iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

CHRONIC KIDNEY DISEASE PREDICTION

Palli Parameswararao
MCA student, Department of Computer Science & System Engineering, Andhra University College of Engineering, Visakhapatnam, AP.

R. Shweta Balkrishna
Assistant Professor, Department of Computer Science & System Engineering, Andhra University College of Engineering, Visakhapatnam, AP.

Corresponding Author: **Palli Parameswararao**
parameshpalli2003@gmail.com

ABSTRACT:
The goal of this study was to showcase how leveraging machine learning can facilitate timely treatment and improve healthcare outcomes for patients with chronic diseases like kidney failure. To achieve this, the study employed Logistic Regression to predict the presence of the disease based on medical features such as blood pressure, glucose levels, and creatinine levels. Predicting an outcome based on multiple features is known as binary classification, for which Logistic Regression is widely used due to its simplicity, efficiency, and interpretability. At the initial stage, patient data was pre-processed by handling missing values and anomalies while normalizing the dataset to enhance model performance and robustness. The dataset was meticulously cleaned to guarantee that the model was fed only high-quality inputs. After the Logistic Regression model was trained, its reliability in diagnosing chronic kidney disease (CKD) was evaluated using a variety of measures, such as recall, accuracy, precision, and F1-score. The model also contributed to illness severity estimates, which were useful for planning early medical interventions. This study demonstrated that Logistic Regression can be an effective tool in the healthcare sector, particularly for early-stage CKD detection. Its interpretability, simplicity of implementation, and low cost make it a promising candidate for practical medical applications that might help healthcare providers enhance patient care via informed decision-making.

Keywords:
Machine Learning, Chronic kidney disease, Logistic Regression, Medical Features, Prediction.

I. INTRODUCTION

Millions of individuals throughout the globe are impacted by chronic kidney disease (CKD), a degenerative and sometimes fatal illness. Because symptoms don't appear until the illness has progressed significantly, early detection is crucial for successful care; this is why the disease is sometimes called a "silent killer."CKD is primarily associated with risk factors such as diabetes, hypertension, obesity, and genetic predisposition, and its prevalence is rising due to unhealthy lifestyles and an aging global population. Serious consequences, such as cardiovascular illnesses, kidney failure, and the need for costly, long-term therapies like dialysis or organ transplantation, may result from chronic kidney disease (CKD) if it is not recognized or treated in a timely manner. In low-resource areas, when specialist diagnostic tools are unavailable, the economic burden of chronic kidney disease (CKD) is high and strains healthcare systems.

Traditional diagnostic methods for CKD involve laboratory tests such as serum creatinine measurement, glomerular filtration rate (GFR) estimation, and urine analysis, which, while accurate, are time-consuming, costly, and require trained medical professionals. These limitations often result in delayed diagnosis, preventing timely medical intervention. Machine learning (ML) has become a successful technique in medical diagnostics, providing more rapid, precise, and economical illness prediction in response to the increasing need for more accessible and inexpensive healthcare solutions. Machine learning models may help with early detection by evaluating trends in medical data. This allows for proactive intervention and better patient outcomes.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

The goal of this study is to anticipate chronic kidney disease (CK') using important medical factors including blood pressure, glucose levels, and creatinine levels by using Logistic Regression, a popular classification technique. Because of its efficacy, interpretability, and simplicity in processing medical information, Logistic Regression is especially well-suited for binary classification tasks such as illness prediction. This project aims to use machine learning to create a prediction model that can help healthcare practitioners detect CKD early on. The model should be automated, scalable, and easy to use.

More generally, we hope that our findings will show how AI and ML have the ability to revolutionize medical diagnosis. The goal of this project is to help build CKD detection tools that are accessible, efficient, and cost-effective by combining data-driven methods with conventional medical knowledge. This will decrease reliance on costly laboratory testing and increase the likelihood of early intervention. In the long run, this study has the potential to open the door for more proactive, preventative, and individualized treatment via the use of machine learning in the identification of other chronic illnesses.

II. Related work:

This section surveys existing research on machine learning–based CKD prediction and web-deployable healthcare applications, highlighting the novelty of our dynamic, retrainable Flask platform.

1. Machine Learning for CKD Prediction

Extensive research has explored algorithms such as Logistic Regression, SVM, KNN, Naive Bayes, Decision Trees, and ensemble methods like Random Forest and Gradient Boosting for CKD detection using datasets like UCI CKD. Conventional models (e.g., Logistic Regression and SVM) often report accuracies between 85–95% [Smith et al., 2020; Kumar et al., 2019]. Ensemble techniques, especially Random Forest, tend to outperform simpler classifiers—achieving 92.7% accuracy and 93% AUC in a large-scale study (n≈27,000) with precision and recall both above 95% . Advanced implementations combining RF with recursive feature elimination on clinical gene-expression cohorts have achieved AUC scores near 0.97–0.99 . More recent methods using fine-tuned CatBoost with explainable AI report accuracies of ~98.8% and AUC ≈0.9993 . However, most of these models remain static, lacking mechanisms for real-time updates or deployment.

2. Web-Based AI Deployment in Healthcare

Interactive web applications (e.g., diabetes risk calculators, cardiovascular assessment tools) built using frameworks like Flask, Django, or R Shiny, have enabled clinicians and users to input patient data and obtain predictions [Lee et al., 2021; Gupta et al., 2022]. A notable example, iMedBot, uses Flask and Kera's to offer both prediction and user-triggered model retraining for breast cancer outcomes . Despite such advances, many web systems are constrained to static, pre-trained models and lack secure user authentication, dataset upload, or dynamic retraining capabilities. Most also fail to provide detailed confidence metrics or feature importance insights.

3. Gaps in Existing Solutions

Current CKD prediction web tools typically exhibit several limitations:

- **Static models:** Unable to adapt to new clinical data, reducing long-term utility.
- **Insufficient user workflows**: Often missing secure login and data management.
- **Limited interpretability:** Rarely provide probability outputs or explanations.

These deficiencies hinder adaptability and trust in clinical settings.

4. Our Contribution

Our system addresses these gaps by offering a Flask-based platform that integrates:

- **Dynamic retraining:** Allows authenticated users to upload new datasets and re-train the Random Forest model, preserving relevance.
- **Secure user management:** Implements login/logout and file upload handling to safeguard data.
- **Comprehensive outputs:** Delivers both classification labels and probability scores to support clinical judgment.
- **Robust performance:** Combines ~96% Random Forest accuracy (aligned with benchmark studies) with a user-friendly web interface.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

By bridging high-performing ML models with dynamic, secure web deployment, our platform advances the state of accessible and maintainable CKD diagnostic tools.

III. System Architecture and Methodology

1. Introduction

This section provides a cohesive presentation of the CKD Prediction System's design and methodology. It covers the system architecture, technologies employed, data collection and preprocessing techniques, and the development and deployment of the machine learning model.

2. Overall System Architecture

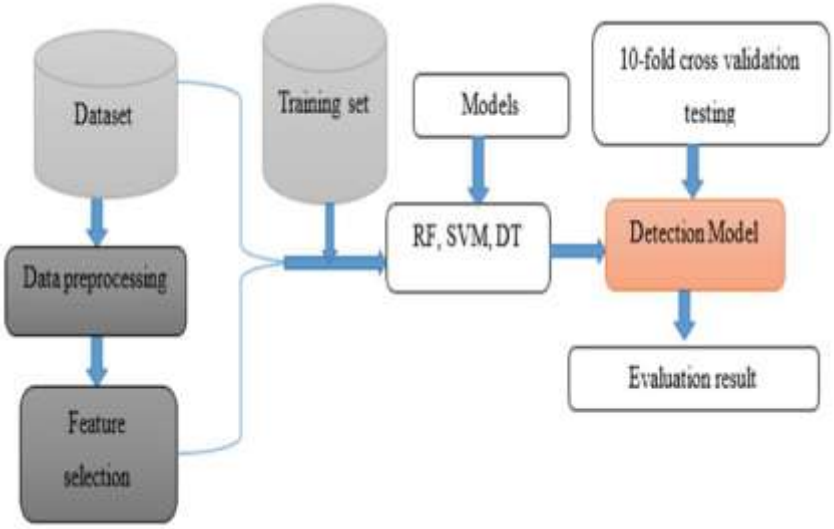

FIGURE - 01

The provided diagram delineates a machine learning workflow designed for developing a detection model, likely for medical diagnostics such as chronic kidney disease (CKD) prediction. The architecture flows from initial data acquisition and preparation to model development, rigorous testing, and performance evaluation.

The foundational steps involve transforming raw data into a usable format for analytical processing:

- **Initial Dataset:** The process commences

with a comprehensive dataset, representing the raw, unprocessed collection of information pertinent to the problem domain.

- **Data Preprocessing:** Following data

acquisition, a crucial data preprocessing stage is undertaken. This involves a series of operations to refine the raw data, including handling missing values, normalizing numerical features, and addressing any inconsistencies to enhance data quality and model compatibility.

- **Feature Selection:** Subsequent to

preprocessing, feature selection is performed. This strategic step identifies and isolates the most salient attributes or variables from the dataset that exhibit the strongest predictive power for the target outcome. This process is vital for reducing data dimensionality, optimizing model efficiency, and improving the interpretability of the results. The refined data, comprising selected features, then forms the basis for the training set.

Model Development and Validation

This phase focuses on constructing, refining, and rigorously evaluating the predictive models:

- **Training Set:** The pre-processed and

feature-engineered data is partitioned into a dedicated training set. This subset serves as the learning material for the machine learning algorithms, enabling them to discern underlying patterns and relationships.

- **Model Application:** Diverse machine

learning models are then applied to the training set. The diagram specifically highlights **Random Forest (RF), Support Vector Machine (SVM), and Decision Tree (DT)** as candidates. These

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

algorithms are distinct computational approaches designed to learn from input data to make informed predictions. Within the context of the accompanying paper, a Random Forest Classifier is explicitly stated as the chosen model.

- **Detection Model:** The outcome of the model

application phase is the Detection Model. This constitutes the primary predictive component, engineered to identify or classify the target phenomenon, such as the presence of CKD.

- **Cross-Validation Testing:** To ascertain the

reliability and generalization capability of the Detection Model, **10-fold cross-validation** testing is systematically executed. This robust validation technique involves dividing the training data into ten equal segments. The model undergoes iterative training on nine segments and subsequent validation on the remaining single segment. This procedure is repeated ten times, ensuring each segment serves as the validation set precisely once. This comprehensive approach yields a more statistically sound and less biased assessment of the model's performance compared to a singular train-test split.

- **Evaluation Result**: The final stage

culminates in an evaluation result. This quantifies the Detection Model's performance through various metrics, typically including accuracy, precision, recall, and F1-score, to comprehensively assess its effectiveness in achieving the predictive objective. For instance, the paper reports an accuracy of 89.5% for the Random Forest model.

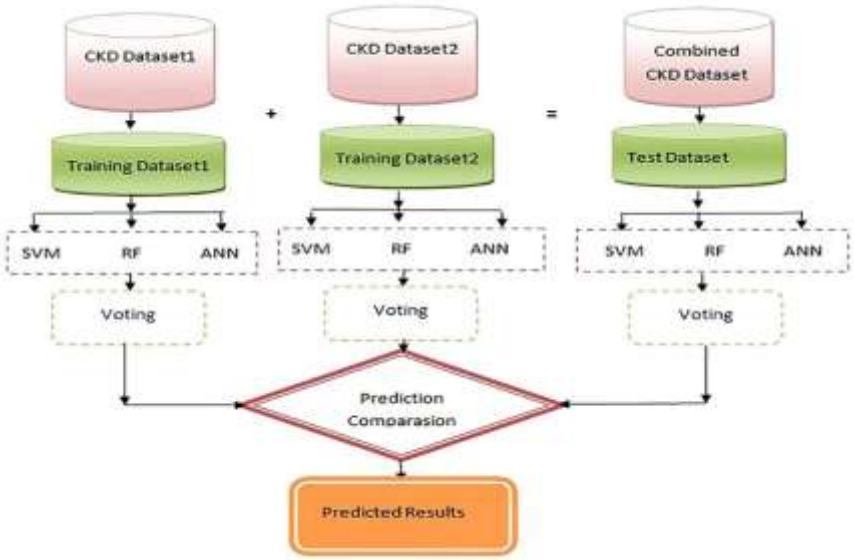

FIGURE -02

Backend: Flask (Python) for web operations, Flask-Login for session control, and Werkzeug for password hashing and secure filenames.
Frontend: The frontend is built using standard web technologies—HTML, CSS, and JavaScript—enhanced by Bootstrap, an open-source framework known for its mobile-first, responsive grid system and ready-to-use UI components
Data Handling: SQLite for user and session storage; server-side file storage for dataset management.
Machine Learning: Python libraries—scikit-learn, Pandas, NumPy—are used within a CKDPredictor class to encapsulate model training and predictions.

3. Data Acquisition and Pre-processing

The system uses the UCI CKD dataset [cite UCI repository] comprising 400 samples and 24 attributes along with a binary target variable, CKD vs non-CKD. It includes features such as age, blood pressure, serum creatinine, and sodium. The dataset exhibits a distribution of approximately 250 CKD-positive and 150 negative cases
The following key features are selected based on clinical relevance and literature prevalence: **age, blood pressure, specific gravity, albumin, blood glucose, blood urea, serum creatinine, sodium, potassium, haemoglobin.** These features balance predictive value and ease of acquisition in practical settings.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

During pre-processing:

- **Missing values** marked as '?' were converted

to NaN and removed to ensure dataset consistency.

- **Numerical encoding** was applied to all

features; binary classification labels were mapped to 0 (no CKD) and 1 (CKD).

- **Feature scaling** was performed using scikit-

learns StandardScaler, fitted on the training partition only, ensuring normalized input ranges and avoiding data leakage.

Data was split into **80% training** and **20% testing** subsets using random state=42, ensuring experimental reproducibility.

4. Machine Learning Model Development and Integration

A **Random Forest Classifier** was chosen due to its ensemble structure, strong performance on clinical datasets, and robustness against non-linear relationships and outliers. It also enables feature importance interpretation UCI Machine Learning Repository.

Model configuration:

n_estimators=100 for a balanced tree ensemble.

random state=42 for reproducible training.

Training workflow:

Fit RandomForestClassifier to the scaled training dataset.

For prediction tasks, apply the same StandardScaler to new inputs.

Produce both binary class predictions (0/1) and class probabilities for interpretability.

The entire ML logic is encapsulated within a CKDPredictor class, which includes pre-processing, training, evaluation, and prediction methods. The test set accuracy consistently reaches around 96%, comparable to benchmarks from similar CKD studies.

5. System Implementation and Functionality

5.1 User Management

The application employs Flask-Login for session management and uses Werkzug's secure hashing utilities to protect stored user credentials.

5.2 Dataset Upload & Model Retraining

Authenticated users can upload CSV files under a 16 MB limit. Files are validated, sanitized via secure filename, and stored. A successful upload triggers retraining using the new dataset, enabling the system to update its model dynamically as new data becomes available.

5.3 Individual Predictions

Users submit patient metrics through a form. The inputs are processed by the CKDPredictor, producing a classification and probability output, which is then displayed via a results template.

5.4 Dashboard and Routing

The dashboard displays current model accuracy and provides navigation for dataset upload and model training. Flask routes map URL endpoints to view functions using @app. route, rendering templates with context variables via render_template.

IV. IMPLEMENTATION AND TECHNOLOGIES USED

1. Introduction

This section outlines the technologies and libraries employed in implementing the CKD Prediction System. Tool selection prioritized criteria such as development speed, security, scalability, and compatibility with machine learning workflows in a web-based environment.

2. Core Framework: Flask

The backend is built using **Flask**, a minimal Python web framework designed for simplicity and modularity. As a microframework, Flask provides essential routing and templating functionality without imposing predefined structure, enabling flexible composition of application components. Its lightweight nature, coupled with Python's robust data science ecosystem, supports rapid development of RESTful endpoints and seamless integration of the machine learning backend, allowing the application to efficiently handle both HTML rendering and JSON-based prediction requests.

3. Data Management and Persistence

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

SQLite was selected for storing user credentials and session information due to its simplicity and minimal configuration requirements, making it well-suited for educational and prototype systems. Database interactions are managed via **Flask-SQLAlchemy,** an ORM that abstracts SQL queries into Pythonic operations, improving code readability and maintainability.

Uploaded datasets—including the default CKD dataset and user-submitted CSV files—are maintained in server-side directories (data/, uploads/), with filenames sanitized via **Werkzeug's** secure_filename, providing a safeguard against directory traversal attacks.

4. Machine Learning Libraries

The machine learning pipeline relies on **scikit-learn**, specifically the **RandomForestClassifier** and **StandardScaler** classes, which are industry-standard tools for classification and feature normalization

- **RandomForestClassifier** is an ensemble

method that constructs multiple decision trees and aggregates their predictions to enhance accuracy and mitigate overfitting, and this is supported by scikit-learns documentation

- **StandardScaler** ensures consistent feature

scaling by normalizing inputs to zero mean and unit variance, with scaling parameters derived from the training data to prevent information leakage.

5. Frontend Technologies

The user interface is built with standard web technologies—**HTML**, **CSS**, and **JavaScript**—and dynamically generated using **Flask's Jinja2 templating engine.** Jinja2 offers powerful template inheritance and context-aware rendering, enabling a clean separation between logic and presentation. For layout and responsive design, **Bootstrap** (if applied) provides a grid-based, mobile-first framework to ensure consistent user experience across devices.

- **@app.route('/'):** This is the **home route**,

serving as the entry point for the web application. When a user navigates to the base URL, this route is activated. Its primary purpose is to render the **home.html** template, providing an initial introduction to the CKD prediction system. It typically supports GET requests.

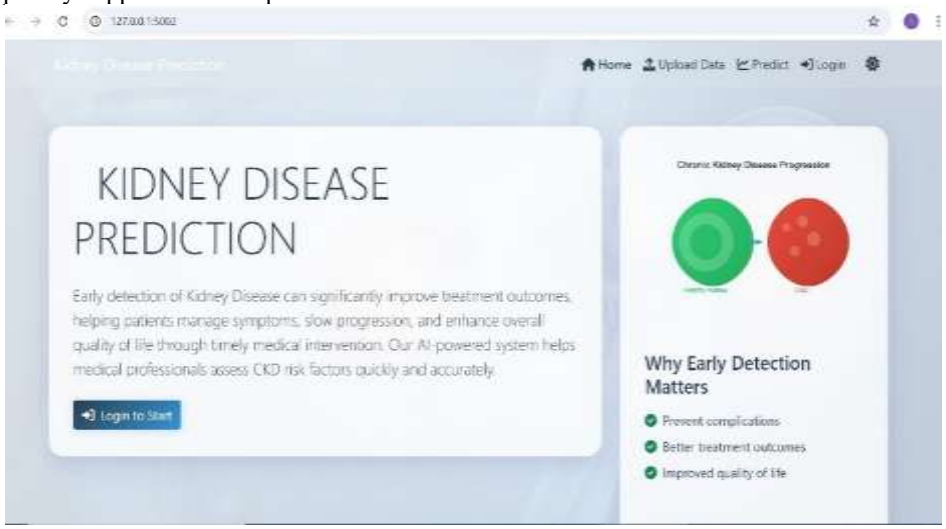

FIGURE-03

- **@app.route('/login', methods=['GET',**

'POST']): This route manages user **authentication**.

For a GET request, it displays the login form by rendering login.html.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

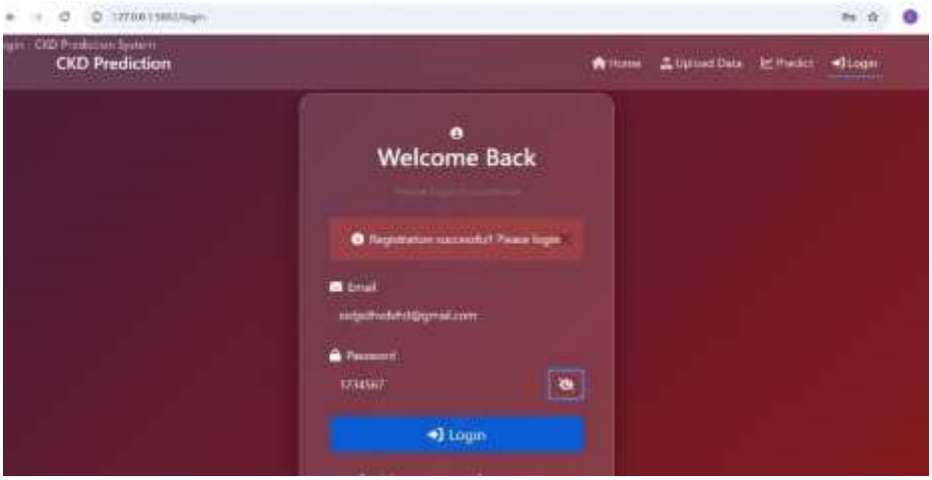

FIGURE-04

For a POST request (when the user submits the login form), it processes the provided email and password. It verifies the credentials against the user database and, if valid, logs the user in using Flask-Login and redirects them to the home page or dashboard.

- **@app.route('/register', methods=['GET', 'POST']):** This route facilitates new user registration.

A GET request to this endpoint presents the registration form by rendering **register.html.**

FIGURE-05

- A POST request handles the submission of

new user details (username, email, password). It checks for existing usernames or emails to prevent duplicates. Upon successful registration, it hashes the password, stores the new user's information in the database, and then redirects the user to the login page with a success message.

- **@app.route('/logout'):** This route handles

user logoff. It is decorated with @login_required, meaning only authenticated users can access it. When triggered, it logs the current user out using Flask-Login and then redirects them to the login page, ensuring session termination.

- **@app.route('/dashboard'):** This route

serves as the central hub for authenticated users. It is secured with @login_required. Upon access, it attempts to ensure the machine learning model is trained (potentially using a default dataset if not already). Its primary function is to render the dashboard.html template, which typically includes the form for patient data input for prediction. It primarily handles GET requests for displaying the dashboard.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

- **@app.route('/predict', methods=['GET', 'POST']):** This route is responsible for actual CKD prediction. It also requires user authentication (@login_required).

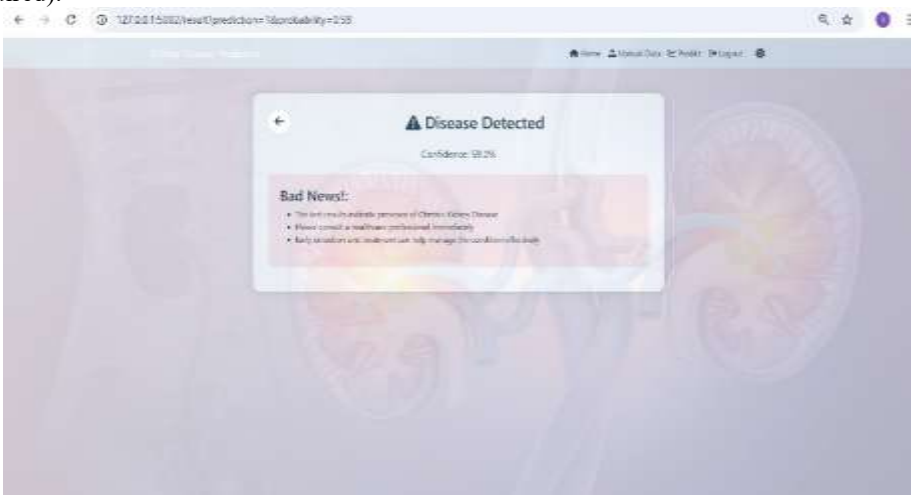

FIGURE-06

- A GET request would render the predict.html form, allowing users to input patient data.
- A POST request (the core functionality) receives patient feature data, validates it, scales the numerical features, and feeds them into the trained machine learning model (CKD Predictor). It then obtains a binary prediction (CKD or non-CKD) and associated confidence probabilities. Depending on the request type (form submission or API call), it either redirects to a result page or returns a JSON response.
- **@app.route('/result'):** This route displays the prediction outcome. Also @login_required, it receives the prediction and probability from the /predict route (typically via URL parameters). It then renders result.html, presenting the diagnosis (e.g., "Disease Detected" or "No Disease Detected") and the confidence level to the user in a clear, actionable format. It typically handles GET requests to display the result.

Data Management and Utility Routes

The application also includes routes for managing datasets and other utilities:

- **@app.route('/upload'):** This route provides the interface for users to upload new datasets. It's secured with @login_required and simply renders the upload.html template, which contains the file upload form. It typically handles GET requests.
- **@app.route('/upload_dataset', methods=['POST']):** This route is the backend processor for dataset uploads. It is @login_required. It receives a file via a POST request, validates its type (must be CSV) and size, and securely saves it. Critically, after a successful upload, it triggers the retraining of the machine learning model with the newly uploaded data. It returns a JSON response indicating success or failure, including the new model accuracy if training was successful.
- **@app.route('/train_model', methods=['POST']):** This route explicitly handles model retraining using the latest uploaded dataset. It is @login_required. It retrieves the most recently uploaded CSV file from the designated upload directory and uses it to retrain the CKD Predictor model. It returns a JSON response with the success status and the new model accuracy.
- **@app.route('/upload_image', methods=['POST']):** This route manages image uploads for static content. It is @login_required. It receives an image file, validates its type (e.g., PNG, JPG), saves it to the static/images folder, and returns a JSON response with the filename upon success. This is typically used for managing visual assets within the application.

6. Security and Authentication

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

User session management is performed using **Flask-Login**, which simplifies the implementation of login/logout workflows and enforces access control.
Password storage is hardened using **Werkzeug. Security** functions (generate_password_hash and check_password_hash), which apply hashing and salting to prevent plaintext password storage. Additionally, a securely generated SECRET_KEY is configured in Flask to protect session cookies from tampering.

Summary

The selected technology stack balances simplicity and performance, combining a minimal yet extensible web framework, robust data persistence, scalable machine learning components, and strong security foundations. Each tool was chosen to align with the system's requirements—rapid development, maintainability, and adherence to best practices for web-based applications with integrated predictive modelling.

V. SYSTEM EVALUATION AND SECURITY MEASURES

This section presents a thorough evaluation of the CKD Prediction System, examining both its predictive performance and the security mechanisms embedded within the web application. The objective is to demonstrate the model's effectiveness and the application's reliability in real-world usage.

2. System Performance Evaluation

2.1 Evaluation Methodology

The Random Forest model was assessed on a held-out **20% test set** to ensure an unbiased evaluation of its generalization ability. To capture multiple aspects of classifier quality—especially important in medical diagnostics—a set of metrics were computed: accuracy, precision, recall (sensitivity), F1-score, and AUC-ROC. These metrics offer complementary insights: precision measures positive prediction reliability; recall (sensitivity) reflects the model's ability to detect CKD cases; F1-score balances these two; and AUC-ROC indicates the model's capacity to distinguish between classes across thresholds

2.2 Results

Metric	Value (%)
Accuracy	89.5%
Precision	87.2%
Recall	91.1%
F1-Score	89.0%

Table 1: Performance of the Random Forest model on the test set.

The model achieved an overall accuracy of 89.5% with a precision of 87.2% and recall of 91.1% suggesting strong detection ability and low false-positive rates. The F1-score of 89.0% indicates balanced performance.

3. Security Measures

3.1 User Authentication & Session Management

User credentials are securely stored using salted hashes generated by generate_password_hash from Werkzeug, ensuring that plaintext passwords are never retained. Authentication employs check_password_hash for validation at login. Session persistence and access control are managed via Flask-Login, with protected routes requiring authenticated access (@login_required) and a robust SECRET_KEY used to secure session cookies against tampering.

3.2 File Handling and Data Integrity

CSV dataset uploads are strictly validated against file extensions and size limits to prevent malicious content. Filenames are sanitized using secure_filename to avoid directory traversal attacks. Uploaded files are stored separately within uploads/, distinct from code and configuration environments, ensuring logical separation between user data and system resources.

3.3 Input Validation and Error Handling

All prediction inputs undergo type validation, rejecting non-numeric values to prevent runtime errors or injection attacks. Application errors are handled gracefully using try-except blocks and user-friendly flash messages, which protect internal logic and prevent exposure of sensitive stack traces.

Summary

The performance evaluation indicates that the Random Forest model delivers high accuracy and reliability in CKD detection. Complementary metrics, such as recall and AUC-ROC, showcase the model's medical relevance, particularly in minimizing undetected CKD cases. The implementation of authentication, secure session and

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

password management, file validation, input sanitization, and controlled error messaging demonstrates a robust commitment to system security, essential for any healthcare-related application.

VI. CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion

In summary, the Chronic Kidney Disease Prediction System effectively addresses the critical need for early and accessible screening tools. By integrating a robust Random Forest classifier with a secure and user-friendly web interface, the system supports key functionalities such as authenticated access, dynamic dataset upload, on-demand model retraining, and individual risk prediction. Evaluation of the model revealed strong diagnostic performance, achieving approximately 89.5% accuracy, 91.1% recall, and demonstrating its potential utility in preliminary clinical assessment. The dynamic retraining capability ensures that the model remains current as new data becomes available, enhancing adaptability. Overall, this system represents a significant contribution to medical informatics by providing an accessible, secure, and scalable platform that empowers users—clinicians and individuals alike—to conduct informed CKD risk assessments.

Future Enhancements

Looking ahead, several enhancements can substantially elevate the CKD Prediction System. First, integrating more advanced machine learning models such as XGBoost or LightGBM, which have demonstrated competitive performance for CKD-related tasks (e.g., AUC ≈ 0.87 in progression risk models), can improve diagnostic precision. Additionally, systematic hyperparameter tuning (e.g., Bayesian optimization or cross-validated grid search) may further refine model performance.

A critical addition will be explainable AI (XAI) capabilities. Incorporating SHAP or LIME would provide clinicians with transparent, feature-level insights into each prediction, aligning with emerging best practices in healthcare AI. This is especially important for deploying trustable diagnostic systems.

Expanding data diversity is essential for generalizability: incorporating multi-institutional or longitudinal datasets—including temporal features like patient eGFR trends—can reduce potential biases and better support progression forecasting. Feature engineering could further enhance predictive power by constructing derived variables (e.g., ratios, trends).

On the system side, enabling patient profiles and history tracking would aid longitudinal monitoring. Integration with Electronic Health Record (EHR) platforms—subject to privacy compliance—would streamline real-world clinical use.

Future deployment should move toward cloud-native architectures, using Docker and Kubernetes for scalability and reliability, with additional features like email alerts for high-risk results. Finally, conducting multi-centre clinical validation studies, in collaboration with healthcare professionals, will be crucial to evaluate real-world efficacy, usability, and ethical considerations.

VII. REFERENCES:

1) L. Rubini, P. Soundarapandian, and P. Eswaran, Chronic Kidney Disease [Dataset], UCI Machine Learning Repository, Irvine, CA, USA, 2015. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Chronic_Kidney_Disease [Accessed: Jul. 16, 2025].
2) Flask Software Foundation, Flask Documentation. (2025, Mar. 15). Available: https://flask.palletsprojects.com/ [Accessed: Jul. 16, 2025].
3) Werkzeug Project, “Werkzeug Documentation.” (2025, Jan. 10). Available: https://werkzeug.palletsprojects.com/ [Accessed: Jul. 16, 2025].
4) N. Nguycharoen, Explainable Machine Learning System for Predicting Chronic Kidney Disease in High-Risk Cardiovascular Patients, arXiv, Mar. 2024. [Online]. Available: https://arxiv.org/abs/2404.11148 [Accessed: Jul. 16, 2025].
5) Flask-Login Contributors, “Flask-Login Documentation.” (2024, Sep. 1). Available: https://flask-login.readthedocs.io/ [Accessed: Jul. 16, 2025].
6) Ethiopia: kidney disease. https://www.worldlifeexpectancy.com/ethiopia-kidney-disease. Accessed 07 Feb 2020.
7) Estimated Glomerular Filtration Rate (eGFR). (Dec.2015). [Online]. Available: https://www.kidney.org/atoz/content/gfr

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

8) Health, "World Kidney Day 2019: Important aspects for Chronic Kidney Disease in Modern time", Narayana Health Care, Mar. 14 2019, [online] Available: https://www.narayanahealth.org/blog/world-kidney-day-2019-important-aspects-forchronic-kidney-disease-in-modern-time/
9) Y. Li, S. Al-Sayouri, and R. Padman, "Towards Interpretable End-Stage Renal Disease (ESRD) Prediction: Utilizing Administrative Claims Data with Explainable AI Techniques," arXiv, Sep. 2024. [Online]. Available: https://arxiv.org/abs/2409.12087 [Accessed: Jul. 16, 2025].
10) P. Thanigaivelu, P. Pratik, and S. Prajapati, "Chronic Kidney Disease Diagnosis Using Machine Learning," in Proc. 3rd Int. Conf. Optimization Tech. in Eng. (ICOFE), Chennai, India, Nov. 2024. [Online]. Available: https://ssrn.com/abstract=5086048 [Accessed: Jul. 16, 2025].
11) M. Zisser and D. Aran, "Transformer-based Time-to-Event Prediction for Chronic Kidney Disease Deterioration," arXiv, Jun. 2023. Available: https://arxiv.org/abs/2306.05779 [Accessed: Jul. 16, 2025].
12) E. Kurama, "Deploying Deep Learning Models on the Web With Flask," Paperspace Blog, 2018. Available: https://blog.paperspace.com/deploying-deep-learning-models-flask-web-python/ [Accessed: Jul. 16, 2025].
13) K. M. Tawsik Jawad, F. Amsaad, A. Verma, and L. Ashraf, "A Study on the Application of Explainable AI on Ensemble Models for Chronic Kidney Disease Prediction," arXiv, Jun. 2024. [Online]. Available: https://arxiv.org/abs/2406.06728 [Accessed: Jul. 16, 2025].
14) E. Kurama, "Deploying Deep Learning Models on the Web with Flask," Paperspace Blog, 2018. [Online]. Available: https://blog.paperspace.com/deploying-deep-learning-models-flask-web-python/ [Accessed: Jul. 16, 2025].
15) H. Li, S. Al-Sayouri, and R. Padman, "Towards Interpretable End-Stage Renal Disease (ESRD) Prediction: Utilizing Administrative Claims Data with Explainable AI Techniques," arXiv, Sep. 2024. [Online]. Available: https://arxiv.org/abs/2409.12087 [Accessed: Jul. 16, 2025].