

**Smart EEG Analyzer**

By

Osama Bassam Yousef Ali

**Supervised By:**

**This Project (Graduation Project I) is Submitted in Partial Fulfillment of the**

**Requirements for bachelor's degree in data science**

**Faculty of Artificial Intelligence**

**Al-Balqa Applied University-Al-Salt- Jordan**

**Fall 2024/2025**

**TABLE OF CONTENTS**

[1 DEDICATIONS iii](#_Toc198159005)

[2 ACKNOWLEDGEMENTS iv](#_Toc198159006)

[3 ABSTRACT v](#_Toc198159007)

[LIST OF TABLES vi](#_Toc198159008)

[LIST OF FIGURES vii](#_Toc198159009)

[4 ABBREVIATIONS viii](#_Toc198159010)

[1 CHAPTER 1: INTRODUCTION 1](#_Toc198159011)

[1.1 Overview: 2](#_Toc198159012)

[1.2 Problem Statement: 2](#_Toc198159013)

[1.3 Objectives of the Project: 3](#_Toc198159014)

[1.4 Scope of the Project: 4](#_Toc198159015)

[1.4.1 Inclusion: 4](#_Toc198159016)

[1.4.2 Exclusion 4](#_Toc198159017)

[1.5 Outline of the Project Report 5](#_Toc198159018)

[2 CHAPTER 2: REVIEW OF RELATED LITERATURE 7](#_Toc198159019)

[2.1 Introduction 7](#_Toc198159020)

[2.2 Related Works 7](#_Toc198159021)

[2.2.1 Deep Learning for Electroencephalogram (EEG) Classification Tasks 7](#_Toc198159022)

[2.2.2 The Temple University Hospital EEG Data Corpus 7](#_Toc198159023)

[2.2.3 Real-time Inference and Detection of Disruptive EEG 8](#_Toc198159024)

[2.2.4 Automated Recognition of Epilepsy from EEG Signals Using Machine Learning: 8](#_Toc198159025)

[2.2.5 Building Open Source LLM based Chatbots using Llama Index 8](#_Toc198159026)

[2.2.6 Integrate machine learning in Android apps 8](#_Toc198159027)

[2.3 Related Works Summary 9](#_Toc198159028)

[3 CHAPTER 3: ANALYSIS AND DESIGN 11](#_Toc198159029)

[3.1 Project Requirements: 11](#_Toc198159030)

[3.1.1 User Requirements 11](#_Toc198159031)

[3.1.2 System Requirements: 12](#_Toc198159032)

[3.1.3 Performance Requirements Set benchmarks for system behavior: 13](#_Toc198159033)

[3.1.4 Chatbot System Requirements 14](#_Toc198159034)

[3.2 System Architecture and Module Design 15](#_Toc198159035)

[3.2.1 Data Preprocessing Module 16](#_Toc198159036)

[3.2.2 Feature Extraction Module: 17](#_Toc198159037)

[3.2.3 Classification Module 17](#_Toc198159038)

[3.2.4 Chatbot Module 18](#_Toc198159039)

[3.3 Data Structure and Analysis 19](#_Toc198159040)

[3.3.1 Data Structure Description 19](#_Toc198159041)

[3.3.2 Database Design 21](#_Toc198159042)

[3.4 Project Methodology/Techniques 22](#_Toc198159043)

[4 Chapter Four Project Design & Implementation 23](#_Toc198159044)

[4.1 Project Design 23](#_Toc198159045)

[4.1.1 Graphical User Interface (GUI) 23](#_Toc198159046)

[4.1.2 Data Workflow 23](#_Toc198159047)

[4.2 Project Implementation 24](#_Toc198159048)

[4.2.1 Data Preparation 24](#_Toc198159049)

[4.2.2 Machine Learning Models 30](#_Toc198159050)

[4.3 Visualizations 34](#_Toc198159051)

[4.3.1 Chatbot Assistance 38](#_Toc198159052)

[5 Chapter Five Evaluation and Experimental Results 40](#_Toc198159053)

[6 Chapter Six 41](#_Toc198159054)

[6.1 Conclusion 41](#_Toc198159055)

[6.2 Future Work 41](#_Toc198159056)

[7 References 43](#_Toc198159057)

# DEDICATIONS

This Project is dedicated to those who have played a significant role in my academic journey, offering their support, encouragement, and inspiration.

*To my family, for their unwavering love, support, and encouragement throughout this journey. Your belief in me has been my greatest strength. To my parents, for their endless sacrifices and for instilling in me the values of hard work and perseverance. To my siblings, for always being there with words of encouragement and a helping hand.*

*To my doctors, whose expertise and care made this work possible. Your dedication to my health and well-being has been invaluable. Thank you for your patience, compassion, and for helping me overcome challenges along the way.*

*To my friends, for their understanding and for providing me with the much-needed breaks and moments of laughter, which kept me sane throughout this process*

# ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to the director of this project, Dr. **Wafaa Zaal Al Maitah,** for the continuous support of this project, for her patience, motivation, and immense knowledge. Her guidance helped us in all the time of research and writing of this project.

My sincere thanks also go to our fellow colleagues, professors, and anyone who provided assistance and support throughout this journey.

I would also like to extend my deepest gratitude to my family for their unwavering support and encouragement. To my friends, for their understanding and for providing me with the much-needed breaks and moments of laughter.

Lastly, I would like to thank the faculty and staff of the Faculty of Artificial Intelligence at Al-Balqa Applied University for providing an excellent academic environment and resources that facilitated this project.

# ABSTRACT

This project focuses on developing an automated system for classifying Electroencephalogram (EEG) signals as normal or abnormal, leveraging advanced machine learning techniques. EEG, a non-invasive method for monitoring brain activity, generates large volumes of data that require efficient and accurate analysis to support clinical decision-making. The project utilizes the Temple University Abnormal EEG Dataset (TUAB) to train and evaluate models like Convolutional Neural Networks (CNNs) and Multilayer Perceptron’s (MLPs), achieving high accuracy in EEG classification tasks. Additionally, an anomaly detection module based on autoencoders, and statistical methods identifies unusual patterns in the EEG data, enhancing diagnostic insights. A chatbot system powered by natural language processing (NLP) techniques provides an intuitive interface for users, offering classification results and explanations for detected anomalies. Preprocessing techniques, including artifact removal, filtering, and segmentation, ensure data quality for machine learning pipelines. The structured organization of EEG data in the European Data Format (EDF) and a NoSQL database further supports efficient data handling and integration. This comprehensive approach lays the groundwork for advancing automated EEG analysis, bridging the gap between clinical needs and technological capabilities.

# LIST OF TABLES

[Table 2‑1 Related Work Summery 10](#_Toc198228072)

[Table 3‑1 Summary of User Requirements 12](#_Toc198228073)

[Table 3‑2 System Requirements 13](#_Toc198228074)

[Table 3‑3 EDF File Structure 20](#_Toc198228075)

[Table 5‑1 comparative summary of all machine learning models 47](#_Toc198228076)

# LIST OF FIGURES

[1‑1 Figure: EEG Brain Wave Patterns 1](#_Toc198228094)

[Figure 3‑1 Chatbot Goals and Functionalities 14](#_Toc198228095)

[Figure 3‑2 System Architecture Diagram 15](#_Toc198228096)

[Figure 3‑3 EEG Data Preprocessing Flowchart 16](#_Toc198228097)

[Figure 3‑4 EEG Processing Pipeline 18](#_Toc198228098)

[Figure 4‑1 Home Page Interface 24](#_Toc198228099)

[Figure 4‑2 EEG File Upload Interface 25](#_Toc198228100)

[Figure 4‑3 "How It Works" Overview 26](#_Toc198228101)

[Figure 4‑4 Step-by-Step EEG Processing Workflow 26](#_Toc198228102)

[Figure 4‑5 Contact Us Page 27](#_Toc198228103)

[Figure 4‑6 Dataset Loading and Labeling 29](#_Toc198228104)

[Figure 4‑7 Channel Selection 29](#_Toc198228105)

[Figure 4‑8 Muscle Artifact Removal 30](#_Toc198228106)

[Figure 4‑9 Signal Preprocessing 31](#_Toc198228107)

[Figure 4‑10 Windowing 31](#_Toc198228108)

[Figure 4‑11 Data Balancing 32](#_Toc198228109)

[Figure 4‑12 Feature Selection Strategy 33](#_Toc198228110)

[Figure 4‑13 Handcrafted Feature Categories 34](#_Toc198228111)

[Figure 4‑14 Chatbot Integration 35](#_Toc198228112)

[Figure 4‑15Final Model: Custom MLP Preprocessed + artifact-cleaned signals 38](#_Toc198228113)

[Figure 4‑16 select from 21 standard EEG channels 39](#_Toc198228114)

[Figure 4‑17 Abnormal Window Navigation 40](#_Toc198228115)

[Figure 4‑18 pie chart visualizes the distribution of predicted labels 41](#_Toc198228116)

[Figure 4‑19 EEG Plot with Abnormal Segment Focus 42](#_Toc198228117)

[Figure 4‑20 Chatbot Assistance 43](#_Toc198228118)

[Figure 4‑21 initializes a secure connection to Cloudflare R2 44](#_Toc198228119)

[Figure 4‑22 Generates a unique MD5 hash 45](#_Toc198228120)

[Figure 4‑23 Verifies whether the file already exists in the cloud before attempting upload 45](#_Toc198228121)

[Figure 4‑24 Uploads large EEG files in chunks 45](#_Toc198228122)

[Figure 4‑25 Downloads a file in small chunks and updates progress dynamically 46](#_Toc198228123)

[Figure 4‑26 Stores the current processing state of the EEG file in a centralized session 46](#_Toc198228124)

[Figure 4‑27 Keeps the front end informed of backend progress in real-time 46](#_Toc198228125)

# ABBREVIATIONS

**EEG**: Electroencephalography

* A method for recording brain wave activity.

**EDF**: European Data Format

* A standardized file format used for storing EEG data.

**TUAB**: Temple University Abnormal EEG Corpus

* A subset of the Temple University EEG dataset used for normal vs. abnormal classification.

**MLP**: Multilayer Perceptron

* A type of neural network used for EEG classification.

**STFT**: Short-Time Fourier Transform

* A method for analyzing time-frequency patterns in EEG signals.

**LLM**: Large Language Model

* A language model (e.g., Meta Llama) used in the chatbot.

**PSD**: Power Spectral Density

* A frequency-domain feature extracted from EEG signals.

**SVM**: Support Vector Machine

* A machine learning algorithm used in classification tasks.

# CHAPTER 1: INTRODUCTION

The rapid advancement in machine learning and artificial intelligence has revolutionized the field of medical diagnostics, enabling systems capable of analyzing complex data autonomously. One such application is the classification of electroencephalogram (EEG) signals, as shown in **Figure 1-1**, which presents the fundamental waveforms recorded during EEG analysis. These waveforms—beta, alpha, theta, and delta—are essential for understanding brain activity and diagnosing neurological disorders.

This advancement addresses the challenges of manual EEG interpretation, such as time consumption, variability, and the potential for errors. By integrating state-of-the-art machine learning models, this project seeks to automate EEG classification, improve diagnostic accuracy, and reduce the workload of clinicians.

Figure 1-1: EEG Brain Wave Patterns


‑ Figure: EEG Brain Wave Patterns

*The figure illustrates the main types of EEG waves (beta, alpha, theta, and delta) commonly analyzed for diagnosing neurological conditions. Each wave corresponds to distinct brain states, forming the basis for automated classification systems.*

## Overview:

Electroencephalography (EEG) is a widely used, noninvasive tool for recording the brain's electrical activity. Despite advancements in technologies such as MRI, EEG remains popular due to its low cost and practicality for diagnosing conditions like epilepsy and sleep disorders. The importance of EEG lies in its ability to provide critical insights into brain activity. However, manual interpretation of EEG recordings by certified neurologists is time-consuming, subject to variability, and prone to errors, creating delays in diagnosis and treatment. Emerging machine learning technologies offer a promising solution to automate EEG analysis, reducing the burden on clinicians and enhancing diagnostic accuracy. Recent developments in deep learning and big data have provided opportunities to improve EEG classification systems, addressing the challenges of subjective analysis and resource constraints.

## Problem Statement:

The manual analysis of EEG data is not only time-consuming and subjective but also requires the use of a lot of resources, resulting in delayed diagnosis and yet even misdiagnoses. In particular, deciding upon the status of an EEG as ‘normal’ or ‘abnormal’ is a significant step in stating whether certain patients should be considered for further workup or not. The variability among human readers, especially for subtle abnormalities, reduces the reproducibility of classic EEG recordings. These factors are compounded by using machine learning algorithms because they will be able to automatically detect and classify abnormal EEG patterns, thereby improving clinical workflow and reducing false alarms. This issue is very important in medicine because it assists in providing quality delivery of care to patients since a diagnosis should be accurate and done within the right time frame.

## Objectives of the Project:

This project aims to design an automatic classification system for abnormal and normal EEG recordings based on machine learning methods to help doctors diagnose disorders concerning any form of disorders. Specific objectives are as follows:

* Develop a Robust Classification System:

Construct a machine learning model with a remarkable ability to identify abnormal EEG criteria with high accuracy and reliable results.

* Add anomaly detection:

Improved the developed system with anomaly detection function to help detect unusual EEG patterns for more detailed analysis.

* Use other advanced algorithms:

Include advanced features such as CNN and other models for deep learning on both feature extraction and classification.

* Streamline clinician's process:

Accomplish the aim of saving time and effort in interpreting the EEG while delivering useful information and minimizing chances of misdiagnosis.

* Develop a User-Friendly Web Interface:

Develop a website that supports data uploading, result interpretation, and communication with users through a chatbot for help.

* Utilize a High-Quality Dataset:

Fit the system onto TUAB database to test its validity and accuracy on numerous EEG variables.

* Create an Android Application:

Develop a mobile app so that ultrasound interpretations can be easily accessed.

## Scope of the Project:

The project focuses on leveraging machine learning to automate the classification of EEG recordings, enhancing clinical workflows and diagnostic accuracy. The scope defines the boundaries of what is included and excluded in the system's development and implementation.

### Inclusion:

* Normal/Abnormal EEG Classification
  + Analysis and classification of adult EEG recordings to distinguish between normal and abnormal patterns.
* Machine Learning Algorithms
  + Implementation and comparison of various algorithms, including:
    - Convolutional Neural Networks (CNNs)
    - Multilayer Perceptron's (MLPs)
    - Hidden Markov Models (HMMs)
* Anomaly Detection
  + Integration of anomaly detection techniques to flag unusual patterns for further investigation.
* Multiplatform Accessibility
  + Development of a web-based interface for easy access.
  + Creation of an Android app for remote data upload and result retrieval.
* Dataset Utilization
  + Use of the TUAB dataset, ensuring a balanced and diverse sample for training and evaluation.

### Exclusion

* **Specific Abnormality Detection**
  + The project does not focus on detecting specific abnormalities, such as seizures or other distinct neurological conditions.
* Pediatric EEG Analysis
  + The scope is limited to adult EEG recordings; pediatric EEGs are not included.
* Real-Time Monitoring
  + The system does not support real-time EEG data analysis; it is designed for post-recording data processing.
* Hardware Development
  + No custom hardware or EEG recording devices will be designed or built as part of this project.

**Note:**

The exclusions outlined above represent features and functionalities that are beyond the current scope of this project. However, these aspects can serve as potential areas for enhancement in future iterations. For instance, the system could be extended to detect specific abnormalities like seizures, analyze pediatric EEG data, support real-time monitoring, or even integrate with custom-designed hardware. These additions would further improve the system's versatility and applicability in clinical settings.

## Outline of the Project Report

The remainder of the project documentation is organized as follows:

* Chapter 2: Related Work This chapter reviews existing literature and technologies relevant to the project topic, focusing on EEG classification and anomaly detection. It provides an overview of prior research and developments in these areas, highlighting advancements and identifying gaps that the current project addresses.
* Chapter 3: Analysis and Design
  + 3.1 Project Requirements This section outlines the specific requirements for the project, including functional and non-functional requirements, and any constraints or considerations.
  + 3.2 Module Design This section describes the system’s architecture and components, including preprocessing, feature extraction, classification, and anomaly detection. It provides detailed explanations of how these modules interact and contribute to the overall system.
  + 3.3 Data Analysis and Design This section focuses on the structure and organization of the EEG data used in the project. It explains the EDF format, database design, and how data is prepared for analysis.
  + 3.4 Methodology/Techniques This section describes the methodologies and techniques employed in the project, including machine learning models (e.g., CNNs, MLPs), anomaly detection algorithms, and natural language processing methods for chatbot integration.
* Future Chapters (Project 2) Future work will include the implementation, testing, evaluation, and conclusion phases, which will be documented in subsequent chapters.

# CHAPTER 2: REVIEW OF RELATED LITERATURE

## Introduction

In this chapter of the dissertation EEG signal classification and abnormalities detection related works are reviewed. The review also covers some of the methods that have been developed throughout the years to create models for detecting those abnormalities. We are trying to answer the question regarding strength and depth of the field, its shortcomings and the areas that require further investigation. This review most importantly helps the reader understand the stipulations of previous works and what can be improved in future ones for the better understanding and solving of the EEG problems. All of these considerations serve as a base for our purposes as we intend to continue with the developments made in the automatic classification of EEG.

## Related Works

### **Deep Learning** for Electroencephalogram (EEG) Classification Tasks

Craik et al. (2019): Reviewed deep learning methods, emphasizing Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for EEG classification tasks. The study highlighted applications like brain-computer interfaces, seizure detection, and cognitive state monitoring, while noting challenges such as data scarcity, subject variability, and computational costs. Future directions proposed subject-independent models and leveraging transfer learning to enhance generalization across datasets.

### The Temple University Hospital EEG Data Corpus

Obeid and Picone (2016): Introduced the Temple University Hospital (TUH) EEG Data Corpus, one of the largest publicly available EEG datasets, comprising over 16,000 recordings. The corpus includes metadata like demographic and clinical details, stored in EDF format for compatibility with preprocessing tools. The TUAB subset within the corpus focuses on normal vs. abnormal EEG classification, addressing class imbalance and supporting effective model training. Applications include abnormal EEG classification, seizure detection, and general EEG analysis. Limitations include class imbalance in the full corpus and the necessity for preprocessing steps like artifact removal and filtering to ensure data quality.

### Real-time Inference and Detection of Disruptive EEG

This study focuses on the use of dynamic learning methods for real-time seizure detection by analyzing disruptions in EEG networks. The method achieved high accuracy but is limited by its computational complexity and need for further validation in clinical settings ([3]).

### Automated Recognition of Epilepsy from EEG Signals Using Machine Learning:

This research explores machine learning for epilepsy detection, leveraging temporal features from EEG signals. While it improves detection accuracy, it requires large datasets and significant computational resources for training and validation ([4]).

### Building Open Source LLM based Chatbots using Llama Index

Recent advancements in Large Language Models (LLMs) have significantly influenced the development of intelligent chatbots capable of retrieving and processing vast amounts of data from external knowledge sources. The introduction of LlamaIndex (formerly GPT Index) has made it easier to build such systems by facilitating the connection between LLMs and external data repositories. This retrieval-augmented generation (RAG) approach allows chatbots to provide more contextually relevant responses by accessing external data sources dynamically. For instance, Poatek demonstrated the use of LlamaIndex to develop open-source chatbots that can handle complex domain-specific queries by integrating various knowledge databases such as text files, APIs, and websites [5].

### Integrate machine learning in Android apps

Integrating machine learning (ML) into Android apps has become a popular method to enhance functionality and user experience. A key example is the guide by AppsDevPro, which focuses on using TensorFlow Lite to integrate ML into Android applications. This approach optimizes models for mobile devices, improving performance and reducing resource consumption [6]

## Related Works Summary

|  |  |  |  |
| --- | --- | --- | --- |
| **Reference** | **Approach** | **Features** | **Limitations** |
| [1] | Review of deep learning models for EEG classification | Covers CNN and RNN architectures and applications in BCI and seizure detection. | Limited by data scarcity and computational costs. |
| [2] | Large dataset for EEG research (TUH EEG Corpus). | Comprehensive, diverse, and widely used for benchmarking EEG classification models. Includes TUAB subset balanced for abnormal classification. | Preprocessing required for raw data. |
| [3] | Real-time Inference and Detection of Disruptive EEG Networks for Epileptic Seizures | Dynamic learning model for seizure prediction based on EEG network disruptions. | Requires further validation for clinical applications and scalability. |
| [4] | Automated Recognition of Epilepsy from EEG Signals Using Machine Learning | Machine learning model analyzing temporal EEG data for epilepsy detection. | Needs large datasets and substantial computational power for training. |
| [5] | Describes the construction of open-source LLM-based chatbots using LlamaIndex for retrieval-augmented generation (RAG). | Flexible, adaptable to various domains including healthcare, finance, and education. | Limited by computational resources for large-scale real-time applications. |
| [6] | Integration of machine learning in Android | - Detailed guide on ML integration in Android apps.  - Examples with TensorFlow Lite. | - Limited to TensorFlow Lite integration. |

Table ‑ Related Work Summery

# CHAPTER 3: ANALYSIS AND DESIGN

This chapter provides a comprehensive roadmap for the development of our EEG classification system. It begins by exploring user requirements and technical specifications, which are critical for ensuring the system's usability and functionality. The chapter further delves into the structure of the data used in the project, highlighting the EDF format and its relevance to EEG analysis. Additionally, it describes the database design, emphasizing the organization and retrieval of EEG files and classification results for future enhancements. By detailing the system’s components and architecture, this chapter lays the groundwork for understanding how the system processes, analyzes, and classifies EEG data effectively.

## Project Requirements:

### **User Requirements**

User requirements are essential to ensure the system meets the needs of its end-users, primarily clinicians and neurologists. This section outlines the functional and non-functional requirements necessary to provide accurate, efficient, and user-friendly EEG classification and anomaly detection.

**Functional Requirements**

* Data Upload:
  + Users can upload EEG data in EDF format for analysis.
  + Allow batch uploads for efficiency.
* EEG Classification:
  + The system should classify EEG recordings as normal or abnormal with high accuracy.
* Results Visualization:
  + Provide clear, interpretable visualizations of classification results, including probabilities or confidence scores.
* Anomaly Alerts:
  + Automatically flag unusual patterns or anomalies for further review.
* Chatbot Assistance:
  + Integrate a chatbot for answering user queries related to the classification results and system functionality.

**Non-Functional Requirements**

* Accuracy:
  + Achieve classification accuracy comparable to experienced neurologists.
* Speed:
  + Ensure results are generated within seconds of data upload.
* Usability:
  + Provide an intuitive and user-friendly interface suitable for clinicians with minimal training.

|  |  |
| --- | --- |
| **Requirement Type** | **Description** |
| Functional | Upload EEG data in EDF format. |
| Functional | Classify EEG as normal or abnormal. |
| Functional | Provide anomaly alerts. |
| Functional | Integrate chatbot for user assistance. |
| Non-Functional | Ensure accuracy comparable to neurologists. |
| Non-Functional | Results generated within seconds. |

Table ‑ Summary of User Requirements

### System Requirements:

System requirements define the technical and operational aspects of the project, ensuring the system functions effectively and efficiently. This section outlines the hardware, software, and performance specifications necessary for implementing the EEG classification and anomaly detection system.

1. Hardware Requirements List the essential hardware needed to develop and run the system:

* Processing Power: Minimum quad-core processor (e.g., Intel i5 or higher).
* Memory: At least 8GB RAM (16GB recommended for training models).
* Storage: 500GB SSD or higher for handling large EEG datasets.
* GPU: NVIDIA GPU with CUDA support (e.g., GTX 1060 or higher) for deep learning model training.

2. Software Requirements List of the required software tools and libraries:

* Operating System: Windows 10/11, macOS, or Linux.
* Programming Language: Python 3.x.
* Libraries:
  + TensorFlow or PyTorch for deep learning.
  + NumPy, SciPy, and pandas for data preprocessing.
  + MNE for EEG-specific data handling.
* IDE: Jupyter Notebook, PyCharm, or Visual Studio Code.

### Performance Requirements Set benchmarks for system behavior:

* Accuracy: Achieve classification accuracy above 90%.
* Processing Time: Generate results within 10 seconds for a single EEG file.
* Scalability: Support batch processing of up to 50 files simultaneously.

|  |  |
| --- | --- |
| **Requirement Type** | **Description** |
| Hardware | Quad-core processor, 16GB RAM, NVIDIA GPU. |
| Software | Python, TensorFlow, MNE, Jupyter Notebook |
| Performance | >90% accuracy, results within 10 seconds, batch processing. |

Table ‑ System Requirements

### Chatbot System Requirements

* LlamaIndex: Utilized for building and indexing the chatbot.
* Pinecone: Serves as the vector database for storing and retrieving data.
* Groq: API calls to Meta's Large Language Models (LLMs) for chatbot interactions.
* Hosting Framework: Flask or FastAPI for backend API handling.
* Frontend Tools: React.js or Flutter for chatbot integration in the website/mobile app

**A diagram of a chatbot

Description automatically generated**

Figure ‑ Chatbot Goals and Functionalities

## System Architecture and Module Design

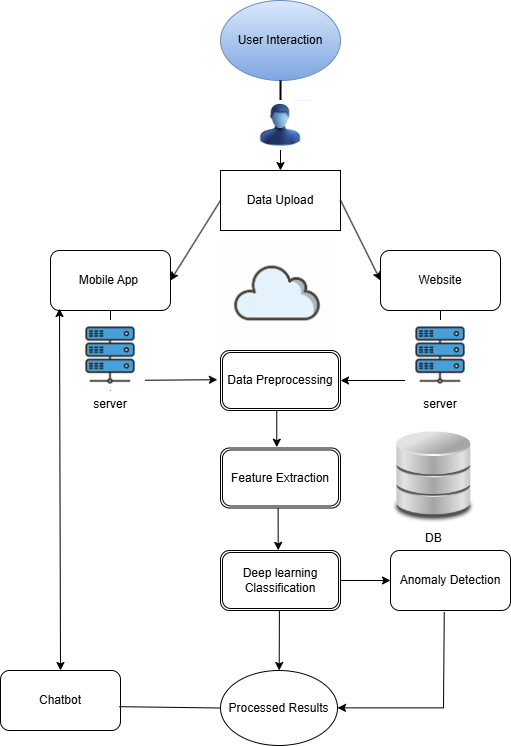


Figure ‑ System Architecture Diagram

### Data Preprocessing Module

* + Artifact **Removal**: Eliminate physiological artifacts such as eye blinks and muscle movements to enhance signal clarity.
  + Filtering: Apply band-pass filters to retain frequencies of interest (e.g., 1-50 Hz) and remove irrelevant noise.
* Segmentation: Divide continuous EEG data into meaningful epochs or time intervals for focused analysis.

A diagram of a process flow

Description automatically generated

Figure ‑ EEG Data Preprocessing Flowchart

### Feature Extraction Module:

* **Time-Domain Features:**

Extract statistical characteristics from EEG signals over time. Examples include mean amplitude, which measures the average signal strength in a segment, and root mean square (RMS), which quantifies signal magnitude. These features capture temporal patterns critical for identifying anomalies or classifications.

* **Frequency-Domain Features**:

Transform EEG signals into the frequency spectrum using methods like Fourier Transform. Key metrics include power spectral density (PSD), which indicates the signal's power distribution, and band power, which measures energy in specific frequency bands (e.g., alpha, beta, theta)

* **Time-Frequency Analysis**:

Analyze both temporal and frequency patterns dynamically using techniques like Short-Time Fourier Transform (STFT). This method captures frequency variations over time, making it suitable for transient event detection in EEG recordings.

### Classification Module

The Classification Module is designed to analyze the extracted features and classify EEG recordings as normal or abnormal. This is achieved using advanced machine learning models such as Convolutional Neural Networks (CNNs), which excel at capturing spatial patterns in EEG data, and Multilayer Perceptron's (MLPs), which are effective at analyzing complex feature relationships. These models are trained using the labeled TUAB dataset to ensure accuracy and reliability in distinguishing normal from abnormal EEG recordings.

The training process involves feeding the feature vectors obtained from the Feature Extraction Module into the classification models. Using supervised learning techniques, the models are optimized based on evaluation metrics such as accuracy, precision, recall, and F1-score. This ensures the models' robustness and generalizability for clinical applications.

The output of the Classification Module includes a binary classification label (normal or abnormal) and confidence scores, which provide a measure of the model's certainty in its predictions. These outputs form the foundation for further analysis and decision-making in the system.

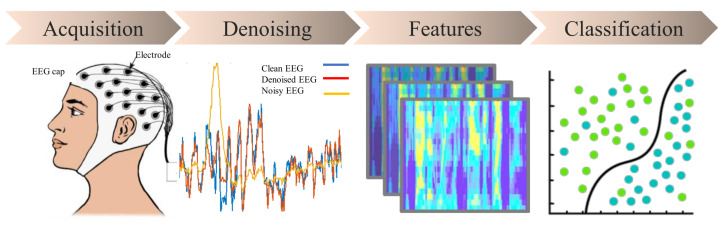


Figure ‑ EEG Processing Pipeline

### Chatbot Module

The Chatbot Module provides user support and facilitates interaction with the system by offering clear explanations of EEG classification results and detected anomalies. It is designed to enhance the system's accessibility and usability for clinicians and other end-users.

This module integrates natural language processing (NLP) techniques to understand user queries related to EEG data. It retrieves relevant information from the system, such as classification labels, confidence scores, and anomaly alerts, and presents the results in an easy-to-understand format. The chatbot also explains the significance of detected anomalies and provides suggestions for next steps, such as additional testing or clinical consultation.

The Chatbot Module supports interactive communication, enabling users to upload new EEG data, ask specific questions about results, and receive guidance on interpreting system outputs. This functionality ensures that users can efficiently leverage the system's capabilities without requiring extensive technical knowledge.

The output of the Chatbot Module includes concise textual explanations, visual representations (if applicable), and actionable insights, enhancing user engagement and decision-making in clinical workflows.

## Data Structure and Analysis

### Data Structure Description

The EDF (European Data Format) file is the standard format used in this project to store EEG (brain wave) recordings. This format organizes data into well-defined sections, making it highly suitable for analysis and integration with machine learning pipelines. The structure begins with a header, occupying 256 bytes, which contains essential metadata about the EEG recording. This metadata includes patient information, such as ID, age, and gender, as well as the date and time of the recording. The header also specifies the number of data records and EEG channels included in the file.

Following the header, each EEG channel or signal is described in a dedicated signal information section, also 256 bytes per signal. This section provides details such as the signal name (e.g., Fp1, Fp2), units of measurement (e.g., microvolts), the sampling rate, and the allowable range of signal values. These attributes are crucial for ensuring that the data is correctly interpreted and processed.

The main body of the EDF file consists of data records, which capture the actual EEG measurements over time. Each record includes signal values sampled at specific intervals and organized in temporal order to reflect changes in brain activity. This sequential arrangement is essential for understanding time-dependent neural patterns.

Optional annotations may also be included in the file, providing additional context about the EEG data. These annotations can mark significant events, such as seizures or eye blinks, or include timestamps and clinical notes that link observations to specific EEG patterns.

For this project, the EDF file format plays a pivotal role as the primary data input for preprocessing, feature extraction, and classification tasks. Its structured format simplifies data management and integration with tools like Python libraries (e.g., MNE). Its structured organization ensures accurate handling of critical details like signal attributes and annotations, which are vital for building reliable and effective machine learning models.

|  |  |
| --- | --- |
| **Section** | **Description** |
| Header (256 bytes) | Contains metadata, including patient ID, age, gender, recording time, number of records, and signals |
| Signal Information | Describes each EEG channel: name, units of measurement, sampling rate, and range |
| Data Records | Captures EEG measurements over time, organized sequentially. |
| Annotations (Optional) | Provides additional details such as event markers (e.g., seizures) and clinical notes. |

Table ‑ EDF File Structure

### Database Design

To support the storage and organization of EEG data for future model training, the focus is placed on saving EEG files (in Cloudflare R2). This approach ensures that the data is securely stored and accessible for future analysis and model enhancements. At this stage, the system will prioritize the storage of .edf files without additional metadata or analysis results.

In the future, as models are trained and refined, metadata such as classification results, confidence scores, and annotations will be added to enhance the system. For now, the Cloudflare R2 storage solution provides the required scalability and reliability for managing large volumes of EEG data, while allowing easy retrieval and processing for future model training.

This approach ensures that the necessary data is available for future machine learning applications, with flexibility for future extensions and enhancements when needed.

## Project Methodology/Techniques

This project begins with the preprocessing of EEG data, specifically EDF files, to ensure the data is clean and ready for analysis. This involves removing artifacts like eye blinks and muscle noise, applying band-pass filters to isolate relevant frequency ranges, and segmenting the data into meaningful time intervals. The preprocessing ensures the data’s compatibility with machine learning pipelines and improves its quality for further analysis.

Following preprocessing, feature extraction is performed to identify relevant patterns in the EEG signals. This step includes extracting time-domain features like mean amplitude and frequency-domain features such as power spectral density (PSD). The features are then combined into a vector format, which serves as input to the classification model.

The classification module employs advanced machine learning techniques, including Convolutional Neural Networks (CNNs) and Multilayer Perceptron’s (MLPs), trained on the TUAB dataset. These models distinguish between normal and abnormal EEG recordings with high accuracy. Post-classification, the anomaly detection module uses autoencoders and statistical thresholding to flag unusual patterns in the EEG data that may indicate critical conditions or irregularities.

For user interaction, a chatbot system has been integrated, utilizing APIs like LlamaIndex and Pinecone. This chatbot interprets user queries, provides classification results, and explains detected anomalies. Natural Language Processing (NLP) techniques are employed to ensure meaningful and accurate responses, enhancing the system's usability.

These methodologies collectively enable robust preprocessing, efficient classification, anomaly detection, and user interaction, forming the foundation of this EEG classification and anomaly detection system.

# Chapter Four Project Design & Implementation

## Project Design

### Graphical User Interface (GUI)

The Smart EEG Analyzer features a clean, web-based Graphical User Interface (GUI) designed to support easy interaction for both clinical users and researchers. The homepage acts as the central entry point, offering a clear call-to-action to upload EEG recordings for analysis. The design focuses on clarity and accessibility, ensuring users can navigate without prior technical knowledge.

The GUI includes:

* **Homepage** with EEG upload functionality
* **“How It Works”** page explaining the overall system flow and preprocessing pipeline
* **Contact Page** with basic user support or feedback form

All visual components are implemented using modern HTML, CSS, and JavaScript, emphasizing responsive design and accessibility. Navigation between pages is seamless, and upload validation provides immediate feedback to the user.

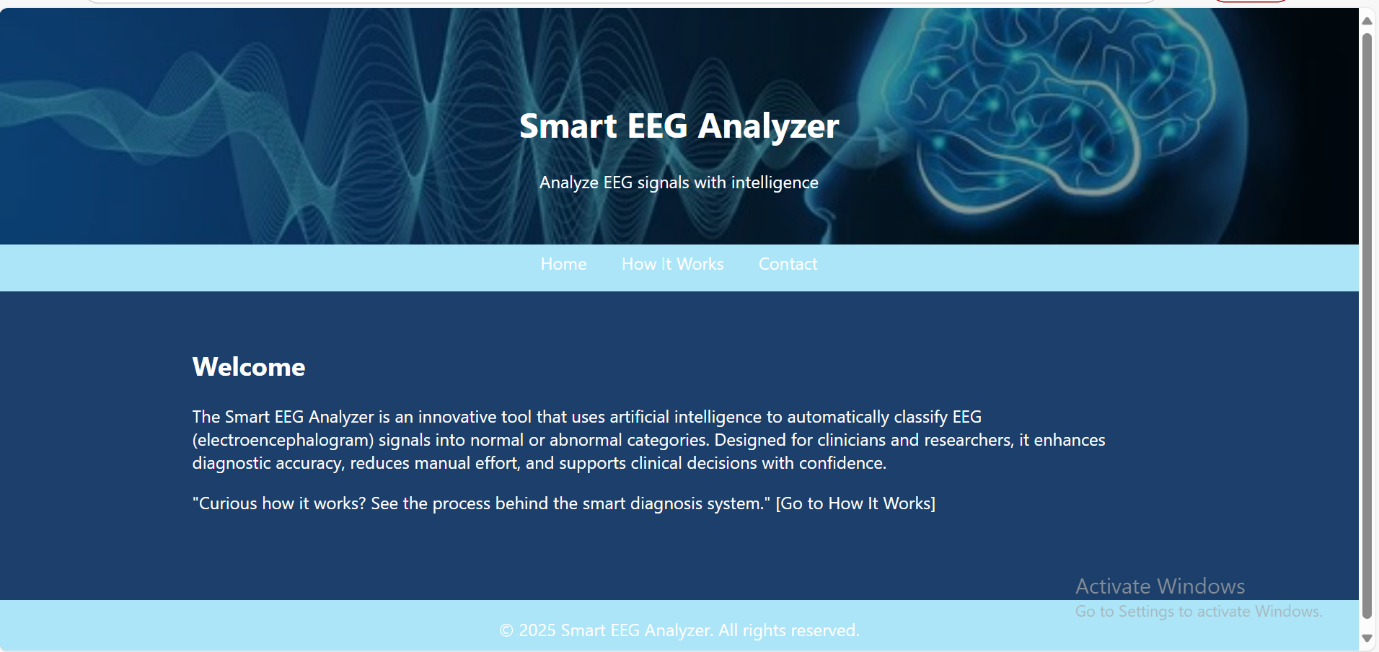


Figure ‑ Home Page Interface

This image shows the **Smart EEG Analyzer** homepage, which is designed to be clean and professional to attract users. The navigation bar at the top includes links like "Home," "How It Works," and "Contact." The welcome message explains the idea of the system, which uses artificial intelligence to analyze EEG signals and classify them as normal or abnormal. This helps doctors make informed decisions and improves diagnostic accuracy.

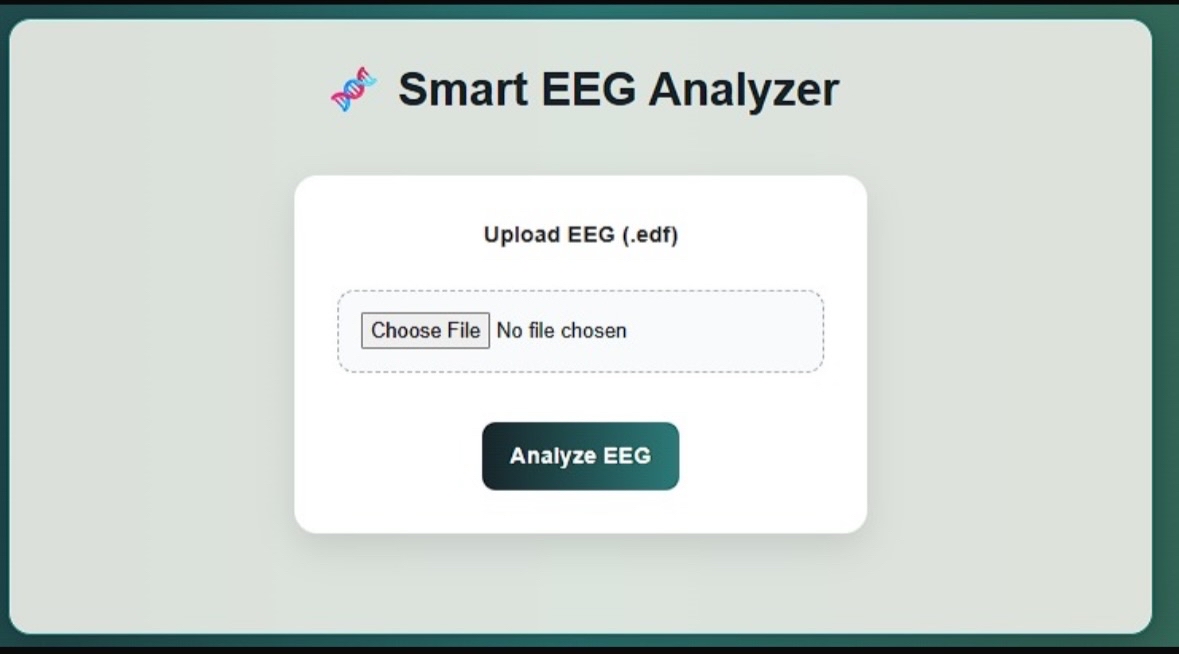


Figure ‑ EEG File Upload Interface

This image displays the section where the user uploads an EEG file in (.edf) format. After selecting a file, the user can click on the Analyze EEG button to start the analysis. The interface is simple and guides the user clearly through the steps, enhancing usability and user experience.

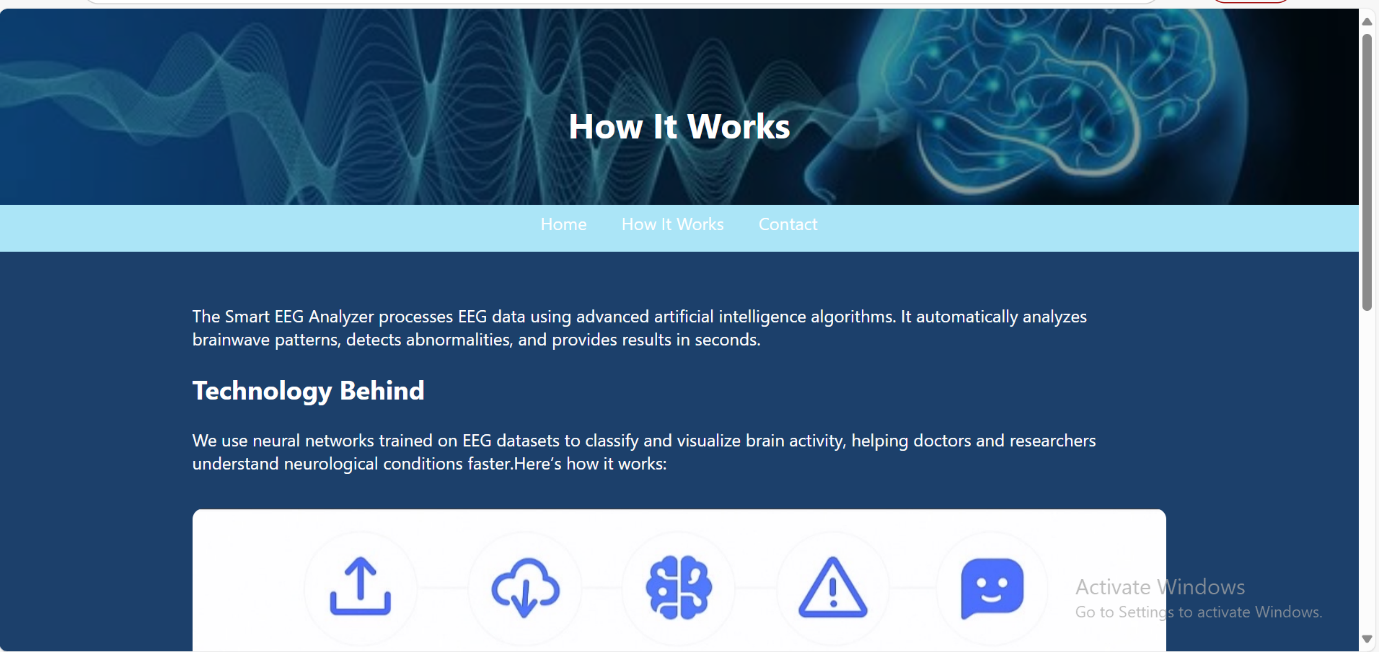
****

Figure ‑ "How It Works" Overview

This image introduces the "How It Works" section of the Smart EEG Analyzer website. It explains how the system uses advanced artificial intelligence algorithms to analyze EEG brainwave signals, detect abnormalities, and deliver results within seconds. The section highlights the use of deep learning technology to support medical professionals in understanding brain activity more efficiently.

**A screenshot of a computer program

AI-generated content may be incorrect.**

Figure ‑ Step-by-Step EEG Processing Workflow

This image explains the full processing pipeline in five clear steps:

1. Upload EEG Data: The user uploads EEG files in (.edf )format.
2. Preprocessing: The system cleans the signal by removing noise and normalizing it.
3. Classification: A trained AI model classifies EEG patterns as normal or abnormal using deep learning techniques such as CNNs and MLPs.
4. Anomaly Detection: If abnormalities are detected, the system highlights them for further analysis.
5. Results & Chatbot: The user receives a report and can interact with a built-in chatbot to get explanations about the results.

This stepwise explanation improves transparency and helps users understand the internal processing behind the smart diagnosis system.

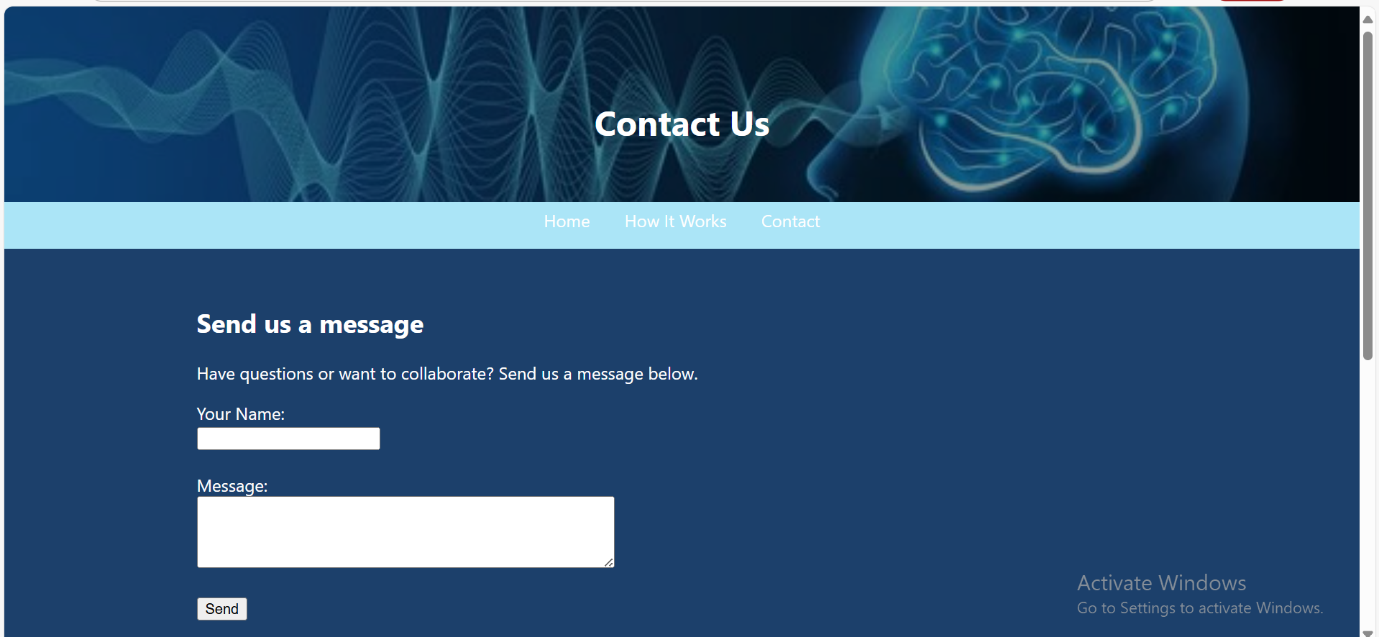
****

Figure ‑ Contact Us Page

This image displays the Contact Us section of the website, where users can send feedback, questions, or collaboration requests. It includes a simple form where users can enter their name and message. This section strengthens user communication and encourages engagement with the project developers or research team.

The graphical user interface (GUI) was carefully designed to ensure clarity, responsiveness, and accessibility for users with varying technical backgrounds. It is organized into distinct visual sections that guide the user through the process—from uploading the EEG file, observing real-time progress indicators, reviewing analysis results through dynamic plots, and interacting with an AI-powered chatbot. The GUI layout balances functionality and aesthetics, offering a modern design that is both minimalistic and highly informative. This design supports smooth user navigation and enhances the overall user experience.

### Data Workflow

The system follows a structured data pipeline for EEG classification:

1. Data Loading: EEG signals are loaded from the TUAB dataset.
2. Channel Selection: 21 common EEG channels are retained.
3. Preprocessing: Bandpass filtering (0.5–40 Hz), scaling to µV,
4. Segmentation: EEG recordings are split into 10-second windows (1000 samples).
5. Feature Extraction (MLP): Statistical, spectral, wavelet, and catch22 features.
6. Model Training: Deep4Net, ShallowFBCSPNet, or MLP is trained on processed data.
7. Evaluation: Classification performance is measured using Accuracy, Precision, Recall, F1.

## Project Implementation

### Data Preparation

#### *Dataset Loading and Labeling*

In this project, we used the TUH Abnormal EEG Corpus (TUAB v3.0.1), a clinically labeled EEG dataset. The data was loaded separately for training and evaluation using the braindecode.datasets.TUHAbnormal loader. EEG recordings were classified into two categories: normal (labeled as 0) and abnormal (labeled as 1), based on their file path. This binary labeling enabled the model to distinguish between healthy and pathological brain activity.

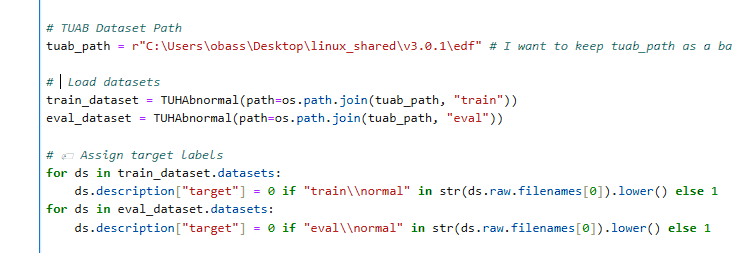


Figure 4‑6 Dataset Loading and Labeling

#### Channel Selection

To ensure consistency across all EEG recordings, a fixed set of 21 commonly used EEG channels was selected and applied to both training and evaluation datasets. This step helps standardize the input dimensions and focuses the analysis on clinically relevant brain regions. Channels not found in a recording were automatically excluded to avoid errors.

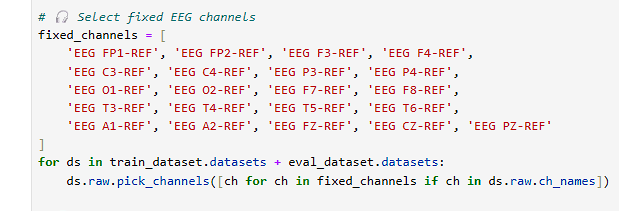


Figure ‑ Channel Selection

#### Muscle Artifact Removal

Muscle artifacts were removed only from normal EEG recordings to avoid introducing bias during model training. This was done using the annotate\_muscle\_zscore function from the MNE library, which identifies segments with high-frequency noise (typically between 30–90 Hz) associated with muscle activity. The annotated segments labeled as 'BAD\_MUSCLE' were then deleted to retain only clean, artifact-free EEG data.

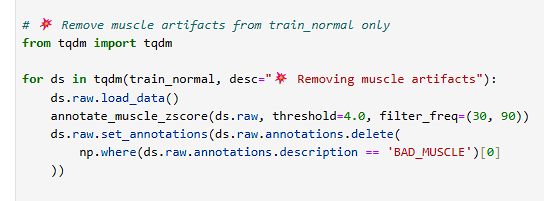


Figure ‑ Muscle Artifact Removal

#### Signal Preprocessing

Each EEG signal underwent two essential preprocessing steps to prepare the data for analysis:

1. Bandpass Filtering (0.5–40 Hz): Applied to remove low-frequency drifts and high-frequency noise, preserving the key brainwave frequency bands (delta to gamma).
2. Unit Conversion to Microvolts (µV): EEG data, originally in volts, was scaled by a factor of 10610^6106 to convert it to µV — the standard unit in clinical EEG analysis

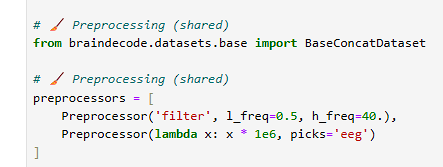


Figure ‑ Signal Preprocessing

#### Windowing

To standardize the input size for the classification model, all EEG signals were segmented into fixed-length windows of 1000 samples. This approach ensures that the model processes consistent input shapes regardless of the original recording length.

The following parameters were used during windowing:

* start\_offset\_samples = 0: Windowing starts from the beginning of each signal.
* Each window contains 1000-time samples.
* Windows are non-overlapping; each new window starts right after the previous one ends.
* Incomplete windows at the end of a signal were discarded to maintain uniform input dimensions.

This windowing process was applied to both training and evaluation datasets.

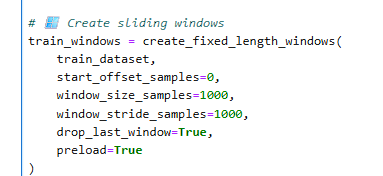


Figure ‑ Windowing

#### Data Balancing

To address class imbalance in the training data, the RandomOverSampler technique from the imblearn library was applied. This method duplicates samples from the minority class to equalize the number of normal and abnormal EEG windows, ensuring the model does not become biased toward the majority class during training.

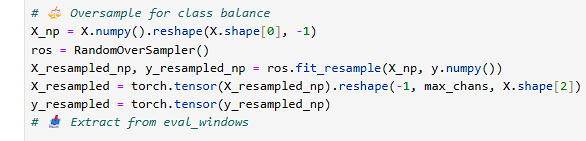


Figure ‑ Data Balancing

#### Feature Engineering

In this stage, handcrafted features were extracted from each EEG window across all channels. To optimize performance, only selected features were computed from five major categories: time-domain, spectral, statistical, and nonlinear descriptors. The following subsections describe the selection strategy, feature types, and the extraction logic.

To capture meaningful patterns from EEG signals, a set of handcrafted features was extracted from each window across all EEG channels. The extracted features were grouped into five main categories:

#### Feature Selection Strategy

To optimize both training performance and model generalization, a feature selection process was applied before training the MLP classifier. The full handcrafted feature set was initially extracted across all EEG channels and categories (time-domain, PSD, Hjorth, Wavelet, and Catch22). However, many of these features were redundant or uninformative for the classification task.

To reduce dimensionality, a feature importance-based selection method was used. Specifically:

The selection was based on evaluating which features contributed most to the classification accuracy of a preliminary MLP model. Only features that helped improve the validation F1-score were retained.

The selected feature names were stored in a text file named selected\_feature\_names.txt. During the feature extraction, only features listed in this file were computed dynamically. This selective computation significantly reduced memory usage, processing time, and overfitting risk.



Figure ‑ Feature Selection Strategy

#### Handcrafted Feature Categories

* Time-domain features: Mean, standard deviation, min, max, skewness, and kurtosis values of the signal.
* Power Spectral Density (PSD): Average power across key frequency bands — delta, theta, alpha, beta, and gamma.
* Hjorth Parameters: Measures of signal activity, mobility, and complexity based on derivatives.
* Wavelet Energy: Energy of wavelet coefficients obtained from multi-level discrete wavelet decomposition.
* Catch22 Features: A standardized collection of 22 statistical features known to capture diverse time-series characteristics.

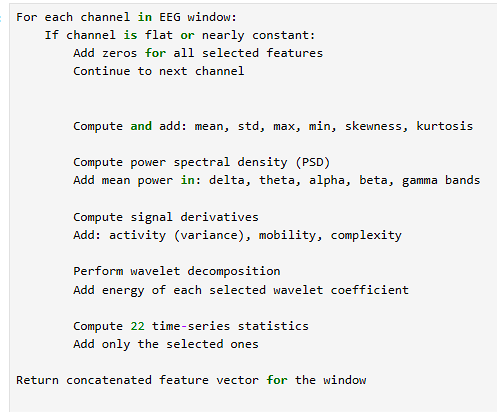


Figure ‑ Handcrafted Feature Categories

#### Chatbot Integration

The Smart EEG Analyzer integrates an intelligent chatbot interface powered by a large language model (DeepSeek/GPT) to assist users in interpreting EEG results in natural language. The chatbot uses the model's prediction summary (e.g., abnormal segment count and percentage) and the user's query to generate human-friendly explanations.

This enhances usability for non-technical users and supports informed decision-making through contextual understanding.

The backend implementation includes:

* Tracking the number of abnormal segments per EEG session.
* Dynamically composing prompts based on user queries.
* Sending prompts to DeepSeek/GPT via an API request.
* Returning conversational explanations to the frontend chatbot.



Figure ‑ Chatbot Integration

### Machine Learning Models

To classify EEG recordings as normal or abnormal, multiple machine learning models were implemented and evaluated. The goal was to identify the architecture that delivers the best balance of accuracy, efficiency, and robustness for our specific dataset.

Initial experiments included models such as Deep4Net, ShallowFBCSPNet, and various other baseline classifiers. However, many of these models resulted in low validation accuracy or failed to generalize well to unseen data due to overfitting or insufficient feature representation.

Following this, we designed and trained a custom Multi-Layer Perceptron (MLP) using carefully selected handcrafted features. This model significantly outperformed the previous architecture, achieving the highest F1-score and stability across validation folds.

#### Baseline and CNN Models

Model 1: Deep4Net

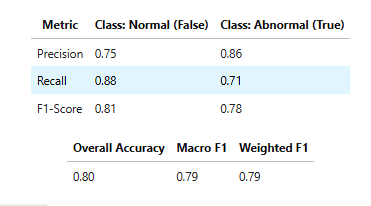
* No oversampling, learning rate: 1e-5
* Accuracy: ~61%
* F1 (Abnormal): 0.38

A graph with blue lines

AI-generated content may be incorrect.

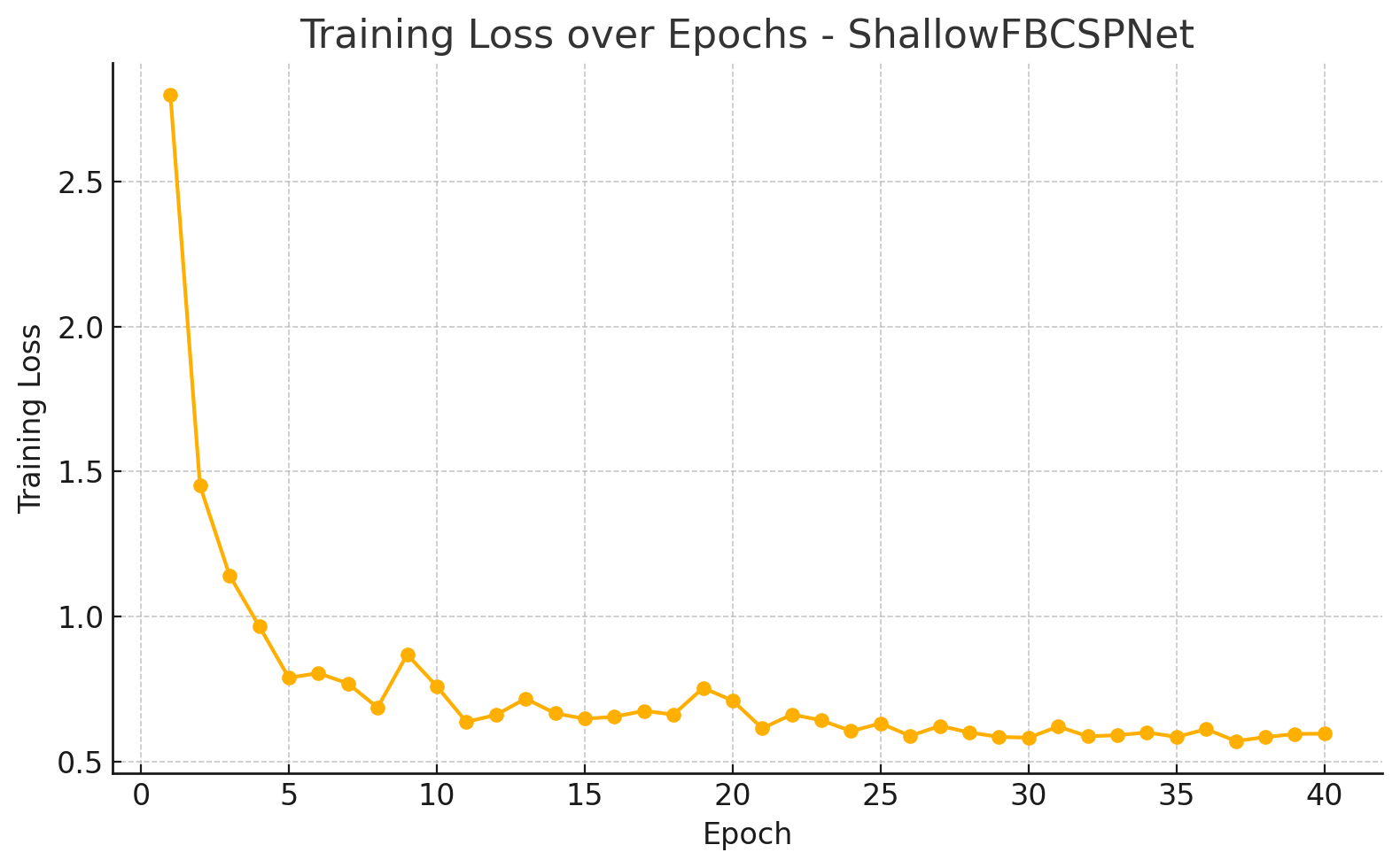
Model 2: Deep4Net (Tuned)

* Oversampling + class weights, learning rate: 0.001
* Accuracy: 80%
* F1 (Abnormal): 0.78



Model 3: ShallowFBCSPNet (Standard)

* 40 Epochs
* Accuracy: 70%
* F1 (Abnormal): 0.74



Model 4: ShallowFBCSPNet (Extended Training)

* 70 Epochs
* Accuracy: 76–78%
* F1: Up to 0.74

#### Final Model: Custom MLP Preprocessed + artifact-cleaned signals

* Feature extraction (Time, PSD, Hjorth, Wavelet, Catch22)
* Best Accuracy: 83.9%, F1: 0.834

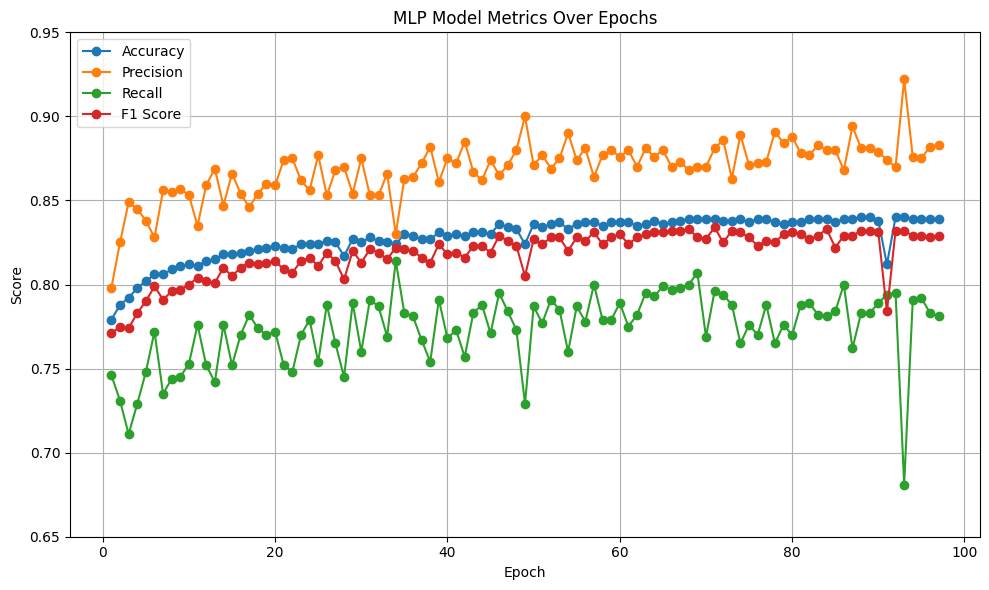


Figure ‑Final Model: Custom MLP Preprocessed + artifact-cleaned signals

## Visualizations

* The system includes dynamic EEG plots and prediction result displays in the UI. These visualizations allow users to:
* Explore EEG channel activity
* Users can select from 21 standard EEG channels through a dropdown menu (see Figure 4-16). This enables:
* Focusing on specific brain regions or lobes.
* Reducing clutter in multi-channel plots.
* Exploring artifact-heavy channels interactively

A screenshot of a computer

AI-generated content may be incorrect.

Figure ‑ select from 21 standard EEG channels

* Abnormal Window Detection and Navigation
  + EEG signals are processed in 1000-sample sliding windows, and each window is independently evaluated by the trained MLP classifier. For each segment:
  + A prediction is generated (normal or abnormal).
  + A confidence score is calculated using softmax.
  + Segments with confidence > 0.90 and labeled as abnormal are flagged.
  + The dashboard allows users to:
  + Navigate through individual signal windows using the “Previous” and “Next” buttons.
  + Manually inspect any window index, aiding detailed review of abnormal regions.

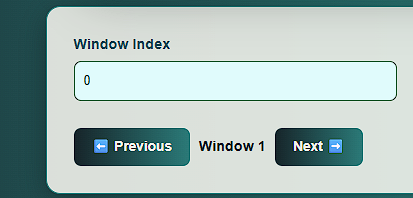


Figure ‑ Abnormal Window Navigation

* After analyzing all EEG windows, the system generates a file-level decision based on the percentage of abnormal segments detected.
  + A pie chart visualizes the distribution of predicted labels (normal vs. abnormal).
  + If the proportion of abnormal windows exceeds a predefined threshold (e.g., 10%), the recording is classified as ABNORMAL.
  + This provides a clear summary to clinicians without manually reviewing each segment.



Figure ‑ pie chart visualizes the distribution of predicted labels

* EEG Plot with Abnormal Segment Focus

To support visual clinical interpretation, the system allows users to plot EEG signals centered on detected abnormal segments. This feature enables users to:

* + Review EEG waveforms across all standard channels within the abnormal region highlighted.

The plotting is handled programmatically via the plot\_window() function, which ensures the selected time window is automatically centered around the abnormal segment while avoiding overflow beyond the signal’s duration.

A graph of a sound wave

AI-generated content may be incorrect.

Figure ‑ EEG Plot with Abnormal Segment Focus

### Chatbot Assistance

The system includes an integrated AI-powered chatbot to assist users in understanding EEG classification results. The chatbot appears as an interactive floating icon and expands into a full modal interface upon engagement. It enables users — especially non-experts — to ask natural-language questions like:

*"What does abnormal mean?"* or *"How confident is the result?"*

The chatbot provides explainable feedback about:

* EEG classification labels (normal/abnormal),
* Confidence levels,
* Model decisions,
* EEG terminology.

It enhances transparency, usability, and clinical accessibility of the system

A chatbot with a robot

AI-generated content may be incorrect.

Figure ‑ Chatbot Assistance

Backend Implementation (with Database Integration)

The backend of the Smart EEG Analyzer system is responsible for managing core functionalities such as file processing, cloud integration, session management, and API communication with the frontend and machine learning modules. It was designed with modular and scalable architecture to ensure secure handling of sensitive EEG data and real-time interaction with the user interface

.

#### Upload and Download Logic (Cloud Database Integration)

One of the central backend components is the EEG file storage system, which is implemented using a secure cloud-based solution (Cloudflare R2). The backend begins by establishing a connection to the cloud using a defined endpoint, secure access credentials, and a dedicated storage bucket.

To ensure file uniqueness and avoid redundant uploads, the system generates a unique MD5 hash based on each file’s content. This hash is then used as the file’s identifier within the cloud. Before uploading, the backend checks whether a file with the same hash already exists. If it does, the upload is skipped entirely, saving bandwidth and storage space.

When the file is not already present, it is uploaded using multipart configuration, where large files are split into smaller parts and sent in chunks. A progress tracking mechanism is used during the upload to provide real-time updates to the frontend, enhancing user feedback.

The download process mirrors this logic. The system streams files in segments from the cloud to a temporary local directory, with progress feedback being updated live through a custom download tracking utility. This entire storage logic is built using reusable utility functions, which improves maintainability, supports future expansion (e.g., file versioning or encryption), and ensures robust data handling

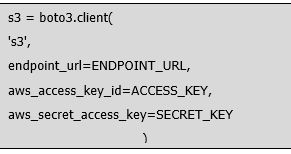
****

Figure ‑ initializes a secure connection to Cloudflare R2

A computer code with black text

AI-generated content may be incorrect.

Figure ‑ Generates a unique MD5 hash

Generates a unique MD5 hash to prevent duplicate file uploads

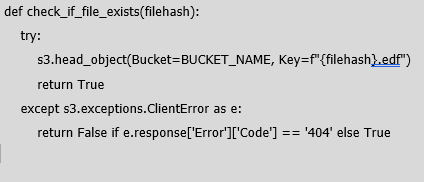
****

Figure ‑ Verifies whether the file already exists in the cloud before attempting upload

.

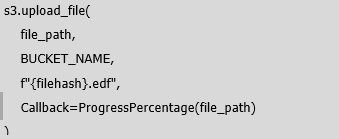
****

Figure ‑ Uploads large EEG files in chunks

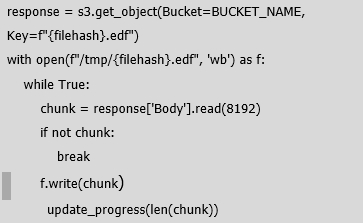
****

Figure ‑ Downloads a file in small chunks and updates progress dynamically

**4.4.2 Session Management**

To support multi-user functionality and asynchronous processing, the backend uses a session-based structure to store and manage file states, progress, model predictions, and extracted features. Each uploaded file is tracked using its unique hash, and its associated processing data is saved in the session. This design allows the frontend to query the backend at any time for the status or results of a particular file, ensuring a responsive and state-aware experience for the user.

Session data also includes information like processing stages, classification results (normal/abnormal), and the raw EEG signal. This enables efficient re-use of previously processed data without repeating computations or storage operations.

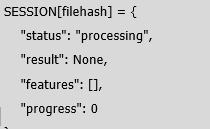


Figure ‑ Stores the current processing state of the EEG file in a centralized session



Figure ‑ Keeps the front end informed of backend progress in real-time

# Chapter Five Evaluation and Experimental Results

The following table presents a comparative summary of all machine learning models implemented in the Smart EEG Analyzer project. Each model was trained and evaluated on windows extracted from the TUAB dataset, with various preprocessing techniques applied.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision**  **(N / A)** | **Recall**  **(N / A)** | **F1-Score**  **(N / A)** |
| Deep4Net | ~61% | 0.56 / 0.96 | 0.99 / 0.24 | 0.72 / 0.38 |
| Deep4Net (Tuned) | 80% | 0.75 / 0.86 | 0.88 / 0.71 | 0.81 / 0.78 |
| ShallowFBCSPNet | 70% | 0.80 / 0.65 | 0.54 / 0.87 | 0.64 / 0.74 |
| ShallowFBCSPNet (70 ep.) | ~76–78% | Like above | Like above | 0.74 |
| MLP on Features | 83.9% | 0.86 / 0.86 | 0.81 / 0.81 | 0.83 / 0.83 |

Table ‑ comparative summary of all machine learning models

The MLP model demonstrated the best overall performance, achieving an F1-score of 0.834 by leveraging feature-rich input, class balancing, and artifact removal. The

# Chapter Six

## Conclusion

From a clinical perspective, the Smart EEG Analyzer enhances the diagnostic workflow for detecting abnormal brain activity. By automating the classification of EEG recordings, the system assists neurologists in identifying critical neurological conditions with improved efficiency, consistency, and accuracy. The integration of anomaly detection alongside explainable chatbot feedback ensures that results are not only accurate but also interpretable and accessible to non-technical users.

From a technical perspective, the system effectively implemented and compared multiple machine learning models on real-world EEG data. Among these, the MLP model using handcrafted statistical features achieved the best performance, with an accuracy of 83.9% and an F1-score of 0.834. This outcome underscores the importance of careful preprocessing, feature selection, and class balancing. Deep learning models such as Deep4Net and ShallowFBCSPNet also exhibited strong performance on raw EEG windows and remain promising candidates for future enhancements.

The system’s modular architecture, support for both raw-signal and feature-based pipelines, and its deployment-ready visualization interface establish it as a scalable and extensible foundation for next-generation EEG diagnostic platforms.

## Future Work

Potential future enhancements include:

* Real-time EEG Monitoring: Extend the system for real-time EEG stream analysis.
* Pediatric EEG Support: Add models tailored to pediatric EEG patterns.
* Advanced Feature Selection: Use genetic algorithms to improve feature choice.
* Multi-class Classification: Move beyond binary classification to identify specific disorders like epilepsy or encephalopathy.
* Clinical Integration: Enable direct data exchange with hospital databases and electronic health record systems**.**
* Mobile Application: Deploy the analyzer as an Android/iOS app with offline capabilities.

These improvements would further increase the system’s reliability, reach, and clinical relevance.

# References

[1] Craik, Alexander, Yongtian He, and Jose L. Contreras-Vidal. "Deep learning for electroencephalogram (EEG) classification tasks: a review." *Journal of Neural Engineering*  16.3 (2019): 035001. DOI: [10.1088/1741-2552/ab0ab5](https://iopscience.iop.org/article/10.1088/1741-2552/ab0ab5/meta).

[2] Obeid, Iyad, and Joseph Picone. "The Temple University Hospital EEG Data Corpus." *Frontiers in Neuroscience* 10 (2016): 196. DOI: 10.3389/fnins.2016.00196.

[3] Zhang, T., Wang, T., Chen, C., Guo, L., & Cheng, G. (2020). Real-time Inference and Detection of Disruptive EEG Networks for Epileptic Seizures. *Scientific Reports, 10*(1), 9056. DOI: 10.1038/s41598-020-65401-6.

[4] Zhang, T., Wang, T., Chen, C., Guo, L., & Cheng, G. (2023). Automated Recognition of Epilepsy from EEG Signals Using Machine Learning. *Scientific Reports, 13*(1), 41537. DOI: 10.1038/s41598-023-41537-z.

[5] Poatek, "Building Open-Source LLM-Based Chatbots Using LlamaIndex," Medium, Dec. 2023. [Online]. Available: <https://medium.com/poatek/building-open-source-llm-based-chatbots-using-llama-index-e6de9999ee76>.  
[6] AppsDevPro. (n.d.). *Integrate machine learning in Android apps*. AppsDevPro. Retrieved December 25, 2024, from <https://www.appsdevpro.com/blog/integrate-machine-learning-in-android-apps/>