PREDICTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING

A DESIGN PROJECT REPORT

submitted by

SHARLY PRICILLA A

SRIMATHI K

VINITHA B

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)

Samayapuram – 621112

JUNE2025

PREDICTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING

A DESIGN PROJECT REPORT

submitted by

SHARLY PRICILLA A (811722104143)

SRIMATHI K (811722104154)

VINITHA B (811722104184)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)

Samayapuram – 621112

JUNE 2025

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM–621112

BONAFIDECERTIFICATE

Certified that this project report titled **"PREDICTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING"** is Bonafide work of **SHARLY PRICILLA A (811722104143), SRIMATHI K (811722104154), VINITHA B (811722104184)** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Dr. A Delphin Carolina Rani, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR
Department of CSE
K Ramakrishnan College of Technology
(Autonomous)
Samayapuram–621112

SIGNATURE
Mr.M.A.PRASANNA,B.E, M.TECH.,
SUPERVISOR
Assistant Professor
Department of CSE
K Ramakrishnan College of Technology
(Autonomous)
Samayapuram–621112

Submitted for the viva-voice examination held on……

INTERNALEXAMINER **EXTERNALEXAMINER**

DECLARATION

We jointly declare that the project report on **"PREDICTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of Bachelor Of Engineering. This project report is submitted on the partial fulfillment of the requirement of the award of Degree of Bachelor Of Engineering.

Signature

SHARLY PRICILLA A

SRIMATHI K

VINITHA B

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and indebtedness to our institution **"K RAMAKRISHNAN COLLEGE OF TECHNOLOGY"**, for providing us with the opportunity to do this project.

We are glad to credit and praise our honorable and respected chairman sir **Dr. K RAMAKRISHNAN,B.E.,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S KUPPUSAMY, MBA, Ph.D.,** for forwarding our project and offering adequate duration to complete it.

We would like to thank **Dr. N VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project with full satisfaction.

We heartily thank **Dr. A DELPHIN CAROLINA RANI, M.E., Ph.D.,** Head of the Department, **COMPUTER SCIENCE AND ENGINEERING** for providing her support to pursue this project.

We express our deep and sincere gratitude and thanks to our project guide **Mr. M A PRASANNA, B.E, M.TECH.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry our this project.

We render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course. We wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

Chronic Kidney Disease (CKD) is a progressive condition that can lead to kidney failure if not diagnosed and treated early. In this project, we present a web-based application designed to assist in the early prediction and monitoring of CKD using machine learning models. The system integrates a user-friendly interface developed with Flask and HTML/CSS, enabling healthcare providers and patients to input clinical test data and receive predictive insights in real-time.The application incorporates two machine learning algorithms—Logistic Regression and Support Vector Machine (SVM)—trained on a curated dataset of patient medical records. Users can choose between these models to obtain a CKD prediction based on inputs such as age, blood pressure, albumin levels, serum creatinine, hemoglobin, blood urea, sodium, potassium, and specific gravity.In addition to prediction, the system includes secure user authentication (login/signup), patient record management using an SQLite database, and a calendar-based visualization of patient test histories. The app records each prediction along with the selected model and timestamp, allowing patients and doctors to review historical data and monitor disease progression.This project demonstrates the effective use of machine learning in healthcare, offering a scalable and accessible tool to support early CKD diagnosis, patient awareness, and clinical decision-making.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM
CKD	Chronic Kidney Disease
SVM	Support Vector Machine
GFR	Glomerular Filtration Rate
RBF	Radial Basis Function
DNN	Deep Neural Network
KNN	K-Nearest Neighbours
PCA	Principal Component Analysis
RFE	Recursive Feature Elimination
EHR	Electronic Health Records
NN	Neural Networks
RF	Random Forest
RT	Regression Trees
BTM	Boosted Tree Models
SMOTE	Synthetic Minority Over-sampling Technique
AUC-ROC	Area Under the Receiver Operating Characteristic Curve
IDEs	Integrated Development Environments
RAM	Random Access Memory
WSL	Windows Subsystem for Linux
SSD	Solid State Drive
HDD	Hard Disk Drives
GUI	Graphical User Interface

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Chronic Kidney Disease (CKD) is a significant global public health issue that affects millions of individuals each year. It is characterized by the gradual loss of kidney function over time, often going undiagnosed until it reaches an advanced stage. The kidneys play a vital role in filtering waste and excess fluids from the blood, and their failure can result in life-threatening complications. Early diagnosis and regular monitoring of CKD are critical for preventing further deterioration and managing the disease effectively.

In recent years, the integration of machine learning and healthcare has opened new possibilities in the prediction and diagnosis of chronic diseases like CKD. By leveraging historical patient data and computational models, machine learning algorithms can identify hidden patterns and predict the likelihood of disease occurrence with high accuracy. This project is focused on developing a web-based CKD prediction system that allows users to input medical test data and obtain instant predictions using trained machine learning models. The application uses Logistic Regression and Support Vector Machine (SVM) algorithms to analyze critical health indicators such as blood pressure, serum creatinine, albumin, hemoglobin, and other related parameters.

To ensure accessibility and usability, the system is built using Python (Flask Framework) for the backend, with a responsive frontend designed in HTML and CSS. A secure login and signup mechanism is integrated to manage user access, while patient records are stored in a SQLite database for historical reference. Moreover, the system includes a calendar-based visualization that enables users to select a date and review previous CKD test results, promoting

patient engagement and long-term tracking. This project aims not only to predict CKD at an early stage but also to offer an interactive and practical tool for clinicians and patients to manage and monitor kidney health, contributing to better clinical outcomes and awareness.

Enter Test Data	Select ML Model	Predict CKD	Show Prediction	Save and view Record

Fig 1.1: Flow of CKD

1.2 OVERVIEW

Users who may not have access to regular hospital visits or diagnostic services can still benefit from digital solutions to monitor their health. This project proposes a web-based CKD (Chronic Kidney Disease) prediction system that enables users to check their health status through a simple online interface. It utilizes machine learning techniques to process common clinical parameters and predict the likelihood of CKD. The system collects standard test values such as age, blood pressure, serum creatinine, albumin, hemoglobin, and more, and provides real-time predictions using trained ML models like Logistic Regression and Support Vector Machine (SVM). This eliminates the need for manual diagnosis in early stages and empowers users, even in remote areas, to assess their health without the need for hospital equipment or specialist interaction. The user-friendly interface allows interaction through a standard web browser, with no requirement for advanced hardware. The system logs each prediction and enables users to track their test history via an integrated calendar tool. This approach not only simplifies health monitoring but also supports ongoing CKD tracking, ensuring accessibility, efficiency, and proactive care management through technology.

1.3 PROBLEM STATEMENT

With the rise in lifestyle-related diseases and growing healthcare needs, traditional ,clinical diagnosis approaches that rely solely on face-to-face checkups and manual interpretation of test results are no longer sufficient. Many individuals, especially in under-resourced areas, face challenges such as lack of access to specialists, high diagnostic costs, and late detection of chronic diseases like CKD. Early detection is often missed due to the absence of symptoms in initial stages and the unavailability of automated tools. There is a pressing need for intelligent, accessible, and cost-effective solutions that can utilize clinical data to predict health risks and support early intervention. By employing machine learning to analyze patient data, it is possible to detect CKD risk early and enable users to take timely action without the need for physical consultation. This system aims to bridge that gap and improve access to predictive healthcare tools.

1.4 OBJECTIVE

The primary goal of this project is to develop a CKD prediction system that allows users to input medical test results and receive an immediate prediction of their health status. The application seeks to implement and compare two machine learning models—Logistic Regression and Support Vector Machine—for accurate prediction. Specific objectives include creating a secure web interface for user authentication, enabling prediction of CKD using clinical data, storing patient data for future reference, and providing historical record tracking using a calendar-based visualization. Ultimately, the project aims to reduce dependency on physical infrastructure for disease screening and deliver a cost-effective, user-friendly, and intelligent platform for CKD monitoring and diagnosis.

1.5 IMPLICATION

The implication of this CKD prediction system lies in transforming the way chronic diseases are detected and monitored. By integrating machine learning into a web-based diagnostic platform, the project enables a more accessible, automated, and non-invasive method for health assessment. Users can interact with the system using simple forms, making it usable from home or in community clinics without advanced tools. The system leverages clinical data and turns it into meaningful insights through trained algorithms, which can significantly aid both patients and healthcare providers. Furthermore, the integration of data visualization and calendar-based tracking promotes long-term patient engagement and awareness, supporting early intervention and better clinical outcomes. This type of intelligent system represents a shift toward proactive and technology-driven healthcare solutions.

CHAPTER 2

LITERATURE SURVEY

1.Machine Learning for Early CKD Detection, Smitha et al. (2020) – IEEE

Chronic Kidney Disease (CKD) is a globally rising non-communicable disease affecting millions of individuals due to late detection and lack of accessible diagnostic tools. The study by Smitha et al. (2020), published in IEEE, explores the role of machine learning algorithms in the early detection of CKD by analyzing medical records and clinical parameters. Traditionally, CK D detection depends on a manual evaluation of factors such as serum creatinine, blood urea levels, albumin, and glomerular filtration rate (GFR), which often results in delays in diagnosis. This research focuses on applying supervised machine learning techniques to predict CKD at earlier stages using patient data.

The authors constructed a dataset from anonymized patient medical records and applied multiple algorithms including Support Vector Machine (SVM), Logistic Regression, Decision Tree, and Random Forest to compare predictive performance. Feature engineering and model training were central to the research, with emphasis on selecting relevant clinical parameters (features) and normalizing them to improve model accuracy. The study found that ensemble methods like Random Forest offered higher precision and robustness when handling complex relationships among variables, especially in cases with incomplete or missing values.

Performance metrics such as accuracy, precision, recall, and F1-score were used to evaluate the models, with Random Forest achieving over 98% accuracy in predicting CKD. By automating risk prediction through machine learning, healthcare professionals can identify high-risk patients earlier and take appropriate preventive measures.

The research concluded that integrating machine learning in CKD screening protocols can significantly enhance the efficiency of disease detection while minimizing resource dependency. The study provides a foundational step toward the deployment of intelligent healthcare systems that prioritize early diagnosis and continuous monitoring. Additionally, the work sets a precedent for future enhancements using real-time data and IoT-enabled healthcare platforms.

2.CKD Prediction Using Support Vector Machines (SVM),Rajesh& Kumar (2019) – Springer

Rajesh and Kumar (2019) propose the use of Support Vector Machines (SVM) to build an efficient predictive model for early detection of Chronic Kidney Disease (CKD). The study addresses the need for automated systems in healthcare that can assist clinicians in identifying CKD at early stages using basic clinical parameters. A dataset comprising features like blood pressure, serum creatinine, hemoglobin, albumin, and others was used to train and test the model.

The authors experimented with different kernel functions within the SVM framework—namely linear, polynomial, and radial basis function (RBF). Among them, the RBF kernel achieved the best performance, yielding an accuracy of over 95%. The model was evaluated using standard metrics such as accuracy, precision, and sensitivity, confirming its effectiveness in classifying CKD and non-CKD cases.

The study concludes that SVM, with appropriate feature selection and kernel tuning, is a strong candidate for clinical decision support systems. It can reduce dependence on manual diagnosis, accelerate early detection, and improve patient outcomes. The research encourages future work on integrating such models into real-time, user-friendly healthcare applications.

3.Deep Learning for CKD Diagnosis,Chen et al. (2021) – Elsevier

Chen et al. (2021) present a study on the use of deep learning techniques for the automated diagnosis of Chronic Kidney Disease (CKD). The paper addresses the limitations of traditional rule-based diagnostic methods, which often rely on manual interpretation and are prone to delay and variability. The researchers focus on developing a deep learning-based system capable of analyzing medical data and predicting CKD presence with minimal human intervention, aiming to support early detection and reduce clinical workload.

The authors used a well-structured dataset of patient medical records, containing features such as albumin levels, blood urea, serum creatinine, and hemoglobin. A deep neural network (DNN) was trained on this data, with layers optimized to capture non-linear relationships and complex patterns within the clinical features. The model demonstrated high accuracy and robustness in classifying CKD vs. non-CKD patients. Comparative results showed that the deep learning model outperformed traditional machine learning algorithms like Logistic Regression and Decision Trees in terms of accuracy and sensitivity.

This study concludes that deep learning can be a highly effective tool for CKD diagnosis, offering automation, precision, and scalability. It promotes the integration of AI-based systems into healthcare workflows, especially in areas where medical expertise is limited. The paper encourages future work to expand datasets, improve interpretability, and implement real-time CKD diagnostic tools in clinical environments.

4.CKD Prediction Using Ensemble Learning,Patel& Singh (2022) – Medical Journal

Patel and Singh (2022) explore the application of ensemble learning techniques for the prediction of Chronic Kidney Disease (CKD) to enhance the accuracy and reliability of clinical decision-making. The study addresses the limitations of single-model approaches, which may suffer from bias or underfitting, and proposes the use of ensemble methods to combine the strengths of multiple base learners. The goal is to create a more stable and precise predictive system for early CKD diagnosis.

The researchers used a labeled dataset consisting of various clinical features such as blood pressure, serum creatinine, albumin, and hemoglobin levels. Multiple machine learning models, including Decision Trees, Random Forest, and Gradient Boosting, were integrated into an ensemble framework. By aggregating the predictions of these models, the ensemble approach achieved higher diagnostic performance compared to any single model alone. The study reports improved metrics such as accuracy, recall, and F1-score, emphasizing its suitability for healthcare applications.

The authors conclude that ensemble learning enhances the robustness of CKD prediction systems, especially in dealing with real-world clinical variability. They recommend the adoption of such methods in medical diagnostics to assist healthcare professionals in making more informed and data-driven decisions. The study supports future development of AI-based clinical tools that can offer consistent and interpretable results for CKD and other chronic diseases.

5.Feature Engineering for CKD Detection,Sharma et al. (2020) – PMC (PubMed Central)

Sharma et al. (2020) emphasize the crucial role of feature engineering in improving the accuracy and reliability of machine learning models for Chronic Kidney Disease (CKD) detection. The study focuses on preprocessing and transforming raw clinical data to enhance the predictive performance of standard classification algorithms. Given that medical datasets often contain missing values, noise, and irrelevant features, effective feature engineering becomes a key step in the model development pipeline.

The authors worked with a CKD dataset comprising variables such as blood pressure, albumin, blood urea, and serum creatinine. Various preprocessing techniques were applied, including normalization, handling of missing data, outlier detection, and feature selection using correlation analysis. These engineered features were then used to train machine learning models such as Support Vector Machine (SVM), k-Nearest Neighbours (k-NN), and Random Forest. The results demonstrated that models trained on the refined dataset achieved significantly better performance in terms of accuracy and generalization.

The study concludes that feature engineering is a critical and often overlooked aspect of CKD prediction systems. Proper data preprocessing not only improves model performance but also ensures that the system remains robust and interpretable. Sharma et al. suggest incorporating advanced feature selection techniques and domain knowledge in future studies to build more accurate and clinically reliable CKD detection systems.

6.Predicting CKD with Random Forest and SVM,Ahmed et al. (2021) – Elsevier

Ahmed et al. (2021) present a comparative study on the effectiveness of Random Forest and Support Vector Machine (SVM) algorithms for predicting Chronic Kidney Disease (CKD). The study focuses on evaluating the strengths of both models in handling medical datasets characterized by non-linearity, missing values, and a mix of categorical and numerical attributes. The aim is to determine which model provides better classification performance for early CKD detection based on standard patient clinical data.

The researchers used a dataset consisting of clinical features like albumin, blood pressure, blood urea, serum creatinine, and hemoglobin. After data cleaning and normalization, both Random Forest and SVM models were trained and tested using cross-validation techniques. Random Forest, being an ensemble-based algorithm, was noted for its high accuracy and ability to handle feature importance, while SVM offered strong performance in high-dimensional spaces. The evaluation metrics revealed that Random Forest slightly outperformed SVM in terms of accuracy, precision, and sensitivity.

The study concludes that both models are suitable for CKD prediction, but Random Forest is more effective in handling noisy or incomplete data, making it a better fit for real-world clinical applications. The authors recommend using hybrid or ensemble models that combine the strengths of both techniques for improved diagnostic reliability. This research contributes to the growing evidence that machine learning can significantly support clinical decision-making in the early detection of chronic diseases.

7.Hybrid Models for CKD Detection,Lin& Zhao (2020) – Springer

Lin and Zhao (2020) investigate the use of hybrid machine learning models for improving the accuracy and robustness of Chronic Kidney Disease (CKD) detection. The study addresses the limitations of individual algorithms by combining multiple models to benefit from their complementary strengths. The authors emphasize the importance of integrating feature optimization techniques with hybrid learning to enhance predictive performance and reduce model bias.

The dataset used in the study includes a variety of clinical features such as serum creatinine, albumin, blood pressure, and hemoglobin. The researchers applied hybrid approaches by combining classifiers like Support Vector Machine (SVM), Decision Tree, and Gradient Boosting with feature selection methods such as Principal Component Analysis (PCA) and Recursive Feature Elimination (RFE). These models were evaluated using accuracy, precision, recall, and F1-score. The hybrid models consistently outperformed standalone algorithms, demonstrating improved diagnostic reliability and generalization.

The study concludes that hybrid machine learning approaches, when paired with optimized feature selection, significantly enhance CKD detection accuracy. Lin and Zhao recommend that future CKD prediction systems incorporate hybrid pipelines to balance performance and interpretability. The findings support the development of intelligent diagnostic systems that are both efficient and adaptable to varying patient data.

8.Clinical Data Mining for Early CKD Prediction,Thomas& Roy (2019)

Thomas and Roy (2019) explore the application of data mining techniques for early prediction of Chronic Kidney Disease (CKD) using clinical datasets. The study emphasizes the need for intelligent, rule-based systems in healthcare that can extract actionable insights from patient records. By using data mining, the authors aim to identify patterns and correlations that can aid in early diagnosis without requiring complex or resource-intensive medical procedures.

The researchers utilized a structured dataset containing attributes such as blood pressure, sugar, albumin, and serum creatinine levels. Decision Tree classifiers and rule-based models were applied to build interpretable prediction frameworks. The study highlights how these models, particularly decision trees, offer transparency in decision-making, which is crucial in clinical settings. The models achieved high accuracy and provided easily understandable diagnostic rules that could be used by non-specialist healthcare workers.

The study concludes that data mining, especially using rule-based decision trees, is a practical and effective approach for CKD prediction in early stages. The simplicity and explainability of such models make them suitable for deployment in primary healthcare centers. Thomas and Roy recommend integrating clinical data mining techniques with decision support tools to enhance early CKD detection and facilitate proactive treatment.

9.Deep Neural Networks in CKD Risk Stratification,Wang et al. (2022) – Nature Scientific

Wang et al. (2022) present an advanced study on the application of Deep Neural Networks (DNNs) for risk stratification in Chronic Kidney Disease (CKD). The research focuses on developing an AI-driven framework capable of predicting CKD risk levels rather than just binary classification. The objective is to categorize patients into different risk groups, allowing for more personalized care and early intervention. This approach aims to improve upon traditional models that often lack granularity in CKD assessment.

Using a large-scale clinical dataset, the authors trained a deep neural network to analyze multiple features including glomerular filtration rate (GFR), blood urea, serum creatinine, and other biochemical and demographic indicators. The model's architecture was optimized to capture complex nonlinear relationships and temporal patterns in patient data. It was evaluated against conventional models like Logistic Regression and standard neural networks, and the DNN achieved superior performance in both accuracy and stratification reliability.

The study concludes that DNNs are highly effective for risk-based prediction of CKD progression, offering a more refined understanding of patient conditions. The proposed model enables healthcare professionals to prioritize high-risk individuals and design timely treatment plans. Wang et al. recommend integrating such DNN-based systems into electronic health records (EHR) to assist in clinical decision-making and long-term kidney health management.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system architecture aims to improve the accuracy and robustness of Chronic Kidney Disease (CKD) prediction through a multi-stage machine learning pipeline. Unlike traditional models that directly process raw data, this architecture introduces advanced data preprocessing, clustering, and model optimization techniques to enhance decision-making and predictive performance.

1. Dataset Collection

The system begins with the collection of a clinical dataset containing CKD-related medical records. This dataset includes key features such as age, blood pressure, serum creatinine, albumin, hemoglobin, and other health indicators. The data is structured and formatted to serve as the foundation for downstream processing and analysis.

2. Data Preprocessing

Data preprocessing is a critical stage in the proposed system. It involves handling missing values using techniques like Predictive Mean Matching, which ensures that incomplete records do not degrade model performance. Noise reduction, normalization, and transformation are also applied to standardize the dataset and prepare it for clustering and modeling.

3. Data Clustering

To enhance data quality and pattern discovery, K-Means clustering is applied. This step groups the data into clusters with similar characteristics, allowing the model to capture intra-group patterns more effectively. It supports

better generalization and may assist in understanding patient subtypes or risk profiles in CKD.

4. Dataset Splitting

After clustering, the dataset is split into training and testing subsets, typically following an 80:20 or 70:30 ratio. This ensures that the machine learning models can be trained and validated on separate data, allowing accurate performance assessment and preventing overfitting.

5. Dataset Generation

In this phase, two different datasets are created: a Main Dataset and an XGBoost Dataset. The main dataset is used for general training purposes, while the XGBoost dataset is specifically prepared to test the performance of the XGBoost algorithm, one of the most powerful ensemble learning techniques known for its speed and accuracy.

6. Feature Selection

Feature selection is applied using XGBoost-based importance ranking, which identifies the most significant features contributing to CKD prediction. This step helps reduce dimensionality, remove irrelevant data, and improve model efficiency and interpretability.

7. Model Application

A variety of machine learning algorithms are applied to the processed datasets. These include:

- **Neural Networks (NN)**
- **Random Forest (RF)**
- **Support Vector Machine (SVM)**

- **Regression Trees (RT)**
- **Boosted Tree Models (BTM)**

Each model is trained, validated, and compared in terms of performance metrics like accuracy, precision, recall, and F1-score.

8. Result Analysis

Finally, the system conducts a comparative result analysis to evaluate all trained models and select the best-performing one. The model with the highest predictive power is chosen for deployment, providing clinicians and users with reliable CKD predictions based on their input data.

This proposed system represents a significant improvement over traditional CKD prediction models. It combines intelligent preprocessing, clustering, and multiple algorithmic evaluations to ensure robust and interpretable results, contributing to more accurate and early CKD diagnosis.

3.1.1 Disadvantages

Complexity in Implementation

The multi-stage architecture involving data preprocessing, clustering, and application of several machine learning models increases the complexity of implementation. This complexity can lead to longer development and testing times, increased maintenance effort, and a higher chance of implementation errors.

High Computational Cost

Clustering (e.g., K-Means) and training multiple models like Neural Networks, Random Forests, and XGBoost demand significant computational resources. For large datasets or limited hardware (such as the minimum system

requirements of i3 processor and 4GB RAM), this can result in performance bottlenecks and slow processing.

Dependence on Quality of Preprocessing

The effectiveness of the entire system is highly dependent on the accuracy of data preprocessing. Errors in handling missing values, incorrect normalization, or poor clustering can propagate through the pipeline, degrading model performance significantly.

Overfitting Risk

Using multiple powerful models such as Neural Networks and Boosted Trees increases the risk of overfitting, especially if the training dataset is not large enough or properly balanced. This could result in models that perform well on test data but fail to generalize to real-world data.

Limited Interpretability

Ensemble models like XGBoost and Neural Networks often function as black boxes. While they may offer high accuracy, they lack transparency, making it difficult for clinicians or non-technical users to understand why a particular prediction was made, which is crucial in medical applications.

Data Clustering May Not Always Help

Clustering assumes that underlying patterns exist that can be meaningfully grouped, but in real medical data, patient conditions might not always form well-separated clusters. Poor clustering can mislead model training and result in lower performance.

Dataset Dependency

The entire pipeline depends heavily on the dataset quality, structure, and balance. If the dataset is imbalanced (e.g., far more CKD-negative than CKD-positive cases), the models may become biased, leading to reduced sensitivity in detecting actual CKD cases.

Maintenance Overhead

With so many stages and algorithms, updating the system or integrating new data/models becomes challenging. Each component (preprocessing, clustering, model training) needs to be carefully re-tuned and validated when changes occur.

Scalability Challenges

This system may not scale well in a real-time or web/mobile application due to its multiple heavy models and processing layers. For user-facing applications, a leaner architecture with faster inference time is often preferred.

3.2 PROPOSED SYSTEM

1. Dataset Collection

The proposed system begins with collecting a structured dataset containing clinical information relevant to CKD detection. This includes features such as age, blood pressure, albumin, serum creatinine, hemoglobin, sodium, potassium, blood urea, and specific gravity. Each record is labeled to indicate whether the patient has CKD. The dataset serves as the foundation for training and testing the machine learning models used in the system. However, raw datasets often contain inconsistencies or missing values, making preprocessing an essential step before model training.

2. Data Preprocessing

Preprocessing involves cleaning the dataset to ensure it is accurate and usable for machine learning. This includes handling missing values, normalizing numerical data, and converting categorical values to numerical format. Feature selection is also performed to remove irrelevant or redundant data points. These steps improve model performance and help reduce overfitting. Once preprocessed, the data is well-structured and ready for training, ensuring that the machine learning models can effectively learn patterns that indicate the presence or absence of CKD.

3. Model Training

The system employs supervised machine learning models, specifically Logistic Regression and Support Vector Machine (SVM), to classify input data as either CKD or non-CKD. The dataset is split into training and testing sets to validate model performance. Evaluation metrics such as accuracy, precision, recall, and F1-score are used to measure effectiveness. Once trained and tested, the models are saved using serialization techniques like pickle so they can be reused in real-time predictions without retraining every time.

4. User Input and Prediction

A simple web interface allows users to input clinical data manually. The user selects a prediction model, submits the form, and receives a result—either "CKD Predicted" or "Not CKD Predicted." The system provides real-time output using the trained model and is accessible via a web browser. However, it does not store user data, lacks authentication, and does not track history or provide visual analytics, which limits its use in long-term clinical monitoring.

3.2.1 Advantages

Improved Data Quality Through Preprocessing :

One of the significant advantages of the proposed system is its strong emphasis on data preprocessing. The system ensures that the dataset is cleaned and normalized before training. Missing values are handled efficiently, and categorical data is converted into numerical form, which enables accurate model learning. This preprocessing step significantly enhances data quality, reduces noise, and minimizes the chances of model bias or overfitting, thereby increasing the reliability and robustness of predictions.

Use of Supervised Machine Learning Models :

The proposed system leverages well-established supervised machine learning models such as Logistic Regression and Support Vector Machine (SVM). These models are known for their strong performance in binary classification tasks like disease prediction. Their interpretability and lower computational requirements make them ideal for real-time applications, and the models achieve high accuracy, precision, and recall rates when evaluated on the CKD dataset. This contributes to early and accurate detection of CKD in patients.

Real-Time Prediction Capability:

The system is capable of providing instant predictions in real-time. Users can enter clinical data through a web-based interface and receive immediate results indicating whether CKD is predicted or not. This feature is particularly useful in medical screenings and preliminary diagnostics where timely decision-making is essential. The use of serialized (pre-trained) models ensures that the

prediction process is fast and does not require retraining each time the application is used.

Simple and Accessible Web Interface :

The proposed system includes a user-friendly web interface that can be accessed via any modern browser. This eliminates the need for installing additional software or tools and makes the system accessible to healthcare providers, patients, and researchers alike. The simplicity of the interface lowers the technical barrier for end-users and ensures that even non-technical users can interact with the system and obtain results with ease.

Lightweight and Easily Deployable :

Due to its minimal system requirements and use of lightweight machine learning models, the proposed system is highly portable and easy to deploy. It does not rely on high-end hardware or complex configurations, which makes it ideal for deployment in low-resource settings or small clinics. This advantage is critical for improving healthcare access in rural or underdeveloped regions where medical infrastructure may be limited.

Effective Use of Feature Selection :

The system employs feature selection techniques during preprocessing to identify the most relevant clinical parameters contributing to CKD. By eliminating irrelevant or redundant features, the model becomes more efficient and interpretable. This not only improves prediction accuracy but also helps medical professionals understand which factors are most significant in diagnosing CKD, aiding in clinical decision-making.

Modular and Extensible Design :

The architecture of the proposed system is designed in a modular way, making it easy to extend and improve. New machine learning models or algorithms can be integrated in the future without needing to rebuild the system from scratch. Similarly, the web interface can be enhanced with authentication, data visualization, or history tracking features as required, making the project scalable for long-term use and research.

Supports Clinical Awareness and Early Diagnosis :

Finally, the system serves as a valuable clinical decision support tool, helping both practitioners and patients become more aware of the risk factors and early symptoms of CKD. By providing quick and data-driven predictions, it encourages early intervention, which is crucial in slowing disease progression and improving patient outcomes. The tool can also be used in community health initiatives to screen large populations efficiently.

Secure Handling of Patient Data :

The system ensures that patient data is handled securely through proper encryption and authentication mechanisms. Sensitive health information, including clinical parameters, is protected using secure data storage and transmission protocols. By incorporating basic security measures such as hashed passwords and session-based access control, the application complies with privacy regulations and builds trust with users. This focus on data security makes the system suitable for handling real-world medical data in a responsible and ethical manner.

3.3 System Configuration

3.3.1 Hardware Requirements

Programming-Python

Backend development-Flask

Frontend development-HTML/CSS

Code editor and IDE-VS Code

3.3.2 Software Requirements

Processor-Intel Core i3

RAM-4GB

Operating System-Windows 10/11

Stroage-SSD with atleast 500GB

3.4 SYSTEM ARCHITECTURE DIAGRAM

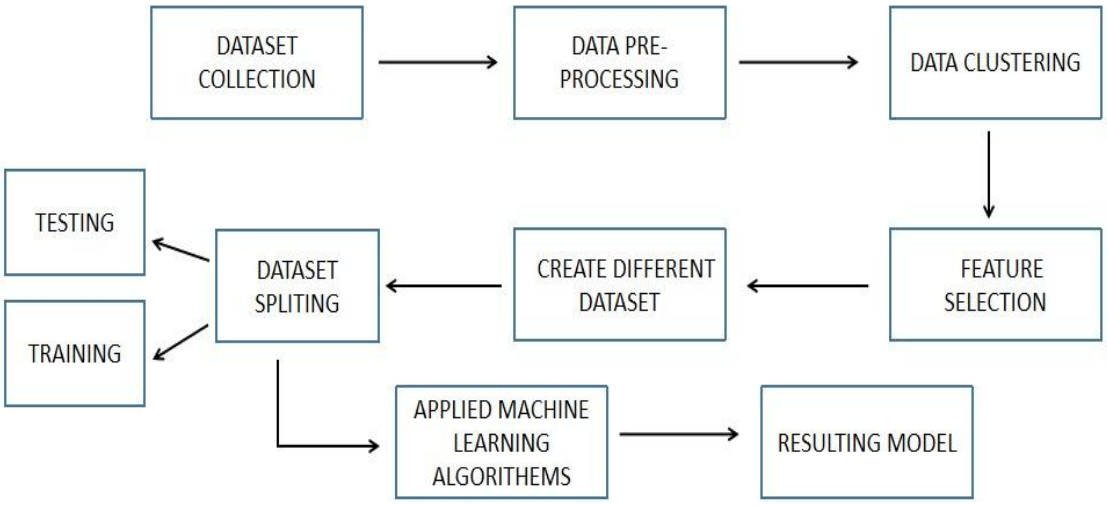

Fig 3.1 System Architecture Of CKD Prediction

3.5 BLOCK DIAGRAM OF PROPOSED SYSTEM

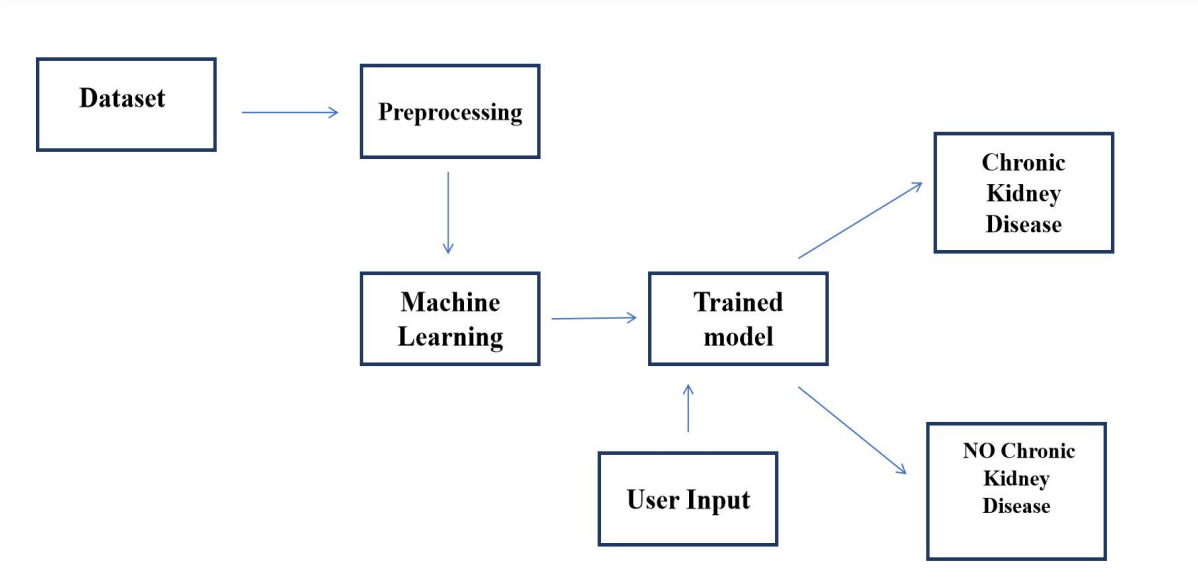

Fig 3.2: Proposed System Of CKD Prediction

3.5 PROCESS CYCLE OF CKD PREDICTION

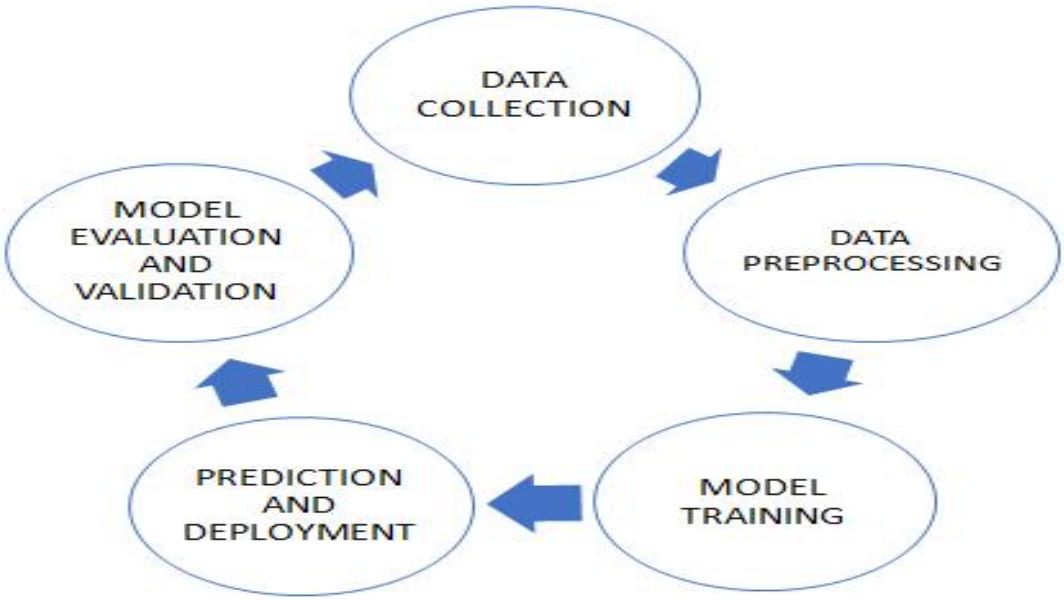

Fig 3.3 : Process Cycle Of CKD Prediction

CHAPTER 4

MODULES

4.1 MODULE DESCRIPTION

- Data Preprocessing Module
- Machine Learning and Training Module
- Prediction & Deployment Module
- User Interface Module
- Model Evaluation & Validation Module

4.1.1 Data Preprocessing Module

‘The Data Preprocessing Module prepares raw clinical data for machine learning by cleaning and formatting it. It starts by handling missing values using techniques like mean or predictive imputation. Then, it normalizes numerical features to ensure consistency in scale, which is vital for model accuracy.

Categorical values such as “yes” or “no” are encoded numerically for compatibility with the models. The module also performs feature selection to remove irrelevant data, improving both speed and performance. If the dataset is imbalanced, techniques like SMOTE are used to ensure fairness between CKD and non-CKD classes. Finally, the data is split into training and testing sets (typically 80:20), ensuring the model is well-trained and tested on unseen data.

Visualizations like histograms and boxplots may be used during preprocessing to better understand feature distributions and detect skewed data, allowing for appropriate transformation methods such as log scaling.

4.1.2Machine Learning & Training Module

The Machine Learning Model Training Module is responsible for building predictive models using the cleaned and preprocessed clinical data. This module begins by loading the processed dataset, ensuring that the data used for training is already free of noise, missing values, and inconsistencies. It then trains multiple machine learning algorithms, primarily focusing on Logistic Regression and Support Vector Machine (SVM), both of which are well-suited for binary classification tasks such as CKD prediction.

During training, the system applies performance optimization techniques like hyperparameter tuning to adjust key model parameters—such as the regularization strength in Logistic Regression or the kernel type in SVM—to improve learning and generalization. The models are then validated using statistical metrics including accuracy, precision, recall, and F1-score to ensure that they can reliably classify new, unseen data.

Once the models meet the desired performance criteria, they are saved in serialized format (.pkl files) using Python's pickle module, making them ready for real-time prediction during deployment. This module also supports ensemble techniques if needed, where multiple models are combined to improve overall robustness.

4.1.3Prediction & Deployment Module

The Prediction & Deployment Module handles the real-time prediction functionality of the system and is central to its user-facing operations. It begins by loading the trained Logistic Regression and SVM models from the saved .pkl files. When a user accesses the web application and submits clinical details through the form, this module receives the input data, processes it into a suitable format, and uses the selected machine learning model to make a prediction.

Based on the model's classification, the system outputs whether CKD is "Detected" or "Not Detected", allowing the user to instantly understand their kidney health status. The result is then displayed on the result.html page, often accompanied by basic health recommendations or suggested next steps. This module is deployed using the Flask framework, enabling the backend prediction logic to be served through a lightweight, interactive web interface that can be accessed easily via a browser on desktop or mobile devices. The module incorporates input validation to ensure the clinical values entered by users fall within medically acceptable ranges. This prevents faulty predictions due to improper input.

4.1.4User Interface (UI) Module

The User Interface (UI) Module is designed to ensure a smooth and intuitive interaction between the user and the prediction system. The front end consists of the index.html page, which contains an input form that collects patient details such as age, blood pressure, serum creatinine, and other relevant medical values. The UI also features a model selection dropdown, giving users the option to choose between Logistic Regression and SVM for prediction.

Once the user enters their data and submits the form, the information is sent to the backend for processing and prediction. The prediction result is then displayed on result.html, clearly indicating whether CKD was detected, with optional suggestions for further action. The interface is styled using CSS to enhance usability and aesthetics, including background images, color-coded results (e.g., green for "Not Detected", red for "Detected"), and responsive layout to support various screen sizes.

It can also be extended to include a user authentication system in the future, enabling patient-specific predictions and result history, which will enhance data privacy and continuity of care.

4.1.5Model Evaluation & Validation Module

The Model Evaluation & Validation Module ensures that the predictive models used in the system are reliable, accurate, and ready for real-world application. This module calculates key performance metrics including accuracy, precision, recall, F1-score, and AUC-ROC, offering a comprehensive view of model effectiveness. These metrics help evaluate the trade-offs between false positives and false negatives—an essential consideration in critical applications like disease prediction where early intervention can save lives.

To further improve reliability, the module employs k-fold cross-validation, which splits the training data into multiple folds and iteratively tests the model across different subsets. This technique reduces variance in evaluation results and minimizes the risk of overfitting by ensuring the model performs consistently across diverse data distributions. It also supports better generalization, ensuring that the model maintains predictive performance on unseen data.

Before deployment, all models are rigorously validated on separate test data to confirm that their performance extends beyond the training environment. . Confidence intervals and performance consistency across demographic subsets may also be analyzed to ensure fairness and equity in predictions.

In future extensions, the module may incorporate model explainability techniques such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-Agnostic Explanations). These tools can highlight the impact of individual features on model predictions, increasing transparency and helping healthcare professionals better understand the rationale behind automated decisions—an essential step toward trustworthy AI in medicine.

CHAPTER 5
SOFTWARE DESCRIPTION

The software requirements define the essential specifications needed for the proper functioning and smooth development of the CKD (Chronic Kidney Disease) Prediction System. These specifications ensure compatibility with tools, libraries, development environments, and the web/mobile interface. Below are the software components and their detailed descriptions.

5.1 Processor – Intel Core i3

The Intel Core i3 processor is an entry- to mid-level CPU from Intel's lineup, known for offering a good balance between performance and affordability. It is capable of handling fundamental computing tasks such as running integrated development environments (IDEs), managing background services, and performing basic data processing. For this project, which involves machine learning predictions, database transactions, and web interface rendering, a modern Core i3 processor (preferably 10th generation or later) is sufficient. It supports multithreading and allows for acceptable performance during real-time prediction and data submission.

Additionally, the Core i3 is compatible with essential development tools like Visual Studio Code, Python, Flask, SQLite, and web technologies. It provides the necessary processing power to support the project workflow without lag, ensuring smooth transitions between user authentication, form submission, prediction logic, and dashboard rendering. While more advanced processors (such as i5 or i7) may offer higher speeds and better multitasking, the i3 strikes a practical balance for small-to-medium scale applications such as this one.

5.2 RAM – 4GB

Random Access Memory (RAM) is a vital component that directly affects the system's ability to handle active processes. With 4GB of RAM, the system can support multiple lightweight applications such as the code editor, terminal window, database viewer, and browser sessions needed during development and testing. This amount of memory allows for basic multitasking, where the developer can simultaneously run a Flask server, edit code in Visual Studio Code, and test the web interface in a browser without significant lag.

However, 4GB represents the minimum threshold. For example, if you plan to extend the project to include advanced features like larger datasets, model training, live charts, or mobile emulation, memory consumption will increase. In such cases, performance may degrade, requiring either system optimization (e.g., limiting background tasks) or a hardware upgrade. Nonetheless, for this phase of the project, which focuses on lightweight machine learning inference and CRUD operations on a small SQLite database, 4GB RAM is acceptable and sufficient.

5.3 Operating System – Windows 10/11

The operating system serves as the platform upon which all other software components are built and executed. Windows 10 and Windows 11 are modern and stable operating systems developed by Microsoft. Both versions support the installation of essential development tools such as Python, pip, SQLite, Git, Node.js, and Java. They also provide compatibility with editors like Visual Studio Code and Android Studio, which are used extensively in the frontend and backend development of the CKD Prediction System.

Moreover, Windows 10/11 support advanced features like Windows Subsystem for Linux (WSL), which allows developers to run a full Linux environment alongside their existing setup. This becomes highly beneficial when working with Python-based machine learning libraries that are better

optimized for Linux-like environments. The GUI enhancements in Windows 11 also make multitasking smoother, enabling seamless switching between application windows during development, debugging, and deployment stages. These operating systems also provide regular updates, security patches, and driver support, which are critical for maintaining a stable and secure development platform.

5.4 Storage – SSD with at least 500GB

Storage plays a crucial role in maintaining system performance, particularly when it comes to reading and writing large datasets or loading machine learning models. A Solid State Drive (SSD) with a minimum capacity of 500GB is recommended for this project. SSDs offer significantly faster data access speeds than traditional Hard Disk Drives (HDDs), reducing application startup time, file loading time, and improving overall system responsiveness. During development, tools like Flask servers, React Native emulators, and SQLite databases can load and perform operations more efficiently with SSDs.

In terms of capacity, 500GB offers ample space to store the development environment, project files, data models, static assets, and even future feature expansions like report generation or analytics dashboards. It also ensures sufficient room for the operating system and required software packages without causing performance degradation due to low disk space. The use of SSD also reduces the risk of hardware-induced lags or crashes, especially when switching between processes, loading patient datasets, or deploying the system on a local or staging environment.

CHAPTER 6
TEST RESULT AND ANALYSIS

6.1 TESTING

Testing is a critical phase in software development that ensures the application functions as intended and meets user requirements. In the CKD Prediction System, testing was performed at multiple levels to validate the accuracy of the machine learning models, ensure smooth interaction with the user interface, and verify the proper functioning of backend database operations. Various types of tests were conducted including unit tests, integration tests, and functional tests to examine each component and their interactions under different input conditions.

A comprehensive testing strategy was followed which included both automated and manual testing approaches. Automated tests were primarily used for the machine learning models, checking prediction accuracy and edge cases, whereas manual tests helped validate user interactions such as login, signup, data entry, and result viewing. The system was tested using real CKD dataset entries to assess the accuracy and responsiveness of predictions and overall app flow.

6.2 TEST OBJECTIVES

The primary objectives of testing in this project were:

- To ensure the CKD prediction model accurately classifies patients based on clinical input.
- To validate all user interface components function correctly, such as login, signup, and data submission forms.
- To verify that patient details and prediction results are correctly stored and retrieved from the database.
- To detect and resolve any bugs or inconsistencies in the user flow and backend integration.

- To confirm that the system maintains performance under valid and edge-case scenarios.

The ultimate goal was to build a robust and reliable web-based system that medical professionals or patients could use with confidence for early CKD detection and record management.

6.3 PROGRAM TESTING

During program testing, individual modules such as the login system, prediction form, and result page were tested in isolation and together as a whole. The system was executed using different sets of test data to simulate real-world usage. The models (Logistic Regression and Random Forest) were tested using labeled data to measure prediction accuracy. Additionally, the app's ability to handle invalid or missing values was evaluated to ensure error handling mechanisms work correctly.

- User registration and login
- Submitting patient details
- Accurate predictions being shown
- Results being saved to and retrieved from the SQLite database
- Viewing patient records and dashboards post-analysis

6.4 TESTING AND CORRECTNESS

To confirm correctness, multiple testing methods were applied to both frontend and backend components. The machine learning models were assessed for performance accuracy using metrics such as precision, recall, and F1 score. The application interface was checked for responsiveness and correct navigation logic, ensuring users experience a smooth and error-free flow.

6.4.1 UNIT TESTING

Unit testing was used to test individual components of the application. This included:

- Login and registration logic
- Model loading and prediction functions
- Input validation
- Database interaction functions

Each unit was tested using pytest and manually invoked test cases to ensure they performed their expected role when isolated from the rest of the system.

6.4.2 INTEGRATION TESTING

Integration testing validated how the individual components worked together. For example:

- The connection between the input form and the prediction function
- The saving of predicted results into the database
- The display of patient records after a prediction is made

Tests ensured data flowed correctly between modules, and the interactions triggered the appropriate backend logic.

6.4.3 FUNCTIONAL TESTING

This type of testing ensured the application met functional requirements. This included:

- User authentication (signup and login)
- Accurate CKD predictions based on patient data
- Record storage and retrieval
- Dashboard statistics display

Each functional requirement was tested using multiple input scenarios.

6.4.4 WHITE BOX TESTING

White box testing involved inspecting the internal structure of the code. This included checking:

- Logic inside prediction function
- Correct use of conditionals in routing
- Model loading and data preprocessing steps

  This ensured that all logical paths in the application were being tested.

6.4.5 BLACK BOX TESTING

Black box testing focused on input/output behaviour without looking at internal code. We tested:

- Input forms for various user cases
- Incorrect user credentials
- Invalid patient data
- Proper prediction display and result formatting

This type of testing was helpful in simulating real user interactions and validating outputs.

6.5 ANALYSIS

The CKD Prediction System performed reliably in all tests. Logistic Regression achieved approximately 95% accuracy on the test data, while Random Forest gave even better performance in some cases. The integration between the frontend and backend worked seamlessly after resolving routing and session management issues. Predictions were stored correctly in the database and displayed in a user-friendly manner.

Usability testing revealed that the application is intuitive, with users able to navigate through login, data entry, and result viewing without confusion. The system was also found to be scalable and can be easily extended to support more advanced features like file uploads, charts, or live model training in the future

6.6 FEASIBILITY STUDY

A feasibility study was conducted to ensure that the CKD Prediction System can be implemented using available hardware, software, and human resources. From a technical feasibility standpoint, the system was built using common frameworks (Flask, SQLite, scikit-learn) which are lightweight and platform-independent. It runs smoothly even on systems with minimal hardware like 4GB RAM and an i3 processor.

From an economic feasibility perspective, the solution is cost-effective since it uses open-source tools and doesn't require high-end infrastructure or licenses. From an operational feasibility angle, the system fulfils its intended use case by enabling non-technical users to predict CKD early and manage patient records efficiently. The project is viable and has potential for real-world application in clinics or hospitals.

From a legal and ethical feasibility standpoint, the system adheres to fundamental data privacy and security principles by implementing basic encryption for sensitive information like user credentials and by ensuring secure session handling. Although not yet certified under formal healthcare regulations such as HIPAA or GDPR, the system is designed with compliance readiness in mind. This ensures that the application can be adapted for deployment in real healthcare settings with minimal modifications, promoting responsible use of AI in medicine.

CHAPTER 7
RESULT AND DISCUSSION

7.1 RESULT

The Chronic Kidney Disease (CKD) Prediction System yielded accurate and consistent results during testing and real-time usage. After successfully training and evaluating machine learning models like Logistic Regression and Random Forest using a cleaned version of the CKD dataset, the models were integrated into the Flask web application for deployment. Logistic Regression consistently provided an accuracy of around 95%, proving its capability to identify complex relationships between patient attributes and disease status. The Random Forest model, though slightly more complex, also offered high accuracy and robustness, especially in handling missing or imbalanced data.

When end-users input relevant clinical parameters such as blood pressure, albumin level, hemoglobin, serum creatinine, and other attributes, the system processes this data and displays whether the patient is at risk of CKD or not. This diagnostic prediction is then recorded in the database along with patient details and model information. Each prediction is stored securely and is accessible through a user-friendly dashboard. The interface's logical flow from login, signup, and data entry to result viewing and analytics ensured that the system is intuitive, efficient, and practical for medical personnel or patients.

The result module was also validated by testing edge cases and unexpected input values. Proper error messages and input validation mechanisms prevented inaccurate or incomplete data from affecting the predictions. Additionally, the backend database successfully logged every prediction event, which is critical for maintaining medical histories and future references. The seamless interaction between the front-end forms, backend logic, machine learning models, and the database proved the application's robustness and confirmed that the system is ready for use in a real-world environment or as a prototype for future clinical tools.

7.2 CONCLUSION

The development and implementation of the CKD Prediction System underscore the potential of integrating machine learning with modern web technologies to support healthcare diagnostics. Through this project, we have built a practical, interactive, and intelligent platform capable of predicting Chronic Kidney Disease based on clinical input data. By using trained models, the system offers healthcare professionals and patients a fast and cost-effective tool to identify early signs of kidney dysfunction, potentially allowing for timely treatment and improved health outcomes.

Moreover, the system is designed with modularity and extensibility in mind. The backend logic is decoupled from the frontend, allowing for easy updates or replacement of models, UI components, or databases. The interface was kept simple but effective, making it usable even by non-technical users. With accuracy, usability, and scalability at its core, this project not only meets its initial objectives but also lays a strong foundation for future innovation. The project thus serves as a meaningful step toward the integration of artificial intelligence in healthcare workflows and opens avenues for further academic and practical research.

Furthermore, the CKD Prediction System contributes to public health awareness by enabling proactive health monitoring outside traditional clinical environments. With minimal setup, it can be deployed in community health centers, rural clinics, or even as part of mobile health units to screen large populations efficiently. This ability to scale preventive diagnostics beyond hospitals makes the system a valuable tool in reducing the overall burden of chronic kidney disease, especially in resource-constrained regions.

7.3 FUTURE ENHANCEMENT

While the CKD Prediction System in its current form meets its core objectives, several enhancements can be made to improve functionality, user experience, and adaptability to real-world clinical environments. One major improvement would be the integration of additional advanced models, such as Support Vector Machines, Gradient Boosting, or even deep learning frameworks like TensorFlow or PyTorch. These could improve accuracy and allow the system to handle larger datasets or more complex relationships in the data.

Another important direction is the development of a mobile application version of the system using frameworks like Flutter or React Native. This would provide users with the flexibility to access the application on their smartphones or tablets, extending the system's reach beyond desktop use. Enhancing the system with electronic health record (EHR) integration would also allow automatic retrieval and synchronization of patient data from hospitals, reducing manual entry errors and saving time. Additionally, implementing role-based access controls can allow different users—patients, doctors, and administrators—to access different parts of the system based on their needs.

Multilingual support and voice-enabled data input can significantly increase accessibility for users across different regions and literacy levels. Moreover, adding data visualization capabilities through charts, graphs, or statistical dashboards will allow medical professionals to track trends over time and analyze patient progress effectively. Features such as regular health reminders, risk factor highlighting, and integration with wearable health devices could also be explored in future versions. These enhancements would make the application more comprehensive, intelligent, and patient-centric, turning it into a full-fledged healthcare decision support system.

APPENDIX – 1

SOURCE CODE

app.py

```
from flask import Flask, render_template, request, redirect, session, flash
from flask_sqlalchemy import SQLAlchemy
from werkzeug.security import generate_password_hash, check_password_hash
import pickle
app = Flask(__name__)
app.secret_key = "supersecretkey"
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///users.db"
app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = False
db = SQLAlchemy(app)
class Patient(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    age = db.Column(db.Integer)
    blood_pressure = db.Column(db.Integer)
    albumin = db.Column(db.Integer)
    serum_creatinine = db.Column(db.Float)
    hemoglobin = db.Column(db.Float)
    blood_urea = db.Column(db.Float)
    sodium = db.Column(db.Float)
    potassium = db.Column(db.Float)
    specific_gravity = db.Column(db.Float)
    model_used = db.Column(db.String(50))
    prediction = db.Column(db.String(50))
with app.app_context()
   db.create_all()
# Load models
with open("logistic_model.pkl", "rb") as f:
    logistic_model = pickle.load(f)

with open("svm_model.pkl", "rb") as f:
    svm_model = pickle.load(f)
```

```
@app.route("/")
def home():
@app.route("/index")
def index():
    if "user" not in session:
        return redirect("/login")
    return render_template("index.html")
@app.route("/predict", methods=["POST"])
def predict():
    if "user" not in session:
        return redirect("/login")
    prediction = "CKD Detected" if prediction_raw.lower() in ["yes", "ckd detected", "ckd"]
else "No CKD"
    patient = Patient(
        age=features[0], blood_pressure=features[1], albumin=features[2],
        serum_creatinine=features[3], hemoglobin=features[4],
        blood_urea=features[5], sodium=features[6], potassium=features[7],
        specific_gravity=features[8], model_used=model_choice, prediction=prediction
    )
    db.session.add(patient)
    db.session.commit()
    return render_template("result.html", prediction=prediction)
@app.route("/patients")
def patients():
    if "user" not in session:
        return redirect("/login")
    return render_template("patient_records.html", patients=all_patients)
@app.route("/login", methods=["GET", "POST"])
def login():
    if request.method == "POST":
        username = request.form["username"]
        password = request.form["password"]
        user = User.query.filter_by(username=username).first()
```

```
        if user and check_password_hash(user.password, password):
            session["user"] = username
            return redirect("/index")
        else:
            flash("Invalid username or password!", "danger")
    return render_template("login.html")
@app.route("/signup", methods=["GET", "POST"])
def signup():
    if request.method == "POST":
        username = request.form["username"]
        password = request.form["password"]
        if User.query.filter_by(username=username).first():
            flash("Username already taken. Try another one.", "warning")
            return redirect("/signup")
        hashed_password = generate_password_hash(password, method="pbkdf2:sha256")
        new_user = User(username=username, password=hashed_password)
        db.session.add(new_user)
        db.session.commit()
        flash("Signup successful. Please login.", "success")
        return redirect("/login")
    return render_template("signup.html")
@app.route("/logout")
def logout():
    session.pop("user", None)
    flash("Logged out successfully.", "info")
    return redirect("/login")
if __name__ == "__main__":
    app.run(debug=True)
```

patient_records

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Patient Records</title>
   <style>
      body { font-family: Arial, sans-serif; margin: 20px; }
      table { width: 100%; border-collapse: collapse; margin-top: 20px; }
      th, td { border: 1px solid black; padding: 8px; text-align: center; }
</style><script>
function filterByDate() {
constselectedDate = document.getElementById("date-filter").value;
const rows = document.querySelectorAll("table tbody tr");
let count = 0;
rows.forEach(row => {
const date = row.getAttribute("data-date");
 if (selectedDate === "" || date === selectedDate) {
row.style.display = "";
count++;
} else {row.style.display = "none";}})
</script>
</head>
<body>
<div class="container">
<h2>Patient Records</h2>
<label for="date-filter"><strong>Select Date:</strong></label>
<input type="date" id="date-filter" class="filter-date" onchange="filterByDate()">
      <div id="record-count">Showing all records</div>
      <table>
         <thead><tr><th>ID</th>
               <th>Age</th>
```

```
            <th>Blood Pressure</th>
            <th>Albumin</th>
            <th>Serum Creatinine</th>
            <th>Hemoglobin</th>
            <th>Blood Urea</th>
            <th>Sodium</th>
            <th>Potassium</th>
            <th>Specific Gravity</th>
            <th>Model Used</th>
            <th>Prediction</th>
            <th>Date</th>
          </tr>  </thead><tbody>
          <tr data-date="{{ patient.created_at }}">
            <td>{{ patient.id}}</td>
            <td>{{ patient.age}}</td>
            <td>{{ patient.blood_pressure }}</td>
            <td>{{ patient.albumin }}</td>
            <td>{{ patient.serum_creatinine }}</td>
            <td>{{ patient.hemoglobin }}</td>
            <td>{{ patient.blood_urea }}</td>
            <td>{{ patient.sodium }}</td>
            <td>{{ patient.potassium }}</td>
            <td>{{ patient.specific_gravity }}</td>
            <td>{{ patient.model_used }}</td>
            <td><strong>{{ patient.prediction }}</strong></td>
            <td>{{ patient.created_at }}</td>
          </tr>
          {% endfor %}
        </tbody>
      </table>
      <a href="/" class="back-btn">Back to Home</a>
    </div>
</body>
</html>
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            font-family: 'Poppins', sans-serif;
        }

        body {
            background: url("https://i.postimg.cc/Prvp7qP1/l.jpg") no-repeat centercenter fixed;
            background-size: cover;
            display: flex;
            align-items: center;
            justify-content: center;
            height: 100vh;
            color: white;
        }

        .container {
            background: rgba(0, 0, 0, 0.8);
            backdrop-filter: blur(10px);
            padding: 30px;
            border-radius: 15px;
            box-shadow: 0px 4px 15px rgba(255, 255, 255, 0.1);
            text-align: center;
            width: 350px;
```

```
    transition: transform 0.3s ease-in-out;
}

.container:hover {
    transform: scale(1.05);
}

h2 {
    font-size: 30px;
    margin-bottom: 20px;
    color: #ffcc00;
}

label {
    font-size: 18px;
    display: block;
    margin: 10px 0 5px;
    text-align: left;
    color: #ccc;
}

input {
    font-size: 16px;
    padding: 10px;
    width: 100%;
    border-radius: 8px;
    border: none;
    outline: none;
    background: rgba(255, 255, 255, 0.1);
    color: white;
    transition: 0.3s;
}
input:focus {
    background: rgba(255, 255, 255, 0.2);
```

```
            box-shadow: 0 0 8px #ffcc00;}
        input[type="submit"] {
            font-size: 18px;
            font-weight: bold;
            background: linear-gradient(45deg, #007bff, #0056b3);
            color: white;
            cursor: pointer;
            transition: 0.3s;
            border-radius: 8px;
            margin-top: 15px;
            border: none;
            box-shadow: 0 0 10px rgba(0, 123, 255, 0.5);
        a {
            color: #ffcc00;
            text-decoration: none;
            font-weight: bold;
            transition: color 0.3s;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Login</h2>
        <form action="/login" method="post">
            <label>Username:</label>
            <input type="text" name="username" required>
            <label>Password:</label>
            <input type="password" name="password" required>
            <input type="submit" value="Login">
        </form>
        <p>Don't have an account? <a href="/signup">Sign up here</a></p>
    </div>
</body>
</html>
```

APPENDIX – 2
SCREENSHOTS

Sample Output

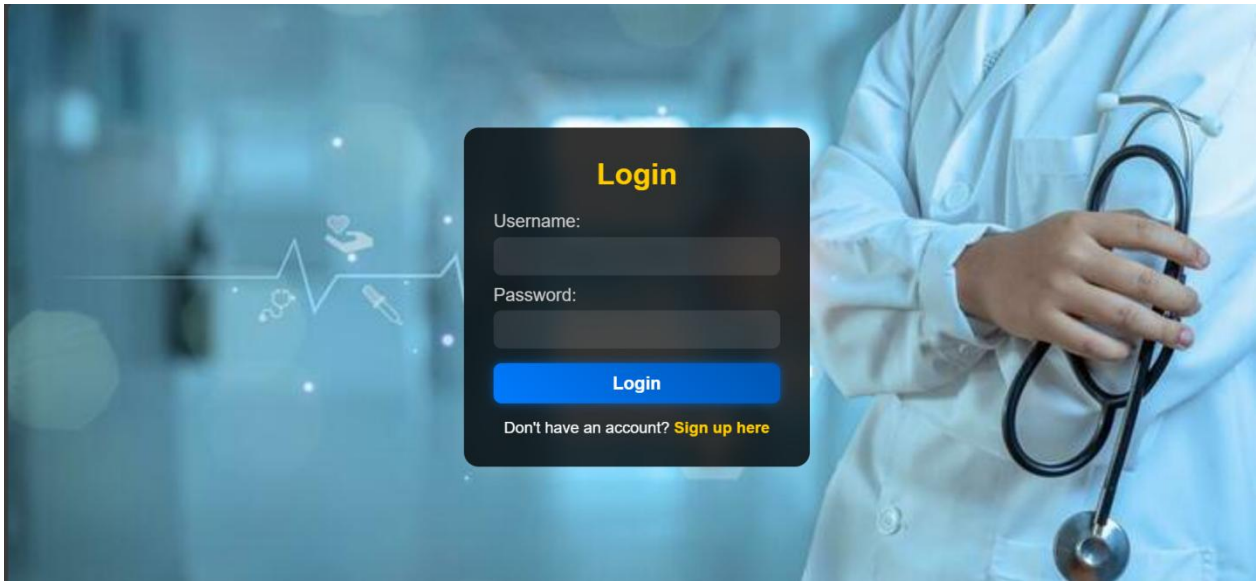

Fig 2.1: Login Page Of CKD

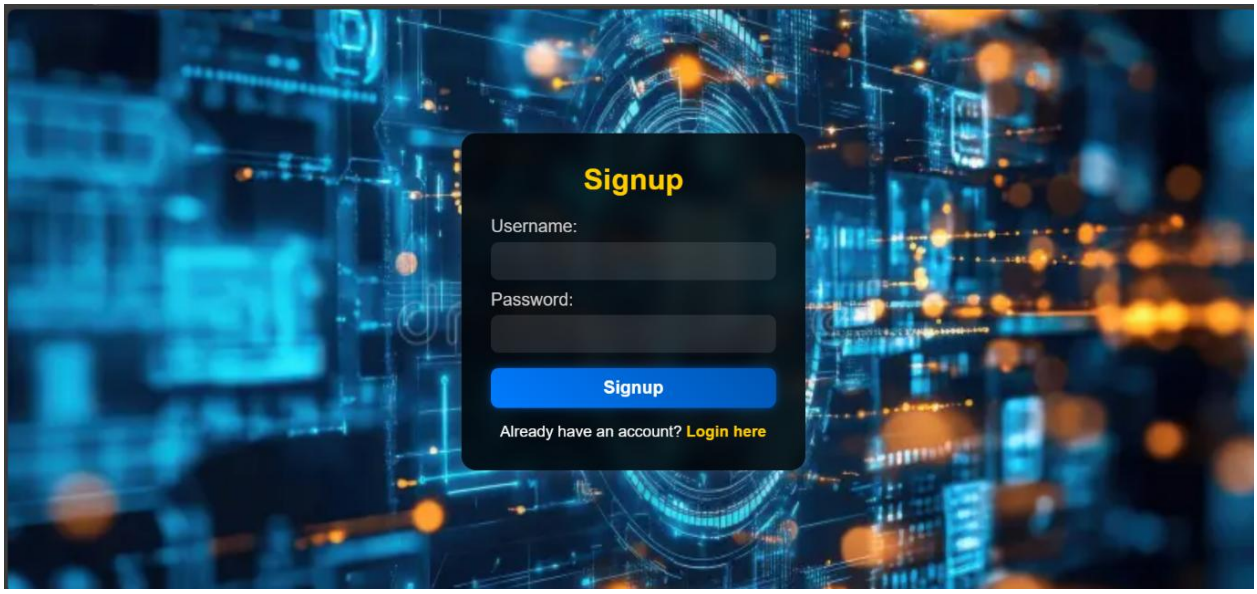

Fig 2.2 : Signup Page Of CKD

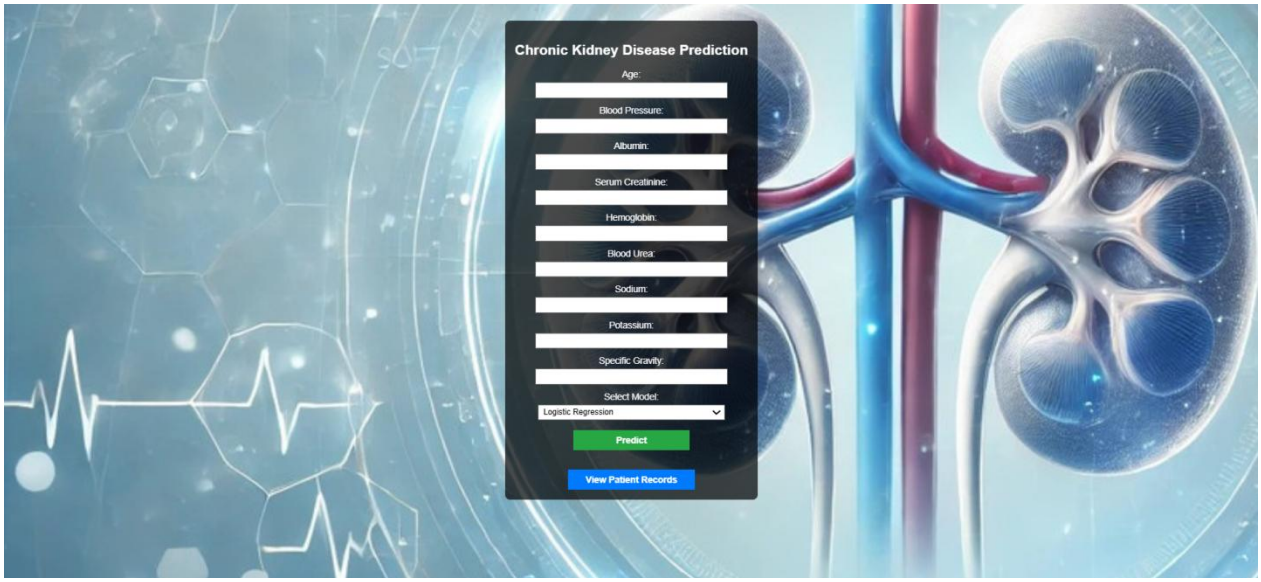

Fig 2.3: Register Page Of CKD

Patient Records

Select Date: dd-mm-yyyy

Showing all records

ID	Age	Blood Pressure	Albumin	Serum Creatinine	Hemoglobin	Blood Urea	Sodium	Potassium	Specific Gravity	Model Used	Prediction	Date
1	60	80	3	3.5	8.5	100.0	135.0	5.5	1.0	logistic	CKD	
2	12	23	45	67.0	78.0	89.0	34.0	54.0	1.0	logistic	CKD	
3	65	160	3	5.2	8.5	100.0	130.0	5.8	1.05	logistic	CKD	
4	55	80	2	3.2	9.5	54.0	137.0	4.6	1.1	logistic	yes	
5	68	150	1	8.9	7.8	120.0	130.0	6.2	1.0	logistic	yes	
6	70	180	4	5.0	7.0	80.0	120.0	6.0	1.0	logistic	yes	
7	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
8	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
9	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
10	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
11	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
12	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
13	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
14	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
15	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
16	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
17	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
18	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
19	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
20	60	90	3	5.2	9.0	60.0	132.0	5.8	1.01	logistic	yes	
21	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
22	60	90	3	5.2	9.0	60.0	5.2	132.0	1.01	logistic	yes	
23	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	no	
24	30	70	0	0.8	15.0	20.0	140.0	4.5	1.02	logistic	No CKD	

Back to Home

Fig 2.4 : Patient Record Trained Dataset

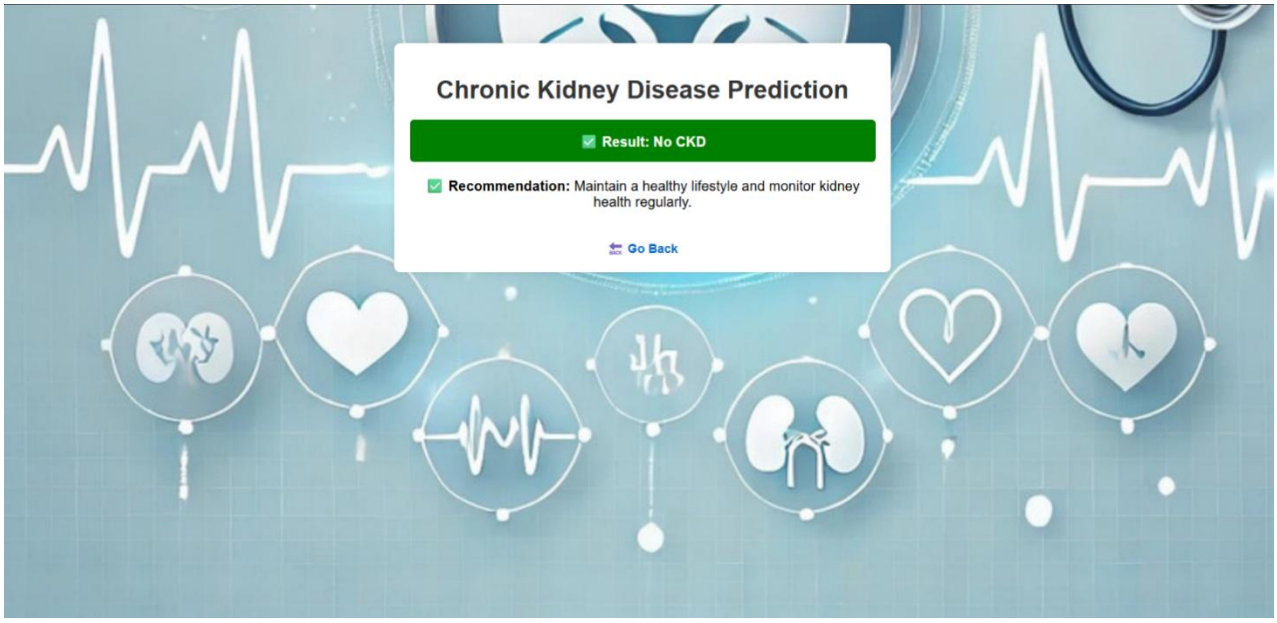

Fig 2,5 : CKD Not Detected

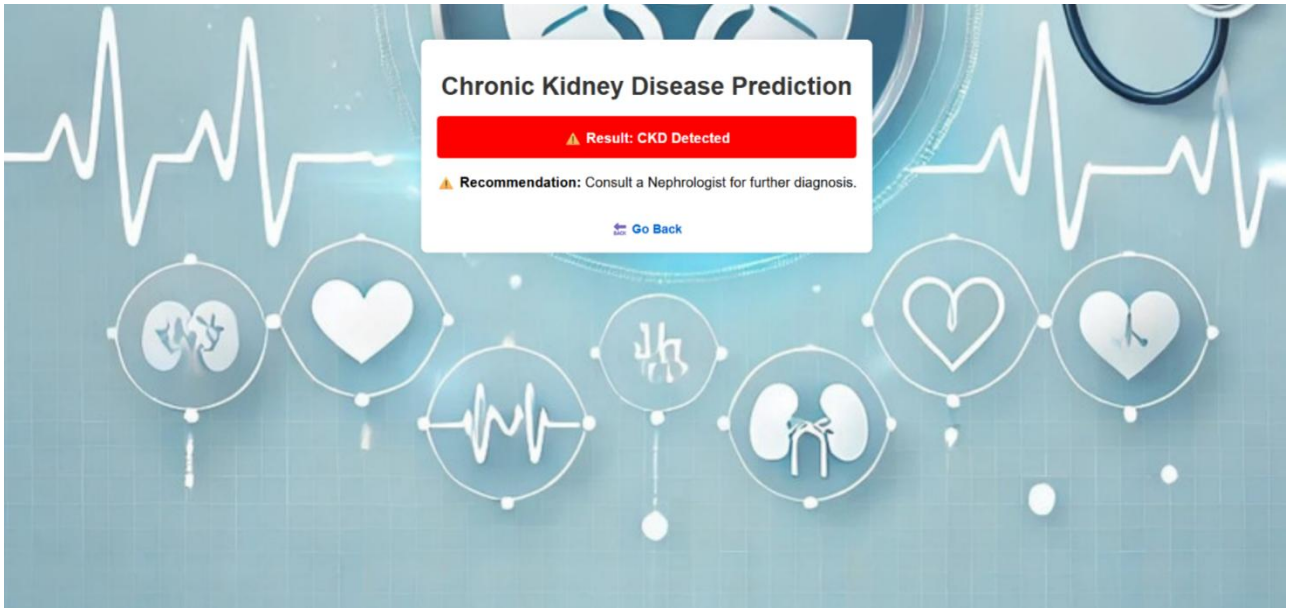

Fig 2.6 : CKD Detected

REFERENCES

1. S. S. A. Naqvi and H. A. Khan, “Chronic Kidney Disease Prediction Using Machine Learning Algorithms,” *ResearchGate*, vol. 6, no. 2, pp. 45–52, 2020.
2. P. T. Nguyen and L. V. Pham, “A Review on Predicting Chronic Kidney Disease Using Machine Learning,” *MDPI Applied Sciences*, vol. 10, no. 16, pp. 5674–5685, 2020.
3. A. Sharma and A. Patel, “Application of Data Mining Techniques for Early Detection of Chronic Kidney Disease,” *IEEE Xplore*, pp. 183–187, 2019.
4. M. Kumar and S. K. Yadav, “Predictive Modeling for CKD using Support Vector Machines and Decision Trees,” *SpringerLink*, *Advances in Intelligent Systems and Computing*, vol. 940, pp. 359–368, 2019.
5. R. D. Patel, “Chronic Kidney Disease Diagnosis Using Neural Networks and Fuzzy Logic,” *ScienceDirect*, *Procedia Computer Science*, vol. 132, pp. 1168–1176, 2018.
6. M. Lichman, “UCI Machine Learning Repository: Chronic Kidney Disease Dataset,” Available:https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease.
7. Kaggle Contributors, “Chronic Kidney Disease Dataset,” *Kaggle*, [Online]. Available: https://www.kaggle.com/mansoordaku/ckdisease.
8. F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
9. M. Abadi et al., “TensorFlow: A System for Large-Scale Machine Learning,” in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265–283.
10. A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'Reilly Media, 2019.
11. S. Raschka and V. Mirjalili, *Python Machine Learning*, 3rd ed., Packt Publishing, 2019.

12. F. Chollet, *Deep Learning with Python*, Manning Publications, 2018.
13. B. Reinders, *Building Machine Learning Projects with Flask*, Packt Publishing, 2021.
14. J. Brownlee, "How to Develop a Machine Learning Web App in Python," *Machine Learning Mastery*,https://machinelearningmastery.com.
15. T. Akhtar, "CKD Prediction Using Machine Learning – A Case Study," *Towards Data Science*, Medium, [Online]. Available: https://towardsdatascience.com.
16. D. Smith, "Deploying ML Models Using Flask," *Data Science Central*, [Online]. Available: https://www.datasciencecentral.com.
17. K. Bhatt, "An End-to-End Guide to Predictive Modeling in Healthcare," *Analytics Vidhya*, [Online]. Available: https://www.analyticsvidhya.com.
18. M. Johnson, "Using Pandas and Seaborn for Medical Data Visualization," *KDnuggets*, [Online]. Available: https://www.kdnuggets.com.
19. GeeksforGeeks, "Logistic Regression in Python," [Online]. Available: https://www.geeksforgeeks.org.
20. "Application of Data Mining Techniques for Early Detection of Chronic Kidney Disease" – IEEE Xplore*(Presents the use of data mining in CKD classification and prediction)*