PROJECT REPORT

On

BRAIN STROKE DETECTION SYSTEM

Submitted in partial fulfillment for the award of the degree

Of

Masters of Computer Applications

By

AISHWARYA RAVIKUMAR (MLM23MCA-2004)

Under the guidance of

Ms. ASHWANI VIJAYACHANDRAN

(Assistant Professor, Dept. of Computer Applications)

DEPARTMENT OF COMPUTER APPLICATIONS

MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR

(Affiliated to APJ Abdul Kalam Technological University)

APRIL 2025

VISION

To become a centre of excellence in computer applications,competent in the global ecosystem with technical knowledge,innovation with a sense of social commitment.

MISSION

- To serve with state of the art education,foster advanced research and cultivate innovation in the field of computer applications.
- To prepare learners with knowledge skills and critical thinking to excel in the technological landscape and contribute positively to society.

Program Educational Objectives

- PEO I :Graduates will possess a solid foundation and in-depth understanding of computer applications and will be equipped to analyze real-world problems, design and create innovative solutions, and effectively manage and maintain these solutions in their professional careers.
- PEO II: Graduates will acquire technological advancements through continued education, lifelong learning and research, thereby making meaningful contributions to the field of computing.
- PEO III: Graduates will cultivate team spirit, leadership, communication skills, ethics, and social values, enabling them to apply their understanding of the societal impacts of computer applications effectively.

Program Specific Outcomes

- **PSO I**: Apply advanced technologies through innovations to enhance the efficiency of design development.
- **PSO II**: Apply the principles of computing to analyze, design and implement sustainable solutions for real world challenges.

MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR
DEPARTMENT OF COMPUTER APPLICATIONS
APRIL 2025

CERTIFICATE

This is to Certify that the project entitled ***"Brain Stroke Detection System"*** *is the Bonafide record of the work done by* ***Aishwarya Ravikumar (MLM23MCA-2004)*** *of Masters of Computer Applications towards the partial fulfillment for the award of the DEGREE OF MASTERS OF COMPUTER APPLICATIONS under APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY during the academic year 2024-2025.*

Project Guide
Ms. Ashwani Vijayachandran
Assistant Professor
Dept. of Computer Application

Head of the Department
Ms. Divya S B
Associate Professor
Dept. of Computer Applications

Project Coordinator
Ms. Banu Sumayya S
Assistant Professor
Dept. of Computer Applications

External Examiner

ACKNOWLEDGEMENT

First, I thank the Almighty **God** for giving me the strength to venture for such an enigmatic logical creation in a jovial way.

I am greatly indebted to the authorities of Mangalam College of Engineering for providing me the necessary facilities to successfully complete my Project on the topic "Brain Stroke Detection System."

I express my sincere thanks to **Dr.Vinodh P Vijayan**, the principal, for providing me with the best facilities to complete my seminar successfully.

I thank and express my solicit of gratitude to **Ms. Divya S B**, Associate Professor & HOD, Dept. of Computer Applications, Mangalam College of Engineering, for his invaluable help and support which helped me a lot in successfully completing this Seminar.

I express my gratitude to my Internal Guide, **Ms. Ashwani Vijayachandran**, Assistant professor, Department of Computer Applications, for her suggestions and encouragement which helped me in the successful completion of my Seminar.

I express my gratitude to my project coordinator **Ms. Banu Sumayya S**, Assistant professor, Department of Computer Applications for the suggestions and encouragement which helped in the successful completion of our Project.

Finally, I would like to express our heartfelt thanks to my parents who were very supportive both financially and mentally and for their encouragement to achieve my goals.

AISHWARYA RAVIKUMAR

(MLM23MCA-2004)

ABSTRACT

Stroke is a major cause of disability and death worldwide, making early detection and classification essential for timely medical intervention. This project presents a machine learning-based stroke detection system that analyzes brain scan images to identify and classify stroke types, including ischemic, hemorrhagic, and normal cases. The system utilizes deep learning models, specifically EfficientNet, to extract relevant features and provide accurate stroke predictions. Designed for both patients and healthcare professionals, the platform allows users to upload medical images, receive automated diagnostic predictions, and schedule consultations with specialists. A secure database stores past records, enabling easy tracking of patient history and appointments. Additionally, the system integrates a real-time chat feature for instant communication with doctors. By leveraging AI-powered image analysis, the proposed approach enhances diagnostic accuracy, efficiency, and accessibility, ultimately supporting faster decision-making, early intervention, and improved patient care. Future enhancements aim to incorporate larger datasets, real-time monitoring, and AI-driven treatment recommendations to further optimize stroke detection and management.

TABLE OF CONTENTS

TITLE

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION		FULL FORM
AI	-	Artificial Intelligence
DL	-	Deep Learning
ML	-	Machine Learning
CNN	-	Convolutional Neural Network
DNN	-	Deep Neural Network
ANN	-	Artificial Neural Network
Bi-LSTM	-	Bidirectional Long Short-Term Memory
CT	-	Computed Tomography

1. INTRODUCTION

1.1.BACKGROUND

Stroke is one of the major cause of death and disability worldwide, affecting millions of individuals each year. It occurs when the brain's blood supply is disrupted, either due to a clot obstructing blood flow (ischemic stroke) or a ruptured blood vessel causing bleeding (hemorrhagic stroke). Immediate and accurate diagnosis is critical in stroke management to prevent severe neurological damage and improve patient outcomes. Traditional diagnostic methods rely heavily on radiologists and manual assessment of medical imaging, which can be time-consuming, prone to human error, and limited by resource availability in many healthcare facilities.

With advancements in artificial intelligence and deep learning, automated stroke detection systems have emerged as a promising solution to enhance diagnostic accuracy and efficiency. This project focuses on developing an AI-based stroke classification system using deep learning techniques to analyze brain scans and classify different types of strokes. The system leverages pre-trained models such as EfficientNet to extract image features and make predictions with high precision. By automating the detection process, the system assists healthcare professionals in making faster decisions, reducing workload, and improving early diagnosis. Additionally, the platform integrates functionalities such as online consultations, appointment scheduling, and real-time analysis, making stroke detection more accessible. Future improvements will include expanding the dataset to enhance model robustness, integrating real-time monitoring for continuous patient assessment, and incorporating AI-driven treatment recommendations to optimize stroke management and recovery strategies.

Early detection and timely intervention are crucial in reducing the impact of stroke. Among imaging techniques, CT scans play a vital role in stroke diagnosis, offering fast and co Unlike MRI, CT scans are particularly useful in emergency cases due to their rapid processing and ability to accommodate critically ill patients. Several techniques have been explored to enhance CT scan images, such as modified tracking algorithms, histogram equalization, and adaptive multi-scale data condensation, which improve contrast and noise reduction.

1.2.OVERVIEW

Stroke is a critical medical condition that occurs when the brain's blood supply is disrupted, leading to potential brain cell damage or death. It is one of the leading causes of disability and mortality worldwide, making early detection essential for timely and effective treatment. Conventional stroke diagnosis relies on manual examination of CT scan images by radiologists, which can be time-intensive and subject to human error. To address these challenges, this project introduces an AI-powered stroke detection system that leverages deep learning techniques to automate and enhance the accuracy of stroke classification.

The system is designed to analyze brain CT scans and categorize them into three primary types: normal (no stroke), ischemic stroke, and hemorrhagic stroke. Ischemic strokes occur due to blockages that restrict blood flow to the brain, while hemorrhagic strokes result from blood vessel ruptures that cause bleeding in the brain. By utilizing artificial neural networks (ANNs) and convolutional neural networks (CNNs), the system enhances detection precision and reliability.

A key advantage of this system is its ability to process data in real time, allowing for rapid and accurate diagnosis. This feature significantly reduces the time required for medical professionals to make critical decisions, which can be crucial in emergency situations. The system also incorporates a user-friendly interface that enables medical staff to upload CT scans and obtain AI-generated diagnostic results within seconds. Additionally, it supports secure data storage, ensuring patient records are archived for future reference and analysis.

The implementation of this system offers several benefits, including early stroke detection, which can help identify high-risk individuals before severe symptoms arise. It also reduces diagnosis time, leading to improved survival rates and treatment outcomes. By automating the classification process, the system alleviates the workload on radiologists, allowing them to focus on more complex cases. Furthermore, the system is designed to be scalable and can be integrated into existing hospital management systems and telemedicine platforms, making it accessible for remote healthcare services.In summary, this AI-driven stroke detection system modernizes traditional medical imaging by introducing automated classification methods..

1.3 PROBLEM STATEMENT

The increasing prevalence of strokes worldwide presents a significant challenge in early detection and timely intervention. Stroke is a leading cause of disability and mortality, often resulting in severe health complications if not diagnosed and treated promptly. Traditional diagnostic methods, such as manual interpretation of medical imaging, can be time-consuming, prone to human error, and require expert radiologists for accurate assessment. In many cases, delays in treatment leads to irreversible brain damage, significantly impacting the patient's quality of life.

Additionally, access to advanced diagnostic facilities and specialized medical professionals is limited in many regions, making it difficult for patients to receive timely and effective stroke assessment. This limitation underscores the need for an automated, accurate, and efficient stroke detection system that can assist medical professionals in diagnosing stroke types quickly and reliably. The integration of artificial intelligence and deep learning techniques in medical imaging has the potential to enhance diagnostic accuracy and reduce dependency on manual interpretation.

Another significant issue is the disparity in access to healthcare, especially in rural and underdeveloped regions, where specialized medical professionals and advanced imaging facilities are scarce. Many patients do not receive timely medical attention due to the unavailability of expert radiologists, resulting in preventable complications and increased fatality rates. This project aims to bridge this gap by leveraging artificial intelligence and deep learning technologies to develop an automated stroke classification system. By analyzing CT scan images, the system can differentiate between different stroke types, allowing for faster and more accurate diagnosis, even in resource-limited settings.

This project aims to develop a stroke classification system utilizing machine learning and deep learning algorithms to analyze CT scan images for the detection and classification of stroke types. The system seeks to provide an automated, real-time solution to aid healthcare professionals in early stroke diagnosis, ensuring timely medical intervention and improved patient outcomes. By leveraging advanced image processing techniques, the project aspires to improve diagnostic efficiency, enhance accessibility to stroke detection, and minimize delays in medical decision-making.

1.4 MOTIVATION

Stroke is a major health concern that affects millions of people worldwide, often leading to severe neurological impairments or even death. The devastating effects of a stroke, including loss of motor function, speech impairment, and cognitive decline, highlight the urgent need for early detection and rapid intervention. Timely medical treatment can significantly reduce brain damage and improve survival rates, yet many challenges hinder efficient stroke diagnosis and treatment. Traditional stroke detection methods rely on the manual interpretation of CT scans by expert radiologists, a process that is time-consuming and prone to human error. In many underdeveloped or rural areas, the lack of skilled medical professionals further delays diagnosis, increasing the risk of severe complications or even death.

One of the most critical aspects of stroke management is the need for immediate intervention. The concept of the "Golden Hour" emphasizes that treatment within the first 60 minutes of a stroke event can greatly improve patient outcomes. However, conventional diagnostic procedures may not always meet this urgency due to processing delays and the dependency on expert evaluation. Additionally, even experienced radiologists can make errors due to fatigue, subjective judgment, or image quality issues, which can lead to misdiagnosis or missed cases.

Another driving factor behind this research is the growing need for automated diagnostic tools in remote and underserved areas. Many regions lack experienced radiologists, making it difficult to diagnose strokes promptly. Implementing a robust deep learning-based system can bridge this gap, providing an accessible and reliable solution for stroke detection. This project aims in the advancement of AI-driven healthcare, making early stroke detection more effective, scalable, and accessible to a broader population.

This study is driven by the transformative potential of artificial intelligence in stroke detection and patient care. By leveraging deep learning methods in medical diagnostics, the project seeks to improve the precision, efficiency, and accessibility of stroke identification, especially in healthcare environments with limited resources. An AI-driven approach can assist clinicians in making data-driven decisions, ensuring that patients receive timely and effective treatment.

1.5 SCOPE

The scope of this project is centered on the development of an intelligent, automated stroke detection and classification system utilizing deep learning techniques. The system is designed to assist medical professionals by analyzing medical imaging data, particularly CT scans, to identify and classify stroke types with high accuracy. By leveraging artificial intelligence, the project aims to enhance the efficiency of stroke diagnosis, minimize misdiagnosis, and support timely medical intervention, ultimately improving patient outcomes.

One of the key aspects of this project is the integration of a robust image-processing pipeline that can preprocess, segment, and analyze brain scans to detect patterns indicative of ischemic or hemorrhagic strokes. The model will be trained on a diverse dataset to ensure generalizability and reliability in real-world clinical settings. Additionally, the system will feature an interactive, user-friendly interface that allows healthcare practitioners to upload and analyze medical images with minimal effort.

Beyond detection, the project also focuses on incorporating a secure and scalable patient data management system, enabling healthcare professionals to track patient history, compare past and present scans, and facilitate better treatment planning. The system may also integrate real-time processing capabilities, making it feasible for emergency rooms and hospitals to receive instant diagnostic results.

Future enhancements of the project could include expanding the dataset to improve model robustness, integrating multi-modal imaging techniques such as MRI scans for better diagnostic precision, and developing a cloud-based solution for remote accessibility. Additionally, the potential incorporation of explainable AI techniques could help medical professionals understand the model’s decision-making process, increasing trust and adoption.

Ultimately, this project aims to bridge the gap between artificial intelligence and healthcare by providing an efficient, accurate, and automated stroke detection system. By reducing the time required for diagnosis and improving accuracy, the system could significantly contribute to better stroke management, timely interventions, and improved survival rates for patients.

2. LITERATURE REVIEW

2.1 Conventional and deep learning methods for skull stripping in brain MRI[1]

Authors: H. Z. U. Rehman, H. Hwang and S. Lee

Publication details: *Appl. Sci.*, vol. 10, no. 5, pp. 1773, Mar. 2020

The various medical imaging techniques, magnetic resonance imaging (MRI) of the brain is one of the most prevalent image acquisitions performed in the diagnostic centers and hospitals. The acquisition of a brain MRI scan is noninvasive and nondestructive Using MRI, it is possible to generate markedly different types of tissue contrast by changing excitation and repetition times that make it a very versatile tool for imaging different structures of the body, particularly the brain. This review article has presented a comprehensive and extensive overview of the state-of-the-art conventional and CNN deep learning-based skull stripping (or whole-brain extraction) methods in brain MRIs. Many of the available approaches employed mainly T_1W brain MRIs due to good soft-tissue contrast. Even though most of the published algorithms have relatively decent results in the field of neuroimage analysis, there is a certain distance and discontinuity between the research community and clinical applications. In most cases, clinicians and practitioners still depend on manual skull stripping due to the communication gap and lack of interaction between the researchers and practitioners. The objective of several skull stripping tools is to do only the research, and they are barely useful for clinicians.

2.2 Predictive analytics approach for stroke prediction[2]

Authors: S. Dev, H. Wang, C. S. Nwosu, N. Jain, B. Veeravalli and D. John

Publication details: *Healthcare Anal.*, vol. 2, Nov. 2022

This paper provides an in-depth analysis of patient attributes within electronic health records for stroke prediction. Various features were systematically examined, including feature correlation analysis and a stepwise approach to identify an optimal feature set. The findings indicate that most features are weakly correlated, and a combination of four key features—A, HD, HT, and AG—may significantly contribute to stroke prediction. Additionally, principal component analysis (PCA) was conducted, revealing that nearly all principal components are required to capture a high variance. However, variable loadings suggest that the first principal component, which accounts for the highest variance, could potentially

explain the underlying patterns in stroke prediction. Furthermore, three machine learning models were implemented using different feature sets and PCA configurations. Among these, the neural network performed best when utilizing A, HD, HT, and AG as features, achieving an accuracy of 78% and a miss rate of 19%. In this multivariate analysis, the dataset is transformed into a set of values of linearly uncorrelated variables called principal components such that maximum variance is extracted from the variables. These principal components act as summaries of the features of the dataset. These new basis functions do not have a physical interpretation. However, these new basis functions are linear combinations of the original feature vectors. In this work, we do not restrict ourselves on the feature analysis using traditional feature elimination techniques.

2.3 A survey on genetic algorithm-based feature selection for disease diagnosis system [3]

Authors: S. Sindhiya and S. Gunasundari

Publication details: *Proc. IEEE Int. Conf. Comput. Commun. Syst.*, pp. 164-169, Feb. 2014.

Computer Aided Diagnosis (CAD) has been a rapidly growing, dynamic area of research in medical imaging. In recent years, significant and serious efforts have been made towards the development of the CAD system in diagnostic radiology. Machine learning (ML) plays a vital role in CAD because objects such as organs may not be signified precisely by a simple equation and therefore pattern recognition essentially involves learning from examples. In order to reduce the dimensions of the dataset and to increase the classification accuracy rate the feature selection has to be done. Feature Selection (FS) is an important issue in building classification systems. Evolutionary algorithms are an important emergent computing methodology. Genetic Algorithm (GA) being a heuristic search algorithm is generally used to detect important features for large dimensional datasets. This paper surveys the existing literature about the GA for the feature selection. This study also comprises a snapshot of GA from the author's perspective, including variations in the algorithm, modifications and refinements introduced to prevent the local convergence and hybridization of GA with other heuristic algorithms.

Genetic algorithm-based feature selection offers a promising approach for enhancing disease diagnosis systems by improving the interpretability, accuracy, and efficiency of predictive models. By reducing dimensionality and selecting the most relevant features, GAs can lead to better decision-making in clinical settings, ultimately contributing to more accurate and timely diagnoses. However, careful consideration of the computational demands and

algorithm parameters is necessary to achieve optimal results.

2.4 Predictive power of XGBoost_BiLSTM model [4]

Authors: A. Javeed, J. S. Berglund, A. L. Dallora, M. A. Saleem

Publication details: Int. J. Comput. Intell. Syst., vol. 16, no. 1, pp. 188, 2023.

This study introduces a hybrid machine learning model, **XGBoost_BiLSTM**, for diagnosing sleep apnea. The proposed model comprises two key modules: the first identifies and ranks the most important features, while the second performs the classification of sleep apnea. To assess its effectiveness, the **XGBoost_BiLSTM** model was evaluated using a 5-fold cross-validation approach. Its performance was compared against various state-of-the-art machine learning models, including a conventional LSTM model. Results indicate that **XGBoost_BiLSTM** achieved the highest accuracy of 97% while utilizing only the top six most relevant features. These features include type 2 diabetes, external injuries, mental and behavioral disorders, psychological stress and emotions, a comprehensive psychopathology rating scale, and respiratory system diseases—key risk factors for sleep apnea in older adults.

Overall, the proposed **XGBoost_BiLSTM** model outperformed conventional LSTM and other machine learning models, showcasing its potential for early sleep apnea detection and diagnosis. This study utilized electronic health records (EHR) as the dataset for experimentation. However, future research can explore multimodal datasets for improved performance. Since deep learning models perform optimally with larger datasets, collecting a dataset with an increased sample size will be beneficial for further advancements.

2.5 Analysis of k-fold cross-validation over hold-out validation on colossal datasets [5]

Authors: S. Yadav and S. Shukla

Publication details: *Proc. IEEE 6th Int. Conf. Adv. Comput. (IACC)*, 2016.

After evaluating the effectiveness of k-fold cross-validation compared to hold-out validation on large-scale datasets, the results presented in Table 5 indicate that the percentage of **C** is significantly higher than that of **D**. This suggests that k-fold cross-validation is a preferable choice over hold-out validation for quality classification in colossal datasets. Despite the additional computational time required, k-fold cross-validation generally yields **0.1–3%** higher accuracy compared to hold-out validation when applied to the same classification algorithm.

However, for extremely large datasets, the computational cost can be substantial. Therefore, it is crucial to determine an optimal **k** value to balance accuracy improvement with processing time. Based on the conducted experiments, we can estimate the appropriate **k** value where the accuracy gain justifies the time investment. Although testing was limited due to physical memory constraints, results obtained from slightly smaller datasets suggest optimal values, as outlined in the table. Further in-depth experimentation is necessary to validate these findings.

2.6 Bi-directional long short-term memory networks for relation classification [6]

Author: Shu Zhang, Dequan Zheng

Publication details: Pacific Asia Conference 2015

The automatic classification of semantic relations is an important task, which could offer useful information for many applications, such as question answering, information extraction, the construction and completion of semantic or relational knowledge base. In this work, we focus on the classification of semantic relations between pairs of nominals (Hendrickx et al., 2010). Given a sentence S with annotated pairs of nominal e1 and e2, the task is to classify which of the following nine semantic relations holds between the nominals: Cause-Effect, Instrument-Agency, Product-Producer, Content Container, Entity-Origin, Entity-Destination, Component-Whole, Member-Collection, Message-Topic, or Other if it does not belong to any of the nine annotated relations.

In the context of relation classification, a BiLSTM network takes a pair of entities and the sentence containing them as input, and learns to classify the relationship between those entities. The BiLSTM captures information from left and right sides of the entities to achieve contextualized representation of the relationship. These representations are then used to predict the relationship label, which could be something like "located_in," "works_for," or "married_to," depending on the dataset and task.

It proposes bidirectional long short-term memory networks (BLSTM) to solve the relation classification. For every word in each sentence, BLSTM has complete, sequential information about all words before and after it. Long distance relationship may be solved in some extent in this network.

3.PROPOSED SYSTEM

The proposed system utilizes deep learning methods to automatically detect strokes by examining brain CT scan images.. This eliminates the need for manual interpretation, reducing human error and improving diagnostic speed. The system can classify stroke types, such as ischemic or hemorrhagic, aiding doctors in making timely medical decisions.

1. Image Upload and Preprocessing

The system provides an intuitive web-based interface where users, including medical professionals and patients, can upload brain CT scans. Upon uploading, the images undergo crucial preprocessing steps such as noise reduction, contrast enhancement, resizing, and normalization. These steps improve the image quality, making it easier for the AI model to analyze and extract relevant features. By standardizing images before classification, the system ensures consistency across various scans, minimizing variability due to different imaging equipment or environmental factors. Effective preprocessing significantly improves the accuracy and reliability of the stroke classification model.

2. Deep Learning-Based Classification

The system is built around a deep learning model, specifically a Convolutional Neural Network (CNN), trained on a diverse dataset of stroke-related CT scans. This model identifies essential patterns and features within the images, enabling accurate classification into categories such as Normal, Ischemic Stroke, or Hemorrhagic Stroke. The CNN architecture comprises multiple layers, including convolutional, pooling, and fully connected layers, which facilitate feature extraction and classification. Furthermore, the system provides confidence scores alongside classification results, helping doctors evaluate the AI's certainty and cross-check predictions before making treatment decisions.**3.**

Database Storage and Patient Records Management

A structured SQLite or cloud-based database is implemented to securely store patient records, including CT scan results, medical history, and diagnosis reports. This feature allows medical professionals to access previous test results, track the progress of stroke patients, and ensure continuity in treatment. The system maintains confidentiality and adheres to medical data security standards such as HIPAA compliance, ensuring that patient information remains secure. By storing and organizing data efficiently, the system aids in research and analysis, allowing healthcare providers to gain insights into stroke trends and improve future treatments

.4. Doctor Consultation and Appointment Booking

To enhance communication between patients and healthcare professionals, the system incorporates an appointment scheduling feature, allowing users to book consultations with neurologists or radiologists. The AI-generated diagnosis functions as an initial report, which doctors can review before the consultation. With built-in telemedicine support, patients can receive medical advice and treatment recommendations remotely, minimizing the necessity for frequent hospital visits. This feature is especially valuable for individuals in rural or underserved areas with limited access to stroke specialists, ultimately improving healthcare accessibility and patient outcomes.

5. Admin and Doctor Access Panel

A dedicated dashboard for doctors and administrators ensures efficient management of patient records, AI-generated diagnoses, and system analytics. Medical professionals can review scan results, validate AI classifications, and make necessary adjustments. The admin panel enables authorized personnel to manage user access, oversee database integrity, and ensure the proper functioning of the stroke detection system. By providing a centralized control panel, the system maintains transparency and ensures that medical professionals retain control over critical decision-making processes.

6. Privacy and Security Measures

Recognizing the sensitive nature of the system, the platform prioritizes privacy and security. User information will be protected by encryption, ensuring that all personal details and communications remain confidential. Strict data protection protocols will be implemented to maintain the security of the system, giving users full control over their information.

4. METHODOLOGY

The methodology of the stroke detection system is structured to ensure efficient and accurate stroke diagnosis using deep learning and image processing techniques. The system follows a multi-step approach, from data acquisition to real-time deployment, ensuring that predictions are reliable, fast, and medically relevant.

1. Data Collection and Preprocessing

The first and most crucial step in building an AI-based stroke detection system is collecting a large and diverse dataset of brain CT scan images. These datasets are sourced from publicly available medical repositories, hospitals, and research institutions. The images include normal cases as well as stroke-affected scans, covering both ischemic and hemorrhagic strokes.

Since raw CT scan images often contain noise, artifacts, and varying brightness levels, preprocessing techniques are applied to enhance image quality. These include:

- **Noise Reduction:** Removing unnecessary noise using filters like Gaussian blur and median filters.
- **Contrast Enhancement:** Adjusting brightness and contrast using histogram equalization to make stroke regions more distinguishable.
- **Normalization:** Ensuring all images have consistent intensity ranges for better model training.
- **Image Resizing and Cropping:** Standardizing image dimensions to maintain uniformity across the dataset.

This preprocessing step ensures that the images are clear, standardized, and optimized for training the deep learning model.

2. Image Segmentation and Feature Extraction

After preprocessing, segmentation techniques are applied to isolate the brain region from the CT scan. This removes unwanted background details and enhances the focus on critical areas of the brain.

Key segmentation techniques include:

- **Thresholding Techniques:** Separating stroke-affected regions based on pixel intensity variations.

- **Edge Detection Algorithms:** Identifying boundaries of brain tissues using methods like Canny Edge Detection.
- **Region-Based Segmentation:** Dividing the image into meaningful regions based on intensity distributions.

Once segmentation is complete, **feature extraction** is performed to identify key patterns in the images. Features such as texture, shape, intensity variations, and symmetry are analyzed, helping distinguish between normal and stroke-affected brain tissues. These extracted features are critical for training the AI model.

3. Deep Learning Model Training

A deep learning model based on a Convolutional Neural Network (CNN) is utilized for stroke classification. CNNs are highly effective in medical image analysis as they automatically extract essential features without manual intervention.

Training Process:

After multiple training iterations, the model achieves high accuracy in distinguishing between normal, ischemic, and hemorrhagic stroke cases.

4. Classification and Prediction

Once the model is trained, it is deployed to classify new CT scans in real time. When a user uploads a CT scan, the model processes the image and predicts the likelihood of stroke. The classification output includes:

- **Normal Brain** – No stroke detected.
- **Ischemic Stroke** – Blocked artery leading to reduced blood flow.
- **Hemorrhagic Stroke** – Bleeding in the brain due to a ruptured blood vessel.

The model also provides a **confidence score**, indicating how certain it is about the prediction. This score helps doctors assess the AI-generated diagnosis before making clinical decisions.

5. Model Evaluation

To ensure model accuracy and efficiency, it is rigorously tested using various performance metrics, including:

- Accuracy: Measures the overall correctness of predictions.

- Precision: Indicates how many predicted stroke cases are actually correct.
- Recall: Evaluates the model's ability to detect actual stroke cases.
- F1-Score: Balances precision and recall for overall model effectiveness.

Additionally, k-fold cross-validation is performed to ensure the model generalizes well across different datasets. Confusion matrices are analyzed to identify misclassification patterns and further refine the model**.** This real-time accessibility ensures that stroke detection is fast, efficient, and widely available, especially in areas with limited medical facilities.

The methodology of this project is designed to ensure high accuracy and efficiency in stroke detection. By integrating deep learning with medical imaging, the system provides an automated, reliable, and scalable solution for early stroke diagnosis. With continuous improvements and real-world validations, this approach has the potential to significantly enhance stroke detection and patient care worldwide

3. SYSTEM ARCHITECTURE

The **system architecture** of the stroke detection system is designed to facilitate an efficient and accurate automated diagnosis of brain strokes using deep learning and medical imaging techniques. It consists of multiple interconnected modules that work together to ensure seamless data flow, processing, and real-time stroke detection. The architecture follows a structured pipeline, starting from image acquisition to prediction and final reporting.

3.1 Data Acquisition Layer

This is the foundational layer where **CT scan images** of the brain are collected. The images are sourced from:

- Hospital databases and medical imaging repositories.
- Real-time patient CT scan uploads.
- Publicly available medical datasets for training purposes.

These images serve as input data for preprocessing and analysis.

3.2 Preprocessing and Feature Extraction Layer

Once images are acquired, they go through preprocessing to enhance quality and standardization. The key preprocessing techniques include:

- **Noise Reduction:** Removing unwanted artifacts from CT scan images.
- **Contrast Enhancement:** Improving visibility of stroke-affected regions.
- **Segmentation:** Extracting the brain region from the scan for focused analysis.
- **Feature Extraction:** Identifying key characteristics like texture, intensity, and asymmetry to differentiate normal and stroke-affected regions.

This layer ensures that the deep learning model receives **optimized input** for better accuracy.

3.3 Deep Learning Model Layer

At the core of the system, a **Convolutional Neural Network (CNN)** processes the images to classify them into three categories:

1. Normal Brain (No Stroke)

2. Ischemic Stroke (Blocked Blood Vessel)
3. Hemorrhagic Stroke (Bleeding in the Brain)

Model Components:

- **Convolutional Layers:** Detects patterns and textures in brain images.
- **Pooling Layers:** Reduces computational complexity while preserving important features.
- **Fully Connected Layers:** Maps extracted features to stroke classification.
- **SoftMax Activation:** Determines the final category with a confidence score.

The trained model is capable of processing **new CT scan images** in real-time, providing quick stroke classification results.

3.4 Database and Storage Layer

A secure **database system** is used to store and manage:

- Patient records and CT scan histories.
- Model training datasets and test results.
- AI-generated predictions for future reference.

Cloud-based storage ensures scalability, allowing doctors and patients to access reports remotely.

3.5 User Interface and Prediction Layer

The user interacts with the system through an intuitive web-based or mobile application. Features include:

- **CT scan Uploading:** Patients or doctors can upload scans for analysis.
- **Instant Stroke Prediction:** The AI model provides real-time classification results.
- **Confidence Score Display:** The system shows the accuracy level of each prediction.
- **Doctor Consultation Option:** Users can consult a neurologist for expert verification.

The UI ensures easy access to stroke diagnosis and medical recommendations.

3.6 Reporting and Doctor Verification Layer

After stroke classification, the system generates a **detailed report** containing:

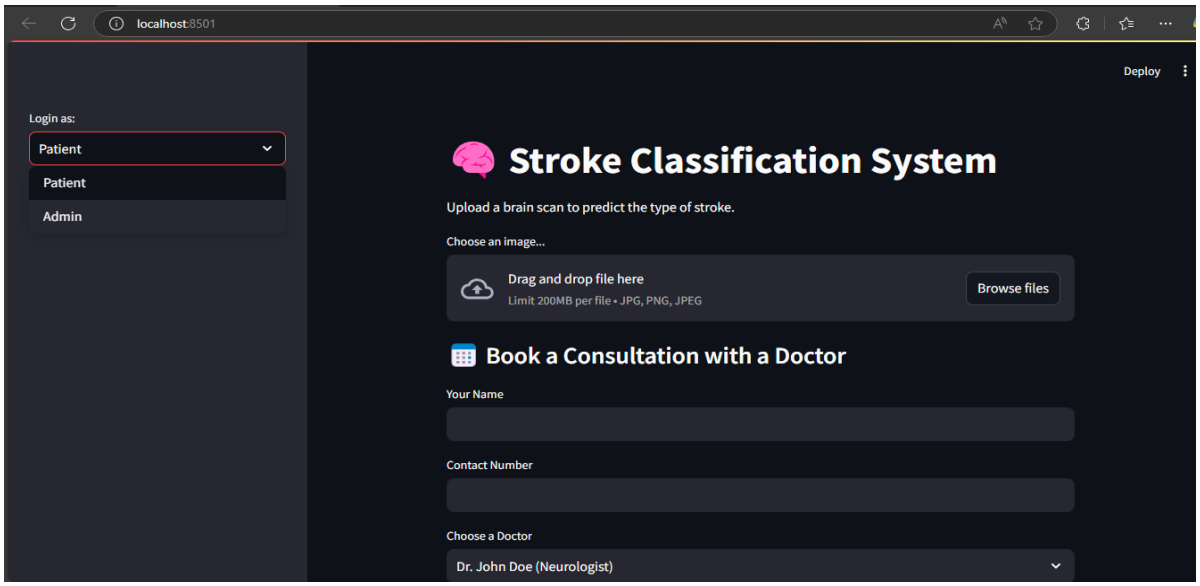
- Classification results (Normal, Ischemic Stroke, or Hemorrhagic Stroke).
- Confidence scores of the AI model.
- Recommendations for further medical assessment.

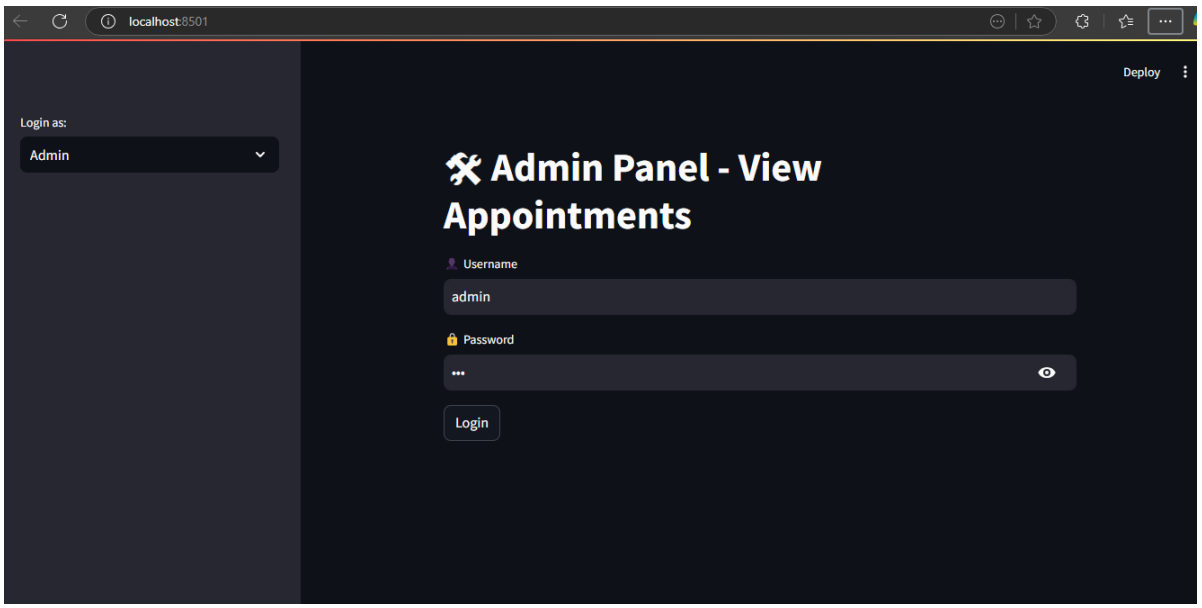
Doctors can review the AI-generated diagnosis, verify the results, and provide expert comments. The report can be downloaded or shared with medical professionals.

6. MODULES

6.1 User Authentication and Admin Dashboard Module

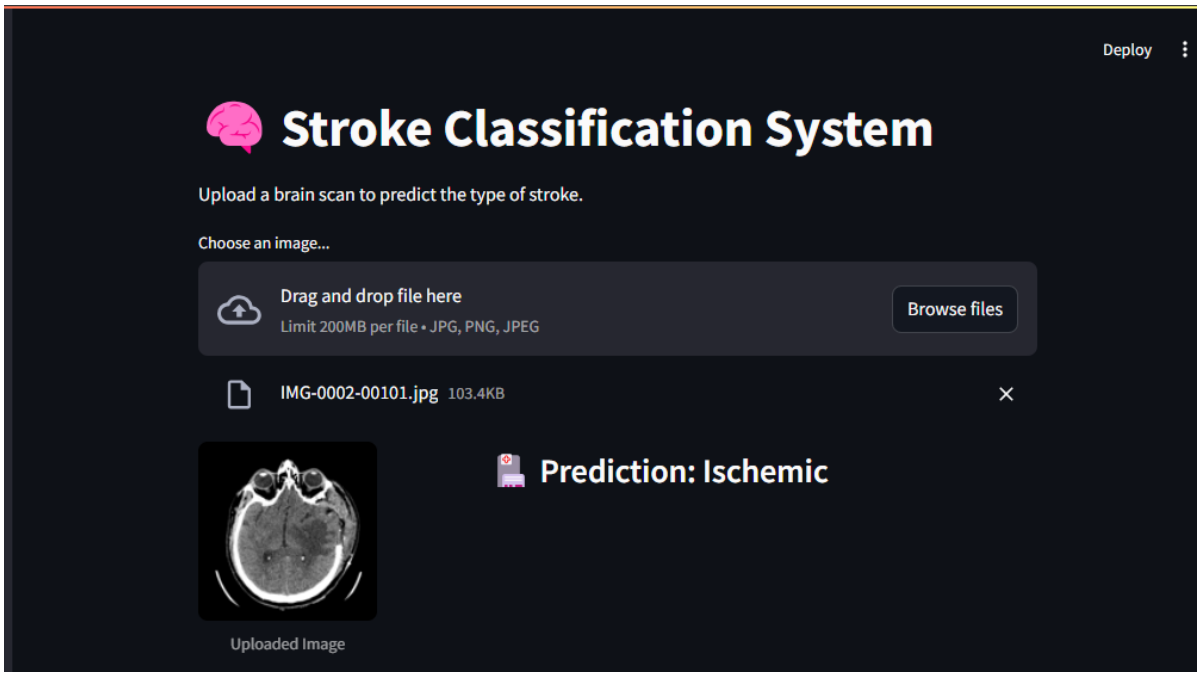
This module manages user roles, allowing patients and administrators to log in securely. Patients can upload scans and book appointments, while admins can view, manage, and track appointments. Security features such as password hashing and session management ensure data privacy.

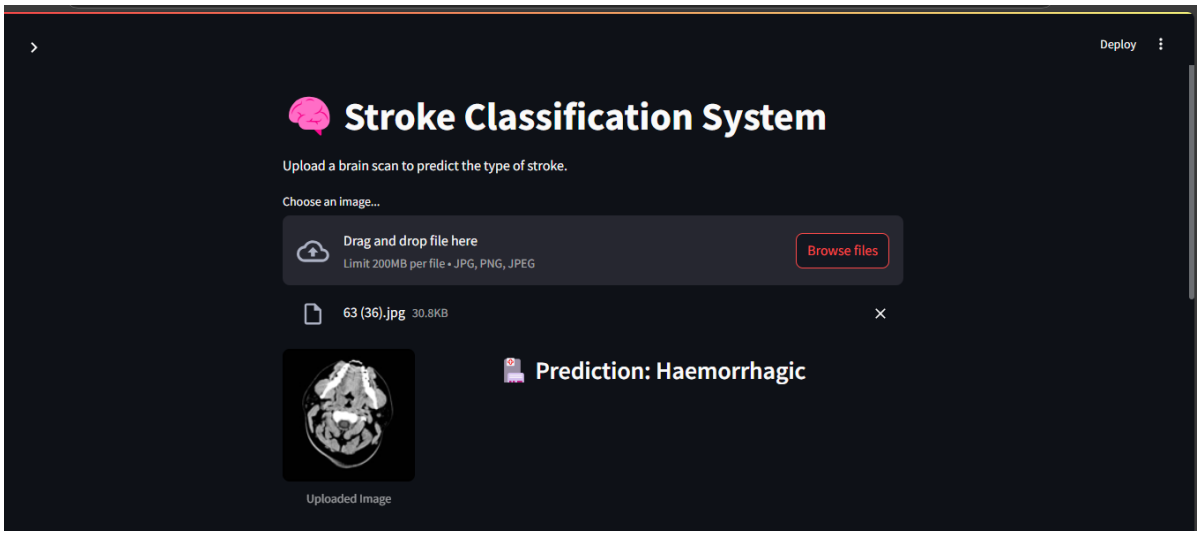




6.2 Image Upload and Preprocessing Module

Patients can upload brain scans in various formats (JPG, PNG, JPEG). The system resizes images, removes noise, and applies preprocessing techniques to prepare them for stroke classification. This step ensures that images are optimized for accurate model predictions.



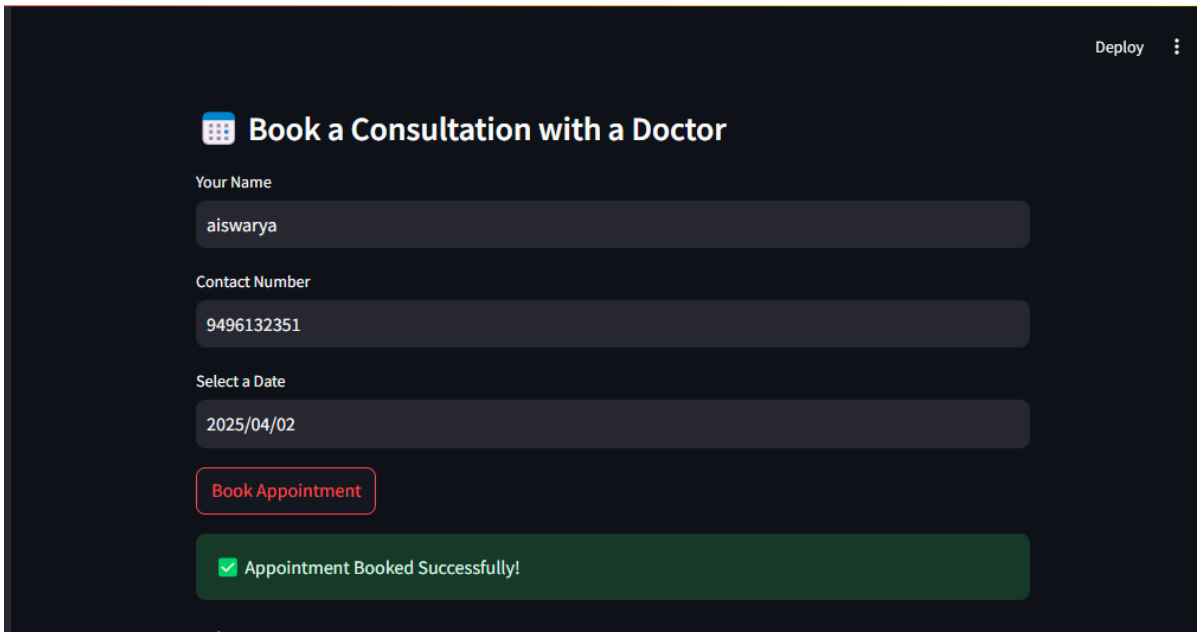


6.3 Appointment Booking and Management Module

Patients can book consultations with neurologists, stroke specialists, or radiologists after receiving their scan results. This module allows users to select doctors, choose appointment dates, and store booking details. Admins can manage these appointments through the admin panel.

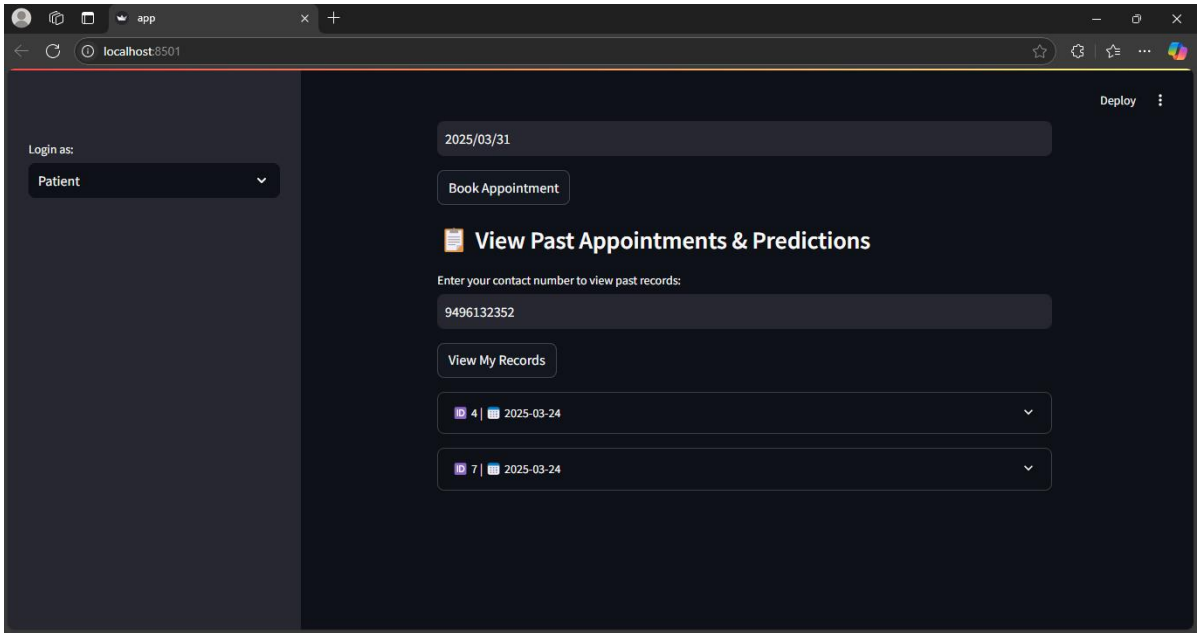
This module features an easy-to-use interface where patients can **select a doctor, choose a suitable date and time, and provide their contact information.** Once the appointment is confirmed, the system securely stores the details in the database and provides a confirmation message to the patient. Additionally, automated reminders help reduce missed appointments.

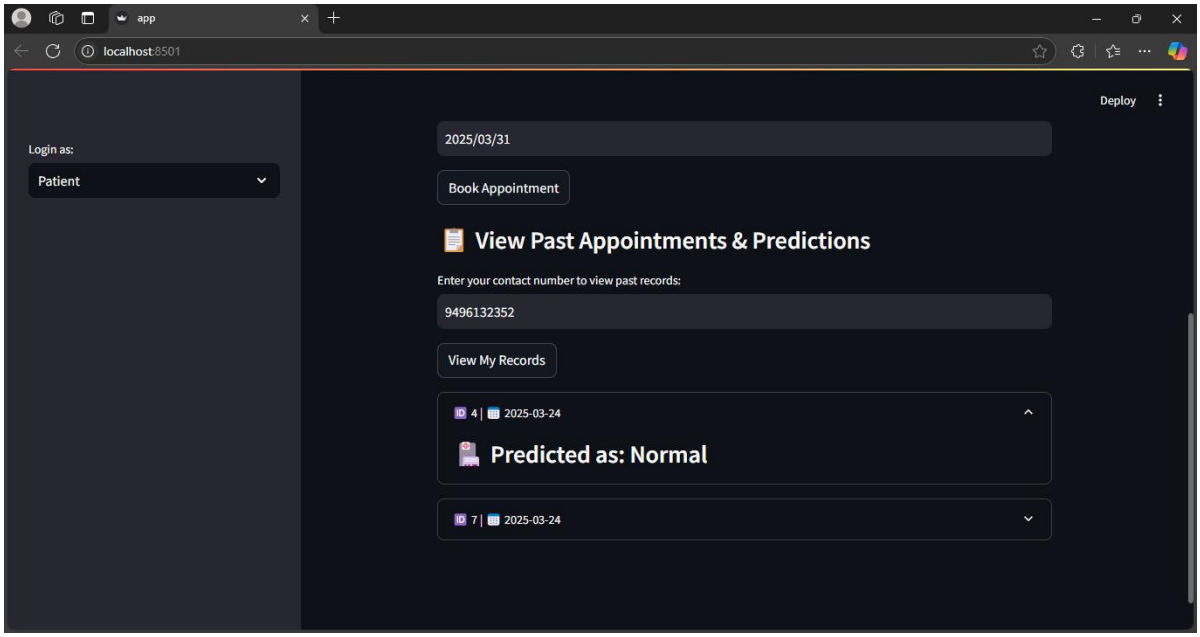
On the **admin side**, administrators can **view, update, or cancel appointments** as necessary. A dashboard displays scheduled consultations, making it easier to manage doctor availability and prevent overlapping appointments. The system also maintains a **record of past and upcoming appointments,** ensuring efficient tracking of patient visits and optimizing workflow for healthcare providers.



6.4 Past Appointments and Scan History Module

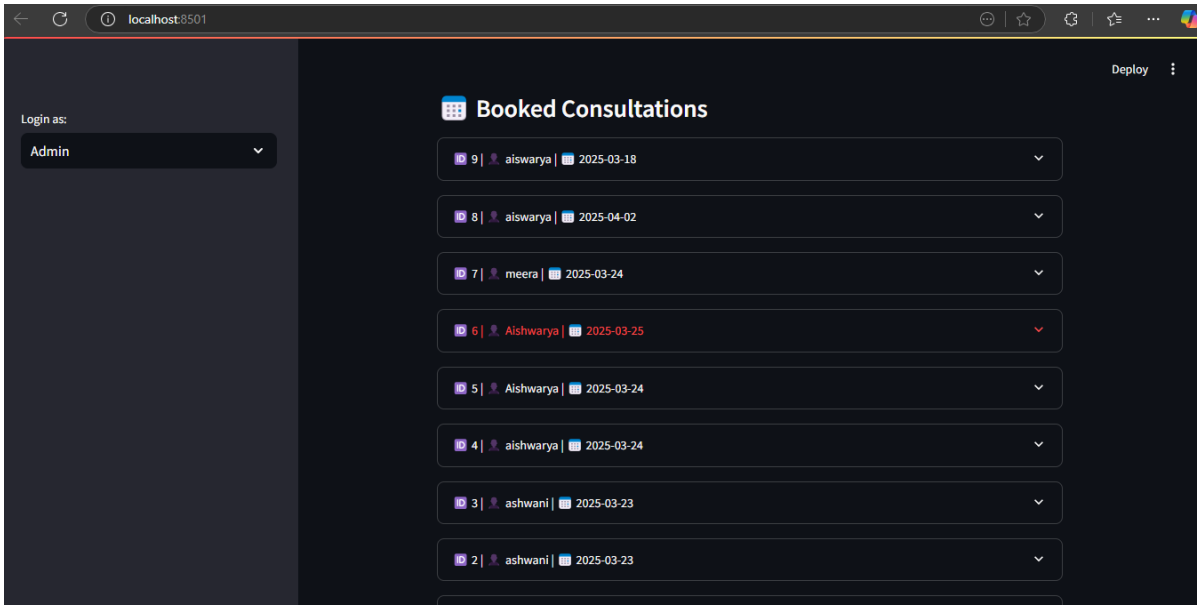
Patients can access their previous stroke predictions and appointments. This module retrieves past scan results and consultations from the database, providing an overview of medical history. Admins can also track patient records for better monitoring and follow-up.

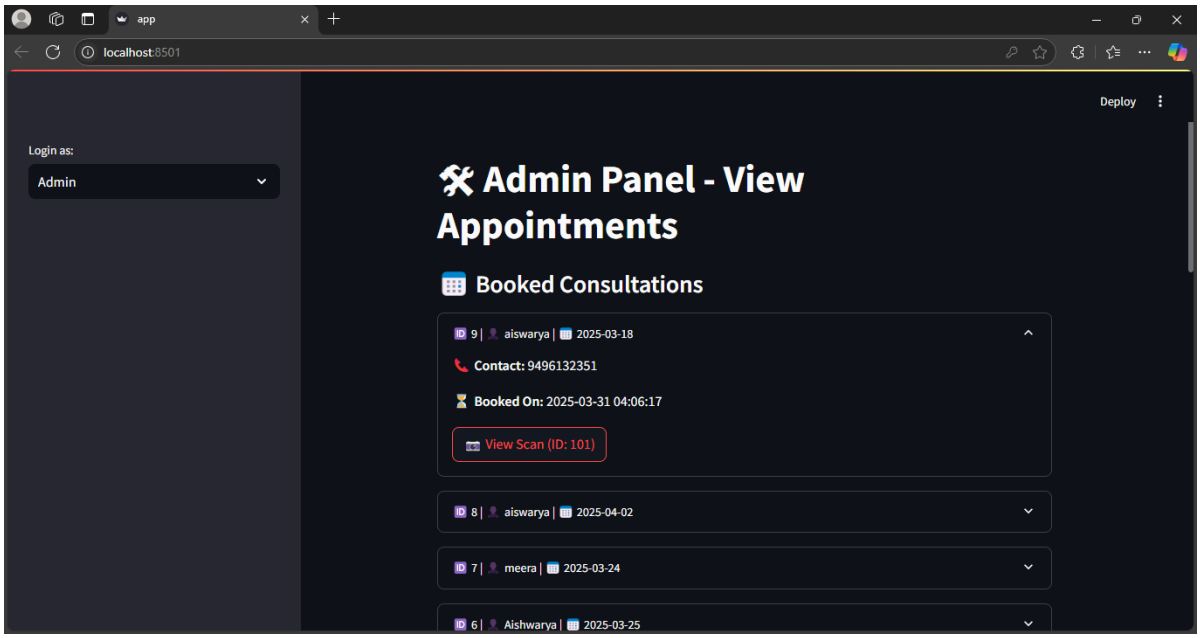




6.5 Admin Viewing Appointments and Scan Module

The admin panel allows easy management of patient appointments and stroke scan records. Admins can view, filter, and track past consultations while accessing brain scan images with predictions. The system ensures secure data handling, enabling authorized personnel to review and manage records efficiently.





7. DIAGRAMS

7.1 DFD (Data Flow Diagram).

A **Data Flow Diagram (DFD)** is a graphical representation of how data flows through a system. It visually maps how the data is stored and retrieved. DFDs are used to understand, analyze, and improve systems by breaking them down into processes, data flows, and data stores.

Levels of a DFD

1. **Level 0 (Context Diagram)**: This is the highest-level DFD that represents the system as a whole with a single process and external entities interacting with it. It provides a broad overview.
2. **Level 1 DFD**: This breaks down the previous Level 0 process into sub-processes,
3. **Level 2 DFD**: Breakdown of Level 1 processes into smaller sub-processes.

DFD Symbols & Relationships

1. **External Entity (Rectangle/Oval)**

    - Represents users or systems that interact with the system but are external to it (e.g., "User," "Doctor," "Admin").
    - They provide inputs to the system and receive outputs.

2. **Process (Circle or Rounded Rectangle)**

    - Represents the actions or functions performed by the system (e.g., "User Registration," "Booking Consultation").
    - Processes transform data from inputs to outputs.

3. **Data Store (Open Rectangle/Two Horizontal Lines)**
4. **Data Flow (Arrow)**

DFD LEVEL 0

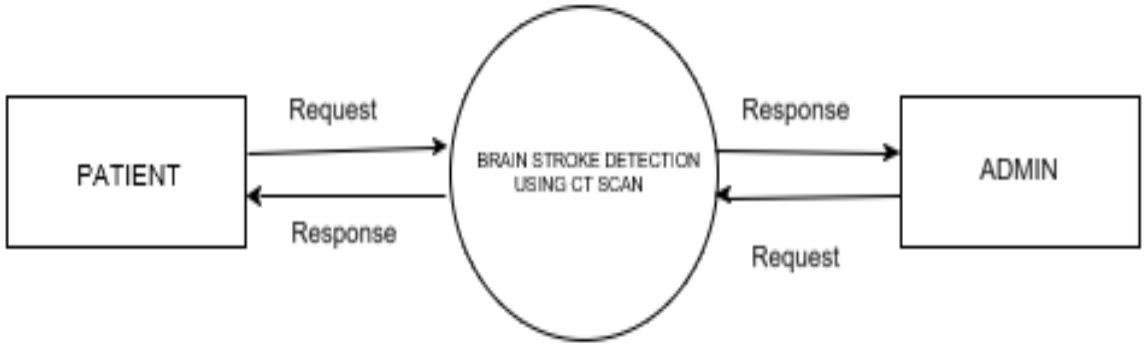


Fig 1: Level0 DFD

DFD LEVEL 1- PATIENT

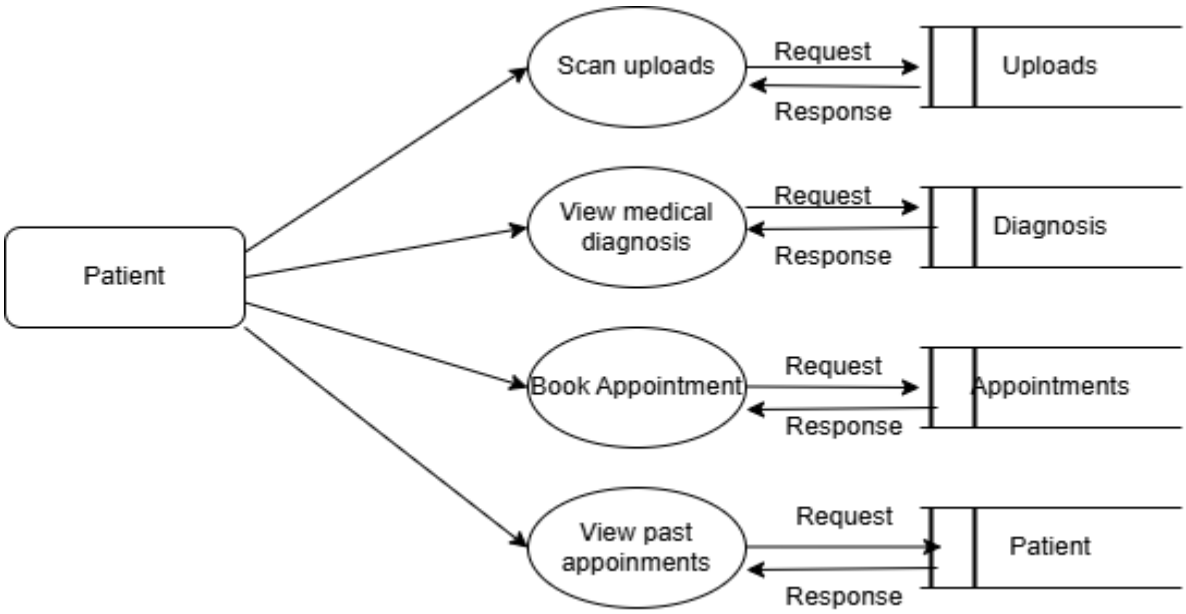


Fig 2: Level 1 DFD-PATIENT

DFD LEVEL 2-ADMIN

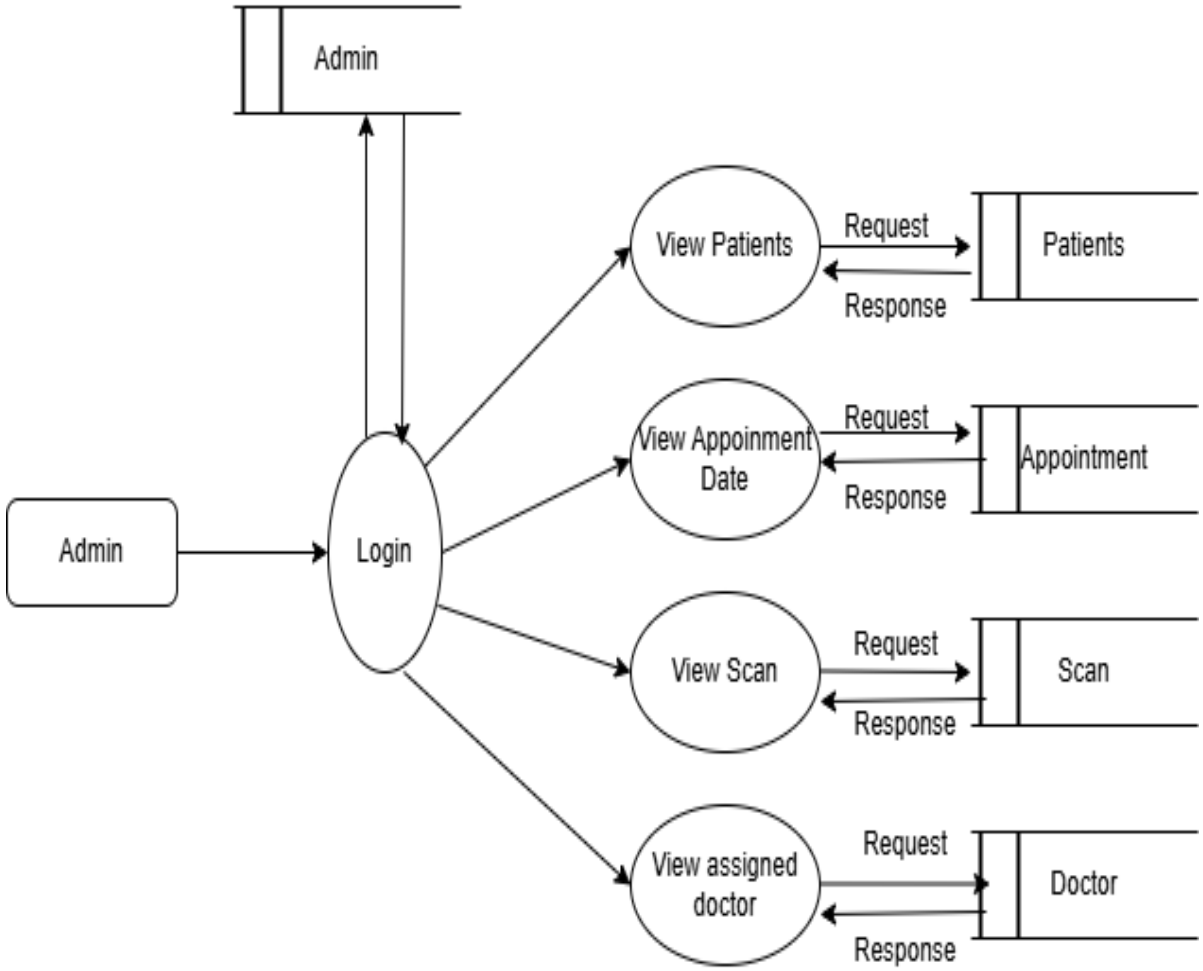


Fig 3: Level 1 DFD-Admin

7.2 Activity Diagram

Activity Diagram Symbols and Their Meanings:

1. **Initial Node (●)**:
    - The entry point of the activity diagram, indicating where the workflow or process begins. It has no incoming flows, only outgoing.
2. **Activity/Action (Rounded Rectangle)**:
    - Represents a specific action or task that occurs during the workflow. Each action typically involves some operation or decision (e.g., "Fill Form," "Process Payment").
3. **Final Node (◎)**:
    - The endpoint of the diagram. It signifies the completion of all activities in the process and has only incoming flows, with no outgoing ones.
4. **Decision Node (◇)**:
    - A point where the process branches into two or more possible paths based on a condition or decision (e.g., "Is payment successful?"). Each outgoing flow represents a different condition.
5. **Merge Node (◇)**:
    - Combines multiple alternative paths back into one. This does not involve a condition, it simply merges flows without decision-making.
6. **Fork Node (Thick Horizontal/Vertical Line)**:
    - Splits one action into multiple concurrent flows, allowing activities to happen simultaneously. For example, after "Log In," both "Check Account" and "Show Dashboard" can occur at the same time.
7. **Join Node (Thick Horizontal/Vertical Line)**:
    - Combines multiple parallel flows back into one, ensuring all tasks are completed before continuing the process.
8. **Control Flow (→)**:
    - A directional arrow representing the flow of control between activities or decisions. It shows the sequence of tasks in the process.
9. **Object Flow (Dashed Arrow)**:
    - Represents the flow of data or objects between activities. It shows how information moves through the process.

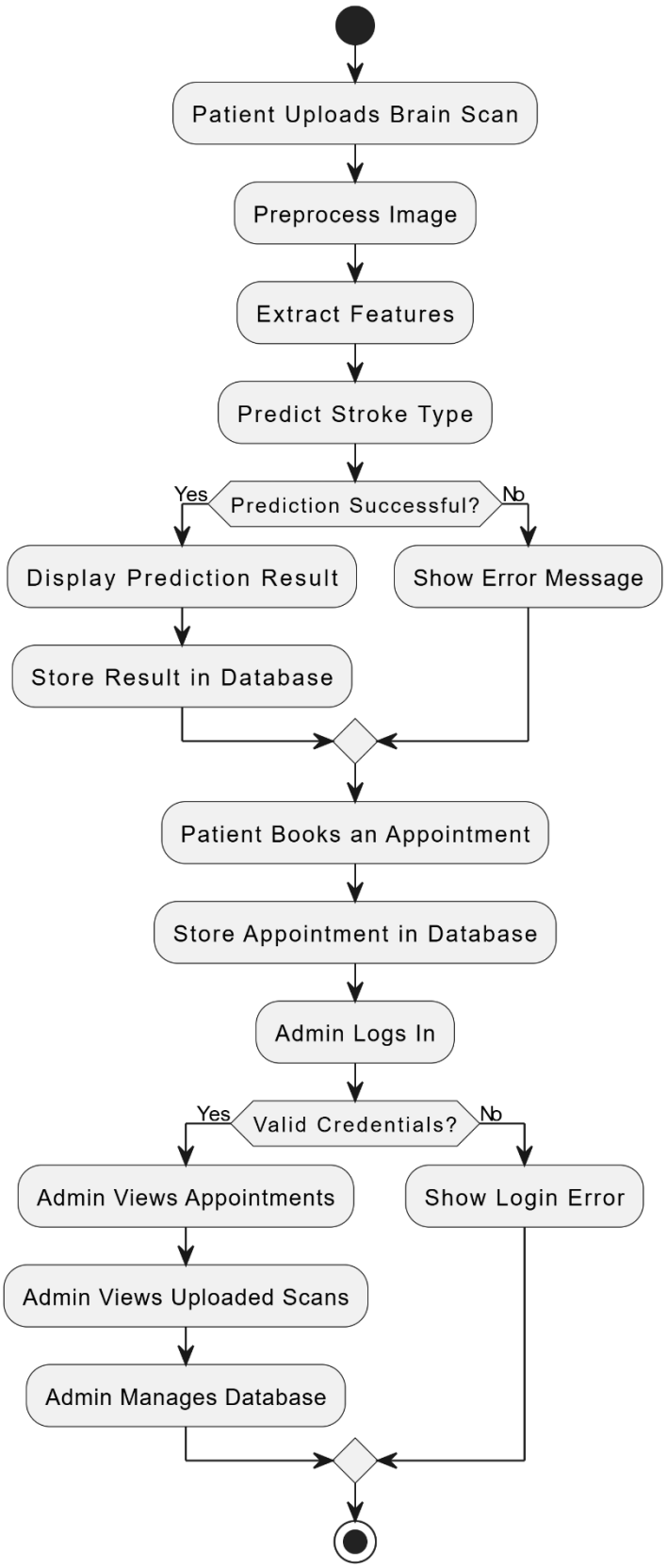


Fig 4: ACTIVITY DIAGRAM

7.3 Class Diagram

The **Class Diagram** for the Mental Health Support System outlines the main entities, including **User**, **Doctor**, **Admin** and **Appointment**. It illustrates the attributes and methods of each class and their relationships, showing how users interact with doctors and administrators within the system. This diagram serves as a blueprint for understanding the structure and functionality of the application.

Class Diagram Symbols and Their Meanings

1. **Class (Rectangle with Three Sections)**:

    - **Functionality**: Represents an object or entity in the system, encapsulating its attributes (data) and behaviors (functions).

2. **Association (Line with Optional Arrow)**:

    - **Description**: A simple line connects two classes, representing a relationship. The optional arrow shows the direction of the relationship (e.g., "User" owns "Profile").
    - **Functionality**: Indicates how classes interact or communicate with each other.

3. **Multiplicity (Numbers on Association Line)**:

    - **Description**: Indicates the number of instances involved in the relationship (e.g., "1" or "1..*").
    - **Functionality**: Shows how many objects from one class are associated with another (e.g., a "User" can have multiple "Appointments").

4. **Generalization/Inheritance (Solid Line with Empty Triangle Arrow)**:

    - **Description**: A line with a hollow triangle arrow pointing toward the parent class.
    - **Functionality**: Represents inheritance, where a subclass (e.g., "Admin") inherits attributes and methods from a parent class (e.g., "User").

5. **Aggregation (Line with Empty Diamond)**:

- **Description**: A line with an empty diamond at the class that contains another (e.g., "Department" has "Employees").
- **Functionality**: Represents a whole-part relationship, where the part (e.g., "

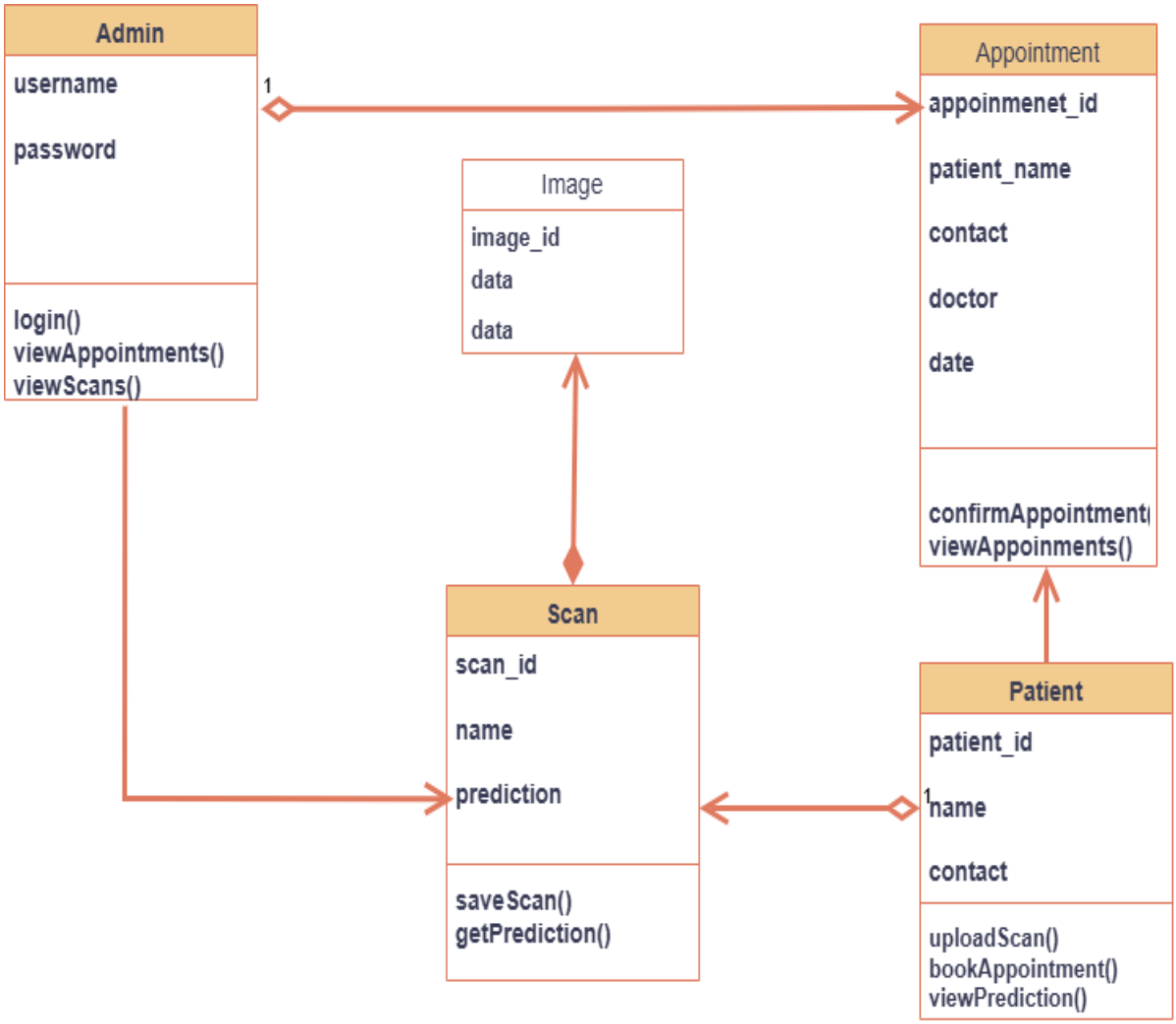


Fig 5: CLASS DIAGRAM

7.4: Use Case Diagram

A Use Case Diagram visualizes the interactions of users (actors) and the system. It illustrates the relationships between actors and the various use cases, providing a clear understanding of the system's

Use Case Diagram Shapes:

- **Actor (Stick Figure or Rectangle):**
  Represents users or external systems that interact with the system.
- **Use Case (Oval):**
  Represents a function or service that the system provides to the actors.
- **System Boundary (Rectangle):**
  It defines the extent of the system, encapsulating all the use cases within it.

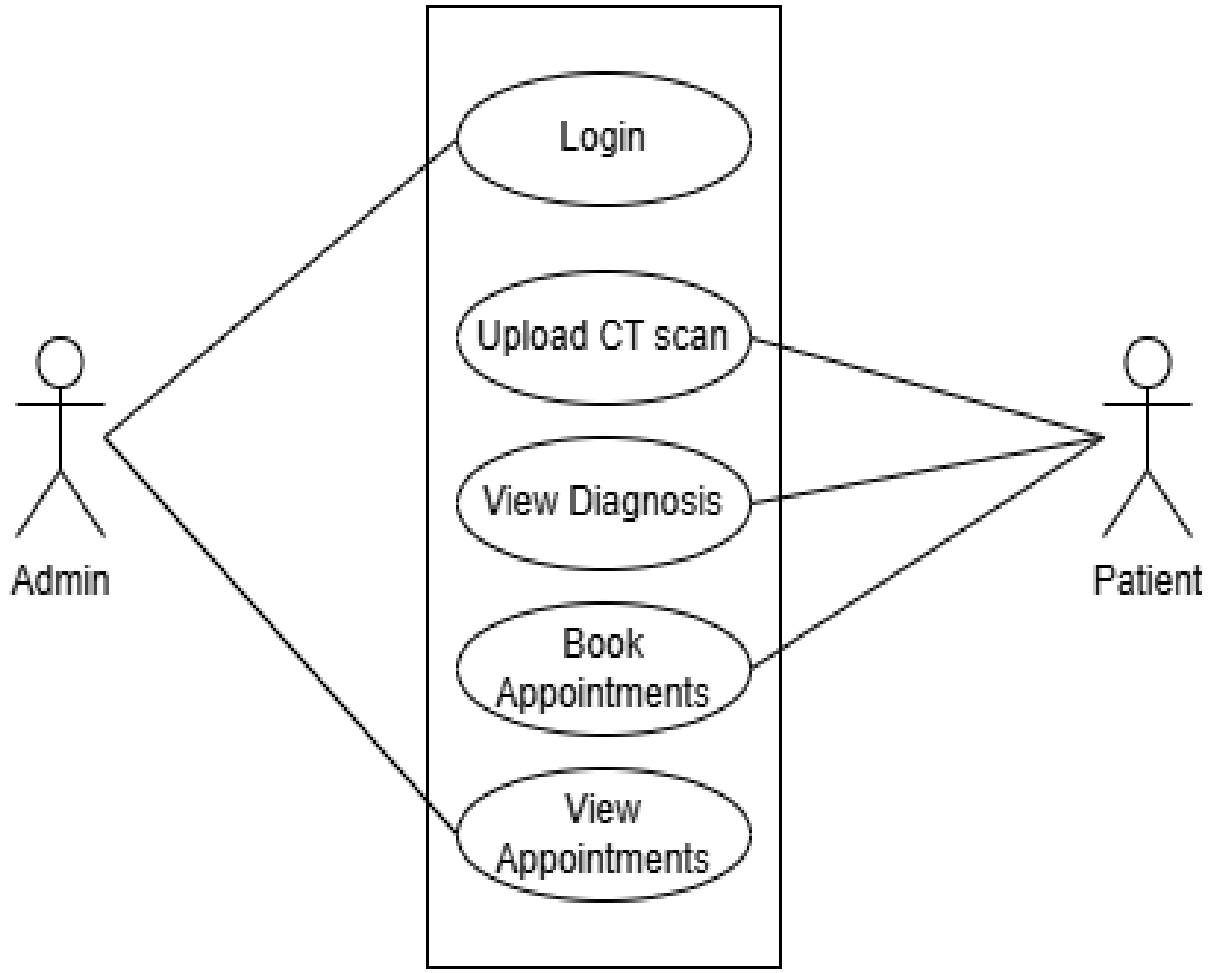


Fig 6: USE-CASE DIAGRAM

8. TESTING

Testing is a crucial phase in the development of any system, ensuring that all components function correctly and meet user requirements. In this project, rigorous testing was conducted to verify the accuracy of troke classification model, the efficiency of data management, and the overall performance of the system. The primary goal was to identify and resolve any potential issues before deployment, ensuring a smooth and reliable experience for users.

One of the most critical aspects of testing in this project was validating the stroke classification model. The model was tested with a variety of brain scan images to assess its accuracy in classifying normal, ischemic, and hemorrhagic strokes. The results were analyzed using performance metrics such as precision, recall, and F1-score to measure the effectiveness of the classification. Any inconsistencies in predictions were carefully examined, and necessary refinements were made to enhance the model's reliability.

Apart from model validation, the integration of different system components was thoroughly tested. The interaction between the patient interface, the classification model, and the admin panel was tested to ensure seamless data flow and prevent errors.

Additionally, user experience played a key role in testing. Patients and administrators were involved in user acceptance testing to ensure the system was easy to navigate and provided accurate results. Their feedback was considered in refining the system, improving response times, enhancing security measures, and optimizing the overall usability of the platform.

Below are the different types of testing conducted on the project:

Types of Testing Conducted on the Project

1. Unit Testing

Unit testing was conducted to evaluate the functionality of individual components, such as image preprocessing, stroke classification, and database operations. Each module was tested separately to ensure that it performed correctly before being integrated into the system.

- **Image Upload & Preprocessing Module:** Tested image resizing, format conversion, and normalization functions. Different image formats were uploaded to check the robustness of preprocessing.

- **Stroke Classification Module:** Verified that the machine learning model loaded correctly and produced consistent predictions for predefined test images.

- **Database Management Module:** Ensured that data insertion, retrieval, and deletion operations were correctly executed for patient records and scan results.

- Appointment **Booking Module:** Checked the validation of input fields like patient name, contact number, and selected date to prevent incorrect data entries.

- Admin **Panel Module:** Tested admin authentication functions to confirm login credentials were correctly verified

2. Functional Testing

Functional testing was performed to verify that all features of the system worked as expected. This testing ensured that users could seamlessly interact with the platform without encountering errors.

- Stroke Classification Module: Ensured that the correct stroke type (Normal, Ischemic, Haemorrhagic) was displayed based on model predictions.
- Appointment Booking Module: Checked whether users could successfully select doctors, pick dates, and book appointments.
- Admin Panel Module: Verified that admins could access patient data, view scan results, and manage appointments efficiently.

3. Integration Testing

Integration testing was carried out to confirm that different components of the system interacted correctly.

- Image **Upload & Stroke Classification:** Tested whether the uploaded images were correctly preprocessed and passed to the model for prediction.

- **Database Management & Appointment Booking:** Verified that booked appointments were stored in the database and were correctly retrievable by both patients and administrators.

- **Admin Panel & Database:** Ensured that the admin panel successfully fetched and displayed stored scans and appointments from the database.

4. Performance Testing

Performance testing assessed the system's efficiency under varying workloads.

- Stroke Classification Module: Measured the model's inference time to ensure that predictions were generated quickly.
- Database Management Module: Tested system performance while handling multiple simultaneous queries to ensure smooth operations.
- Appointment Booking Module: Assessed response time when multiple users attempted to book appointments simultaneously.

5. Security Testing

Security testing was conducted to safeguard sensitive patient data and prevent unauthorized access.

- Database Management Module: Performed SQL injection testing to check for vulnerabilities in database queries.
- Admin Panel Module: Ensured secure login with encrypted credentials to prevent unauthorized access.
- Appointment Booking Module: Verified input validation to prevent malicious data entries.

6. User Acceptance Testing (UAT)

User acceptance testing involves real users interacting with the system to provide feedback on its usability and effectiveness. Feedback from UAT was used to rectify the system and improve its overall user experience.

- Patients Tested: Image upload, stroke prediction results, and appointment booking features.
- Doctors Tested: Access to stored patient data, scan images, and appointment

By conducting these tests, the system was optimized for accuracy, performance, security, and usability, ensuring reliability in stroke detection and patient management.

Tools Used for Testing

- **PyTest:** Used for unit testing different functions, such as image preprocessing, model predictions, and database operations.
- **Selenium:** Utilized for functional and UI testing to ensure smooth navigation and proper interaction with the user interface.
- **Postman:** Used for API testing to validate data exchange between the frontend and backend.
- **SQLite Browser:** Helped in manually verifying stored data and testing database queries for correctness.

Testing Process

1. Test Plan

A test plan was created to define the scope, objectives, and approach of testing. The main objectives were:

- Ensuring the stroke classification model provides accurate predictions.
- Verifying seamless integration between different modules (image upload, classification, database, appointments, admin panel).
- Checking system performance under different loads.
- Identifying and fixing security vulnerabilities.
- Ensuring user-friendliness through UI testing.

2. Test Case Development

Test cases were designed to cover all functionalities of the system. Each test case included:

- **Test ID:** Unique identifier for tracking.
- **Test Scenario:** Description of the feature being tested.
- **Preconditions:** Any required setup before execution.
- **Test Steps:** Actions to be performed.
- **Expected Result:** Desired system response.
- **Actual Result:** System's actual behavior.

- **Status:** Pass or Fail based on comparison with the expected result.

3. Test Execution

- Unit tests were executed using PyTest, covering individual functions like image processing, database operations, and prediction logic.
- Functional tests were carried out using Selenium to check UI interactions and navigation.
- API tests with Postman ensured that data was correctly transmitted between the frontend and backend.
- Performance tests using JMeter simulated multiple users booking appointments simultaneously.
- Security tests were performed using OWASP ZAP to detect possible vulnerabilities.

4. Defect Reporting

- Bugs and issues found during testing were recorded in a defect tracking system (e.g., JIRA, GitHub Issues).
- Each defect report included:
    - **Defect ID:** Unique identifier.
    - **Module Affected:** Feature where the issue occurred.
    - **Description:** Brief explanation of the issue.
    - **Severity:** Critical, Major, Minor.
    - **Steps to Reproduce:** Instructions to replicate the bug.
    - **Assigned To:** Developer responsible for fixing it.
    - **Status:** Open, In Progress, Resolved, Closed.

  Developers fixed the reported defects, and testers verified the fixes in subsequent test cycles.

5. Regression Testing

After bug fixes and feature updates, regression testing was performed to ensure that no

existing functionality was broken.

- Automated regression tests were run using Selenium for UI testing.
- Test cases were re-executed to confirm that resolved issues did not introduce new bugs.
- Performance and security tests were repeated to validate stability after modifications.

Testing played a crucial role in ensuring the reliability, accuracy, and efficiency of the stroke classification system. Various testing methodologies, including **unit testing, integration testing, system testing, and user acceptance testing,** were implemented to validate the functionality of individual modules and their seamless interaction.

Unit testing helped verify the correctness of core functions, such as image processing, stroke classification, and database operations. Integration testing ensured smooth communication between different components, such as uploading brain scans, generating predictions, and booking appointments. System testing evaluated the overall performance, security, and usability of the platform, while user acceptance testing confirmed that the system met patient and admin requirements effectively.

Additionally, rigorous test planning, test case development, and defect tracking enabled the identification and resolution of errors before deployment. Regression testing was performed to maintain system stability after modifications. By following a structured testing approach, the project achieved **high accuracy in stroke classification, seamless appointment management, and a user-friendly interface.**

Overall, the testing process significantly contributed to enhancing the **system's dependability, ensuring an error-free user experience, and improving patient care through efficient stroke detection and consultation management.**

9. ADVANTAGES AND DISADVANTAGES

Advantages:

1. Early Stroke Detection

The system enables quick and accurate detection of strokes, helping doctors initiate timely treatment and reduce the risk of severe complications.

2. Automated Analysis

The use of deep learning eliminates manual analysis, reducing human error and increasing the reliability of stroke classification.

3. User-Friendly Interface

Patients and doctors can easily upload scans, view results, and book appointments through a simple and intuitive interface.

4. Remote Access

The system allows users to access stroke predictions and book consultations from anywhere, making healthcare more accessible

Disadvantages:

1. Dependency on Image Quality

The accuracy of the model depends on the quality of the uploaded brain scans; poor-quality images may lead to incorrect predictions.

2. Limited Dataset Training

If the model is trained on a limited dataset, it may struggle to accurately classify rare or complex stroke cases.

3. Internet Dependency

Since the system operates online, it requires a stable internet connection, which may not be accessible in remote areas

4. False Positives or Negatives.

The AI model is not 100% accurate and may sometimes misclassify scans, leading to unnecessary panic or overlooked stroke cases

10. RESULTS AND CONCLUSION

Results

The stroke detection system accurately categorizes brain scans into three types: Normal, Hemorrhagic, and Ischemic strokes. Leveraging deep learning techniques, particularly EfficientNetB0 for feature extraction, the model achieves high precision in stroke classification. It efficiently processes CT scan images and provides rapid predictions, facilitating quicker diagnosis and decision-making.

Additionally, the system incorporates a secure database for storing patient records, enabling easy access to previous scans and scheduled consultations. The integrated appointment booking feature streamlines scheduling with specialists, minimizing treatment delays. Designed with a user-friendly interface, the system enhances accessibility for both patients and healthcare professionals, improving the overall efficiency of stroke diagnosis and consultation management. The model has been rigorously tested on various datasets to ensure accuracy, demonstrating its effectiveness in real-world medical applications.

Conclusion

The project successfully integrates artificial intelligence with medical diagnostics, offering a reliable and accessible stroke classification system. The combination of automated image processing and real-time prediction reduces manual effort and improves diagnostic accuracy. By providing a streamlined platform for stroke detection and consultation scheduling, the system aids in early intervention, which is critical in reducing stroke-related complications. Future enhancements may include expanding the dataset for better model generalization, improving algorithm performance, and incorporating additional imaging modalities like MRI for more comprehensive stroke detection. Furthermore, integrating cloud-based storage and telemedicine features could enhance accessibility and usability. Overall, the system represents a significant advancement in stroke diagnosis, contributing to improved patient outcomes and healthcare efficiency.

11. APPENDICES

```
import streamlit as st
import tensorflow as tf
import numpy as np
import sqlite3
import cv2
import os
import io
from tensorflow.keras.preprocessing import image  # type: ignore
from PIL import Image
import datetime

# ----------------- Load the Trained Model -----------------
@st.cache_resource
def load_model():
    model = tf.keras.models.load_model("DenseNet121_stroke_model.h5")
    return model

model = load_model()

# Define class labels
classes = ['Normal', 'Haemorrhagic', 'Ischemic']

# ----------------- Database Setup -----------------
conn = sqlite3.connect("stroke_predictions.db",
check_same_thread=False)
c = conn.cursor()

c.execute('''CREATE TABLE IF NOT EXISTS uploads
             (id INTEGER PRIMARY KEY, name TEXT, prediction TEXT, image
BLOB, date TIMESTAMP DEFAULT CURRENT_TIMESTAMP)''')
c.execute('''CREATE TABLE IF NOT EXISTS appointments
             (id INTEGER PRIMARY KEY, patient_name TEXT, contact TEXT,
doctor TEXT, date TEXT, scan_id INTEGER,
             booked_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
             FOREIGN KEY(scan_id) REFERENCES uploads(id))''')
conn.commit()

# ----------------- Initialize Session State -----------------
if "admin_logged_in" not in st.session_state:
    st.session_state.admin_logged_in = False
if "selected_image" not in st.session_state:
    st.session_state.selected_image = None
    st.session_state.selected_image_id = None

# ----------------- Sidebar for User Type Selection -----------------
user_type = st.sidebar.selectbox("Login as:", ["Patient", "Admin"])
```

```
47.
48.# ---------------- Function for Predicting Stroke Type ---------------
   --
49.import cv2
50.import numpy as np
51.
52.def is_ct_scan(image_array):
53.    """Improved function to check if an image is a valid CT scan."""
54.    # Resize the image to a consistent size (avoids bias based on
   dimensions)
55.    resized_image = cv2.resize(image_array, (256, 256))
56.
57.    # Convert to grayscale
58.    gray = cv2.cvtColor(resized_image, cv2.COLOR_RGB2GRAY)
59.
60.    # Calculate mean intensity
61.    mean_intensity = np.mean(gray)
62.
63.    # Compute contrast variance (helps detect CT scans)
64.    contrast_variance = np.var(gray)
65.
66.    # Histogram analysis: Count pixels in the mid-tone range (CT scans
   have more mid-tone pixels)
67.    hist = cv2.calcHist([gray], [0], None, [256], [0, 256])
68.    mid_tone_pixels = np.sum(hist[50:200])  # Count pixels in intensity
   range 50-200
69.
70.    # Debugging prints to check values
71.    print(f"Mean Intensity: {mean_intensity}")
72.    print(f"Contrast Variance: {contrast_variance}")
73.    print(f"Mid-tone Pixel Count: {mid_tone_pixels}")
74.
75.    # Adjust thresholds based on real CT scans
76.    if 40 < mean_intensity < 210 and contrast_variance > 1000 and
   mid_tone_pixels > 10000:
77.        return True  # Likely CT scan
78.    return False  # Likely non-CT
79.
80.@st.cache_data
81.def predict_stroke(image_array):
82.    """Processes and predicts stroke type from an image array."""
83.    image_array = image_array / 255.0  # Normalize pixel values
84.    prediction = model.predict(image_array)
85.    predicted_class = np.argmax(prediction)
86.    return classes[predicted_class]
87.
88.# ---------------- PATIENT SIDE ----------------
89.if user_type == "Patient":
```

```
    st.title("🧠 Stroke Classification System")
    st.write("Upload a brain scan to predict the type of stroke.")

    uploaded_file = st.file_uploader("Choose an image...", type=["jpg",
"png", "jpeg"])

    scan_id = None
    result = None  # Initialize result
    if uploaded_file is not None:
        # Load image in RGB format first for CT scan validation
        img = Image.open(uploaded_file).convert("RGB")
            open_cv_img = np.array(img)
            open_cv_img = cv2.resize(open_cv_img, (256, 256))  #
Standardize size for CT scan detection

            # Check if it's a valid CT scan before proceeding
            if not is_ct_scan(open_cv_img):
                st.warning("⚠️ This image does not appear to be a
valid CT scan. It might be an MRI or other type of scan.")
                st.write("Proceeding with prediction anyway...")  #
Show this message when image is not CT but we still want to predict.
                img = img.convert("L")  # Convert to grayscale if
it's not a CT scan
                img = img.resize((128, 128))  # Resize to match
model input shape
                img_array = image.img_to_array(img)
                img_array = np.expand_dims(img_array, axis=-1)  #
Add grayscale channel
                img_array = np.expand_dims(img_array, axis=0)  # Add
batch dimension
            else:
                img = img.convert("L")  # Convert to grayscale for
CT scan
                img = img.resize((128, 128))  # Resize to match
model input shape
                img_array = image.img_to_array(img)
                img_array = np.expand_dims(img_array, axis=-1)  #
Add grayscale channel
                img_array = np.expand_dims(img_array, axis=0)  # Add
batch dimension

            # Convert image to bytes for database storage
            img_byte_arr = io.BytesIO()
            img.save(img_byte_arr, format='PNG')
            img_byte_arr = img_byte_arr.getvalue()

            # Predict stroke type
            result = predict_stroke(img_array)
```

```
126.
127.             # Insert scan into database (even if previously
   uploaded)
128.             c.execute("INSERT INTO uploads (name, prediction, image)
   VALUES (?, ?, ?)",
129.                       (uploaded_file.name, result, img_byte_arr))
130.             conn.commit()
131.             scan_id = c.lastrowid  # Get scan ID
132.
133.             # Display prediction result
134.             col1, col2 = st.columns([1, 2])
135.             with col1:
136.                 st.image(img, caption="Uploaded Image", width=150)
137.             with col2:
138.                 st.write(f"### 🖨️ Prediction: {result}")
139.
140.         # ---------------- Booking Consultation ----------------
141.         st.subheader("🗓️ Book a Consultation with a Doctor")
142.         patient_name = st.text_input("Your Name")
143.         contact = st.text_input("Contact Number")
144.         appointment_date = st.date_input("Select a Date")
145.
146.         if st.button("Book Appointment"):
147.             if patient_name and contact and appointment_date:
148.                 c.execute("INSERT INTO appointments (patient_name,
   contact, date, scan_id) VALUES (?, ?, ?, ?)",
149.                           (patient_name, contact, appointment_date,
   scan_id))
150.                 conn.commit()
151.                 st.success("☑️ Appointment Booked Successfully!")
152.             else:
153.                 st.error("⚠️ Please fill in all fields!")
154.
155.         # ---------------- View Past Appointments & Predictions ---
   --------------
156.         st.subheader("📋 View Past Appointments & Predictions")
157.         patient_contact = st.text_input("Enter your contact number
   to view past records:")
158.
159.         if st.button("View My Records"):
160.             if patient_contact:
161.                 today_date = datetime.date.today().strftime("%Y-%m-
   %d")
162.                 c.execute('''SELECT a.id, a.doctor, a.date,
   u.prediction, u.image, u.id
163.                              FROM appointments a
164.                              LEFT JOIN uploads u ON a.scan_id = u.id
```

```
                                WHERE a.contact = ? AND DATE(a.date) < DATE(?)
                                ORDER BY DATE(a.date) DESC''', (patient_contact, today_date))
                past_appointments = c.fetchall()

                if past_appointments:
                    for appt in past_appointments:
                        scan_id = appt[5] if appt[5] else "N/A"
                        prediction_result = appt[3] if appt[3] else "Not Available"

                        with st.expander(f"🆔 {appt[0]} | 📅 {appt[2]}"):
                            st.write(f"### 🤖 Predicted as: {prediction_result}")

                            if appt[4]:  # If image exists
                                st.session_state.selected_image = appt[4]

                else:
                    st.info("No past records found.")
            else:
                st.warning("Please enter your contact number!")

    # ---------------- ADMIN SIDE ----------------
    elif user_type == "Admin":
        st.title("🛠 **Admin Panel** - View Appointments")

        # Admin login system
        if not st.session_state.admin_logged_in:
            username = st.text_input("👤 Username")
            password = st.text_input("🔒 Password", type="password")

            if st.button("Login"):
                if username == "admin" and password == "123":
                    st.session_state.admin_logged_in = True
                    st.success("☑ Login Successful!")
                else:
                    st.error("❌ Invalid credentials!")

        if st.session_state.admin_logged_in:
            st.subheader("📅 Booked Consultations")

            # Fetch appointments
            c.execute("SELECT * FROM appointments ORDER BY booked_at DESC")
```

```
206.             appointments = c.fetchall()
207.
208.             if appointments:
209.                 for appt in appointments:
210.                     with st.expander(f"🆔 {appt[0]} | 👤 {appt[1]}
    | 📅 {appt[4]}"):
211.                         st.write(f"📞 **Contact:** {appt[2]}")
212.                         st.write(f"⏳ **Booked On:** {appt[6]}")
213.
214.                         if appt[5]:  # If scan is associated
215.                             if st.button(f"📷 View Scan (ID:
    {appt[5]})"):
216.                                 c.execute("SELECT image FROM uploads
    WHERE id=?", (appt[5],))
217.                                 scan_img = c.fetchone()
218.                                 if scan_img:
219.                                     st.session_state.selected_image
    = scan_img[0]
220.                                     st.session_state.selected_image_
    id = appt[5]
221.
222.             # Display selected image
223.             if st.session_state.selected_image:
224.                 st.subheader(f"📷 Brain Scan (ID:
    {st.session_state.selected_image_id})")
225.                 img =
    Image.open(io.BytesIO(st.session_state.selected_image))
226.                 st.image(img, caption="Brain Scan", width=500)
227.
228.             if st.button("Logout"):
229.                 st.session_state.admin_logged_in = False
230.                 st.session_state.selected_image = None
231.                 st.session_state.selected_image_id = None
232.                 st.rerun()
```

12.REFERENCES

[1] H. Z. U. Rehman, H. Hwang and S. Lee "Conventional and deep learning methods for skull stripping in brain MRI". *Appl. Sci.*, vol. 10, no. 5, pp. 1773, Mar. 2020.

[2] S. Dev, H. Wang, C. S. Nwosu "Predictive analytics approach for stroke prediction". *Healthcare Anal.*, vol. 2, Nov. 2022.

[3] S. Sindhiya and S. Gunasundari "A survey on genetic algorithm-based feature selection for disease diagnosis system". *Proc. IEEE Int. Conf. Comput. Commun. Syst.*, pp. 164-169, Feb. 2014.

[4] A. Javeed, J. S. Berglund, A. L. Dallora, M. A. Saleem and P. Anderberg," Predictive power of XGBoost_BiLSTM model: A machine-learning approach for accurate sleep apnea detection using electronic health data", *Int. J. Comput. Intell. Syst.*, vol. 16, no. 1, pp. 188, Nov. 2023.

[5] S. Yadav and S. Shukla, "Analysis of k-Fold cross-validation over hold-out validation on colossal datasets for quality classification", *Proc. IEEE 6th Int. Conf. Adv. Comput. (IACC)*, pp. 78-83, Feb. 2016.

[6] Zihan Wang, Bo Yang, "Attention-based Bidirectional Long Short-Term Memory Networks for Relation Classification", 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing.

[7] A. Tursynova, B. Omarov, N. Tukenova, I. Salgozha, O. Khaaval, R. Ramazanov, et al., "Deep learning-enabled brain stroke classification on computed tomography mages", *Comput. Mater. Continua*, vol. 75, no. 1, pp. 1431-1446, 2023.

[8] G. Sailasya and G. L. A. Kumari, "Analyzing the performance of stroke prediction using ML classification algorithms", *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 6, 2021.

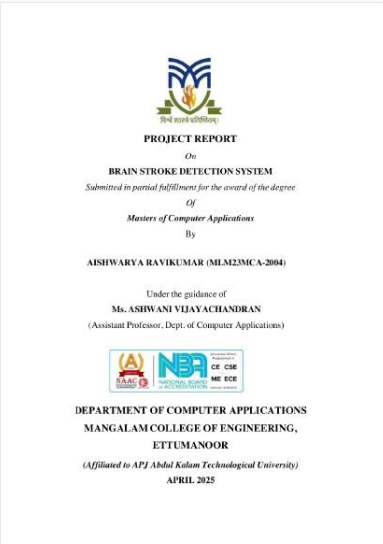
turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Aishwarya RAVIKUMAR
Assignment title:	PROJECT REPORT
Submission title:	pro.pdf
File name:	pro.pdf
File size:	5.33M
Page count:	53
Word count:	10,507
Character count:	63,504
Submission date:	02-Apr-2025 03:27AM (UTC+0000)
Submission ID:	2632206663



PROJECT REPORT

On

BRAIN STROKE DETECTION SYSTEM

Submitted in partial fulfillment for the award of the degree

Of

Masters of Computer Applications

By

AISHWARYA RAVIKUMAR (MLM23MCA-2004)

Under the guidance of

Ms. ASHWANI VIJAYACHANDRAN

(Assistant Professor, Dept. of Computer Applications)

DEPARTMENT OF COMPUTER APPLICATIONS

MANGALAM COLLEGE OF ENGINEERING,

ETTUMANOOR

(Affiliated to APJ Abdul Kalam Technological University)

APRIL 2025