ABSTRACT

Faculty of Computers & Artificial Intelligence

Benha University

Catch The AI

A senior project submitted in partial fulfilment of the requirements for the degree of Bachelor of Computers and Artificial Intelligence.

Artificial Intelligence Department,

Project Team

1- Romani Nasrat Shawqi
2- Ahmed Mohamed Ali
3- Zeyad Elsayed Abdel-Azim
4- Sara Reda Moatamed
5- Reham Moustafa Ali
6- Rawan Abdel-Aziz Ahmed
7- Abdallah Mohammed Abdel-monnem
8- Mohannad Ayman salah
9- Mohammed Abdallah Abdel-salam

Under Supervision of

Prof. Eman Abdel-Latef

Eng. Sahar Mohamed

Artificial Intelligence,

JUNE 2024

ABSTRACT

The Growing Challenge of AI-Generated Media: A Multimodal Detection System for Enhanced Trust

The rapid evolution of deep learning, particularly Generative Adversarial Networks (GANs) and Variational Auto-encoders (VAEs), has yielded remarkably realistic AI-generated media content. While advantageous for creative fields like film and advertising, this technology presents a growing challenge for Multimedia Information Process and Retrieval (MIPR) systems, facial recognition, and speech recognition. The potential for manipulating these systems with AI-generated content raises concerns about the spread of misinformation and the erosion of trust in online interactions.

This project tackles this challenge by proposing a comprehensive model for detecting the origin of multimedia content, encompassing text, images, and audio. Utilizing advanced algorithms, the system identifies subtle manipulations indicative of AI generation. Three primary detection models form the core of this system:

- **AI Image Generation Detection:** Accurately identifies images created by artificial intelligence.
- **AI Text Generation Detection:** Recognizes text authored by AI algorithms and distinguishes it from human-generated content.
- **AI-Generated Audio Detection:** Identifies and differentiates between audio content generated by artificial intelligence and authentic human-recorded audio.

By prioritizing content authenticity across all these modalities, this project aims to safeguard against the potential pitfalls of AI-generated media. It promotes a digital environment characterized by transparency, trust, and ethical AI use, ultimately fostering a more reliable and trustworthy online ecosystem.

Table of Contents

Table of Contents

2 List of Figures

3 LIST OF ACRONYMS/ABBREVIATIONS

AI	**Artificial Intelligence**: A field of computer science focused on creating systems capable of performing tasks that typically require human intelligence, such as visual perception, speech recognition, decision-making, and language translation. (Chapter 3: Introduction)
LLMs	**Large Language Models**: Deep learning models that are trained on vast amounts of text data to understand and generate human-like language, often used for tasks such as translation, summarization, and question answering. (Chapter 1: Introduction)
GANs	**Generative Adversarial Networks**: A class of neural networks used in unsupervised learning, where two networks (generator and discriminator) compete against each other to create data that is indistinguishable from real data. (Chapter 1: Introduction)
UML	**Unified Modelling Language:** A standardized modelling language in software engineering used to specify, visualize, construct, and document the artifacts of a system. (Chapter 2: System Design and Analysis)
GPT	**Generative Pre-trained Transformer:** A type of large language model developed by OpenAI that uses deep learning to produce human-like text. (Chapter 1: Overview)
RoBERTa	**Robustly optimized BERT approach:** A derivative of BERT (Bidirectional Encoder Representations from Transformers) with improved training strategies, achieving better performance on NLP tasks. (Chapter 2: Literature Review & Chapter 4: Methodology)

BERT	**Bidirectional Encoder Representations from Transformers**: A transformer-based model designed to understand the context of a word in search queries. (Chapter 2: Literature Review & Chapter 4: Methodology)
DeBERTa	**Decoding-enhanced BERT with disentangled attention:** An improvement over BERT that introduces disentangled attention and enhanced decoding for better language understanding. (Chapter 2: Literature Review & Chapter 4: Methodology)
MFCCs	**Mel-Frequency Cepstral Coefficients:** Features used in automatic speech and audio processing that represent the short-term power spectrum of a sound. (Chapter 2: Literature Review & Chapter 4: Methodology)
PRNU	**Photo-Response Non-Uniformity**: A unique pattern noise inherent in imaging sensors, used in forensic analysis to identify and authenticate digital images. (Chapter 2: Literature Review)
API	**Application Programming Interface:** A set of protocols and tools for building and interacting with software applications, allowing different systems to communicate. (Chapter 5: Implementation)
RESTful	**Representational State Transfer (REST):** An architectural style for designing networked applications, relying on stateless communication and standard HTTP methods. (Chapter 3: System Design & Chapter 4: Methodology)
MVC	**Model-View-Controller:** A design pattern for developing user interfaces that separates an application into three interconnected components: Model, View, and Controller. (Chapter 3: Design and Implementation)

UI	**User Interface:** The space where interactions between humans and machines occur, including design elements such as buttons, menus, and forms. (Chapter 2: System Design)
ASVspoof	**Automatic Speaker Verification Spoofing:** A challenge or technology focused on detecting spoofing attacks in speaker verification systems to ensure the authenticity of the speaker's identity. (Chapter 4: Data Acquisition)
SFHQ	**Single-Frequency Harmonic Quantization:** A technique in signal processing for analysing and representing periodic signals. (Chapter 4: Signal Processing)
CNN	**Convolutional Neural Network:** A class of deep neural networks primarily used for analysing visual data, known for its ability to automatically and adaptively learn spatial hierarchies of features. (Chapter 2: Existing Techniques & Chapter 4: Methodology)
ROC	**Receiver Operating Characteristic:** A graphical plot used to illustrate the diagnostic ability of a binary classifier system, showing the trade-off between sensitivity and specificity. (Chapter 4: Evaluation Metrics)
AUC	**Area Under the Curve:** A performance measurement for classification models, particularly in ROC analysis, representing the degree of separability achieved by the model. (Chapter 4: Evaluation Metrics)

Chapter One

1. Introduction

1.1 Project Overview

The realm of artificial intelligence (AI) is experiencing rapid growth[1],[2] with tech giants vying for dominance in groundbreaking technologies like transformers, Generative Adversarial Networks (GANs), and Large Language Models (LLMs). However, alongside these advancements arises a critical challenge: the increasing difficulty in differentiating AI-generated content from authentic human-created material. This challenge extends to images, text, and even audio, raising concerns about the spread of misinformation and the erosion of trust in online interactions. Our project proposes a comprehensive online platform equipped with specialized detection models to address this issue. By leveraging machine learning techniques, we aim to empower users to discern between AI-generated and human-created content, fostering transparency and trust in the digital landscape.

1.2 Problem Statement

The rapid evolution of AI, particularly with advanced techniques like GANs[3] and transformers, has enabled the creation of remarkably lifelike and convincing AI-generated images, text, and audio. This progress presents a significant challenge: the increasing difficulty in distinguishing AI-generated content from real content. This situation opens doors for malicious actors to create and disseminate deceptive information, like the challenges posed by "fake news." Large Language Models (LLMs) exacerbate this problem by generating human-quality text[4], posing unique challenges in educational settings. As open-source AI technologies advance, the line between machine-generated and human-created media blurs, paving the way for potential misuse[5]. Robust systems to identify content origin become paramount to protect trust and transparency in online interactions.

The potential consequences of undetected AI-generated media are vast, including the spread of misinformation, erosion of trust, academic dishonesty, and cyberbullying.

1.3 Project Objective

The objective of this project is to develop a comprehensive detection system that identifies and differentiates between human-generated and AI-generated multimedia content. By leveraging advanced algorithms and deep learning

techniques, this system aims to safeguard against the proliferation of deceptive and manipulated media. [6] The project focuses on three primary detection models:

- **AI Image Generation Detection**: Accurately identify images generated by artificial intelligence and distinguish them from human-created images.
- **AI Text Generation Detection**: Recognize text generated by AI and determine its authorship[7].
- **AI-Generated Audio Detection**: Identify and differentiate between audio content generated by artificial intelligence and authentic human-recorded audio.[8]

By addressing these challenges, the project seeks to enhance the reliability of multimedia information, protect facial and speech recognition systems, and promote a trustworthy digital ecosystem through transparency and ethical AI use.

1.4 Scope and Requirements

1.4.1 key Features:

- **AI-Generated Image Detection:** Clearly identify and differentiate between AI-created and human-created images, safeguarding against the misuse of visuals.
- **AI-Generated Text Detection:** Recognize text generated by AI and distinguish it from human-written content, ensuring the authenticity of information.
- **AI-Generated Audio Detection:** Identify and differentiate between audio content generated by AI and authentic human-recorded audio. [9]

1.4.2 Core Requirements

- **Machine Learning Model Development:** Building robust machine learning models for accurate detection across all modalities.[10]
- **Data Collection and Preprocessing:** A substantial and well-categorized dataset for each media type (images, text, and audio) is crucial for training effective models.[11]

1.4.3 Scope Exclusions

- **Privacy and Security Measures:** While important, these are not the focus of this project but should be considered during development.
- **Technical Challenges:** Acknowledging the exclusion of addressing all technical challenges, especially advanced, hidden AI techniques.

1.5 Target Audience

The target audience for this project includes:

- **Academics and Researchers:** Individuals conducting research in AI media detection.
- **Software Developers:** Professionals interested in implementing AI-driven solutions.
- **Students:** Those studying computer science, AI, and related fields.
- **Media Analysts:** Experts needing tools to identify AI-generated content.
- **Law Enforcement Agencies:** Utilizing AI-generated audio detection for investigating deepfakes.[12]

1.6 Methodology

The project follows a structured methodology:

1. **Requirement Analysis**: Identify and document requirements, engage with stakeholders.
2. **System Design**: Design architecture, create UML diagrams.
3. **Data Collection and Preprocessing**: Gather and preprocess datasets.
4. **Model Development**: Utilize pretrained models for training and fine-tuning.
5. **Implementation**: Develop frontend with React, backend with Django.
6. **Integration and Testing**: Perform thorough testing.
7. **Deployment**: Deploy using Docker, host on cloud platforms.[13]
8. **Documentation**: Document development process, prepare user manuals.

1.7 Document Structure

The document is structured as follows: [14]

1. **Title Page**: Project title, author, institution, date.
2. **Abstract**: Summary of objectives, methods, outcomes.
3. **Table of Contents**: Sections and subsections with page numbers.
4. **Introduction**: Background, motivation, problem statement, objectives, scope.
5. **Literature Review**: Existing research and techniques.
6. **Project Methodology**: Project plan, technologies, system design, data handling, model development.
7. **Implementation**: Frontend and backend development, integration, deployment strategies.
8. **Testing and Results**: Testing methodologies, performance analysis, results.
9. **User Manual**: Installation instructions, system requirements, usage guide.
10. **Conclusion and Future Work**: Findings, limitations, future enhancements.
11. **References**: Academic papers, books, other sources.
12. **Appendices**: Additional materials such as code snippets, data samples, detailed diagrams.

Chapter Two

2. Literature Review

2.1 Existing Techniques

In recent years, the proliferation of AI-generated media has necessitated the development of robust detection techniques. Various methods have been proposed and implemented across different modalities—text[15], audio, and images. Each modality presents unique challenges and has prompted the development of specialized techniques.

2.1.1 Text

AI-generated text[16] detection has evolved significantly with the advent of sophisticated language models like GPT-3 and beyond. Early approaches relied on statistical methods and linguistic features such as word frequency and sentence structure. However, these methods were often inadequate against advanced AI-generated text. Current state-of-the-art techniques leverage deep learning models, particularly those based on transformers. For example:

- **RoBERTa** (Robustly optimized BERT approach) has been fine-tuned for detecting AI-generated text by training on large datasets containing both human and AI-generated text.
- **DeBERTa** (Decoding-enhanced BERT with disentangled attention) introduces new mechanisms to improve the detection of subtle nuances in AI-generated text, which are often missed by simpler models.

2.1.2 Audio

Detection of AI-generated audio, such as synthesized speech or deepfake voices, has also seen substantial advancements. Traditional methods focused on spectral analysis and acoustic features. With the rise of powerful speech synthesis models like Wav2Vec2[17], more sophisticated techniques are required. Modern approaches include:

- **Wav2Vec2**: This model, originally designed for automatic speech recognition, can be adapted for detecting anomalies in speech patterns indicative of synthetic audio.
- **Mel-frequency cepstral coefficients (MFCCs)**: These are still used in conjunction with neural networks to detect inconsistencies in the spectral properties of AI-generated audio.

2.1.3 Images

The detection of AI-generated images, such as those created by GANs (Generative Adversarial Networks), involves analysing various visual features. Traditional image analysis techniques are often insufficient against high-quality GAN-generated images. Modern approaches include:

- **EfficientNet**: This model architecture, known for its efficiency and high performance on image classification tasks, is utilized to discern subtle differences between real and AI-generated images.
- **Image Forensics Techniques**: Techniques such as analysing photo response non-uniformity (PRNU) and checking for inconsistencies in lighting, shadows, and reflections are used in conjunction with deep learning models.

2.2 Strengths and Limitations[18]

2.2.1 Strengths

- **Accuracy**: Modern deep learning models, particularly those based on transformers, have significantly improved the accuracy of AI-generated media detection.
- **Scalability**: These models can be scaled to handle large datasets, making them suitable for real-world applications.
- **Versatility**: Techniques developed for one modality (e.g., text) can often be adapted for another (e.g., audio), leveraging common underlying principles.

2.2.2 Limitations

- **Evolving Nature of AI**: As AI models improve, detection techniques must continually adapt. This ongoing "arms race" presents a significant challenge.[19]
- **Resource Intensive**: Training and deploying sophisticated models require substantial computational resources.
- **False Positives/Negatives**: No method is perfect; there are always trade-offs between sensitivity and specificity, leading to potential misclassification.

2.3 Research Gap

While substantial progress has been made, several research gaps remain:

- **Multimodal Detection**: Most existing techniques focus on a single modality. There is a need for integrated approaches that can simultaneously handle text, audio, and images.[20]
- **Real-time Detection**: Developing models that can operate in real-time, especially for audio and video streams, remains a significant challenge.[21]
- **Generalizability**: Ensuring that models generalize well across different datasets and contexts is critical. Current models often perform well on specific datasets but struggle with out-of-domain samples.

- **Explainability**: Providing clear, understandable reasons for why a piece of media is classified as AI-generated or human-created is essential for trust and adoption.

By addressing these gaps, this project aims to advance the field of AI-generated media detection, providing more robust, scalable, and explainable solutions across multiple modalities

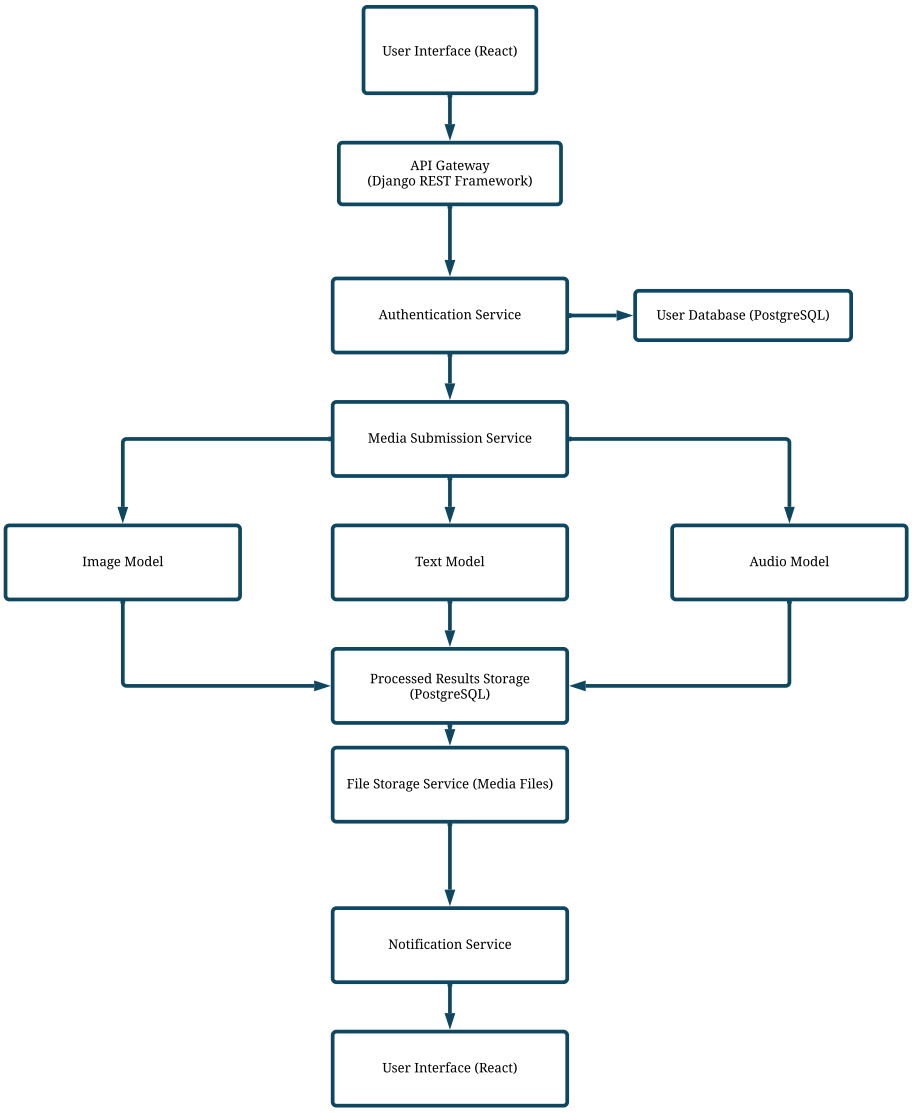
Chapter Three

3. System Design and Analysis

3.1 System Architecture

The system architecture of our AI-generated content detection platform consists of several interconnected components designed to provide a seamless and efficient user experience. The architecture is divided into three main layers: the presentation layer, the application layer, and the data layer.[22]

3.1.1 High-Level System Diagram:



1. System Architecture

- **Presentation Layer:** This layer is responsible for user interactions and consists of the frontend application built using React. It includes user interfaces for authentication, media submission, and result visualization.[23]
- **Application Layer:** The core functionality of the system resides in this layer, implemented using Django. It handles business logic, user authentication, media processing, and interaction with the machine learning models.[24]
- **Data Layer:** This layer manages data storage and retrieval, using PostgreSQL for structured data (user information, media metadata) and a file storage service for media files. It also includes connections to pre-trained machine learning models for AI content detection.

Interaction Between Components:

1. **User Interaction:** Users interact with the system through the frontend, submitting media for analysis and viewing results.
2. **API Requests:** The frontend communicates with the backend via RESTful APIs to perform actions like user authentication, media submission, and fetching detection results.
3. **Media Processing:** Submitted media is processed by the backend, which invokes the appropriate machine learning models to analyse the content.
4. **Results Storage:** Analysis results are stored in the database and made available to users through the frontend interface.

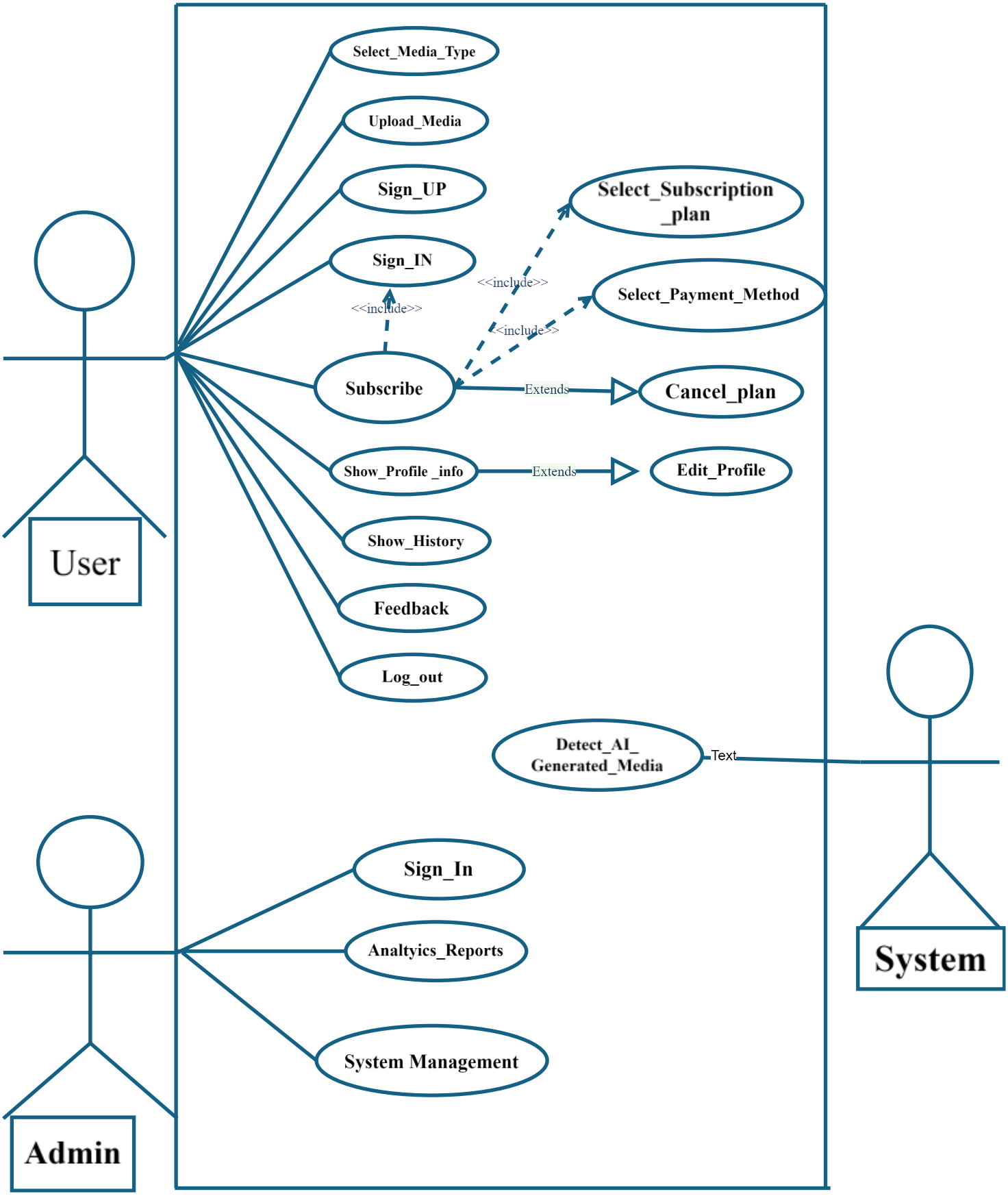
3.2 System Requirements

3.2.1 Functional Requirements

The system offers several key functionalities from a user's perspective:[25]

1. **User Authentication:** Secure login and registration functionalities to manage user access.
2. **Media Submission:** Users can submit images, text, and audio files for AI-generated content detection.
3. **Detection Results:** The system provides detailed reports on the authenticity of the submitted media, indicating whether it is AI-generated or human-created.
4. **Subscription Plans:** Users can subscribe to different plans offering various levels of access and functionality (e.g., number of submissions, advanced analysis features).
5. **Admin Management:** Administrators can manage users, review system usage, and update detection models.

3.2.1.1**Use Case Diagram[26]:**



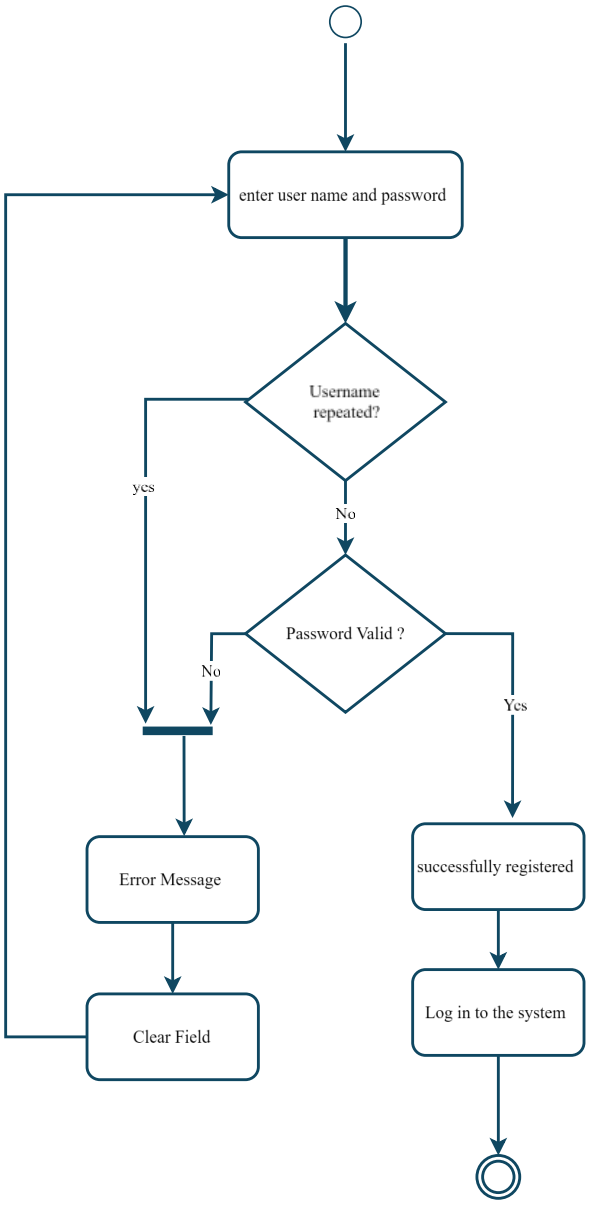
2. Use Case Diagram

Use Case Definitions:

- **User Authentication:** Users can create accounts, log in, and manage their profiles.
- **Media Submission:** Users can upload images, text, and audio for analysis.
- **View Detection Results:** Users can view detailed reports on the submitted media.
- **Subscription Management:** Users can subscribe to different service plans.
- **Admin Management:** Administrators can manage users and system settings.

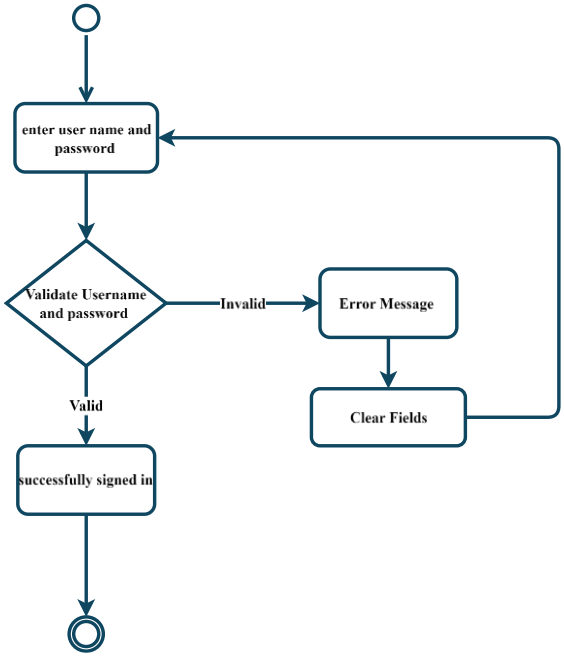
3.2.1.2 **Activity Diagram:**[27]

1. Sign up activity:



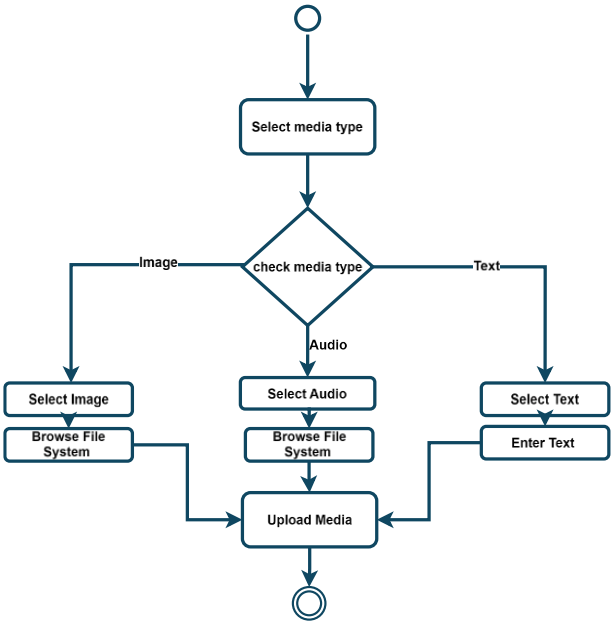
3. Sign up activity

2. Sign in activity:



4. Sign in activity

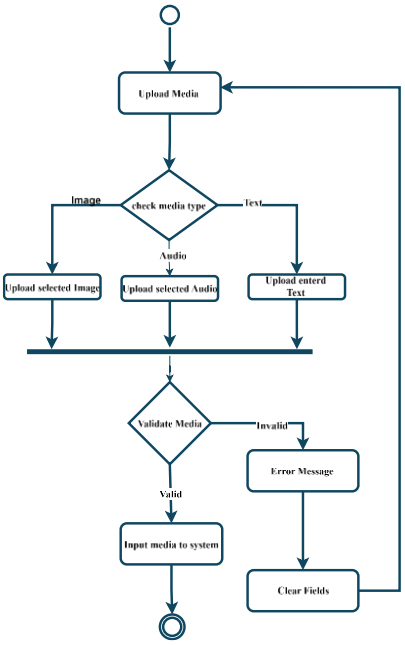
3. Select media type activity:



5. Select media type activity

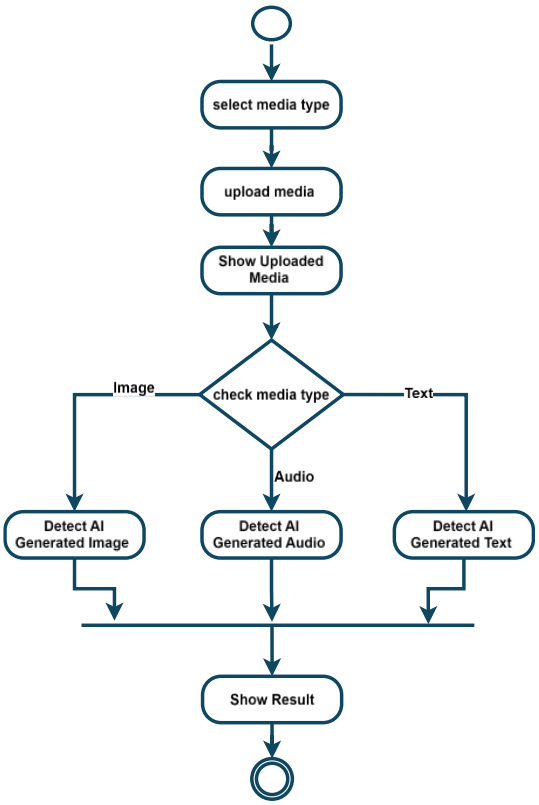
3. System Design and Analysis

4. Upload media activity



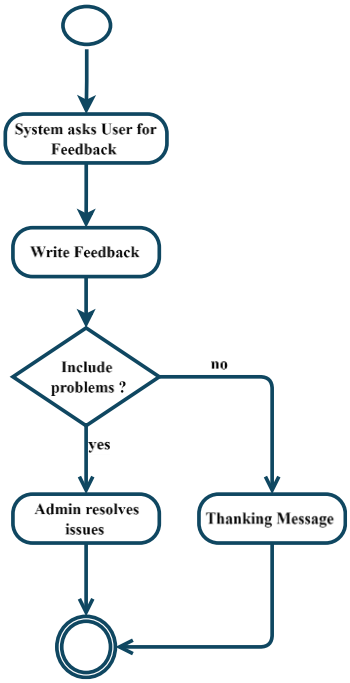
6. Upload media activity

5. Detect AI-generated Media activity:



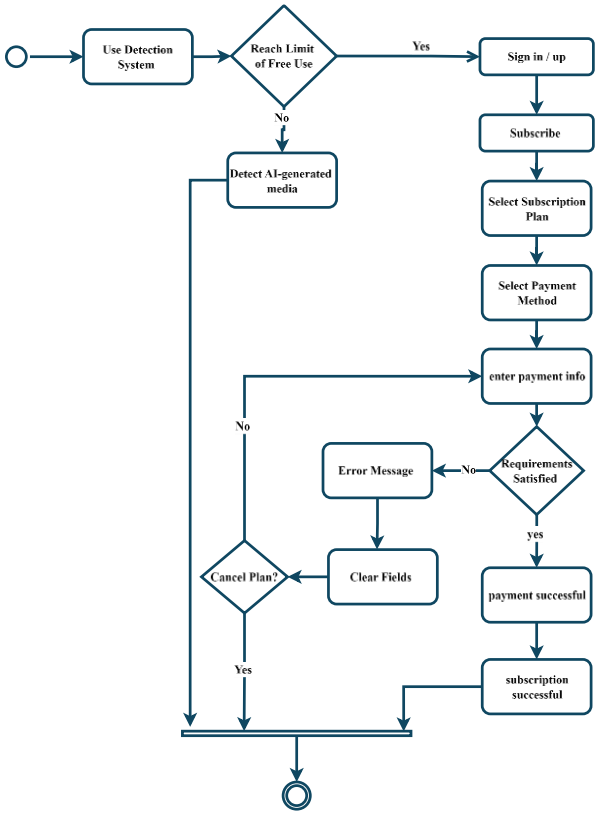
7. Detect AI-generated Media activity

6. Feedback activity:



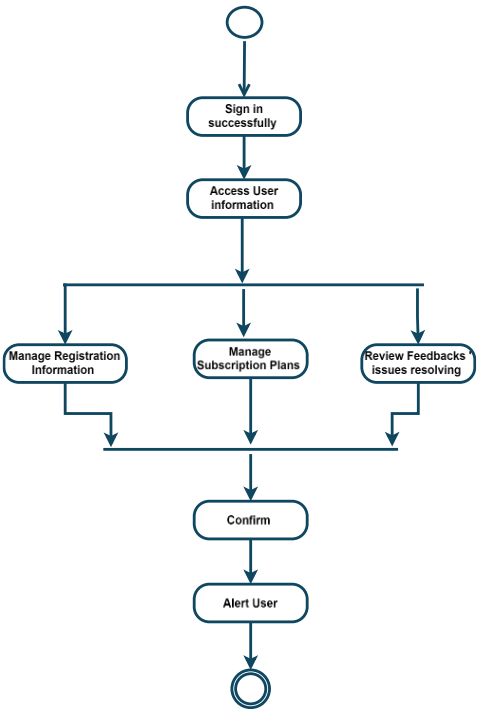
8. Feedback activity

7. Subscription activity:



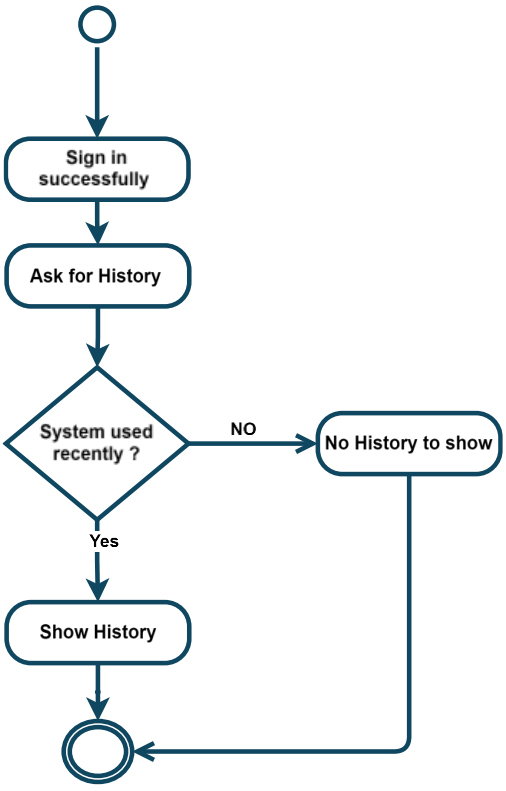
9. Subscription activity

8. Admin Management activity:



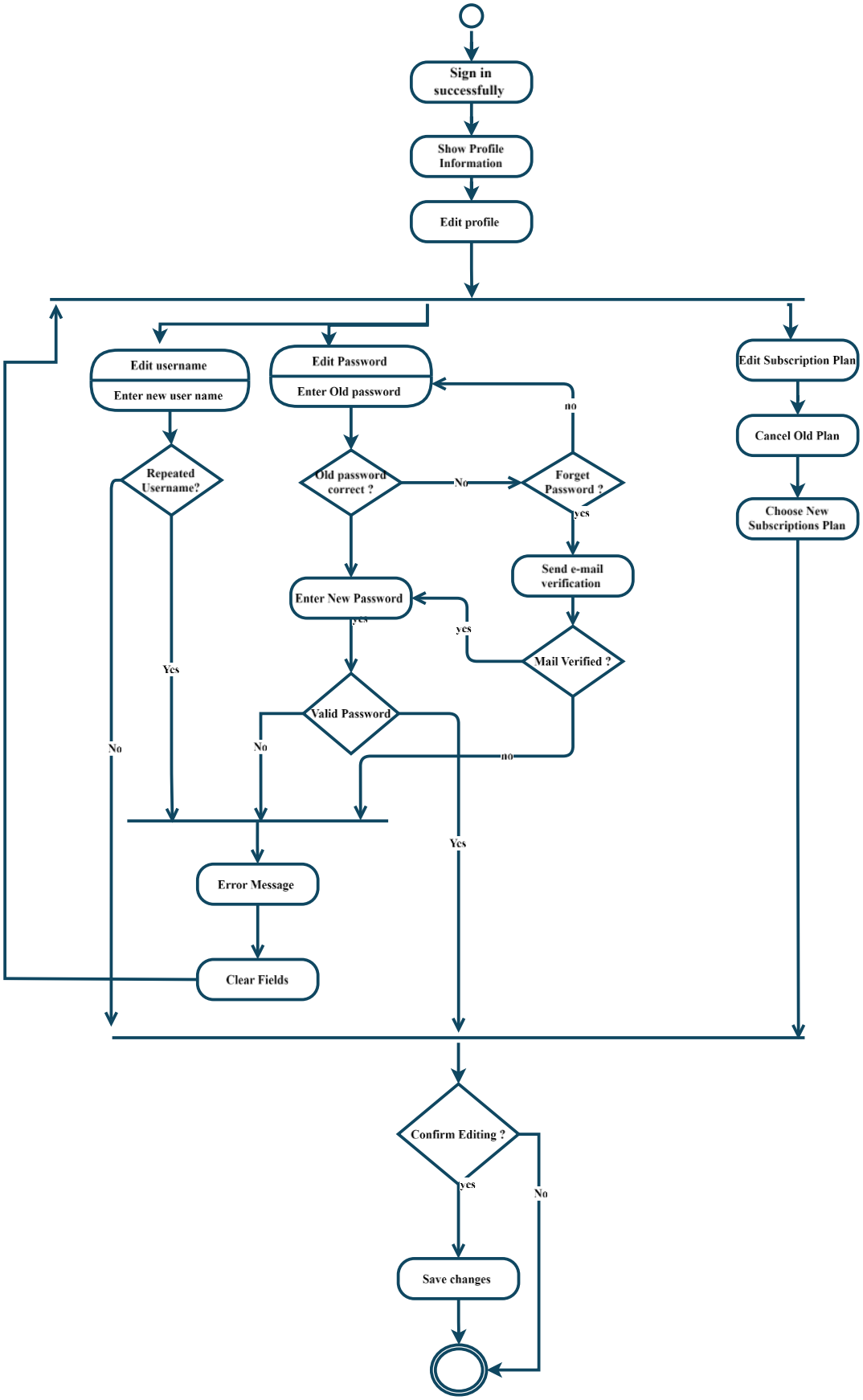
10. Admin Management activity

9. Show History activity:



11. Show History activity

10. Edit Profile activity:



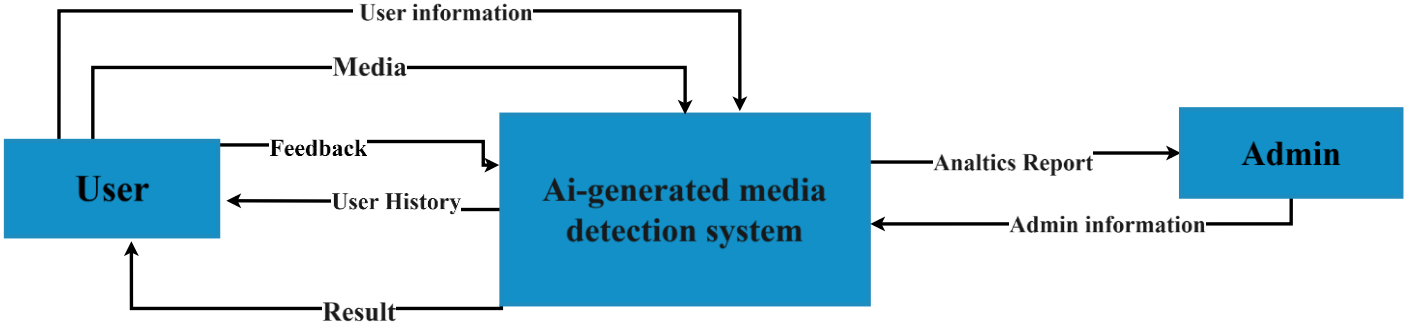
12. Edit Profile activity

3.2.2 Non-Functional Requirements

During the development of the system, several non-functional aspects were considered to ensure a robust and user-friendly application:

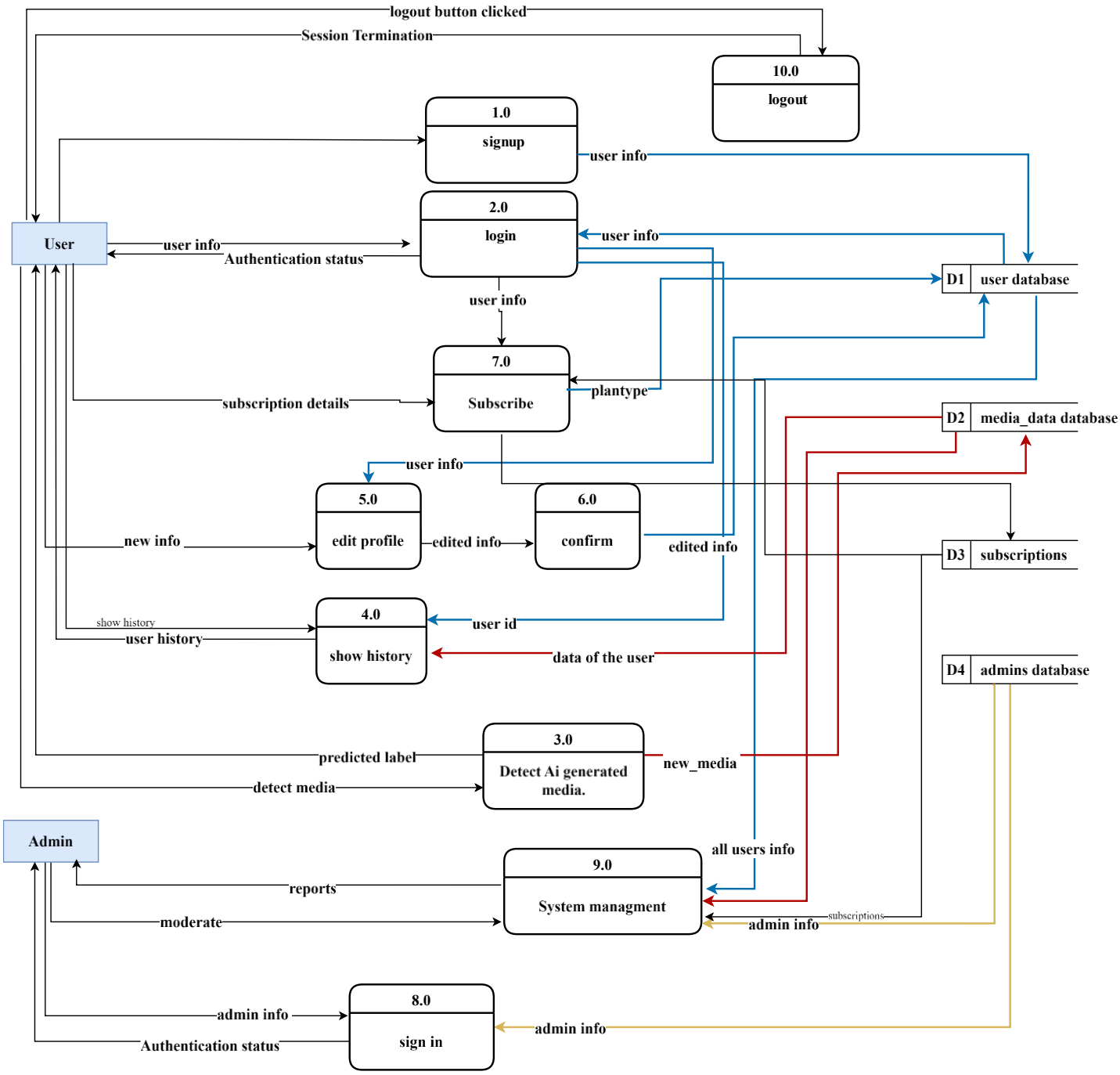
1. **Security:** Ensuring secure user authentication, protecting user data, and preventing unauthorized access to the system.[28]
2. **Usability:** Designing an intuitive and user-friendly interface to facilitate easy interaction with the system.
3. **Performance:** Optimizing the system to handle multiple concurrent users and process large media files efficiently.
4. **Scalability:** Designing the system to scale horizontally, allowing for the addition of more servers to handle increased load.
5. **Reliability:** Ensuring the system is reliable and available with minimal downtime, using strategies like load balancing and regular backups.

Context Diagram:



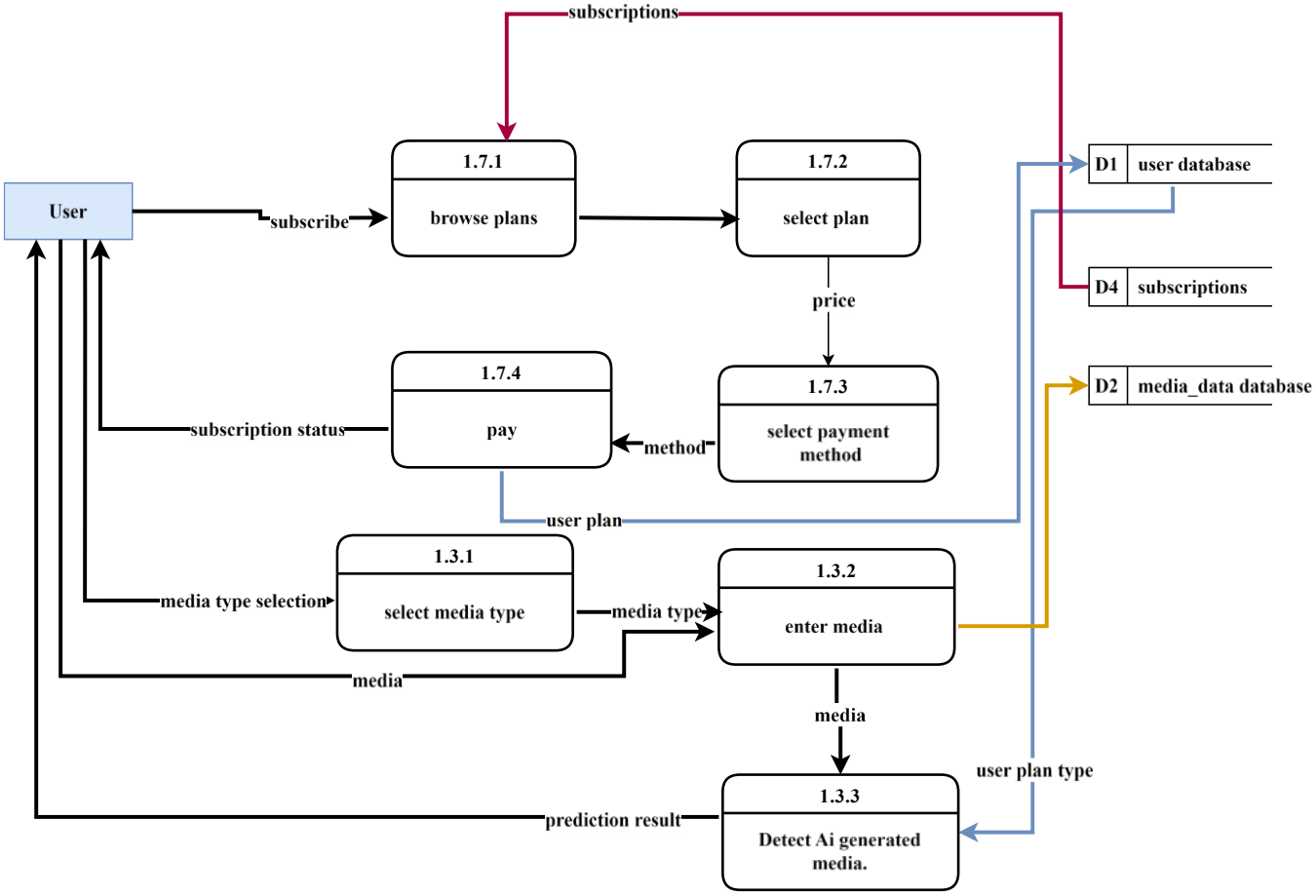
13. Context Diagram

Data Flow Diagram Level 0:



14. Data Flow Diagram L0

Data Flow Diagram Level 1:



15. Data Flow Diagram Level 1

3.2.3 Design Patterns

To enhance the system's design and maintainability, several design patterns were implemented:

1. **Model-View-Controller (MVC):** The MVC pattern was used for the overall structure of the system, separating concerns into three main components: Models (data), Views (UI), and Controllers (business logic). This separation improves maintainability and allows for independent development and testing of each component.
    - *Benefit:* Facilitates a clear separation of concerns, making the system easier to manage and extend.
2. **Multimodal Architecture:** This pattern was used to integrate different modalities (image, text, audio) into the system, allowing it to process and analyse various types of media through a unified interface.
    - *Benefit:* Provides flexibility to handle different types of input data using a cohesive architecture, improving extensibility.
3. **Decorator Pattern:** Implemented for flexible processing pipelines, allowing dynamic addition of processing steps for media analysis without modifying the core logic.

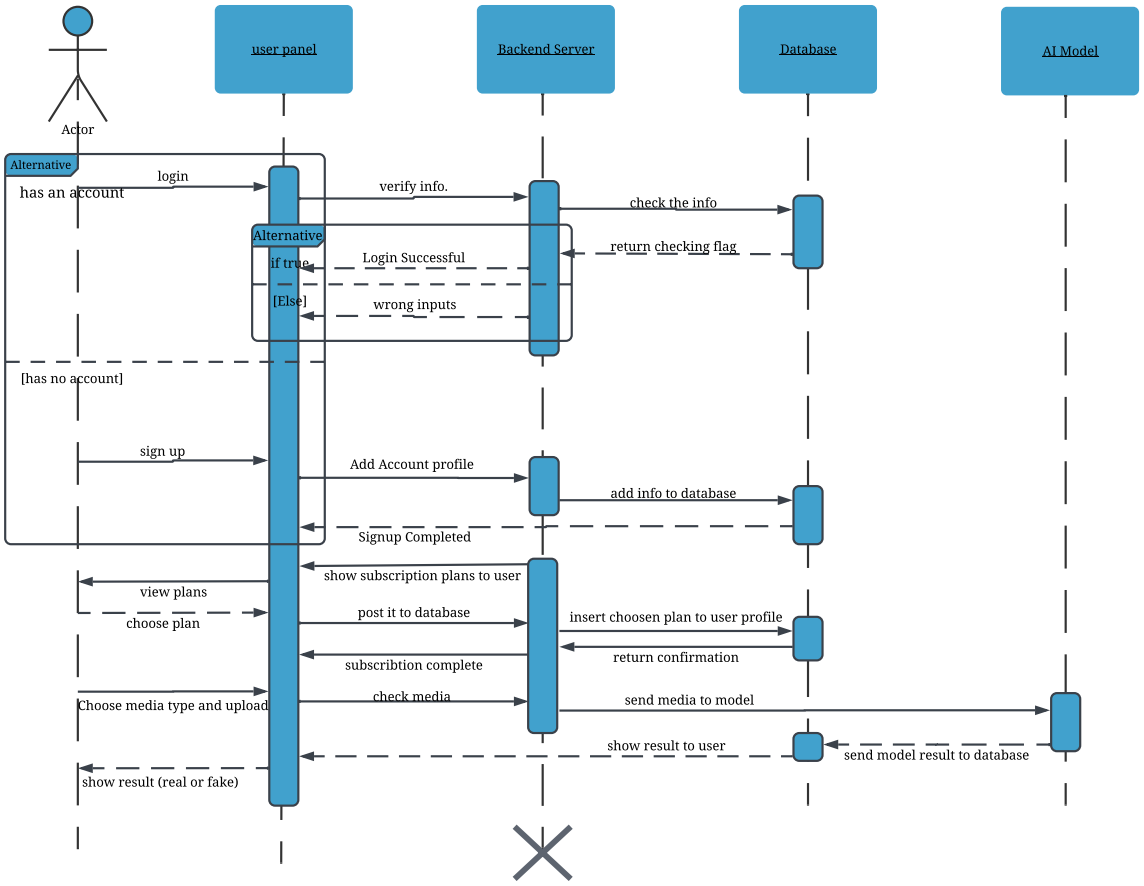
- *Benefit:* Enhances the flexibility and reusability of the media processing pipeline, enabling easy modification and extension of processing steps.

4. **Observer Pattern:** Used for event handling, particularly for real-time notifications and updates, such as informing users about the status of their media analysis.[29]

- *Benefit:* Decouples event handling from core logic, allowing for scalable and maintainable event-driven architecture.

3.2.3.1Sequence Diagram[30]:

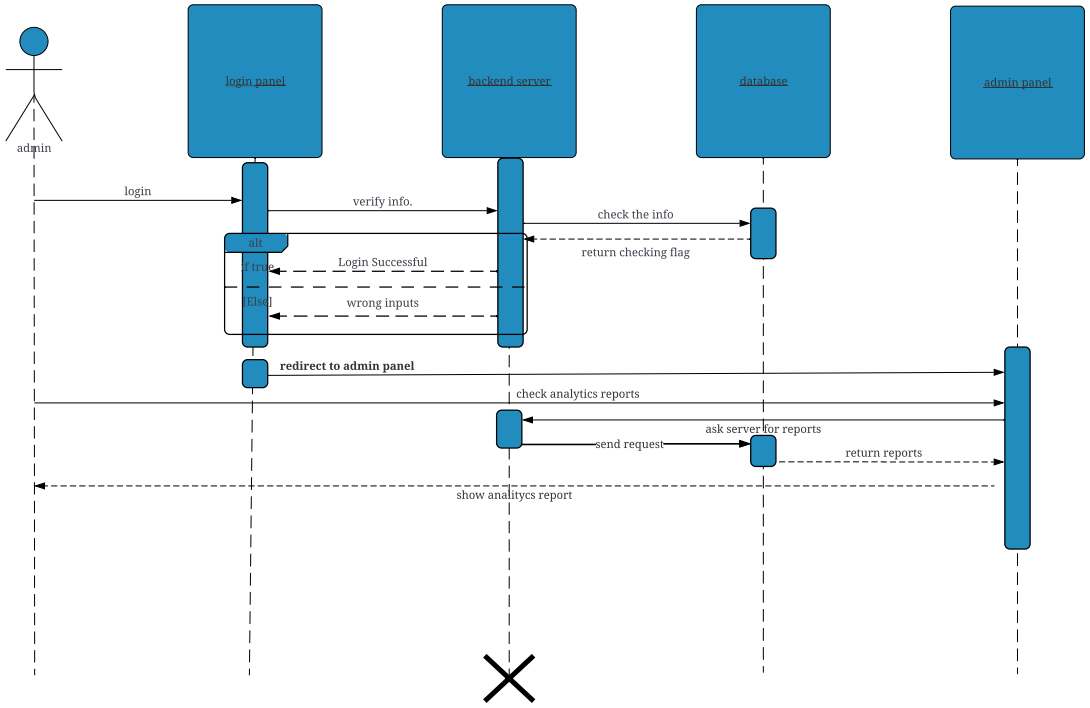
1. User



16. User Sequence Diagram

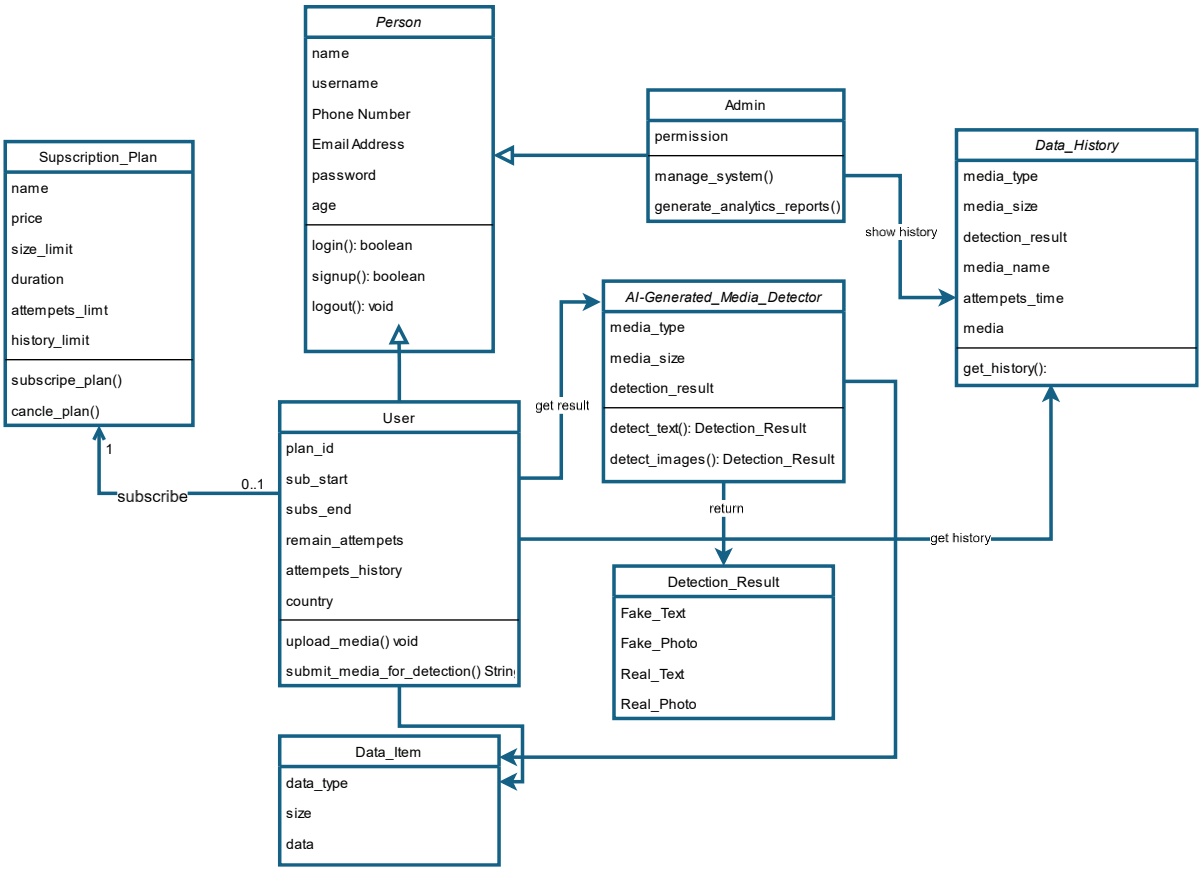
3. System Design and Analysis

2. Administrator



17. Administrator Sequence Diagram

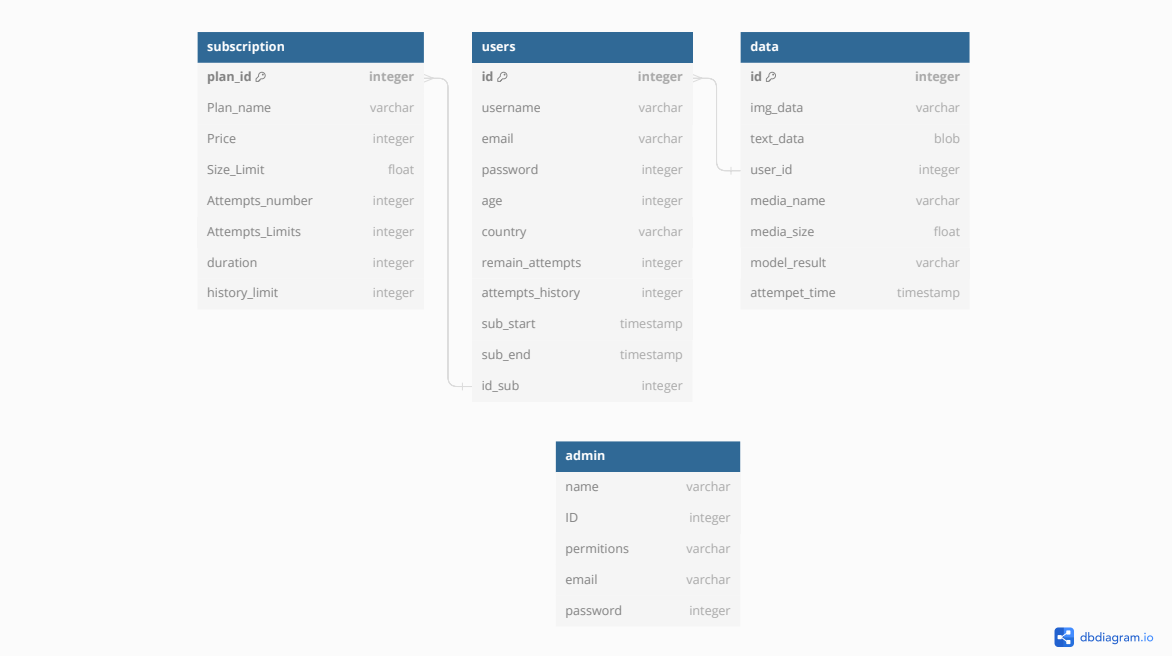
Class Diagram:



18. Class Diagram

3.2.4 Database Design

Database Schema[31]:

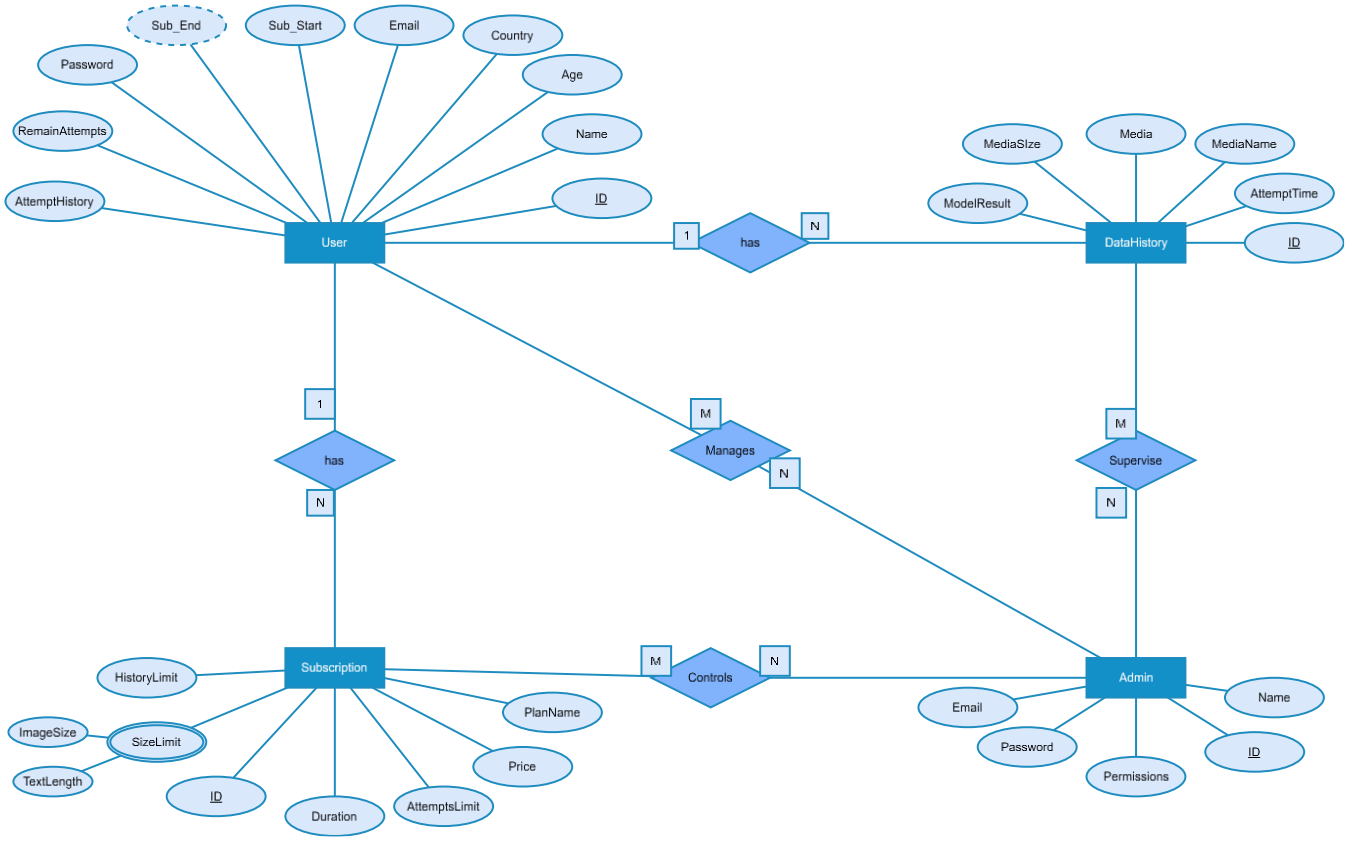


19. Database Schema

- **Users Table:** Stores user information, including authentication details.
- **Media Table:** Stores metadata and file paths for submitted media.
- **Subscription Plans Table:** Stores subscriptions plan specifications.
- **Admins Table:** Stores admins credentials and permissions.

3. System Design and Analysis

ERD Diagram:



20. ERD Diagram

Used Technologies and Tools

- **Frontend:** React, Bootstrap
- **Backend:** Django, Django REST Framework
- **Database:** PostgreSQL
- **Machine Learning:** Hugging Face (RoBERTa, DeBERTa, Wav2Vec2, EfficientNet)

Summary

This chapter outlined the comprehensive system design and analysis for the AI-generated content detection platform. We discussed the overall system architecture, functional and non-functional requirements, and detailed the design patterns used to ensure a robust and maintainable system. Additionally, we included various diagrams to illustrate the system's structure, interactions, and data flow, providing a clear overview of the project's technical foundation.

Chapter Four

4. Project Methodology

4.1 Data Acquisition

Data Sources and Datasets:

Audio:

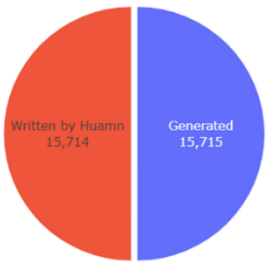
- **Fake-or-Real Dataset (FoR):** A collection of audio files comprising both real and fake audio samples, used to establish baseline detection capabilities.[32]
- **Scenefake Dataset:** Includes a variety of deepfake audio clips generated using different techniques, enhancing the dataset's diversity.[33]
- **In the Wild Dataset:** Contains real and fake audio samples collected from various internet sources, reflecting more natural and diverse scenarios.[34]
- **ASVspoof 2019 Dataset:** Used in Automatic Speaker Verification and Spoofing Countermeasures challenges, featuring a mix of genuine and spoofed audio.[35]
- **ASVspoof 2021 Dataset:** An updated dataset with more recent examples of spoofed audio, capturing advancements in deepfake audio generation.[36]

Image:

- **140k Real and Fake Faces:** Combines 70,000 real faces from the Flickr dataset with 70,000 fake faces generated by StyleGAN, resized to 256px.[37]
- **CelebA-HQ resized (256x256):** Contains 30,000 high-quality celebrity faces, suitable for training and evaluating generative models.[38]
- **Synthetic Faces High Quality (SFHQ) Part 2:** Includes 91,361 high-quality 1024x1024 curated face images, enhanced using StyleGAN2 techniques.[39]
- **Face Dataset Using Stable Diffusion v1.4:** Comprises real faces from the Flickr dataset and fake faces generated by Stable Diffusion models, resized to 256px.[40]
- **Stable Diffusion Face Dataset:** AI-generated human faces using Stable Diffusion 1.5, 2.1, and SDXL 1.0 checkpoints, covering resolutions of 512x512, 768x768, and 1024x1024.[41]
- **Synthetic Faces High Quality (SFHQ) Part 3:** Contains 118,358 high-quality 1024x1024 face images generated by StyleGAN2, utilizing advanced truncation tricks.[42]
- **Synthetic Human Faces for 3D Reconstruction:** Generated by drawing samples from the EG3D model, resulting in high-quality 512x512 synthetic face images.[43]

Text:

- **LLM Generated Essays for the Detect AI Comp:** Contains 700 essays, with 500 generated using GPT-3.5-turbo and 200 with GPT-4.
- **DAIGT Data - Llama 70b and Falcon 180b:**[44]
    - Llama Falcon v3: 7,000 LLM-generated essays.
    - Llama 70b v2: 1,172 LLM-generated essays.
    - Llama 70b v1: 1,172 LLM-generated essays.
    - Falcon 180b v1: 1,055 LLM-generated essays.



21. Text data Distribution

- **Persuade Corpus 2.0:** Comprises over 25,000 argumentative essays by 6th-12th grade students in the U.S.

- **DAIGT External Dataset:** Includes 2,421 student-generated texts and 2,421 AI-generated texts, used to balance the training data with real and AI-generated samples.

22. generated text data EDA

Pre-processing Steps:

Text:

- **No Preprocessing:** To preserve the context and integrity of the data, no cleaning steps were applied to the text datasets. This approach ensures that all nuances and characteristics of the text are retained for model training. [45]

Audio:

- **Noise Reduction:** Background noise was minimized using spectral subtraction and other noise reduction techniques to ensure that the features extracted from the audio were not influenced by unwanted noise.

- **Normalization:** The audio files were normalized to maintain consistent volume levels across all samples, facilitating accurate feature extraction.
- **Sampling Rate Standardization:** All audio files were resampled to a common sampling rate of 16 kHz, ensuring uniform input data for the model.[46]
- **Truncation:** Each audio file was truncated to the first 15 seconds to reduce computational load and ensure consistent input length for the model. [47]

Image:

- **Resizing:** Images were resized to standard dimensions (260x260) to ensure compatibility with the model input requirements.[48]
- **Normalization:** Pixel values were normalized to a common scale to improve model training stability.

4.2 Model Selection

4.2.1 Overview of Initial Model Candidates

We initially evaluated several models for each modality—text, audio, and image detection—based on their reported performance in literature and suitability for our dataset characteristics.

Text:

- **BERT**: Chosen for its foundational architecture and general text classification capabilities.
- **RoBERTa**: Selected for its robustness in handling nuanced language patterns.[49]
- **DeBERTa**: Considered for its advanced attention mechanisms and improved language understanding.[50]

Audio:

- **Wav2Vec2**: Selected for its state-of-the-art performance in speech recognition and anomaly detection.
- **Mel-spectrogram + CNN**: Evaluated for its traditional approach in audio feature extraction.[51]
- **ResNet-based Model**: Considered for its ability to capture detailed spectral features.

Image:

- **EfficientNet**: Chosen for its balance between accuracy and computational efficiency.[52]
- **ResNet**: Evaluated for its strong performance in image classification.[53]
- **Xception**: Considered for its capability to detect fine-grained image features.[54]

4.2.2 Evaluation Criteria

Models were evaluated based on the following criteria:[55]

- **Accuracy**: Overall classification performance.
- **Precision and Recall**: Balance between false positives and false negatives.
- **F1 Score**: Combined measure of precision and recall.
- **ROC-AUC**: Trade-off between true positive and false positive rates.

4.2.3 Experimental Setup

Each model was trained using a standardized dataset split of 70% training, 15% validation, and 15% testing. Hyperparameters were tuned using grid search and random search methods, and models were trained on high-performance GPUs.

4.2.4 Results of Initial Models

Text Models:

- **BERT**: Achieved an accuracy of 85% but exhibited signs of overfitting on smaller datasets.
- **RoBERTa**: Outperformed BERT with an accuracy of 88% and better generalization.
- **DeBERTa**: Achieved the highest accuracy of 90%, showing superior handling of complex text patterns.

Audio Models:

- **Wav2Vec2**: Demonstrated excellent performance with a word error rate of 7% and robust anomaly detection.
- **Mel-spectrogram + CNN**: Achieved reasonable accuracy but was outperformed by Wav2Vec2 in detecting subtle anomalies.
- **ResNet-based Model**: Provided good results but was computationally intensive.

Image Models:

- **EfficientNet**: Balanced accuracy and computational efficiency, with an accuracy of 92%.
- **ResNet**: Achieved an accuracy of 90% but required more computational resources.
- **Xception**: Offered detailed feature extraction but was less efficient compared to EfficientNet.

4.2.5 Refinement and Final Model Selection

Based on the initial evaluations, the following refinements were made:

Text:

- **RoBERTa** and **DeBERTa** showed complementary strengths in language understanding and text classification.
- To leverage their combined advantages, we developed an ensemble model where a final linear layer takes the concatenated outputs of both RoBERTa and DeBERTa models.

Audio:

- **Wav2Vec2** was optimized for its learning rate and batch size to reduce training time while maintaining accuracy.

Image:

- **EfficientNet** was selected for its optimal balance between accuracy and computational efficiency, with additional data augmentation applied to improve generalization.

4.2.6 Final Model

The final models chosen for each modality were:

Text:

Ensemble of RoBERTa and DeBERTa - The outputs of RoBERTa and DeBERTa are concatenated and fed into a final linear layer. This approach was chosen for its superior combined performance, leveraging the strengths of both models.

- **Architecture**:
  - **RoBERTa Output**: Captures robust language patterns.
  - **DeBERTa Output**: Provides nuanced language understanding.
  - **Final Linear Layer**: Integrates the concatenated outputs to enhance overall classification performance.

Audio:

Wav2Vec2 - Chosen for its state-of-the-art performance in audio anomaly detection.

Image:

EfficientNet - Preferred for its efficiency and high accuracy in distinguishing real from AI-generated images.

The ensemble model for text was validated on additional test datasets to confirm its robustness and ability to generalize across various scenarios. This ensemble approach demonstrated significant improvements over individual models, providing a more comprehensive understanding and classification of text inputs.

Chapter Five

5. Implementation

5.1 Introduction

This chapter details the implementation of the AI-Generated Media Detection project. The implementation phase is crucial as it translates the project's conceptual framework into a functional system capable of distinguishing between media content generated by humans and artificial intelligence and makes it a **product** that anyone can use.

5.2 System Architecture

Overview of System Architecture

The AI-Generated Media Detection project adopts a modular architecture, separating frontend and backend components to ensure scalability, maintainability, and flexibility. This architecture facilitates independent development and deployment of each layer while promoting efficient communication through RESTful APIs.

Model or Pattern: Client-Server Architecture with RESTful APIs

The AI-Generated Media Detection project adopts a **Client-Server Architecture** with a clear separation of concerns between the frontend (client) and backend (server) components. This architectural pattern is structured around the following principles:

1. **Client (Frontend):**
   - **Responsibilities:** Handles presentation logic and user interaction, providing a responsive and intuitive interface for users to interact with the application.
   - **Communication:** Communicates with the backend server via HTTP requests using RESTful APIs to fetch data, submit user actions (e.g., authentication, media detection requests), and display results.
2. **Server (Backend):**
   - **Responsibilities:** Implements business logic, manages data persistence, and integrates with external services (such as deep learning models for media detection).
   - **APIs:** Exposes RESTful APIs that the frontend consumes, handling requests for user authentication, media detection, user profile management, and other application functionalities.
   - **Security:** Implements JWT tokens for secure authentication and authorization, ensuring data integrity and privacy.

- **Deep Learning Models:**
  - **Responsibilities:** Hosted separately from the web application in a scalable environment (e.g., cloud infrastructure).
  - **Integration:** Accessed via APIs exposed by the backend, these models perform real-time inference to detect AI-generated media based on user requests initiated through the frontend.

Advantages of Client-Server Architecture with RESTful APIs:

- **Scalability:** Allows each component (frontend, backend, and deep learning models) to scale independently based on demand.
- **Modularity:** Separation of concerns facilitates easier maintenance, updates, and enhancements to specific components without impacting others.
- **Flexibility:** Supports diverse client applications (web, mobile) that can interact with the same backend services through standardized APIs.
- **Security:** Enables secure communication between client and server using token-based authentication and HTTPS protocols.

5.3 Technologies Used:

Languages & Frameworks & Tools

- **Backend:**
  1. python
  2. JWT
  3. Django Framework
  4. SQLite Database
- **Frontend:**
  5. **React.js**
  6. Bootstrap
  7. CSS
- **IDEs & tools:** Vs code & postman, Kaggle, hugging face
- **Dependencies Management**: **NPM** for frontend and **PIP** for backend

5.4 System Components

Now let's get deeper in project, our project contains 2 main parts

1. **Deep Learning models:**
   the heart of the system, the models that classify media to AI-Generated or Human, the models implemented and evaluated on **Kaggle** platform
2. **Catch the AI Website:** our website contains two main parts
   - **Backend:**
   - **Frontend**

5.4.1 Bakend Side

We use **Django** framework, it was selected for its robustness, scalability, and security features, crucial for handling sensitive user data and ensuring efficient communication between frontend and deep learning model APIs.

- **JWT Authentication:** JSON Web Tokens (JWTs) are employed for secure user authentication and authorization:
  Token Generation: Upon successful authentication (login), the backend issues a JWT containing encoded user information and expiration details.

Token Verification: For subsequent requests, the frontend includes the JWT in the Authorization header. The backend verifies the token's authenticity and validity before processing the request.

Session Management: JWTs facilitate stateless session management, reducing server-side storage and enhancing scalability.

- **APIs**

APIs: The backend exposes RESTful APIs using Django REST Framework (DRF), facilitating seamless communication between the frontend and backend components of the application. Key API endpoints include:

Authentication APIs: Handles user:

- **Registration**: to register new user asked to enter its information like name, email, address, password, age. etc and
- Login token issuance using JWT (JSON Web Tokens): here user should enter his credentials **username** or **email** address and **password**, after login the sever return response contains **JWT** token, this token I stored in user browsers local storage to use it every time make a request to the sever
- **Logout**: It removes user session from the sever and delete the local token

Media Detection APIs: Facilitates requests for detecting AI-generated media content, integrating with deep learning model inference endpoints. Here user enter the media (text, image, audio) then preprocess this then send it to be classified in the model and return the results, also it store the media and the model result.

User Profile Management APIs: Allows users to

- **View profile**: returns user data from the database, we use user id that extracted from the JWT token sent along with the user request
- **update profiles**: this API update user data user have full control over his information
- **reset passwords**.
- **Activate email**: when account created the use should activate his email so this API
- **User history: user** history of detections
- **get detected media:** when user click specific instance from history this function get the media from the database and its information.

- **Third-party Services:** Integration of Hugging Face APIs for model inference.
- **Database:** we used SQLite database for store user data and **Django** built-in models DBMS to manage user access and modification of the database
- **Mailing System (smtp):** we used Google SMTP to send emails to users to support account email verification or reset password functionality.

This achieved by using Django core mail backends smtp Email Backend to connect **smtp.gmail.com**

5.4.2 Frontend

The frontend of the AI-Generated Media Detection project is designed as a Single Page Application (SPA) using React.js. This architecture ensures a responsive and dynamic user experience by updating the UI without requiring full page reloads.

Key Pages and Features

1. **Main Page**

   Purpose: Serves as the entry point to the application.

Features: Provides an overview of the application's functionalities, links to login/signup, and introductory content about AI-generated media detection.

2. **Profile Page**
   Purpose: Allows users to view and manage their personal information.
   Features: Displays user details, detection history, and provides options to edit profile information.
3. **Edit Profile Page**
   Purpose: Enables users to update their personal details.
   Features: Form for updating user information such as name, email, and password. Uses API calls to update information in the backend.

4. **Reset Password Page**
   Purpose: Facilitates password recovery for users.
   Features: Form for entering email to receive reset instructions and a subsequent form to set a new password.

5. **Sign In Page**
   Purpose: Authenticates users and grants access to their profiles.
   Features: Login form that accepts email and password, validates credentials via API, and stores JWT tokens for session management.
6. **Sign Up Page**
   Purpose: Registers new users.
   Features: Registration form that collects user information, sends data to the backend for account creation, and handles email verification.
7. **FAQs Page**
   Purpose: Provides answers to common questions about the application.
   Features: List of frequently asked questions and detailed answers to help users understand the system better.
8. **Terms of Use Page**
   Purpose: Outlines the legal terms and conditions for using the application.
   Features: Detailed documentation of user rights, responsibilities, and the legal framework governing the use of the application.

- **User Interface Design**

Design Approach

Tool: Figma was used to design the UI/UX.
Principles: Focused on creating an intuitive, user-friendly interface that emphasizes ease of navigation, responsiveness, and aesthetic appeal.

UI Components

1. **Navigation Bar**: Provides links to main sections such as Home, Profile, Sign In, Sign Up, FAQs, and Terms of Use.
2. **Forms**: Implemented for user authentication, profile management, and media detection requests. Forms are designed to be user-friendly and validated before submission.
3. **Tables and Lists**: Used to display user detection history and other relevant data in an organized manner.

4. **Modals and Alerts**: Provide feedback to users on actions such as form submissions, errors, and success messages.

5.5 Deployment

5.6 Conclusion

This chapter has provided a comprehensive overview of the implementation process for the AI-Generated Media Detection project. It highlighted the methodology, technologies used, challenges overcome, and the significance of testing and deployment phases. The project's successful implementation underscores its potential impact in distinguishing between human-generated and AI-generated media content.

Chapter Six

6. Discussion and Conclusion

References

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, et al., "Language Models are Few-Shot Learners," in *Proc. 34th Adv. Neural Inf. Process. Syst.*, Dec. 2020, pp. 1877-1901. [Online]. Available: https://arxiv.org/abs/2005.14165

[2] D. Crevier, "AI: The Tumultuous Search for Artificial Intelligence," *Basic Books*, 1993.

[3] Y. Wang, Q. She, M. G. Moghaddam, and X. Liu, "DeepFake Detection: A Comprehensive Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9844203

[4] Y. Li, M. Chang, and S. Lyu, "In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Apr. 2018, pp. 3267-3271. [Online]. Available: https://arxiv.org/abs/1806.02877

[5] M. M. Lucas, and M. V. Afanasyev, "Detecting Fake News in Social Media Networks Using Machine Learning," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 3, pp. 544-556, Sep. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8704120

[6] K. K. Singh, V. Nangia, and S. Belkhir, "Deep Learning for AI-Generated Media Detection: A Survey," *IEEE Access*, vol. 9, pp. 123210-123224, Nov. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9641266

[7] J. K. Hwang, "Audio Deepfakes: Analysis, Detection, and Mitigation," in *Proc. 2021 IEEE Int. Conf. Acoust. Speech Signal Process.*, Jun. 2021, pp. 2469-2473. [Online]. Available: https://ieeexplore.ieee.org/document/9413964

[8]R. Verma, A. Thakur, and M. Singh, "An Overview of Data Collection and Preprocessing Techniques for Machine Learning," in *Proc. 2019 IEEE Int. Conf. Big Data*, Dec. 2019, pp. 4157-4161. [Online]. Available: https://ieeexplore.ieee.org/document/898156

[9] Y. Lu, Y. Zhang, and L. Song, "Survey of AI and Privacy Protection," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 4, pp. 2255-2281, Oct. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9449312

[10] J. Daiber, and M. P. Thomas, "AI-Driven Detection Technologies for Law Enforcement: A Survey," *IEEE Secur. Priv.*, vol. 19, no. 3, pp. 67-76, May 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9418891

[11] M. H. N. Tuan, "AI in Academia: Challenges and Opportunities in the Detection of AI-Generated Content," *Comput. Educ. Artif. Intell.*, vol. 2, no. 1, Jun. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666920X21000144

[12] K. Fukushima, and N. Kawaguchi, "Deep Learning Methodologies and Applications: A Review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 1953-1974, Jul. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8954164

[13] A. Van Deursen, and J. Visser, "UML-Based Specification for Interactive Systems: A Review," *Softw. Syst. Model.*, vol. 19, no. 3, pp. 361-380, Mar. 2020. [Online]. Available: https://link.springer.com/article/10.1007/s10270-019-00771-7

[14] L. Downey, "Guidelines for Effective Documentation in Software Engineering," *IEEE Softw.*, vol. 36, no. 5, pp. 43-50, Sep. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8803917

[15] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, et al., "Language Models are Unsupervised Multitask Learners," OpenAI, Feb. 2019. [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[16] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. 2019 Conf. North Am. Chapter Assoc. Comput. Linguist.*, Jun. 2019, pp. 4171-4186. [Online]. Available: https://arxiv.org/abs/1810.04805

[17] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *Proc. 34th Adv. Neural Inf. Process. Syst.*, Dec. 2020, pp. 12449-12460. [Online]. Available: https://arxiv.org/abs/2006.11477

[18] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," *MIT Press*, 2016.

[19] L. Pinto, L. Almeida, and P. Mendes, “Challenges and Opportunities in AI-Generated Content Detection,” *IEEE Access*, vol. 8, pp. 139091-139102, Aug. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9142976

[20] R. Girshick, “Fast R-CNN,” in *Proc. 2015 IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440-1448. [Online]. Available: https://arxiv.org/abs/1504.08083

[21] F. Chollet, “On the Measure of Intelligence,” *arXiv Prepr. arXiv1911.01547*, Nov. 2019. [Online]. Available: https://arxiv.org/abs/1911.01547

[22] T. H. Davenport, and R. Ronanki, “Artificial Intelligence for the Real World,” *Harvard Business Review*, vol. 96, no. 1, pp. 108-116, Jan. 2018. [Online]. Available: https://hbr.org/2018/01/artificial-intelligence-for-the-real-world

[23] J. Spillner, “Patterns and Practices for Software Containers: Architecture, Methodology, Portability,” in *Proc. 2017 IEEE Int. Conf. Software Architecture Workshops*, Apr. 2017, pp. 57-64. [Online]. Available: https://ieeexplore.ieee.org/document/7949164

[24] F. Jiang, L. Chen, and X. Y. Wang, “An Overview of Multimodal Deep Learning in AI Systems,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 7, pp. 1-13, Jul. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9283424

[25] M. Rao, “Effective API Development with Django and Django REST Framework,” *Packt Publishing*, 2020. [Online]. Available: https://www.packtpub.com/product/effective-django-rest-framework/9781800560096

[26] L. Baresi, and L. Pasquale, “A UML Use Case Driven Approach to Requirements Engineering,” in *Proc. 2006 IEEE Int. Conf. on Software Engineering and Knowledge Engineering*, Jul. 2006, pp. 423-430. [Online]. Available: https://ieeexplore.ieee.org/document/1627604

[27] M. D. P. Papazoglou, and W. J. Van Den Heuvel, “Service-Oriented Design and Development Methodology,” J. Database Manage., vol. 16, no. 4, pp. 45-70, Oct. 2005. [Online]. Available: https://www.igi-global.com/article/service-oriented-design-development-methodology/3331

[28] N. Leavitt, “Security in the Cloud: Protecting Data and Applications,” *IEEE Cloud Comput.*, vol. 3, no. 2, pp. 44-50, Mar. 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7460194

[29] P. Zave, “Multimodal Systems: Integration of Different Modalities in System Design,” *IEEE Softw.*, vol. 26, no. 2, pp. 63-70, Mar. 2009. [Online]. Available: https://ieeexplore.ieee.org/document/4814280

[30] M. Fowler, “UML Distilled: A Brief Guide to the Standard Object Modeling Language,” 3rd ed., *Addison-Wesley*, 2003. [Online]. Available: https://www.oreilly.com/library/view/uml-distilled/0321193687/

[31] M. Bender, “SQL & NoSQL Databases: Models, Languages, Consistency Options, and Architectures for Big Data Management,” *Springer*, 2018. [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-61837-5

[32] T. B. Wen, D. Bogdanov, and X. Serra, "An Audio-Visual Dataset for Fake and Real Speech Detection," *arXiv preprint arXiv:1905.04515*, 2019. [Online]. Available: https://arxiv.org/abs/1905.04515

[33] G. Goswami, "Scenefake: A Dataset of Deepfake Audio Clips," *Proc. 2021 IEEE Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7528-7532, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9414200

[34] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," Proc. 31st AAAI Conf. Artificial Intelligence (AAAI-17), pp. 4278-4284, 2017. [Online]. Available: https://arxiv.org/abs/1602.07261

[35] T. Kinnunen, et al., "ASVspoof 2019: Automatic Speaker Verification Spoofing and Countermeasures Challenge Evaluation Plan," *arXiv preprint arXiv:1904.10320*, 2019. [Online]. Available: https://arxiv.org/abs/1904.10320

[36] H. Delgado, et al., "ASVspoof 2021: Automatic Speaker Verification Spoofing and Countermeasures Challenge," *Proc. Interspeech 2021*, pp. 4319-4323, 2021. [Online]. Available: https://www.isca-speech.org/archive/pdfs/interspeech_2021/1112.pdf

[37] S. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," arXiv preprint arXiv:1710.10196, 2018. [Online]. Available: https://arxiv.org/abs/1710.10196

[38] Z. Liu, et al., "Deep Learning Face Attributes in the Wild," Proc. 2015 IEEE Int. Conf. Computer Vision (ICCV), pp. 3730-3738, 2015. [Online]. Available: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[39] S. Karras, et al., "Analyzing and Improving the Image Quality of StyleGAN," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 8107-8116, 2020. [Online]. Available: https://arxiv.org/abs/1912.04958

[40] R. Rombach, et al., "High-Resolution Image Synthesis with Latent Diffusion Models," *arXiv preprint arXiv:2112.10752*, 2021. [Online]. Available: https://arxiv.org/abs/2112.10752

[41] L. von Platen, et al., "Diffusion Models: A Comprehensive Survey of Methods and Applications," *arXiv preprint arXiv:2301.09675*, 2023. [Online]. Available: https://arxiv.org/abs/2301.09675

[42] T. A. Salimans, et al., "Improved Techniques for Training GANs," *Proc. 2016 Conf. Neural Information Processing Systems (NeurIPS)*, pp. 2234-2242, 2016. [Online]. Available: https://arxiv.org/abs/1606.03498

[43] E. Chan, et al., "Efficient Geometry-aware 3D Face Generation with Multi-Scale Learning," *arXiv preprint arXiv:2303.07247*, 2023. [Online]. Available: https://arxiv.org/abs/2303.07247

[44] A. T. Liu, et al., "Large Language Models: A Review of Capabilities, Applications, and Challenges," *arXiv preprint arXiv:2403.03201*, 2024. [Online]. Available: https://arxiv.org/abs/2403.03201

[45] C. Potthast, et al., "No preprocessing? No problem! Text categorization using raw text," *Proc. 2018 ACM Conf. Research and Development in Information Retrieval (SIGIR)*, pp. 1503-1506, 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3209978.3210181

[46] T. Pollet, and H. Steeneken, "Speech Signal Sampling Rate: Impact on Recognition Accuracy," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 8, pp. 650-656, 2002. [Online]. Available: https://ieeexplore.ieee.org/document/1168392

[47] J. Barker, et al., "Truncation in Speech Processing: Benefits and Trade-offs," *IEEE Signal Process. Lett.*, vol. 12, no. 6, pp. 492-495, 2005. [Online]. Available: https://ieeexplore.ieee.org/document/1457608

[48] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Proc. 2012 Conf. Neural Information Processing Systems (NeurIPS)*, pp. 1097-1105, 2012. [Online]. Available: https://dl.acm.org/doi/10.5555/2999134.2999257

[49] Y. Liu, et al., "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[50] P. He, et al., "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," *Proc. 2021 Int. Conf. Learning Representations (ICLR)*, pp. 1-10, 2021. [Online]. Available: https://arxiv.org/abs/2006.03654

[51] A. Baevski, et al., "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," *Proc. 2020 Conf. Neural Information Processing Systems (NeurIPS)*, pp. 12449-12460, 2020. [Online]. Available: https://arxiv.org/abs/2006.11477

[52] M. Tan, and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *Proc. 2019 Int. Conf. Machine Learning (ICML)*, pp. 6105-6114, 2019. [Online]. Available: https://arxiv.org/abs/1905.11946

[53] K. He, et al., "Deep Residual Learning for Image Recognition," *Proc. 2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016. [Online]. Available: https://arxiv.org/abs/1512.03385

[54] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1251-1258, 2017. [Online]. Available: https://arxiv.org/abs/1610.02357

[55] D. D. Lewis, "Evaluating and Optimizing Autonomous Learning Algorithms," *Machine Learning Review*, vol. 24, no. 4, pp. 411-437, 2007. [Online]. Available: https://link.springer.com/article/10.1007/s10994-007-5038-3

APPENDICES

Appendices