

PROJECT REPORT DOCUMENTATION

EARLY DETECTION OF CHRONIC KIDNEY DISEASE
TEAM ID - PNT2022TMID21262

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning

6.2 Sprint Estimation and Delivery Schedule

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

- 7.1 User Registration and Login
- 7.2 Dashboard and Result
- 7.3 Flask integration and Deployment
- 7.4 Database

8. TESTING

- 8.1 Test Cases
- 8.2 Sample Testing

9. RESULTS

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

GitHub & Demo Link

1. INTRODUCTION

1.1 Project Overview: -

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

1.2 Purpose: -

The purpose of the project is to alert doctors for an early detection of kidney disease and hence ensure speedy recovery or prevention of kidney disease.

This Project aims at creating a model for early detection of Chronic Kidney Disease using Machine Learning technology. The output is integrated with Flask. The front end developed in html is used to receive user input on various parameters needed to decide on the early detection of kidney disease. The same model is deployed into IBM cloud using API keys and scoring endpoints.

2. LITERATURE SURVEY

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 Perkusich	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

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 Machine Learning Techniques	Reem Alassaf, Khawla Alsulaim, Noura Alroomi, Nouf Alsharif, Mishael Aljubeir, Sunday Olatunji, Alaa Alahmadi, Mohammed Imran,	ANN, SVM, Naïve Bayes along with k-NN comparison approach	ANN,SVM ,Naïve Bayes - 98% k-NN - 93.9%	2018

		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 FahimSyed, Nuha Radwan Khayat,Rawan KanaanAltheeb,Renad Emad Juri,Jamal Alhiyafi,Sale	Comparative analysis was carried out on the two models-ANN and SVM	ANN - 99.75% SVM - 97.75%	2019

		<p>h Alrashed,Su nday O.Olatunji</p>			
--	--	--	--	--	--

2.3 Problem Statement Definition

Chronic kidney disease (CKD) is one of the most critical health problems due to its increasing prevalence. Chronic kidney disease, also known as chronic renal disease or CKD, is a condition characterized by a gradual loss of kidney function over time. It includes conditions that damage your kidneys and decrease their ability to keep you healthy by filtering wastes from your blood. Diabetes and high blood pressure, or hypertension, are responsible for two-thirds of chronic kidney disease cases. Anyone can get chronic kidney disease at any age. However, some people are more likely than others to develop kidney disease. Most people may not have any severe symptoms until their kidney disease is advanced.

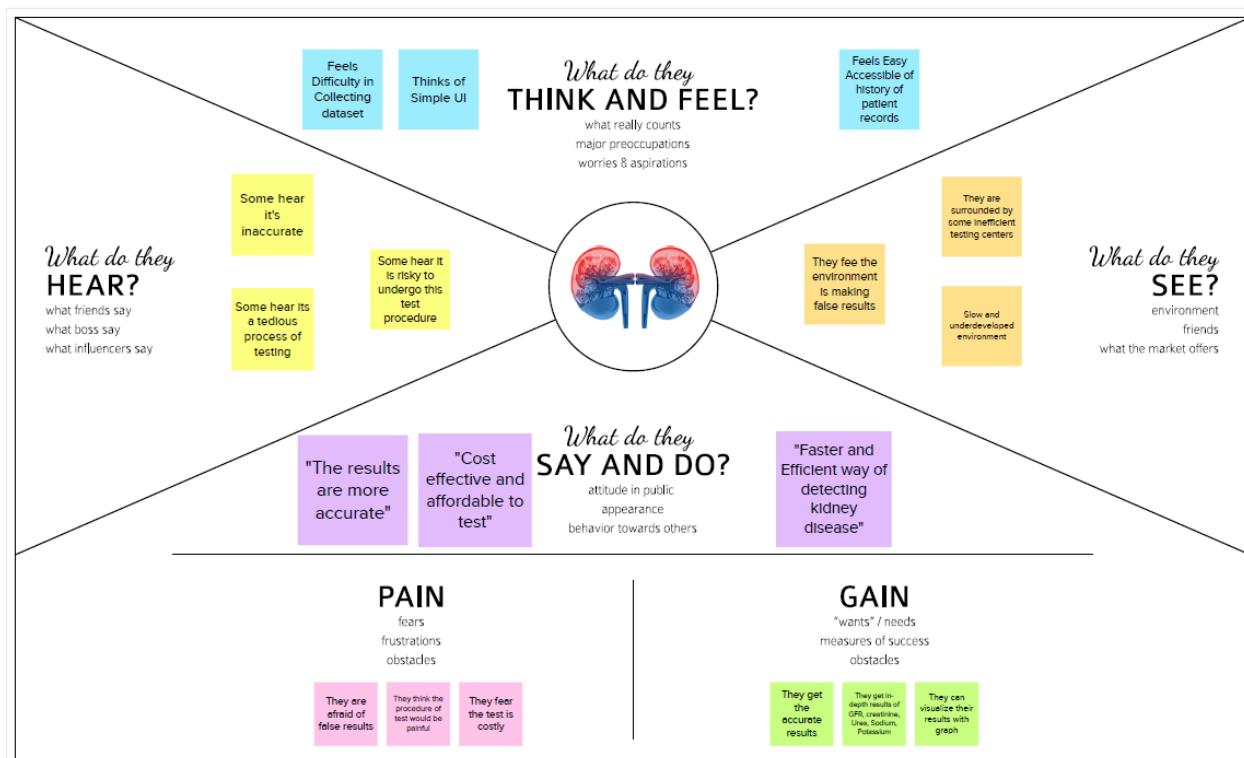
A better testing method which could possibly detect CKD in the early stages would be much more useful. Medical test results taken for other purposes are used to detect CKD at early

stages. 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. Peculiar and contributing attributes from the above-mentioned test results are combined to develop a Machine Learning Model.

This Machine Learning Model will be used to predict CKDs rather early than the presently existing methods.

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas:-



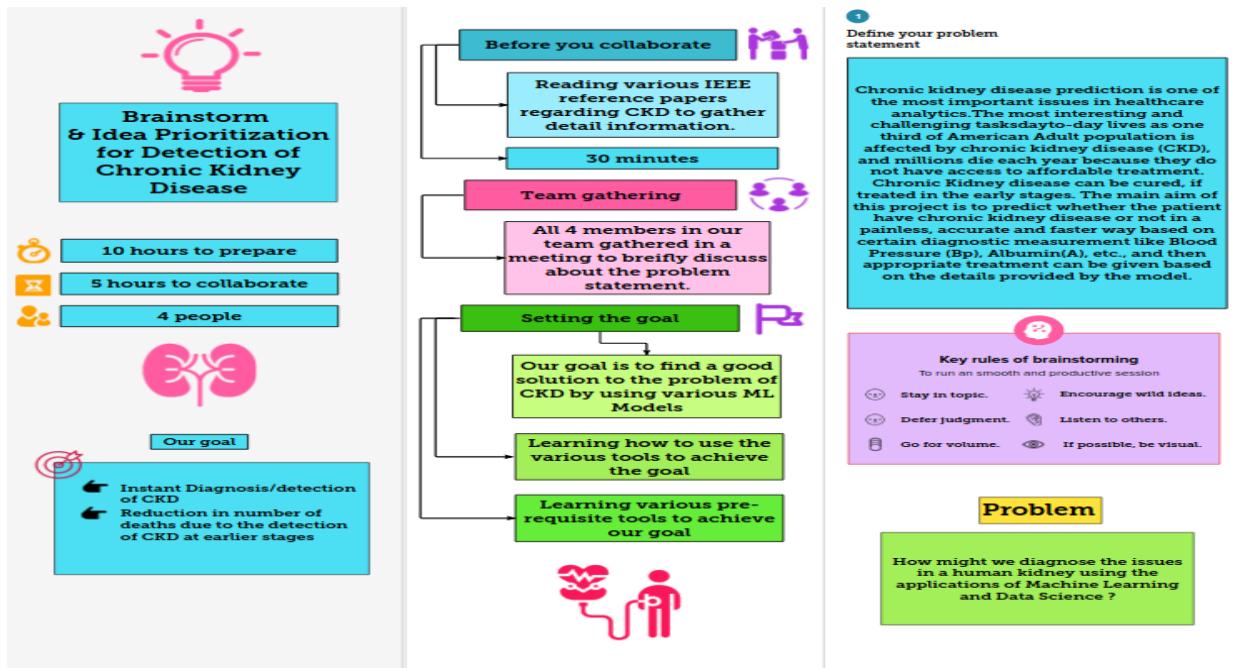
3.2 Ideation & Brainstorming:-

Brainstorming is an activity that will help you generate more innovative ideas. It's one of many methods of ideation—the process of coming up with new ideas—and it's core to the design thinking process.

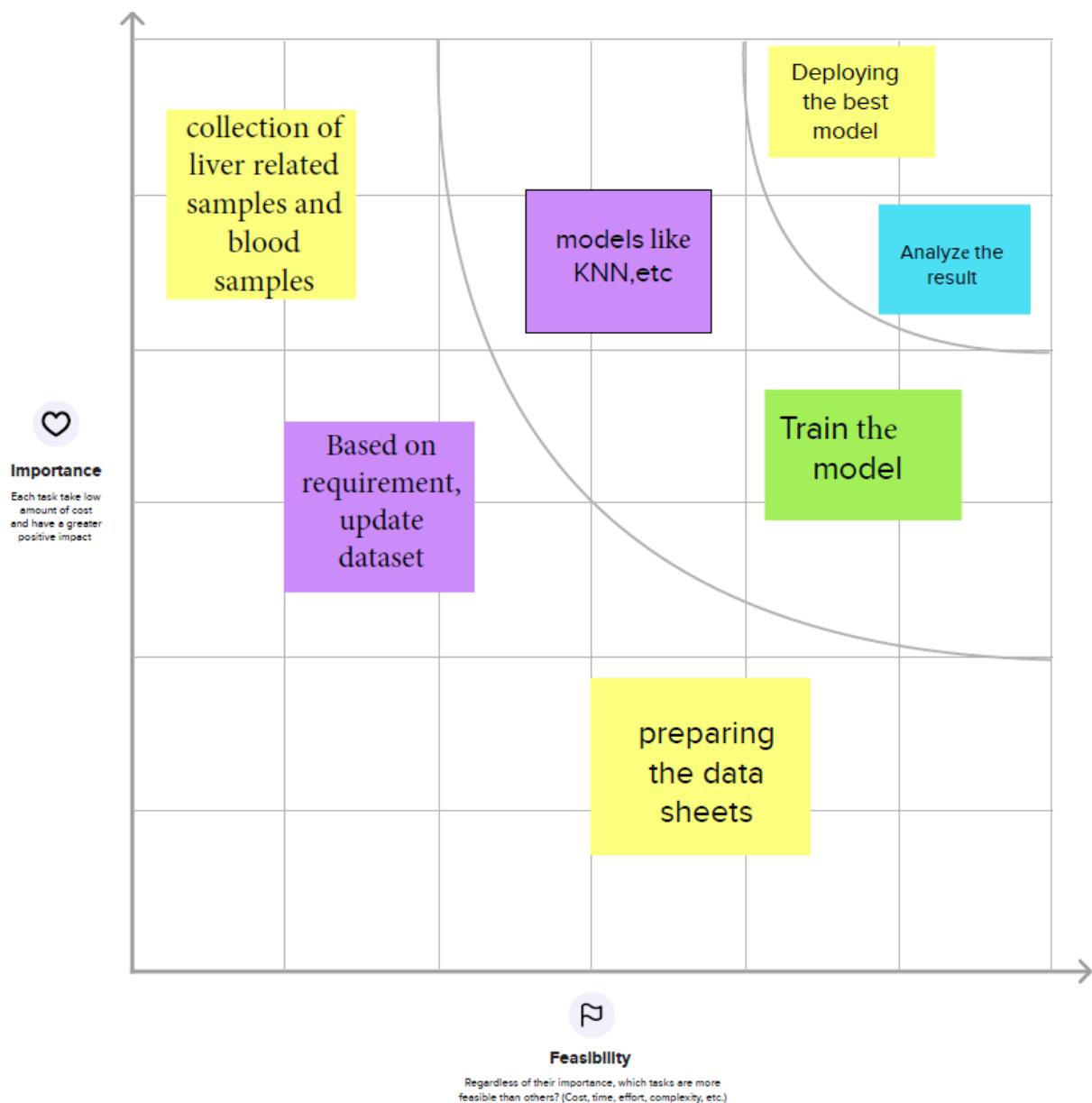
Brainstorming refers to a problem-solving technique used by teams or individuals. In this process, participants generate various ideas or solutions, then begin discussing and narrowing them down to the best options.

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. 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.**

Ideation is the process where you generate ideas and solutions through sessions such as Sketching, Prototyping, Brainstorming, Brainwriting, Worst Possible Idea, and a wealth of other ideation techniques.



Stage 4 - prioritize



3.3 Proposed Solution:-

The purpose of this tool is to provide a structured process for identifying a problem, understanding the

root causes, ascertaining solution steps, and progress monitoring.

With a solution template, you can organize development content that you want to reuse for customer-specific solutions. Solution templates enable you to easily start the development of customer-specific solutions, for example, for a specific industry.

The term business model refers to a company's plan for making a profit. It identifies the products or services the business plans to sell, its identified target market, and any anticipated expenses. Business models are important for both new and established businesses.

Proposed Solution Template:

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	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
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> ● Easy to use User interface (UI) ● accurate accuracy by comparing the performance of different ml model technique

4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> ● Greater cost reduction in hospitals for testing ● Helps in early diagnosis of the disease ● Chances of recovery is higher
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> ● subscription based model with initial trial basis ● charges/commission for the actual prediction and recovery of a person
6.	Scalability of the Solution	<ul style="list-style-type: none"> ● The server in which the app is deployed containing the ml model must be capable of handling concurrent request and handle multiple request

BUSINESS MODEL

Business Model				
Key Partners	Key Activities	Value Propositions	Customer Relationships	Customer Segments
Our Key partners have a latest testing laboratory to collect samples of blood and urine to analyse various parameters with a testing capacity of almost 1000 persons a day. Our ML Models can test a large number of samples and give accurate output within a short period of time.	Our key activities are to find out whether persons have chronic kidney disease or not.	We are trying to solve chronic kidney disease at an early stage thereby helping them to recover at a faster rate. We are targeting persons liable to get chronic kidney disease using statistical analysis of the collected data.	We need to have a cordial relationship with the persons coming forward for giving blood test. They expect accurate result from our ML model. The cost of building the model may outfit the early detection of chronic kidney disease.	We are creating value for humans. Our customers are common people who have work culture or wrong lifestyle habits which may lead to chronic kidney disease. Customer base will be a mass market as everyone will give importance to health. We will also be providing a diet regulatory counselling to them.
Key Resources			Channels	
	Our key resources are testing laboratories, ML Models and their data. To create an excel sheet from the data.		Need to reach our patients through doctors. Using ML model to suggest best practices which can help doctors and patients in avoiding and recovering chronic kidney disease.	
Cost Structure			Revenue Streams	
Most important cost is mainly for data collection through laboratories. This model is cost driven as it has maximum automation with only a few questions needed to be asked to patients. As compared to older methods, ML Detection technique is cost effective.			Patient's visiting doctors are willing to pay for knowing if they have chronic kidney disease or not. Overall revenue is good and it depends on the accuracy of ML model which will help the doctors to do a correct prediction.	

3.4 Problem Solution Fit:-

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.

Problem-Solution Fit - this occurs when you have evidence that customers care about certain jobs, pains, and gains. At this stage you've proved the existence of a problem and have designed a value proposition that addresses your customers' jobs, pains and gains.

Problem-Solution fit canvas 2.0				Early Detection of Chronic Kidney Disease Using Machine Learning
Define CS, fit into CC	1. CUSTOMER SEGMENT(S)	6. CUSTOMER CONSTRAINTS	5. AVAILABLE SOLUTIONS	AS
Focus on J&P, tap into BE, understand RC	<p>Patients who are facing issues related to kidneys. Elderly people, are more prone to get kidney disease. Diabetic Patients Alcoholic addicted Patients</p>	<p>Patients are afraid about risk of using new technology They are limiting themselves as they are not aware of the test accuracies</p>	<p>Currently in the Medical field, the tests that are performed to detect chronic kidney disease are:</p> <ol style="list-style-type: none"> 1. Ultra Sound Scan 2. MRI Scan 3. CT Scan 	Explore AS, differentiate
Identity strong TR & EM	<p>2. JOBS-TO-BE-DONE / PROBLEMS</p> <p>Problems related to identifying the chronic kidney disease Accuracy of patients test results Time taken to produce test results</p>	<p>9. PROBLEM ROOT CAUSE</p> <p>The root cause of the problem is inaccurate results. The test takes much time to evaluate the results.</p>	<p>7. BEHAVIOUR</p> <p>They take costly Scans because they had no other choice. They blindly trust the inaccurate test results and become more anxious and sad.</p>	Focus on J&P, tap into BE, understand RC
TR	<p>3. TRIGGERS</p> <p>Their dilemma or confusion of whether they really have chronic kidney disease or not!</p>	<p>10. YOUR SOLUTION</p> <p>Predicts Faster and accurately. Time and Cost of Test is drastically reduced Helps to take treatment at right time.</p>	<p>8. CHANNELS OF BEHAVIOUR</p> <p>They consider taking tests costing lower from any of the online labs.</p>	CH
EM	<p>4. EMOTIONS: BEFORE / AFTER</p> <p>BEFORE: Anxious about their medical condition. AFTER: Determined and able to follow doctor's advice on what to do next to improve their condition</p>		<p>8.2 OFFLINE</p> <p>They take many tests in offline labs and wait for enormous time to get results</p>	Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:-

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail
FR-2	User Verification	Confirmation via Email
FR-3	User Login	Login through Email
FR-4	User Help	Report issues through Email
FR-5	Disease prediction	User can predict the disease based on the valid inputs given by him/her

4.2 Non- Functional Requirements:-

Non-functional Requirements:

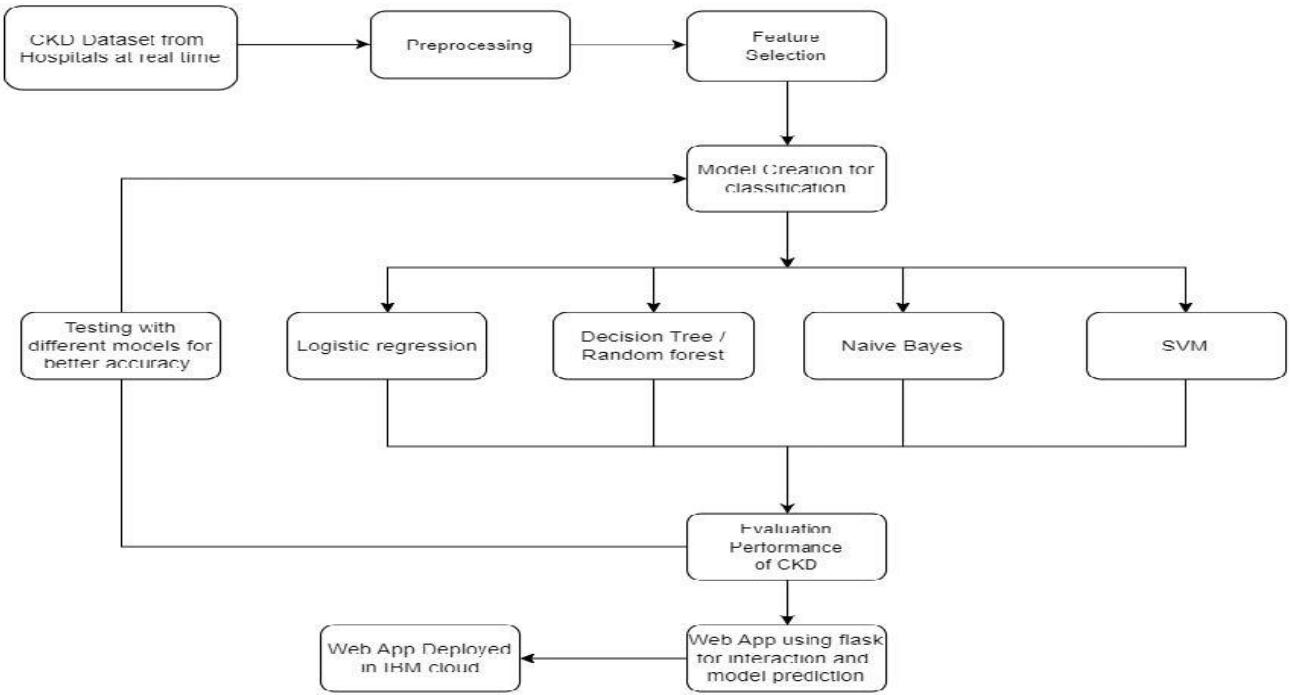
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	In terms of accuracy and efficiency, this project is the best for predicting kidney disease
NFR-2	Security	It is highly secured as there is separate user id and password to login
NFR-3	Reliability	It is reliable as it is user friendly and cost-efficient
NFR-4	Performance	In terms of time, this project is highly efficient as we have used very efficient algorithms like regression
NFR-5	Availability	Since it is web based one, it is available to those who have access to web
NFR-6	Scalability	If user demand increases, it can be scaled up easily as we made everything dynamic

5. PROJECT DESIGN

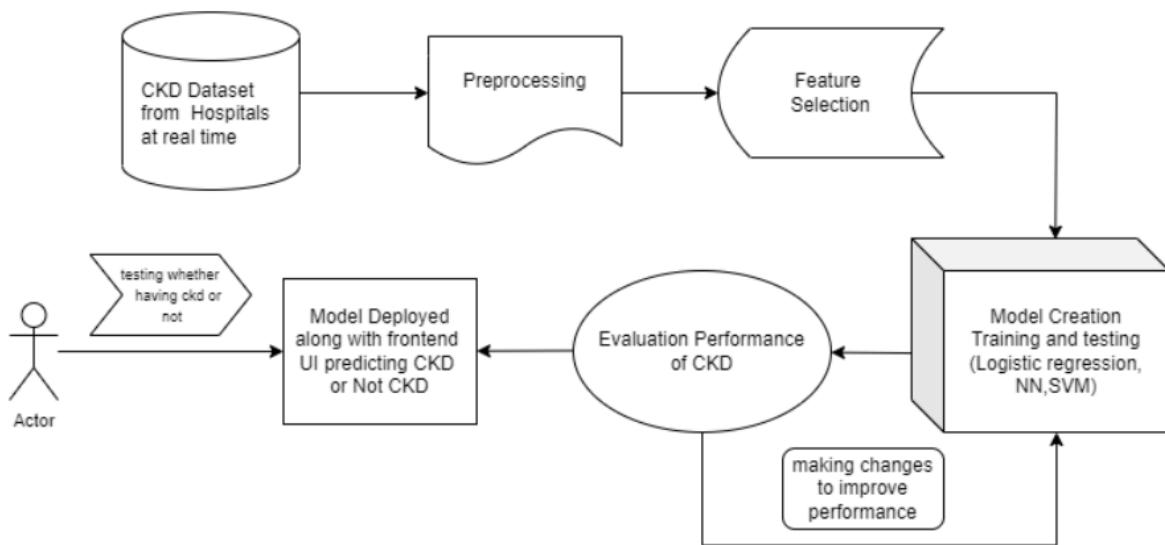
5.1 Data Flow Diagrams:-

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 stored.



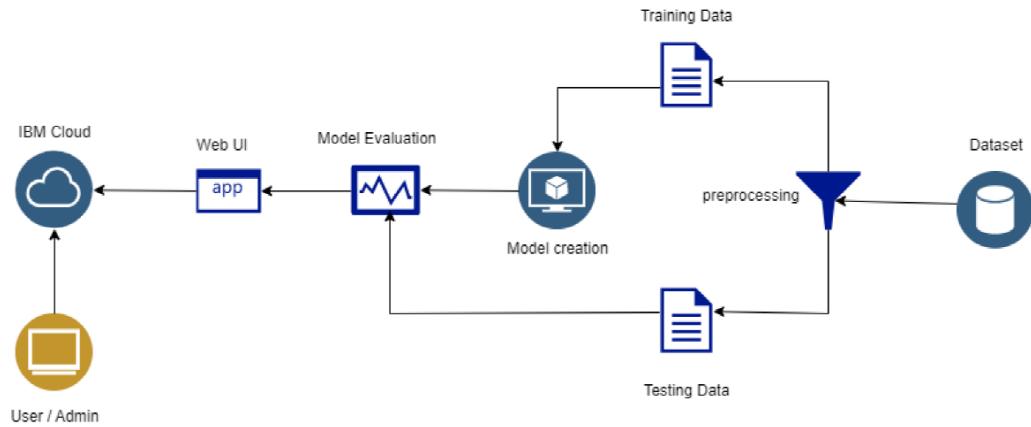
4.2 Solution & Technical Architecture: -

Solution Architecture:



Technical Architecture:

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 AND SCHEDULING

6.1 Sprint Planning

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. Sprint 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.

Technical Architecture:

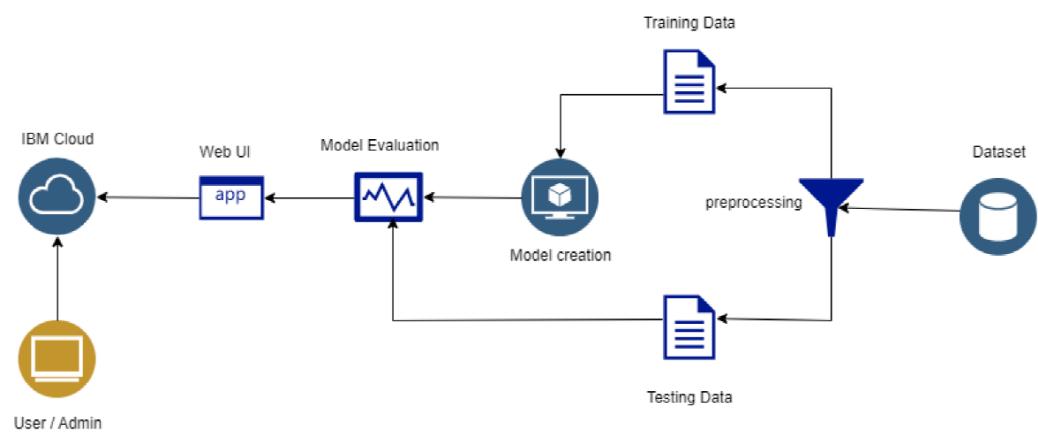


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User Interact with our application through web user interface	HTML, CSS , Flask , React (Subsidiary)
2.	Application Logic-1 (Registration)	User is redirected to register page for registering themselves by providing valid details	HTML, CSS , Flask , React (Subsidiary)
3.	Application Logic-2 (Login)	Once the user is registered he is now able to login to access the web service . There is an external login button to redirect to login page	HTML, CSS , Flask , React (Subsidiary)
4.	Application Logic-3 (Test / Prediction)	The test or prediction page is present for the user who has logged in and can predict the disease by providing input in the form.	HTML, CSS , Flask , React (Subsidiary)
5.	Database	Data Type - String , Numbers	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Role Based Access is provided for using the API	Backend API
9.	External API-2	Purpose of External API used in the application	NIL
10.	Machine Learning Model	To predict the output based on the training and testing of the data from dataset	Data Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	NIL

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Frameworks are used in both in making web app and for model creation	Flask (micro web framework) , python (web framework) , Scikit-learn (ML framework)
2.	Security Implementations	passwords are hashed for the user , as well roles are provided for access based control system	SHA 256
3.	Scalable Architecture	The Scalability can be improvised by using three-tier architecture	Three tier architecture
4.	Availability	Scalability includes availability, the service must be available even if there are more user request ,load balancer is needed to do the above task	Load Balancer
5.	Performance	Performance is key for increased revenue , handling multiple requests and expanding it can be done using Load Balancer.	Load Balancer

6.2 Sprint Estimation and Delivery Schedule:

A sprint estimation shows how much effort a series of tasks require. It's based on assumptions, requirements, and dependencies of a project.

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

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Collecting the dataset	USN-1	Collected dataset from Kaggle related to our problem statement.	2	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-1	Cleaning the dataset	USN-2	Cleaning dataset by handling and replacing missing values and making effective for prediction.	3	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-1	Model Building	USN-3	Building the model and selecting best ML model based on accuracy for accurate prediction.	4	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Model Evaluation	USN-4	Evaluating the model and finding accuracy for the model.	4	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-2	User Registration	USN-5	As a user, I can register for the application by entering my email, password, and confirming my password.	2	Medium	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-2	User Login	USN-6	As a user, I can log into the application by entering email & password	2	Medium	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-2	User Verification	USN-7	Verifying the user through email.	1	Low	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-2	Dashboard	USN-8	Designing HTML Dashboard page to navigate for the logged in users.	3	Medium	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-3	Integration	USN-9	Using Flask to integrate user dashboard with trained model to predict and request result.	4	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Train ML Model in IBM	USN-10	The ML model will be trained in IBM by hosting ipynb file in IBM Cloud.	3	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-3	Integrating with IBM Cloud	USN-11	Integrating trained IBM Cloud Model with scoring endpoints using flask	4	High	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-4	Deployment	USN-12	Deploying the IBM model backend flask and frontend application in Cloud	3	Medium	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S
Sprint-4	Further Classification	USN-13	Getting user experience feedback and improving the model and application through customer feedback	2	Medium	Periyakaruppan N Yashwant C M Venkatesh N Ashwath S

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

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	13	6 Days	24 Oct 2022	29 Oct 2022	13	29 Oct 2022
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	11	6 Days	07 Nov 2022	12 Nov 2022	11	07 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022	5	14 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$\text{Sprint 1 AV} = \text{Sprint duration}/\text{velocity} = 13/6 = 2.16$$

$$\text{Sprint 2 AV} = \text{Sprint duration}/\text{velocity} = 8/6 = 1.33$$

$$\text{Sprint 3 AV} = \text{Sprint duration}/\text{velocity} = 11/6 = 1.83$$

$$\text{Sprint 4 AV} = \text{Sprint duration}/\text{velocity} = 5/6 = 0.83$$

6.3 Reports from JIRA:

The screenshot shows the Jira Software interface for the project "PNT2022TMID21262". The left sidebar includes links for Roadmap, Backlog, Board, Code, Project pages, Add shortcut, and Project settings. The main area displays a "Roadmap" view with a timeline from October to January 2023. A vertical orange line marks the end of November. Several tasks are plotted across the sprints: "PNT2022TMI-2 model building" (Oct 31 - Nov 10), "PNT2022TMI-4 front end registration lo..." (Nov 10 - Nov 15), "PNT2022TMI-6 flask integration with lo..." (Nov 15 - Nov 20), and "PNT2022TMI-8 IBM deployment" (Nov 20 - Nov 25). A search bar and filter options are at the top of the main area.

The screenshot shows the Jira Software interface for the project "PNT2022TMID21262". The left sidebar includes links for Backlog, Board, Code, Project pages, Add shortcut, and Project settings. A banner at the top offers a free trial of the Standard plan. The main area displays a "Backlog" view with three sprint sections: Sprint 2 (31 Oct – 5 Nov, 1 issue), Sprint 3 (7 Nov – 12 Nov, 1 issue), and Sprint 4 (14 Nov – 19 Nov, 1 issue). Each sprint section contains a single task: "PNT2022TMI-3 PNT2022TMID21262_Front_end_Registration" (Done), "PNT2022TMI-5 PNT2022TMID21262_Flask_Integration_local" (In Progress), and "PNT2022TMI-7 PNT2022TMID21262_IBM_Deployment" (In Progress). A search bar and filter options are at the top of the main area.

The screenshot shows the Jira Software interface for the project PNT2022TMID21262. The left sidebar includes links for Planning (Roadmap, Backlog), Development (Code), Project pages, Add shortcut, and Project settings. The main area displays a board titled "All sprints" with three columns: "TO DO", "IN PROGRESS 1 ISSUE", and "DONE 2 ISSUES". The "DONE" column lists two issues: "PNT2022TMID21262_Front_end_Registration" and "PNT2022TMI-3". A "Quickstart" button is visible at the bottom right.

7. CODING & SOLUTIONING

7.1 User Registration and login :

Register.html:

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
    <title>CKD Predictor</title>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css') }}>
    <link rel="icon" type="image/x-icon" href="{{ url_for('static', filename='logo.png') }}>
  </head>
  <body class=" border-0">
    <!-- Example Code -->
    <nav class="navbar-expand-lg navbar-dark bg-primary bottom-0">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">
        </a>
        <a class="navbar-brand" href="#">
          CKD Predictor</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
          aria-controls="navbarNav" aria-expanded="false" aria-label="toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse d-flex" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link" href="{{ url_for('login') }}>Login</a>
            </li>
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="#">Register</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>
  </body>
</html>

```

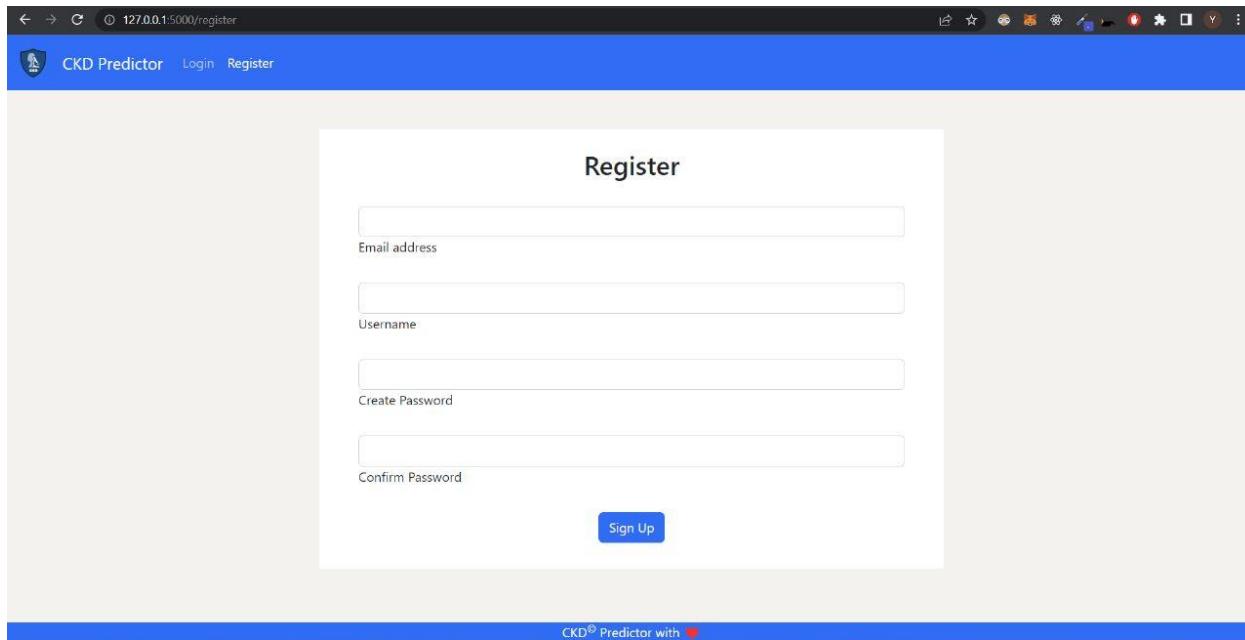
The code editor shows the content of the register.html file. It includes a navigation bar with a logo, a title "CKD Predictor", and links for "Login" and "Register". It also includes Bootstrap CSS imports and a script for the CKD Predictor. The code is written in HTML with some Python templating syntax for URLs.

The screenshot shows a code editor interface with a sidebar on the left containing project files and navigation buttons like 'OUTLINE' and 'TIMELINE'. The main area displays a Python template file named 'register.html' with line numbers from 36 to 92. The template uses Jinja2 syntax to render a registration form. It includes sections for messages, email input, username input, and password input. The code is well-formatted with syntax highlighting for HTML, CSS classes, and Jinja2 tags.

```
register.html
templates > register.html > html > body.border-0 > nav.navbar.navbar-expand-lg.navbar-dark.bg-primary.bottom-0 > div.container-fluid > div#navbarNav.collapse.navbar-collapse.d-flex
36     <a class="nav-link active" aria-current="page" href="#">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">Register</h2><br>
45     <div>
46       {% with messages = get_flashed_messages(with_categories=true) %}
47       {% if messages %}
48         <p class="flash">
49           {% for category, message in messages %}
50             {{ message }}
51           {% endfor %}
52         </p>
53       {% endif %}
54     </div>
55     <!-- Email input -->
56     <div class="form-outline mb-4">
57       <input type="email" id="email" name="email" class="form-control" required />
58       <label class="form-label" for="email">Email address</label>
59     </div>
60
61     <!-- Username input -->
62     <div class="form-outline mb-4">
63       <input type="username" id="username" name="username" class="form-control" required />
64       <label class="form-label" for="username">Username</label>
65     </div>
66
67     <!-- Password input -->
68     <div class="form-outline mb-4">
69       <input type="password" id="password" name="password" class="form-control" required />
70       <label class="form-label" for="password">Create Password</label>
71     </div>
72
```

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure with files like `register.html`, `home.html`, `login.html`, `result.html`, `app.pkl.py`, `ckd.pkl`, `database.db`, and `requirements.txt`.
- Code Editor:** The `register.html` file is open, displaying HTML code for a registration form. The code includes fields for password and confirm password, a submit button, and a footer with a copyright notice.
- Bottom Status Bar:** Shows the current file is `register.html`, line 71, column 38. It also displays icons for master branch, diff, status, and other development tools.



Login.html

File structure:

```

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

```

Content of login.html:

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
    <title>CKD Predictor</title>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}>
    <link rel="icon" type="image/x-icon" href="{{ url_for('static', filename='logo.png') }}>
  </head>
  <body class="border-e">
    <!-- Example Code -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">![Logo]({{ url_for('static', filename='logo.png') }})Login</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="{{ url_for('register') }}>Register</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>
  </body>

```

```
EXPLORER    login.html x
LOCAL DEPLOYMENT
> static
  templates
    dashboard.html
    home.html
    login.html
    register.html
    result.html
> venv
  app-pk.py
  cdk.pkd
  database.db
  requirements.txt

templates > login.html > html
  <!-- Navigation bar -->
  <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0" >
    <div class="container-fluid" >
      <ul class="nav flex-column justify-content-end" style="list-style-type: none;" >
        <li class="nav-item" >
          <a class="nav-link" href="{{url_for('register')}}">Register</a>
        </li>
      </ul>
    </div>
  </nav>

  <div class="w-50 mx-auto mt-5 bg-white px-5 py-4" >
    <form method="post" action="">
      <h2 class="text-center">Login</h2><br>

      <!-- Username input -->
      <div class="form-outline mb-4" >
        <input type="username" id="username" name="username" class="form-control" required />
        <label class="form-label" for="username">Username</label>
      </div>

      <!-- Password input -->
      <div class="form-outline mb-4" >
        <input type="password" id="password" name="password" class="form-control" required />
        <label class="form-label" for="password">Password</label>
      </div>

      <% with messages = get_flashed_messages(with_categories=True) %>
      <% if messages %>
        <p class="flash">
          <% for category, message in messages %>
            {{ message }}<br>
          <% endfor %>
        </p>
        <% endif %>
      <% endifwith %>
    </div>

    <!-- Submit button -->
    <div class="text-center"><button type="submit" class="btn btn-primary btn-block mb-2 ">Sign in</button>
    </div>
    <div class="text-center mt-3">Not yet registered? <a href="{{url_for('register')}}">Register</a></div>
  </div>

  <!-- Footer -->
  <div class="text-center footer" >
    CKD<sup class="footerc">&copy;</sup> Predictor with ❤
  </div>
</body>
</html>
```

Ln 20, Col 35 Spaces:4 UTF-8 CRLF HTML ⚡ Go Live ✨ Prettier

```
EXPLORER    login.html x
LOCAL DEPLOYMENT
> static
  templates
    dashboard.html
    home.html
    login.html
    register.html
    result.html
> venv
  app-pk.py
  cdk.pkd
  database.db
  requirements.txt

templates > login.html > html
  <!-- Navigation bar -->
  <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0" >
    <div class="container-fluid" >
      <ul class="nav flex-column justify-content-end" style="list-style-type: none;" >
        <li class="nav-item" >
          <a class="nav-link" href="{{url_for('register')}}">Register</a>
        </li>
      </ul>
    </div>
  </nav>

  <div class="w-50 mx-auto mt-5 bg-white px-5 py-4" >
    <form method="post" action="">
      <h2 class="text-center">Login</h2><br>

      <!-- Username input -->
      <div class="form-outline mb-4" >
        <input type="username" id="username" name="username" class="form-control" required />
        <label class="form-label" for="username">Username</label>
      </div>

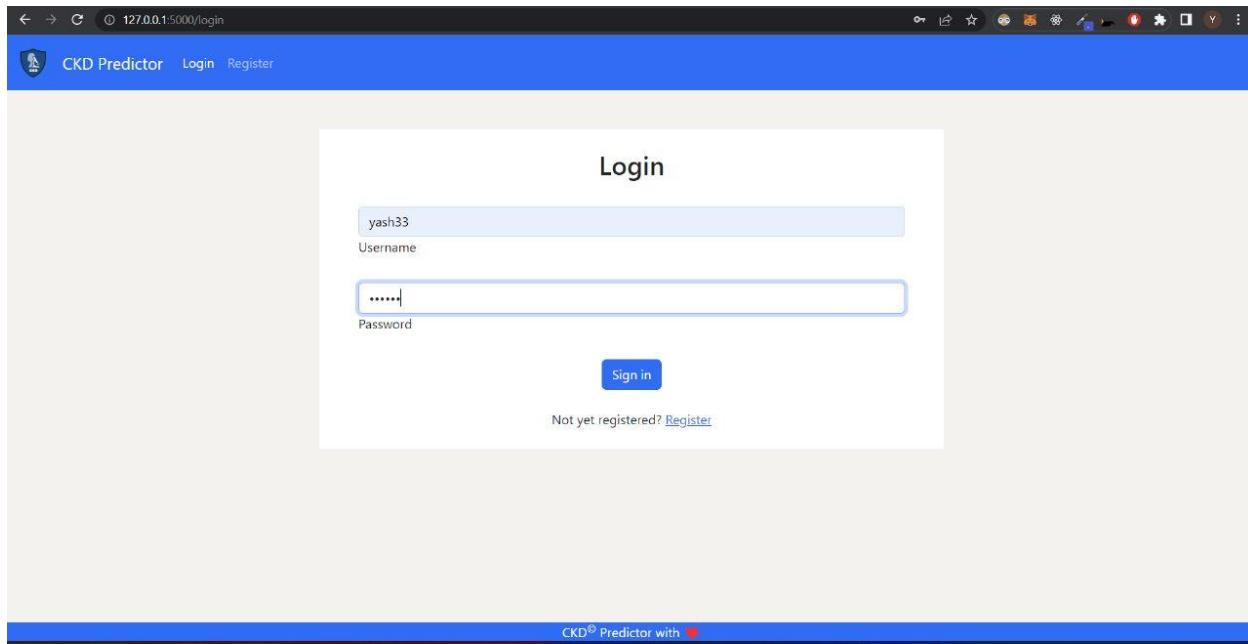
      <!-- Password input -->
      <div class="form-outline mb-4" >
        <input type="password" id="password" name="password" class="form-control" required />
        <label class="form-label" for="password">Password</label>
      </div>

      <% with messages = get_flashed_messages(with_categories=True) %>
      <% if messages %>
        <p class="flash">
          <% for category, message in messages %>
            {{ message }}<br>
          <% endfor %>
        </p>
        <% endif %>
      <% endifwith %>
    </div>

    <!-- Submit button -->
    <div class="text-center"><button type="submit" class="btn btn-primary btn-block mb-2 ">Sign in</button>
    </div>
    <div class="text-center mt-3">Not yet registered? <a href="{{url_for('register')}}">Register</a></div>
  </div>

  <!-- Footer -->
  <div class="text-center footer" >
    CKD<sup class="footerc">&copy;</sup> Predictor with ❤
  </div>
</body>
</html>
```

Ln 79, Col 6 Spaces:4 UTF-8 CRLF HTML ⚡ Go Live ✨ Prettier



7.2 Dashboard and Result

Dashboard.html

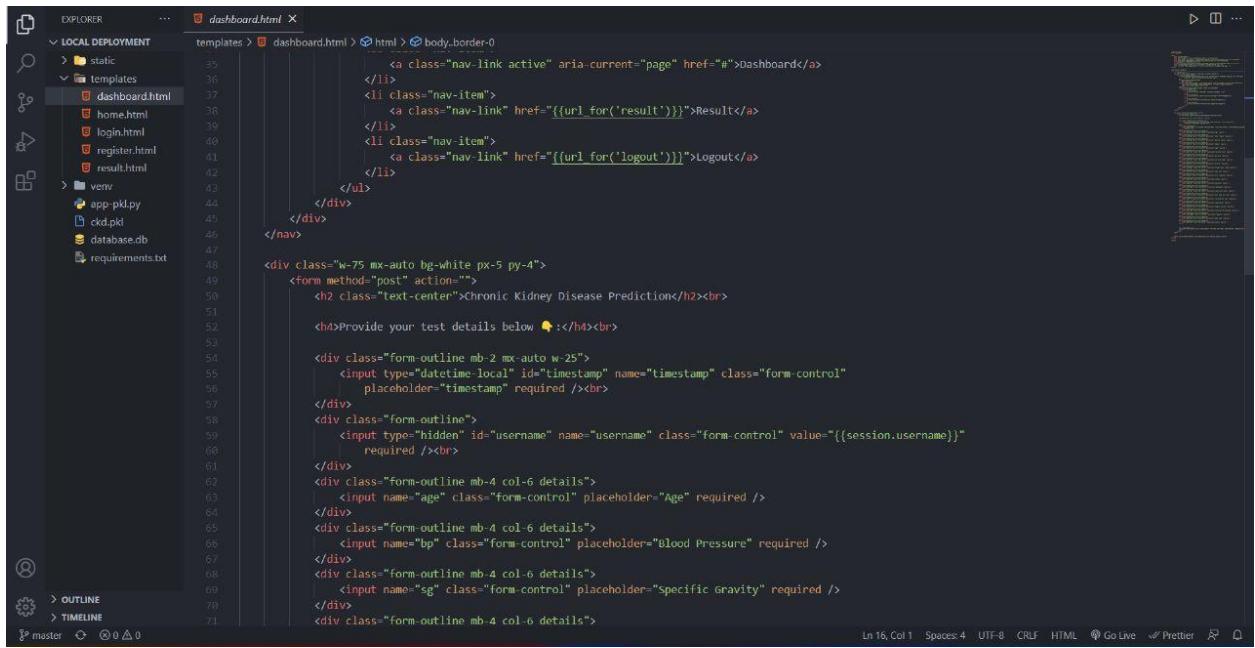
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
    <title>Dashboard - CKD Predictor</title>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}>
    <link rel="icon" type="image/x-icon" href="{{ url_for('static', filename='logo.png') }}>
  </head>
  <body class="border-0">
    <!-- Example Code -->
    <nav class="navbar navbar-expand-lg navbar-dark bg-primary bottom-0">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">![Logo]({{ url_for('static', filename='logo.png') }})
        <a class="navbar-brand" href="#">CKD Predictor</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse d-flex" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" href="#">

# {{ session.username }} 💚

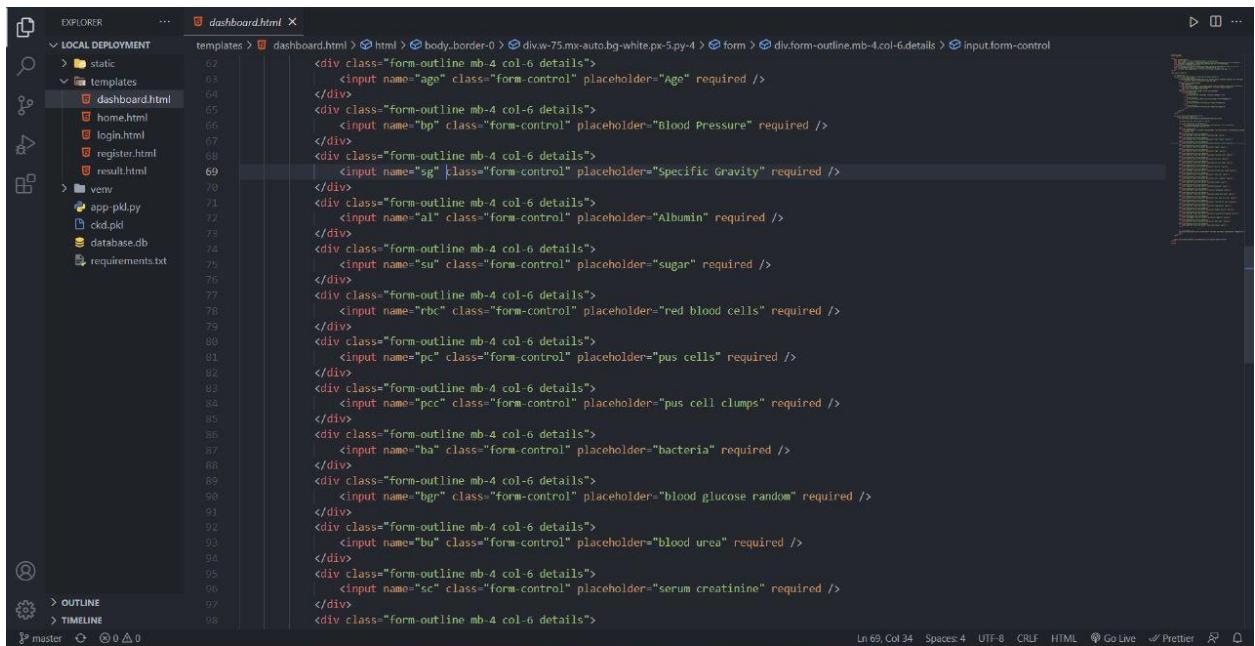
>
            </li>
            <li class="nav-item">
              <a class="nav-link active" aria-current="page" href="#">

## Dashboard

>
            </li>
            <li class="nav-item">
```



```
35         <a class="nav-link active" aria-current="page" href="#">Dashboard</a>
36     </li>
37     <li class="nav-item">
38         <a class="nav-link" href="#">{url_for('result')}>Result</a>
39     </li>
40     <li class="nav-item">
41         <a class="nav-link" href="#">{url_for('logout')}>Logout</a>
42     </li>
43 </ul>
44 </div>
45 </div>
46 </nav>
47
48 <div class="w-75 mx-auto bg-white px-5 py-4">
49     <form method="post" action="#">
50         <h2 class="text-center">Chronic Kidney Disease Prediction</h2><br>
51
52         <h4>Provide your test details below <img alt="pencil icon" style="vertical-align: middle;" data-bbox="450 275 465 290"/></h4><br>
53
54         <div class="form-outline mb-2 mx-auto w-25">
55             <input type="datetime-local" id="timestamp" name="timestamp" class="form-control" placeholder="timestamp" required /><br>
56         </div>
57         <div class="form-outline mb-4 col-6 details">
58             <input type="hidden" id="username" name="username" class="form-control" value="{session.username}" required /><br>
59         </div>
60         <div class="form-outline mb-4 col-6 details">
61             <input name="age" class="form-control" placeholder="Age" required />
62         </div>
63         <div class="form-outline mb-4 col-6 details">
64             <input name="bp" class="form-control" placeholder="Blood Pressure" required />
65         </div>
66         <div class="form-outline mb-4 col-6 details">
67             <input name="sg" class="form-control" placeholder="Specific Gravity" required />
68         </div>
69         <div class="form-outline mb-4 col-6 details">
70             <input name="al" class="form-control" placeholder="Albumin" required />
71         </div>
72         <div class="form-outline mb-4 col-6 details">
73             <input name="su" class="form-control" placeholder="sugar" required />
74         </div>
75         <div class="form-outline mb-4 col-6 details">
76             <input name="rbc" class="form-control" placeholder="red blood cells" required />
77         </div>
78         <div class="form-outline mb-4 col-6 details">
79             <input name="pc" class="form-control" placeholder="pus cells" required />
80         </div>
81         <div class="form-outline mb-4 col-6 details">
82             <input name="pcc" class="form-control" placeholder="pus cell clumps" required />
83         </div>
84         <div class="form-outline mb-4 col-6 details">
85             <input name="ba" class="form-control" placeholder="bacteria" required />
86         </div>
87         <div class="form-outline mb-4 col-6 details">
88             <input name="bgr" class="form-control" placeholder="blood glucose random" required />
89         </div>
90         <div class="form-outline mb-4 col-6 details">
91             <input name="bu" class="form-control" placeholder="blood urea" required />
92         </div>
93         <div class="form-outline mb-4 col-6 details">
94             <input name="sc" class="form-control" placeholder="serum creatinine" required />
95         </div>
96         <div class="form-outline mb-4 col-6 details">
97             <input name="htn" class="form-control" placeholder="high blood pressure" required />
98         </div>
```



```
62
63         <input name="ago" 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">
```

```
templates > dashboard.html > html > body.border-0 > div.w-75.mx-auto.bg-white.px-5.py-4 > form > div.form-outline.mb-4.col-6.details > input.form-control

    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="sod" class="form-control" placeholder="sodium" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="pot" class="form-control" placeholder="potassium" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="hemo" class="form-control" placeholder="hemoglobin" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="pcv" class="form-control" placeholder="packet cell volume" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="wc" class="form-control" placeholder="white blood cell count" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="rc" class="form-control" placeholder="red blood cell count" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="htn" class="form-control" placeholder="hypertension" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="dm" class="form-control" placeholder="diabetes mellitus" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="cad" class="form-control" placeholder="coronary artery disease" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="appet" class="form-control" placeholder="appetite" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="pe" class="form-control" placeholder="pedal edema" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="ane" class="form-control" placeholder="anemia" required />
    </div>
```

Ln 69, Col 34 Spaces:4 UTRF HTML ⚡ Go Live ✨ Prettier ⌂

```
templates > dashboard.html > html
    <div class="form-outline mb-4 col-6 details">
        <input name="pe" class="form-control" placeholder="pedal edema" required />
    </div>
    <div class="form-outline mb-4 col-6 details">
        <input name="ane" class="form-control" placeholder="anemia" required />
    </div>

    <!-- Submit button -->
    <div class="text-center"><button type="submit" class="btn btn-primary btn-block mb-4">Submit</button>
    </div>
</form>
</div>

<footer class="footer">CKD<sup class="footerc">@</sup> Predictor with ❤</footer>
</body>
</html>
```

Ln 146, Col 8 Spaces:4 UTRF HTML ⚡ Go Live ✨ Prettier ⌂

Dashboard - CKD Predictor

127.0.0.1:5000/dashboard

Hil yash33 🌟 Dashboard Result Logout

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® Predictor with ❤️

Result.html

File Edit Selection View Go Run Terminal Help

result.html - Local Deployment - Visual Studio Code

```

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

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 href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet">
7      <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css" rel="stylesheet">
8    </head>
9    <title>Test Results - 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 </body>
14 </html>
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37

```

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

File Edit Selection View Go Run Terminal Help

LOCAL DEPLOYMENT

templates > result.html

```
36     width="40"
37     class="d-inline-block align-text-top"
38   />
39   <a class="navbar-brand" href="#">CKD Predictors</a>
40   <button
41     class="navbar-toggler"
42     type="button"
43     data-bs-toggle="collapse"
44     data-bs-target="#navbarNav"
45     aria-controls="navbarNav"
46     aria-expanded="false"
47     aria-label="toggle navigation"
48   >
49     <span class="navbar-toggler-icon"></span>
50   </button>
51   <div class="collapse navbar-collapse d-flex" id="navbarNav">
52     <ul class="navbar-nav">
53       <li class="nav-item">
54         <a class="nav-link active" href="#">Hi! {{session.username}} 🌟</a>
55       </li>
56       <li class="nav-item">
57         <a class="nav-link" href="{{url_for('dashboard')}}>Dashboard</a>
58       </li>
59       <li class="nav-item">
60         <a class="nav-link active" aria-current="page" href="#">Result</a>
61       </li>
62       <li class="nav-item">
63         <a class="nav-link" id="logout" href="{{url_for('logout')}}">Logout</a>
64       </li>
65     </ul>
66   </div>
67 </div>
68 <div class="mx-auto bg-white mt-5 px-1 py-4 text-center">
```

Ln 27, Col 26 Spaces: 2 UTF-8 CRLF HTML ⚡ Go Live ✅ Prettier

File Edit Selection View Go Run Terminal Help

LOCAL DEPLOYMENT

templates > result.html

```
71   </div>
72   </div>
73   <div class="mx-auto bg-white mt-5 px-1 py-4 text-center">
74     Your Test Result is: {% if predict == 1 %}
75     <span style="color: red">Positive</span>
76     {% elif predict == 0 %}
77     <span style="color: green">Negative</span>
78     {% else %}
79     <span>Not Available Now!</span>
80     {% endif %}
81   </div>
82   <br />
83   <h4>Your Past Test Results</h4>
84   {% if table.size == 0 %}
85   <h1>No Records Found</h1>
86   {% else %}
87   <table>
88     <tr>
89       <th>Date</th>
90       <th>Age</th>
91       <th>BP</th>
92       <th>SG</th>
93       <th>AL</th>
94       <th>SU</th>
95       <th>RBC</th>
96       <th>PC</th>
97       <th>PCC</th>
98       <th>BA</th>
99       <th>BGR</th>
100      <th>BU</th>
101      <th>SC</th>
102      <th>SO</th>
103      <th>POT</th>
104      <th>HEMO</th>
105      <th>PCV</th>
106      <th>WC</th>
107      <th>RBC</th>
```

Ln 71, Col 11 Spaces: 2 UTF-8 CRLF HTML ⚡ Go Live ✅ Prettier

File Edit Selection View Go Run Terminal Help

result.html - Local Deployment - Visual Studio Code

EXPLORER templates > result.html

```
<tr>
    <% for row in table %>
        <td>{{row.timestamp}}</td>
        <td>{{row.age}}</td>
        <td>{{row_bp}}</td>
        <td>{{row_sg}}</td>
        <td>{{row_all}}</td>
        <td>{{row_sul}}</td>
        <td>{{row_rnc}}</td>
        <td>{{row_pc}}</td>
        <td>{{row_pcc}}</td>
        <td>{{row_ba}}</td>
        <td>{{row_bgn}}</td>
        <td>{{row_bu}}</td>
        <td>{{row_sc}}</td>
        <td>{{row_sod}}</td>
        <td>{{row_pot}}</td>
        <td>{{row_hemo}}</td>
        <td>{{row_pv}}</td>
        <td>{{row_wc}}</td>
        <td>{{row_rc}}</td>
        <td>{{row_hn}}</td>
        <td>{{row_dm}}</td>
        <td>{{row_cad}}</td>
        <td>{{row_appet}}</td>
        <td>{{row_pe}}</td>
```

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

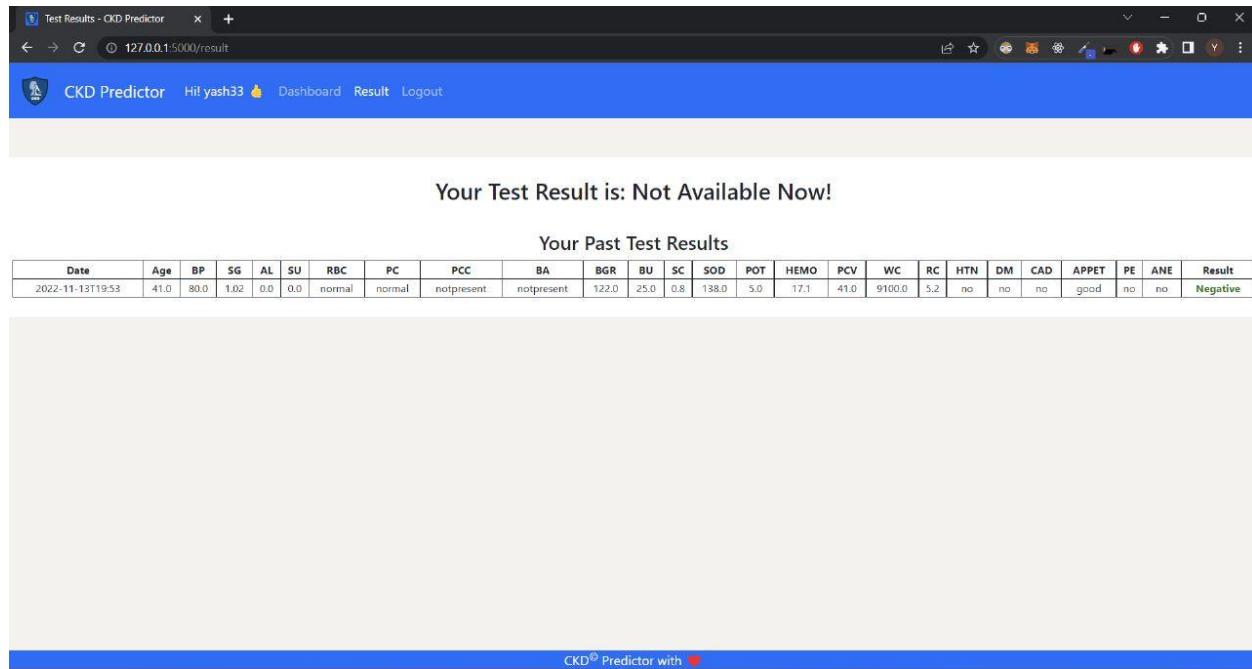
File Edit Selection View Go Run Terminal Help

result.html - Local Deployment - Visual Studio Code

EXPLORER templates > result.html

```
<td class="resultcol">
    <b>
        <% if row.result=='Positive' %>
            <span style="color: red">Positive</span>
        <% else %>
            <span style="color: green">Negative</span>
        <% endif%>
    </b>
</td>
</tr>
<% endfor -%>
</table>
<% endif %>
</div>
</body>
</html>
```

Ln 163, Col 1 Spaces: 2 UTF-8 CRLF HTML Go Live Prettier



7.3 Flask Integration and Deployment

```
File Edit Selection View Go Run Terminal Help app-pklpy - Local Deployment - Visual Studio Code
EXPLORER app-pklpy
LOCAL ... templates
> static dashboard.html
> home.html
> login.html
> register.html
> result.html
> venv app-pklpy
  ckd.pkl
  database.db
requirements.txt
app-pkl.py
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, LoginManager, 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 IOM Cloud account.
11
12
13 app = Flask(__name__)
14 cors(app)
15 bcrypt = Bcrypt(app)
16 app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://sql12564552:nKtfhv756@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)
```

```
class User(db.Model, UserMixin):
    __tablename__ = 'users'
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20), nullable=False, unique=True)
    password = db.Column(db.String(80), nullable=False)
    email = db.Column(db.String(80), nullable=False)

class Prediction(db.Model):
    __tablename__ = 'predict'
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20), nullable=False)
    timestamp = db.Column(db.String(20), nullable=False)
    age = db.Column(db.Float, nullable=False)
    bp = db.Column(db.Float, nullable=False)
    sg = db.Column(db.Float, nullable=False)
    al = db.Column(db.Float, nullable=False)
    su = db.Column(db.Float, nullable=False)
    rbc = db.Column(db.String(20), nullable=False)
    pcc = db.Column(db.String(20), nullable=False)
    ba = db.Column(db.String(20), nullable=False)
    bg = db.Column(db.Float, nullable=False)
    bu = db.Column(db.Float, nullable=False)
    sc = db.Column(db.Float, nullable=False)
    sod = db.Column(db.Float, nullable=False)
    pot = db.Column(db.Float, nullable=False)
    hemo = db.Column(db.Float, nullable=False)
    pcv = db.Column(db.Float, nullable=False)
    wc = db.Column(db.Float, nullable=False)
    rc = db.Column(db.Float, nullable=False)
    htin = db.Column(db.String(20), nullable=False)
    dm = db.Column(db.String(20), nullable=False)
    cad = db.Column(db.String(20), nullable=False)
    appet = db.Column(db.String(20), nullable=False)
    pe = db.Column(db.String(20), nullable=False)
    ane = db.Column(db.String(20), nullable=False)
```

```
@app.route('/')
def home():
    return render_template('home.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == "POST":
        user = User.query.filter_by(username=request.form.get('username')).first()
        if user:
            if bcrypt.check_password_hash(user.password, request.form.get('password')):
                login_user(user)
                session['username'] = request.form.get('username')
                return redirect(url_for('dashboard'))
            else:
                flash("Password is Incorrect!")
        return render_template('login.html')

@app.route('/result', methods=['GET', 'POST'])
@login_required
def result():
    username = session['username']
    table = Prediction.query.filter_by(username=username).order_by(Prediction.timestamp.desc())
    return render_template('result.html', predict=predict, table=table)

@app.route('/dashboard', methods=['GET', 'POST'])
@login_required
def dashboard():
    form = request.form
    if request.method == "POST":
        appet = float(form.get('appet'))
```

```
form = request.form
if request.method == "POST":
    age = float(form.get('age'))
    bp = float(form.get('bp'))
    sg = float(form.get('sg'))
    al= float(form.get('al'))
    su= float(form.get('su'))
    rbc= 0 if form.get('rbc') == 'normal' else 1
    pc= 0 if form.get('pc') == 'normal' else 1
    pcc= 0 if form.get('pcc') == 'notpresent' else 1
    ba= 0 if form.get('ba') == 'notpresent' else 1
    bgr= float(form.get('bgr'))
    bu= float(form.get('bu'))
    sc= float(form.get('sc'))
    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]]
    # ****using the pickle file for prediction*****
    model = joblib.load("ckd.pkl")
    predict = model.predict(X)[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
    )
    db.session.add(new_user)
    db.session.commit()
    username = session['username']
    print(username)
```

```
form = request.form
if request.method == "POST":
    age = float(form.get('age'))
    bp = float(form.get('bp'))
    sg = float(form.get('sg'))
    al= float(form.get('al'))
    su= float(form.get('su'))
    rbc= 0 if form.get('rbc') == 'normal' else 1
    pc= 0 if form.get('pc') == 'normal' else 1
    pcc= 0 if form.get('pcc') == 'notpresent' else 1
    ba= 0 if form.get('ba') == 'notpresent' else 1
    bgr= float(form.get('bgr'))
    bu= float(form.get('bu'))
    sc= float(form.get('sc'))
    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]]
    # ****using the pickle file for prediction*****
    model = joblib.load("ckd.pkl")
    predict = model.predict(X)[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
    )
    db.session.add(new_user)
    db.session.commit()
    username = session['username']
    print(username)
```

```

File Edit Selection View Go Run Terminal Help
LOCAL DEPLOYMENT app-pkl.py
static
templates
  dashboard.html
  home.html
  login.html
  register.html
  result.html
venv
app-pkl.py
  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')

@app.route('/logout', methods=['GET', 'POST'])
@login_required
def logout():
    logout_user()
    return redirect(url_for('login'))

@app.route('/register', methods=['GET', 'POST'])
def register():
    form = request.form
    if request.method == "POST":
        existing_user = User.query.filter_by(username=form.get('username')).first()
        if existing_user:
            flash("That username already exists! Please choose a different one.")
        elif form.get('password') != form.get('password_confirmation'):
            flash('The password confirmation does not match!')
        else:
            hashed_password = bcrypt.generate_password_hash(form.get('password'))
            new_user = User(email=form.get('email'), username=form.get('username'), password=hashed_password)
            db.session.add(new_user)
            db.session.commit()
            return redirect(url_for('login'))
    return render_template('register.html')

if __name__ == "__main__":
    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

```

File Edit Selection View Go Run Terminal Help
IBM DEPLOYMENT app-ibm.py
static
templates
venv
app-ibm.py
  from flask import Flask, render_template, url_for, redirect, request, flash, session
  from flask_sqlalchemy import SQLAlchemy
  from flask_login import UserMixin, login_user, loginManager, login_required, logout_user
  from flask_bcrypt import Bcrypt
  from flask_cors import CORS
  import requests

  # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
  API_KEY = "hM1gdcKmIOfzugulqjmZQcwDxrXdl1hWih9RJ"
  token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API_KEY,
  "grant_type": "urn:ibm:params:oauth:grant-type:apikey"})
  mltoken = token_response.json()["access_token"]

  header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

  app = Flask(__name__)
  CORS(app)
  bcrypt = Bcrypt(app)
  app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://sql12564552:nkk1fhv7se@sql12.freemysqlhosting.net/sql12564552'
  app.config['SECRET_KEY'] = 'thisisasecretkey'
  db = SQLAlchemy(app)
  app.app_context().push()

  predict = None

  login_manager = loginManager()
  login_manager.init_app(app)
  login_manager.login_view = 'login'

  @login_manager.user_loader
  def load_user(user_id):
      return User.query.get(int(user_id))

```

```
    age= float(form.get('age'))
    bp= float(form.get('bp'))
    hemoglobin= float(form.get('hemoglobin'))
    pcv= float(form.get('pcv'))
    wc= float(form.get('wc'))
    rc= float(form.get('rc'))
    httn= 0 if form.get('httn') == '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, hemoglobin, pcv, wc, rc, httn, dm, cad, appet, pe, ane)
    X = [[age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemoglobin, pcv, wc, rc, httn, 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", "hemoglobin", "pcv", "wc", "rc", "httn", "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_id=v4', 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()
    new_user.username=form.get('username'),
    new_user.timestamp = form.get('timestamp'),
    new_user.age = age,
    new_user.bp = bp,
    new_user.sg = sg,
    new_user.al = al,
    new_user.su = su,
    new_user.rbc= form.get('rbc'),
    new_user.pc= form.get('pc'),
    new_user.pcc= form.get('pcc'),
    new_user.ba= form.get('ba'),
    new_user.bgr= bgr,
    new_user.bu= bu,
    new_user.sc= sc,
    new_user.sod= sod,
    new_user.pot= pot,
    new_user.hemoglobin= hemoglobin,
    new_user.pcv= pcv,
    new_user.wc= wc,
    new_user.rc= rc,
    new_user.httn= httn,
    new_user.dm= dm,
    new_user.cad= cad,
    new_user.appet= appet,
    new_user.pe= pe,
    new_user.ane= ane,
    new_user.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')
```

Ln 155, Col 32 Spaces: 4 UTF-8 CRLF Python Go Live Prettier

```
    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,
    hemoglobin= HEMOGLOBIN,
    pcv= PCV,
    wc= WC,
    rc= RC,
    httn= HTTN,
    dm= DM,
    cad= CAD,
    appet= APPET,
    pe= PE,
    ane= 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')
```

Ln 150, Col 20 Spaces: 4 UTF-8 CRLF Python Go Live Prettier

IBM deployed ipynb file

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: a few seconds ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

In [1]:

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='ATobIZPTPiMdyZTqXqk6L2JmrjQXE15AQMPtqSeG',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'earlydetectionofchronickidneydiseasenotdelete-pr-257txfmjjystw'
object_key = 'chronickidneydisease.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df = pd.read_csv(body)
df.head()
```

Out[1]:

	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	60.0	70.0	1.020	1.0	0.0	normal	normal	notpresent	notpresent	...	32	6700	5.0	yes	yes	no	good	no	no	ckd

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 2 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

IBM -Deployment

In [41]:

```
!pip install -U ibm-watson-machine-learning
```

Requirement already satisfied: ibm-watson-machine-learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.8.9)
Requirement already satisfied: lmndon in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.3.3)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.3.4)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (4.8.2)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (21.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2022.9.24)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.26.0)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.26.7)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (2021.3)

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 4 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

```
In [42]: from ibm_watson_machine_learning import APIClient
import json
```

Authenticate and set space

```
In [43]: wml_credentials = {
    "apikey": "HmLTgdkCkMfIQFtugTulqjmTZQcwx07rXdl1Wiw92RJ",
    "url": "https://us-south.ml.cloud.ibm.com"
}
```

```
In [44]: wml_client = APIClient(wml_credentials)
```

```
In [46]: wml_client.spaces.list()
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
```

ID	NAME	CREATED
7fba65c6-6ecf-429a-911d-5b16e4848fde	PNT2022TMID21262 CKD Prediction Deploy space	2022-11-11T11:21:30.601Z
dc20d1ad-dff8-4bab-85ef-d7735feea6ad	ckd-deploy-space	2022-10-29T15:22:08.943Z
80d66093-63cb-4281-b761-4fd15d460aca	iris-deploy-space	2022-10-25T12:15:03.865Z

```
In [47]: SPACE_ID= "7fba65c6-6ecf-429a-911d-5b16e4848fde"
```

```
In [48]: wml_client.set.default_space(SPACE_ID)
Out[48]: 'SUCCESS'
```

```
In [49]: wml_client.software_specifications.list(100)
```

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 5 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

```
In [49]: wml_client.software_specifications.list(100)
```

NAME	ASSET_ID	TYPE
default_py3_6	0062b8c9-8b7d-44a0-a9b9-46c416adcb9	base
kernel-spark3.2-scala2.12	020d69ce-7ac1-5e68-ac1a-31189867356a	base
pytorch-onnx_1.3-py3.7-edt	069ea13a-3346-5748-b513-49120e15d288	base
scikit-learn_0.20-py3.6	09c5a1d0-9c1e-4473-a344-eb7b665ff687	base
spark-mllib_3.0-scala_2.12	09fc4ff0-90a7-5899-b9ed-1ef3f48abede	base
pytorch-onnx_rt22.1-py3.9	0b848dd4-e681-5599-be41-b5f6fcccd471	base
a1-function_0.1-py3.6	0cd0bf1e-5376-4fad-92dd-d3bd69aa9bda	base
shiny-r3.6	066e79df-875e-4f24-8ae9-62dc21483b0	base
tensorflow_2.4-py3.7-horovod	1092590a-307e-563d-9b62-4eb7d6dbf22	base
pytorch_1.1-py3.6	10a12d6-6130-4cc4-8392-3e922c096a92	base
tensorflow_1.15-py3.6-ddl	111e41b3-de2d-5422-a4d6-bf776828c4b7	base
autoai-kb_rt22.2-py3.10	125b6d9a-5b1f-5e8d-972a-b251688ccf40	base
runtime-22.1-py3.9	12b83a17-24d4-5082-900f-0ab31ff03cb	base
scikit-learn_0.22-py3.6	154010fa-5b3b-4ac1-82af-4d5ee5abbcc85	base
default_r3.6	1b70aeac3-ab34-4b87-8aa0-a4a3c8296a36	base
pytorch-onnx_1.3-py3.6	1bc6029a-cc97-56da-b8e0-39c3880dbbe7	base
kernel-spark3.3-r3.6	1c9e5454-f216-59d4-a29e-47445cf5988	base
pytorch-onnx_rt22.1-py3.9-edt	1d3c2186-7ads-5b59-8b8c-9d08080de37f	base
tensorflow_2.1-py3.6	1eb25b84-66ed-5dde-b6a5-3fbdf166566	base
spark-mllib_3.2	20047f72-0a9f-58c7-9ff5-a77b012eb8f5	base
tensorflow_2.4-py3.8-horovod	217c16f6-178f-56bf-824a-b19f20564c49	base
runtime-22.1-py3.9-cuda	26215f05-08c3-5a41-a1b0-d46630c6e658	base
do_py3.8	295adb5-9eff-547e-9bf4-92ae3563e720	base
autoai-ts_3.8-py3.8	2aa0c932-798f-5ae9-abd6-15e0c2402fb5	base
tensorflow_1.15-py3.6	2b73a275-7cbf-42b0-a912-eae7f436e0b	base
kernel-spark3.3-py3.9	2b7961e2-e3b1-5a8b-a491-482c8368839	base
pytorch_1.2-py3.6	2c8ef57d-2687-4b7d-acce-01ff94976dac1	base
spark-mllib_2.3	2e51f700-bca0-4b0d-88dc-5c6791338875	base
pytorch-onnx_1.1-py3.6-edt	32983cea-3f32-4400-8965-dde874a0d67	base
spark-mllib_3.0-py37	36507ebe-8770-5bba-ab2a-eafe787600e9	base
spark-mllib_2.4	390d21f8-e58b-4fa4-9c55-d7cedad621326	base
autoai-ts_rt22.2-py3.10	396b2e83-0953-5b86-9a55-7ce1628a406f	base
xgboost_0.87-py3.6	39e313a4-5f38-41dc-aed4-609233c803056a	base

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 6 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Saving and Deploying the Model

```
In [50]: import sklearn
sklearn._version_
Out[50]: '1.0.2'

In [51]: # Set Python Version
MODEL_NAME = 'PNT2022TMID21262 CKD Prediction'
DEPLOYMENT_NAME = 'PNT2022TMID21262 CKD Prediction'
DEMO_MODEL = model

In [53]: # Set Python Version
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')

In [54]: # Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}

In [56]: #Save model
model_details = wml_client.repository.store_model(
    model=DEMO_MODEL,
    meta_props=model_props,
    training_data=X_train,
    training_target=y_train
)

In [57]: model_details
```

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 7 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [57]: model_details
Out[57]: {'entity': {'hybrid_pipeline_software_specs': [], 'label_column': 'classification', 'schemas': {'input': [{ 'fields': [{ 'name': 'age', 'type': 'float64'}, { 'name': 'bp', 'type': 'float64'}, { 'name': 'sg', 'type': 'float64'}, { 'name': 'al', 'type': 'float64'}, { 'name': 'su', 'type': 'float64'}, { 'name': 'rbc', 'type': 'int64'}, { 'name': 'pc', 'type': 'int64'}, { 'name': 'pcc', 'type': 'int64'}, { 'name': 'ba', 'type': 'int64'}, { 'name': 'bgr', 'type': 'float64'}, { 'name': 'bu', 'type': 'float64'}, { 'name': 'sc', 'type': 'float64'}, { 'name': 'sod', 'type': 'float64'}, { 'name': 'pov', 'type': 'float64'}, { 'name': 'hemo', 'type': 'float64'}, { 'name': 'pvc', 'type': 'float64'}, { 'name': 'wc', 'type': 'float64'}, { 'name': 'rc', 'type': 'float64'}, { 'name': 'htn', 'type': 'int64'}, { 'name': 'dm', 'type': 'int64'}, { 'name': 'cad', 'type': 'int64'}, { 'name': 'apet', 'type': 'int64'}, { 'name': 'pe', 'type': 'int64'}, { 'name': 'ane', 'type': 'int64'}], 'id': 1, 'type': 'struct'}], 'output': []}, 'software_spec': {'id': '12b83aa17-24d8-5082-900f-0ab31fbfd3cb', 'name': 'runtime-22.1-py3.9'}, 'type': 'scikit-learn_1.0'}, 'metadata': {'created_at': '2022-11-11T11:31:24.426Z', 'id': '459f059d-6677-4692-9b3a-947b408aa8fd', 'modified_at': '2022-11-11T11:31:27.899Z', 'version': '2022.11.11T11:31:24.426Z'}}
```

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 8 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [58]: model_id = wml_client.repository.get_model_id(model_details)
model_id
Out[58]: '459f059d-6677-4692-9b3a-947b408aa8fd'

In [59]: # Set meta
deployment_props = [
    wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
]

In [60]: # Deploy
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)

#####
# Synchronous deployment creation for uid: '459f059d-6677-4692-9b3a-947b408aa8fd' started
#####

initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.
```

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 8 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [60]: # Deploy
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)

#####
# Synchronous deployment creation for uid: '459f059d-6677-4692-9b3a-947b408aa8fd' started
#####

initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.
ready

-----
Successfully finished deployment creation, deployment_uid='4bb20f2e-e060-412e-875e-a336e199f1aa'

In [ ]:
```

7.4 Database (mysql)

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Query 1

```

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

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

	id	username	password	email
▶	3	Ajay	\$2b\$12\$ISLQuO.v9KnZYz0zBNk'sOuqa0k/JQ...	abc@gmail.com
4	Barath	\$2b\$12\$7QnyMpQfFV3fTbgnoChN.13dC0...	xyz@gmail.com	
5	Harish	\$2b\$12\$6gabCsWSN/dhDWs0IGJLe6Zu7v54...	def@gmail.com	
6	yash33	\$2b\$12\$HtAOQfANzvOWymZP0KwOgSgSgv...	yash33@example.com	
7	sanTroo	\$2b\$12\$cgCIL0L.WPmsHwyyZPU-4sVmAq...	mno@gmail.com	
8	ABCD	\$2b\$12\$K1nlt77KqeRLrfy7h0JUdg32l/PKE...	abcd@gmail.com	
9	bala	\$2b\$12\$Un5c5g1jQz3hG6dWhe8uyj2.3GDjE...	bbmbala2002@gmail.com	
10	191223	\$2b\$12\$3q1wIMRs91IQ6WY3DDKeotHz2en/k...	191223@pgtech.ac.in	

users 1

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

File Edit View Query Database Server Tools Scripting Help

Query 1

```

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

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

	temp	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

Output:

Action Output

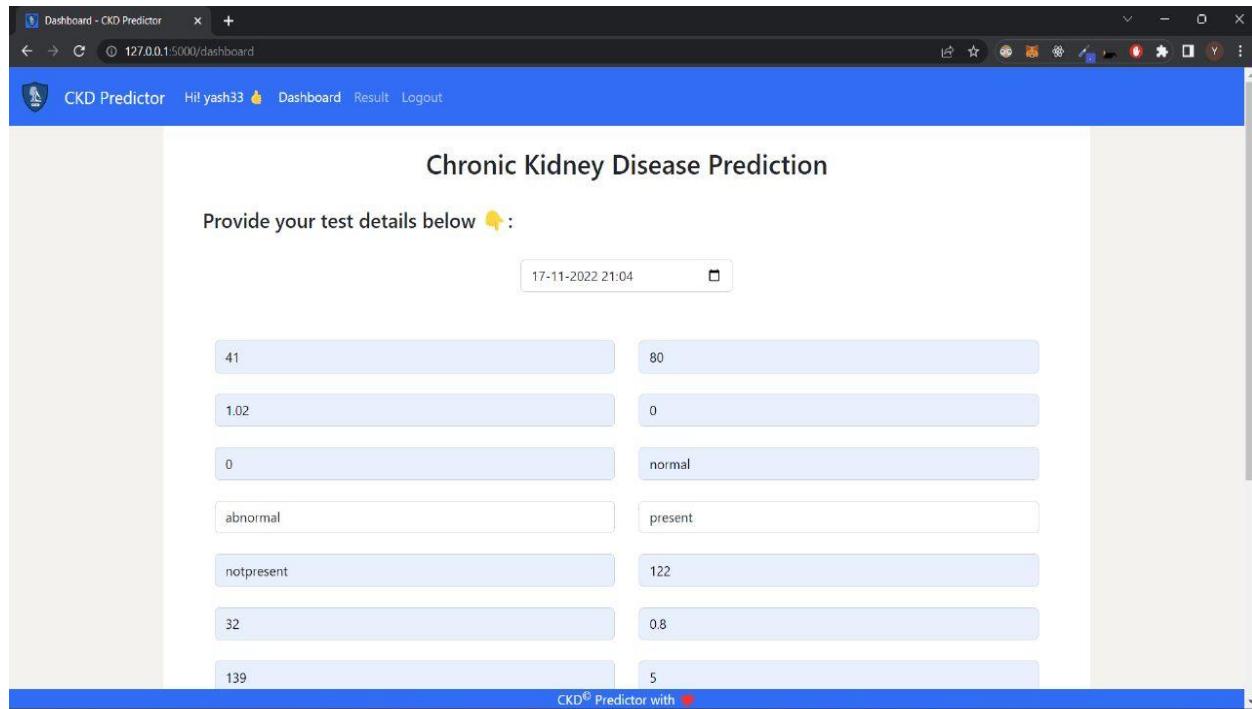
#	Time	Action	Message	Duration / Fetch
2	20:55:05	use sql12564552	0 row(s) affected	0.078 sec
3	20:55:08	select * from users LIMIT 0, 1000	12 row(s) returned	0.078 sec / 0.000 sec
4	20:55:47	select * from predict LIMIT 0, 1000	19 row(s) returned	0.078 sec / 0.000 sec

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	https://www.kaggle.com/	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	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 http://127.0.0.1:5000/predict	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 result as vs code flask integration				

Sample tests:



The screenshot shows a web browser window titled "Dashboard - CKD Predictor". The URL is "127.0.0.1:5000/dashboard". The top navigation bar includes a user icon, the title "CKD Predictor", and links for "Hil yash33", "Dashboard", "Result", and "Logout". The main content area is titled "Chronic Kidney Disease Prediction" and contains a form with the following inputs:

Provide your test details below :	
41	80
1.02	0
0	normal
abnormal	present
notpresent	122
32	0.8
139	5

At the bottom of the page, there is a footer bar with the text "CKD® Predictor with ❤️".

The screenshot shows a web browser window titled "Test Results - CKD Predictor". The URL is "127.0.0.1:5000/dashboard". The top navigation bar includes "CKD Predictor", "Hi! yash33", "Dashboard", "Result", and "Logout". The main content area displays the message "Your Test Result is: Positive" in red. Below it, a table titled "Your Past Test Results" shows two rows of historical data. The last row indicates a "Positive" result. At the bottom, a blue footer bar says "CKD® Predictor with ❤️".

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 (UAT)

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Early Detection of Chronic Kidney Disease] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	1	1	1	6
Duplicate	4	0	2	0	6
External	2	2	0	1	5
Fixed	1	1	1	1	4
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	10	4	4	3	21

3. Test Case Analysis

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

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

9. RESULTS

9.1 Performance Metrics (Random Forest Classifier)

jupyter PNT2022TMID21262 CKD Prediction IBM Last Checkpoint: 33 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

In [36]: #importing metrics to check the performance of the model
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix

In [37]: #displaying accuracy score of testing data
accuracy_score(y_test,y_pred) #higher accuracy
Out[37]: 0.9916666666666667

In [38]: #confusion matrix display
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True)
Out[38]: <AxesSubplot:>

In [40]: #classification report displaying the other performance metrics
print(classification_report(y_test,y_pred));
precision recall f1-score support

```
In [40]: #classification report displaying the other performance metrics
print(classification_report(y_test,y_pred));
          precision    recall  f1-score   support
0           1.00     0.98     0.99      42
1           0.99     1.00     0.99      78
accuracy                           0.99      120
macro avg       0.99     0.99     0.99      120
weighted avg    0.99     0.99     0.99      120
```

10. ADVANTAGES & DISADVANTAGES

Advantages:

Chronic kidney disease (CKD) is one of the most critical health problems due to its increasing prevalence. It is also known as chronic renal disease which is a condition characterized by a gradual loss of kidney function over time.

A better testing method which could possibly detect CKD in the early stages would be much more useful using machine learning algorithm

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

Disadvantages:

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

- Chances of prediction to be wrong for least number of time which can cause problems
- Vast feature in dataset on discovery of time for the disease making the model inefficient to keep up the metrics
- Since 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 learning model 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, and blood 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 be time 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 existing dataset 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-26228-1660021841>

DEMO LINK :

<https://drive.google.com/file/d/1IEqB8TNz4rScn0TqT4uNr3NX53lo1vdQ/view?usp=sharing>