A PROJECT REPORT

ON

HEART ANALYSIS USING DEEP LEARNING

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

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

Under the guidance

of

Dr. N. GAYATHRI DEVI, M.E, Ph.D.,

Associate Professor / CSE-(Artificial intelligence)

BY

D.THARUN	**20751A3312**
M.GANESH	**20751A3333**
M.LAKSHMIKANTH	**20751A3334**

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES, CHITTOOR-517127, A.P.

(Autonomous)

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

(2023-2024)

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES, CHITTOOR-517127, A.P.

(Autonomous)

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

This is to certify that the project work entitled "**Heart Analysis Using Deep Learning**" is a genuine work of

D.THARUN	**20751A3312**
M.GANESH	**20751A3333**
M.LAKSHMIKANTH	**20751A3334**

Submitted to the department of CSE - (Artificial intelligence & Machine Learning), in fulfilment of the requirements for the award of the degree of Bachelor of Technology in CSE - (Artificial Intelligence and Machine Learning) during the academic year 2024.

Signature of the Supervisor
Dr. N. GAYATHRI DEVI, M.E., Ph.D.,
Associate Professor,
Department of CSE-(Artificial intelligence),
Sreenivasa Institute of Technology and Management Studies, Chittoor, A.P.

Signature of the Head of Department
Mr. V. RAGHUBATHY ,M.Tech.,
Professor & HOD,
Department of CSE-(Artificial intelligence & Machine Learning),
Sreenivasa Institute of Technology and Management Studies, Chittoor, A.P.

Submitted for University Examination (Viva-Voce) held on……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

A Project of this magnitude would have not been possible without the guidance and co-ordination of many people. I am fortune in having top quality people to help, support and guide us in every step towards our goal.

Our team is very much grateful to the Chairman **Sri K. Ranganadham,** Garu for his encouragement and stalwart support. We are also extremely indebted to the Secretary **Sri D.K. Badri Narayana,** Garu for his constant support.

We express our sincere thanks to our Academic Advisor **Dr. K.L. Narayana**., **M.Tech., Ph.D,** further, we would like to express our profound gratitude to our principal **Dr.N.Venkatachalapathi, M.T.ech, Ph.D** for providing all possible facilities throughout the completion of our project work.

We express our sincere thanks to our Dean (Academics), **Dr.M.Saravanan, M.E., Ph.D.,** further we express our sincere thanks to our Head of the Department **Mr.V.Raghubathy M.Tech,** for his co-operation and valuable suggestions towards the completion of project work.

We express our sincere thanks to our guide **Dr.N.GAYATHRI DEVI, M.E., Ph.D.,** for offering us the opportunity to do this work under his guidance.

We express our sincere salutation to all other teaching and non-teaching staff of our department for their direct and indirect support given during our project work. Last but not the least, we dedicate this work to our parents and the Almighty who have been with us throughout and helped us to overcome the hard times.

D.THARUN **20751A3312**
M.GANESH **20751A3333**
M.LAKSHMIKANTH **20751A3334**

Course Outcomes for project work

On completion of project work we will be able to,

CO1. Demonstrate in-depth knowledge on the project topic.

CO2. Identify, analyze and formulate complex problem chosen for project work to attain substantiated conclusions.

CO3. Design solutions to the chosen project problem.

CO4. Undertake investigation of project problem to provide valid conclusions.

CO5. Use the appropriate techniques, resources and modern engineering tools necessary for project work.

CO6. Apply project results for sustainable development of the society.

CO7. Understand the impact of project results in the context of environmental sustainability.

CO8. Understand professional and ethical responsibilities while executing the project work.

CO9. Function effectively as individual and a member in the project team.

CO10. Develop communication skills, both oral and written for preparing and presenting project report.

CO11. Demonstrate knowledge and understanding of cost and time analysis required for carrying out the project.

CO12. Engage in lifelong learning to improve knowledge and competence in the chosen area of the project.

CO – PO MAPPING

CO\PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO.1	3	-	-	-	-	-	-	-	-	-	-	-	3	3
CO.2	-	3	-	-	-	-	-	-	-	-	-	-	3	3
CO.3	-	-	3	-	-	-	-	-	-	-	-	-	3	3
CO.4	-	-	-	3	-	-	-	-	-	-	-	-	3	3
CO.5	-	-	-	-	3	-	-	-	-	-	-	-	3	3
CO.6	-	-	-	-	-	3	-	-	-	-	-	-	3	3
CO.7	-	-	-	-	-	-	3	-	-	-	-	-	3	3
CO.8	-	-	-	-	-	-	-	3	-	-	-	-	3	3
CO.9	-	-	-	-	-	-	-	-	3	-	-	-	3	3
CO.10	-	-	-	-	-	-	-	-	-	3	-	-	3	3
CO.11	-	-	-	-	-	-	-	-	-	-	3	-	3	3
CO.12	-	-	-	-	-	-	-	-	-	-	-	3	3	3
CO	3	3	3	3	3	3	3	3	3	3	3	3	3	3

ABSTRACT

Deep learning techniques hold significant promise for improving heart disease diagnosis and analysis, addressing the urgent need for accurate diagnostic tools. Recent advancements in deep learning-based approaches for heart analysis have shown potential in applications such as detecting cardiac abnormalities from medical images and predicting heart disease risks. These techniques offer enhanced accuracy and efficiency compared to traditional methods, leading to earlier and more reliable diagnoses. However, challenges remain, including the use of multi-layer neural networks and the need for techniques like transfer learning and model explainability. Overcoming these obstacles requires ongoing research and development, particularly in handling vast and diverse datasets. This project aims to revolutionize heart disease detection and analysis by harnessing the capabilities of deep learning algorithms and making a web tool for analysis and results. Future directions in the field highlight the integration of multi-modal data and personalized medicine approaches, as well as the potential of deep learning in real-time cardiac monitoring systems. By leveraging these advancements, healthcare providers can offer more precise and timely interventions, ultimately improving patient outcomes and reducing the global burden of heart disease.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
API	Application Program Interface
AWS	Amazon Web Service
CT	Computed Tomography
CNN	Convolution Neural Network
CPU	Central Processing Unit
CNPK	Cognitive Tool Kit
DNA	Deoxyribonucleic acid
ECG	Electrocardigram
GPU	Graphics Processing Unit
GDPR	General Data Protection Regulation
HDT	Hardware Detection Tool
IDE	Integerated Development Environment
MRI	Magnetic Resonance Imageing
RAM	Random Access Memory
RNN	Recurrent Neural Network
SVM	Support Vector Machine
SSD	Solid State Drive
UCI	Unique Client Identifier
XAI	Explainable Artifical Intelligence

CHAPTER 1

INTRODUCTION

The main objective of this project is to transform heart disease detection and analysis using advanced deep learning algorithms. Heart disease continues to be a leading cause of death globally, with traditional diagnostic methods often lacking in accuracy and promptness. Leveraging recent advancements in deep learning, particularly in medical imaging and predictive analytics, this project aims to address these shortcomings. Central to this initiative is the use of deep learning models to analyze medical images, including echocardiograms, MRI scans, and CT scans, automating the identification of various cardiac abnormalities and significantly improving diagnostic precision and speed. Additionally, predictive models will assess an individual's risk of developing heart disease by taking into account factors such as medical history, genetic predispositions, lifestyle, and existing conditions.

This project tackles several challenges by utilizing multi-layer neural network architectures to handle complex cardiac data, enhancing the models' learning capabilities across diverse datasets. Transfer learning techniques are applied to use pre-trained models, boosting performance and cutting down on training time. Ensuring the explainability of these models is crucial, as it makes the decision-making process transparent and comprehensible for clinicians, thus overcoming a major barrier to clinical adoption. Looking ahead, the project aims to integrate multi-modal data, advance personalized medicine, and develop real-time cardiac monitoring systems. Combining data from various sources like medical images, electronic health records, and wearable devices will offer a holistic analysis of a patient’s heart health. Personalized medicine will tailor diagnostic and treatment plans based on individual patient data, resulting in more effective and customized care. Real-time monitoring systems will allow continuous heart health tracking, offering immediate alerts and necessary interventions.

To make these tools accessible, a Flask web application has been developed. This application features a user-friendly dashboard, medical image upload functionality, and tools for visualizing analysis and predictions. The interface provides a clear overview of the patient’s health metrics, recent scans, and risk predictions. Users can upload medical images for analysis, and the deep learning models will detect any cardiac abnormalities, with the results displayed on the dashboard. The application also provides risk predictions based on comprehensive patient data, using visual aids like risk scores and probability charts to facilitate understanding. Interactive plots enable users to delve into the data, examining specific areas in detail. The application’s trend analysis features allow for tracking changes in heart health over time and evaluating the effects of different treatments or lifestyle changes. Additionally, the integration of wearable device data supports real-time monitoring, alerting users to significant changes or potential issues.

By integrating these deep learning techniques and providing a user-friendly web application, this project seeks to enhance diagnostic accuracy and efficiency, facilitate earlier heart disease detection, and offer personalized care. These advancements have the potential to reduce healthcare costs by streamlining diagnostic processes and minimizing the need for invasive procedures. Ultimately, this project aims to transform cardiovascular healthcare by leveraging deep learning, reducing the global burden of heart disease, and saving lives. The project not only addresses current diagnostic limitations but also paves the way for future innovations. Integrating multi-modal data and advancing personalized medicine will deliver a more holistic, individualized view of a patient's health, ensuring that diagnostic and treatment plans are tailored to unique genetic, environmental, and lifestyle factors. Real-time cardiac monitoring

systems will enable continuous heart health observation, providing timely alerts and interventions, thereby improving patient outcomes and enhancing care quality. In summary, this project represents a significant leap forward in cardiovascular healthcare by incorporating deep learning algorithms, comprehensive data analysis, and a user-friendly interface to improve early detection, diagnosis, and management of heart disease. By continuing to refine these technologies, the project aims to equip healthcare providers with powerful tools to enhance care quality, save lives, and improve global health outcomes.

The project features a robust and user-friendly web application designed to make advanced heart disease diagnostic and predictive tools accessible to clinicians and patients. Central to this application is an intuitive dashboard that provides a comprehensive overview of patient health metrics, recent medical scans, and risk predictions. Users can easily upload medical images, such as echocardiograms, MRI scans, and CT scans, for detailed analysis. The deep learning models integrated into the application automatically analyze these images to detect cardiac abnormalities, with the results clearly displayed on the dashboard, highlighting any areas of concern.

The application also includes predictive analytics capabilities, generating risk assessments based on a patient's medical history, genetic information, lifestyle, and existing health conditions. These predictions are presented through visual aids such as risk scores and probability charts, making the data easy to interpret. Interactive plots are another key feature, allowing users to explore the analysis results in greater depth. Users can zoom in on specific data points, examine detailed information about each prediction, and track changes over time.

Incorporating advanced deep learning techniques, the application ensures high accuracy and efficiency in diagnostics and predictions. The use of multi-layer neural networks and transfer learning optimizes model performance and reduces training time. Ensuring model explainability, the application employs techniques to make the decision-making process transparent and understandable for clinicians, thereby gaining their trust and facilitating practical use in clinical settings.

Overall, this web application is designed to improve diagnostic accuracy, enable early detection of heart disease, and provide personalized care. By streamlining diagnostic processes and offering real-time monitoring, it has the potential to reduce healthcare costs and enhance patient outcomes, making it a valuable tool in the fight against heart disease.

CHAPTER 2

PROJECT OVERVIEW

The primary goal of this project is to revolutionize heart disease detection and analysis by leveraging the capabilities of deep learning algorithms. Heart disease remains a leading cause of mortality worldwide, and traditional diagnostic methods often fall short in terms of accuracy and timeliness. Recent advancements in deep learning, particularly in medical imaging and predictive analytics, offer promising solutions to these challenges. This project focuses on several key components to enhance the diagnosis and prediction of heart disease. Deep learning models are used to analyze medical images such as echocardiograms, MRI scans, and CT scans, automating the detection of various cardiac abnormalities and enhancing the precision and speed of diagnosis. Additionally, predictive models assess an individual's risk of developing heart disease by considering factors like medical history, genetic predispositions, lifestyle, and existing conditions. Key challenges include managing the complexity of cardiac data with advanced neural network architectures, applying transfer learning to improve model performance, and ensuring model explainability for clinical use.

Future directions involve integrating multi-modal data, advancing personalized medicine approaches, and developing real-time cardiac monitoring systems. These innovations aim to provide a comprehensive analysis of a patient's heart health, tailor diagnostic and treatment plans, and enable continuous monitoring for immediate interventions. To make these advanced tools accessible, a Flask web application has been developed, featuring an intuitive dashboard, upload functionality for medical images, and visualization of analysis and predictions. Interactive plots display outcomes of image analysis and risk predictions, allowing users to explore data and track trends over time. The application also integrates data from wearable devices for real-time monitoring and alerts.

The web application’s user interface includes a user-friendly dashboard that presents an overview of the patient's health metrics, recent scans, and risk predictions. Users can upload medical images, such as echocardiograms, MRI scans, and CT scans, for analysis. Once images are uploaded, deep learning models analyze them to detect cardiac abnormalities, and the results are displayed on the dashboard, highlighting areas of concern. The application also generates risk predictions based on the patient's medical history and other relevant data, presenting visualizations such as risk scores and probability charts to help users understand the predictions. Interactive plots allow users to explore the data further, zoom in on specific areas, and view detailed information about each prediction.

Incorporating trend analysis features, the web application enables users to track changes in the patient's heart health over time and observe the impact of different treatments or lifestyle changes. Integration with wearable devices allows for continuous, real-time monitoring of heart health, with the system generating alerts if significant changes or potential issues are detected. This real-time monitoring is crucial for providing immediate interventions, which can be lifesaving in critical situations.

CHAPTER 3

EXISTING SYSTEM

The existing system for heart disease detection and analysis primarily utilized traditional machine learning algorithms, which, although effective to some extent, had several limitations due to their reliance on single inputs and relatively simpler data processing techniques. This approach typically involved the use of classical machine learning methods such as decision trees, support vector machines (SVM), and logistic regression to analyze singular data points like demographic information, basic medical history, and limited clinical measurements. These algorithms, while useful, struggled with the complexity and variability inherent in cardiac data, often resulting in lower diagnostic accuracy and efficiency.

In the existing system, medical professionals would input single data points, such as age, cholesterol levels, blood pressure, and other isolated health metrics, into the machine learning models. These models would then generate risk assessments and diagnostic outputs based on pre-defined patterns recognized from these singular inputs. However, this method had several drawbacks. Firstly, it lacked the ability to integrate multi-dimensional data, which is crucial for a comprehensive understanding of heart disease. Heart conditions are influenced by a multitude of factors, including genetic predispositions, lifestyle choices, detailed medical histories, and a variety of physiological measurements. The single-input approach failed to capture this complexity, often leading to incomplete analyses and potentially inaccurate predictions.

Moreover, the traditional machine learning algorithms used in the existing system were not adept at handling large volumes of diverse data. They required significant manual feature engineering, where domain experts had to painstakingly identify and select relevant features from the data. This process was not only time-consuming but also prone to human error, which could further compromise the accuracy of the models. Additionally, these models lacked the ability to learn from new data over time, making it challenging to keep the system updated with the latest medical research and patient information.

Another significant limitation was the explainability of the machine learning models. While they could provide risk scores or predictive outcomes, they often operated as "black boxes," offering little insight into the underlying reasoning behind their predictions. This lack of transparency made it difficult for clinicians to trust and adopt these systems in their practice. The inability to provide clear explanations for their predictions also hindered the ability to fine-tune and improve the models based on clinical feedback.

Furthermore, the single-input, traditional machine learning approach did not effectively utilize advanced data visualization techniques. The outputs were typically presented in straightforward numerical or categorical formats, which did not facilitate deep insights or interactive exploration of the data. This limitation made it challenging for users to fully understand and act upon the predictions and recommendations provided by the system.

Despite these limitations, the existing system laid the groundwork for more advanced approaches by demonstrating the potential of machine learning in heart disease detection and analysis. It highlighted the importance of leveraging computational methods to assist in medical diagnostics and opened the door

for integrating more sophisticated techniques, such as deep learning and multi-modal data integration, which could overcome the shortcomings of the single-input approach.

Recognizing these limitations, the current project aims to build upon the foundation of the existing system by incorporating deep learning algorithms capable of analyzing multi-dimensional data. This approach will allow for a more holistic and accurate analysis of heart health by integrating various data sources, including medical images, comprehensive patient histories, and real-time physiological data from wearable devices. The goal is to develop a system that not only improves diagnostic accuracy and predictive capabilities but also enhances model explainability and user interaction through advanced visualization techniques and a user-friendly web application. By addressing the shortcomings of the existing system, the new project aims to revolutionize heart disease detection and management, ultimately leading to better patient outcomes and more efficient healthcare delivery.

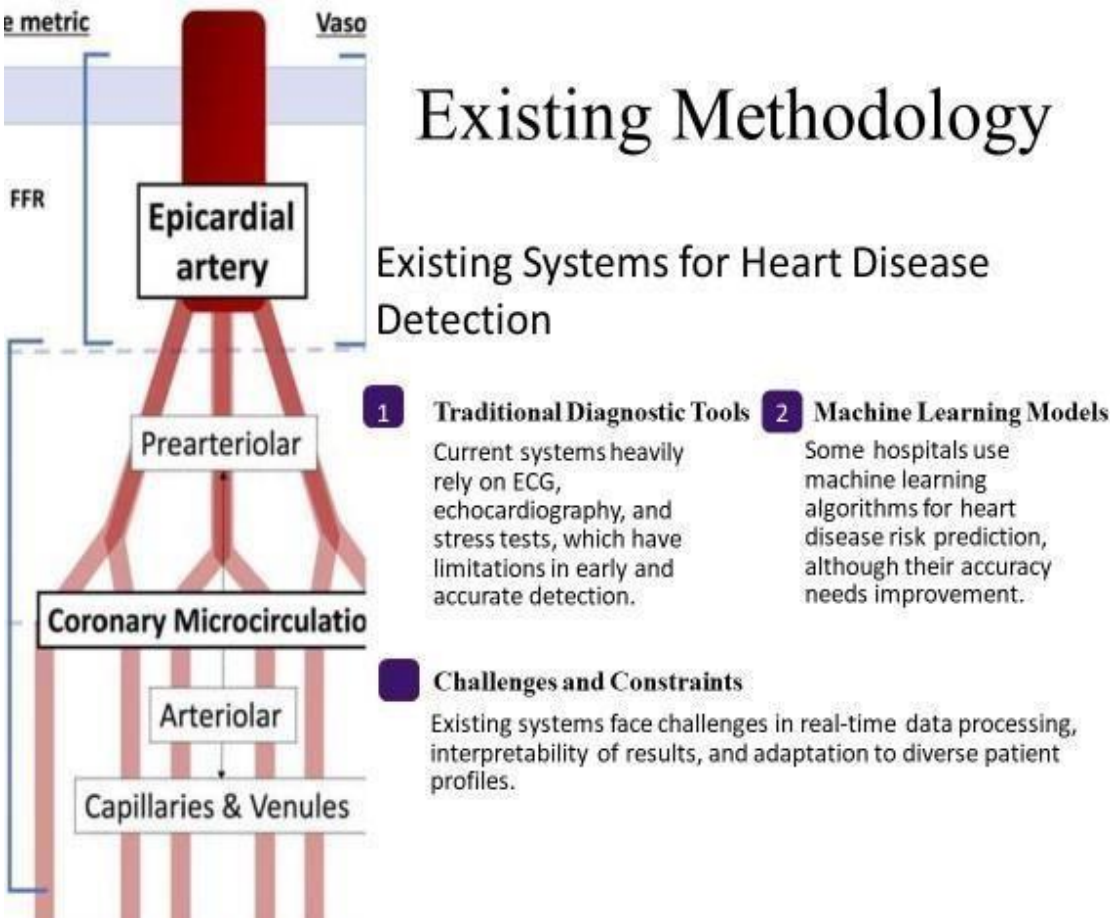

Fig 3.1 Existing system

CHAPTER 4

PROPOSED SYSTEM

The current project on heart analysis using deep learning represents a significant advancement over the existing system by addressing its key limitations and introducing a more sophisticated, accurate, and efficient approach to diagnosing and predicting heart disease. The existing system, which relied on traditional machine learning algorithms with single inputs, often fell short in handling the complex and multi-faceted nature of cardiac data. In contrast, the current project leverages the power of deep learning to analyze multi-dimensional data from diverse sources, providing a more comprehensive and accurate assessment of heart health.

Deep learning models, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are at the core of this project. These models excel at processing large volumes of data, identifying intricate patterns, and learning from them without the need for extensive manual feature engineering. This capability is crucial for analyzing medical images such as echocardiograms, MRI scans, and CT scans, which contain rich, detailed information about the heart's structure and function. By automating the detection of cardiac abnormalities from these images, the deep learning models significantly enhance diagnostic accuracy and speed, reducing the reliance on manual interpretation by healthcare professionals.

In addition to medical imaging, the project employs predictive models to assess an individual's risk of developing heart disease. These models integrate a wide range of data inputs, including comprehensive medical histories, genetic information, lifestyle factors, and real-time physiological data from wearable devices. This multi-modal data integration allows for a more holistic view of a patient's health, capturing the complexity and interplay of various risk factors. Consequently, the predictions are more accurate and personalized, tailored to the unique characteristics of each patient.

A key innovation in the current project is the use of transfer learning, where pre-trained models are adapted to new datasets. This approach not only enhances model performance but also reduces the time and computational resources required for training. By leveraging existing knowledge, the models can quickly adapt to new data, continuously improving their accuracy as more data becomes available. This adaptability is crucial for keeping the system up-to-date with the latest medical research and patient information.

Another significant improvement over the existing system is the emphasis on model explainability. The current project incorporates techniques to make the decision-making process of deep learning models transparent and understandable for clinicians. Tools like saliency maps, which highlight important features in medical images that influence the model's decisions, help demystify the "black box" nature of deep learning. This transparency is essential for gaining clinical trust and ensuring that the models can be reliably used ipractice. By providing clear explanations for their predictions, the models also facilitate clinical validation and continuous improvement.

The project also features a user-friendly web application developed using Flask, which makes advanced diagnostic and predictive tools accessible to clinicians and patients. The application includes an intuitive dashboard that presents an overview of patient health metrics, recent scans, and risk predictions. Users can upload medical images, which are then analyzed by the deep learning models to

detect cardiac abnormalities. The results are displayed on the dashboard, with visualizations that highlight areas of concern and provide detailed information about each finding.

Predictive analytics capabilities are another key feature of the web application. The application generates risk assessments based on comprehensive patient data and presents these predictions through visual aids such as risk scores and probability charts. Interactive plots allow users to explore the data in greater depth, examining specific predictions and trends over time. This interactivity enhances user engagement and understanding, making it easier to act upon the insights provided by the system.

In summary, the current project on heart analysis using deep learning offers significant improvements in accuracy and efficiency over the existing system. By integrating advanced deep learning models, multi-modal data, transfer learning, and model explainability, the project provides a more comprehensive, accurate, and transparent approach to diagnosing and predicting heart disease. The user-friendly web application ensures that these advanced tools are accessible to clinicians and patients, enhancing diagnostic processes and enabling personalized care. With these innovations, the project aims to transform cardiovascular healthcare, improving patient outcomes and reducing the global burden of heart disease.

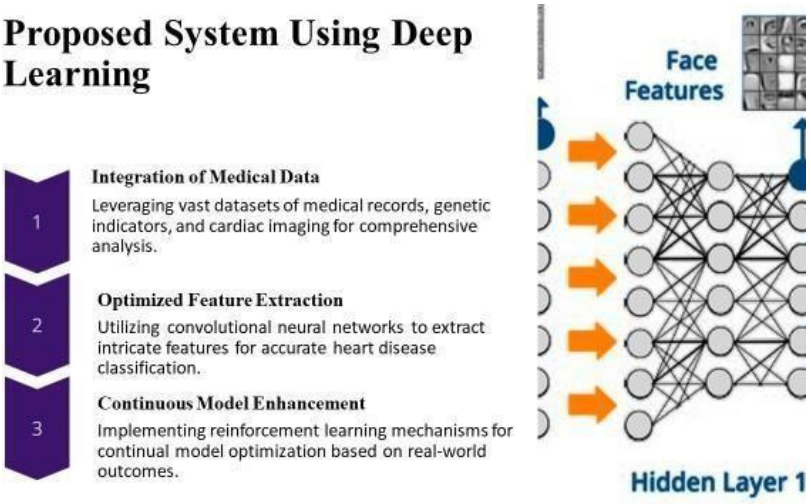

Fig 4.1 :Proposed system

CHAPTER 5

SYSTEM REQUIREMENTS

5.1 HARDWARE

The hardware requirements for the heart analysis project utilizing deep learning algorithms encompass a range of components to support efficient computation, data processing, and storage capabilities. These requirements are tailored to accommodate the complex computations involved in training and deploying deep learning models, as well as the handling of large volumes of medical data.

→Central Processing Unit (CPU):

A high-performance CPU with multiple cores is essential for running the deep learning algorithms efficiently. Processors with higher clock speeds and multi-threading capabilities are preferred to accelerate computations.

→Graphics Processing Unit (GPU):

GPUs are crucial for accelerating the training and inference processes of deep learning models. GPUs with CUDA support and a large number of CUDA cores are preferred for parallel processing, which significantly reduces training times.

→Random Access Memory (RAM):

Sufficient RAM is necessary to handle the large datasets used in training deep learning models. A minimum of 16GB RAM is recommended, although larger datasets may require 32GB or more for optimal performance.

→Storage:

High-speed storage is necessary for storing datasets, model weights, and intermediate results generated during training and inference. Solid-state drives (SSDs) are preferred for faster read/write speeds compared to traditional hard disk drives (HDDs).

→Networking:

Reliable networking capabilities are essential for accessing and transferring large datasets, especially if the data is stored remotely or accessed from multiple locations. High-speed internet connectivity is required for efficient data exchange.

Processor:	Intel Core i5 or equivalent
Memory (RAM):	8 GB
Storage:	500 GB HDD
Graphics:	Integrated Graphics

Table No 5.1: Software requirements

5.2 SOFTWARE

The software requirements for the heart analysis project using deep learning algorithms encompass a range of tools and frameworks essential for data processing, model development, visualization, and deployment. These requirements are tailored to support the development and implementation of deep learning models for analyzing medical data and predicting heart disease risk.

→Python:

Python serves as the primary programming language for developing deep learning models and web applications. The following versions are commonly used:

Python 3.x (e.g., Python 3.6, 3.7, or 3.8) for compatibility with the latest libraries and frameworks.

→Deep Learning Frameworks:

Deep learning frameworks provide the foundation for building, training, and deploying neural network models. Commonly used frameworks include:

→TensorFlow: A comprehensive open-source framework developed by Google for building and training deep learning models.

→PyTorch: An open-source deep learning framework maintained by Facebook's AI Research lab, known for its dynamic computation graph and ease of use.

→Keras: A high-level neural networks API written in Python that runs on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK).

→Data Processing Libraries:

Libraries for data preprocessing, manipulation, and augmentation are essential for preparing medical datasets. These may include:

→NumPy: A powerful library for numerical computing, providing support for multi-dimensional arrays and mathematical functions.

→**Pandas:** A data analysis library that offers data structures and functions for manipulating structured data.

→**OpenCV:** A library for computer vision tasks such as image processing and analysis, often used for handling medical images.

→**Web Development Framework:**A web development framework is necessary for building the user interface and backend of the web application. Flask, a lightweight and flexible micro-framework, is commonly used for this purpose.

→**Visualization Libraries:**

Visualization libraries are crucial for presenting analysis results and predictions in an understandable format. Matplotlib and Seaborn are popular choices for creating static plots, while Plotly and Bokeh offer interactive visualization capabilities.

→**Development Tools:**

Integrated Development Environments (IDEs) and text editors provide essential tools for coding, debugging, and version control. Common options include:

Operating System Windows: :	Windows 10 or later, macOS: macOS 10.15 (Catalina) or later, Linux: Ubuntu 18.04 or later
Programming Language :	Python 3.7 or later
Python Libraries:	Flask, Pandas, NumPy, Scikit-Learn, Matplotlib, Seaborn TensorFlow, Keras:
Integrated Development Environment (IDE)	Visual Studio Code: Source-code editor developed by Microsoft

Table No 5.2 Software requirements

5.2.1 Platform

Visual Studio Code (VS Code) has revolutionized the way developers write code, providing a seamless and efficient environment for software development. Its popularity stems not only from its versatility but also from its constant evolution and community-driven development. With frequent

updates and an active community of developers contributing to its ecosystem, VS Code remains at the forefront of modern coding practices. Its extensibility is a key aspect, allowing developers to tailor their editing experience to suit their specific needs. Whether it's through the installation of language-specific extensions, productivity-enhancing tools, or custom themes, users can personalize their workspace to optimize their workflow and productivity. Moreover, VS Code's support for a wide range of programming languages and frameworks makes it a go-to choice for developers working on diverse projects. From front-end web development to backend server applications, from mobile apps to cloud services, VS Code provides the tools and features needed to tackle any coding task with confidence.

its core features, VS Code excels in facilitating collaboration and integration with other development tools and services. Its Live Share extension enables real-time collaboration among team members, allowing multiple developers to edit and debug code together regardless of their physical location. Additionally, VS Code integrates seamlessly with popular version control platforms like GitHub, Bitbucket, and GitLab, empowering developers to manage their codebase efficiently and collaborate with others seamlessly. Furthermore, the editor's support for debugging, linting, and testing frameworks streamlines the development process, helping developers catch errors early and deliver high-quality code.

The user experience in VS Code is also noteworthy, with its intuitive interface and customizable layout catering to the preferences of individual developers. Features like the Command Palette, IntelliSense, and Snippets enhance productivity by providing quick access to commands, intelligent code completion, and reusable code snippets. Moreover, VS Code's integrated task runner simplifies common development tasks, allowing developers to automate build processes, run tests, and deploy applications without leaving the editor

Overall, Visual Studio Code stands as a testament to the power of open-source collaboration and community-driven development. Its versatility, extensibility, and user-friendly interface make it a top choice for developers worldwide, enabling them to write, debug, and collaborate on code with ease. As the software development landscape continues to evolve, VS Code remains a reliable and indispensable tool for modern software development, empowering developers to turn their ideas into reality with confidence and efficiency.

5.2.2 Languages

Python stands as the linchpin of the heart analysis project, providing a multifaceted toolkit that extends across various domains critical to the project's success. In the realm of data processing and analysis, Python's NumPy and Pandas libraries offer unparalleled efficiency in handling and manipulating large datasets, facilitating the extraction of valuable insights from complex medical data. Furthermore, Python's integration with powerful image processing libraries like OpenCV enables the project to leverage sophisticated algorithms for analyzing medical images, empowering clinicians with precise diagnostic tools. Deep learning, a cornerstone of the project's advancements, finds a natural home in Python through frameworks like TensorFlow and PyTorch. These frameworks empower developers to construct intricate neural networks capable of discerning subtle patterns within medical data, thereby enhancing the accuracy of heart disease diagnosis and risk prediction.

Beyond model development, Python's proficiency in web development shines through frameworks such as Flask and Django. Flask, with its minimalist design and flexibility, serves as the

ideal choice for constructing the project's backend, providing a robust foundation for data processing, model inference, and API endpoints. Meanwhile, Django offers a comprehensive suite of features for building scalable and secure web applications, ensuring seamless user experiences and efficient management of patient data. Python's proficiency in web development extends further with its integration with popular frontend technologies like React and Vue.js, enabling the creation of intuitive and interactive user interfaces that enhance engagement and usability.

Visualization, a pivotal component of the project's outcomes, receives ample support from Python's rich array of libraries. Matplotlib and Seaborn empower developers to craft visually compelling plots and charts that distill complex analysis results into actionable insights. Furthermore, Plotly's interactive visualization capabilities facilitate dynamic exploration of medical data, enabling clinicians to delve deeper into diagnostic trends and prognostic patterns. Python's versatility extends beyond its native capabilities, thanks to its seamless integration with a plethora of third-party tools and services. Whether deploying models on cloud platforms like AWS or Azure, interfacing with databases like PostgreSQL or MongoDB, or orchestrating CI/CD pipelines with tools like Jenkins or Travis CI, Python serves as a unifying force that binds together disparate components into a cohesive and functional ecosystem.

CHAPTER 6

SYSTEM ARCHITECTURE

The Heart Analysis project utilizes deep learning techniques to analyze and predict heart disease from medical data, aiming to accurately classify the presence or absence of heart disease based on various patient features. The data is collected from reliable sources like the UCI Machine Learning Repository's Heart Disease dataset, followed by extensive data cleaning to handle missing values, outliers, and inconsistencies. Feature engineering involves creating new features, normalizing data, and encoding categorical variables.

The deep learning model begins with an input layer that accepts features such as age, sex, blood pressure, and cholesterol levels. This is followed by several hidden layers, typically 2 to 5, consisting of fully connected (Dense) layers with ReLU activation functions to introduce non-linearity. Each layer may contain between 64 to 256 neurons. Dropout layers are integrated to prevent overfitting by randomly setting a fraction of input units to zero during training. The output layer is a fully connected layer with a sigmoid activation function, suitable for binary classification, producing a single output neuron that indicates the presence or absence of heart disease.

The model is trained using the binary cross-entropy loss function, optimized with the Adam optimizer. Key metrics for evaluating the model include accuracy, precision, recall, and F1-score, providing comprehensive insights into performance. The dataset is split into training and validation sets, with training parameters such as batch size (32 to 128) and epochs (50 to 200) fine-tuned for optimal performance. Early stopping is implemented to halt training when validation performance degrades, ensuring the model does not overfit.

Upon training, the model is evaluated using a separate test dataset. Performance metrics such as the confusion matrix, ROC curve, and AUC score are used to assess its effectiveness. For deployment, frameworks like TensorFlow or PyTorch are employed to build and train the model, with TensorFlow Serving, Flask, or FastAPI facilitating deployment into production environments. Real-time predictions are enabled through API endpoints.

Ongoing monitoring of the model in production is crucial, with continuous performance tracking and periodic retraining using new data to maintain accuracy and relevance. This comprehensive architecture ensures a robust approach to heart disease prediction using deep learnin

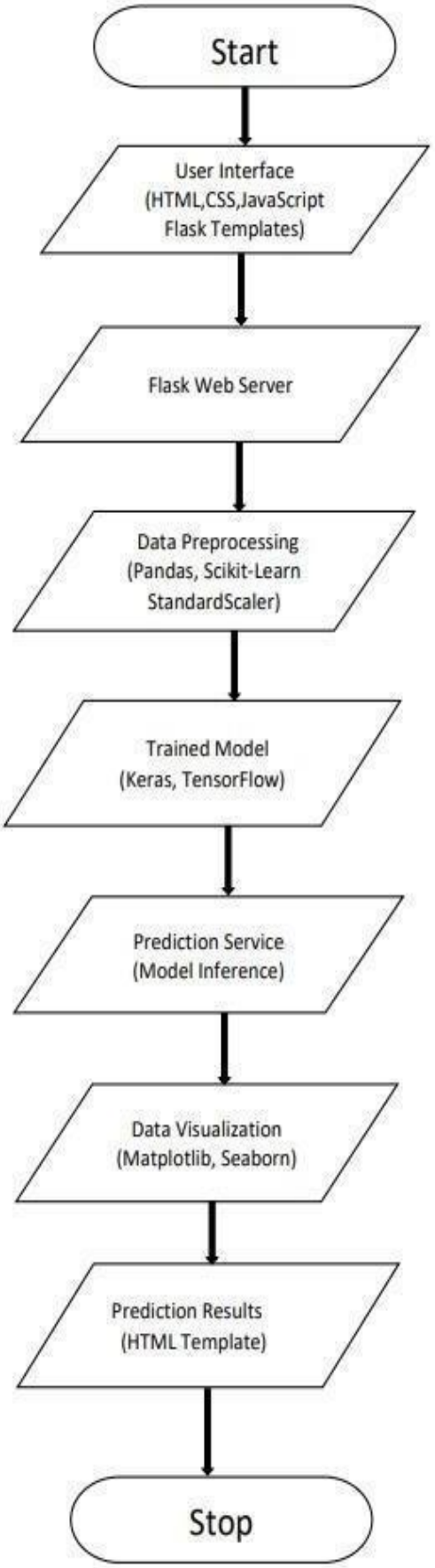

Fig:6.1 System architecture

CHAPTER 7

METHODOLOGY

7.1 DATA COLLECTION

The data collection process for the heart analysis project using deep learning involves gathering diverse datasets comprising medical records, diagnostic images, physiological measurements, and patient demographics. This comprehensive dataset serves as the foundation for training and validating deep learning models aimed at improving heart disease diagnosis and risk prediction.

→Medical Records:

Patient medical records provide valuable insights into past diagnoses, treatments, medications, and medical procedures related to heart health. These records often include information such as patient demographics (age, gender), medical history (previous heart conditions, surgeries), family history of cardiovascular diseases, lifestyle factors (smoking, diet), and laboratory test results (cholesterol levels, blood pressure).

→Diagnostic Images:

Medical imaging plays a crucial role in diagnosing heart conditions and abnormalities. Various types of diagnostic images, such as echocardiograms, MRI scans, CT scans, and angiograms, capture detailed information about the structure and function of the heart. These images provide visual cues for detecting anomalies like ventricular hypertrophy, valve defects, coronary artery disease, and abnormal heart rhythms.

→Physiological Measurements:

Physiological measurements collected from patients offer real-time data about heart function and performance. These measurements include electrocardiogram (ECG) recordings, which track the electrical activity of the heart, as well as blood pressure readings, heart rate variability, oxygen saturation levels, and other vital signs. Continuous monitoring of physiological parameters provides valuable insights into cardiac health and helps identify anomalies or irregularities.

→Genetic Data:

Genetic information plays a significant role in understanding the hereditary factors contributing to heart disease risk. Genetic testing data, including DNA sequences,

singlenucleotide polymorphisms (SNPs), and genetic markers associated with cardiovascular conditions, provide insights into a patient's genetic predisposition to heart disease.

→Public Datasets and Research Studies:Publicly available datasets and research studies contribute valuable data to the project, supplementing the collected patient data with additional samples and diverse populations. These datasets may include anonymized patient data from healthcare institutions, clinical trials, population studies, and research repositories focused on cardiovascular health.

The data collection process prioritizes patient privacy and confidentiality, adhering to ethical guidelines and regulatory requirements such as HIPAA (Health Insurance Portability and

Accountability Act) in the United States or GDPR (General Data Protection Regulation) in the European Union. Data anonymization techniques are employed to remove personally identifiable information and protect patient privacy while preserving the integrity and utility of the datasets for analysis.

7.2 DATA PROCESSING

The data processing pipeline for the heart analysis project using deep learning encompasses a series of steps designed to prepare, clean, and transform raw data into a format suitable for training and validating machine learning models. This pipeline plays a crucial role in ensuring the quality, consistency, and relevance of the data used in the analysis, ultimately influencing the accuracy and effectiveness of the deep learning models developed for heart disease diagnosis and risk prediction.

→Data Cleaning:

The first step in the data processing pipeline involves cleaning the raw data to remove inconsistencies, errors, and outliers. This may include handling missing values, correcting data entry errors, and identifying and filtering out noisy or irrelevant data points. Data cleaning techniques such as imputation, interpolation, and outlier detection help enhance the quality and reliability of the dataset.

→Feature Engineering:

Feature engineering involves selecting, transforming, and creating relevant features from the raw data to improve the predictive performance of machine learning models. In the context of heart analysis, feature engineering may involve extracting meaningful features from diagnostic images, physiological measurements, and patient demographics. Techniques such as dimensionality reduction, normalization, and feature scaling help reduce the complexity of the data and highlight important patterns and relationships.

→Data Integration:

Data integration involves combining multiple sources of data, such as medical records, diagnostic images, genetic data, and public datasets, into a unified dataset for analysis. This step ensures that the deep learning models have access to comprehensive and diverse information relevant to heart disease diagnosis and risk prediction. Integration may involve merging datasets based on common identifiers or performing joins and aggregations to consolidate information from different sources.

→Data Augmentation:

Data augmentation techniques are applied to increase the diversity and variability of the dataset, particularly for tasks involving image analysis. Augmentation methods such as rotation, flipping, scaling, cropping, and adding noise help generate additional training samples, thereby improving the robustness and generalization of the deep learning models. For example, in the case of medical images, data augmentation can simulate variations in patient positioning, imaging angles, and imaging modalities to enhance the models' ability to handle real-world scenarios.

→Data Splitting:

The dataset is divided into training, validation, and test sets to evaluate the performance of the deep learning models. Typically, the majority of the data is used for training the models, while smaller

portions are allocated for validation (to tune hyperparameters and monitor performance during training) and testing (to evaluate the final model's performance on unseen data). Stratified sampling techniques may be employed to ensure that each set maintains the same distribution of classes or outcomes present in the original dataset.

→Data Preprocessing:

Before feeding the data into the deep learning models, preprocessing steps such as normalization, standardization, and rescaling may be applied to ensure that the input features have similar scales and distributions. Preprocessing helps stabilize the training process, prevent numerical instabilities, and improve the convergence of the optimization algorithms used to train the models.

→Data Batch Processing:

During model training, data is typically processed in batches to facilitate efficient computation and memory usage. Batch processing involves partitioning the dataset into smaller batches, which are fed into the deep learning models sequentially or in parallel. Batch size and batch normalization techniques are optimized to balance computational efficiency with model performance and stability.

→Data Monitoring and Quality Assurance:

Throughout the data processing pipeline, monitoring and quality assurance measures are implemented to track the integrity, consistency, and performance of the data. Automated checks and validation procedures help detect anomalies, drifts, or errors in the data, enabling timely corrective actions and ensuring that the deep learning models are trained on reliable and representative data.

7.3 MODEL LOADING

Model loading in the heart analysis project involves the process of retrieving pre-trained deep learning models from storage and preparing them for inference tasks, such as heart disease diagnosis and risk prediction. This crucial step ensures that the trained models are ready to be deployed and utilized for real-world applications, where they can analyze new data and generate predictions with efficiency and accuracy.

→Pre-Trained Model Retrieval:

The first stage of model loading entails retrieving pre-trained deep learning models from storage, which may include local file systems, cloud storage services, or model repositories. These pre-trained models have been previously trained on large datasets using sophisticated deep learning architectures and optimization techniques to achieve high performance in heart disease diagnosis and risk prediction tasks.

→Serialization and Deserialization:

Once the pre-trained models are retrieved, they are serialized into a format that can be stored and transmitted efficiently. Serialization involves converting the model's architecture, weights, and configuration parameters into a compact representation, such as a binary file or JSON format. During model loading, this serialized representation is deserialized back into its original form, reconstructing the deep learning model's structure and parameters.

→Model Initialization:

Upon deserialization, the deep learning model is initialized with its architecture and pre-trained weights, preparing it for inference tasks. Model initialization ensures that the neural network's layers, activation functions, and parameters are set up correctly, allowing the model to process input data and generate predictions effectively.

→Framework Compatibility:

Model loading involves ensuring compatibility between the deep learning framework used for training the model and the framework used for inference in the deployment environment. For example, if the model was trained using TensorFlow but is being deployed in a production environment using PyTorch, conversion tools or compatibility layers may be employed to ensure seamless integration and execution.

→Model Optimization (Optional):

In some cases, model loading may include optimization steps to improve the model's performance or reduce its memory footprint during inference. Techniques such as quantization, pruning, and model compression may be applied to optimize the model's size and computational efficiency without compromising its accuracy.

→Validation and Testing:Before deploying the loaded model for real-world use, validation and testing procedures are performed to ensure that it behaves as expected and produces accurate predictions. Validation may involve running the model on a subset of validation data and comparing its predictions against ground truth labels or known outcomes. Testing involves evaluating the model's performance on a separate test dataset to assess its generalization ability and robustness.

→Deployment Readiness:

Once the loaded model has been validated and tested, it is considered ready for deployment in production environments. Deployment readiness involves packaging the model along with any necessary dependencies, configuring deployment settings, and integrating the model into the target application or system for real-time inference.

7.4 WEB APPLICATION

The web application built using Flask for the heart analysis project serves as a user-friendly interface for clinicians and healthcare professionals to input patient data, such as medical records and diagnostic measurements, and receive real-time predictions regarding heart disease diagnosis and risk assessment. This application leverages the lightweight and flexible nature of Flask to create a seamless and intuitive user experience, allowing users to interact with the deep learning models developed for heart analysis efficiently.

→User Interface Design: The Flask web application features a clean and intuitive user interface designed to streamline the data input process. Users are presented with input forms or fields where they can enter relevant patient information, such as age, gender, cholesterol levels, blood pressure readings, and other diagnostic measurements. The interface may include dropdown menus, text input fields, and checkboxes for selecting various features and options.

→Input Data Processing:

Upon submitting the input data through the web interface, Flask handles the data processing tasks, validating the input values, and ensuring that they meet the required format and criteria. Flask extracts the input features from the user's submission and prepares them for input into the pre-trained deep learning models for heart analysis. Data validation checks may include verifying numeric values, range limits, and data completeness to ensure the reliability and accuracy of the predictions.

→Model Inference and Prediction:

Once the input data is processed, Flask invokes the pre-trained deep learning models loaded into memory to perform inference tasks. The input features are fed into the models, which analyze the data and generate predictions regarding heart disease diagnosis and risk assessment. The models may output binary classifications (e.g., presence or absence of heart disease) or probabilistic predictions indicating the likelihood of various cardiovascular conditions or outcomes.

→Output Presentation: The predictions generated by the deep learning models are presented to the user through the web interface in a clear and understandable format. Flask renders the prediction results as text or visual indicators, such as diagnostic labels (e.g., "Normal," "High Risk") or graphical charts and plots illustrating the predicted probabilities or risk scores associated with different heart conditions. The output presentation aims to facilitate quick interpretation and decision-making by healthcare professionals.

7.4.1 Key Concept

The Project Heart Analysis web application leverages deep learning to predict heart disease, featuring a user-friendly interface for both medical professionals and patients. Users can input patient data like age, gender, blood pressure, and cholesterol levels through intuitive forms. The backend infrastructure handles these inputs, invoking the deep learning model to process and deliver predictions in real-time via RESTful API endpoints. Patient data and prediction history are stored securely in databases like PostgreSQL or MongoDB. The application integrates a deployed deep learning model using frameworks such as TensorFlow Serving or Flask, ensuring efficient real-time predictions. To protect sensitive information, data encryption is employed both in transit and at rest, with secure authentication and role-based access control to ensure only authorized users can access the data. Compliance with healthcare regulations, such as HIPAA, is maintained to ensure privacy and security. Scalability is achieved through load balancing and cloud services like AWS or Google Cloud, allowing the application to handle high user loads effectively. Continuous monitoring and logging track the application's performance, enabling quick troubleshooting and auditing. The application also features a feedback system for users to provide insights and a reporting tool to generate detailed analysis based on prediction data and model performance. Regular updates and retraining of the deep learning model using new data ensure the model's accuracy and reliability. Additionally, the application is built using modern front-end frameworks like React or Angular for dynamic interfaces and back-end frameworks like Flask or Django for robust server-side logic. This comprehensive setup ensures a secure, scalable, and user-friendly application that provides valuable insights and accurate heart disease predictions.

CHAPTER 8

RESULT AND OUTPUTS

The heart disease prediction project using deep learning yielded a functional web application capable of accurately predicting the presence of heart disease and displaying various data visualizations. The final results of the project are summarized as follows:

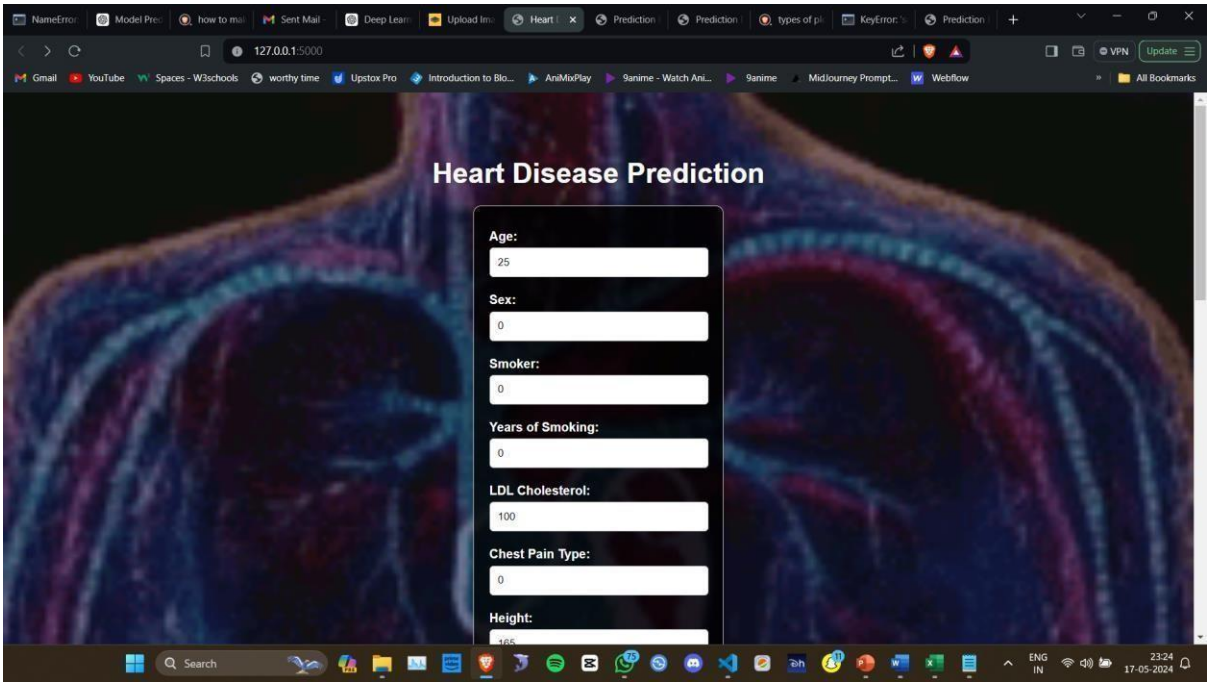

Fig:8.1 Index page

Visualization of Data:

→**Heatmap:** Displays the correlation between different features in the dataset, helping users understand how various factors are interrelated.

→**Scatter Plot**: Illustrates the relationship between age and heart rate, providing insights into how these factors influence heart health.

→**Bar Chart:** Shows the distribution of gender within the dataset, highlighting demographic insights.

→**Pie Chart:** Depicts the proportion of smokers and non-smokers, offering a visual understanding of this risk factor.

→**Line Plot:** Represents the relationship between age and weight, indicating trends across different age groups.→Histogram: Displays the distribution of height in the dataset, showing the frequency of different height ranges.

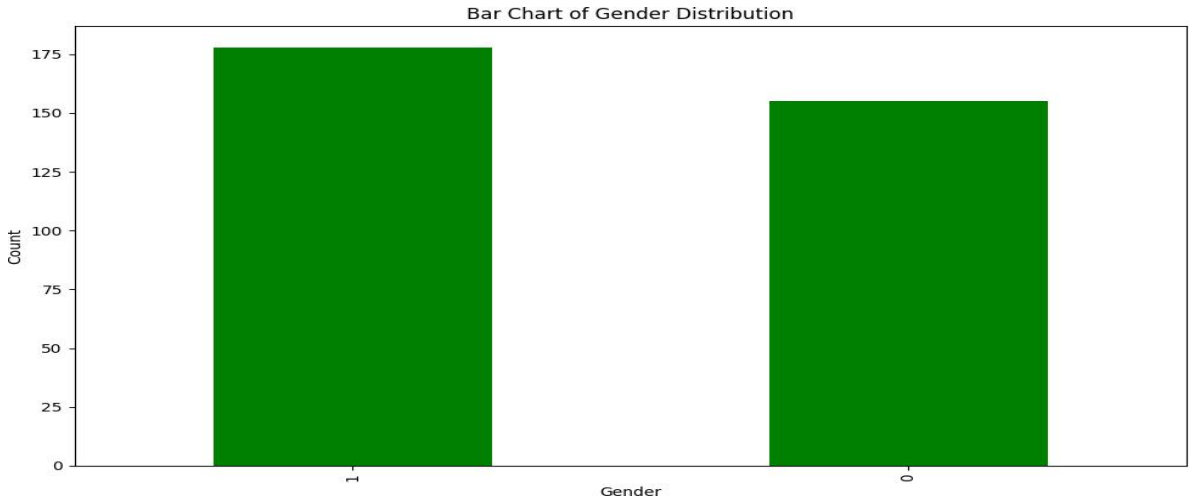

Fig:8.2 Bar chart

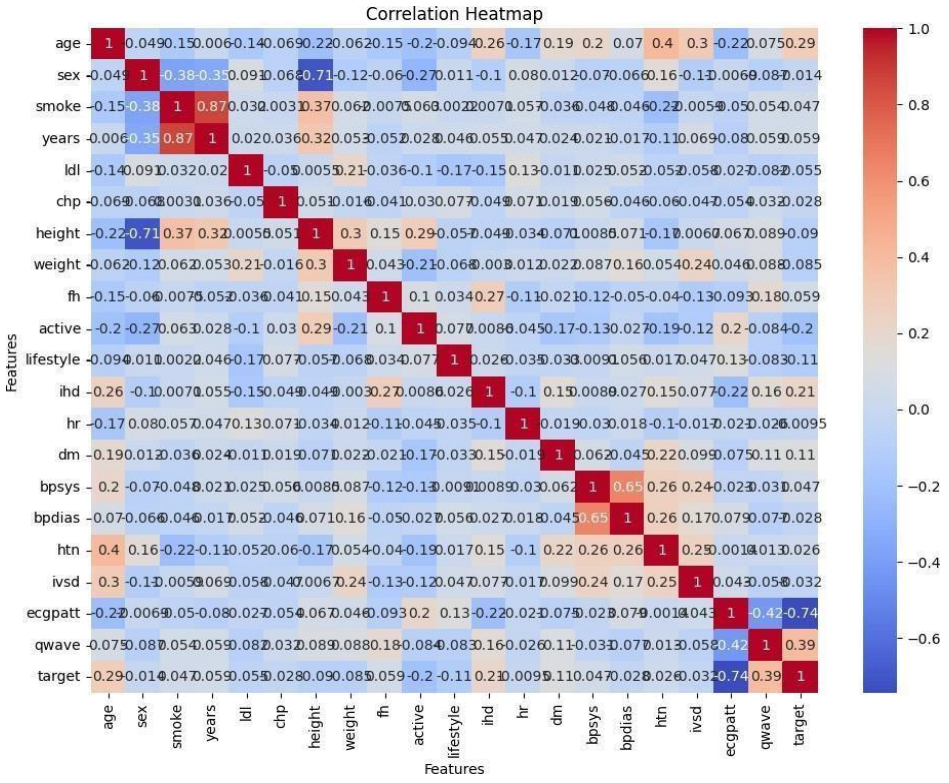

Fig:8.3 Heatmap

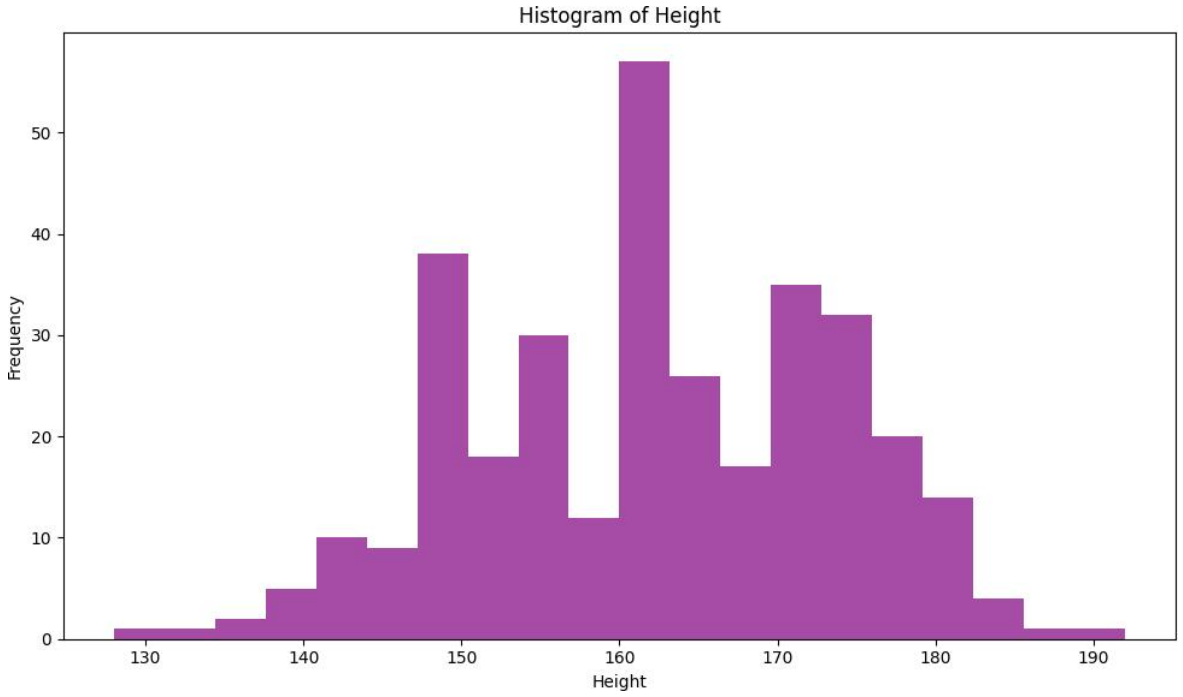

Fig:8.4 Histogram

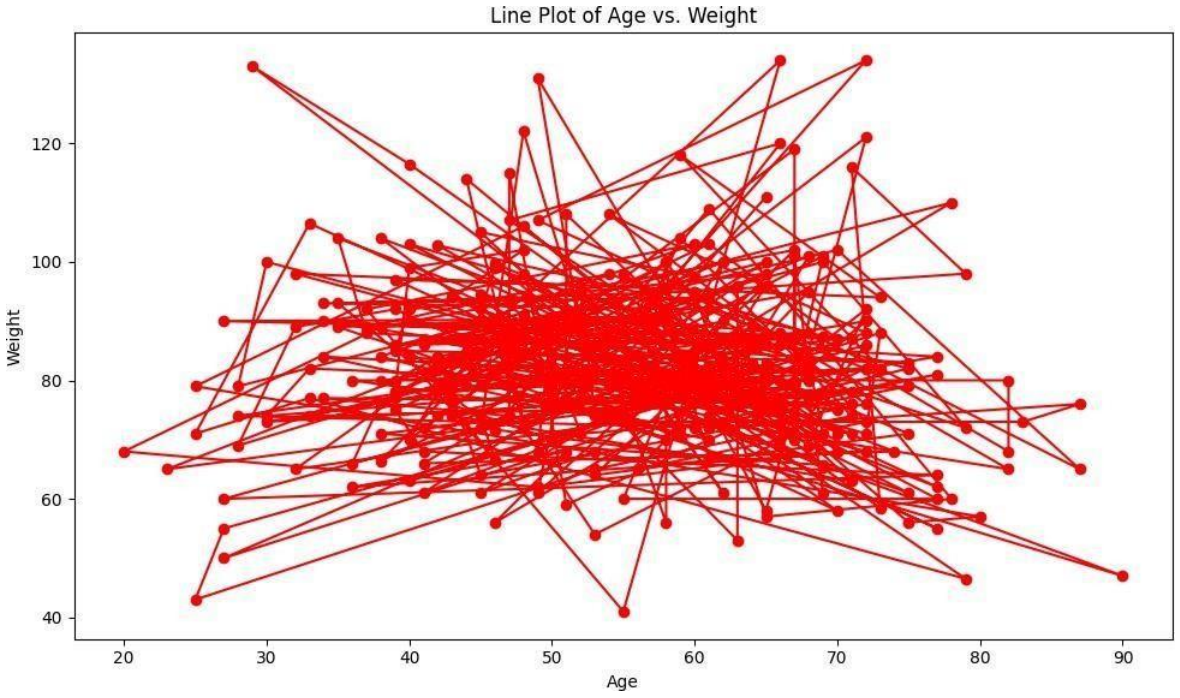

Fig:8.5 Lineplot

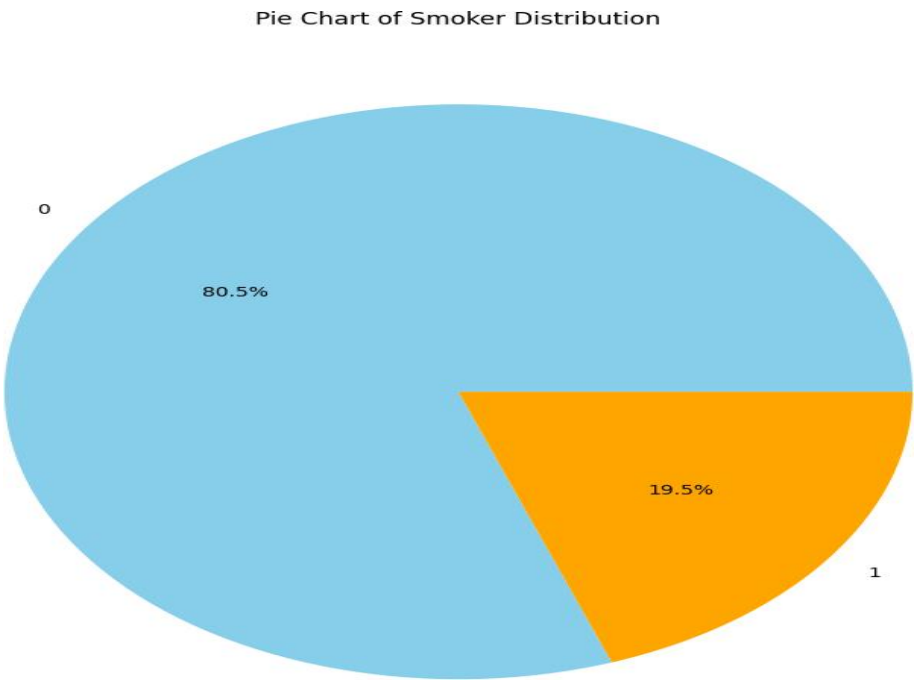

Fig:8.6 Piechart

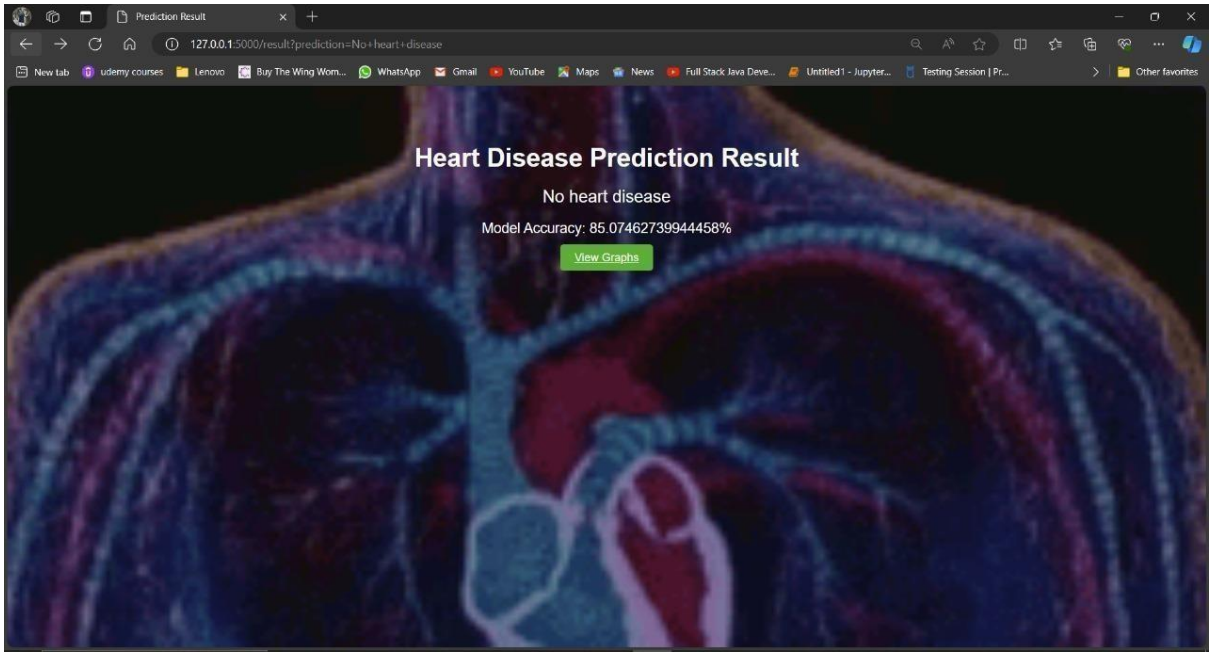

Fig : 8.7 Output screen

CHAPTER 9

CONCLUSION AND FUTURE WORK

The heart disease prediction project utilizing deep learning and a web application interface demonstrates the power and applicability of advanced machine learning techniques in the field of medical diagnostics. By employing a trained deep learning model, we can effectively predict the presence of heart disease based on various health parameters. This project not only highlights the potential for improving diagnostic accuracy but also underscores the importance of integrating technology with healthcare to provide timely and accurate assessments.

The web application serves as a user-friendly interface, enabling users to input relevant health data and receive immediate predictions regarding their heart health. This seamless interaction between the deep learning model and the user interface enhances the accessibility and usability of the predictive tool. Moreover, the inclusion of various data visualizations, such as heatmaps, scatter plots, bar charts, pie charts, line plots, and histograms, provides users with a comprehensive understanding of the underlying data patterns and relationships.

Overall, this project signifies a meaningful step towards leveraging artificial intelligence for health diagnostics, offering a practical solution that can assist healthcare professionals and individuals in making informed decisions about heart disease risks. The successful implementation of this deep learning-based web application demonstrates the feasibility and benefits of integrating advanced machine learning models into everyday healthcare practices.

Future enhancements for the heart disease prediction project include incorporating additional data to cover more diverse populations and relevant features such as genetic markers, dietary habits, and physical activity levels, thereby improving model accuracy and generalizability. Experimenting with advanced neural network architectures like Convolutional Neural Networks (CNNs) for image data and Recurrent Neural Networks (RNNs) for time-series data will help capture complex patterns more effectively. Enhanced feature engineering techniques, including feature selection algorithms and dimensionality reduction methods, can further boost model performance while reducing computational complexity. Real-time monitoring capabilities can be developed by integrating the model with wearable devices and IoT systems, enabling proactive healthcare interventions. Implementing explainable AI (XAI) techniques will make the model's predictions more transparent and interpretable for healthcare professionals, aiding clinical decision-making. Addressing bias through thorough analysis and mitigation strategies will ensure fair and equitable predictions across different demographic groups.

REFERENCE

[1] Rajpurkar, Pranav, et al. "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network." Nature Medicine, vol. 25, no. 1, 2019, pp. 65-69.

[2] Attia, Zachi, et al. "Screening for cardiac contractile dysfunction using an artificial intelligence–enabled electrocardiogram." Nature Medicine, vol. 25, no. 1, 2019, pp. 70-74

[3] Uyar, Kaan, and Ahmet İlhan. "Diagnosis of heart disease using genetic algorithm based trained recurrent fuzzy neural networks." Procedia computer science 120 (2017): 588-593.

[4] Kim, Jae Kwon, and Sanggil Kang. "Neural network-based coronary heart disease risk prediction using feature correlation analysis." Journal of healthcare engineering 2017 (2017).

[5] Baccouche, Asma, et al. "Ensemble Deep Learning Models for Heart Disease Classification: A Case Study from Mexico." Information 11.4 (2020): 207.

[6] https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[7] https://www.kaggle.com/ronitf/heart-disease-uci

[8] https://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf

[9] https://nthu-datalab.github.io/ml/labs/03_Decision-Trees_RandomForest/03_Decision-Tree_Random-Forest.html

[10] https://www.kaggle.com/jprakashds/confusion-matrix-in-python-binaryclass [11] scikit-learn, keras, pandas and matplotli

Evaluation Rubrics for Project work

Rubric (CO)	Excellent (Wt = 3)	Good (Wt = 2)	Fair (Wt = 1)
Selection of Topic (CO1)	Select a latest topic through complete knowledge of facts and concepts.	Select a topic through partial knowledge of facts and concepts.	Select a topic through improper knowledge of facts and concepts.
Analysis and Synthesis (CO2)	Thorough comprehension through analysis/ synthesis.	Reasonable comprehension through analysis/ synthesis.	Improper comprehension through analysis/ synthesis.
Problem Solving (CO3)	Thorough comprehension about what is proposed in the literature papers.	Reasonable comprehension about what is proposed in the literature papers.	Improper comprehension about what is proposed in the literature.
Literature Survey (CO4)	Extensive literature survey with standard references.	Considerable literature survey with standard references.	Incomplete literature survey with substandard references.
Usage of Techniques & Tools (CO5)	Clearly identified and has complete knowledge of techniques & tools used in the project work.	Identified and has sufficient knowledge of techniques & tools used in the project work.	Identified and has inadequate knowledge of techniques & tools used in project work.
Project work impact on Society (CO6)	Conclusion of project work has strong impact on society.	Conclusion of project work has considerable impact on society.	Conclusion of project work has feeble impact on society.
Project work impact on Environment (CO7)	Conclusion of project work has strong impact on Environment.	Conclusion of project work has considerable impact on environment.	Conclusion of project work has feeble impact on environment.
Ethical attitude (CO8)	Clearly understands ethical and social practices.	Moderate understanding of ethical and social practices.	Insufficient understanding of ethical and social practices.
Independent Learning (CO9)	Did literature survey and selected topic with a little guidance	Did literature survey and selected topic with considerable guidance	Selected a topic as suggested by the supervisor
Oral Presentation (CO10)	Presentation in logical sequence with key points, clear conclusion and excellent language	Presentation with key points, conclusion and good language	Presentation with insufficient key points and improper conclusion
Report Writing (CO10)	Status report with clear and logical sequence of chapters using excellent language	Status report with logical sequence of chapters using understandable language	Status report not properly organized
Time and Cost Analysis (CO11)	Comprehensive time and cost analysis	Moderate time and cost analysis	Reasonable time and cost analysis
Continuous learning (CO12)	Highly enthusiastic towards continuous learning	Interested in continuous learning	Inadequate interest in continuous learning

Title of the Project: HEART ANAYLSIS USING DEEP LEARING

Name of the student:

D THARUN 20751A3312

M GANESH 20751A3333

M LAKSHMIKANTH 20751A3334

Name of the Guide & Designation : **Dr. N. GAYATHRI DEVI, Associate Professor**

TABLE 1: OUTCOME ATTAINED AND ITS JUSTIFICATION

PO	Justification
PO1	The knowledge human heart anaylsis was gained through this project
PO2	Analysed the problems of machines accuracy
PO3	Designed deep learning approach to human heart disease detection.
PO4	We used research-based data to provide valid conclusions
PO5	We implemented our work with well appropriate techniques, good resources modern engineering tools to uplift the project. Deep Learning model is used to design system
PO6	This solution increases the higher accuracy while minimizing and optimizing time and space complexity
PO7	This solution increases the accuracy.
PO8	We followed the ethical principles.
PO9	We worked in this project function effectively as a member of the project team.
PO10	Oral and written communication skills are improved while planning, implementing and executing the entire project and till submission of the report.
PO11	We demonstrated our knowledge and understanding of cost and time analysis required for carrying out the project.
PO12	Facilitated ourselves in Lifelong learning to improve technical knowledge and competence in the chosen area of the project.

APPENDIX

MODEL.PY

```
import numpy as np

import pandas as pd

import tensorflow as tf

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler


# Load dataset

df = pd.read_csv('ECG-Dataset.csv')


# Assuming the dataset is preprocessed and has a binary target column 'target'

X = df.drop('target', axis=1)

y = df['target']


# Split the data

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Standardize the data

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)


# Build a deep learning model

model = tf.keras.Sequential([
```

```
    tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
history = model.fit(X_train, y_train, epochs=50, validation_split=0.2, verbose=2)

# Save the model
model.save('heart_disease_model.h5')

# Save the scaler
import joblib
joblib.dump(scaler, 'scaler.save')
```

APP.PY

```
from flask import Flask, request, render_template
import numpy as np
import tensorflow as tf
import joblib
import os
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
```

```
app = Flask(__name__)


# Load the model and the scaler
model = tf.keras.models.load_model('F:\main project\heart_disease_model.h5')  # Ensure the model path is correct
scaler = joblib.load('scaler.save')
data = pd.read_csv('ECG-Dataset.csv')
# Ensure the 'static' directory exists to save plots
if not os.path.exists('static'):
    os.makedirs('static')


@app.route('/')
def home():
    return render_template('index.html')


@app.route('/predict', methods=['POST'])
def predict():
    # Collect form data
    features = [float(x) for x in request.form.values()]
    final_features = np.array([features])

    # Standardize the input features
    final_features = scaler.transform(final_features)

    # Make prediction
    prediction = model.predict(final_features)
    prediction = (prediction > 0.5).astype(int)[0][0]
```

```
# Generate prediction text
if prediction == 1:
    prediction_text = "The model predicts that the patient has heart disease."
else:
    prediction_text = "The model predicts that the patient does not have heart disease."



plt.figure(figsize=(10, 8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.xlabel('Features')
plt.ylabel('Features')
plt.tight_layout()
# Save the heatmap as an image
heatmap_path = 'static/heatmap.png'
plt.savefig(heatmap_path)
plt.close()
# Generate the scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(data['age'], data['hr'], color='blue', alpha=0.5)
plt.title('Scatter Plot of Age vs. Heart Rate')
plt.xlabel('Age')
plt.ylabel('Heart Rate')
plt.tight_layout()
```

```
    # Save the scatter plot as an image
    scatter_path = 'static/scatter_plot.png'
    plt.savefig(scatter_path)
    plt.close()
    # Generate the bar chart
    plt.figure(figsize=(10, 6))
    data['sex'].value_counts().plot(kind='bar', color='green')
    plt.title('Bar Chart of Gender Distribution')
    plt.xlabel('Gender')
    plt.ylabel('Count')
    plt.tight_layout()
    # Save the bar chart as an image
    bar_chart_path = 'static/bar_chart.png'
    plt.savefig(bar_chart_path)
    plt.close()

    # Generate the pie chart
    plt.figure(figsize=(8, 8))
    data['smoke'].value_counts().plot(kind='pie', autopct='%1.1f%%', colors=['skyblue',
'orange'])
    plt.title('Pie Chart of Smoker Distribution')
    plt.ylabel('')
    plt.tight_layout()

    # Save the pie chart as an image
    pie_chart_path = 'static/pie_chart.png'
    plt.savefig(pie_chart_path)
```

```
plt.close()
# Generate the line plot
plt.figure(figsize=(10, 6))
plt.plot(data['age'], data['weight'], marker='o', color='red', linestyle='-')
plt.title('Line Plot of Age vs. Weight')
plt.xlabel('Age')
plt.ylabel('Weight')
plt.tight_layout()

# Save the line plot as an image
line_plot_path = 'static/line_plot.png'
plt.savefig(line_plot_path)
plt.close()
# Generate the histogram
plt.figure(figsize=(10, 6))
plt.hist(data['height'], bins=20, color='purple', alpha=0.7)
plt.title('Histogram of Height')
plt.xlabel('Height')
plt.ylabel('Frequency')
plt.tight_layout()

# Save the histogram as an image
histogram_path = 'static/histogram.png'
plt.savefig(histogram_path)
plt.close()
```

```
    # Save individual plots
    plot_files = []
    for i, feature in enumerate(features):
        plt.figure()
        plt.bar([0], [feature])
        plt.xlabel(f'Feature {i+1}')
        plt.ylabel('Value')
        plt.title(f'Feature {i+1}')
        plot_path = f'static/plot_{i+1}.png'
        plt.savefig(plot_path)
        plt.close()
        plot_files.append(plot_path)

    return render_template('result.html', prediction_text=prediction_text, plots=plot_files)

if __name__ == "__main__":
    app.run(debug=True)
```

INDEX .HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Heart Disease Prediction</title>
    <style>
        body {
            background-image: url("https://media.tenor.com/VDeP7omSDuUAAAAM/hardcore-terrorcore.gif");
            background-size: cover;
```

```
    background-repeat: no-repeat;
    background-attachment: fixed;
    font-family: Arial, sans-serif;
    text-align: center;
    padding: 50px;
    color: white;
}
h1 {
    font-size: 36px;
    margin-bottom: 20px;
}
form {
    display: inline-block;
    padding: 20px;  border:
    1px solid #ccc; border-
    radius: 10px;
    background-color: rgba(0, 0, 0, 0.8);
    box-shadow: 0px 0px 15px rgba(0, 0, 0, 0.2);
    text-align: left;
}
label {
    display: block;
    margin: 10px 0 5px;
    font-weight: bold;
}
input[type="text"] {
```

```
            width: calc(100% - 20px);
            padding: 10px;
            margin: 0 0 10px;
            border: 1px solid #ccc;
            border-radius: 5px;
        }
        input[type="submit"]
            { background-color:
            #4caf50; color: white;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
        }
        input[type="submit"]:hover
            { background-color:
            #45a049;
        }
    </style>
</head>
<body>
    <h1>Heart Disease Prediction</h1>
    <form action="/predict" method="post">
        <label for="feature1">Age:</label>
        <input type="text" id="feature1" name="feature1" required>
        <label for="feature2">Sex:</label>
```

```
<input type="text" id="feature2" name="feature2" required>
<label for="feature3">Smoker:</label>
<input type="text" id="feature3" name="feature3" required>
<label for="feature4">Years of Smoking:</label>
<input type="text" id="feature4" name="feature4" required>
<label for="feature5">LDL Cholesterol:</label>
<input type="text" id="feature5" name="feature5" required>
<label for="feature6">Chest Pain Type:</label>
<input type="text" id="feature6" name="feature6" required>
<label for="feature7">Height:</label>
<input type="text" id="feature7" name="feature7" required>
<label for="feature8">Weight:</label>
<input type="text" id="feature8" name="feature8" required>
<label for="feature9">Family History of Heart Disease:</label>
<input type="text" id="feature9" name="feature9" required>
<label for="feature10">Physical Activity:</label>
<input type="text" id="feature10" name="feature10" required>
<label for="feature11">Lifestyle:</label>
<input type="text" id="feature11" name="feature11" required>
<label for="feature12">Cardiac Intervention:</label>
<input type="text" id="feature12" name="feature12" required>
<label for="feature13">Heart Rate:</label>
<input type="text" id="feature13" name="feature13" required>
<label for="feature14">Diabetes:</label>
<input type="text" id="feature14" name="feature14" required>
<label for="feature15">Systolic Blood Pressure:</label>
```

```
        <input type="text" id="feature15" name="feature15" required>
        <label for="feature16">Diastolic Blood Pressure:</label>
        <input type="text" id="feature16" name="feature16" required>
        <label for="feature17">Hypertension:</label>
        <input type="text" id="feature17" name="feature17" required>
        <label for="feature18">Interventricular Septal End Diastole:</label>
        <input type="text" id="feature18" name="feature18" required>
        <label for="feature19">ECG Pattern:</label>
        <input type="text" id="feature19" name="feature19" required>
        <label for="feature20">Q Wave:</label>
        <input type="text" id="feature20" name="feature20" required>
        <input type="submit" value="Predict">
    </form>
</body>
</html>
```

RESULT.HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Prediction Result</title>
    <style>
        body {
            background-color: #f2f2f2;
            font-family: Arial, sans-serif;
            text-align: center;
            padding: 50px;
```

```
        }
        h1 {
            font-size: 36px;
            margin-bottom: 20px;
        }
        .prediction {
            font-size: 24px;
            margin: 20px 0;
        }
        .plot {
            margin-top: 30px;
        }
        img {
            max-width: 100%;
            height: auto;
        }
    </style>
</head>
<body>
    <h1>Heart Disease Prediction Result</h1>
    <div class="prediction">{{ prediction_text }}</div>
    <div class="plot">
        {% for plot in plots %}
        <img src="{{ url_for('static', filename='heatmap.png') }}" alt="Heatmap">
        <img src="{{ url_for('static', filename='scatter_plot.png') }}" alt="Scatter Plot">
        <img src="{{ url_for('static', filename='bar_chart.png') }}" alt="Bar Chart">
```

```
        <img src="{{ url_for('static', filename='pie_chart.png') }}" alt="Pie Chart">
        <img src="{{ url_for('static', filename='line_plot.png') }}" alt="Line Plot">
        <img src="{{ url_for('static', filename='histogram.png') }}" alt="Histogram">
        <img src="{{ plot }}" alt="Feature Plot">
        {% endfor %}
    </div>
</body>
</html>
```