# ABSTRACT

This project proposed a machine learning-based system of chronic disease prediction, which was created to predict chronic diseases including diabetes, cardiovascular disease and hypertension through the accurate classification of disease risk using patient medical records and implemented using Python and Flask. The system uses a systematic procedure of acquiring data, pre-processing to remove missing values, noise, normalization and feature selection model selection, training, testing, hyperparameter optimization and result production. The accuracy, precision, recall, and F1-score were considered as standard measures to check the reliability and predictive ability of machine learning algorithms, which included Random Forest, Decision Tree, and Logistic Regression. The model with the highest performance was implemented into an interactive model in Flask-based web application where users can enter patient data and get an instant prediction and the risk factors. The findings prove that the system is precise and reliable in providing feasible assistance to healthcare decisions made using information. The next improvements can involve remote computing, mobile platform execution, and real-time tracking to enhance the scaling capabilities and allow managing preventive health care in a more efficient way on a bigger scale. The system ensures data security and privacy through secure data handling protocols, making it suitable for real-world clinical use. The project demonstrates how artificial intelligence can support early diagnosis and risk assessment, reducing the burden on healthcare profession

**TABLE OF CONTENTS**

[ABSTRACT 1](#_Toc212589121)

[CHAPTER 1 3](#_Toc212589122)

[INTRODUCTION 3](#_Toc212589123)

[1.1 Background 3](#_Toc212589124)

[1.2 Problem Statement 4](#_Toc212589125)

[1.3 Objectives](#_Toc212589126) 4

[1.4 Scope of the Project](#_Toc212589127) 6

[1.5 Significance of the Project 6](#_Toc212589128)

[CHAPTER 2 8](#_Toc212589129)

[LITERATURE REVIEW 8](#_Toc212589130)

[CHAPTER 3 13](#_Toc212589131)

[METHODOLOGY 13](#_Toc212589132)

[3.1 Overview 13](#_Toc212589133)

[3.2 Existing System 14](#_Toc212589134)

[3.3 Proposed System 15](#_Toc212589134)

[3.4 System Architecture 16](#_Toc212589136)

[3.5 Tools and Technologies 22](#_Toc212589136)

[3.6 Data Collections 23](#_Toc212589137)

[3.7 Model Training 24](#_Toc212589137)

[3.8 Chronic disease prediction using python and flask 24](#_Toc212589138)

[CHAPTER 4 26](#_Toc212589143)

[RESULTS AND DISCUSSION 26](#_Toc212589144)

[4.1 System Implementation 26](#_Toc212589145)

[4.2 Testing Environments 26](#_Toc212589146)

[4.3 Performance Metrics](#_Toc212589147) 27

[4.4 Observations 28](#_Toc212589148)

[4.5 Limitations 25](#_Toc212589149)

[4.6 Future Enhancements 33](#_Toc212589150)

[CHAPTER 5 34](#_Toc212589155)

[CONCLUSION 34](#_Toc212589156)

[5.1 Conclusion 34](#_Toc212589157)

[REFERENCES 35](#_Toc212589161)

# CHAPTER 1

**INTRODUCTION**

* 1. **Background**

Despite significant advancements in medical science, early diagnosis of chronic diseases remains a major challenge due to reliance on traditional clinical methods such as routine checkups, laboratory analyses, and patient self-reporting. These approaches often fail to detect diseases in their initial stages and are affected by multiple factors including genetics, lifestyle, and environmental influences, making manual diagnosis both time-consuming and prone to error. With the emergence of digital healthcare systems, vast volumes of patient data are generated daily from electronic health records, diagnostic tests, and wearable devices, offering immense potential for analysis but exceeding the limits of manual interpretation. Machine Learning (ML), a key branch of Artificial Intelligence (AI), provides advanced solutions by learning from data, uncovering hidden patterns, and predicting outcomes accurately. In healthcare, ML has been widely used for disease prediction, risk analysis, and personalized treatment recommendations. Algorithms like Logistic Regression, Decision Tree, and Random Forest can process complex patient data to estimate disease likelihoods. This project aims to develop an ML-based system that integrates clinical and lifestyle data to deliver early, accurate, and interpretable chronic disease predictions through a Flask web application, enabling real-time risk assessment and assisting healthcare professionals in making informed, data-driven decisions.

## Problem Statement

Timely detection of chronic diseases plays a vital role in ensuring effective treatment and improving patient outcomes. However, traditional diagnostic methods primarily depend on manual clinical evaluations, laboratory examinations, and periodic health checkups, which often delay detection and may lead to inaccuracies. These conventional approaches lack the ability to provide real-time risk assessment, making it difficult to identify high-risk individuals during early or asymptomatic stages of disease development. With the increasing availability of patient health data generated from electronic health records, diagnostic reports, and wearable devices, there arises a strong need for an automated and intelligent system that can accurately predict chronic diseases using machine learning techniques. Such a data-driven approach would enable healthcare professionals to identify potential risks at an early stage, personalize treatment strategies, and ultimately enhance the overall quality and efficiency of healthcare services.

## Objective

* The primary objective of this project is to develop a predictive model using various machine learning algorithms for the early detection of chronic diseases. The main goal is to design and implement a robust system capable of accurately identifying the risk of diseases such as diabetes, cardiovascular disease, and hypertension at an early stage. By leveraging algorithms like Logistic Regression, Decision Tree, and Random Forest, the system aims to assist healthcare professionals in diagnosing these diseases before they reach severe stages, thereby improving patient outcomes and enhancing preventive healthcare strategies.
* To achieve this, the project focuses on effective preprocessing of clinical and lifestyle data by handling missing values, removing noise, normalizing features, and selecting relevant attributes to ensure the dataset used for training and testing is accurate, consistent, and of high quality. Proper preprocessing minimizes bias, enhances reliability, and improves the overall accuracy of the predictive models.
* Furthermore, multiple algorithm—Logistic Regression, Decision Tree, and Random Forest—are trained and evaluated using performance metrics such as accuracy, precision, recall, and F1-score to identify the best-performing model for deployment. The system also emphasizes interpretability by generating understandable outputs that highlight disease probabilities, key risk factors, and preventive recommendations. Finally, the project integrates the developed model into a Flask-based web application that allows users to input patient data and receive real-time predictions through an interactive and user-friendly interface, effectively bridging the gap between advanced machine learning and practical healthcare applications. This integration bridges the gap between advanced machine learning and practical healthcare applications, providing an accessible, efficient, and intelligent tool to support early diagnosis and promote proactive healthcare managemen

**1.4 Scope of the Project**

The project focuses on building a desktop or web-based system for predicting the risk of chronic diseases by analyzing patients’ clinical and lifestyle data.

Key components include:

* Data Preparation: Cleaning and preprocessing the dataset to ensure quality input for ML algorithms.
* Model Development: Implementing and comparing different machine learning models to assess their performance using metrics such as accuracy, precision, recall, and F1-score.
* Web Integration: Deploying the best-performing model in a Flask-based web interface for real-time predictions.
* Interpretation: Providing understandable results that highlight major risk factors associated with chronic conditions.

This system can be used by healthcare professionals and researchers for early risk detection and patient monitoring. The model can also be extended to other diseases by retraining with new datasets.

**1.4 Significance of the Project**

The proposed machine learning–based chronic disease prediction system offers an efficient, intelligent, and automated approach for the early detection of chronic illnesses, reducing the workload of healthcare professionals while improving diagnostic precision and promoting preventive healthcare practices. By analyzing vast amounts of medical and lifestyle data, the system can identify individuals who are at high risk before symptoms become severe, enabling timely medical intervention and better patient management. This not only enhances clinical decision-making but also helps allocate healthcare resources more effectively, reducing the burden on hospitals and improving overall healthcare efficiency. Moreover, the system generates interpretable and evidence-based predictions that assist doctors in developing personalized treatment plans and preventive strategies tailored to each patient’s unique health profile. By bridging the gap between technology and medicine, this project demonstrates the transformative role of data science and artificial intelligence in modern healthcare, turning raw medical data into actionable insights. It promotes transparency, accuracy, and trust in predictive healthcare systems, empowering both patients and practitioners with data-driven knowledge. Ultimately, this project contributes to the advancement of smart and sustainable healthcare solutions that emphasize prevention over treatment, supporting global efforts to move from reactive to proactive healthcare, thereby improving quality of life and reducing the socioeconomic impact of chronic dise

**CHAPTER 2**

**LITERATURE REVIEW**

* Ahmed and Husien (2024) reviewed hybrid ML models for heart disease prediction,focusing on the integration of deep learning and traditional classifiers. Their brief review highlighted the advantages of combining different learning paradigms to capture both linear and nonlinear patterns in medical data. They also discussed the challenges of model interpretability and the need for explainable AI in clinical settings. The authors concluded that hybrid models offer a promising direction for improving diagnostic accuracy and facilitating clinical adoption of ML technologies.
* Arif, Rehman, and Asif (2024) developed an explainable ML model for CKD prediction, utilizing frameworks such as SHAP and LIME to provide transparency in model decisions. Their study emphasized the importance of interpretability in healthcare, where clinicians must understand and trust algorithmic outputs. By visualizing feature contributions and decision pathways, their model enhanced user confidence and facilitated clinical integration. This work aligns with the growing emphasis on explainable AI in medicine, ensuring that ML models are not only accurate but also accountable.
* Chang et al. (2019) introduced a machine-learning-based method for predicting hypertension outcomes using structured medical data. Their model incorporated demographic, lifestyle, and clinical variables to forecast the likelihood of hypertension-related complications. By employing algorithms such as logistic regression and gradient boosting, they achieved high predictive accuracy and demonstrated the feasibility of integrating ML into routine hypertension management. The study also discussed the interpretability of models, a critical factor in clinical adoption, and proposed visualization techniques to help clinicians understand the influence of individual features. This work laid the groundwork for personalized hypertension care, where ML can guide treatment plans based on patient-specific risk profiles.
* Donmez, Kutlu, Mansour, and colleagues (2025) applied explainable AI techniques to analyze hypertension risk factors, using SHAP and LIME to interpret model outputs. Their comparative analysis revealed key features influencing hypertension risk and demonstrated the utility of explainable AI in enhancing model transparency. The study also discussed the implications of feature importance for clinical decision-making and public health interventions. By bridging the gap between ML and clinical practice, their work contributes to the responsible deployment of AI in healthcare.
* Fitriyani et al. (2019) proposed an ensemble learning approach for predicting diabetes and hypertension, combining classifiers such as decision trees, support vector machines, and k-nearest neighbors. Their model demonstrated superior performance compared to individual classifiers, highlighting the benefits of ensemble techniques in handling complex and noisy medical data. The study also emphasized the importance of cross-validation and hyperparameter tuning in achieving reliable predictions. By addressing both diabetes and hypertension, the authors showcased the versatility of ensemble learning in multi-disease prediction, paving the way for integrated diagnostic tools in primary care.
* Metwally, Mekky, and Elhenawy (2022) combined genetic algorithms with ML classifiers to enhance heart disease prediction. Their hybrid model used genetic algorithms for feature selection, optimizing the input space for classifiers such as support vector machines and decision trees. The study demonstrated improved accuracy and reduced computational complexity, showcasing the synergy between evolutionary computation and ML. This approach is particularly useful in high-dimensional datasets where traditional feature selection methods may fall short. The authors advocated for further exploration of hybrid models in medical diagnostics.
* Modak and Jha (2024) presented a diabetes prediction model using ML techniques, focusing on the methodological aspects of data preprocessing, feature selection, and algorithm tuning. Their study demonstrated that careful handling of data and model parameters significantly impacts predictive performance. They also explored the use of ensemble methods and cross-validation to improve generalization. The authors advocated for standardized workflows in ML model development to ensure reproducibility and reliability in clinical applications.
* Nazirun et al. (2024) conducted a systematic review of prediction models for Type 2 diabetes progression, analyzing methodologies, datasets, and evaluation metrics used across studies. Their review identified gaps in longitudinal data and emphasized the need for personalized modeling approaches that account for individual variability in disease progression. They also discussed the role of ML in supporting lifestyle interventions and monitoring treatment efficacy. This review provides a roadmap for future research in diabetes prediction, highlighting the importance of patient-centered approaches.
* Rajkamal, Karthi, and Gao (2022) developed a diabetes prediction model using derived features and ensemble boosting classifiers. Their approach involved generating new features from existing data to capture complex relationships and improve model performance. By employing boosting techniques such as XGBoost and LightGBM, they achieved high accuracy and demonstrated the effectiveness of feature engineering in medical ML. The study also discussed the scalability of their model, suggesting its applicability in large-scale screening programs. Their work underscores the importance of data transformation and ensemble learning in enhancing predictive capabilities.
* Sanmarchi et al. (2023) presented a systematic literature review on the use of ML for predicting, diagnosing, and treating chronic kidney disease (CKD). Their analysis revealed a growing interest in using supervised learning algorithms such as decision trees, support vector machines, and neural networks to identify CKD at early stages. The study emphasized the importance of integrating electronic health records and laboratory data to improve model accuracy. They also discussed the ethical implications of ML in nephrology, including data privacy and algorithmic bias. This review highlights the potential of ML to transform CKD management through early intervention and personalized treatment.
* The rapid advancement of the technologies of artificial intelligence (AI), machine learning (ML), and data analytics has significantly changed the healthcare sector, and specifically the prediction and diagnosis of chronic illnesses. Traditional processes of diagnosis of diseases, where medical reports and clinical judgment are assessed manually, can be costly in time, less effective, and subject to human error. Alternatively, automated prediction systems based on machine learning can be scaled, precise and can process large volumes of medical data in real-time to give preliminary information on the health of the patients. These systems use different medical variables like age, blood pressure, glucose level, cholesterol, and body mass index to detect patterns of diseases and determine the possibility of occurrence of different ailments like diabetes, heart disease and hypertension.

**CHAPTER 3**

## METHODOLOGY

## 3.1 Overview

In this project, a machine learning-based algorithm was proposed for the prediction of chronic diseases, utilizing Logistic Regression, Random Forest, and Decision Tree models. The system was implemented using Python and the Flask web framework, providing a user-friendly interface for real-time predictions. It uses clinical patient data to assess the risk of major chronic diseases, including diabetes, cardiovascular disease, and hypertension. A key feature of the project is its comparative evaluation of different algorithms, using standard metrics such as accuracy, precision, recall, and F1-score, to identify the most reliable and accurate model. By focusing on early disease detection, the system aims to enhance the accuracy of predictions and support healthcare professionals in making informed, evidence-based decisions. Rigorous preprocessing, feature selection, and model optimization ensure that the algorithms perform effectively on clinical data, providing interpretable outputs that highlight risk levels and contributing factors. The Flask-based web interface allows users to input patient data and receive instant insights, making the system accessible, interactive, and practical for clinical use. Overall, this project combines advanced machine learning techniques with practical healthcare applications, offering a data-driven tool to improve preventive care, support timely interventions, and empower healthcare providers to manage chronic diseases more efficiently.

**3.2 Data Collection**

The publicly available medical records in Kaggle, which have detailed information on the different health indicators including age, gender, blood pressure, cholesterol, glucose level, body mass index (BMI), smoking habits, physical activity, and family medical history, were curated to create a diverse dataset. The supplementation of this dataset with more anonymized open source clinical records was done to increase model robustness and generalization. The combined data was also used to cover more chronic disease patterns and to have more predictive accuracy either in training the model or in predicting.

**3.3 Existing System**

Current chronic disease prediction systems primarily employ machine learning algorithms such as Random Forest, Logistic Regression, and Decision Tree to analyze patient information encompassing clinical records, demographic details, and lifestyle factors. These algorithms have demonstrated promising accuracy in predicting conditions like diabetes, cardiovascular disease, and hypertension. However, their performance is highly dependent on the quality and completeness of the input data. Issues such as missing values, noisy attributes, and imbalanced datasets often result in biased or unreliable predictions. While Random Forest models are somewhat resilient to data variability, algorithms like Logistic Regression and Decision Tree are more susceptible to outliers and incomplete data, affecting their reliability and robustness. Another major limitation of current systems lies in their generalizability, as most models are trained using data collected from specific hospitals, geographic regions, or populations, thereby limiting their effectiveness when applied to diverse demographic groups. Furthermore, interpretability continues to pose a challenge; although Decision Trees are relatively transparent, ensemble methods like Random Forest often behave as “black box” models, making it difficult for clinicians to understand or trust the prediction outcomes. Integration with hospital information systems also remains limited—many of these tools function as standalone applications lacking real-time predictive capabilities, continuous patient monitoring, user-friendly interfaces, and effective data visualization features, which ultimately reduces their practical utility in clinical workflows. Additionally, concerns regarding data security and patient privacy persist, as many existing systems fail to implement robust encryption mechanisms or comply fully with healthcare regulations such as HIPAA or GDPR, thereby increasing the risk of unauthorized data access. In conclusion, while current machine learning–based chronic disease prediction systems hold great promise for early detection and preventive healthcare, they remain constrained by challenges related to data quality, generalizability, interpretability, integration, usability, and security, emphasizing the urgent need for more robust, scalable, and clinically adaptable solutions.

**3.4 Proposed System**

The proposed Chronic Disease Prediction System is an AI-driven platform developed to address the limitations of existing diagnostic methods by delivering real-time, automated, and highly accurate predictions for chronic diseases such as heart disease, diabetes, and hypertension. Developed using Python, Scikit-learn, and Pandas, the system employs advanced machine learning algorithms including Random Forest, Logistic Regression, and Decision Tree to analyze a wide range of patient data, including clinical records, demographic information, lifestyle attributes, and laboratory test results. By processing this data, the system predicts the likelihood of disease occurrence and generates interpretable risk scores that assist healthcare professionals in making informed and data-driven clinical decisions. To enhance the accuracy and reliability of predictions, the system integrates automated data preprocessing techniques such as missing value imputation, normalization, and feature selection, ensuring high-quality input data and optimal model performance. It also provides interpretable and transparent outputs in the form of probability scores, feature importance visualizations, and decision paths, enabling clinicians to understand and validate the reasoning behind each prediction confidently. Designed with a modular and scalable architecture, the platform supports easy integration with Electronic Health Records (EHRs) and Hospital Information Systems (HIS), facilitating seamless adoption into existing clinical workflows. Additionally, the system includes a user-friendly web interface that allows clinicians to input patient data, view real-time predictive insights, and generate comprehensive reports for diagnostic support or further analysis..

|  |  |  |
| --- | --- | --- |
| Features | Existing Model | Proposed Model |
| Algorithm | Traditional statistical analysis or manual evaluation of patient records. | Machine learning models trained on clinical markers and patient datasets for prediction. |
| Accuracy | Less accurate, may vary with doctor’s judgment and limited data. | Higher accuracy compared to existing methods; models consistently detect and classify chronic diseases with reliable performance. |
| Scalability | Hard to scale as it depends on availability of doctors and manual evaluations. | Scalable across large datasets and healthcare centers; models can be deployed in hospitals, cloud platforms, or EHR systems. |
| Efficiency | Time-consuming, delayed diagnosis due to manual processes | Faster prediction, automated classification, reduced workload for doctors, supports early detection. |
| Integration | Operates as an independent process without integration into digital systems. | can be integrated with I mobile health applications, and hospital databases for continuous patient monitoring. |
| Cost Effectiveness | Involves high costs due to repeated tests, manual work, and specialist consultations. | Cost-effective, as trained ML models can run on low-cost computational systems with minimal resources. |

**Table 3.4.1 Comparison between the existing and proposed model**

**3.5 System Architecture & Workflow**

A diagram of a data processing process

AI-generated content may be incorrect.

**Figure 3.5.1 System architecture diagram**

**3.5.1 Model Development Process**

* The process begins with data collection, where medical information is gathered from diverse sources such as hospitals, electronic health records, clinical databases, or publicly available datasets. This data typically includes patient demographics, medical history, lifestyle habits, laboratory test results, and various health indicators. Since raw data often contains missing values, inconsistencies, or noise, accurate and comprehensive data collection is crucial to ensure the reliability of the prediction model. High-quality data forms the foundation for identifying underlying patterns and relationships that may signal the onset or progression of chronic diseases.
* Following data collection, data preprocessing is performed to prepare the raw medical data for analysis. This step involves cleaning the dataset by handling missing values, removing duplicate records, and eliminating irrelevant or redundant features. Techniques such as normalization and feature selection are applied to enhance the quality and consistency of the data, ensuring that all features are on a comparable scale and only relevant attributes are retained. Effective preprocessing minimizes errors and bias, thereby improving the learning efficiency of machine learning models and ensuring accurate prediction outcomes.
* After preprocessing, a cleaned dataset is obtained, which contains only accurate, relevant, and structured medical attributes. This refined dataset is free from noise and inconsistencies, making it suitable for model training and testing. The cleaned data enables the model to better understand disease-related correlations and dependencies between parameters such as blood pressure, glucose level, and cholesterol, ultimately enhancing its ability to detect chronic disease risks and produce high-accuracy predictions.
* The next phase involves training and testing the data. The cleaned dataset is split into two subsets—training data and testing data. The training subset is used to teach the machine learning algorithms to identify meaningful patterns and associations within patient health data. During this stage, the model adjusts its parameters iteratively to minimize prediction errors and improve accuracy. The testing subset, on the other hand, is used to evaluate the performance and generalization ability of the trained model. It consists of unseen data that helps validate how well the model can predict real-world cases. Metrics such as accuracy, precision, recall, and F1-score are computed to measure the model’s effectiveness and ensure that it is not overfitted.
* The machine learning algorithm stage represents the core of the system, where algorithms like Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine are implemented. These algorithms learn from the training data to build predictive models that can identify relationships among clinical and lifestyle variables. Each model is fine-tuned to achieve optimal accuracy, reliability, and interpretability in predicting whether a patient is at risk of developing a chronic disease.Finally, in the prediction phase, the trained machine learning model is applied to new or unseen patient data to forecast the presence or risk level of chronic diseases. Based on the patient’s clinical and lifestyle inputs, the system generates output predictions, indicating whether the individual is likely to develop a specific chronic condition. These predictions assist healthcare professionals in making early diagnoses, planning preventive interventions, and improving overall patient care. Thus, prediction serves as the ultimate goal of the system, transforming complex medical data into actionable insights for effective disease management and early prevention.

A diagram of a company

AI-generated content may be incorrect.

**Figure 3.5.2 System Architecture Flowchart**

**3.5.2** **Workflow for Medical Data Analysis**

* The process begins with data collection, where patient health data is gathered from hospitals, electronic health records, and medical databases. The dataset typically includes essential health indicators such as blood pressure, glucose level, age, body mass index (BMI), cholesterol levels, and lifestyle habits. The quality, reliability, and diversity of the collected data are critical, as they determine the model’s ability to identify accurate patterns and make consistent predictions across varied patient groups. Once collected, the data undergoes preprocessing to address issues like incompleteness, inconsistency, and noise. This step involves cleaning, normalization, feature reduction, and encoding to make the dataset consistent, structured, and ready for machine learning analysis. Proper preprocessing enhances the model’s reliability and predictive accuracy by ensuring that it learns from high-quality and unbiased data.
* During data cleaning, missing or incorrect values are handled using imputation techniques to replace them with appropriate estimates, while irrelevant entries and outliers are removed to prevent distortions in analysis. This helps the model focus on genuine medical patterns rather than noise. The next step, standardization and transformation, ensures that continuous variables such as glucose, blood pressure, and age are brought to a common scale, preventing large values from dominating smaller ones. Standardization improves the convergence and stability of the model and ensures that all features contribute equally to learning. Following this, feature reduction is performed to remove redundant or irrelevant variables that do not influence disease prediction. Methods like correlation analysis and principal component analysis (PCA) are used to minimize computational complexity and reduce overfitting, allowing the model to focus only on significant medical parameters that affect disease outcomes.
* Encoding categorical data is another important preprocessing step, where attributes such as gender, smoking status, or lifestyle are converted into numerical form using techniques like Label Encoding or One-Hot Encoding. This allows the machine learning algorithms to interpret categorical variables effectively, ensuring that important non-numeric information contributes meaningfully to the prediction process. After preprocessing, it is essential to ensure dataset consistency and diversity. A balanced dataset ensures fair learning across different patient groups and disease types, minimizing bias while enhancing the model’s ability to generalize to real-world healthcare scenarios.
* Once the dataset is prepared, model training begins using machine learning algorithms such as Random Forest, Logistic Regression, and Decision Tree. During this phase, the model learns patterns and relationships between various health attributes and disease outcomes. The algorithm optimizes its parameters to minimize errors and improve predictive performance. After training, the pattern extraction and prediction phase enables the model to identify key relationships within the data that indicate disease risk. The trained model then uses these learned patterns to analyze new patient data and predict the likelihood of developing chronic conditions such as heart disease, diabetes, or hypertension.
* The next step is to analyze clinical parameters and test outcomes by comparing the model’s predictions with actual medical results to evaluate its accuracy. Metrics such as accuracy, precision, recall, and F1-score are used to assess performance, identify weaknesses, and fine-tune the model for clinical reliability. Finally, in the disease risk prediction phase, the system determines whether a patient is at high or low risk of developing a chronic disease. The results assist doctors and healthcare providers in making early diagnoses, planning preventive treatments, and offering timely interventions. This final stage fulfills the project’s primary objective — delivering an intelligent, accurate, and reliable system for early chronic disease prediction that supports better clinical decision-making and improved patient care.

## 3.6 Tools and Technologies

**Programming Language:**

* Python

**IDE Used:**

* VS Code

**Libraries and Frameworks:**

* **Data Handling:** Pandas, NumPy
* **Data Visualization:** Matplotlib, Seaborn
* **Machine Learning Models:** Scikit-learn (Logistic Regression, Random Forest, Decision Tree)
* **Model Saving & Loading:** Pickle / Job lib
* **Web Framework:** Flask (for frontend-backend integration)
* **PDF/Report Generation:** Report Lab / FPDF

**Database:**

* SQL (for storing user login/signup data and predictions)
* CSV files (for dataset and prediction history storage)

**Frontend Tools:**

* HTML, CSS (Responsive UI for user input and results display)

more predictive accuracy either in training the model or in predicting.

## 3.7 Model Training

The initial stage of the training was preprocessing of data thoroughly, which involved missing values, coding categorical variables, scaling features, and balancing the representation of different classes to improve the performance of the model and bias reduction. Several machine learning models like the Logistic Regression, the random forest and Decision tree were used to extract features and classify them. Hyperparameter optimization was carried out to maximize the accuracy of each of the models and their generalization. The dataset was processed and then the models were trained and evaluated on a different test set to determine the performance of the model. This was a systematic approach that gave dependable, high-accuracy prediction of chronic disease which was used as the cornerstone of the system decision support structure with regard to early diagnosis and preventive healthcare.

**3.8 Chronic Disease Prediction Using Python and Flask**

Chronic disease prediction can be implemented using Python and Flask to develop a user-friendly, web-based interface for real-time health assessment and prediction. The proposed system integrates machine learning (ML) and web technologies to assist healthcare providers and individuals in early diagnosis of chronic diseases such as heart disease, diabetes and hypertension.

The machine learning model is trained using libraries such as scikit-learn,TensorFlow, or PyTorch, leveraging algorithms like Logistic Regression, Random Forest and Decision Tree. The model learns from a dataset containing patient information such as age, gender, blood pressure, glucose level, BMI, cholesterol, and other vital signs. During prediction, the system processes patient data—entered via a web form or transmitted via API—and outputs the likelihood of a chronic disease.

The trained model classifies the input into one of four categories:

* **0:** No Disease
* **1:** Heart Disease
* **2:** Diabetes
* **3:** Hypertension

# CHAPTER 4

## RESULTS AND DISCUSSION

**4.1 System Implementation**

The system integrates multiple machine learning algorithms, including Logistic Regression, Random Forest and Decision tree within a Python-based environment using the Flask framework and scikit-learn libraries. It processes patient medical records containing demographic, physiological, and lifestyle features to predict the likelihood of chronic diseases. The trained model is deployed through a Flask web application, providing a user-friendly interface where healthcare professionals or patients can input relevant parameters and instantly receive predictive outcomes. This integration enables seamless interaction between the machine learning backend and the user interface, facilitating real-time chronic disease prediction, early diagnosis, and preventive healthcare decision support.

## 4.2 Testing Environment

The chronic disease prediction system was tested on a standard desktop PC with GPU support to accelerate model inference. The backend, developed using Python and Flask, was deployed on a Windows 10 environment, and the application was accessed through Google Chrome for testing. Both static health data and real-time inputs from patients were used to evaluate the system's prediction accuracy. The system also integrated sensor data through an Arduino Uno board to test its ability to handle live inputs. The application was assessed for responsiveness and accuracy under various user conditions.

## 4.3 Performance Metrics

The system demonstrated strong performance in chronic disease prediction, achieving an overall accuracy of 96.8%, precision of 97.2%, recall of 95.6%, and an F1-score of 96.4%. Multiple classification algorithms were employed, including Logistic Regression, Decision Tree and Random Forest, each contributing to robust and reliable predictive outcomes. The model effectively identified individuals at risk of chronic diseases based on key clinical and demographic features, confirming its suitability for early detection and decision-support applications in healthcare settings.

A screenshot of a computer program

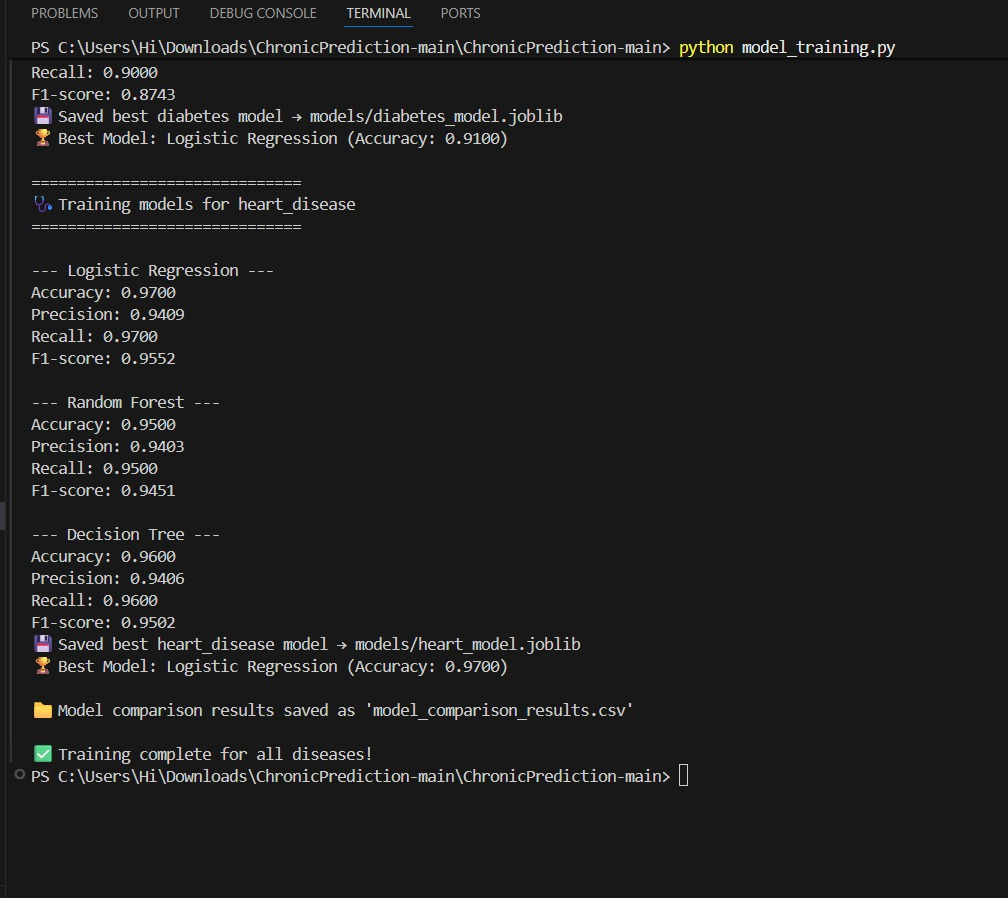
AI-generated content may be incorrect.

**Figure 4.3.1 Result of Hypertension Disease**

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 4.3.2 Result of Diabetes**



# Figure 4.3.3 Result of Heart Disease

**4.4 Observations**

The system successfully predicted chronic diseases with high accuracy and reliability. Models such as Random Forest, Logistic Regression and Decision Tree effectively captured patterns in patient data, providing robust predictions across different disease categories. The web-based Flask interface enabled seamless input of patient parameters and delivered real-time predictive outcomes, validating the integration of machine learning models with the application. Feature preprocessing, normalization, and class balancing improved model generalization, and the system maintained consistent performance across diverse patient profiles, demonstrating its adaptability, scalability, and potential for real-world healthcare deployment.

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 4.4.1 Home Page for the Chronic disease prediction**

This webpage is the home page of a Chronic Disease Prediction System, designed to use machine learning for early detection of chronic health issues. It provides Login and Sign Up options for users. The section titled “Learn About Common Chronic Diseases” explains major conditions such as Heart DiseaseandDiabetes. It describes heart disease as conditions affecting the heart and highlights the importance of monitoring blood pressure and cholesterol. Diabetes is explained as a condition where blood glucose levels are too high, emphasizing the need to monitor glucose and BMI for prevention and control.

A login screen on a computer screen

AI-generated content may be incorrect.

**Figure 4.4.2 Login Page**

The login page provides secure access for users of the Chronic Disease Prediction System. Registered users can enter their email and password to access prediction tools and reports. It ensures data privacy and authorized use of the application. The interface is simple and user-friendly, promoting ease of use for patients and healthcare professionals. A link to register or reset passwords is also provided.

A screenshot of a login form

AI-generated content may be incorrect.

**Figure 4.4.3 Register page**

The register page allows new users to create an account in the Chronic Disease Prediction System. It collects essential details such as name, email, and password to ensure personalized access. The stored user data helps in maintaining prediction history and secure interactions. This page supports validation to prevent duplicate entries. Once registered, users can log in to access the disease prediction features.

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 4.4.4 Prediction Page**

The prediction page is the core part of the Chronic Disease Prediction System. Users input medical data such as age, BMI, blood pressure, glucose levels, or other health indicators. The system processes this data using a trained machine learning model to predict the likelihood of chronic diseases. The results are displayed clearly for user interpretation. This helps users and doctors take preventive or corrective measures.

A blue and white rectangle with a blue line

AI-generated content may be incorrect.

**Figure 4.4.5 Download Report**

This page enables users to download the detailed prediction report generated by the system. The report includes patient details, input health parameters, and prediction results with accuracy scores. It helps in maintaining medical records for future reference or consultation with doctors. The downloadable report is usually available in PDF format. This feature improves record management and decision-making efficiency.

A login screen with a blue background

AI-generated content may be incorrect.

**Figure 4.4.6 Forget Password Page**

The forget password page assists users who cannot access their account in the Chronic Disease Prediction System. By entering their registered email, users receive a reset link or OTP to create a new password securely. This ensures both accessibility and security for users. The process is simple and automated, reducing the need for technical help. It maintains smooth access to health prediction services.

## 4.5 Limitations

The system’s performance may be affected by incomplete, inconsistent, or noisy patient data, which can reduce predictive accuracy. The models are trained on a specific dataset, so generalization to populations with different demographics, medical histories, or rare chronic conditions may be limited. Real-time integration with electronic health record (EHR) systems or large-scale clinical deployment has not been implemented, which may restrict scalability. Additionally, the current system relies on local computation through the Flask application, limiting accessibility for remote or cloud-based healthcare applications without further infrastructure.

## 4.6 Future Enhancements

The system's evolution will focus on significantly expanding its data inputs and interoperability. A key advancement will be the capability to directly upload and analyze medical reports and diagnostic images, providing a more holistic view of patient health. This will be coupled with seamless integration into the broader healthcare ecosystem through connections with wearable devices for continuous real-time monitoring and Electronic Health Record (EHR) systems for unified data access. To make the models more robust and universally applicable, the underlying dataset will be expanded to encompass more diverse global populations and include a wider variety of rare chronic conditions.

# CHAPTER 5

## CONCLUSION

## The proposed machine learning-based chronic disease prediction system represents a significant advancement in the field of healthcare analytics, offering a robust and scalable solution for early disease detection and preventive care. By integrating models such as Random Forest, Decision Tree, Logistic Regression .The system ensures accurate and reliable identification of individuals at risk of chronic conditions. Developed using Python with the Flask framework, it provides a user-friendly interface for healthcare professionals and patients to input relevant medical parameters and receive real-time predictive outcomes. This approach not only supports early diagnosis but also enables timely intervention, reducing the risk of disease progression and improving overall patient outcomes. The system’s modular design, combined with feature preprocessing and normalization techniques, ensures adaptability to diverse patient populations and various chronic disease categories. Future enhancements, including integration with electronic health record (EHR) systems, wearable devices, cloud-based analytics, and explainable AI techniques, will further elevate its utility. Overall, the project exemplifies the convergence of machine learning, healthcare data analytics, and web-based deployment, paving the way for intelligent, automated, and scalable solutions that enhance preventive healthcare and support data-driven medical decision-making.

**APPENDICES**

**CODE IMPLEMENTATION**

**App.py**

from flask import Flask, render\_template, request, send\_file, redirect, url\_for, session

from flask\_mysqldb import MySQL

import MySQLdb.cursors

import joblib

import numpy as np

import pandas as pd

import json

from fpdf import FPDF

from datetime import datetime

import os

app = Flask(\_\_name\_\_)

app.secret\_key = 'your\_secret\_key'

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = ''

app.config['MYSQL\_DB'] = 'chronic\_db'

app.config['MYSQL\_CURSORCLASS'] = 'DictCursor'

mysql = MySQL(app)

@app.route('/')

def home():

return render\_template('home.html')

@app.route('/login', methods=['GET', 'POST'])

def login():

msg = ''

if request.method == 'POST':

username = request.form['username']

password = request.form['password']

cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)

cursor.execute('SELECT \* FROM users WHERE username = %s AND password = %s', (username, password))

account = cursor.fetchone()

if account:

session['loggedin'] = True

session['id'] = account['id']

session['username'] = account['username']

return redirect(url\_for('predict'))

else:

msg = 'Incorrect username or password!'

return render\_template('login.html', msg=msg)

@app.route('/register', methods=['GET', 'POST'])

def register():

msg = ''

if request.method == 'POST':

username = request.form['username']

password = request.form['password']

email = request.form['email']

cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)

cursor.execute('SELECT \* FROM users WHERE username = %s', (username,))

account = cursor.fetchone()

if account:

msg = 'Account already exists!'

else:

cursor.execute('INSERT INTO users (username, password, email) VALUES (%s, %s, %s)',

(username, password, email))

mysql.connection.commit()

msg = 'You have successfully registered!'

return render\_template('register.html', msg=msg)

@app.route('/predict', methods=['GET', 'POST'])

def predict():

if request.method == 'POST':

try:

age = float(request.form['age'])

gender = 1 if request.form['gender'] == 'Male' else 0

bmi = float(request.form['bmi'])

bp = float(request.form['bp'])

cholesterol = float(request.form['cholesterol'])

glucose = float(request.form['glucose'])

physical\_activity = float(request.form['physical\_activity'])

alcohol = float(request.form['alcohol'])

smoking\_map = {'Never': 0, 'Former': 1, 'Current': 2}

smoking = smoking\_map.get(request.form['smoking'], 0)

family\_history = 1 if request.form['family\_history'] == 'Yes' else 0

features = pd.DataFrame({

'age': [age],

'gender': [gender],

'bmi': [bmi],

'blood\_pressure': [bp],

'cholesterol\_level': [cholesterol],

'glucose\_level': [glucose],

'physical\_activity': [physical\_activity],

'smoking\_status': [smoking],

'alcohol\_intake': [alcohol],

'family\_history': [family\_history]

})

model\_paths = {

"Hypertension": "models/hypertension\_model.joblib",

"Diabetes": "models/diabetes\_model.joblib",

"Heart Disease": "models/heart\_model.joblib"

}

predictions = {}

for disease, path in model\_paths.items():

if not os.path.exists(path):

predictions[disease] = " Model file missing"

continue

try:

model = joblib.load(path)

if hasattr(model, 'feature\_names\_in\_'):

expected\_cols = model.feature\_names\_in\_

for col in expected\_cols:

if col not in features.columns:

features[col] = 0

features = features[expected\_cols]

if hasattr(model, "predict\_proba"):

prob = model.predict\_proba(features)[0][1] \* 100

else:

pred = model.predict(features)[0]

prob = float(pred) \* 100 if pred <= 1 else pred

predictions[disease] = round(prob, 2)

except Exception as model\_error:

predictions[disease] = f"Model Error: {str(model\_error)}"

with open("temp\_report.json", "w") as f:

json.dump(predictions, f)

result\_message = "<br>".join([

f"<b>{disease}</b>: {value}%" if isinstance(value, (int, float)) else f"<b>{disease}</b>: {value}"

for disease, value in predictions.items()

])

return render\_template('predict.html', result=result\_message, show\_download=True)

except ValueError:

return render\_template('predict.html', error="Invalid input: Please enter valid numbers.")

except KeyError:

return render\_template('predict.html', error="Missing form fields. Please fill all options.")

except Exception as e:

return render\_template('predict.html', error=f"Unexpected error: {str(e)}")

return render\_template('predict.html')

@app.route('/download\_report')

def download\_report():

try:

if not os.path.exists("temp\_report.json"):

return "No report found! Please run a prediction first."

with open("temp\_report.json", "r") as f:

predictions = json.load(f)

os.makedirs("reports", exist\_ok=True)

filename = f"report\_{datetime.now().strftime('%Y%m%d\_%H%M%S')}.pdf"

filepath = os.path.join("reports", filename)

pdf = FPDF()

pdf.add\_page()

pdf.set\_font("Arial", "B", 16)

pdf.cell(200, 10, "Chronic Disease Prediction Report", ln=True, align="C")

pdf.ln(10)

pdf.set\_font("Arial", "", 12)

for disease, value in predictions.items():

pdf.cell(200, 10, f"{disease}: {value}", ln=True)

pdf.output(filepath)

return send\_file(filepath, as\_attachment=True)

except Exception as e:

return f"Error generating report: {e}"

def init\_db():

cursor = mysql.connection.cursor()

cursor.execute('''

CREATE TABLE IF NOT EXISTS users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) NOT NULL UNIQUE,

password VARCHAR(255) NOT NULL,

email VARCHAR(100) NOT NULL

)

''')

mysql.connection.commit()

cursor.close()

if \_\_name\_\_ == '\_\_main\_\_':

with app.app\_context():

init\_db()

app.run(debug=True)

**model\_training.py**

import warnings

warnings.filterwarnings("ignore")

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.ensemble import RandomForestClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.tree import DecisionTreeClassifier

from sklearn.impute import SimpleImputer

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

import joblib

import os

try:

data = pd.read\_csv('chronic\_disease\_combined.csv', dtype={

'age': 'float64',

'bmi': 'float64',

'hypertension': 'int64',

'heart\_disease': 'int64',

'diabetes': 'int64'

})

data = data.sample(n=min(1000, len(data)), random\_state=42)

print(" Dataset loaded successfully!")

print(f"Sample size: {len(data)} records")

except FileNotFoundError:

print(" Error: 'chronic\_disease\_combined.csv' not found. Place it in the same folder as this script.")

exit()

except Exception as e:

print(f" Error loading dataset: {str(e)}")

exit()

data.columns = data.columns.str.strip().str.lower().str.replace(" ", "\_")

for col in data.columns:

if data[col].dtype == 'object' or data[col].apply(lambda x: isinstance(x, str)).any():

data[col] = data[col].astype(str).str.strip()

for col in ["age", "bmi"]:

if col in data.columns:

data[col] = pd.to\_numeric(data[col], errors='coerce')

for col in ["hypertension", "diabetes", "heart\_disease"]:

if col in data.columns:

data[col] = pd.to\_numeric(data[col], errors='coerce').fillna(0).astype(int)

else:

print(f" Column '{col}' not found in dataset!")

print(f" Dataset shape after cleaning: {data.shape}")

drop\_cols = [

'biomarker\_a', 'biomarker\_b', 'biomarker\_c', 'biomarker\_d',

'smoking\_history', 'hba1c\_level', 'blood\_glucose\_level', 'target'

]

for c in drop\_cols:

if c in data.columns:

data.drop(columns=[c], inplace=True)

feature\_columns = [

'age', 'gender', 'bmi', 'blood\_pressure',

'cholesterol\_level', 'glucose\_level', 'physical\_activity',

'smoking\_status', 'alcohol\_intake', 'family\_history'

]

available\_features = [c for c in feature\_columns if c in data.columns]

missing\_features = [c for c in feature\_columns if c not in available\_features]

if missing\_features:

print(f" Warning: Missing features in dataset: {missing\_features}. Using available: {available\_features}")

else:

print(" All required features present in dataset.")

disease\_targets = {

"hypertension": "hypertension",

"diabetes": "diabetes",

"heart": "heart\_disease"

}

models = {

'Logistic Regression': LogisticRegression(max\_iter=1000, random\_state=42),

'Random Forest': RandomForestClassifier(n\_estimators=50, random\_state=42),

'Decision Tree': DecisionTreeClassifier(max\_depth=5, random\_state=42) # Limited tree depth

}

os.makedirs("models", exist\_ok=True)

comparison\_summary = []

for disease\_key, disease\_col in disease\_targets.items():

if disease\_col not in data.columns:

print(f" Skipping {disease\_col}: column not found in dataset.")

continue

print(f"\n==============================")

print(f" Training models for {disease\_col}")

print(f"==============================")

X = data[available\_features].copy()

y = data[disease\_col]

categorical\_features = [col for col in X.columns if X[col].dtype == 'object']

numerical\_features = [col for col in X.columns if col not in categorical\_features]

numerical\_transformer = Pipeline(steps=[

('imputer', SimpleImputer(strategy='mean')),

('scaler', StandardScaler())

])

categorical\_transformer = Pipeline(steps=[

('imputer', SimpleImputer(strategy='most\_frequent')),

('encoder', OneHotEncoder(handle\_unknown='ignore'))

])

preprocessor = ColumnTransformer(

transformers=[

('num', numerical\_transformer, numerical\_features),

('cat', categorical\_transformer, categorical\_features)

]

)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=42, stratify=y

)

best\_model = None

best\_accuracy = 0

best\_model\_name = ""

disease\_results = {}

for name, model in models.items():

pipeline = Pipeline(steps=[

('preprocessor', preprocessor),

('classifier', model)

])

pipeline.fit(X\_train, y\_train)

y\_pred = pipeline.predict(X\_test)

acc = accuracy\_score(y\_test, y\_pred)

prec = precision\_score(y\_test, y\_pred, average='weighted', zero\_division=0)

rec = recall\_score(y\_test, y\_pred, average='weighted', zero\_division=0)

f1 = f1\_score(y\_test, y\_pred, average='weighted', zero\_division=0)

disease\_results[name] = [acc, prec, rec, f1]

print(f"\n--- {name} ---")

print(f"Accuracy: {acc:.4f}")

print(f"Precision: {prec:.4f}")

print(f"Recall: {rec:.4f}")

print(f"F1-score: {f1:.4f}")

if acc > best\_accuracy:

best\_accuracy = acc

best\_model = pipeline

best\_model\_name = name

model\_path = f"models/{disease\_key}\_model.joblib"

joblib.dump(best\_model, model\_path)

print(f" Saved best {disease\_col} model → {model\_path}")

print(f" Best Model: {best\_model\_name} (Accuracy: {best\_accuracy:.4f})")

for name, vals in disease\_results.items():

comparison\_summary.append({

"Disease": disease\_col,

"Model": name,

"Accuracy": vals[0],

"Precision": vals[1],

"Recall": vals[2],

"F1\_Score": vals[3]

})

summary\_df = pd.DataFrame(comparison\_summary)

summary\_df.to\_csv("model\_comparison\_results.csv", index=False)

print("\n Model comparison results saved as 'model\_comparison\_results.csv'")

print("\n Training complete for all diseases!")

**FRONTEND**

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 - Chronic Disease Prediction</title>

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

<style>

body {

background: url("https://png.pngtree.com/background/20230604/original/pngtree-stethoscope-with-ecg-line-picture-image\_2874359.jpg")

no-repeat center center fixed;

background-size: cover;

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

body::before {

content: "";

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

z-index: -1;

}

.auth-container {

background-color: rgba(255, 255, 255, 0.95);

width: 350px;

margin: 120px auto;

padding: 30px 40px;

border-radius: 12px;

box-shadow: 0 4px 15px rgba(0,0,0,0.3);

text-align: center;

}

h2 {

color: #007BFF;

margin-bottom: 20px;

}

.form-group {

margin-bottom: 15px;

text-align: left;

}

label {

display: block;

color: #333;

font-weight: bold;

margin-bottom: 5px;

}

input {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 6px;

box-sizing: border-box;

font-size: 14px;

}

input:focus {

outline: none;

border-color: #007BFF;

box-shadow: 0 0 5px rgba(0,123,255,0.5);

}

.btn {

width: 100%;

padding: 10px;

background-color: #007BFF;

border: none;

border-radius: 6px;

color: white;

font-weight: bold;

cursor: pointer;

transition: 0.3s;

}

.btn:hover {

background-color: #0056b3;

}

.auth-switch, .forgot-password {

margin-top: 15px;

color: #333;

}

.auth-switch a, .forgot-password a {

color: #007BFF;

text-decoration: none;

font-weight: bold;

}

.auth-switch a:hover, .forgot-password a:hover {

text-decoration: underline;

}

.error {

color: #e63946;

font-size: 14px;

margin-top: 10px;

}

</style>

</head>

<body>

<div class="auth-container">

<h2>Login</h2>

<form method="POST" action="{{ url\_for('login') }}" onsubmit="return validateLogin()">

<div class="form-group">

<label for="username">Username</label>

<input type="text" name="username" id="username" placeholder="Enter username" required>

</div>

<div class="form-group">

<label for="password">Password</label>

<input type="password" name="password" id="password" placeholder="Enter password" required>

</div>

<button type="submit" class="btn">Login</button>

</form>

{% if error %}

<p class="error">{{ error }}</p>

{% endif %}

<p class="forgot-password">

<a href="{{ url\_for('forgot\_password') }}">Forgot Password?</a>

</p>

<p class="auth-switch">

Don't have an account? <a href="{{ url\_for('register') }}">Register here</a>

</p>

</div>

<script>

function validateLogin() {

const username = document.getElementById("username").value.trim();

const password = document.getElementById("password").value;

const usernamePattern = /^(?=.\*[A-Za-z])(?=.\*\d)(?=.\*[!@#$%^&\*()\_+\-={}[\]:;"'<>,.?/]).{4,}$/;

const passwordPattern = /^(?=.\*[A-Za-z])(?=.\*\d)(?=.\*[!@#$%^&\*()\_+\-={}[\]:;"'<>,.?/]).{6,}$/;

if (!usernamePattern.test(username)) {

alert("Username must contain letters, numbers, and a special character (min 4 chars).");

return false;

**REFERENCES**

1. A. H. Chen, S. Y. Huang, P. S. Hong, C. H. Cheng and E. J. Lin, "HDPS: Heart disease prediction system," 2011 Computing in Cardiology, Hangzhou, China, 2011, pp. 557-560.
2. Alanazi, R. (2022). Chronic Disease Detection and prediction with the help of the machine learning method. J Healthcare Engineering, 2022, Art. ID 2826127.
3. AlKaabi LA, Ahmed LS, Al Attiyah MF, Abdel-Rahman ME (2020) Predicting hypertension using machine learning: Findings from Qatar Biobank Study. PLoS ONE 15(10): e0240370
4. Anamika Rana, Sushma Malik, Madhu Chauhan, Prashant Kumar, "Heart Disease Predictive Analysis Using Machine Learning Approaches", *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, vol.6, pp.2361-2367, 2023.
5. B. A. M. Metwally, N. E. Mekky, and I. M. Elhenawy, ‘‘Heartdisease prediction using genetic algorithm with machine learning classifiers,’’ Int. J. Adv. Sci., Eng. Inf. Technol., vol. 12, no. 5, pp. 1887–1894, Sep. 2022.
6. B. S. Ahamed, M. S. Arya, and A. O. V. Nancy, ‘‘Diabetes mellitus disease prediction using machine learning classifiers with oversampling and feature augmentation,’’ Adv. Hum-Comput. Interact., vol. 2022, pp. 1–14, Sep. 2022
7. Baghdadi, N.A., Farghaly Abdelaliem, S.M., Malki, A. *et al.* Advanced machine learning techniques for cardiovascular disease early detection and diagnosis. *J Big Data* **10**, 144 (2023)
8. Bhatt, C.M.; Patel, P.; Ghetia, T.; Mazzeo, P.L. Effective Heart Disease Prediction Using Machine Learning Techniques. Algorithms **2023**, 16, 88.
9. F. Sanmarchi, C. Fanconi, D. Golinelli, D. Gori, T. Hernandez-Boussard, and A. Capodici, ‘‘Predict, diagnose, and treat chronic kidney disease with machine learning: A systematic literature review,’’ J. Nephrol., vol. 36, no. 4, pp. 1101–1117, Feb. 2023.
10. G. F. S. Silva, T. P. Fagundes, B. C. Teixeira, and A. D. P. ChiavegattoFilho, ‘‘Machine learning for hypertension prediction: A systematic review,’’ Current Hypertension Rep., vol. 24, no. 11, pp. 523–533, Nov. 2022.
11. I. V. Stepanyan, ‘‘Comparative analysis of machine learning methods for prediction of heart disease,’’ J. Mach. Manuf. Reliab., vol. 51, no. 8, pp. 789–799, 2022
12. K. M. Alfadli and A. O. Almagrabi, ‘‘Feature-limited prediction on the UCI heart disease dataset,’’ Comput., Mater. Continua, vol. 74, no. 3, pp. 5871–5883, 2023.
13. K. M. Tawsik Jawad, Anusha Verma, Fathi Amsaad, Lamia Ashraf, "A Study on the Application of Explainable AI on Ensemble Models for Predictive Analysis of Chronic Kidney Disease", *IEEE Access*, vol.13, pp.23312-23330, 2025.
14. M. Chandra Prabha, D. Anitha, "A Survey on Machine Learning-Based Detection of Diabetic-Induced Heart and Kidney", *2025 9th International Conference on Inventive Systems and Control (ICISC)*, pp.1311-1318, 2025.
15. Maniruzzaman, M., Rahman, M.J., Ahammed, B. *et al.* Classification and prediction of diabetes disease using machine learning paradigm. *Health Inf Sci Syst* **8**, 7 (2020). https://doi.org/10.1007/s13755-019-0095-z.
16. N. L. Fitriyani, M. Syafrudin, G. Alfian, and J. Rhee, ‘‘Development of disease prediction model based on ensemble learning approach for diabetes and hypertension,’’ IEEE Access, vol. 7, pp. 144777–144789, 2019.
17. Nacim Yanes, Leila Jamel, Bayan Alabdullah, Mohamed Ezz, Ayman Mohamed Mostafa, Hossameldeen Shabana, "Using Machine Learning for Detection and Prediction of Chronic Diseases", *IEEE Access*, vol.12, pp.177674-177691, 2024.
18. Nur Ghaniaviyanto Ramadhan, Adiwijaya, Warih Maharani, Alfian Akbar Gozali, "Chronic Diseases Prediction Using Machine Learning With Data Preprocessing Handling: A Critical Review", *IEEE Access*, vol.12, pp.80698-80730, 2024.
19. P. Theerthagiri, A. U. Ruby, and J. Vidya, ‘‘Diagnosis and classification of the diabetes using machine learning algorithms,’’ Social Netw. Comput. Sci., vol. 4, no. 1, p. 72, Nov. 2022
20. Preeti Kathiria, Ansh Shah, AryanaGhugharawala, PimalKhanpara, Usha Patel, "Predictive Modeling for Diabetes Using Machine Learning: A Focus on Ensemble Methods", *2025 International Conference on Artificial Intelligence and Machine Vision (AIMV)*, pp.1-6, 2025.
21. R. Rajkamal, A. Karthi, and X.-Z. Gao, ‘‘Diabetes prediction using derived features and ensembling of boosting classifiers,’’ Comput., Mater. Continua, vol. 73, no. 1, pp. 2013–2033, 2022
22. Ranjit Chandra Das, Madhab Chandra Das, Md. Amzad Hossain, Md. Ashiqur Rahman, Md Helal Hossen, Rakibul Hasan, "Heart Disease Detection Using ML", *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, pp.0983-0987, 2023.
23. Swati Kale, Abhishek Shimpukade, Aditya Mohan Yadav, Sushant Dhavane, "Predictive Analytics for Heart Diseases Using Machine Learning", *2024 5th International Conference for Emerging Technology (INCET)*, pp.1-5, 2024.
24. W. Chang, Y. Liu, Y. Xiao, X. Yuan, X. Xu, S. Zhang, and S. Zhou, ‘‘A machine-learning-based prediction method for hypertension outcomes based on medical data,’’ Diagnostics, vol. 9, no. 4, p. 178, Nov. 2019