
132     print("final prediction :",predict)
133     res = 'Positive' if predict==1 else 'Negative'
134     new_user = Prediction(
135         username=form.get('username'),
136         timestamp = form.get('timestamp'),
137         age = age,
138         bp = bp,
139         sg = sg,
140         al= al,
141         su= su,
142         rbc= form.get('rbc'),
143         pc= form.get('pc'),
144         pcc= form.get('pcc'),
145         ba= form.get('ba'),
146         bgr= bgr,
147         bu= bu,
148         sc= sc,
149         sod= sod,
150         pot= pot,
151         hemo= hemo,
152         pcv= pcv,
153         wc= wc,
154         rc= rc,
155         htn= form.get('htn'),
156         dm= form.get('dm'),
157         cad= form.get('cad'),
158         appet= form.get('appet'),
159         pe= form.get('pe'),
160         ane= form.get('ane'),
161         result = res
162     )
163     db.session.add(new_user)
164     db.session.commit()
165     username = session['username']
```

```
130 # *****using the pickle file for prediction*****
131 model = joblib.load('ckd.pkl')
132 predict = model.predict(X)[0]
133 print("final prediction :",predict)
134 res = 'Positive' if predict==1 else 'Negative'
135 new_user = Prediction(
136     username=form.get('username'),
137     timestamp = form.get('timestamp'),
138     age = age,
139     bp = bp,
140     sg = sg,
141     al= al,
142     su= su,
143     rbc= form.get('rbc'),
144     pc= form.get('pc'),
145     pcc= form.get('pcc'),
146     ba= form.get('ba'),
147     bgr= bgr,
148     bu= bu,
149     sc= sc,
150     sod= sod,
151     pot= pot,
152     hemo= hemo,
153     pcv= pcv,
154     wc= wc,
155     rc= rc,
156     htn= form.get('htn'),
157     dm= form.get('dm'),
158     cad= form.get('cad'),
159     appet= form.get('appet'),
160     pe= form.get('pe'),
161     ane= form.get('ane'),
162     result = res
163 )
164 db.session.add(new_user)
165 db.session.commit()
166 username = session['username']
```

```
File Edit Selection View Go Run Terminal Help
app.pkl.py - Local Deployment - Visual Studio Code

EXPLORER
LOCAL DEPLOYMENT
  static
  templates
    dashboard.html
    home.html
    login.html
    register.html
    result.html
  views
    app.pkl.py
    ckd.pkl
    database.db
    requirements.txt

app.pkl.py
165 username = session['username']
166 table = Prediction.query.filter_by(username=username).order_by(Prediction.timestamp.desc())
167 print(table)
168 return render_template('result.html', predict=predict, table=table)
169
170 return render_template('dashboard.html')
171
172
173
174 @app.route('/logout', methods=['GET', 'POST'])
175 @login_required
176 def logout():
177     logout_user()
178     return redirect(url_for('login'))
179
180
181 @app.route('/register', methods=['GET', 'POST'])
182 def register():
183     form = request.form
184     if request.method == "POST":
185         existing_user = User.query.filter_by(username=form.get('username')).first()
186         if existing_user:
187             flash('That username already exists! Please choose a different one.')
188         elif form.get('password') != form.get('cpassword'):
189             flash('The password confirmation does not match!')
190         else:
191             hashed_password = bcrypt.generate_password_hash(form.get('password'))
192             new_user = User(email=form.get('email'), username=form.get('username'), password=hashed_password)
193             db.session.add(new_user)
194             db.session.commit()
195             return redirect(url_for('login'))
196
197     return render_template('register.html')
198
199
200 if __name__ == "__main__":
201     app.run(debug=True)
```

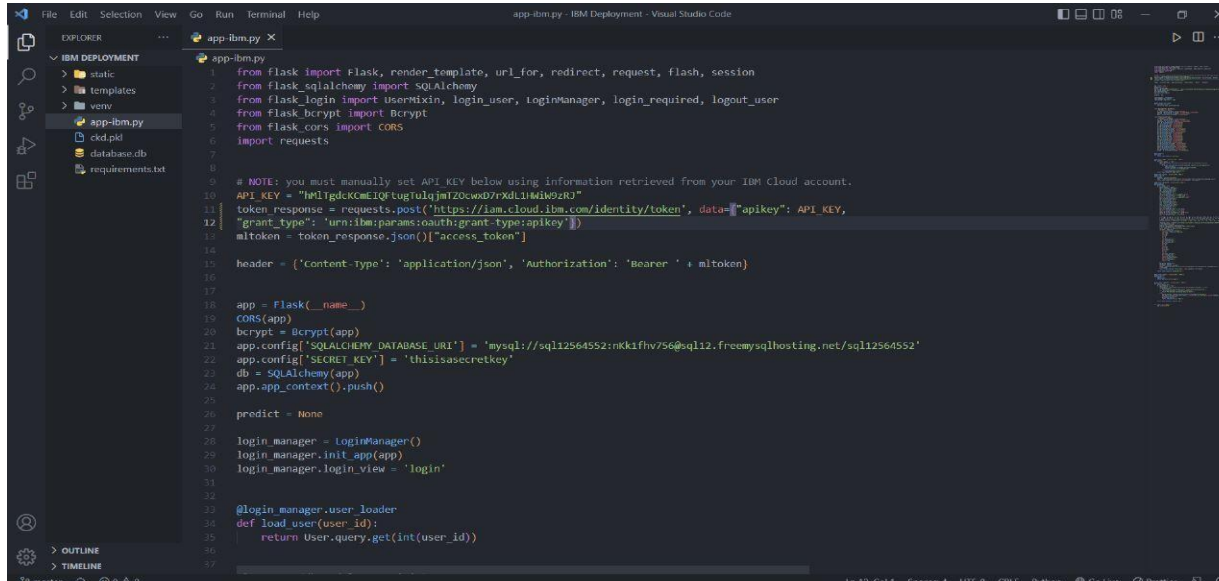
## Using pickle to integrate with flask

```
In [29]: import joblib #created a pickle file using joblib to export the model for frontend usage
        joblib.dump(model,'rfc.pkl') # model - Random Forest Classifier
```

```
Out[29]: ['rfc.pkl']
```

```
In [ ]:
```

## Flask changes for ibm deployment



```
1 from flask import Flask, render_template, url_for, redirect, request, flash, session
2 from flask_sqlalchemy import SQLAlchemy
3 from flask_login import UserMixin, login_user, login_manager, login_required, logout_user
4 from flask_bcrypt import Bcrypt
5 from flask_cors import CORS
6 import requests
7
8
9 # NOTE: you must manually set API KEY below using information retrieved from your IBM Cloud account.
10 API_KEY = "hm1TgdcKMcIQftugIulqjMTZCwxD7rXDL1H64W9zRj"
11 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey": API_KEY,
12 "grant_type": "urn:ibm:params:oauth:grant-type:apikey"})
13 mtoken = token_response.json()["access_token"]
14
15 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mtoken}
16
17
18 app = Flask(__name__)
19 CORS(app)
20 bcrypt = Bcrypt(app)
21 app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://sql12564552:mKk1fhv756@sql12.freemysqlhosting.net/sql12564552'
22 app.config['SECRET_KEY'] = 'thisisasecretkey'
23 db = SQLAlchemy(app)
24 app.app_context().push()
25
26 predict = None
27
28 login_manager = login_manager()
29 login_manager.init_app(app)
30 login_manager.login_view = 'login'
31
32
33 @login_manager.user_loader
34 def load_user(user_id):
35     return User.query.get(int(user_id))
36
37
```



```
sod= float(form.get('sod'))
pot= float(form.get('pot'))
hemo= float(form.get('hemo'))
pcv= float(form.get('pcv'))
wc= float(form.get('wc'))
rc= float(form.get('rc'))
htn= 0 if form.get('htn')== 'no' else 1
dm= 0 if form.get('dm')== 'no' else 1
cad= 0 if form.get('cad')== 'no' else 1
appet= 0 if form.get('appet')== 'good' else 1
pe= 0 if form.get('pe')== 'no' else 1
ane= 0 if form.get('ane')== 'no' else 1

print(age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemo, pcv, wc, rc, htn, dm, cad, appet, pe, ane)
x = [[age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemo, pcv, wc, rc, htn, dm, cad, appet, pe, ane]]
# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"field": ["age", 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu', 'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad', 'appet', 'pe', 'ane']}]

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/4bb20f2e-e060-412e-875e-a336e199f1aa/predictions?version=1', json=payload_scoring, headers={'Authorization': 'Bearer ' + mltoken})
print(response_scoring)
predictions = response_scoring.json()
predict = predictions['predictions'][0]['values'][0][0]
print("final prediction :",predict)
res = 'Positive' if predict==1 else 'Negative'
new_user = prediction[
    username=form.get('username'),
    timestamp = form.get('timestamp'),
    age = age,
    bp = bp,
    sg = sg,
    al= al,
    su= su,
    rbc= form.get('rbc'),
    pc= form.get('pc'),
    pcc= form.get('pcc'),
    ba= form.get('ba'),]
    bgr= bgr,
    bu= bu,
    sc= sc,
    sod= sod,
    pot= pot,
    hemo= hemo,
    pcv= pcv,
    wc= wc,
    rc= rc,
    htn= form.get('htn'),
    dm= form.get('dm'),
    cad= form.get('cad'),
    appet= form.get('appet'),
    pe= form.get('pe'),
    ane= form.get('ane'),
    result = res
]
```

```
timestamp = form.get('timestamp'),
age = age,
bp = bp,
sg = sg,
al= al,
su= su,
rbc= form.get('rbc'),
pc= form.get('pc'),
pcc= form.get('pcc'),
ba= form.get('ba'),
bgr= bgr,
bu= bu,
sc= sc,
sod= sod,
pot= pot,
hemo= hemo,
pcv= pcv,
wc= wc,
rc= rc,
htn= form.get('htn'),
dm= form.get('dm'),
cad= form.get('cad'),
appet= form.get('appet'),
pe= form.get('pe'),
ane= form.get('ane'),
result = res
]

db.session.add(new_user)
db.session.commit()
username = session['username']
table = Prediction.query.filter_by(username=username).order_by(Prediction.timestamp.desc())
print(table)
return render_template('result.html', predict=predict, table=table)

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

# Database (mysql)

MySQL Workbench

IBM - Warning - not supported

File Edit View Query Database Server Tools Scripting Help

Query 1

Limit to 1000 rows

```
1 use sql12564552;
2 select * from users;
3 drop table predict;
4 show tables;
5
```

Result Grid

	id	username	password	email
3	Ajay	\$2b\$12\$1SLQuO.v9KznZYzozBnkYsOugaoOk/QB...	abc@gmail.com	
4	Barath	\$2b\$12\$1.79qnyMpQLFV3f/TEgnocOrN.13kD0C...	xyz@gmail.com	
5	Harish	\$2b\$12\$6gabCKsWSNjA8HOwS1GJuEozu7v54...	def@gmail.com	
6	yash33	\$2b\$12\$HtAQ8FANzvOWYmhZPKWOG5g5gv...	yash33@example.com	
7	santoo	\$2b\$12\$zCg8CL0L.WPmsztwyyzZpP4sWmAqz...	mno@gmail.com	
8	ABCD	\$2b\$12\$K1Ln1t77KqeRkLzfy7XhOJUDq3Zi/PkE...	abcd@gmail.com	
9	bala	\$2b\$12\$JnSc5g1Qz3JnGkDWNe9ujY2.s3GDJE...	bbmbala2002@gmail.com	
10	191223	\$2b\$12\$3q1wIMvRs911Q6WYJKDXeoH22en/k...	191223@psgtech.ac.in	

users 1 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	20:55:01	select * from users LIMIT 0, 1000	Error Code: 1046. No database selected Select the default DB to be used by double-clicking its name in the...	0.078 sec
2	20:55:05	use sql12564552	0 row(s) affected	0.078 sec
3	20:55:08	select * from users LIMIT 0, 1000	12 row(s) returned	0.078 sec / 0.000 sec

MySQL Workbench

IBM - Warning - not supported

File Edit View Query Database Server Tools Scripting Help

Query 1

Limit to 1000 rows

```
1 • use sql12564552;
2 • select * from predict;
3 • drop table predict;
4 • show tables;
5
```

Result Grid

	tamp	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn	dm	cad	appet	pe	ane	result
▶	11-13T16:00	48	70	1.005	4	0	normal	abnormal	present	notpresent	117	56	3.8	111	2.5	11.2	32	6700	3.9	yes	no	no	poor	yes	yes	Positive
	11-13T16:03	41	80	1.02	0	0	normal	normal	notpresent	notpresent	122	25	0.8	138	5	17.1	41	9100	5.2	no	no	no	good	no	no	Negative
	11-13T16:23	63	70	1.01	3	0	abnormal	abnormal	present	notpresent	380	60	2.7	131	4.2	10.8	32	4500	3.8	yes	yes	no	poor	yes	no	Positive
	11-13T17:47	19	70	1.005	0	0	normal	abnormal	present	notpresent	117	60	3.8	111	4.2	17.1	41	6700	5.2	yes	no	no	poor	yes	yes	Positive
	11-13T19:53	41	80	1.02	0	0	normal	normal	notpresent	notpresent	122	25	0.8	138	5	17.1	41	9100	5.2	no	no	no	good	no	no	Negative
	01-24T15:02	48	80	1.02	1	0	normal	normal	notpresent	notpresent	121	36	1.2	104	2.5	15.4	44	7700	5.2	yes	yes	no	good	no	no	Positive
	03-09T09:54	48	80	1.02	1	0	normal	normal	notpresent	notpresent	121	36	1.2	104	2.5	15.4	44	7700	5.2	yes	yes	no	good	no	no	Positive

predict 2

Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2	20:55:05	use sql12564552	0 row(s) affected	0.078 sec
3	20:55:08	select * from users LIMIT 0, 1000	12 row(s) returned	0.078 sec / 0.000 sec
4	20:55:47	select * from predict LIMIT 0, 1000	19 row(s) returned	0.078 sec / 0.000 sec

## 8. TESTING

### 8.1 Test Cases

Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)
Kaggle	1.Enter into kaggle website 2.Download the dataset	<a href="https://www.kaggle.com/">https://www.kaggle.com/</a>	Download the Dataset	Working as expected	Pass		NO
Anaconda prompt , Jupyter Notebook	1.Enter Anaconda prompt 2.Enter Jupyter Notebook & do Data pre-processing		Pre-processing the dataset using machine learning Algorithm	Working as expected	pass		NO
Anaconda prompt , Jupyter Notebook	1.Enter Anaconda prompt 2.Enter Jupyter Notebook & do Model Building	Model building using logistic regression	Build a Machine Learning Model	Working as expected	pass		NO
Visual Studio Code	1.Click on VS code ,create html pages . Run html pages on app.py by using live server .	Run a website in localhost server <a href="http://127.0.0.1:5000/">http://127.0.0.1:5000/</a>	Appears a Prediction page on local host server	Working as expected	pass		NO
Visual Studio Code	Click on the http link Enter the values as in the dataset Click on submit	Gives prediction result as patient have CKD or NOT <a href="http://127.0.0.1:5000/predict">http://127.0.0.1:5000/predict</a>	Predict the Result	Working as expected	Pass		NO
	1.Enter IBM Cloud using login credentials 2.Use jupyter notebook in IBM	Deploy the project in IBM CLOUD	Application should show same resut as vs code flask integration				

**Sample tests:**

Dashboard - CKD Predictor

127.0.0.1:5000/dashboard

CKD PredictorHit yash33DashboardResultLogout

### Chronic Kidney Disease Prediction

Provide your test details below 📌 :

17-11-2022 21:04

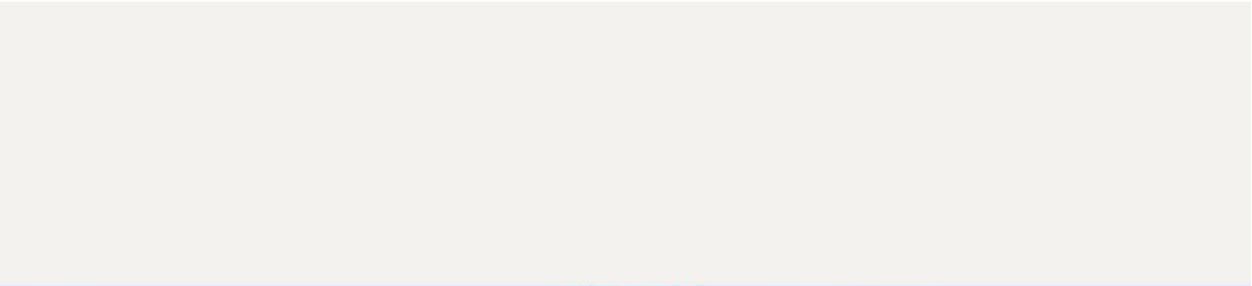
41	80
1.02	0
0	normal
abnormal	present
notpresent	122
32	0.8
139	5

CKD<sup>®</sup> Predictor with ❤️

Your Test Result is: **Positive**

Your Past Test Results

Date	Age	BP	SG	AL	SU	RBC	PC	PCC	BA	BGR	BU	SC	SOD	POT	HEMO	PCV	WC	RC	HTN	DM	CAD	APPET	PE	ANE	Result
2022-11-17T21:04	41.0	80.0	1.02	0.0	0.0	normal	abnormal	present	notpresent	122.0	32.0	0.8	139.0	5.0	14.1	41.0	9100.0	5.2	yes	yes	no	good	no	no	Positive
2022-11-13T19:53	41.0	80.0	1.02	0.0	0.0	normal	normal	notpresent	notpresent	122.0	25.0	0.8	138.0	5.0	17.1	41.0	9100.0	5.2	no	no	no	good	no	no	Negative



## 8.2 User Acceptance Testing

### 1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [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

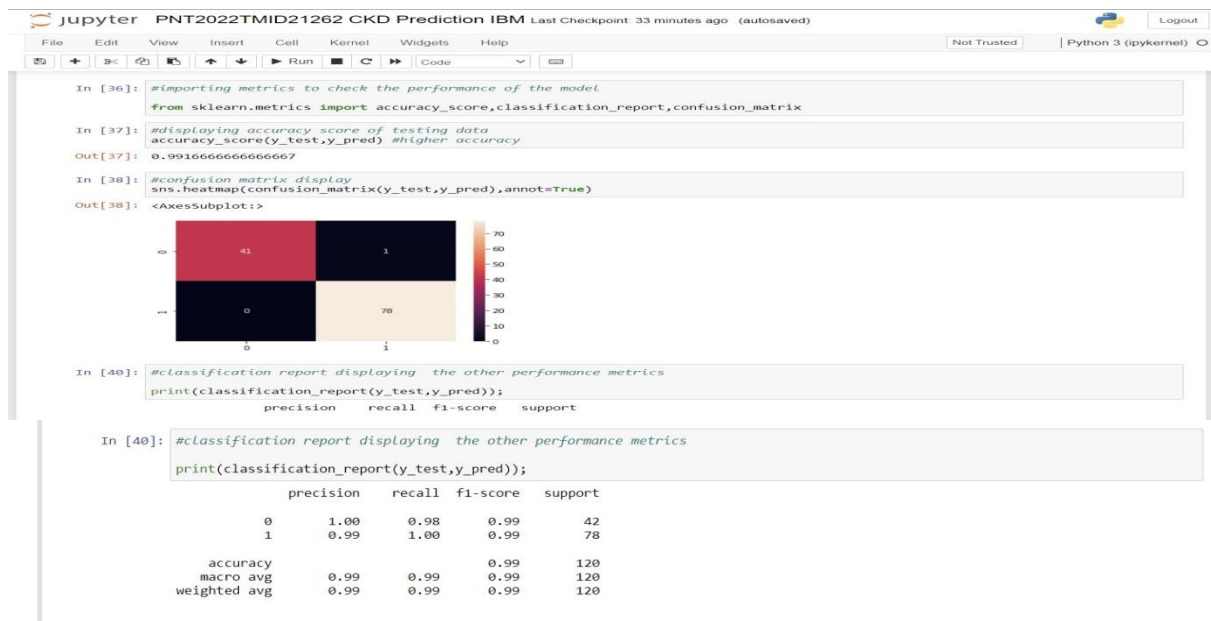
This report shows the number of test cases that have passed, failed, and untested

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

### 4.9.1 Performance Metrics (Random Forest Classifier)



## 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 times 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 its a web application it requires scaling of web application to handle concurrent requests after certain threshold

## 11. CONCLUSION

Chronic Kidney Disease as the name suggests it's a chronic disease,any chronic disease would make the person miserable and last longer till their livelihood . If in such cases the disease gets unnoticed in early stages which can be cured by medical facilities it's a huge carelessness and risking a person's life . In such cases finding an optimal solution is important ,thus there comes the use of a machine learningmodel for early detection and prediction of the chronic kidney disease which can greatly reduce the potential risk of getting the disease and get cured immediately if it is detected in early stages of the disease. Think of the traditional way of diagnosing kidney disease,it is through blood test,andblood test reports take longer than expected ,but blood test is not the only step for diagnosing there are still many more tests taken , which can betime consuming . In those cases the model prediction plays an important role in predicting the disease sooner and faster for the medical team to treat the person if he/she is vulnerable.

Thus early detection of chronic kidney disease is very much necessary in current hospital functioning to diagnose the patient in no time and do necessary treatment to cure if found.

## 12.FUTURE WORK :

The current work remains the base for the prediction model primarily used by everyone extending from hospitals to normal users .The future aspects can be as follows:

- subscription based model can be created with initial trial basis
- Scaling the existing application for simultaneous user to request
- Modifying the model based on adding new feature in the existingdataset based on the hospitals input and standards

## 13. APPENDIX

<https://ieeexplore.ieee.org/abstract/document/8029917> <https://iopscience.iop.org/article/10.1088/1742-6596/1255/1/012024/meta>  
<https://start.atlassian.com/> <https://ieeexplore.ieee.org/abstract/document/9333572>

**GITHUB LINK :**

**<https://github.com/IBM-EPBL/IBM-Project-25744-1659972038>**

**OUR REPOISOTORIES - <https://github.com/Savundhariya7?tab=repositories>**