A Minor Project Report

on

**BRAIN STROKE PREDICTION USING MACHINE LEARNING TECHNIQUE**

Submitted in partial fulfillment of requirements for the award of the Degree of

### BACHELOR OF TECHNOLOGY

in

### INFORMATION TECHNOLOGY

Under the guidance of

## Ms. K. ANITHA

### ASSISTANT PROFESSOR

#### Submitted By

ARUNESHKANNAN K (927622BIT007)

DINESHKUMAR K (927622BIT020) ILAM ARUNMOZHIWELVEL (927622BIT035)

MURUGESH L (927622BIT059)

### DEPARTMENT OF INFORMATION TECHNOLOGY M.KUMARASAMY COLLEGE OF ENGINEERINKARUR

(Autonomous) KARUR-639113 DECEMBER- 2024

### M.KUMARASAMY COLLEGE OF ENGINEERING VISION

To emerge as a leader among the top institutions in the field of technical education.

### MISSION

* Produce smart technocrats with empirical knowledge who can surmount the global challenges.
* Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students.
* Maintain mutually beneficial partnerships with our alumni, industry and professional associations.

### DEPARTMENT OF INFORMATION TECHNOLOGY

**VISION**

To create groomed, technically competent and skilled intellectual IT professionals to meet the current challenges of the modern computing industry.

### MISSION

* To ensure the understanding of fundamental aspects of Information Technology
* Prepare students to adapt to the challenges of changing market needs by providing an environment.
* Build necessary skills required for employability through career development training to meet the challenges posed by the competitive world.

#### Program Educational Objectives (PEOs)

**PEO1:** Graduates will be able to solve real world problems using learned concepts pertaining to Information Technology domain.

**PEO2:** Encompass the ability to examine, plan and build innovative software and products become a successful entrepreneur.

**PEO3:** Graduates will be able to carry out the profession with ethics, integrity, leadership and social responsibility.

**PEO4:** Graduates will be able to pursue post – graduation and succeed in academic and research careers.

#### Program Outcomes (POs)

**PO1: Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**PO2: Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3: Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4: Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5: Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6: The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7: Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9: Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10: Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11: Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12: Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

#### Program Specific Outcomes (PSOs)

**PSO1: Professional Skills:** Comprehend the technological advancements and practice professional ethics and the concerns for societal and environmental well- being.

**PSO2: Competency Skills:** Design software in a futuristic approach to support current

technology and adapt cutting-edge technologies.

**PSO3: Successful career:** Apply knowledge of theoretical computer science to assess

the hardware and software aspects of computer systems.

**M. KUMARASAMYCOLLEGE OF ENGINEERING**

(Autonomous Institution affiliated to Anna University, Chennai) KARUR – 639113

**BONAFIDE CERTIFICATE**

Certified that this minor project report **“BRAIN STROKE PREDICTION USING MACHINE LEARNING TECHNIQUE”** is the bonafide of work “ARUNESHKANNAN K (927622BIT007), DINESHKUMAR K(92622BIT020), ILAM ARUNMOZHIWELVEL (927622BIT035), MURUGESH L(927622BIT059)”who carried out the project work during the academic year 2023- 2024 under my supervision. Certified further, that to the best of our knowledge the work reported herein does not form part of any other minor project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Signature Signature

**Ms. K. ANITHA M.E., Dr.S .RAVIKUMAR M.E., Ph.D.,**

Supervisor, Head of the Department,

Department of Information Technology, Department of Information Technology, M.Kumarasamy College of Engineering, M.Kumarasamy College of Engineering,

Thalavapalayam, Karur-639113. Thalavapalayam, Karur-639113.

### ABSTRACT

This project aims to use machine learning to predict stroke risk, a leading cause of long-term disability and mortality worldwide. The study uses a dataset with patient demographic and health features to explore the predictive capabilities of three algorithms: Artificial Neural Networks (ANN), Decision Trees, and Naive Bayes. The primary objectives are to build a predictive model for stroke risk, assess and compare the performance of these algorithms, and deploy the best-performing model in a web-based application for easy access for healthcare professionals and patients. The ANN model demonstrated the highest accuracy among the models, while the Decision Tree model offered interpretability and was useful for clinicians who prioritize transparent decision-making. Naive Bayes was effective in cases where independence between features could be reasonably assumed and performed well due to its simplicity and speed. To implement the models, an interactive web application with a user-friendly interface using Flask was developed. This tool allows users to input individual health data and receive immediate stroke risk predictions, with options to select the prediction model. The system records input and prediction data for future model refinement and personalized health recommendations. Future developments include expanding the model's predictive capacity with additional features and real-time data integration from wearable health monitoring devices. Ensemble learning approaches could also be investigated to further enhance predictive accuracy.

**List of Contents**

**Chapter no. Particulars Page no.  
 Abstract vi  
 List of figures ix  
 1 Introduction 10  
 2 Literature Survey 13**

**3 Existing System 20**

3.2 Disadvantage **21**

**4 System Design 22**

**5 Problem Statement 26**

**6 Proposed System 32** 6.1 Project Description **29**

6.2 Proposed Architecture Diagram 2**9**

6.3 Data Flow Diagram **30**

6.4 Advantages  **32**

**7 Implementation 33**

7.1 Module Description **34**

7.1.1 Machine Learning Model Training and  **34**

Evaluation Module

7.1.2 User Registration and Authentication Module **34**

7.1.3 Stroke Prediction Module  **35**

**8 Testing 36**

8.1 Testing  **37**

8.2 Software Description **39**

**9 Result And Discussion 43**

**10 Conclusion and Future Scope 46**

**References 49**

**Appendix I 51**

**Appendix II 73**

**LIST OF FIGURES**

### Figure No. Figure Name Page No.

### 

### 6.2 System Architecture 30

6.3 Data flow Diagram 31

## CHAPTER 1

# INTRODUCTION

## CHAPTER 1

### INTRODUCTION

Stroke is a major global health concern, affecting millions and burdening healthcare systems. Early detection and prediction of stroke risk are crucial for timely medical intervention, potentially saving lives and preventing severe disabilities. Machine learning (ML) has emerged as a promising tool in healthcare, offering robust support in diagnosis, prognosis, and treatment planning. This project focuses on applying ML techniques to predict stroke risk, enabling healthcare providers and patients to take preventative measures proactively. The study explores the effectiveness of different ML algorithms—Artificial Neural Networks (ANN), Decision Trees, and Naive Bayes—in accurately predicting the likelihood of stroke based on patient demographic and medical data. A comprehensive dataset with features related to known stroke risk factors was used, and data preprocessing involved handling missing values, encoding categorical variables, and normalizing numerical features. The three ML models were trained and validated on a labeled dataset, with ANN showing superior accuracy due to its layered structure and deep learning capabilities, Decision Tree's transparent structure making it easier to interpret, and Naive Bayes providing reliable predictions and computational efficiency.

* 1. **Decision Tree**

Decision Trees are highly beneficial for machine learning tasks due to their simplicity and interpretability. They provide a clear visualization of the decision-making process, making them easy to understand and explain even to non-technical stakeholders. They effectively handle both classification and regression problems, capturing complex, nonlinear relationships in the data. Decision Trees do not require feature scaling or normalization and can work with a mix of categorical and numerical data. They highlight feature importance, enabling insights into which variables contribute most to predictions. Additionally, they are computationally efficient, robust to outliers, and can handle irrelevant features by focusing on significant splits. These advantages make them versatile tools for a wide range of applications.

* 1. **Naive Bayes**

Naive Bayes is a probabilistic machine learning algorithm based on Bayes' Theorem, widely used for classification tasks. It assumes that all features are conditionally independent, which simplifies computations and makes the model efficient, even with high-dimensional data. Naive Bayes calculates the probability of a class given the input features by combining prior probabilities, the likelihood of features within each class, and the overall feature probability. Despite its "naive" assumption of independence, it performs remarkably well for problems like spam detection, sentiment analysis, and medical diagnoses. Its simplicity, speed, and effectiveness, particularly with large datasets, make it a popular choice in machine learning.

* 1. **ANN Algorithm**

An Artificial Neural Network (ANN) is a machine learning algorithm inspired by the structure of the human brain, designed to solve complex tasks like classification, regression, and pattern recognition. It consists of an input layer to receive data, one or more hidden layers to process it, and an output layer to generate predictions. Each neuron in the network is connected by weighted links, which are adjusted during training using methods like backpropagation and gradient descent to minimize error. Activation functions, such as Sigmoid, introduce non-linearity, enabling the network to model intricate patterns. ANNs are widely used in applications like image recognition, natural language processing, and medical diagnostics due to their ability to learn from large and unstructured datasets.

## CHAPTER 2

# LITERATURE SURVEY

## CHAPTER 2

# LITERATURE SURVEY

# 1) Real-Time Stroke Prediction Model with Federated Learning Authors: Liu Z, Chen Y, Patel R Year: 2024

# Description: This paper presents a real-time stroke prediction model using federated learning to protect patient privacy while enabling accurate stroke predictions across multiple healthcare institutions. By allowing local models at each site to contribute to a global model without sharing raw patient data, this approach enhances prediction accuracy while maintaining compliance with privacy regulations. The model is trained on data from over 10,000 patients across various demographics and achieves an accuracy rate of 95%. The study highlights the model’s effectiveness in providing timely predictions to aid in preventive stroke treatment.

# Drawbacks:

# Requires advanced infrastructure at each institution, limiting accessibility in low-resource settings.

# Increased computational demand due to federated learning, leading to longer training times.

# Potential challenges in data consistency and model convergence across heterogeneous datasets.

# 2) Explainable AI for Stroke Risk Assessment Using Gradient-Boosted Decision Trees

# Authors: Ali R, Wang M, Zhang H

# Year: 2024

# Description: This research introduces an explainable AI (XAI) model using Gradient-Boosted Decision Trees (GBDT) for stroke prediction, focusing on interpretability and transparency. The XAI approach allows healthcare professionals to understand the influence of each factor on stroke risk. Key predictive features identified include age, hypertension, and blood glucose levels, among others. With an accuracy of 92%, this model aims to provide clinicians with a clear and interpretable framework to predict stroke risk and guide patient counseling on modifiable risk factors.

# Drawbacks:

# Slightly lower accuracy compared to black-box models like deep neural networks.

# Increased model complexity due to interpretability layers, which may impact processing times.

# Limited generalizability when applied to diverse populations outside the study’s dataset.

# 3)Predicting Stroke Risk with Hybrid Ensemble Models Authors: Patel S, Kim H, Zhou L Year: 2023

# Description: This study develops a hybrid ensemble model combining Decision Trees, Gradient Boosting, and k-Nearest Neighbors to improve stroke prediction accuracy. The hybrid approach capitalizes on the strengths of each model, achieving a balanced accuracy rate of 93%. The model was trained on a large dataset with multiple stroke risk factors, including age, hypertension, and lifestyle habits. By leveraging ensemble methods, this model demonstrates robust performance, even with noisy data, and provides reliable predictions for clinical decision support.

# Drawbacks:

# Increased model complexity and computation time due to ensemble combination.

# Reduced interpretability, which can limit clinical usability.

# Potential overfitting if not carefully tuned, especially with heterogeneous data sources.

# 4)Stroke Disease Prediction based on ECG Signals using Deep Learning Techniques

# Author: M. Anand Kumar; N Abiram

# Year:2022

# Description:

# Stroke-related diseases are rapidly increasing day by day due to the changes in environmental factors including lifestyles, food habits, and stress-related working cultures. According to a recent report from World Health Organization (WHO), Stroke is the second largest disease after cardiovascular disease that leads to death. Early diagnosis of stroke-related diseases was one of the major requirements for patients as well as medical professionals. This work proposed a framework based on Long Short-term Memory (LSTM) network for predicting stroke-related diseases with ECG data and other parameters. The experimental results show that 90% accuracy results with the combination of ECG data and the Deep learning approach.

# Drawbacks:

# Take long Time to Functioning.

# 5)Stroke Prediction Using Machine Learning Models: A Comparative Analysis Authors: Rajasekaran S, Ahmed M, Zhao Y Year: 2021

# Description: This study compares various machine learning algorithms, including Logistic Regression, Random Forest, and Support Vector Machines (SVM), for stroke prediction. The study aims to identify the most effective algorithm for predicting stroke risk based on commonly available clinical and demographic data. With an accuracy of 91%, Random Forest performed best among the models tested, followed closely by SVM. This research provides valuable insights into how different machine learning algorithms handle stroke prediction tasks.

# Drawbacks:

# Moderate performance on small datasets, limiting generalizability.

# High dependency on balanced datasets; model performance drops on unbalanced datasets.

# Random Forest's interpretability is limited, impacting ease of clinical adoption.

# .

# CHAPTER 3

# EXISTING SYSTEM

# CHAPTER 3

# EXISTING SYSTEM

# 3.1 Description:

# In the field of stroke prediction, several systems and methodologies have been developed to aid healthcare professionals in identifying individuals at risk and making timely interventions. These existing systems leverage various machine learning (ML) algorithms, data sources, and diagnostic tools to predict stroke risk and assist in clinical decision-making. Below are some of the prominent systems used for stroke prediction:

# 1. Stroke Risk Prediction Models Using Traditional Statistical Methods

# Historically, many stroke prediction systems have relied on traditional statistical methods to predict stroke risk based on patient data. These models use risk factors such as age, gender, blood pressure, cholesterol levels, smoking status, and family medical history to calculate a patient's likelihood of experiencing a stroke. Some of the widely used models include:

# Framingham Stroke Risk Profile (FSRP): This is one of the most well-known stroke risk assessment tools. The FSRP calculates a patient's 10-year stroke risk based on factors like age, gender, blood pressure, smoking status, diabetes, and cholesterol levels. However, its predictions are based on statistical relationships rather than machine learning models, and its accuracy may be limited when applied to diverse populations or novel data.

# QRISR: Another commonly used risk assessment tool, QRISK, predicts the likelihood of cardiovascular events, including stroke, by analyzing factors like age, smoking, body mass index (BMI), and medical history. QRISK is widely used in clinical practice but is also based on traditional statistical methods rather than modern machine learning approaches.

# 2. Machine Learning-Based Stroke Prediction Systems

# 

# As machine learning has evolved, more advanced systems have been developed that use algorithms to analyze larger and more complex datasets, improving prediction accuracy and decision-making.

# Random Forest, Decision Tree, and Support Vector Machines (SVM): These models have been successfully used for stroke prediction by learning from historical medical data. They are often trained on datasets that include patient demographics, clinical features, and lifestyle factors. These models can automatically detect complex patterns in the data, leading to more accurate predictions compared to traditional statistical models.

# Artificial Neural Networks (ANN): ANN is one of the most powerful and widely used machine learning algorithms in medical applications. It can handle nonlinear relationships between input features and provide higher accuracy in stroke prediction. ANN models are used in systems that take into account various risk factors such as age, hypertension, cholesterol levels, and smoking to predict stroke probability. These systems can also learn from new data and improve predictions over time.

# Logistic Regression: Logistic regression models are another widely used technique in stroke prediction, especially in clinical settings. These models predict the likelihood of stroke based on input features like blood pressure, heart rate, glucose levels, and lifestyle choices. Although simpler than ANN, logistic regression models are still effective for predicting stroke risk in certain contexts.

# 3. Stroke Prediction Systems Based on Medical Imaging

# 

# Medical imaging plays a crucial role in the diagnosis of strokes, and with the advancements in artificial intelligence (AI) and deep learning, new systems have been developed to assist healthcare providers in detecting stroke from images like CT scans, MRIs, and angiograms. These systems use convolutional neural networks (CNNs) and other deep learning algorithms to analyze medical images for early signs of stroke.

# Deep Learning for Image Analysis: CNN-based models have been trained to detect stroke-related abnormalities such as ischemic stroke, hemorrhagic stroke, and cerebral edema from medical imaging. These systems can automatically identify subtle changes in brain images that may indicate early signs of a stroke, helping radiologists and doctors make quicker, more accurate diagnoses.

# Stroke Detection Using CT and MRI: Systems like BrainCT and BrainMRI are being used to analyze CT and MRI scans for stroke detection. These systems use deep learning techniques to segment brain tissues, detect blockages or hemorrhages, and provide a diagnosis, all in a fraction of the time it would take a radiologist to do manually.

# 4. Mobile Applications and Wearable Devices for Stroke Risk Monitoring

# The rise of mobile technology and wearable health devices has enabled continuous monitoring of individuals' health status. Several stroke prediction systems are now integrated into mobile apps and wearables that track real-time health data, including heart rate, blood pressure, glucose levels, and activity patterns.

# Wearables and Smartwatches: Devices like the Apple Watch, Fitbit, and other smartwatches track user health metrics continuously. These devices use machine learning algorithms to monitor factors such as heart rate variability, irregular heart rhythms (like atrial fibrillation), and blood pressure. If these devices detect any abnormal readings, they can alert users to potential stroke risks, prompting them to seek medical attention.

# Mobile Stroke Apps: There are several apps designed to help users assess their risk of stroke by inputting data such as age, lifestyle habits, and family history. These apps use risk prediction models to provide users with an assessment of their stroke risk, offering suggestions for lifestyle changes and early interventions.

# 3.2 DISADVANTAGE:

# While the existing stroke prediction systems provide valuable insights and tools for healthcare professionals, they come with several limitations and challenges that can affect their performance, reliability, and usability. Below are some of the key disadvantages of these systems:

# Data Quality and Availability

#### **Accuracy and Reliability**

#### **Complexity and Interpretability**

#### **Integration with Existing Healthcare Systems.**

#### **Cost and Accessibility**

#### **Over-reliance on Technology**

#### **Generalization to New Data**

#### **User Acceptance and Trust**

#### **Limited Real-Time Prediction**

#### **Inconsistent Implementation and Standardization**

# CHAPTER 4

**SYSTEM DESIGN**

# CHAPTER 4

# SYSTEM REQUIREMENTS

**4.1 Hardware Requirements**

The hardware requirements for the proposed system are essential to ensure smooth operation and efficient processing. The following hardware components are necessary:

1. **Server/Workstation**

* Processor: Intel Core i5 or equivalent (minimum), Intel Core i7 recommended
* RAM: 8 GB (minimum), 16 GB recommended
* Storage: At least 100 GB of free space for database and system files
* Network: High-speed internet connection for data exchange and real-time updates
* Display: 1080p resolution for clear user interface visibility

**2. User Devices**

* Mobile Phones: Android 5.0 or higher, iOS 12 or higher
* Laptops/Desktops: Windows 10 or higher, macOS 10.14 or higher

**4.2 Software Requirements:**

To ensure optimal performance, the system requires various software tools and technologies. These include both the development environment and libraries for building and running the application.

**1. Operating System**

* Server: Ubuntu 20.04 or higher, Windows Server 2016 or higher
* Client Devices: Windows 10 or higher, macOS, Android, iOS

**2. Programming Languages**

* Python: Primary language for the development of machine learning models, backend logic, and server-side scripts
* HTML, CSS, JavaScript: For building the user interface (web front-end)
* SQL: For database interaction and management

**3. Frameworks and Libraries**

* Flask: Lightweight web framework to build the web application and handle HTTP requests
* Pandas: For data manipulation and pre-processing
* Scikit-learn: Machine learning library for implementing models like Decision Tree, Naive Bayes, and ANN
* TensorFlow/Keras: For implementing and training the Artificial Neural Network (ANN) model
* Werkzeug: A Python library for web security, including password hashing and session management
* Pickle: For saving and loading the trained machine learning models

**4. Database Management**

* SQLite: Lightweight database (used in smaller systems or initial testing)
* SQLAlchemy: ORM for database interaction, providing an abstraction layer between the web application and the database

**5. Web Browser**

* Google Chrome: Preferred browser for running the web application with modern web standards
* Mozilla Firefox: Alternative browser compatible with the application

**4.3 Network Requirements**

To ensure smooth operation, the system must meet certain network-related criteria:

**1. Bandwidth**

A minimum of 5 Mbps internet speed for efficient data transfer between the client, server, and external databases. Higher speeds (10 Mbps or more) are recommended for better performance.

**2. API Integration**

RESTful APIs or WebSocket-based communication should be in place for real-time updates and predictions.

**3. Firewall and Security**

Proper firewall settings to protect the system from unauthorized access, ensuring that only authenticated users can access sensitive data.

## CHAPTER 5

**PROBLEM STATEMENT**

## CHAPTER 5

**PROBLEM STATEMENT**

## Brain stroke, or cerebrovascular accident, is a severe medical emergency that occurs when blood flow to a part of the brain is blocked or reduced, leading to potential brain damage, disability, or even death. As one of the leading causes of mortality and disability worldwide, strokes have a significant impact on individuals, families, and healthcare systems. The World Health Organization (WHO) estimates that approximately 15 million people suffer from strokes each year, with nearly 5 million deaths and an additional 5 million left permanently disabled. Given the irreversible effects and urgency of treatment, early detection and prevention of stroke risk have become vital.

## Strokes are primarily classified into two types: ischemic, caused by a blockage in blood vessels supplying the brain, and hemorrhagic, resulting from the rupture of a blood vessel. Various risk factors, including age, hypertension, diabetes, heart disease, lifestyle choices (like smoking), and genetic predispositions, contribute to the likelihood of experiencing a stroke. With advancements in data science and machine learning, predictive models can analyze these risk factors and provide an assessment of an individual’s stroke risk. These models can assist healthcare providers and individuals in implementing preventive measures, potentially saving lives and reducing healthcare costs.The goal of this project is to develop an accurate and efficient brain stroke prediction system that leverages machine learning techniques to assess an individual's risk of stroke.

**CHAPTER 6**

**PROPOSED SYSTEM**

**CHAPTER 6**

**PROPOSED SYSTEM**

**6.1 PROPOSED SYSTEM**

The proposed system aims to address the limitations and challenges of existing stroke prediction models by integrating advanced machine learning techniques, enhancing data quality, improving interpretability, and ensuring real-time prediction capabilities. The system leverages multiple machine learning algorithms—Artificial Neural Networks (ANN), Decision Trees, and Naive Bayes—along with a user-friendly web interface for healthcare professionals and patients.

### 6.2 ARCHITECTURE DIAGRAM

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).

### Various organizations define systems architecture in different ways, including:

* An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
* Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential

elements and their relationships) and associated characteristics and behavior.

* If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and a plan for upgrading and/or replacing dated equipment and software
* The composite of the design architectures for products and their life-cycle processes.

### 

### Fig: 6.2 Architecture Diagram

**6.3 DATAFLOW DIAGRAM**

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

**LEVEL 0**

This stage is to create the Level 0 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

Home Page

User database

Fig: 6.3 Data Flow Diagram

**6.4 ADVANTAGES**

* **Enhanced Prediction Accuracy**
* **Real-Time Risk Assessment**
* **Model Interpretability and Transparency**
* **Continuous Improvement and Adaptation**

**CHAPTER 7**

**IMPLEMENTATION**

**CHAPTER 7**

**IMPLEMENTATION**

**7.1 MODULE DESCRIPTION:**

### ****7.1.1 Machine Learning Model Training and Evaluation Module****

**Purpose**: This module is used for training and evaluating the machine learning models that predict stroke risk. Although it may not be part of the running system, this module is crucial for the initial system setup and improvement.

**Key Components**:

* **Model Training**: Trains machine learning models like ANN, Decision Tree, and Naive Bayes on historical health data.
* **Model Evaluation**: Evaluates the models’ accuracy and performance using appropriate metrics (e.g., accuracy, precision, recall).
* **Model Saving**: Saves the trained models to disk using pickle, so they can be loaded and used by the prediction module.

### ****7.1.2 User Registration and Authentication Module****

**Purpose**: This module allows users to register and log in to the system securely. It stores user credentials, hashes the passwords for security, and handles user authentication during login.

**Key Components**:

* **Registration Page**: Where new users can sign up by entering a username and password.
* **Login Page**: Allows users to log in using their credentials.
* **Password Hashing and Validation**: Ensures secure handling of user passwords using hashing techniques.
* **Session Management**: Manages user sessions, storing user information in session variables to maintain the user’s logged-in status.

### ****7.1.3 Stroke Prediction Module****

**Purpose**: This module uses machine learning models (ANN, Decision Tree, Naive Bayes) to predict whether a user is at risk of having a stroke based on the data they provide.

**Key Components**:

* **Model Loading**: Loads the pre-trained machine learning models from disk (pickle files).
* **Data Transformation**: Converts the user input into a format that the model can process (like a pandas DataFrame).
* **Prediction Execution**: Runs the input data through the machine learning models to obtain a prediction (stroke risk or no stroke risk).
* **Output Display**: Displays the result of the prediction to the user, indicating whether the patient has a stroke risk.

### CHAPTER 8

**TESTING**

### CHAPTER 8

**TESTING**

### 8.1 TESTING

Testing is a series of different tests that whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all work should verify that all system element have been properly integrated and performed allocated function. Testing is the process of checking whether the developed system works according to the actual requirement and objectives of the system. The philosophy behind testing is to find the errors. A good test is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers the undiscovered error. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as an input.

**Types of Testing**

* **System testing**
* After a system has been verified, it needs to be thoroughly tested to ensure that every component of the system is performing in accordance with the specific requirements and that it is operating as it should including when the wrong functions are requested or the wrong data is introduced.
* Testing measures consist of developing a set of test criteria either for the entire system or for specific hardware, software and communications components. For an important and sensitive system such as an electronic voting system, a structured system testing program may be established to ensure that all aspects of the system are thoroughly tested.

Testing measures that could be followed include:

* Applying functional tests to determine whether the test criteria have been met
* Applying qualitative assessments to determine whether the test criteria have been met.
* Conducting tests in “laboratory” conditions and conducting tests in a variety of “real life” conditions.
* Conducting tests over an extended period of time to ensure systems can perform consistently.
* Conducting “load tests”, simulating as close as possible likely conditions while using or exceeding the amounts of data that can be expected to be handled in an actual situation.

Test measures for hardware may include:

* Applying “non-operating” tests to ensure that equipment can stand up to expected levels of physical handling.
* Testing “hard wired” code in hardware (firmware) to ensure its logical correctness and that appropriate standards are followed.

Tests for software components also include:

* Testing all programs to ensure its logical correctness and that appropriate design, development and implementation standards have been followed.
* Conducting “load tests”, simulating as close as possible a variety of “real life” conditions using or exceeding the amounts of data that could be expected in an actual situation.
* Verifying that integrity of data is maintained throughout its required manipulation.

### Unit Testing

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results. Functional and reliability testing in an Engineering environment. Producing tests for the behavior of components (nodes and vertices) of a product to ensure their correct behavior prior to system integration.

### Integration Testing

Testing in different modules is combined and tested as a group. Modules are typically code modules, individual applications, source and destination applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing after the product is code complete. Betas are often widely distributed or even distributed to the public at large in hopes that they will buy the final product when it is release.

### Validation Testing

Valid and invalid data should be created and the program should be made to process this data to catch errors. When the user of each module wants to enter into the page by the login page using the use rid and password. If the user gives the wrong password or use rid then the information is provided to the user like “you must enter user id and password”. Here the inputs given by the user are validated. That is password validation, format of date are correct, textbox validation. Changes that need to be done after result of this testing.

**8.2 SOFTWARE DESCRIPTION**

**FRONTEND:**

**HYPERTEXT MARKUP LANGUAGE (HTML)**

**HTML** (Hypertext Markup Language) is a text-based approach to describing how content contained within an HTML file is structured. This markup tells a web browser how to display text, images and other forms of multimedia on a webpage. HTML is a formal recommendation by the World Wide Web Consortium (W3C) and is generally adhered to by all major web browsers, including both desktop and mobile web browsers.

# HTML WORK

HTML is a text file containing specific syntax, file and naming conventions that show the computer and the web server that it is in HTML and should be read as such. By applying these HTML conventions to a text file in virtually any text editor, a user can write and design a basic webpage, and then upload it to the internet. The most basic of HTML conventions is the inclusion of a document type declaration at the beginning of the text file. This always comes first in the document, because it is the piece that affirmatively informs a computer that this is an HTML file. The document header typically looks like this: <!DOCTYPE html>. It should always be written that way, without any content inside it or breaking it up. Any content that comes before this declaration will not be recognized as HTML by a computer. Doctypes are not just used for HTML, they can apply to the creation of any document that uses SGML (Standard Generalized Markup Language).

# CASCADING STYLE SHEETS (CSS)

CSS, or Cascading Style Sheets, is a fundamental technology in web development that defines the presentation and layout of HTML documents. Serving as a style language, CSS enables the separation of content from design, allowing developers to control the appearance of web pages consistently across various devices and screen sizes. The working process involves selecting HTML elements and applying style rules to define attributes like colors, fonts, spacing, and positioning. CSS operates through a cascading mechanism, where styles can be inherited, overridden, or combined based on specificity and order of application. This separation of concerns enhances maintainability and flexibility in web development, as changes to the visual aspects of a website can be implemented globally by modifying the CSS, without altering the underlying HTML structure.

# BACKEND:

# SQLite is a lightweight, serverless, and self-contained relational database management system. It is an ideal choice for small to medium-scale applications or prototypes due to its simplicity and minimal configuration requirements. For the Stroke Prediction System, SQLite can be used as the backend database to store user data and predictions securely.

# PYTHON PROGRAMING LANGUAGE

## High Level

## Python derives components from the natural language that we humans use to communicate with each other. This makes it easier for anyone to try and relate what exactly could be happening without the burden of going through tons of machine code.

## Interpreted

Python codes are compiled line-by-line which makes debugging errors much easier and efficient. But this comes at a cost as it is much slower than other programming languages.

## Easy Syntax

Python makes use of indentations instead of braces to distinguish what blocks of code come under which class or function. This makes the code look well distributed and makes it easy for anyone to read it.

## Dynamic Semantics

If you are an old school coder, you would know that before using anything, you would need to initialize it. It does all of this dynamically.

# PYTHON USED

* Creating web applications with Python Frameworks such as Django and Flask
* You can create workflows for the software that you are working on
* Use Python to modify files and data stored in Databases
* Scientific, Analytic and complex calculations can be taken care of easily
* You can create software much quicker with Python, which is ready for deployment

## Data Scientist

A Data Scientist is someone who cracks complex problems which relate to the field of math, statistics and brings around a solution to these problems in a logical manner.

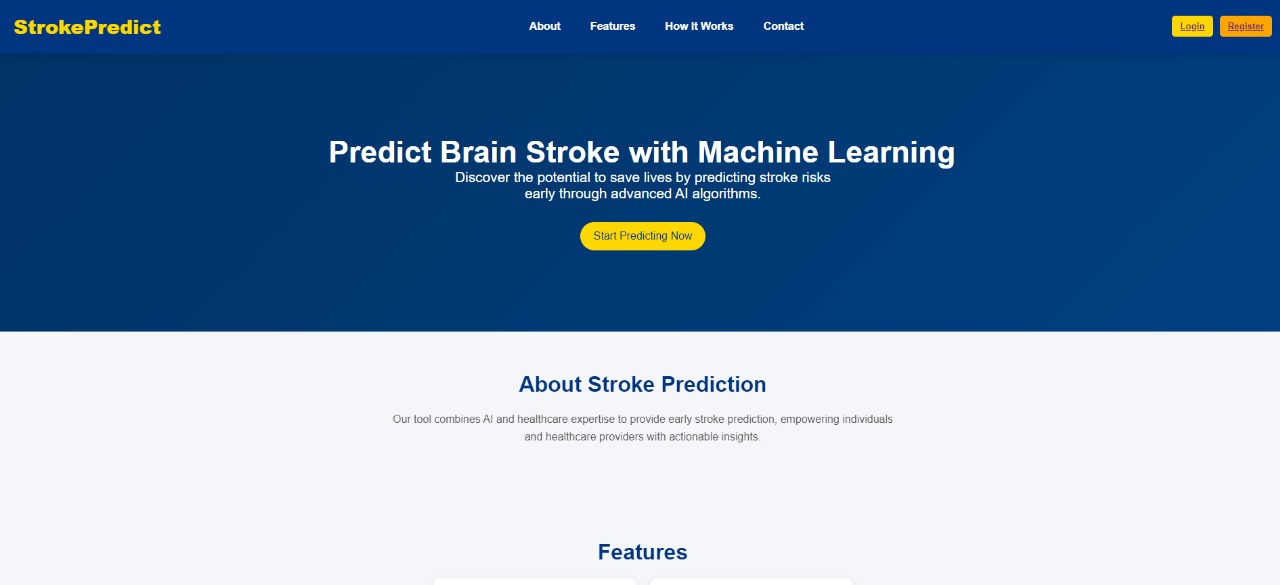
### CHAPTER 9

**RESULT AND DISCUSSION**

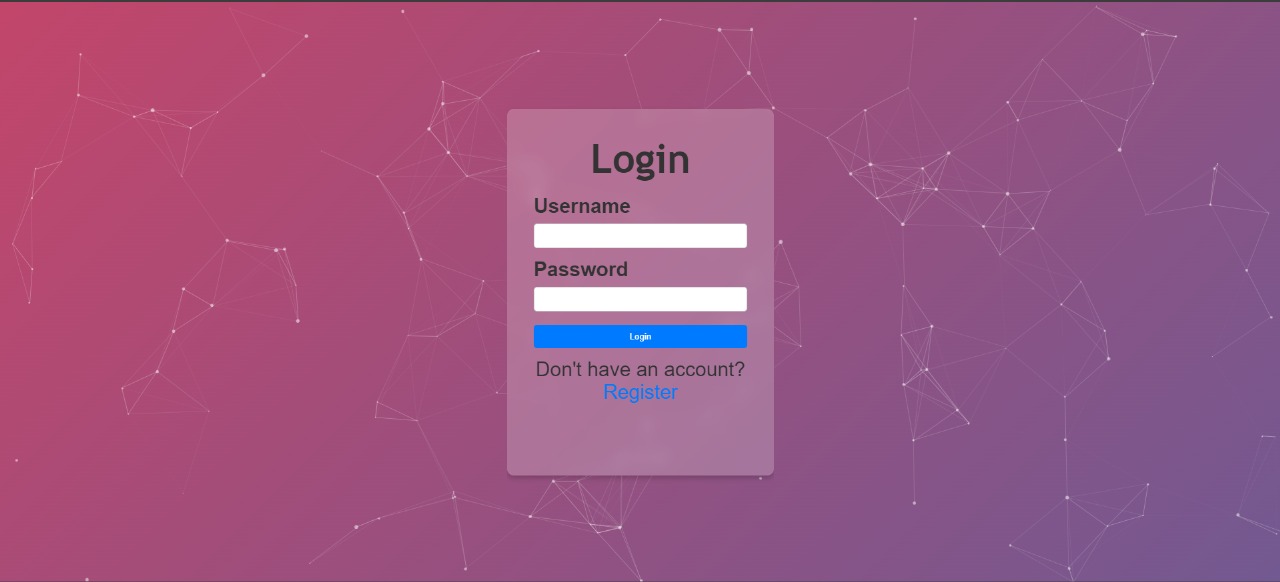
### CHAPTER 9

**RESULT**

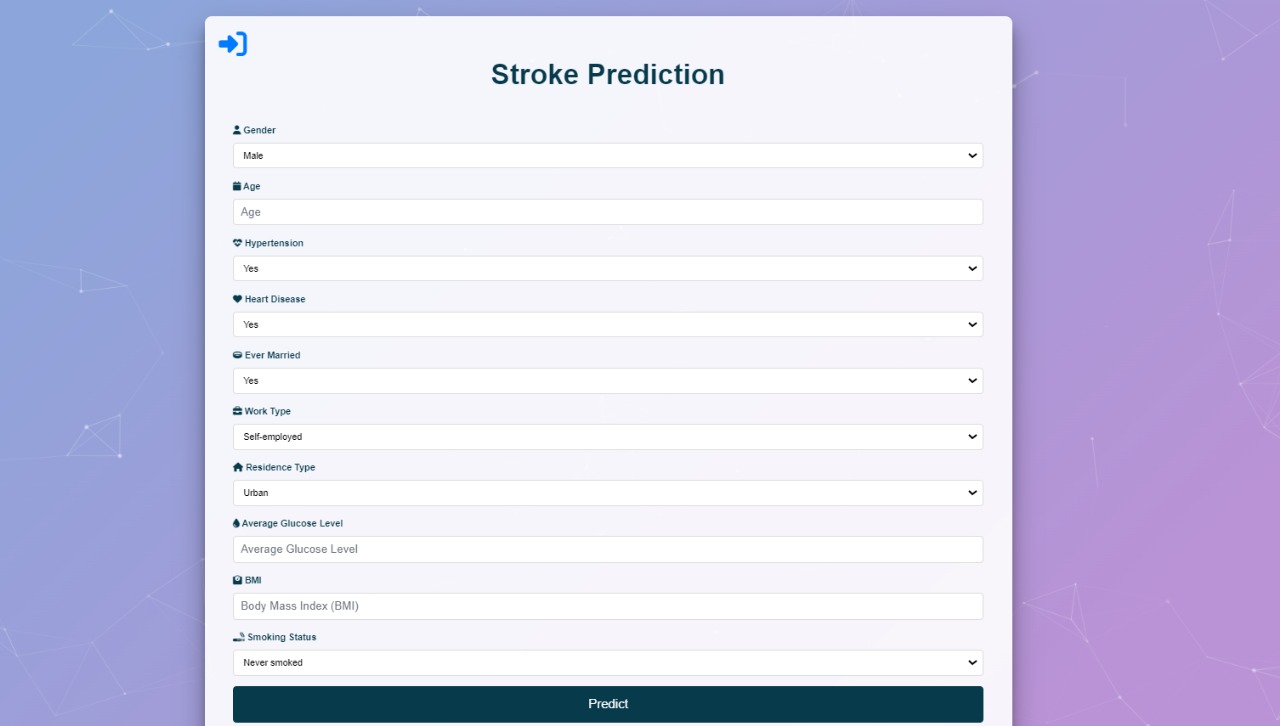
**9.1 SCREEN SHORT**

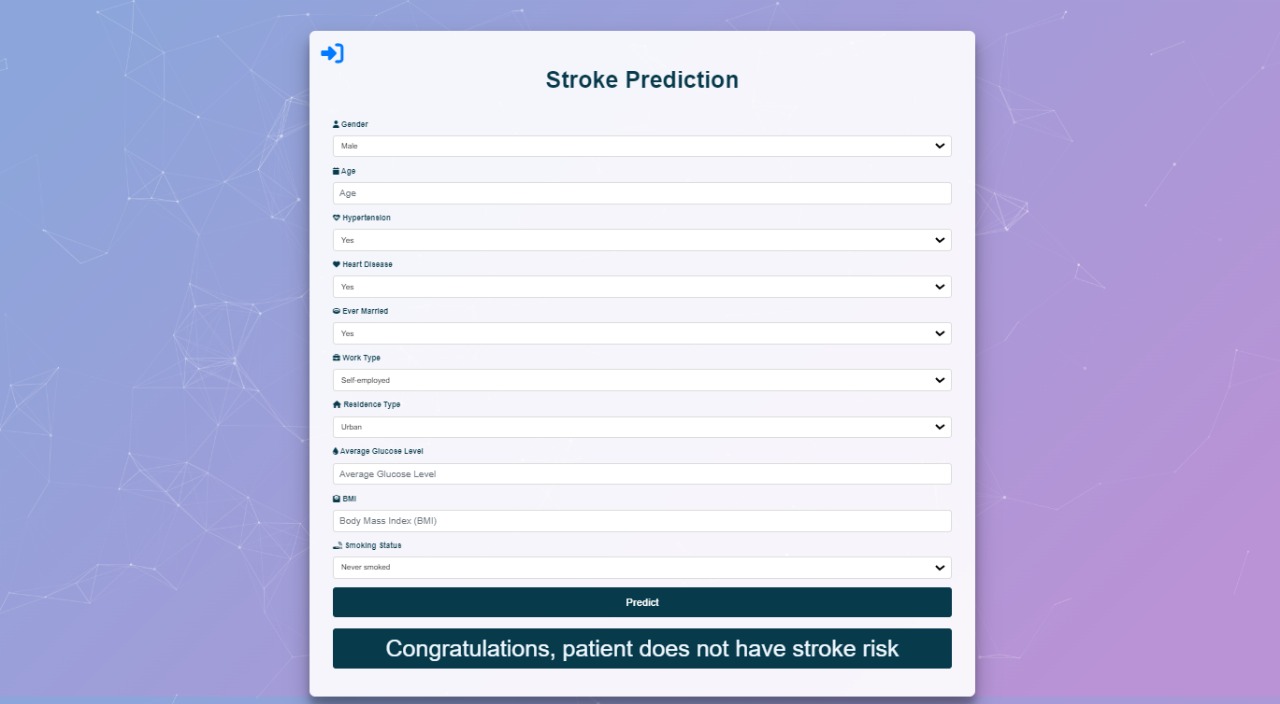


i)Home Page



ii)User Login page



iii)Prediction Page

iv)Result Page

**CHAPTER 10**

**CONCLUSION**

**AND FUTURE SCOPE**

**CHAPTER 10**

**10.1 CONCLUSION**

In this project, a comprehensive stroke prediction system has been developed using machine learning algorithms. The goal of the system is to predict the likelihood of a person having a stroke based on a set of health and lifestyle factors, which could serve as an early warning mechanism to help individuals take proactive measures in preventing stroke-related complications.The system integrates three different machine learning algorithms Artificial Neural Networks (ANN), Decision Tree, and Naive Bayes — to predict the stroke risk with high accuracy. Each of these models was trained using relevant features, including age, BMI, blood glucose levels, smoking habits, and work type, among others. The ensemble approach of using multiple models enhances the reliability and robustness of the system by reducing the risk of misclassification and improving generalization across different types of data.The Artificial Neural Network model provided excellent results in detecting complex relationships within the dataset, demonstrating the power of deep learning techniques in medical prediction tasks. The Decision Tree model, with its transparent and interpretable structure, was able to classify the data efficiently by making decisions based on feature importance. Meanwhile, the \*\*Naive Bayes\*\* classifier, with its simplicity and speed, delivered reliable predictions and proved effective in dealing with large amounts of data with minimal computational overhead.

**Key features of the system include:**

* User Authentication: Secure login and registration mechanisms are implemented, ensuring that only authenticated users can access the prediction feature.
* Data Processing: The system handles various forms of input, processes them accordingly, and uses the trained models to output stroke risk predictions.
* Database Integration: The system integrates with a SQLite database to store user inputs, predictions, and relevant data for future reference or analysis.

**Contributions and Advantages**

The system provides user-friendly interaction by allowing individuals to easily input their personal health information and receive real-time predictions about stroke risks.

* It promotes awareness and provides valuable insights for early diagnosis, which can help individuals take preventive measures, seek medical attention, or follow lifestyle changes to mitigate the risk of stroke.
* The modularity of the system makes it flexible for future enhancements. Additional models, more features, or even a mobile application version could be integrated to extend its utility.

**10.2 Limitations**

While the models work effectively with the available dataset, the accuracy of predictions is highly dependent on the quality and diversity of the training data. Data limitations, such as missing values or biased samples, may impact the system's performance.

The system could be improved by incorporating more comprehensive data sources, such as real-time health monitoring via wearable devices, to improve the accuracy of predictions.

**10.3 Future Work**

To enhance the precision of predictions, more sophisticated models, such as ensemble methods or hybrid machine learning approaches, could be explored. Real-time data integration through mobile apps or wearable devices could be incorporated to provide dynamic predictions based on ongoing health measurements. The system could be expanded to include more comprehensive health-related data (e.g., medical history, family history) to provide more detailed and personalized risk assessments.

.

# REFERENCE

# REFERENCE

1. K. Yazhini and D. Loganathan, "A State of Art Approaches on Deep Learning Models in Healthcare: An Application Perspective", 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2020.
2. Suresh K. Peddoju, Himanshu Upadhyay and Shekhar Bhansali, "Health Monitoring with Low Power IoT Devices using Anomaly Detection Algorithm", Fourth International Conference on Fog and Mobile Edge Computing (FMEC), 2021
3. Pattanapong Chantamit and Madhu Goyal, "Prediction of Stroke Using Deep Learning Model" in ICONIP, Springer International Publishing, pp. 774-781, 2020.
4. Heo JoonNyung and G. Yoon Jihoon, "Machine Learning-Based Model for Prediction of Outcomes in Acute Stroke", American Heart Association, vol. 50, pp. 00-00, 2020.
5. X. Zhang, X. Wei, F. Li, F. Hu, W. Jia and C. Wang, "Fuzzy Support Vector Machine with Imbalanced Regulator and its Application in Stroke Classification,4–9 April 2022.
6. Songhee Cheon, Jungyoon Kim and Jihye Lim, The Use of Deep Learning to Predict Stroke Patient Mortality, Korea:Department of Physical Therapy, Youngsan University, Yangsan, pp. 626-790, April 2021.
7. L. Ding, C. Liu, Z. Li and Y. Wang, "Incorporating artificial intelligence into stroke care and research", *Stroke*, vol. 51, no. 12, pp. e351-e354, 2020.
8. Alharbi, W. Alosaimi, R. Sahal and H. Saleh, "Real-time system prediction for heart rate using deep learning and stream processing platforms", *Complexity*, vol. 2021, 2021.
9. S. Dev, H. Wang, C. S. Nwosu, N. Jain, B. Veeravalli and D. John, "A predictive analytics approach for stroke prediction using machine learning and neural networks", *Healthcare Analytics*, vol. 2, pp. 100032, 2022.
10. J. K. Kim, Y. J. Choo and M. C. Chang, "Prediction of motor function in stroke patients using machine learning algorithm: Development of practical models", *Journal of Stroke and Cerebrovascular Diseases*, vol. 30, no. 8, pp. 105856, 2021.

**APPENDIX I**

**APPENDIX**

**CODING**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Brain Stroke Prediction</title>**

**<style>**

**/\* Reset and Global Styles \*/**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**}**

**body {**

**font-family: Arial, sans-serif;**

**color: #333;**

**background: #f4f6f9; /\* Softer Grey Background \*/**

**overflow-x: hidden;**

**scroll-behavior: smooth;**

**}**

**/\* Navbar \*/**

**nav {**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**padding: 20px;**

**background: #003580; /\* Deep Blue \*/**

**color: #fff;**

**box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);**

**position: sticky;**

**top: 0;**

**z-index: 10;**

**}**

**.logo {**

**font-size: 1.8rem;**

**font-weight: bold;**

**font-family: 'Arial Black', sans-serif;**

**color: #FFD700;**

**animation: logoFadeIn 1s ease-in-out;**

**}**

**@keyframes logoFadeIn {**

**from { opacity: 0; }**

**to { opacity: 1; }**

**}**

**.nav-links {**

**list-style: none;**

**display: flex;**

**gap: 20px;**

**}**

**.nav-links a, .nav-buttons button {**

**color: #fff;**

**text-decoration: none;**

**font-weight: bold;**

**padding: 8px 12px;**

**border: none;**

**cursor: pointer;**

**transition: background 0.3s;**

**border-radius: 5px;**

**text-decoration: none;**

**}**

**.nav-links a:hover, .nav-buttons button:hover {**

**background: #ffffff;**

**}**

**.nav-buttons {**

**display: flex;**

**gap: 10px;**

**}**

**.login, .register {**

**padding: 8px 12px;**

**background: #FFD700;**

**color: #003580;**

**border-radius: 5px;**

**cursor: pointer;**

**transition: background 0.3s;**

**}**

**.register {**

**background: #FFA500;**

**}**

**/\* Hero Section \*/**

**.hero {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**text-align: center;**

**padding: 120px 20px;**

**background: linear-gradient(135deg, #003366, #004080);**

**color: #fff;**

**position: relative;**

**overflow: hidden;**

**}**

**.hero h1 {**

**font-size: 2.8rem;**

**font-weight: 700;**

**animation: typing 3s steps(30, end), blink-caret 0.5s step-end infinite;**

**overflow: hidden;**

**border-right: 3px solid #FFD700;**

**white-space: nowrap;**

**}**

**.hero p {**

**font-size: 1.3rem;**

**margin-bottom: 30px;**

**max-width: 600px;**

**}**

**.cta-button {**

**padding: 12px 20px;**

**font-size: 1rem;**

**color: #003580;**

**background-color: #FFD700;**

**border: none;**

**border-radius: 50px;**

**cursor: pointer;**

**transition: background 0.3s;**

**}**

**.cta-button:hover {**

**background-color: #FFC107;**

**}**

**/\* Typing animation \*/**

**@keyframes typing {**

**from { width: 0; }**

**to { width: 100%; }**

**}**

**@keyframes blink-caret {**

**from, to { border-color: transparent; }**

**50% { border-color: #FFD700; }**

**}**

**/\* Content Sections \*/**

**.content-section {**

**padding: 60px 20px;**

**max-width: 800px;**

**margin: auto;**

**}**

**.content-section h2 {**

**font-size: 2rem;**

**margin-bottom: 20px;**

**text-align: center;**

**color: #003580;**

**}**

**.content-section p {**

**font-size: 1rem;**

**line-height: 1.6;**

**margin-bottom: 20px;**

**text-align: center;**

**color: #555;**

**}**

**/\* Feature Cards \*/**

**.feature-cards {**

**display: flex;**

**gap: 20px;**

**flex-wrap: wrap;**

**justify-content: center;**

**}**

**.card {**

**flex: 1 1 250px;**

**padding: 20px;**

**background: #ffffff;**

**border-radius: 10px;**

**box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);**

**text-align: center;**

**transition: transform 0.3s;**

**max-width: 300px;**

**}**

**.card:hover {**

**transform: translateY(-10px);**

**}**

**.card h3 {**

**font-size: 1.2rem;**

**color: #003580;**

**}**

**/\* Contact Section \*/**

**.contact-form {**

**display: flex;**

**flex-direction: column;**

**gap: 10px;**

**max-width: 500px;**

**margin: 0 auto;**

**}**

**.contact-form input, .contact-form textarea, .contact-form button {**

**padding: 12px;**

**font-size: 1rem;**

**border-radius: 5px;**

**border: 1px solid #ddd;**

**}**

**.contact-form button {**

**background-color: #003580;**

**color: #fff;**

**cursor: pointer;**

**border: none;**

**transition: background 0.3s;**

**}**

**.contact-form button:hover {**

**background-color: #004080;**

**}**

**/\* Footer \*/**

**footer {**

**background: #003366;**

**color: #fff;**

**text-align: center;**

**padding: 20px;**

**}**

**/\* Animations \*/**

**.fade-in {**

**opacity: 0;**

**animation: fadeIn 2s ease forwards;**

**}**

**@keyframes fadeIn {**

**to { opacity: 1; }**

**}**

**</style>**

**</head>**

**<body>**

**<!-- Navbar -->**

**<nav>**

**<div class="logo">StrokePredict</div>**

**<ul class="nav-links">**

**<li><a href="#about">About</a></li>**

**<li><a href="#features">Features</a></li>**

**<li><a href="#how-it-works">How It Works</a></li>**

**<li><a href="#contact">Contact</a></li>**

**</ul>**

**<div class="nav-buttons">**

**<button class="login"><a href="http://127.0.0.1:5000/login">Login</a></button>**

**<button class="register"><a href="http://127.0.0.1:5000/register">Register</a></button>**

**</div>**

**</nav>**

**<!-- Hero Section -->**

**<section class="hero fade-in">**

**<h1>Predict Brain Stroke with Machine Learning</h1>**

**<p>Discover the potential to save lives by predicting stroke risks early through advanced AI algorithms.</p>**

**<button class="cta-button">Start Predicting Now</button>**

**</section>**

**<!-- About Section -->**

**<section id="about" class="content-section fade-in">**

**<h2>About Stroke Prediction</h2>**

**<p>Our tool combines AI and healthcare expertise to provide early stroke prediction, empowering individuals and healthcare providers with actionable insights.</p>**

**</section>**

**<!-- Features Section -->**

**<section id="features" class="content-section fade-in">**

**<h2>Features</h2>**

**<div class="feature-cards">**

**<div class="card">**

**<h3>AI-Powered Analysis</h3>**

**<p>Leverages advanced algorithms to analyze health data with precision.</p>**

**</div>**

**<div class="card">**

**<h3>User-Friendly Interface</h3>**

**<p>Accessible to users from all backgrounds for easy health insights.</p>**

**</div>**

**<div class="card">**

**<h3>Immediate Feedback</h3>**

**<p>Instant predictions to inform proactive healthcare decisions.</p>**

**</div>**

**</div>**

**</section>**

**<!-- How It Works Section -->**

**<section id="how-it-works" class="content-section fade-in">**

**<h2>How It Works</h2>**

**<ol>**

**<li>Upload Your Health Data</li>**

**<li>AI Processes and Analyzes</li>**

**<li>Receive Your Stroke Prediction</li>**

**</ol>**

**</section>**

**<!-- Contact Section -->**

**<section id="contact" class="content-section fade-in">**

**<h2>Contact Us</h2>**

**<form class="contact-form">**

**<input type="text" placeholder="Your Name" required>**

**<input type="email" placeholder="Your Email" required>**

**<textarea placeholder="Your Message" required></textarea>**

**<button type="submit">Send Message</button>**

**</form>**

**</section>**

**<!-- Footer -->**

**<footer>**

**<p>&copy; 2024 StrokePredict | All rights reserved.</p>**

**</footer>**

**</body>**

**</html>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<!-- Bootstrap CSS -->**

**<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">**

**<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.6.0/css/all.min.css">**

**<title>Stroke Prediction</title>**

**<style>**

**/\* Background and body styling \*/**

**body {**

**margin: 0;**

**font-family: Arial, sans-serif;**

**background-color: #808080;**

**background-image: linear-gradient(315deg, #B993D6 19%, #8CA6DB 85%);**

**overflow-x: hidden;**

**font-size: small;**

**}**

**#particles-js {**

**position: fixed;**

**width: 100%;**

**height: 100%;**

**top: 0;**

**left: 0;**

**z-index: -1;**

**}**

**.container {**

**margin-top: 60px;**

**border-radius: 10px;**

**padding: 20px;**

**background: rgba(255, 255, 255, 0.9);**

**box-shadow: 0 19px 38px rgba(0, 0, 0, 0.3), 0 15px 12px rgba(0, 0, 0, 0.22);**

**}**

**.btn-container {**

**padding: 20px;**

**}**

**.head {**

**font-weight: bold;**

**color: #073b4c;**

**}**

**.btn-primary {**

**color: #ffffff;**

**background-color: #073b4c;**

**border: none;**

**padding: 12px 20px;**

**font-size: 18px;**

**border-radius: 5px;**

**transition: 0.3s;**

**}**

**.btn-primary:hover {**

**background-color: #065a73;**

**box-shadow: 0 8px 16px rgba(0, 0, 0, 0.3);**

**}**

**.form-group {**

**margin-bottom: 15px;**

**text-align: left;**

**}**

**.form-select,**

**.form-control {**

**width: 100%;**

**padding: 10px;**

**border-radius: 5px;**

**border: 1px solid #ddd;**

**}**

**.form-select:focus,**

**.form-control:focus {**

**border-color: #073b4c;**

**box-shadow: 0px 0px 5px rgba(7, 59, 76, 0.4);**

**}**

**label {**

**font-weight: bold;**

**color: #073b4c;**

**}**

**.prediction {**

**background: #073b4c;**

**color: aliceblue;**

**padding: 10px;**

**border-radius: 5px;**

**}**

**</style>**

**<!-- Particles.js library -->**

**<script src="https://cdn.jsdelivr.net/particles.js/2.0.0/particles.min.js"></script>**

**</head>**

**<body>**

**<!-- Particles Background -->**

**<div id="particles-js"></div>**

**<div class="container">**

**<a href="login.html">**

**<i class="fa-solid fa-right-to-bracket fa-3x"></i>**

**</a>**

**<h1 class="head text-center mb-4">Stroke Prediction</h1>**

**<!-- Form for Prediction -->**

**<div class="btn-container">**

**<form action="{{ url\_for('predict')}}" method="post">**

**<div class="form-group">**

**<label for="gender"><i class="fa fa-user"></i> Gender</label>**

**<select class="form-select" id="gender" name="gender" aria-label="Default select example">**

**<option value="1">Male</option>**

**<option value="0">Female</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="age"><i class="fa fa-calendar"></i> Age</label>**

**<input type="number" class="form-control" name="age" required placeholder="Age" min="1" max="100">**

**</div>**

**<div class="form-group">**

**<label for="hypertension"><i class="fa fa-heartbeat"></i> Hypertension</label>**

**<select class="form-select" id="hypertension" name="hypertension">**

**<option value="1" selected>Yes</option>**

**<option value="0">No</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="heart\_disease"><i class="fa fa-heart"></i> Heart Disease</label>**

**<select class="form-select" id="disease" name="disease">**

**<option value="1" selected>Yes</option>**

**<option value="0">No</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="ever\_married"><i class="fa fa-ring"></i> Ever Married</label>**

**<select class="form-select" id="married" name="married">**

**<option value="1" selected>Yes</option>**

**<option value="0">No</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="work\_type"><i class="fa fa-briefcase"></i> Work Type</label>**

**<select class="form-select" id="work" name="work">**

**<option value="3" selected>Self-employed</option>**

**<option value="2">Private</option>**

**<option value="4">Children</option>**

**<option value="0">Government Job</option>**

**<option value="1">Never worked</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="residence\_type"><i class="fa fa-home"></i> Residence Type</label>**

**<select class="form-select" id="residence" name="residence">**

**<option value="1" selected>Urban</option>**

**<option value="0">Rural</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="avg\_glucose\_level"><i class="fa fa-tint"></i> Average Glucose Level</label>**

**<input type="text" class="form-control" name="avg\_glucose\_level" required placeholder="Average Glucose Level">**

**</div>**

**<div class="form-group">**

**<label for="bmi"><i class="fa fa-weight"></i> BMI</label>**

**<input type="text" class="form-control" name="bmi" required placeholder="Body Mass Index (BMI)">**

**</div>**

**<div class="form-group">**

**<label for="smoking"><i class="fa fa-smoking"></i> Smoking Status</label>**

**<select class="form-select" id="smoking" name="smoking">**

**<option value="0">Unknown</option>**

**<option value="2" selected>Never smoked</option>**

**<option value="1">Formerly smoked</option>**

**<option value="3">Smokes</option>**

**</select>**

**</div>**

**<button type="submit" class="btn btn-primary btn-lg btn-block">Predict</button>**

**<br>**

**<center>**

**<h1 class="prediction">{{ prediction\_text }}</h1>**

**</center>**

**</form>**

**</div>**

**</div>**

**<!-- Particle Configuration -->**

**<script>**

**particlesJS('particles-js', {**

**"particles": {**

**"number": {**

**"value": 80,**

**"density": { "enable": true, "value\_area": 800 }**

**},**

**"color": { "value": "#ffffff" },**

**"shape": {**

**"type": "circle",**

**"stroke": { "width": 0, "color": "#000000" }**

**},**

**"opacity": { "value": 0.5, "random": true },**

**"size": { "value": 3, "random": true },**

**"line\_linked": { "enable": true, "distance": 150, "color": "#ffffff", "opacity": 0.4, "width": 1 },**

**"move": { "enable": true, "speed": 2, "direction": "none" }**

**},**

**"interactivity": {**

**"detect\_on": "canvas",**

**"events": { "onhover": { "enable": true, "mode": "repulse" }, "onclick": { "enable": true, "mode": "push" } },**

**"modes": {**

**"repulse": { "distance": 150, "duration": 0.4 }**

**}**

**},**

**"retina\_detect": true**

**});**

**</script>**

**</body>**

**</html>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Login Page</title>**

**<link rel="stylesheet" href="styles.css">**

**<style>**

**/\* Reset default styles \*/**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**}**

**body, html {**

**height: 100%;**

**font-family: Arial, sans-serif;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**overflow: hidden;**

**}**

**/\* Background Animation \*/**

**#background {**

**position: absolute;**

**top: 0;**

**left: 0;**

**width: 100%;**

**height: 100%;**

**background: linear-gradient(-45deg, #3b8d99, #6b5b95, #d64161, #ff7b25);**

**background-size: 400% 400%;**

**animation: gradient 15s ease infinite;**

**z-index: -2;**

**}**

**@keyframes gradient {**

**0% { background-position: 0% 50%; }**

**50% { background-position: 100% 50%; }**

**100% { background-position: 0% 50%; }**

**}**

**/\* Particle Effect \*/**

**#particles-js {**

**position: absolute;**

**width: 100%;**

**height: 100%;**

**z-index: -1;**

**}**

**/\* Login Container with Frosted Glass Effect \*/**

**.login-container {**

**background: rgba(255, 255, 255, 0.2);**

**backdrop-filter: blur(10px);**

**padding: 40px;**

**border-radius: 10px;**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);**

**font-size: 30px;**

**text-align: center;**

**color: #333;**

**width: 400px;**

**height: 550px;**

**}**

**.login-container h1 {**

**margin-bottom: 20px;**

**}**

**.login-container label {**

**display: block;**

**text-align: left;**

**margin-bottom: 5px;**

**font-weight: bold;**

**}**

**.login-container input {**

**width: 100%;**

**padding: 10px;**

**margin-bottom: 15px;**

**border: 1px solid #ccc;**

**border-radius: 5px;**

**}**

**.login-container button {**

**width: 100%;**

**padding: 10px;**

**background-color: #007bff;**

**color: #fff;**

**border: none;**

**border-radius: 5px;**

**cursor: pointer;**

**}**

**.login-container button:hover {**

**background-color: #0056b3;**

**}**

**.login-container p {**

**margin-top: 15px;**

**}**

**.login-container a {**

**color: #007bff;**

**text-decoration: none;**

**}**

**</style>**

**</head>**

**<body>**

**<!-- Background Animation -->**

**<div id="background"></div>**

**<!-- Login Form Container -->**

**<div class="login-container">**

**<h1 style="font-family:'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif ;">Login</h1>**

**<form action="{{ url\_for('login') }}" method="post">**

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

**<input type="text" id="username" name="username" required>**

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

**<input type="password" id="password" name="password" required>**

**{% if error %}**

**<div class="alert alert-danger">{{ error }}</div>**

**{% endif %}**

**<button type="submit" class="btn btn-primary">Login</button>**

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

**</form>**

**</div>**

**<!-- Particle.js Script -->**

**<script src="https://cdn.jsdelivr.net/particles.js/2.0.0/particles.min.js"></script>**

**<script>particlesJS('background',**

**{**

**"particles": {**

**"number": {**

**"value": 80,**

**"density": {**

**"enable": true,**

**"value\_area": 800**

**}**

**},**

**"color": {**

**"value": "#ffffff"**

**},**

**"shape": {**

**"type": "circle",**

**"stroke": {**

**"width": 0,**

**"color": "#000000"**

**},**

**"polygon": {**

**"nb\_sides": 5**

**},**

**},**

**"opacity": {**

**"value": 0.5,**

**"random": false,**

**"anim": {**

**"enable": false,**

**"speed": 1,**

**"opacity\_min": 0.1,**

**"sync": false**

**}**

**},**

**"size": {**

**"value": 3,**

**"random": true,**

**"anim": {**

**"enable": false,**

**"speed": 40,**

**"size\_min": 0.1,**

**"sync": false**

**}**

**},**

**"line\_linked": {**

**"enable": true,**

**"distance": 150,**

**"color": "#ffffff",**

**"opacity": 0.4,**

**"width": 1**

**},**

**"move": {**

**"enable": true,**

**"speed": 6,**

**"direction": "none",**

**"random": false,**

**"straight": false,**

**"out\_mode": "out",**

**"bounce": false,**

**"attract": {**

**"enable": false,**

**"rotateX": 600,**

**"rotateY": 1200**

**}**

**}**

**},**

**"interactivity": {**

**"detect\_on": "canvas",**

**"events": {**

**"onhover": {**

**"enable": true,**

**"mode": "repulse"**

**},**

**"onclick": {**

**"enable": true,**

**"mode": "push"**

**},**

**"resize": true**

**},**

**"modes": {**

**"grab": {**

**"distance": 400,**

**"line\_linked": {**

**"opacity": 1**

**}**

**},**

**"bubble": {**

**"distance": 400,**

**"size": 40,**

**"duration": 2,**

**"opacity": 8,**

**"speed": 3**

**},**

**"repulse": {**

**"distance": 200,**

**"duration": 0.4**

**},**

**"push": {**

**"particles\_nb": 4**

**},**

**"remove": {**

**"particles\_nb": 2**

**}**

**}**

**},**

**"retina\_detect": true**

**}**

**);**

**</script>**

**</body>**

**</html>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Login Page</title>**

**<link rel="stylesheet" href="styles.css">**

**<style>**

**/\* Reset default styles \*/**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**}**

**body, html {**

**height: 100%;**

**font-family: Arial, sans-serif;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**overflow: hidden;**

**}**

**/\* Background Animation \*/**

**#background {**

**position: absolute;**

**top: 0;**

**left: 0;**

**width: 100%;**

**height: 100%;**

**background: linear-gradient(-45deg, #3b8d99, #6b5b95, #d64161, #ff7b25);**

**background-size: 400% 400%;**

**animation: gradient 15s ease infinite;**

**z-index: -2;**

**}**

**@keyframes gradient {**

**0% { background-position: 0% 50%; }**

**50% { background-position: 100% 50%; }**

**100% { background-position: 0% 50%; }**

**}**

**/\* Particle Effect \*/**

**#particles-js {**

**position: absolute;**

**width: 100%;**

**height: 100%;**

**z-index: -1;**

**}**

**/\* Login Container with Frosted Glass Effect \*/**

**.login-container {**

**background: rgba(255, 255, 255, 0.2);**

**backdrop-filter: blur(10px);**

**padding: 40px;**

**border-radius: 10px;**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);**

**font-size: 30px;**

**text-align: center;**

**color: #333;**

**width: 400px;**

**height: 550px;**

**}**

**.login-container h1 {**

**margin-bottom: 20px;**

**}**

**.login-container label {**

**display: block;**

**text-align: left;**

**margin-bottom: 5px;**

**font-weight: bold;**

**}**

**.login-container input {**

**width: 100%;**

**padding: 10px;**

**margin-bottom: 15px;**

**border: 1px solid #ccc;**

**border-radius: 5px;**

**}**

**.login-container button {**

**width: 100%;**

**padding: 10px;**

**background-color: #007bff;**

**color: #fff;**

**border: none;**

**border-radius: 5px;**

**cursor: pointer;**

**}**

**.login-container button:hover {**

**background-color: #0056b3;**

**}**

**.login-container p {**

**margin-top: 15px;**

**}**

**.login-container a {**

**color: #007bff;**

**text-decoration: none;**

**}**

**</style>**

**</head>**

**<body>**

**<!-- Background Animation -->**

**<div id="background"></div>**

**<!-- Login Form Container -->**

**<div class="login-container">**

**<h1 style="font-family:'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif ;">Register</h1>**

**<form action="{{ url\_for('register') }}" method="post">**

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

**<input type="text" id="username" name="username" required>**

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

**<input type="password" id="password" name="password" required>**

**{% if error %}**

**<div class="alert alert-danger">{{ error }}</div>**

**{% endif %}**

**<button type="submit" class="btn btn-primary">Register</button>**

**<p>Already have an account? <a href="{{ url\_for('login') }}">Login</a></p>**

**</form>**

**</div>**

**<!-- Particle.js Script -->**

**<script src="https://cdn.jsdelivr.net/particles.js/2.0.0/particles.min.js"></script>**

**<script>particlesJS('background',**

**{**

**"particles": {**

**"number": {**

**"value": 80,**

**"density": {**

**"enable": true,**

**"value\_area": 800**

**}**

**},**

**"color": {**

**"value": "#ffffff"**

**},**

**"shape": {**

**"type": "circle",**

**"stroke": {**

**"width": 0,**

**"color": "#000000"**

**},**

**"polygon": {**

**"nb\_sides": 5**

**},**

**},**

**"opacity": {**

**"value": 0.5,**

**"random": false,**

**"anim": {**

**"enable": false,**

**"speed": 1,**

**"opacity\_min": 0.1,**

**"sync": false**

**}**

**},**

**"size": {**

**"value": 3,**

**"random": true,**

**"anim": {**

**"enable": false,**

**"speed": 40,**

**"size\_min": 0.1,**

**"sync": false**

**}**

**},**

**"line\_linked": {**

**"enable": true,**

**"distance": 150,**

**"color": "#ffffff",**

**"opacity": 0.4,**

**"width": 1**

**},**

**"move": {**

**"enable": true,**

**"speed": 6,**

**"direction": "none",**

**"random": false,**

**"straight": false,**

**"out\_mode": "out",**

**"bounce": false,**

**"attract": {**

**"enable": false,**

**"rotateX": 600,**

**"rotateY": 1200**

**}**

**}**

**},**

**"interactivity": {**

**"detect\_on": "canvas",**

**"events": {**

**"onhover": {**

**"enable": true,**

**"mode": "repulse"**

**},**

**"onclick": {**

**"enable": true,**

**"mode": "push"**

**},**

**"resize": true**

**},**

**"modes": {**

**"grab": {**

**"distance": 400,**

**"line\_linked": {**

**"opacity": 1**

**}**

**},**

**"bubble": {**

**"distance": 400,**

**"size": 40,**

**"duration": 2,**

**"opacity": 8,**

**"speed": 3**

**},**

**"repulse": {**

**"distance": 200,**

**"duration": 0.4**

**},**

**"push": {**

**"particles\_nb": 4**

**},**

**"remove": {**

**"particles\_nb": 2**

**}**

**}**

**},**

**"retina\_detect": true**

**}**

**);**

**</script>**

**</body>**

**</html>**

**from flask.helpers import url\_for**

**import numpy as np**

**import pandas as pd**

**from flask import Flask, request, render\_template, redirect, session**

**import pickle**

**from werkzeug.security import generate\_password\_hash, check\_password\_hash**

**from database import init\_db, add\_user, get\_user, save\_user\_input**

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

**app.secret\_key = 'your\_secret\_key'  # Set a secret key for session management**

**model = pickle.load(open('model.pickle', 'rb'))**

**# Initialize the database**

**init\_db()**

**@app.route("/")**

**def home():**

**return render\_template('home.html')  # Render home page for all users**

**@app.route("/index")**

**def index():**

**return render\_template('index.html')**

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

**def register():**

**if request.method == 'POST':**

**username = request.form['username']**

**password = generate\_password\_hash(request.form['password'])**

**add\_user(username, password)**

**return redirect(url\_for('login'))  # Redirect to login after registration**

**return render\_template('register.html')**

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

**def login():**

**if request.method == 'POST':**

**username = request.form['username']**

**password = request.form['password']**

**user = get\_user(username)**

**if user and check\_password\_hash(user[2], password):  # Validate credentials**

**session['username'] = username  # Store username in session**

**return redirect(url\_for('index'))  # Redirect to home after successful login**

**else:**

**return render\_template('login.html', error='Invalid credentials')  # Pass error message to the template**

**return render\_template('login.html')  # Show login form**

**@app.route('/logout')**

**def logout():**

**session.pop('username', None)  # Clear session**

**return redirect(url\_for('home'))  # Redirect to home page after logout**

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

**def predict():**

**if 'username' not in session:  # Check if user is logged in**

**return redirect(url\_for('login'))  # Redirect to login if not**

**if request.method == "POST":**

**# Extract features from the form**

**gender\_Male = int(request.form['gender'])**

**age = int(request.form['age'])**

**hypertension\_1 = int(request.form['hypertension'])**

**heart\_disease\_1 = int(request.form['disease'])**

**ever\_married\_Yes = int(request.form['married'])**

**work = int(request.form['work'])**

**Residence\_type\_Urban = int(request.form['residence'])**

**avg\_glucose\_level = float(request.form['avg\_glucose\_level'])**

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

**smoking = int(request.form['smoking'])**

**# Work type encoding**

**work\_type\_Never\_worked = 1 if work == 1 else 0**

**work\_type\_Private = 1 if work == 2 else 0**

**work\_type\_Self\_employed = 1 if work == 3 else 0**

**work\_type\_children = 1 if work == 4 else 0**

**# Smoking status encoding**

**smoking\_status\_formerly\_smoked = 1 if smoking == 1 else 0**

**smoking\_status\_never\_smoked = 1 if smoking == 2 else 0**

**smoking\_status\_smokes = 1 if smoking == 3 else 0**

**# Prepare input features for prediction**

**input\_features = [age, avg\_glucose\_level, bmi, gender\_Male, hypertension\_1, heart\_disease\_1,**

**ever\_married\_Yes, work\_type\_Never\_worked, work\_type\_Private,**

**work\_type\_Self\_employed, work\_type\_children, Residence\_type\_Urban,**

**smoking\_status\_formerly\_smoked, smoking\_status\_never\_smoked, smoking\_status\_smokes]**

**# Create DataFrame for prediction**

**df = pd.DataFrame([input\_features], columns=['age', 'avg\_glucose\_level', 'bmi', 'gender\_Male',**

**'hypertension\_1', 'heart\_disease\_1', 'ever\_married\_Yes',**

**'work\_type\_Never\_worked', 'work\_type\_Private',**

**'work\_type\_Self-employed', 'work\_type\_children',**

**'Residence\_type\_Urban', 'smoking\_status\_formerly smoked',**

**'smoking\_status\_never smoked', 'smoking\_status\_smokes'])**

**# Make prediction**

**prediction = model.predict(df)[0]**

**# Save the user input and prediction to the database**

**username = session['username']**

**save\_user\_input(username, gender\_Male, age, hypertension\_1, heart\_disease\_1, ever\_married\_Yes, work,**

**Residence\_type\_Urban, avg\_glucose\_level, bmi, smoking, prediction)**

**# Render prediction result**

**if prediction == 1:**

**return render\_template('index.html', prediction\_text='Patient has stroke risk')**

**else:**

**return render\_template('index.html', prediction\_text='Congratulations, patient does not have stroke risk')**

**if \_\_name\_\_ == "\_\_main\_\_":**

**app.run(debug=True)  # Run the app in debug mode for better error messages**

**import sqlite3**

**def init\_db():**

**try:**

**with sqlite3.connect('users.db') as conn:**

**c = conn.cursor()**

**# Create users table if not exists**

**c.execute('''**

**CREATE TABLE IF NOT EXISTS users (**

**id INTEGER PRIMARY KEY AUTOINCREMENT,**

**username TEXT NOT NULL UNIQUE,**

**password TEXT NOT NULL**

**)**

**''')**

**# Create user\_inputs table to store prediction inputs**

**c.execute('''**

**CREATE TABLE IF NOT EXISTS user\_inputs (**

**id INTEGER PRIMARY KEY AUTOINCREMENT,**

**username TEXT,**

**gender INTEGER,**

**age INTEGER,**

**hypertension INTEGER,**

**heart\_disease INTEGER,**

**ever\_married INTEGER,**

**work INTEGER,**

**residence INTEGER,**

**avg\_glucose\_level REAL,**

**bmi REAL,**

**smoking INTEGER,**

**prediction INTEGER,**

**FOREIGN KEY (username) REFERENCES users(username)**

**)**

**''')**

**conn.commit()**

**except sqlite3.Error as e:**

**print(f"An error occurred while initializing the database: {e}")**

**def add\_user(username, password):**

**try:**

**with sqlite3.connect('users.db') as conn:**

**c = conn.cursor()**

**c.execute('INSERT INTO users (username, password) VALUES (?, ?)', (username, password))**

**conn.commit()**

**except sqlite3.Error as e:**

**print(f"An error occurred while adding a user: {e}")**

**def get\_user(username):**

**try:**

**with sqlite3.connect('users.db') as conn:**

**c = conn.cursor()**

**c.execute('SELECT \* FROM users WHERE username = ?', (username,))**

**user = c.fetchone()**

**return user**

**except sqlite3.Error as e:**

**print(f"An error occurred while fetching the user: {e}")**

**return None**

**def save\_user\_input(username, gender, age, hypertension, heart\_disease, ever\_married, work, residence,**

**avg\_glucose\_level, bmi, smoking, prediction):**

**try:**

**with sqlite3.connect('users.db') as conn:**

**c = conn.cursor()**

**c.execute('''**

**INSERT INTO user\_inputs (username, gender, age, hypertension, heart\_disease, ever\_married,**

**work, residence, avg\_glucose\_level, bmi, smoking, prediction)**

**VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)**

**''', (username, gender, age, hypertension, heart\_disease, ever\_married, work, residence, avg\_glucose\_level,**

**bmi, smoking, prediction))**

**conn.commit()**

**except sqlite3.Error as e:**

**print(f"An error occurred while saving user input: {e}")**

**# Import necessary libraries**

**import numpy as np**

**import pandas as pd**

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

**from sklearn.preprocessing import StandardScaler, LabelEncoder**

**from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix**

**from sklearn.tree import DecisionTreeClassifier**

**from sklearn.naive\_bayes import GaussianNB**

**from tensorflow.keras.models import Sequential**

**from tensorflow.keras.layers import Dense**

**# Load the dataset**

**df = pd.read\_csv("healthcare-dataset-stroke-data.csv")**

**# Check for missing values and fill or drop as appropriate**

**df.fillna(df.mean(), inplace=True)**

**# Encode categorical features**

**label\_encoder = LabelEncoder()**

**df['gender'] = label\_encoder.fit\_transform(df['gender'])**

**df['ever\_married'] = label\_encoder.fit\_transform(df['ever\_married'])**

**df['work\_type'] = label\_encoder.fit\_transform(df['work\_type'])**

**df['Residence\_type'] = label\_encoder.fit\_transform(df['Residence\_type'])**

**df['smoking\_status'] = label\_encoder.fit\_transform(df['smoking\_status'])**

**# Split the dataset into features and target**

**X = df.drop('stroke', axis=1)  # Features**

**y = df['stroke']               # Target variable**

**# Split data into training and test sets**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# Standardize the features**

**scaler = StandardScaler()**

**X\_train = scaler.fit\_transform(X\_train)**

**X\_test = scaler.transform(X\_test)**

**### Decision Tree Classifier**

**dt\_model = DecisionTreeClassifier(random\_state=42)**

**dt\_model.fit(X\_train, y\_train)**

**dt\_pred = dt\_model.predict(X\_test)**

**print("Decision Tree Accuracy:", accuracy\_score(y\_test, dt\_pred))**

**print(classification\_report(y\_test, dt\_pred))**

**### Naive Bayes Classifier**

**nb\_model = GaussianNB()**

**nb\_model.fit(X\_train, y\_train)**

**nb\_pred = nb\_model.predict(X\_test)**

**print("Naive Bayes Accuracy:", accuracy\_score(y\_test, nb\_pred))**

**print(classification\_report(y\_test, nb\_pred))**

**### Artificial Neural Network (ANN)**

**ann\_model = Sequential()**

**ann\_model.add(Dense(32, activation='relu', input\_shape=(X\_train.shape[1],)))**

**ann\_model.add(Dense(16, activation='relu'))**

**ann\_model.add(Dense(1, activation='sigmoid'))**

**# Compile the model**

**ann\_model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])**

**# Train the model**

**ann\_model.fit(X\_train, y\_train, epochs=50, batch\_size=32, validation\_split=0.2)**

**# Evaluate the model**

**ann\_pred = (ann\_model.predict(X\_test) > 0.5).astype("int32")**

**print("ANN Accuracy:", accuracy\_score(y\_test, ann\_pred))**

**print(classification\_report(y\_test, ann\_pred))**

**APPENDIX II**