National University of Computer and Emerging Sciences, Lahore

Flora: AI-Powered Platform

Zaid Asif 21L-5179 BS(CS)

Muhammad Suhaib Rashid 21L-1787 BS(CS)

Zaid Bin Zahoor 21L-1807 BS(CS)

Supervisor: Dr. Arshad Ali

Co-Supervisor: Dr. Maryam Bashir

Final Year Project

April 16, 2025

Anti-Plagiarism Declaration

This is to declare that the above publication was produced under the:

Title: Flora: AI-Powered Platform

is the sole contribution of the author(s), and no part hereof has been reproduced as it is the basis (cut and paste) that can be considered Plagiarism. All referenced parts have been used to argue the idea and cited properly. I/We will be responsible and liable for any consequence if a violation of this declaration is determined.

Date: April 16, 2025

Name: Zaid Asif

Signature: 

Name: Muhammad Suhaib Rashid

Signature: 

Name: Zaid Bin Zahoor

Signature: 

Author's Declaration

This states Authors' declaration that the work presented in the report is their own, and has not been submitted/presented previously to any other institution or organization.

Abstract

In an increasingly digital world, where humans as social beings seek meaningful connections, existing platforms often fall short in providing secure, real-time, interest-based interactions. Flora, a web app, addresses this gap by facilitating one-to-one video chats between users who share common interests, matched through an NLP-based system. To ensure user safety, the platform integrates real-time audio moderation. Built on a hybrid architecture with a React frontend and Django backend, Flora also emphasizes developing an effective audio moderation system using NLP and speech processing techniques to detect toxicity in Urdu-English mixed-language communication as part of its research and development efforts.

Executive Summary

Humans are inherently social beings who seek meaningful connections. While social media platforms offer global communication, the concept of a professional, interest-based platform for one-on-one video interactions with random individuals is still underdeveloped. Current platforms often lack moderation and expose users to harmful content such as hate speech, verbal abuse, and inappropriate language. This gap highlights the need for a secure platform that prioritizes professionalism and protects users during these exchanges, ensuring a safe and respectful environment.

Flora is a web-based platform that facilitates secure, real-time, interest-based interactions through one-on-one video chats, with random users matched based on shared interests using an NLP-based system. Built with a React frontend and a Django backend, Flora leverages technologies such as WebRTC, WebSockets, PostgreSQL, Redis, and Celery to ensure an efficient and seamless user experience. As part of our ongoing research, the platform integrates real-time audio moderation to create a safe environment by detecting toxicity in mixed-language communication (Urdu-English).

The platform includes key features such as user matching, video chatting with text chat functionality, and real-time audio detection to moderate toxicity during conversations. These features aim to foster secure and meaningful digital interactions by maintaining a safe environment.

Current research in toxicity detection largely relies on text-based methods, but real-time speech presents challenges due to noisy audio environments. To address this, Flora combines both audio and text modalities to improve toxicity detection. The team has trained models such as XLM-RoBERTa, mBERT, and Wav2Vec, with early results showing that adding audio enhances classification accuracy.

By integrating audio and text encodings, Flora is proving to be more effective than relying on text classifiers alone. This approach aims to improve the reliability of toxicity detection in real-time, mixed-language conversations. Flora's innovative use of technologies like NLP, ASR, and speech processing positions it to create safer and more meaningful digital interactions.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

Humans are social beings who like to communicate. Social media enables communication with anyone globally, but the concept of a professional and secure platform for one-to-one communication with random individuals based on shared interests is still relatively new. Flora is a web-based application that randomly connects users interested in specific domains and shared interests, allowing them to engage in one-to-one communication via video calls. We aim to create a secure and professional platform where users will not encounter hate speech, verbal abuse, profanity, or pornographic language.

1.1 Purpose of this Document

The purpose of this report is to document the design, development, and evaluation of Flora, a professional one-to-one random communication platform. Specifically, this document will outline Flora's architecture and its core features, including user profile analysis, matching based on interest categories, and real-time moderation through NLP and speech processing. Social media enables communication with anyone globally, but the concept of a professional and secure platform for one-to-one communication with random individuals based on shared interests is still relatively new.

The purpose of the research is the classification of harmful speech in Urdu-English code-mixed settings. The goal of this research is to develop a system capable of detecting verbal abuse, hate speech, profanity, and pornographic language that could potentially harm users on our platform.

1.2 Intended Audience

The intended audience for this project includes:

- The development team, who may need to understand the project better to make improvements.
- Users of our platform, who might want to know how a professional platform allows them to engage with random people while protecting them from hate speech and verbal abuse.
- The supervisor of Flora and the evaluation committee of the Final Year Project.

1.3 Definitions, Acronyms, and Abbreviations

List all important definitions, acronyms, and abbreviations used in this document. For example: **SDG**: Sustainable Development Goal

FYP: Final Year Project

MVP: Minimum Viable Product

UI: User Interface

UX: User Experience

Agile: Agile Development

SCRUM: Scrum Development

REST: Representational State Transfer

ORM: Object-Relational Mapping

CRUD: Create, Read, Update, Delete

NLP: Natural Language Processing

API: Application Programming Interface

AI: Artificial Intelligence

CRUD: Create, Read, Update, Delete

GUI: Graphical User Interface

1.4 Conclusion

To conclude, this document will explain the social nature of humans, how Flora addresses this in a secure manner, and provide a detailed breakdown of how the platform functions. It also includes an overview of Flora's architecture, features, software requirements, and implementation details.

Chapter 2 Project Vision

In this chapter we'll discuss the vision we have for flora. Why we plan to develop an interest-based random one-to-one communication platform. We'll elaborate on the project scope, as well as both technical and business goals. We'll also discuss why moderation and security is such a major part of this project

2.1 Problem Domain Overview

We can divide our problem domain into two categories. The first one is the need for a specific group of people to communicate with others to share and discuss there ideas and interests. And the other one is the issue of security, which is an important part when you are communicating with someone random live on the internet

2.2 Problem Statement

Current random one-to-one video communication platforms lack interest-based interactions and expose users to hate speech, verbal abuse, profanity, and pornographic language due to insufficient real-time moderation. This creates a need for a secure platform that ensures professionalism and protects users during these exchanges.

2.3 Problem Elaboration

Let's break this into two parts: Communication and Security.

2.3.1 Communication

People like to communicate, both introverts and extroverts. Some prefer to communicate in public gatherings, while others prefer private interactions. Some are very expressive with their ideas and just want someone with whom they can discuss them, while others only discuss them with friends or trusted individuals. There are various topics where people feel enthusiastic about expressing themselves and communicating, such as startups, politics, sports, and research.

All of these needs are well handled in the offline world. People join communities and meet like-minded individuals or make friends at their school, university, workplace, or neighborhood with whom they can discuss these topics. However, in the online world, the concept of interest-based communication is still limited, mostly to text. For example, we have Reddit, where people join communities and share their

thoughts. Then we have platforms like YouTube, where ideas can be shared, but the element of one-to-one communication is missing. Additionally, users must build their own audience. There are platforms that allow for one-to-one communication via video calls, but they are mostly used for fun and casual talks rather than interest-based connections.

2.3.2 Security

Next comes the concept of security. On text-based communities like Reddit, where people share their thoughts in writing, they can receive negative responses that include hate speech, abuse, profanity, or pornographic language. However, it is easier to control these issues using keyword filtering, and text-based hate can often be ignored. The situation is very different when communicating live with someone on a video call.

Direct verbal abuse, hate speech, or manipulation can have a much more damaging effect. Additionally, it is difficult to detect such behavior because it is audio-based, and people can also speak in a mixed language (e.g., English and Urdu), making it much harder to resolve this issue using speech processing.

2.4 Goals and Objectives

2.4.1 Goals

- Create a communication space where users can connect over common interests, leading to meaningful conversations in a safe environment.
- Develop and use advanced technologies to monitor and filter out inappropriate speech during video calls.
- Perform both of the above-mentioned tasks in such a way that the video calling process remains smooth.

2.4.2 Objectives

- Develop a system that analyzes user profiles, the categories they want to discuss, and matches them with similar users.
- Implement techniques to analyze spoken language in real-time and monitor user speech during video interactions.
- Create easy ways for users to report inappropriate behavior, keeping the community safe.

2.5 Project Scope

The scope of this project is to design and develop a web-based application for users to video-chat with random user based on similar interests. The application will allow users to:

- Create their profile on Flora
- Update their profile
- Select specific categories (e.g., Startups, Sports, IT) for connecting with others.
- Ensure real-time content moderation using advanced speech and text analysis technologies.
- Participate in video calls with randomly matched users who share common interests.
- Report inappropriate behavior during video interactions for immediate action.

The project will be built using HTML, CSS, Javascript, React, Redux toolkit for the frontend, while Django will be used for backend development, along with celery, nginx, redis and PostgreSQL as the database. To moderate conversations, technologies like Natural Language Processing (NLP) and speech recognition will be integrated. WebRTC or its alternative like Zoom API, Jitsi etc. will be utilized for real-time video communication, and the backend will manage user data and profile matching through RESTful APIs. The project will follow Agile development methodologies, with SCRUM used for project management.

The project deliverables will include:

- A functional web-based application
- A user manual for the application
- A documentation of the design, implementation, and testing process

The project scope does not include the following:

- Providing hosting or server infrastructure for the application
- Developing mobile applications (iOS or Android)

This project scope will be used as a guide throughout the development

2.6 Sustainable Development Goal (SDG)

2.6.1 SDG 9: Industry, Innovation, and Infrastructure

Our project Flora aligns with SDG 9 the most. Interest-based one-to-one random communication and creating a system that can detect verbal abuse, profanity, pornographic language, and hate speech in a mix of Urdu and English is the innovation we are focusing on.

Figure 2.1: Sustainable Development Goal: Industry Innovation and Infrastructure

2.6.2 SDG 10: Reduced Inequalities

The project Flora is not limited to any specific group of people or country. It will be available to people around the globe. One user can connect with another person with similar interests, even if they are from a different race or country. This diversity in users will lead to diversity in discussions, ideas, and knowledge.

Figure 2.2: Sustainable Development Goal: Reduced Inequalities

2.7 Constraints

Flora's development will face several constraints, including a restricted budget and challenges in developing real-time audio moderation technology. Creating such technology and its potential deployment is a complex task, especially when integrating a strong system for audio moderation based on NLP and speech recognition. Additionally, we need to ensure that the video call experience remains lag-free.

2.8 Business Opportunity

Flora will host a diverse and professional audience from all over the world, creating multiple business opportunities. Regarding the revenue model, various approaches can be implemented. We can introduce paid categories, where niche topics like ”VC Funding” are made paid, while broader categories like ”Tech” and ”Sports” remain free. Additionally, we can charge for connection filters. For example, if a user wants to communicate with people from a specific region, they would pay a fee.

2.9 Stakeholders Description/ User Characteristics

Flora is a platform open to anyone, where individuals can discuss topics of their interest with others.

2.9.1 Stakeholders Summary

Flora’s stakeholders include a diverse group of users. The primary users of Flora will be individuals who want to engage in interest-based, one-to-one conversations.

- **Professionals**: Experts in specific fields who would like to communicate about their expertise with others, such as researchers.
- **Casual Talkers**: Individuals who would like to speak with people sharing similar interests. While conversations are topic-based, they don’t have to be strictly for knowledge-sharing purposes.
- **Developers**: Responsible for developing and maintaining Flora over time.

2.9.2 Key High-Level Goals and Problems of Stakeholders

2.9.2.1 Primary Users

- **Goal:** Find like-minded individuals for meaningful, secure conversations.
- **Problem:** Difficulty finding a platform for professional communication online.

2.9.2.2 Developers

- **Goal:** Develop the platform along with the audio moderation system.
- **Problem:** Managing the technological challenges of integrating NLP, speech recognition, and real-time video systems while ensuring the video call remains lag-free.

Chapter 3 Literature Review / Related Work

3.1 Definitions, Acronyms, and Abbreviations

Seamless-Align : It is a dataset that was curated based on metadata mined for Speech to Speech, Text to Speech and Speech to text release by Meta AI.

Detoxify : An open source library built by Pytorch Lightening to detect toxic comments and is trained on Jigsaw's Kaggle Dataset. It covers 7 languages.

ETOX : It is a toxicity detection tool based on word lists and covers 200 languages.

ViT : Vision Transformer

MFCC : Mel-Frequency Cepstral Coefficients

ViViT : Video Vision Transformer

Zero-Shot : It is the ability of model to do tasks it has never seen before without giving any examples.

Few-Shot : It is the ability of model to do tasks it has never seen before based on a few examples.

GPT-4V : Variant of GPT-4 that has visual capabilities i.e. it can process both images and text.

Qwen-VL : A large vision-language model designed for tasks that require understanding both image and text.

Hing-GPT : Hindi-English code-mixed GPT2 model trained on L3Cube-HingCorpus.

Hing-Roberta : Hindi-English code-mixed XLM-RoBERTa model trained on L3Cube-HingCorpus.

VideoMAE : Video based model that uses Masked Auto-Encoder Approach i.e. understanding videos by masking parts and predicting them.

TimesFormer : Transformer based model for understanding temporal aspect of videos.

IEMOCAP : Interactive Emotional Dyadic Motion Capture, an audio-visual dataset for emotion recognition.

ZCR : Zero Crossing Rate

TF-IDF : Term Frequency-Inverse Document Frequency

Albert : A Lite BERT. Smaller and more efficient version of BERT.

HSDVD : Hate Speech Detection Video Dataset

WAVLM : A transformer based model for speech processing that is optimized for raw audio and used in ASR.

POS : Part of Speech

NER : Named Entity Recognition

WER : Word Error Rate

MMS : Massively Multilingual Speech, a research initiative by Meta to advance speech technologies among vast number of langauges and dialects.

ASR : Automatic Speech Recognition

3.2 Detailed Literature Review

This section covers literature published in the domain of Multi-Modal Toxicity Detection. Toxicity can be defined as any type of hate speech, abuse and profanity. Although our focus is on Urdu, this section mostly covers literature published for other relatively high-resource languages due to the lack of research in toxicity detection in Urdu especially for modalities other than text.

3.2.1 Development of a Multilingual Dataset for Audio-based Toxicity Detection and Zero-shot Analysis [1]

3.2.1.1 Summary of the research item

The authors collected multilingual voice data from the Mozilla Common Voice initiative and the Seamless-Align dataset, employing the ”Detoxify” and ETOX toxicity classifiers for English and Urdu, respectively. They curated a dataset of 4000 sentences, including 2500 hate sentences in Urdu, by applying toxicity classification techniques. Annotators were guided to determine if utterances contained toxicity (hatespeech, profanity, pornographic language or abuse) and to specify the time stamps leading to the decision.

However, the Urdu segment mainly consists of audio-based translations from religious texts and includes advertisements, raising concerns about its applicability for hate speech detection.

3.2.1.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The paper presents a huge dataset containing 4,000 sentences including 2,500 hate sentences in Urdu, and uses established classifiers like ETOX and Detoxify, with clear annotation guidelines to bring an improvement in identifying toxic content.
- However, the Urdu portion primarily consists of translations from religious texts like Quran and Bible, which may not capture all kinds of hate speech effectively. Additionally, the inclusion of this kind of content could introduce biases, and the lack of clearly defined Detoxify scores or ETOX tokens for Urdu raises questions about selection criteria. Inconsistent labeling, allowing for multiple toxicity types, could complicate the identification of specific toxic expressions. The dataset features Indian (Hindi) advertisements that are not desirable when working for Urdu

specific tasks.

3.2.1.3 Relationship to the proposed research work

This research is relevant to our work as it highlights the challenges in collecting and labeling data for toxicity detection in Urdu. It underscores the need for datasets that is unbiased and that incorporates code-switching which aligns with our focus on multimodal abuse detection in Urdu-English code-mixed conversations.

3.2.2 A Multimodal Dataset for Classifying Hate in Video Content [2]

3.2.2.1 Summary of the research item

This research paper is about multimodal hate speech detection in videos and considers a combination of text, audio, and video modalities to classify content as hateful. The authors curated one of the largest known datasets of hateful videos consisting of 1083 videos spanning approximately 43 hours and about 144,000 frames. Each video was annotated as either hateful or non-hateful with time stamps leading to the decision. The study uses different detection models, including BERT for text processing, MFCC for audio analysis, and ViT for video analysis. The best fusion model which combines all three modalities, achieves a macro F1-score of 0.790 with precision and recall for the hate class at 0.742 and 0.758, respectively. Observations show that the text-based model excels when the transcript is relatively clean, the audio based model is effective during aggressive shouting and the vision model performs best when visual evidence of hateful activity is present.

3.2.2.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The paper highlights a significant gap in the literature regarding the detection of hateful content in videos especially when compared to existing works focused primarily on text. This research uses audio and video modalities enhancing detection accuracy with a curated dataset of over 1000 videos, which is a big improvement from previous datasets containing fewer than 500 videos providing a more robust foundation for analysis. The authors also detail a thorough annotation process improving the reliability of their findings.
- However, the study may benefit from exploring more diverse sources of hateful content beyond BitChute to ensure broader applicability of the findings.

3.2.2.3 Relationship to the proposed research work

Our research aligns with what the authors have explored i.e. multimodal hate speech detection. The need for effective moderation on platforms like Omegle underscores the relevance of our work. As the paper highlights that while some major social media platforms have strict content moderation policies such as Facebook's use of 15,000 moderators. Many smaller platforms lack the technology and resources to develop robust models for hate speech detection. The inclusion of multimodal approaches i.e. including the use of audio and visual data is crucial in improving detection rates especially in conversations where text may not fully capture the intent behind hateful speech. The challenges identified in the study regarding the effectiveness of text-based models when transcripts are noisy further explain the necessity for our approach which emphasizes the importance of audio analysis in understanding contextual cues in Urdu-English conversations.

3.2.3 A Multilingual Dataset for Gender-based Hate Video Detection across Cultural Contexts [3]

3.2.3.1 Summary of the research item

MultiHateClip presents a big enhancement in the detection of hateful video content through a multimodal and culturally sensitive approach. This research introduces a multilingual dataset specifically focusing on gender-based hate videos from diverse cultural contexts including English clips from YouTube and Chinese clips from Bilibili. The dataset consists of 10,000 videos from which 1000 were curated including over 300 identified as hate speech. The annotation process involved 80 hate lexicons tailored for both languages using native speakers who were familiar with the culture to label videos as hateful, offensive, or neutral, and to identify contributing modalities. The research highlights significant findings such as the low prevalence of hate videos and a notable bias against women. Models like GPT-4V which achieved an F1 score of 0.62 in English and BERT MFCC and ViViT, which performed best in Chinese with an F1 score of 0.5 underscore the importance of multimodal information in detection tasks and also reveal the inadequacy of pretrained models for non-Western languages.

3.2.3.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- This study addresses the need for a more refined understanding of hateful video content by including both textual and audio-visual modalities improving the dataset's reliability and validity through a strong annotation methodology that relies on native speakers and culturally aware annotators.

- However the low frequency of hate videos discovered raises questions about the effectiveness of content moderation policies on these platforms possibly leading to underrepresentation in the dataset. Additionally the reliance on a few-shot approach may limit the generalizability of the employed models across different contexts and languages while the difference between hateful and offensive content presents challenges particularly in the context of Urdu-English code-mixed language necessitating further exploration of culturally specific hate speech indicators.

3.2.3.3 Relationship to the proposed research work

The findings from MultiHateClip are directly relevant to our research focus on Urdu-English code-mixed language when it comes to toxicity classification. While this paper primarily examines gender-based hate speech in English and Chinese it emphasizes the necessity of culturally sensitive approaches that match with our goal of improving moderation in conversational platforms. The use of multimodal models, particularly Vision Language Models such as GPT-4V and Qwen-VL provides a framework that can be adapted to include Urdu-English data leveraging the insights gained about the problems of text-only models. The identified patterns and metrics such as sound intensity and Zero Crossing Rate can elaborate the audio analysis component of our research contributing to a more comprehensive understanding of toxicity detection in low-resource languages like Urdu.

3.2.4 ADIMA : Detecting Abusive Content in Multilingual Audio [4]

3.2.4.1 Summary of the research item

This research focuses on the detection of abusive content in audio highlighting the limitations of traditional Automatic Speech Recognition due to latency and underrepresented training examples for abusive words. The study utilizes the ADIMA dataset, which contains 11,775 audio samples in 10 Indic languages to classify audio as abusive or non-abusive. The classifiers leverage features extracted using VGG and Wav2Vec2 models which use context and underlying cues such as pitch, tone, intensity, and emotion in audio recordings. The study emphasizes the challenges associated with keyword spotting methods that rely on a dictionary which may overlook subtle cues in dialects, accents, and the often indistinct nature of abusive language.

3.2.4.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- This study provides an excellent approach to abusive content detection in audio by focusing on context and using transformer-based models which shows the potential of multimodal techniques.

- However relying on a limited dataset and audio only approach may restrict the generalizability of findings especially given the underrepresentation of abusive words in training examples. Further exploration of additional data sources could improve the study's conclusions.

3.2.4.3 Relationship to the proposed research work

This research highlights the need for effective moderation tools in conversational social media platforms particularly for Urdu-English code-mixed interactions which currently lack ample resources. The findings highlight the importance of including audio-based approaches along with text modalities in making systems for hate speech and toxicity detection. Making similar frameworks for Urdu will be crucial as it is a low-resource language with unique cultural contexts. The use of advanced models like Wav2Vec2.0 further highlight the need for a custom approach to address Urdu's specific linguistic challenges in detection of abusive content.

3.2.5 A Multimodal Framework for Detecting Toxicity in Code-Mixed Hindi-English Videos [5]

3.2.5.1 Summary of the research item

This study presents a multimodal multitask framework for detecting toxic content in code-mixed Hindi-English videos collected from YouTube. The dataset consists of 931 videos containing 4021 utterances where each utterance is labeled for toxicity, severity, and sentiment. The framework includes three main components: the Encoder module, Cross-Modal Synchronization module, and Multitask module. It uses many models like HingRoberta and HingGPT for text encoding, VideoMAE and Timesformer for video encoding and Whisper and MMS for audio encoding. The findings show a significant performance improvement in toxicity detection when various video modalities are included achieving an accuracy of 94.29% and an F1 score of 94.35%. The paper further highlights the challenges in existing text-based toxicity detection methods particularly in multilingual contexts like India where code-mixing complicates machine learning models. The authors uses a systematic encoding approach for aligning audio and video embeddings with text embeddings to enhance model performance.

3.2.5.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The inclusion of multimodal approaches significantly enhance the performance of toxic content detection explaining the effectiveness of the gated modality fusion mechanism over traditional fusion techniques. Additionally the use of multiple encoders reveals that HingRoberta and Hing-

GPT for text along with VideoMAE and Whisper for video and audio modalities are best models for this task.

- However the exclusion of implicit or indirect toxic expressions could make the framework's applicability in real-world scenarios less likely.

3.2.5.3 Relationship to the proposed research work

This research aligns closely with the proposed focus on multimodality particularly focusing on the integration of audio and text modalities for toxicity detection. The study highlights the need for developing similar systems for Urdu given its status as a low-resource language. By using audio along with text the proposed framework aims to find strong cultural expressions of toxicity that may be prevalent in Urdu too. The insights from this study further elaborate the importance of making robust audio-based hate speech detection systems in Urdu making way for improved moderation in conversational social media platforms.

3.2.6 Classifying Toxic Language in Audio Using Self-Attentive Neural Networks [6]

3.2.6.1 Summary of the research item

This paper presents an audio-based toxic language classifier that focuses on identifying toxic utterances primarily through acoustic signals that is quite different from traditional text-based or text-embedded methods. The researchers have studied two attention mechanisms i.e. Learnable Query Attention and Self-Attention to classify whether a short audio clip is toxic or not. They used data from online multiplayer gaming platforms, where 113,252 utterances were labeled as toxic and 25,660 as non-toxic. The model operates on Logarithmic Mel-Filter Banks features and is evaluated on both an internal toxic corpus and the public IEMOCAP dataset. The network is optimized using Stochastic Gradient Descent and the results show that self-attention provides more meaningful features for toxicity classification.

3.2.6.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- This is the first work to use audio modality primarily for toxic language classification filling a gap in research that mostly focuses on text. The model shows good performance and does well on two different datasets and also provides useful insights into how attention mechanisms capture toxic cues with self-attention proving superior for handling complex audio features.
- The Learnable Query Attention mechanism may also lose toxic-relevant information if a weak query is learned during training. Also relying on only acoustic aspects of an audio is not a good

idea in real world scenarios where people can be calm and relaxed yet spreading toxicity.

3.2.6.3 Relationship to the proposed research work

This research aligns with multimodal toxicity detection by explaining that toxicity can be identified through audio cues not just text. It highlights the importance of developing such systems for Urdu-English speech particularly as ASR underperforms in code-mixed and noisy environments like those found on conversational social media platforms. As Urdu is a low-resource language creating an audio-based toxicity classifier could make a significant contribution. The study underscores the role of audio features like tone and emotional cues which are hard to capture through text alone suggesting that focusing on Urdu audio data would provide valuable insights. Incorporating attention mechanisms like self-attention could effectively handle Urdu's acoustic complexities offering a better solution than current ASR-based models.

3.2.7 Leveraging Multimodal Machine Learning for Hate Speech Detection [7]

3.2.7.1 Summary of the research item

This paper explores multimodal hate speech detection using video, audio, and text modalities. The dataset includes 1051 video clips taken from YouTube and EMBY mainly featuring movie or series content. Each video lasts for about 3 to 5 seconds. Audio was extracted with MoviePy and transcribed using Google Speech Recognition. Video frames were processed with ViT and LSTM, audio features were extracted using MFCC, ZCR, and Chroma and textual features were obtained using BoW and Tf-Idf. A voting system determined the classification where content was marked as hateful if two or more modalities indicated hate. However results did not fully support the initial claim.

3.2.7.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The study's multimodal approach including distinct visual, auditory, and textual features provides a broader understanding of content for hate speech detection.
- The voting mechanism oversimplifies the complex interactions between modalities, potentially overlooking complex cases where one modality alone indicates hate.

3.2.7.3 Relationship to the proposed research work

This research highlights the significance of integrating multiple modalities particularly audio which is crucial for detecting hate speech in diverse contexts. For Urdu-English language this paper underscores

the importance of focusing on audio cues as they capture tone and emotional state often lost in text alone. Given Urdu's cultural complexities establishing a system that emphasizes auditory features as explored in this study is important. The paper's results stress the need to improve multimodal detection methods particularly in low-resource languages like Urdu where such approaches are not deeply explored and can make a significant impact.

3.2.8 Detecting Hate Speech in Short-Form Videos Using Audio Features [8]

3.2.8.1 Summary of the research item

This research studies hate speech detection in short-form videos taken from Filipino TikTok users focusing on content related to politics and commentary. A total of 4746 videos were scraped using an unofficial TikTok API from which videos with unclear voice and those that were dubbed were removed. For the annotation process 1000 video files were labeled by three annotators resulting in 387 hateful and 613 non-hateful ones. Noise reduction was performed using the ThinkDSP library and various audio features were extracted utilizing the Python Librosa library. Key features included Spectral Centroid, Spectral Rolloff, Spectral Bandwidth, Zero-Crossing Rate, Mel Frequency Cepstral Coefficients (MFCCs), and Chroma Features. Three classifiers i.e. SVM, Logistic Regression, and Random Forest—were tested with the Random Forest classifier achieving the highest performance. It was determined that Spectral Rolloff, MFCC, and Spectral Bandwidth were the most significant features for hate speech classification.

3.2.8.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The study uses a comprehensive feature extraction approach that captures various dimensions of the audio signal enhancing the model's capability to capture complexities in speech related to hate content.
- The normalization process for videos of varying lengths to create a uniform feature vector lacks clarity which raises questions about the reliability of the feature representation across the dataset.

3.2.8.3 Relationship to the proposed research work

This paper highlights the potential of leveraging audio features for hate speech detection aligning with the our research focus on multimodal classification that includes audio along with text. The points noted from this study highlight the necessity of developing systems for Urdu where cultural context is pivotal in understanding hate speech. By identifying effective audio features such as MFCC and Spectral Rolloff this paper highlights the importance of incorporating similar methodologies for Urdu addressing

the current gap in resources for low-resource languages and further explaining the need for effective moderation systems in conversational Social Medial Platforms.

3.2.9 DeToxy: A Benchmark Multimodal Dataset for Toxicity Detection in Spoken English [9]

3.2.9.1 Summary of the research item

The research introduces DeToxy the first publicly available toxicity-annotated dataset for spoken English taken from various open speech databases and consisting of over 2 million utterances. The dataset serves as a benchmark for detecting toxicity in spoken language highlighting the limitations of traditional text-based approaches that rely heavily on human-annotated transcripts which are prone to keyword bias. DeToxy includes two versions: the original and DeToxy-B which is a balanced dataset curated to account for helping factors such as trigger terms and sentiment labels. The authors implement a two-step approach to obtain transcripts using the pre-trained Wav2Vec-2.0 model for Automatic Speech Recognition and classify toxicity through a BERT BASE model. Additionally they experiment with an end-to-end approach utilizing Filter-Bank based feature extraction. The results indicate that both methods perform well in toxicity classification but highlight the importance of utilizing audio features to capture tonal and emotional cues that may be lost in transcription.

3.2.9.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The introduction of the DeToxy dataset fills a significant gap in the field of spoken language processing offering a foundation for future research in toxicity detection from audio and addressing a critical need for labeled datasets.
- The reliance on sentiment labels for oversampling non-toxic utterances could introduce bias as the sentiment annotation was conducted by only one annotator which may affect the overall diversity and representativeness of the dataset.

3.2.9.3 Relationship to the proposed research work

This research aligns with the proposed focus on multimodal approaches to toxicity detection by emphasizing the importance of incorporating audio as a modality alongside text. By employing a comprehensive methodology that utilizes both ASR and classification models, the findings underscore the need for similar systems in the Urdu-English context, particularly given Urdu's low-resource status. The recognition of audio features, such as tone and emotion, as critical indicators of toxicity provides a strong

argument for developing a tailored approach to Urdu audio-based hate speech detection. This research not only highlights the necessity of building resources for Urdu but also suggests that leveraging cultural nuances in Urdu audio could lead to more effective moderation in conversational platforms like Omegle, where such capabilities are urgently needed.

3.2.10 Detection of Hate Speech and Offensive Language in Roman Urdu [10]

3.2.10.1 Summary of the research item

The research highlights the need for automated hate speech detection in Urdu which often incorporates code-switching with English. Significant contributions include a lexicon of 621 hateful words specific to Urdu and a gold-standard dataset called Roman Urdu Hate-Speech and Offensive Language Detection (RUHSOLD) derived from tweets with binary and multi-class labels for analysis. The authors explore five multilingual embedding models and propose the Convolutional Neural Network n-gram (CNN-gram) architecture which outperforms several baseline models. The lack of acoustic cues in video data can hinder classification effectiveness as classifiers may struggle to capture emotional tone and context. However the developed lexicon can help search videos for hate speech classification by enabling keyword-based identification of offensive language.

3.2.10.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- Contributions such as the lexicon and dataset for Urdu address a critical gap in resources for hate speech detection in low-resource languages. The CNN-gram model improves detection accuracy by leveraging n-gram patterns. The comprehensive evaluation of multilingual embedding models and the introduction of domain-specific embeddings (RomUrEm) is quite valuable.
- The dataset may not capture the full spectrum of online interactions due to the evolving nature of slang and offensive language. Focusing primarily on Twitter may limit the generalizability of findings to other social media platforms. Also depending on only text modality may undermine the importance of acoustic cues in classification of an utterance as toxic.

3.2.10.3 Relationship to the proposed research work

This research aligns with the proposed focus on multi-modal hate speech detection, emphasizing the need for audio analysis in Urdu. It highlights the inadequacies of manual review processes and the subjective nature of hate speech definitions, underscoring the necessity for automated systems tailored to Urdu's unique linguistic and cultural aspects. This indicates a strong rationale for prioritizing research

in Urdu audio-based hate speech detection, which remains under-explored despite the growth of multilingual content online, advocating for integrating audio modalities to enhance detection accuracy and cultural relevance.

3.2.11 Emotion-Aware Multimodal Framework for Hate Speech Detection [11]

3.2.11.1 Summary of the research item

This paper presents the first multimodal deep learning framework designed to detect hateful content by integrating audio features that keep in mind emotional cues along with semantic features derived from text. It highlights the significance of emotional attributes in enhancing hate speech detection in multimedia formats arguing that hateful speech cannot by captured only by textual analysis but is deeply influenced by the emotional state of the speaker which is reflected in their tone and delivery. The authors introduce a novel dataset i.e. the Hate Speech Detection Video Dataset (HSDVD) specifically curated for multimodal learning as existing datasets fail to capture the complexities of hate speech in multimedia contexts.

The framework consists of three main models: the first utilizes transformer networks such as BERT and ALBERT to classify hate speech in text, the second employs a multi-task learning model trained on the IEMOCAP dataset to assess emotional attributes like valence, arousal, and dominance from audio inputs while the third implements a multilayer perceptron model for multimodal learning. The results elaborate that the multimodal framework outperforms traditional text-based models in precision and recall metrics signifying the importance of including multiple modalities for effective hate speech detection.

3.2.11.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The study provides a framework that successfully integrates emotional analysis from audio with text-based features resulting in improved hate speech detection. This dual approach offers a better understanding of the context in which hate speech occurs leading to more effective moderation strategies.
- However the research relies heavily on the HSDVD dataset which may not fit in the full spectrum of real-world hate speech scenarios. This limitation could impact the generalizability of the findings as the model's effectiveness in diverse contexts remains uncertain. Additionally the complexity of the proposed framework can make it challenging for practical deployment in real-time applications.

3.2.11.3 Relationship to the proposed research work

The integration of emotional features from audio could improve detection capabilities in conversational platforms where moderation is important to protect users from abusive content. By adapting the methodologies employed in this paper the proposed research can create a more effective framework specific for cultural contexts of Urdu. The inclusion of audio-based hate speech detection will not only fill a major gap in the existing literature but also use cultural understanding to enhance moderation efforts in Urdu-speaking environments.

3.2.12 A Transformer-based Model for Multimodal Hate Speech Detection Using Audio and Text Features [12]

3.2.12.1 Summary of the research item

The study explores a unique approach for detecting hate speech by using both audio and text modalities. Traditional methodologies primarily focused on text analysis. However this research recognizes the limitations of such approaches particularly in the context of complex communication where sarcasm and tone play important roles. To address these challenges the researchers used a model based on the Transformer framework which includes a layer called ”Attentive Fusion.” This model combines audio features extracted using log mel spectrograms with text data for a better identification of hate speech. The model outperformed previous state-of-the-art techniques achieving a macro F1 score of 0.927 on the test set indicating significant advancements in multimodal hate speech detection.

3.2.12.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The study effectively usess a Transformer framework combined with a unique ”Attentive Fusion” layer to combine both audio and text modalities achieving a high F1 score of 0.927 which tells how good it is in detecting hate speech and toxicity.
- Although the study uses a comprehensive dataset it lacks representation from low-resource languages like Urdu which could limit the applicability and effectiveness of the model in diverse linguistic contexts particularly in regions where Urdu is spoken.

3.2.12.3 Relationship to the proposed research work

This study is directly aligned with the proposed research focus on developing a system for detecting hate speech in Urdu-English code-mixed language. By highlighting the importance of audio along

with text it supports the point that a similar multimodal approach should be established for Urdu. The findings highlight the necessity for dedicated research in this low-resource languages where cultural nuances and conversational styles significantly impact communication. Given the non-availability of abundant audio-based hate speech detection systems in Urdu this research highlights the critical need for exploring audio modalities making a strong foundation for future studies that aim to improve moderation in conversational platforms.

3.2.13 Improving Voice Toxicity Classification with Cross-Modal Speech-Text Learning [13]

3.2.13.1 Summary of the research item

The research presents a novel framework for toxicity classification in voice chat environments by combining cross-modal learning to enhance audio-based classifiers with text. It focuses on the development of a multi-label speech toxicity classifier that uses semantic embeddings from text during training allowing for a better classification system that requires only audio input during inference. The framework uses a pre-trained speech encoder such as Wav2Vec 2.0 or WavLM combined with a multilingual text encoder like the Multilingual E5 Large. The approach is evaluated on huge datasets across multiple languages showing significant improvements in voice toxicity classification for various categories including Profanity, Bullying, and Racism.

3.2.13.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The framework effectively enhances the performance of audio-based toxicity classifiers by using rich text embeddings ensuring real-world applicability across multiple languages.
- Despite advancements the approach still relies on existing datasets which are often small or do not contain comprehensive real-world characteristics limiting the generalizability of the findings across diverse linguistic contexts.

3.2.13.3 Relationship to the proposed research work

This research aligns with the goals of testing multimodal toxicity detection by highlighting the importance of audio as one modality. The use of cross-modal learning improves the model's understanding of toxic speech highlighting the potential for similar frameworks to be developed for Urdu particularly in contexts where cultural complexities play an important role. As Urdu is a low-resource language with limited research on audio toxicity detection the findings suggest a strong need for a dedicated system

that addresses Urdu-English contexts. Making such a system would contribute significantly to the moderation of conversational platforms where audio interactions are abundant filling a critical gap in the existing literature and making safer online environments.

3.2.14 SONAR: Enhancing Cross-Lingual Representations with Multimodal Encoder-Decoder Frameworks [14]

3.2.14.1 Summary of the research item

The study presents a multilingual encoder-decoder framework for enhancing cross-lingual representation. It uses a teacher-student approach and aligns embeddings from a pretrained English encoder with those generated by a student model for Urdu and other languages minimizing differences via Mean Squared Error. The methodology extends to audio modalities aligning audio representations with their text representations. The architecture is trained on machine translation objectives with evaluations based on cross-lingual similarity (xsim, xsim++) and zero-shot speech-to-text translation.

3.2.14.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- This highlights emphasizes cross-lingual embedding challenges by aligning text and audio modalities signifying the importance of multimodal approaches. But depending on a single MT objective can limit adaptability to contextual language complexities particularly in code-mixed scenarios.
- The shallow decoder minimizes overfitting risks but might overlook deeper relationships important for classification accuracy. Although it could benefit from more diverse datasets for real-world applications.

3.2.14.3 Relationship to the proposed research work

The integration of audio and text modalities give clues for embedding alignment essential for our hate speech detection model. The multilingual encoder-decoder framework can make our design better and improve performance in this low-resource language contributing to effective moderation on social media platforms.

3.2.15 Cross-Lingual Emotion Recognition for Urdu: A Comparison with Western Languages [15]

3.2.15.1 Summary of the research item

This paper introduces an emotion recognition system specifically designed for the Urdu language. It is the first spontaneous emotional dataset in Urdu showing true emotional speech. Keeping in view the previous studies in cross-corpus emotion recognition from Western languages the authors experiment with various scenarios to assess both within and cross-corpus emotion recognition. By comparing their system against datasets from English, Italian, and German they highlight its cross-lingual capabilities. The results suggest that this approach not only improves emotion detection accuracy but also has exciting practical applications particularly in Pakistan and other regions.

3.2.15.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The creation of an emotion detection dataset in Urdu is indeed a significant achievement especially given the lack of resources for emotion recognition in low-resource languages.
- However the dataset's audios are too small (just 2-5 seconds) and are from political figures in news channels that doesn't quite capture the rich emotional landscape we face in everyday conversations.

3.2.15.3 Relationship to the proposed research work

This research provide great details that can contribute to our project on hate speech detection. The establishment of a publicly available Urdu emotional dataset is indeed helpful but it doesn't fully address the conversational setting we're interested in. By understanding the methodologies for emotion detection we can make informed choices in combining multimodal data for effective moderation. However we need to focus on creating datasets that truly reflect the emotional expressions and interactions found in everyday conversations to ensure our model is robust and relevant.

3.2.16 Comparison of General and Fine-Tuned Large Language Models for Urdu [16]

3.2.16.1 Summary of the research item

This paper compares the performance of general-purpose pretrained models like GPT-4-Turbo and Llama-3-8b-Instruct with task-specific fine-tuned models such as XLM-Roberta-large, mT5-large, and a fine-tuned Llama-3-8b-Instruct across seven classification tasks (e.g., sentiment analysis, abuse de-

tection, sarcasm detection, fake news detection, topic classification, PoS tagging, and NER) and six generation tasks (e.g., summarization, paraphrasing, translation).

The results show that fine-tuned models generally outperform general models especially in classification tasks. For example, XLM-R achieves the highest Macro-F1 score of 90.92 in abuse detection, slightly better than Llama-FT (89.15) and GPT-4-Turbo (89.01). In sentiment analysis, Llama-FT leads with a Macro-F1 of 95.30, just ahead of GPT-4-Turbo (94.90) showing GPT-4's competitive edge even without fine-tuning. For generation tasks, GPT-4-Turbo aligns more closely with human evaluations. The study highlights the difficulties of working with low-resource languages like Urdu and the importance of fine-tuning for tasks like NER and sarcasm detection where general models underperform.

3.2.16.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- The study evaluates both general-purpose and task-specific fine-tuned models across diverse tasks in Urdu giving a detailed view of model performance in a low-resource language languages.
- The research explains the importance of specialized fine-tuning showing significant performance gains in tasks like abuse detection, NER tagging, and sarcasm detection where general models are not that good.

3.2.16.3 Relationship to the proposed research work

This research dives deep into model selection and optimization for NLP tasks in the Urdu particularly for hate speech and abuse detection. The performance comparison between general-purpose and task-specific models tells us the decision to fine-tune models for better accuracy in conversational settings that is what our focus on effective moderation is about. The results further highlight the importance of building task-specific datasets for improved detection accuracy in abusive language detection.

3.2.17 Benchmarking ASR Models for Urdu [17]

3.2.17.1 Summary of the research item

This paper gives a comprehensive evaluation of Urdu Automatic Speech Recognition (ASR) models by comparing three prominent ASR families that are Whisper, MMS, and Seamless-M4T. The evaluation measures performance using WER across two datasets i.e. read speech and conversational speech. A conversational speech dataset is introduced designed specifically for benchmarking Urdu ASR models. The study finds that Seamless-large performs best on read speech datasets, while Whisper-large excels in conversational speech. Additionally the paper highlights the challenges of evaluating ASR models in

low-resource languages like Urdu and tells how important it is to normalize text for improving model accuracy.

3.2.17.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- This research is the first to introduce a conversational speech dataset for benchmarking Urdu ASR models which is a big contribution to the development of ASR for low-resource languages.
- The study emphasizes the complexity of evaluating ASR models with traditional quantitative metrics showing the importance of including qualitative analyses particularly in low-resource contexts such as Urdu conversational speech.

3.2.17.3 Relationship to the proposed research work

This research is relevant to our focus on multimodal detection and moderation in low-resource languages. The insights from this evaluation, especially the emphasis on conversational speech and text normalization can help and improve our strategies for making ASR models better to improve accuracy in identifying abusive language in real-time interactions.

3.2.18 Transformer-Based Approach for Roman Urdu Hate Speech Detection [18]

3.2.18.1 Summary of the research item

This study presents a detailed methodology for detecting hate speech in Roman Urdu using the RU-HSD-30K comprising 30,000 messages from social media platforms like Twitter and Facebook. The dataset has both ”Hate” and ”Neutral” classes in an even manner providing a balanced training approach. Major contributions include a preprocessing pipeline that involves tokenization, normalization, and feature extraction using transformer-based embeddings.

The architecture uses a transformer model, known for its ability to capture contextual information effectively to classify hate speech. The proposed model seems highly accurate for hate speech detection in low-resource languages like Roman Urdu filling a critical gap in already existing research.

3.2.18.2 Critical analysis of the research item

The strengths and weaknesses of this research are as follows:

- The use of a large and balanced dataset fills a big gap in hate speech detection for Roman Urdu making it a huge contribution to the field. The integration of a transformer-based model makes

capturing complex patterns in text easier improving classification performance.

- The preprocessing steps like normalization are very important in highlighting the lexical variability found in Roman Urdu but relying on a defined dictionary may limit adaptability to new slang and evolving language use. Also the focus on text-based analysis may overlook the importance of other modalities including audio or visual elements, which are highly relevant in online interactions.

3.2.18.3 Relationship to the proposed research work

This research aligns very closely with the proposed focus on multimodal hate speech detection and highlight the need for automated systems that can fit to the complexities of Roman Urdu. It also highlights the problems of manual review processes and tells about the dynamic nature of hate speech definitions advocating for systems made specifically for the linguistic and cultural aspects of Urdu. The findings tell that there is a need for enhancing detection accuracy through a multimodal approach due to the rich context of communications in social media environments.

3.2.19 wav2vec 2.0: Self-Supervised Speech Representation Learning [19]

3.2.19.1 Summary of the research item

Wav2Vec 2.0 is a big step forward in learning from audio data. It is a self-supervised framework that takes raw audio waveforms and transforms them into meaningful speech representations. The model uses convolutional layers to extract features and then a transformer encoder to process these features. During training it masks parts of the audio input and it teaches the model to predict those hidden segments based on the context provided by the surrounding audio. This approach not only helps in learning but also helps in achieving to state-of-the-art performance in automatic speech recognition tasks even with limited labeled data.

3.2.19.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- Wav2Vec 2.0 addresses the challenge of limited labeled data in speech recognition through self-supervised learning making it particularly useful for languages that do not have huge resources. Its focus on predicting masked segments can restrict its ability to fully understand spoken language including complexities like emotion.
- Its focus on predicting masked segments can restrict its ability to fully understand spoken language including complexities like emotion.

3.2.19.3 Relationship to the proposed research work

The concepts and results from Wav2Vec 2.0 are directly relevant to our research on hate speech detection. By using its audio representations and combining them with text and visual data we can enhance our model's capabilities ensuring it can moderate content on social media platforms effectively.

3.2.20 Vision Transformer: Transformers for Scalable Image Recognition [20]

3.2.20.1 Summary of the research item

Vision Transformer is an image classification technique that uses transformer architecture that is commonly associated with natural language processing tasks. Instead of analyzing images as a whole ViT breaks them into smaller fix-sized patches and flattens these patches into a sequence of data. This sequence is then passed through a transformer encoder that uses self-attention mechanisms to understand the relationships between these patches. ViT is trained on large datasets and has shown that it can compete with traditional convolutional neural networks in various image classification tasks.

3.2.20.2 Critical analysis of the research item

The strengths and weaknesses of this research paper are:

- ViT shows how powerful transformers can be for visual tasks clearly explaining the ability of self-attention.
- However it does require substantial amounts of training data, which might limit its use in scenarios with fewer resources. It can miss out local details that CNNs typically excel at capturing. This could be a drawback in tasks requiring precise classification. Also although ViT achieves strong results understanding how it makes decisions—compared to traditional CNNs is still something that needs to be explored further.

3.2.20.3 Relationship to the proposed research work

ViT's architecture can provide guidance for our multimodal hate speech detection model especially in how we handle visual information. By understanding how visual representations align with text and audio features we can improve our model's overall performance and make it better for moderating interactions on conversational platforms.

3.3 Literature Review Summary Table

Table 3.1: This table has a general overview of the papers that have been studied so far and their results.

Author	Method	Results	Limitations
Costa-jussà et al. [1]	Utilized SONAR for speech classification with a multilingual encoder and decoder; trained using a machine translation objective to align embeddings across languages.	Achieved an average recall of 0.57 over all 30 languages	Urdu dataset primarily consists of translations from Quran and Bible introducing a bias.
Das et al. [2]	Multimodal hate speech detection in English using text, audio, and video (BERT + ViT + MFCC fusion model) on BitChute's data.	Achieved a macro F1-score of 0.790, with precision of 0.742 and recall of 0.758 for the hate class.	Dataset limitations include noise in transcripts, and the model struggles with unrelated visual content in some videos.
Gupta et al. [4]	Utilized VGG for log-mel spectrogram features and Wav2Vec2 models (XLSR-53, CLSRIL-23, Him-4200) for abusive audio detection.	The CLSRIL-23 model achieved the highest performance with 80.57% average accuracy and 77.96 F1 score.	The dataset primarily contains recordings with fewer than 5 profane words, making it challenging to identify profanity in some instances.
Maity et al. [5]	Utilized a MultiModal Multitask framework incorporating three modules: Encoder, Cross-Modal Synchronization, and Multitask modules. Gated modality fusion was performed.	Achieved 94.29% accuracy and 94.35 F1 score on toxic content detection using text, audio, and video.	Did not consider the broader context of video clips, and implicit/indirect toxic expressions were excluded from the study.

Wang et al. [3]	MultiHateClip presents a multimodal dataset focusing on gender-based hate videos from English and Chinese contexts, employing 80 hate lexicons for annotation by culturally aware native speakers.	Evaluated current models with GPT-4V achieving F1 score of 0.62 for English and BERT MFCC/-ViViT with F1 score of 0.5 for Chinese.	A low frequency of hate videos was found, with challenges in distinguishing between hateful and normal videos based solely on text, highlighting limitations in text-based hate speech detection.
Yousefi and Emmanouilidou [6]	An audio-based toxic language classifier using Learnable Query Attention and Self-Attention to analyze the acoustical context of short audio clips.	Achieved 75.87% accuracy and 79.82 F1 score on Corpus A, and 68.85% accuracy on IEMOCAP.	Performance may degrade with background noise and overlapping speech; weak query training can lead to loss of toxic information.
Boishakhi et al. [7]	This study employs a multimodal approach for hate speech detection, analyzing 1051 video clips (3-5 seconds) using audio (MFCC, ZCR), video (ViT, LSTM), and text features (BoW, Tf-Idf) with a voting system for classification.	Best Multi-modal F1 Score: SVM: 0.875	Results did not fully support the hypothesis, indicating integration challenges.
Iba˜nez et al. [8]	Traditional Machine Learning Algorithms: Logistic Regression, Random Forest, Support Vector Machines (SVM) applied to audio features (MFCCs, Spectral Centroid, etc.)	Achieved up to 78.5% accuracy on hate speech detection using an optimized Random Forest model. Identified Spectral Rolloff and MFCCs as top predictors.	Limited to Filipino language data; relies on pre-extracted audio features without transcribing speech.

Ghosh et al. [9]	Introduced DeToxy, a toxicity-annotated dataset with over 2 million utterances. Employed a 2-step approach combining ASR (wav2vec-2.0) for transcription and BERT BASE for toxicity classification. DeToxy-B is a balanced version considering sentiment labels.	Achieved superior performance with an end-to-end (E2E) model over the 2-step approach when gold transcripts are unavailable.	ASR models struggle with domain generalization, resulting in average WER of 33%, 43%, and 26.9% across different datasets. Text sequence classifiers are not robust to domain changes.
Rizwan et al. [10]	Introduced CNN-gram, a deep learning model for detecting hate speech in Roman Urdu, using a custom dataset (RUHSOLD) and a lexicon of 621 hateful words.	CNN-gram achieved a macro F1-score of 0.84, outperforming seven baseline models.	Limited to textual aspect of hate speech only. Doesn't take into account the acoustic features that are crucial for classification in many scenarios.
Rana and Jha [11]	A multimodal deep learning framework using BERT/ALBERT for text and emotional audio features for hate speech detection, supported by a newly created dataset (HSDVD) via transfer learning.	Improved precision and recall compared to baseline models, highlighting the benefits of multimodal approaches.	Limited dataset size and potential biases may affect generalizability and performance.
Mandal et al. [12]	A Transformer-based approach combining audio and text sampling with an "Attentive Fusion" layer to detect hate speech.	Achieved a macro F1 score of 0.927, surpassing previous state-of-the-art techniques.	May still struggle with nuanced speech interpretations and sarcasm, as well as reliance on the quality of the combined datasets.

Liu et al. [13]	LSTM, Random Forest, Decision Trees, KNN, Logistic Regression, Multinomial Naïve Bayes (Google Jigsaw Dataset); BERT + ANN, BERT + MLP (HateXplain Dataset)	- LSTM: 97.6% Accuracy - Random Forest: 90% Precision - BERT + ANN: 93.67% Accuracy - BERT + MLP: 93.55% Accuracy	Overall performance varied across models, with some showing limited interpretability and effectiveness across diverse communities.
Duquenne et al. [14]	Multilingual Encoder-Decoder: Trained with a machine translation/ Auto-Encoding objective using a teacher-student approach to align embeddings across languages and modalities (text and audio).	SONAR slightly outperforms Whisper v2 in zero-shot STT with a BLEU score 25.3 averaged over 37 languages as compared to 24.5. Best performance on Indian languages (Bengali, Hindi, Kannada, Telugu, Tamil, Urdu).	Limited to textual aspect of hate speech only. Doesn't take into account the acoustic features that are crucial for classification in many scenarios.
Latif et al. [15]	Support Vector Machine (SVM) with eGeMAPS features, Logistic Regression and Random Forest across multiple datasets (URDU, EMO-DB, SAVEE, EMOVO)	SVM gave the best accuracy over all 4 datasets 83.40%, 65.10%, 81.30% and 74.10% respectively	Limited to only 4 emotions. Anger portion is limited to short clips of politicians shouting on new channels.
Arif et al. [16]	Generalist models: GPT-4-Turbo, Llama-3-8b-Instruct; Specialists models: XLM-R-large, Llama-3-8b-Instruct-FT	Sentiment Analysis: GPT-4 (94.90), Llama-FT (95.30); Abuse Detection: XLM-R (90.92), Llama-FT (89.15); Sarcasm Detection: XLM-R (84.37), Llama-FT (81.48); NER Tagging: Llama-FT (90.90); PoS Tagging: Llama-FT (67.55)	Limited performance on specialized tasks without fine-tuning, especially in sarcasm detection and NER recognition.

Arif et al. [17]	ASR Model Families: Whisper, MMS, Seamless-M4T	Seamless-large: Best performance on read speech dataset (WER: 17.09 on ARL) and Whisper-large: Best on conversational speech dataset (WER: 22.35 on CSaLT)	Limited Conversational dataset (Urdu) for finetuning. Poor ASR performance on overlapping speech still persists.
Bilal et al. [18]	Used BERT-RU and BiLSTM, leveraging transfer learning for improved hate speech detection.	Achieved accuracy of 99% with the transformer model. BERT-RU + BiLSTM accuracy varied from 0.5524 to 0.9649	Pre-trained BERT-RU models underperformed compared to English versions; fine-tuning led to overfitting.
Baevski et al. [19]	Vision Transformers (ViTs) process images by dividing them into fixed-size patches (16x16 pixels) and treating these patches as tokens. The model uses a transformer architecture to capture global context and spatial relationships.	ViTs achieved state-of-the-art performance on several benchmarks including ImageNet and COCO surpassing traditional convolutional neural networks (CNNs) in accuracy while requiring less inductive bias.	ViTs require large amounts of labeled data for training, making them less effective in low-data scenarios. Additionally, their reliance on self-attention can lead to computational inefficiencies, especially with larger images, and may hinder real-time applications.
Dosovitskiy et al. [20]	Wav2Vec 2.0 leverages self-supervised learning to extract meaningful speech representations from raw audio. The model is pre-trained on unlabeled audio data, using masked prediction to learn contextual features.	The model achieved state-of-the-art performance in speech recognition tasks, significantly reducing the need for labeled data.	Performance is highly dependent on the quality and diversity of the pre-training data; models may struggle with underrepresented accents and dialects.

3.4 Related Applications

3.4.1 Azar Live

3.4.1.1 Summary

Azar Live is a one-to-one videochat platform. When signing in, users can set the country they want to meet from before getting in touch with random people. For communication with the other country users, one is compelled to buy gems with real cash. Azar Live gives a variety of effects and filters for group video calls.

3.4.1.2 Flora's Distinction

Safety features like ID verification are available on Azar Live but the application has no professional tone. Flora is created exclusively for the professional connections with focus on skill sharing instead of casual acquaintance or casual chit-chat. Further, Flora will use new advanced safety features such as using NLP for content control and a well-structured matching algorithm to enhance safety amongst the professionals in the industry.

3.4.2 Chatroulette

3.4.2.1 Summary

Chatroulette is a site which helps people to chat via video connected with other individuals. It has a 'top users' filter that currently can only be accessed if you purchase quids for connections. User can make friends in chatroulette. Also, it recognizes if there is a face infront of camera. If there isn't it doesn't let you videochat. It means they detect facial features.

3.4.2.2 Flora's Distinction

Unlike in Chatroulette which focuses on random, strained conversations, Flora is more focused on people with particular common interests. The more function that Flora will include are content moderation to ensure every conversation is polite, something that's missing in Chatroulette to some extent.

3.4.3 Houseparty

3.4.3.1 Summary

Houseparty is an application that is a social network that enables individuals engage in group video calls, including with friends, or unknown people. It is created for casual exchanges of messages and grouping

features, which make it very friendly for social purposes.

3.4.3.2 Flora's Distinction

Flora is not designed to do random casual chitchat, but rather to provide connections that interest the users. Whereas, Houseparty has casual communication, Flora will be the app that only aims to corner the user with the primary intention to help them to bond with people having similar interests. Flora will also have content moderation and user verification elements so as to allow professionalism and security.

3.4.4 Omegle

3.4.4.1 Summary

Omegle, The most famous platform which is now discontinued after a lawsuit accused it of facilitating child abuse. The said platform was a popular platform for users to connect them randomly to chat using texts or via video calls. It was a platform to chat without exposure of the user's details to others on the same platform.

3.4.4.2 Flora's Distinction

Omegle was mainly used for casual discussions. And due to very little to no strictness, the platform was being used in a very casual and unprofessional way. And that's the core difference flora will have, Flora isn't for casual chitchat and User safety is the top priority.

3.5 Related Applications Summary Table

Table 3.2: Comparison of Related Applications for Flora

Application	Features	Relevance	Limitations
Azar Live	One-to-one video chat, country selection, gem purchases for international communication, effects/filters for calls	Provides insights on user preferences for country-based matching and safety features	Lacks professional talks and audience; primarily casual interactions without structured matching

Chatroulette	Random video chat, 'top users' filter (requires purchase), facial recognition for chat access	Highlights the appeal of random interactions and the need for moderation in video chats	Focuses on random, strained conversations with little emphasis on common interests or professionalism
Houseparty	Group video calls, casual messaging, friendly social features	Demonstrates the importance of social engagement in video calls	Primarily designed for casual communication; lacks focus on professional networking or interest-based connections
Omegle	Random text/video chat, anonymity, no user detail exposure	Shows demand for anonymity in casual chats	Previously associated with unprofessional usage and safety concerns; lacks structure and moderation for meaningful interactions

3.6 Conclusion

To conclude, no significant work has been done to include audio and visual modalities in toxic language detection in Urdu. Considering the fact that Urdu has more than 70 million native speakers it is necessary to dive deep into moderation specific to Urdu speaking environments. The only large audio dataset that aimed for toxicity detection, to the best of our knowledge, was proposed by Juss'a et al (2024) [1] but considering that the dataset used is translations from Holy Scriptures using it as a benchmark for evaluating models on multimodal toxicity detection tasks is not what is required. Some work has also been done on sentiment classification (happy, sad and angry) using Urdu Audios [15] but the videos tagged as anger were short videos from new channels containing people shouting to prove their stance. These datasets are not ideal for what we are trying to achieve as they are non-contextual and very short in length and are limited to a very small part of a huge problem.

The abundance of research highlighting the importance of modalities other than text in classifying hate speech and toxicity in general makes it clear that text alone can not work as a classifier in all scenarios. Das et al. (2023) [2] curated a multimodal dataset for hate speech classification in English from videos obtained from BitChute and saw a significant improvement in accuracy when various modalities were fused. Text Modality alone achieved an F1 score of 0.609 on videos less than or equal to 105 second and 0.7 on greater than 105 seconds, while fusion of all modalities by ViT, MFCC and BERT achieved the best score of 0.772 and 0.759 on videos less than or equal 105 and greater than 105 seconds respectively. Similarly, Maity et al. (2024) [5] studied how multimodal fusion can aid in detecting toxicity in Hindi-English Code-Mixed scenarios. Hing-BERT performed best on text modality with and F1 score of 86.98% while adding just audio modality by fusing Whisper increases F1 score to 87.26%. Combining

all three modalities text, audio and vision (Hing-BERT, Whisper and VideoMAE) achieves the highest F1 score of 88.09%. A significant amount of work has also been done on audio as a standalone modality for toxicity detection [4] [6] highlighting it's significance in solution of a growing problem.

These results signify that acoustic cues can help in detecting aspects that text do not cover like pitch and tone etc. that become quite important in conversations containing sarcasm and double-meaning utterances. Therefore, we aim to extend this research area that is widely being explored in other languages to Urdu for improving automatic moderation systems for real-time conversational applications.

Chapter 4 Software Requirement Specifications

Flora's Software Requirement Specifications(SRS), will highlight the major features, functions and use cases. This is to help all the future developers and users guide through the project.

4.1 List of Features

- **Video Chat** Connects with others through video chats. Whether for casual conversations or deep discussions. Flora brings people together.
- **AI-Powered Toxicity Detection** Flora goes beyond simple moderation by detecting toxicity. AI ensures a positive environment for all users
- **Registration** Users can create accounts. Information like their name, email and password is needed to make information. This process ensures that only verified individuals can access the platform.
- **Login** Registered users can access their accounts using email and password. The login feature includes safeguards such as password recovery options.
- **Email Verification** Users receive a verification email to confirm their identity. This step helps to keep the user secure.
- **Update Profile** Users have the ability to edit and update their profiles.
- **Frontpage** It is designed for easy navigation, allowing users to easily find what they need.
- **Notifications Page** Users can view important updates regarding their account activities and new matches.
- **Video Call Page** This dedicated page allows users to engage in one-on-one video conversations with their matches.
- **Send Message During Video Call** Users can send text messages while in a video call.
- **Report Feature** Users can report toxic during interactions. This feature allows the app's moderation to take necessary actions.
- **Profile Matching Feature** Flora connects users based on compatibility between users.
- **Real Time Messaging** Users can send and receive messages with each other in real time, during videochat.

4.2 Functional Requirements

- **User Profiles**: Users should be able create personalized profiles. It include bio, interests.
- **Toxicity Filtering**: Flora automatically detects and filters out toxicity in both text and speech.
- **Interest-Based Connections**: Join chats focused on specific interests or topics, allowing users to engage in video chats and discussions with other participants.
- **Report and Block Features**: Users can report inappropriate behavior or block other users directly from the app, empowering the community to maintain a respectful environment.
- **Push Notifications**: Users receive notifications for new messages, chat requests, and scheduled events, ensuring they never miss an opportunity to connect.

4.3 Quality Attributes

1. **Usability**: Flora should have a simple to use interface.
2. **Reliability**: The system ought to deliver reliable outcomes.
3. **Performance**: The system must ensure quick responses to queries and user activities.

4.4 Non-Functional Requirements

Flora aims to be a secure video chatting platform that prioritizes user safety. The following quality attributes form the backbone of Flora, driving its non-functional requirements and ensuring an optimal user experience:

- **Reuse-ability**: The model trained and the data gained from the app can be used for further research.
- **Correctness**: Flora hopes operates with precision. Ensuring that all features from interest based matching to real time filtering perform accurately and as expected.
- **Flexibility**: Flora offers customization features like interest based matching. This flexibility ensures that users can tailor their experience to meet their individual needs.

4.5 Assumptions

- Users have access to internet

- Have access to a PC with browser installed
- users have basic understanding of how to use a web apps

4.6 Use Cases

Table 4.1: Registration

Name	Registration		
Actors	User ,System		
Summary	The user provides necessary information to create an account and the system verifies and stores this information for future logins.		
Pre-Conditions	The user has the app open		
Post-Conditions	The user has been registered		
Special Requirements	None		
Basic Flow			
Actor Action		**System Response**	
1	User goes to registration page.	1	Registration page asks for email and password.
2	User enters email and password.	2	The system verifies the email and password.
Alternative Flow			
2-A	User inputs wrong email or password.	2-A	System shows a message explaining the error.

Table 4.2: User Login Use Case

Name	User Login	
Actors	User, System	
Summary	User provides information to access their account and system validates it.	
Pre-Conditions	The user must have an account and access to the login page of the app.	
Post-Conditions	User is successfully logged-in, and their session starts for further use.	
Special Requirements	None	
Basic Flow		
Actor Action		**System Response**
1	User goes to the login page.	System shows the login form.
2	User inputs email and password.	The System validates password and email.
3	User submits the login form.	System checks input and if valid then continues.
4	User is moved from the login page	User is logged in and can access features.
Alternative Flow		
1-A	User enters invalid password .	System shows error for invalid credentials.
2-A	User doesn't remember password.	System provides password reset link and sends email check.

Table 4.3: User Logout Use Case

Name		User Logout
Actors		User, System
Summary		User clicks the logout button to end their session and is redirected to the login page.
Pre-Conditions		User is logged in and can access the navigation bar with the logout button.
Post-Conditions		User is logged out, session is terminated, and the login page is displayed.
Special Requirements		None
Basic Flow		
Actor Action		**System Response**
1	User clicks the logout button in the top-right navigation bar.	System initiates logout by clearing the JWT token and session data.
2	User is redirected from the current page.	System displays the login page.
Alternative Flow		
1-A	User accidentally clicks the logout button and cancels if prompted.	System keeps the user logged in on the current page.
2-A	System fails to clear session data due to an error.	System shows an error message and prompts retry.

Table 4.4: Category-Based Matching Use Case

Name		Category-Based Matching
Actors		User, System
Summary		User selects a category and is matched with another user for a video call.
Pre-Conditions		User is logged in and can access the category selection page.
Post-Conditions		User is matched and ready to start a video call.
Special Requirements		At least two users must be in the same category.
Basic Flow		
Actor Action		**System Response**
1	User navigates to category selection page.	System displays available categories.
2	User selects a category.	System adds user to category waiting list.
3	User waits for a match.	System pairs user with another in same category.
4	User is notified of match.	System initiates video call setup.
Alternative Flow		
1-A	No other users are in the selected category.	System informs user to wait or choose another category.
2-A	System fails to match due to an error.	System shows error and prompts retry.

Table 4.5: AI-Powered Toxicity Detection

Name	Toxicity Detection		
Actors	User, System		
Summary	The system detects and filters out toxic or abusive language.		
Pre-Conditions	The system successfully filters inappropriate content and warns the user		
Post-Conditions	The system successfully filters inappropriate content and warns the user		
Special Requirements	AI models must be trained on toxicity detection for the language.		
Basic Flow			
Actor Action		**System Response**	
1	The user communicates in a language AI is trained in.	2	The system monitors the conversation in real-time.
3	The user uses offensive language.	4	The system detects toxic content and sends a warning message to the user
Alternative Flow			
3-A	Repeated offensive language is detected.	4-A	The system may mute the user or end the chat.

Table 4.6: Reporting Feature

Name	User Reporting Feature		
Actors	User, System, Moderator		
Summary	User can report inappropriate behavior during a chat session.		
Pre-Conditions	User must be in an active chat session.		
Post-Conditions	The report is successfully submitted, and a moderator reviews it for appropriate action.		
Basic Flow			
Actor Action		**System Response**	
1	User encounters inappropriate behavior.	2	User selects the 'Report' option.
3	User provides details of the issue.	4	System collects the report and confirms submission.
5	User receives confirmation.	6	System logs the report for moderator review.
Alternative Flow			
3-A	User cancels the report.	4-A	System discards the information and returns to chat.

Table 4.7: Text Messaging Use Case

Name		Text Messaging
Actors		User, System
Summary		User sends and receives text messages during a video call.
Pre-Conditions		User is in an active video call with another user.
Post-Conditions		Messages are exchanged and displayed in the call interface.
Special Requirements		WebSocket connection must be active.
Basic Flow		
Actor Action		**System Response**
1	User opens text messaging panel in call interface.	System displays the messaging side panel.
2	User types and sends a message.	System delivers message via WebSocket to other user.
3	User receives a message from other user.	System displays message in the side panel.
Alternative Flow		
1-A	User sends message during WebSocket disconnection.	System shows error and prompts retry.
2-A	Message content violates platform guidelines.	System flags message and notifies user.

Table 4.8: Frontpage Display Use Case

Name		Frontpage Display
Actors		User, System
Summary		User views frontpage to navigate features.
Pre-Conditions		User is logged in and accesses frontpage.
Post-Conditions		User can navigate to features from frontpage.
Special Requirements		Internet connection required.
Basic Flow		
Actor Action		**System Response**
1	User accesses frontpage after login.	System shows personalized frontpage.
2	User clicks a section (e.g., notifications).	System redirects to selected section.
Alternative Flow		
1-A	User is not logged in.	System redirects to login page.
2-A	Frontpage fails to load due to error.	System shows error and prompts retry.

Table 4.9: Notifications Page

Name	Notifications Page		
Actors	User, System		
Summary	The notifications page displays alerts about messages, matches, and other important updates.		
Pre-Conditions	The user must be logged in to view their notifications.		
Post-Conditions	The user has an overview of recent notifications and can take action as necessary.		
Basic Flow			
Actor Action		**System Response**	
1	The user navigates to the notifications page.	2	The system displays a list of recent notifications.
3	The user reviews the notifications.	4	The system highlights unread notifications.

Table 4.10: Video Call Page

Name	Video Call Page		
Actors	User, System		
Summary	The video call page allows users to engage in real-time video conversations with their matches.		
Pre-Conditions	user must be matched with another user to initiate a video call.		
Post-Conditions	Communication between users via video and audio during the call.		
Basic Flow			
Actor Action		**System Response**	
1	The user navigates to the video call page.	2	The system establishes a connection with the matched user.
3	The user starts the video call.	4	The system activates the video and audio feeds.

Table 4.11: Email verification

Name	Email verification		
Actors	User, System		
Summary	The user can get verification email after registrations, .		
Pre-Conditions	The user must have finished registration developer.		
Post-Conditions	The user can get verified and stay secured.		
Special Requirements	None		
Basic Flow			
Actor Action		**System Response**	
1	The user checks email after registration.	2	System sends the email.
2	The user clicks the verification link.	4	The system verifies that is it the user who has signed up.
Alternative Flow			
3	email was not sent	4-A	The system sends the email again.

Table 4.12: Profile Matching

Name	Profile Matching		
Actors	User, System		
Summary	The system analyzes user profiles and preferences to suggest potential matches based on compatibility.		
Pre-Conditions	Users must have completed their profiles and indicated their preferences for matching.		
Post-Conditions	Users receive a list of suggested matches based on their profiles and preferences.		
Basic Flow			
Actor Action		**System Response**	
1	The user requests profile matching by clicking on the ”Find Matches” button.	2	The system retrieves the user’s profile and preferences.
3	The system analyzes the user’s profile against the profiles of other users in the database.	4	The system generates a list of potential matches based on compatibility scores.
5	The system displays the list of suggested matches to the user.	6	The user reviews the suggested matches and can choose to initiate contact or view more details about each match.
Alternative Flow			
3-A	The system finds no matches based on the user’s profile.	4-A	The system informs the user that no matches are found and suggests completing their profile for better results.

Table 4.13: update profile

Name	Update profile		
Actors	Sy5stem, user		
Summary	The user can change the aspects of the profile.		
Pre-Conditions	A profile already exists		
Post-Conditions	The user profile has been changed.		
Special Requirements	None		
Basic Flow			
Actor Action		**System Response**	
1	User goes to profile page and inputs required fields.	2	System displays profile information.
2	User changes the fields and the saves.	4	The system saves the changes.
Alternative Flow			
3-A	inputs an invalid field.	4-A	The system asks for the input again message: Incorrect email or password entered.

4.7 Hardware and Software Requirements

List the hardware and software requirements that will be required to develop and deploy the project.

4.7.1 Hardware Requirements

- Sufficient Memory
- Adequate Computation Power to moderate the chat
- Stable and High-Speed Internet Connection

4.7.2 Software Requirements

- Redux Toolkit
- React.js
- WebRTC
- Nginx
- Django
- PostgreSQL
- Redis
- PyTorch

4.8 Graphical User Interface

This section provides very basic GUIs for FLORA web app just to give an overview of how things will work.

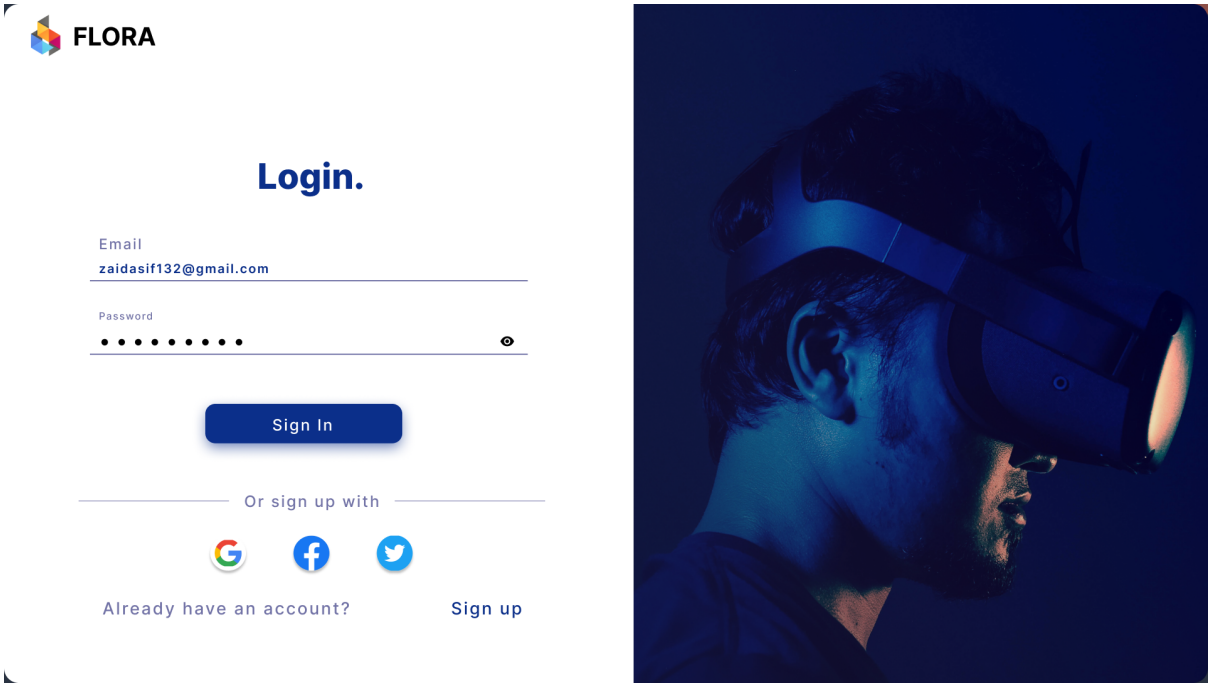

Figure 4.1: Login page

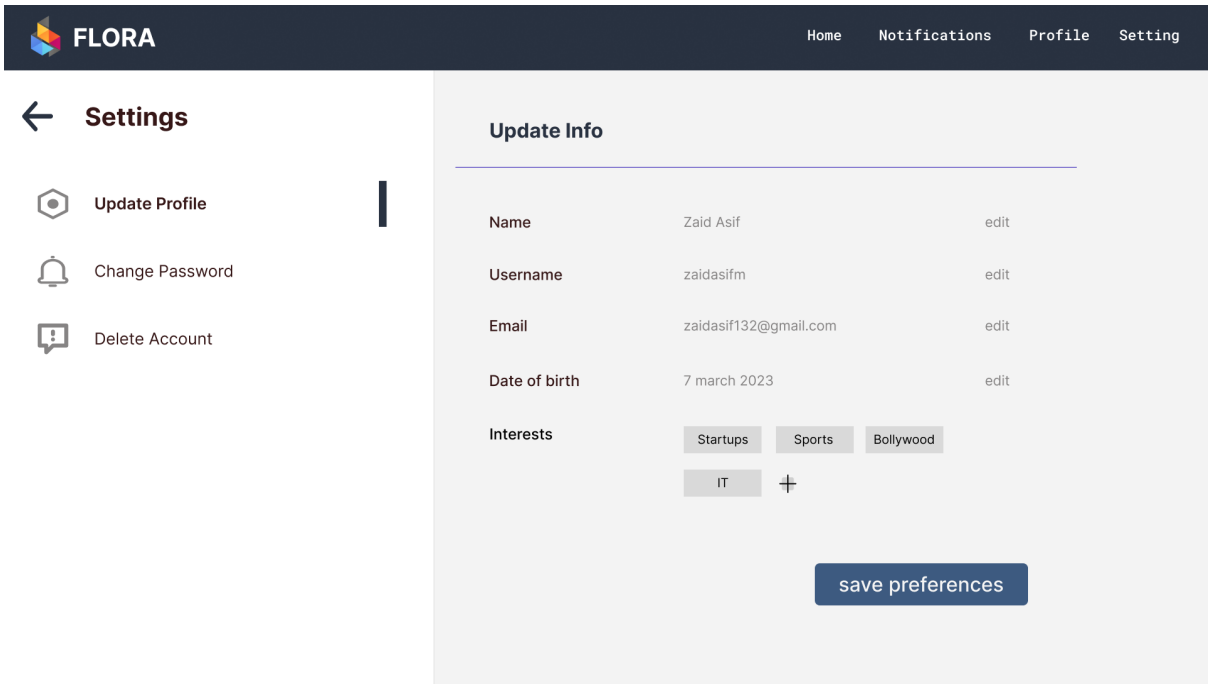

Figure 4.2: Update profile

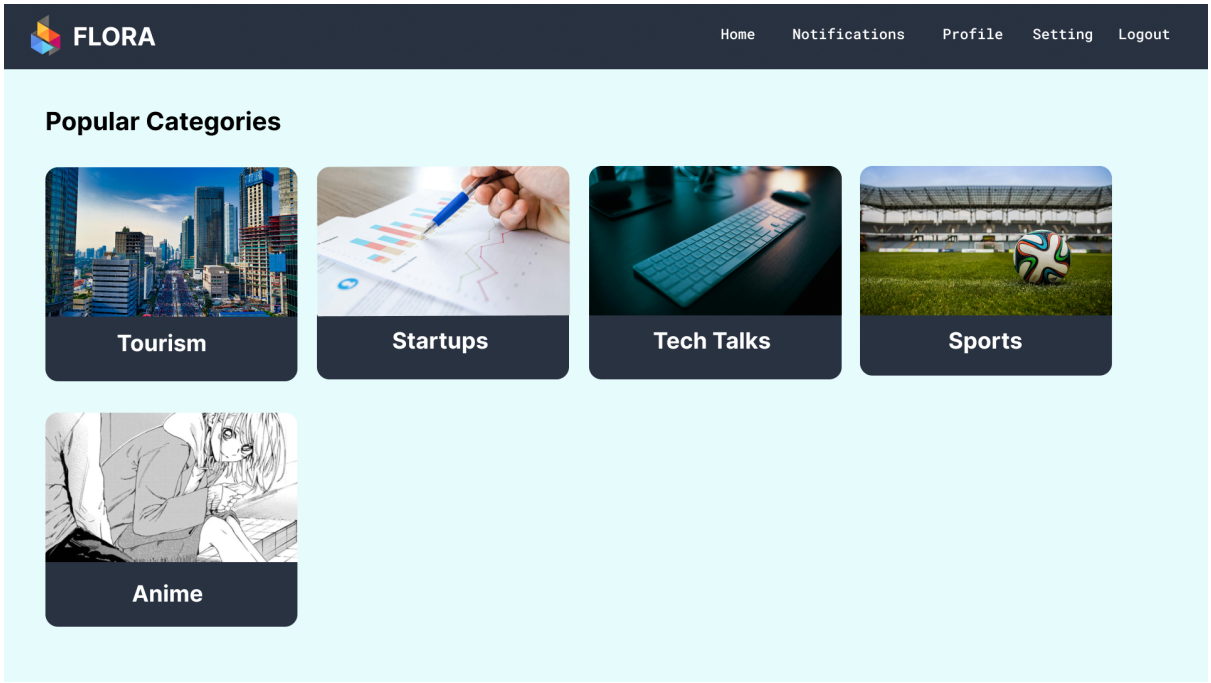

Figure 4.3: Home page

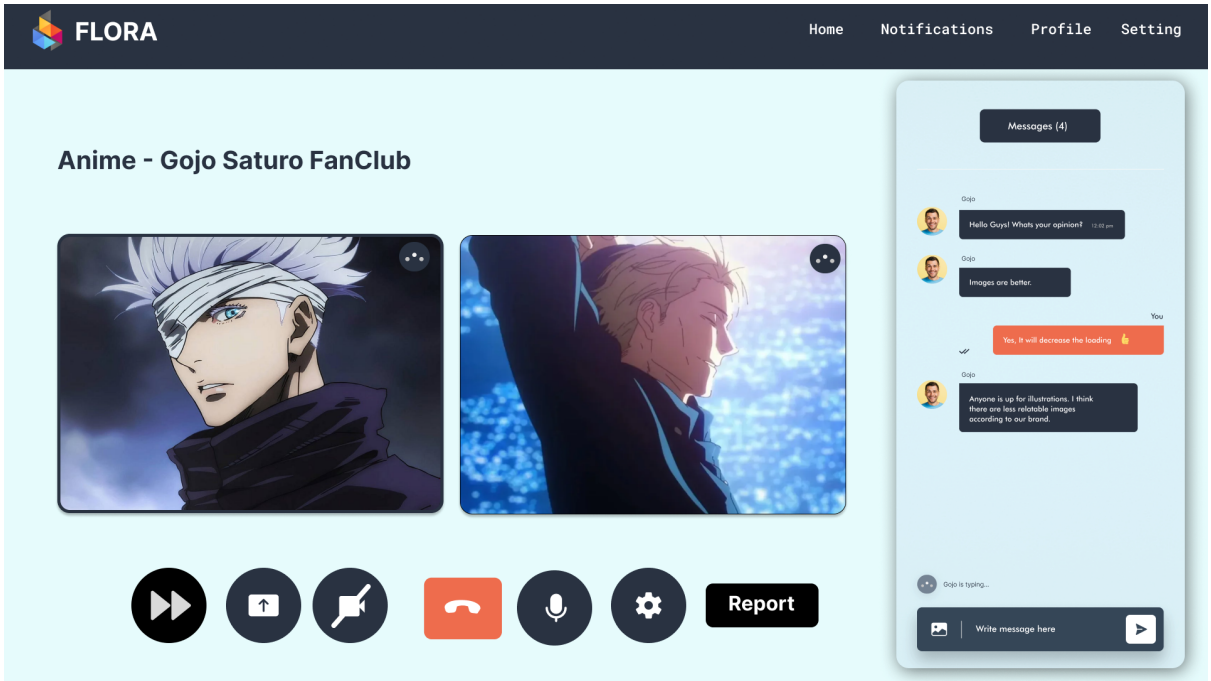

Figure 4.4: Video Call Page

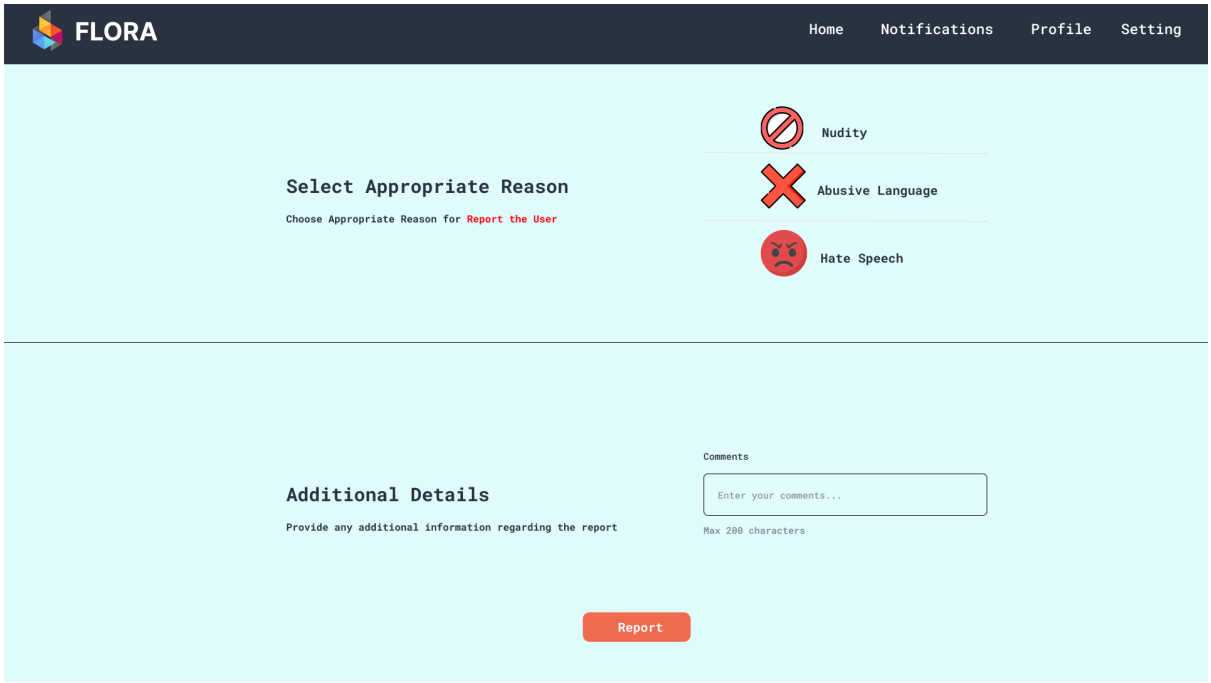

Figure 4.5: Report User Page

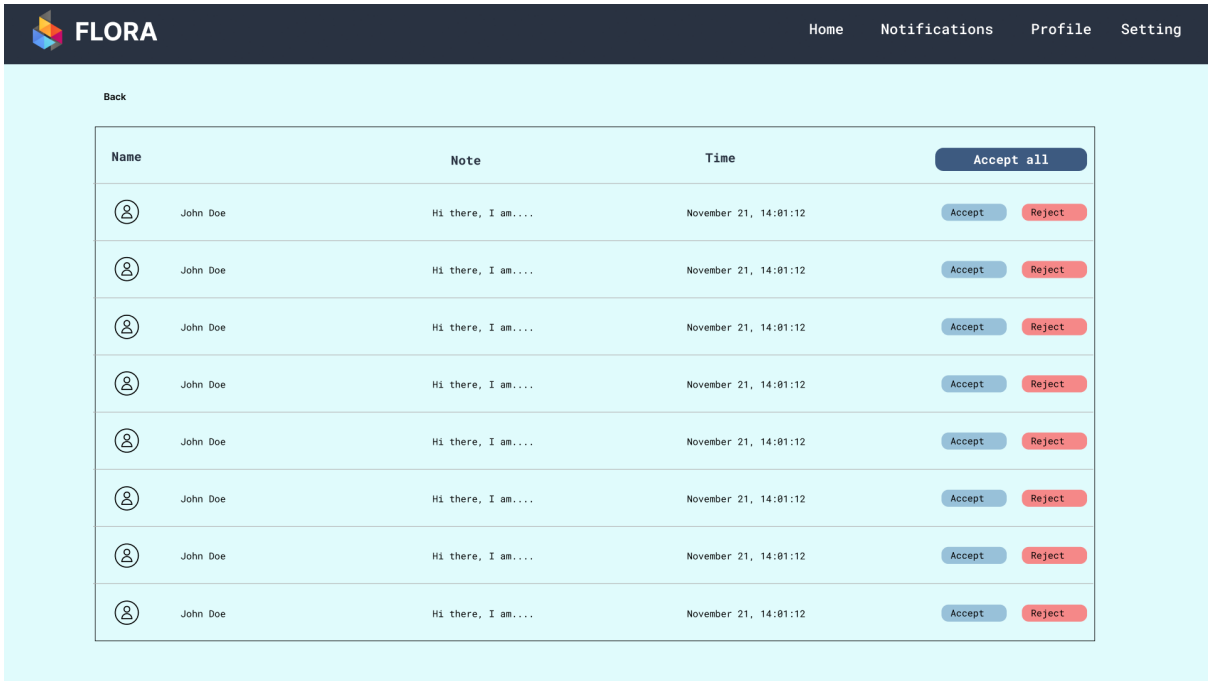

Figure 4.6: Notifications

4.9 Database Design

4.9.1 ER Diagram

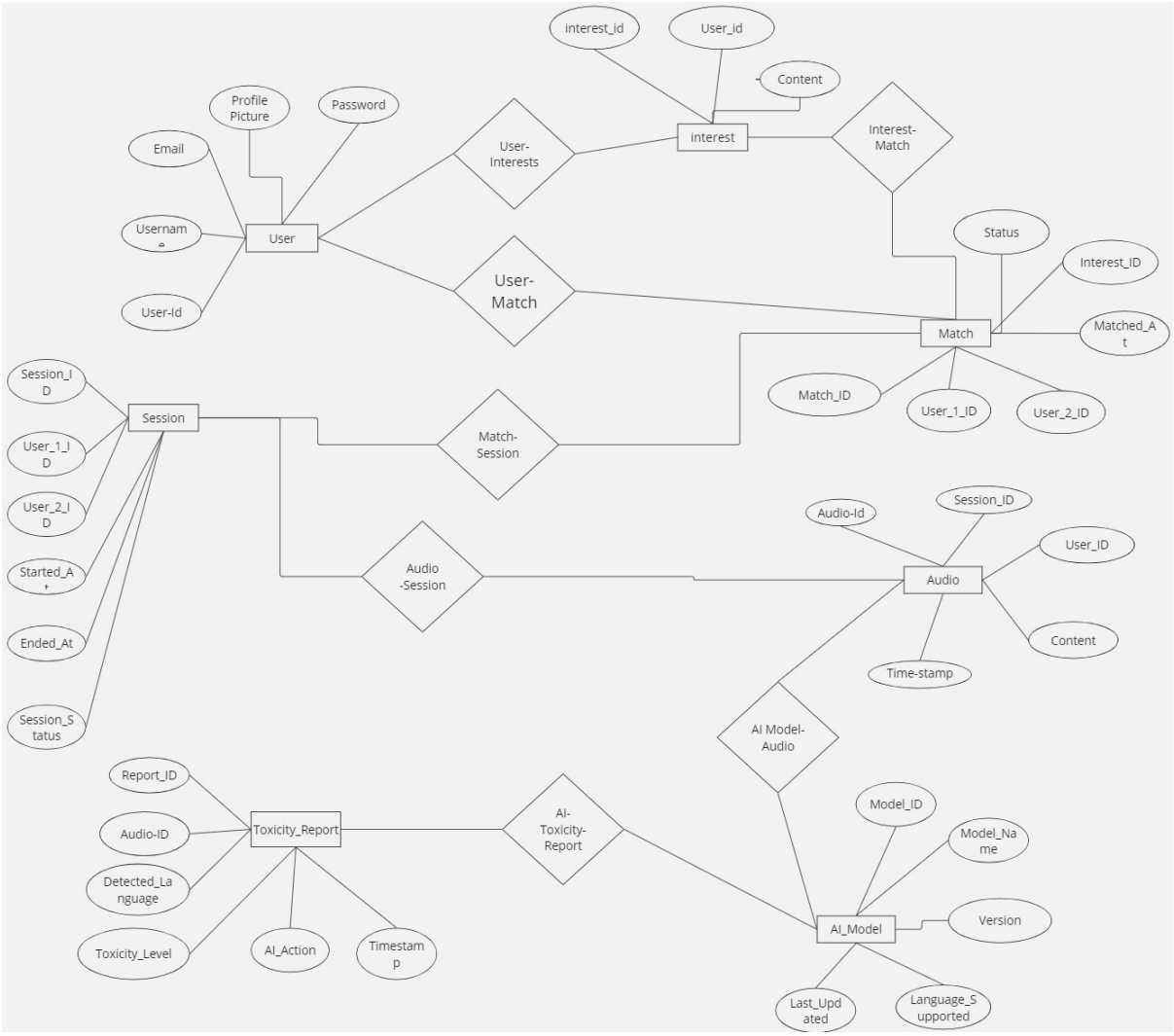

Figure 4.7: ER Diagram

4.9.2 Data Dictionary

Table 4.14: Data Dictionary

Entity	Attribute	Data Type	Description
User	User_ID	Integer	Primary Key.
	Username	Varchar(255)	The display name of the user.
	Email	Varchar(255)	The email address used for account registration.

Continued on next page

(Continued from previous page)

Entity	Attribute	Data Type	Description
	Password	Varchar(255)	The password used for login, securely stored.
	Language	Varchar(100)	The primary language set by the user .
	Profile_Picture	Blob	The profile image uploaded by the user.
	Status	Varchar(50)	Indicates whether the user is online or offline.
Interests	Interest_ID	Integer	Unique identifier for each interest Primary Key.
	User_ID	Integer	Foreign Key linking to the User who created .
	Content	Text	The name of the interest.
Match	Match_ID	Integer	Unique identifier for each match (Primary Key).
	User_1_ID	Integer	Foreign Key referencing the first user.
	User_2_ID	Integer	Foreign Key referencing the second user.
	Interest_ID	Integer	Foreign Key linking to the interests that led to the match.
	Matched_At	Timestamp	The time when the users were matched.
	Status	Varchar(50)	Status of the match.
Video_Chat_Session	Session_ID	Integer	Unique identifier for each video chat session (Primary Key).
	User_1_ID	Integer	Foreign Key referencing the first user in the session.
	User_2_ID	Integer	Foreign Key referencing the second user in the session.
	Started_At	Timestamp	Start time of the video chat session.
	Ended_At	Timestamp	End time of the video chat session.
	Session_Status	Varchar(50)	Status of the session.

Continued on next page

(Continued from previous page)

Entity	Attribute	Data Type	Description
Audio	Audio_ID	Integer	Unique identifier for each audio clip (Primary Key).
	Session_ID	Integer	Foreign Key linking to the video chat session.
	User_ID	Integer	Foreign Key for user from which the message was sent.
	Timestamp	Timestamp	The exact time the message was sent.
Toxicity_Report	Report_ID	Integer	Unique identifier for each toxicity report (Primary Key).
	Audio_ID	Integer	Foreign Key linking to the message flagged for toxicity.
	Detected_Language	Varchar(50)	The language detected in the flagged message.
	Toxicity_Level	Varchar(50)	The severity of the toxicity detected.
	AI_Action	Varchar(50)	The action taken by the AI.
	Timestamp	Timestamp	The time when the report was generated.
AI_Model	Model_ID Version	Integer Varchar(50)	Unique identifier for each AI model, Primary Key. Version of the AI model.
	Language_Supported Last_Updated	Varchar(50) Timestamp	The languages the AI model supports . The last time the AI model was updated.

4.10 Risk Analysis

- **Users Acquisition**: User adoption is one of the major risk for social websites. The UI and UX as well as features unique to Flora should help users adopt Flora.
- **Costs**: There are significant costs for development and maintenance for such a project.
- **Legal Risks**: When handling user data there is a legal risk if there is a complication with the data.
- **Competition**: Right now there are a lot of competing apps such as Chatroulette and Azar Live.

Chapter 5 Proposed Approach and Methodology

The main focus of our research is to contribute to toxic language detection in Urdu considering modalities other than text. Research shows how other modalities like audio and video show an improvement in detection of toxic activities. To the best of our knowledge, no Multi-Modal dataset has been curated in Urdu for this problem, except for Meta's MuTox [1], which is an audio based dataset but has a clear bias towards religious hate and includes translations from Quran and Bible. We find it necessary to have a benchmark dataset to evaluate:

- How fusion of different modalities improve toxicity classification task in Urdu, if at all.
- What aspects of modalities other than text contribute to the detection task.
- Compare combinations of current state of the art text, speech and vision based models for toxicity classification to find the one that performs best overall. Then, trying them out individually to find the modality that contributes the most and which model gives best results for that.

5.1 Overview

The overall process will be divided into data collection, data annotation and trying out already existing models available in literature, tested in a multilingual or Urdu specific setting. The main task would be to find the right combination by training different combinations on toxicity classification task. However, fine-tuning individual models on our dataset is also crucial to check how it improves the result, if at all. The main modalities we will be focusing on currently would be audio and text. The inclusion of video modality will be kept optional as it's incorporation will be recourse intensive. However, if we find the right resources and time for doing that it will be included in the research as well.

5.2 Data Collection

The data collection phase of our research is the most crucial part. As by toxicity we are referring to any kind of hate speech (racial, gender-based, religion-based, political affiliation based, ethnicity based) , profanity, abusive language and pornographic language, our best resort would be to go for social media platforms like Youtube. Also, as we are focusing on the conversational aspect, and making an app like Omegle as a wrapper around our research. The data we would require would mostly be conversational data from platforms like Tiktok (Tiktok Live), Bigo Live and Omegle.

These platforms have abundant Pakistani community involved in toxic conversations. These live videos are mostly screen-recorded and placed on Youtube by many people. Data Can be scraped from Youtube

by using data scraping tools like Selenium or Youtube's Data API by searching keywords based on hate lexicons provided by [10] and additionaly by analyzing transcripts of videos. Other than that for categories like religion-based and political affiliation based hate Pakistani news channels and videos spreading hate against a particular religion and sect will be gathered. To summarize the process here is how the data will be collected:

- Use of hate and profane lexicons along with a specific platform name (in some cases) to search for videos using Youtube's Data API or Selenium.
- Getting audio of top results by using tools like moviePy.
- Performing Speech to text on top results and giving them to gpt3.5 for initial classification.
- Manually including 60 seconds chunks of videos classified as toxic by gpt.
- Making sure there is a good balance between all categories.
- For normal videos without any toxic content only platform based search will be done and videos will be randomly sampled, checked for toxicity, if not found a random chunk will be included as the part of dataset.

5.3 Data Annotation

Data will be annotated by 3 people. 2 of them will be from our group and one additional annotator will be there to ensure there is no bias. A set of 30 videos will be annotated by the 2 members and given to the third annotator for guidance. The annotation process will involve annotating the video as toxic/ non-toxic. The video will further be annotated as the specific type of toxicity described above for analyzing data statistics and multi-class classification later on. The time stamps of the video contributing to the decision of marking it as toxic will also be mentioned. Also, annotators will mention the contributing modality (one or more than one) so that we have clear stats of the data at hand.

The table after annotation will have a structure like this:

Video Path	**Audio Path**	**Transcript**	**Toxic/Non-Toxic**	**Type of toxicity**	**Start End Timestamps**	**Contributing Modality**

5.4 Research Workflow

5.4.1 Testing of individual Modalities

We will start by testing out already existing Urdu specific (ones that handle code-mixed scenarios like English + Urdu) and Multilingual Models available in literature.

5.4.1.1 Text Modality

As there has already been a ton of study in textual domain when it comes to classifying toxicity. Current state of the art models fine-tuned already [10] [16] will be tested for toxicity classification. But as no literature in Urdu exists that targets toxicity as a whole including everything we have included. We might have to fine-tune models like mBERT and XLM-Roberta which have been proven by literature to excel in Urdu specific tasks in general. The focus here would be on two things: One being what models performs best on zero-shot toxicity classification and what compromises are we willing to make in order to find a balance between good results and viable real-time classification solution.

5.4.1.2 Audio Modality

A wide variety of audio features have been studied in literature [8] for audio classification tasks. Some of them are listed below:

- Spectral Centroid is a measure of a spectrum indicating the value of the weighted mean resembling its center of mass in the audio.
- Spectral Rolloff is another measure of a spectrum indicating the proportion of the shape of the signal which defines the frequency below which a specified percentage of the total spectral energy lies
- Spectral Bandwidth is another measure of a spectrum indicating the width of the band of light at one-half the peak maximum or full width at half maximum (FWHM).
- Zero-Crossing Rate is a measure estimating the smoothness of a signal to figure out the quantity of zero crossing inside a segment of that signal.
- Mel Frequency Cepstral Coefficients (MFCCs) of a signal are a small set of features which compactly depict the general state of a spectral envelope.
- Chroma Feature is a feature vector showing how much energy of each pitch class is available in the signal.

An approach that has been used by Iba˜nez et al. (2021) [8] for finding out the best metric for classi-

fication for a certain area of interest is to use Decision Trees. All of these features are obtained for a particular audio using Python's librosa library. After flattening the features, we start making splits based on features giving the purest splits. A cut off for levels is decided and for every feature up until that level percentage is calculated to check what percentage of features were from MFCC, Spectral Roll-off etc.

Similarly we can try out all the 6 characteristics and find out what works best. Other than that CNNs that study spectogrms and wav2vec2.0 has also been used for down stream classification tasks. So, those too will be considered as a part of research.

5.4.1.3 Video Modality

This is an optional part that we are willing to include given we get the time and computational resources to do so. In literature, the approach that is being followed mostly is to sample one to two frames from each second randomly. Use deep CNN architectures like Inception v2 and Resnet-50 or ViT to get encodings for each image individually and then using a sequencial encoder like LSTMs to encode all the images into one single embedding. Which can then be used to perform downstream classification tasks.

5.4.2 Fusion of Modalities

After testing out performance based on individual modalities, we will start combining modalities using early-fusion (combining features from individual modalities before classification i.e combining encodings) or late-fusion (combining results after each modality has made its decision). In case of early fusion the process will include training that final FC layers for classification. While in case of late-fusion a majority vote or a weighted average can be taken.

Most common fusions used in literature include BERT + MFCC + ViT [2]. But different combinations will be tried keeping in view the language at hand.

5.5 Conclusion

To conclude, our focus will mostly be on how audio modality specifically aids in toxicity classification in addition to text and how certain acoustic cues are essential to catch the intent and nature of the discussion. As we will be dealing with using the approach on our one-to-one conversational platform, we have decided to send 60 seconds chunks for conversation with an overlap of 30 seconds for real-time toxicity classification. Our research aims not only to devise the best solution for classifying content as toxic but also to find the one that is most scalable.

Chapter 6 High-Level and Low-Level Design

The main goal is to match people in real-time based on their profiles, and then connect them through a random video chat. While they're talking, a speech processing model will be running in the background, monitoring the conversation for any inappropriate language or behavior.

6.1 System Overview

Provide a general description of the software system, including its functionality and matters related to the overall system and its design (perhaps including a discussion of the basic design approach or organization). Feel free to split this discussion up into subsections (and sub-subsections, etc. ...).

6.2 Design Considerations

This section describes many issues that need to be addressed or resolved before attempting to devise a complete design solution.

6.2.1 Assumptions and Dependencies

6.2.1.1 Assumptions

- The connection between the video chat and the speech processing model will be smooth and won't disrupt the conversation.
- The system will collect data from two types of users, professionals and casual users, who are looking to have topic-based discussions.
- The AI speech model will work without lagging or interfering with the conversation.

6.2.1.2 Dependencies

- **WebRTC:** Handles the live video and audio between users.
- **PostgreSQL:** Stores all the data—like user profiles, chat history, and matching information.
- **Speech Processing AI:** Model that monitors the live conversation to catch anything inappropriate.
- **Other tools:** Other developement tools like redis, celery, react, redux etc.

6.2.2 General Constraints

The following constraints will significantly influence the design and performance of the system:

- **Adequate Resources:** We'll have enough computing power to train our models efficiently.
- **Smooth Performance:** The system should run smoothly, even with unexpected traffic increases, without extra resources.
- **Internet Access:** Users need internet access for the system to work properly.
- **Cross-Platform:** As a web-based system, it will work on all major OS platforms, with exceptions handled as needed.
- **Data Protection:** We must comply with regulations like GDPR and CCPA to protect user data.

6.2.3 Goals and Guidelines

The design of our system is driven by a few key principles to ensure a balance between user experience, performance, and simplicity:

6.2.3.1 KISS (Keep it Simple, Stupid)

We want to keep things simple in terms of design, as well as the way our application works. The system is designed to be simple and easy to lay on with little emphasis in its complication in the development and usage.

6.2.3.2 Performance Over Memory Usage

Our major focus is on velocity and agility. We are ready to consume more memory to get faster and smoother experience at that moment when we are using real time video chats and processing.

6.2.3.3 Consistent User Experience

The system should look and behave the same way on every platform, because users want it to be integrated. This is in order that a user who switches over from one device to another does not have to struggle to familiarize himself or herself with a new device and as a result loses the feel of the rich user experience he or she is used to on another device.

Such goals are achieved to guarantee that the system is efficient, and users find it convenient when performing its operations under particular circumstances.

6.2.4 Development Methods

To develop this project, we are choosing the Agile development methodology. This method enables us to work in cycles – so that the updated and improved versions are provided more often. We have also

divided the design into a set of activities, which would allow the adjustment of the project according to the feedback from users and testing during the work on the project. Other methods like Waterfall were also considered unfit because of flexibility issues. We require the faster and the interlinked revisions of the Agile model due to the several real-time features and speech processing that will be involved in the design of the system.

6.3 System Architecture

For Flora, we plan to use a **monolithic architecture**, consolidating the various components into a single codebase for simplicity and ease of management. We're breaking down Flora into key components, which makes it easier to scale and maintain. The system will operate on a hybrid **Client-Server** and **Event-driven** model for user interactions, while the video calling feature will rely on **Peer-to-Peer** connections via WebRTC. System breakdown will look like:

- **User Management** – Responsible for registration, login, profiles, and user preferences.
- **Matching Engine** – Pairs users based on preferences using NLP.
- **WebRTC Video Calling** – Handles real-time video and audio communication.
- **Moderation** – Ensures a safe environment by monitoring user interactions and content.

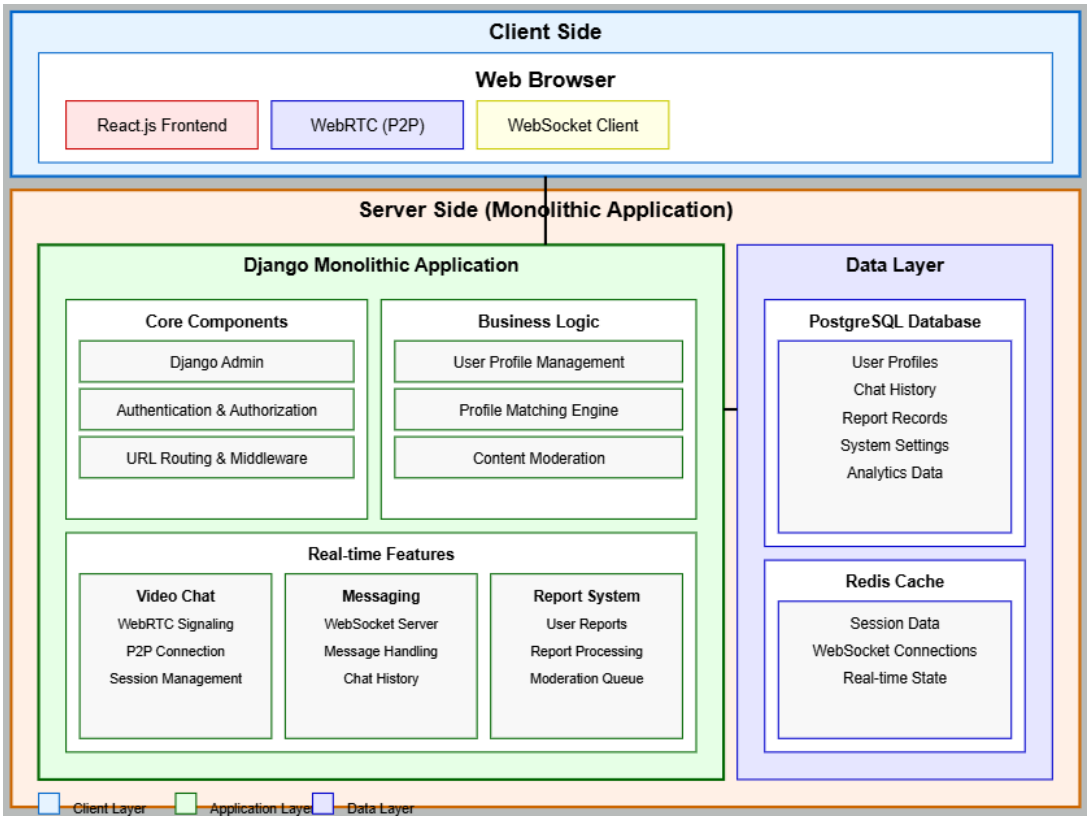

Figure 6.1: Architecture Diagram of System: Flora

6.3.1 Subsystem Architecture

Now let's dive a little deeper into how these components will work together:

6.3.1.1 User Management Subsystem

- It will handle authentication (probably using JWT tokens) and communicate with other services like Matching and Video calling to fetch user data when needed.
- The user interface will send requests to the User Management API, which will validate and process the data before responding.

6.3.1.2 Matching Engine Subsystem

- This subsystem will use NLP algorithms to analyze user interests and preferences.
- It will act as a mediator between users and the video calling service. Once it matches users, it triggers the WebRTC service to establish a connection.
- The Matching Engine will communicate with the User Management service to get the necessary data on user preferences.

6.3.1.3 WebRTC Video Calling Subsystem

- This subsystem will initiate and manage Peer-to-Peer connections between users once they are matched.
- While the WebRTC API will handle the actual transmission of video/audio, it will also interact with the User Management and Matching subsystems to ensure the correct users are paired.
- This part of the system will rely on WebRTC's protocols to ensure smooth video calling and could use STUN/TURN servers to establish connections when necessary.

6.3.1.4 Moderation Subsystem

- The Moderation subsystem is crucial for maintaining a safe and respectful environment for all users.
- It will monitor user interactions and content during video calls and chats, using both automated tools and human moderators to ensure compliance with community standards.
- If any inappropriate behavior is detected, this subsystem will take necessary actions, such as warnings, temporary bans, or permanent removal from the platform, depending on the severity of

the violation.

- We'll be making our system event-driven for moderation subsystem. So if there's something bad detected in audio, an event will be triggered where audio and other info will be stored in database and other work such as cutting call can be done

All of these subsystems will be containerized and managed through an API Gateway to keep things organized.

6.4 Architectural Strategies

Given is a quick breakdown of the key strategies we've decided to go with for Flora. We're focusing on simplicity, scalability, and making sure the platform is both reliable and easy to use.

6.4.1 Choice of Technologies

We've carefully picked technologies that help us build Flora efficiently while keeping things flexible for future updates.

- **Backend**: Django (Python-based web framework)
- **Frontend**: React (for a responsive and dynamic UI)
- **Database**: PostgreSQL for structured data and Redis for in-memory caching
- **Real-time Communication**: WebRTC for video calls, Socket.IO for real-time messaging
- **Containerization and Deployment**: Docker for packaging, NGINX as a reverse proxy, and Celery + Redis for background task management. Deployment is not included as the project deliverable, but can be done after successful project completion
- **Moderation**: NLP tools like Pytorch or spaCy for moderation, with event-driven triggers for inappropriate content detection

6.4.2 Hybrid Architecture Approach

We're going with a hybrid model that combines **Client-Server** for traditional request-response tasks (like loading a user's dashboard) and **Event-Driven** for real-time features (such as audio chat moderation).

6.4.2.1 Reasoning

- The client-server model is reliable for handling structured requests like logging in, browsing categories, and fetching data.

- The event-driven part kicks in for instant actions like sending alerts when bad speech or inappropriate content is detected during a chat or video call.
-

6.4.3 Using Open-Source Technologies

Flora is built using open-source tools because they're free, well-documented, and scalable.

6.4.3.1 Reasoning

- Django and React are proven, robust frameworks with large communities for support.
- Tools like Redis and WebRTC give us great performance for real-time communication, which is critical for Flora's core feature set.

6.4.4 User Design

The design of Flora's user interface prioritizes speed and simplicity. A user who's signed in should be able to start a video chat within **a few seconds** of opening the app.

6.4.4.1 Reasoning

- Reducing the time between opening the app and starting a conversation is key to making Flora feel seamless.
- We'll streamline navigation and avoid unnecessary steps, focusing on getting users into their preferred category and starting a conversation quickly.

6.4.5 Concurrency and Synchronization

Flora handles multiple tasks simultaneously to ensure smooth operation during video calls, while also managing real-time moderation.

6.4.5.1 Reasoning

- **Video and Chat**: WebRTC will handle live video communication, and text chat will be synced using Socket.IO, ensuring everything runs smoothly.
- **Moderation**: As users chat, our event-driven moderation system works in the background, processing speech and text concurrently, and triggering actions if something inappropriate happens.

6.4.6 Future Plans for Extending or Enhancing the Software

Right now our main focus is audio moderation. Additionally, text chat could be moderated during the project. But that wont be enough if we want a complete software. That's why we plan to add realtime video moderation, because that's where most of the profanity, vulgarity, nudity occurs.

6.4.6.1 Reasoning

- The video moderation will run concurrently with the existing audio and text chat moderation.
- We'll use computer vision techniques, potentially leveraging PyTorch or a similar deep learning framework, to analyze video content in real-time.

6.5 Domain Model/Class Diagram

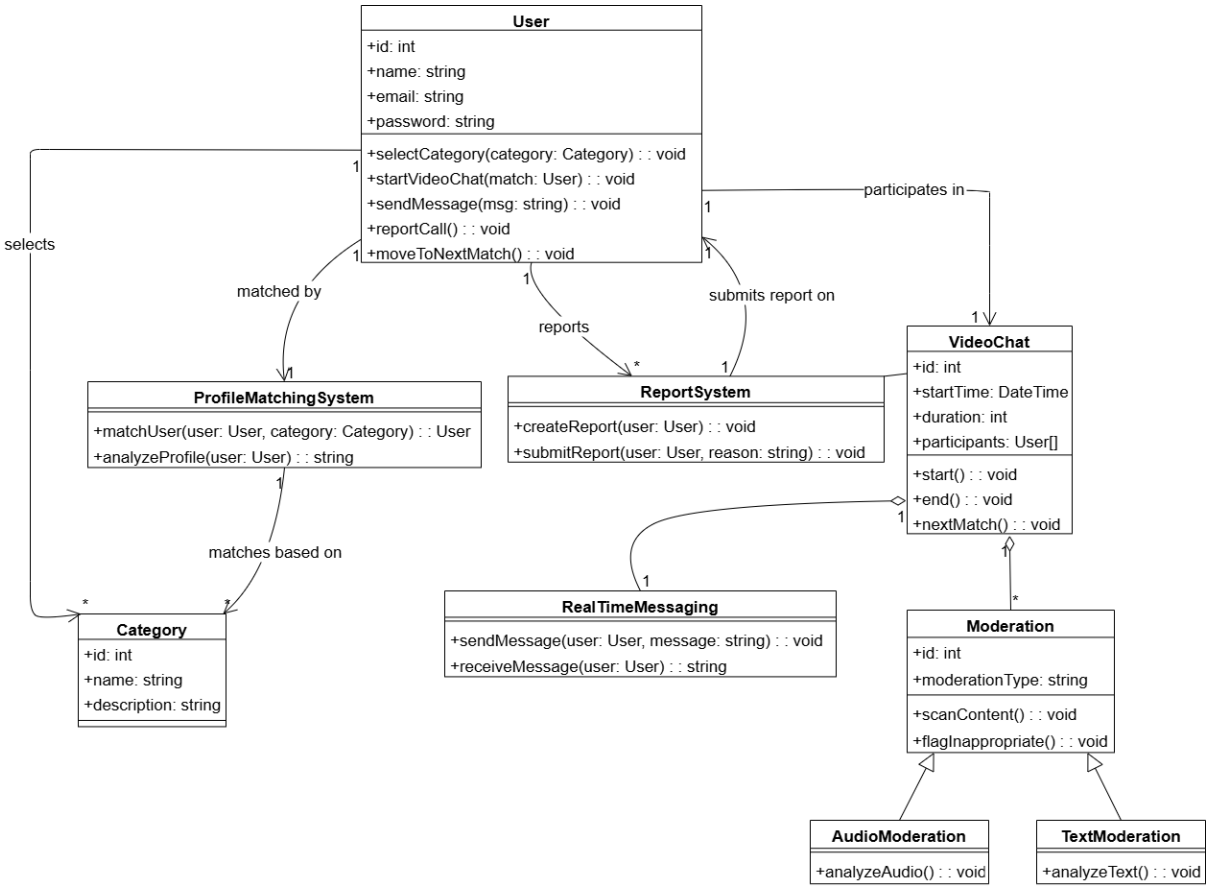

Figure 6.2: Class diagram of Flora's Complete System

6.6 Sequence Diagram

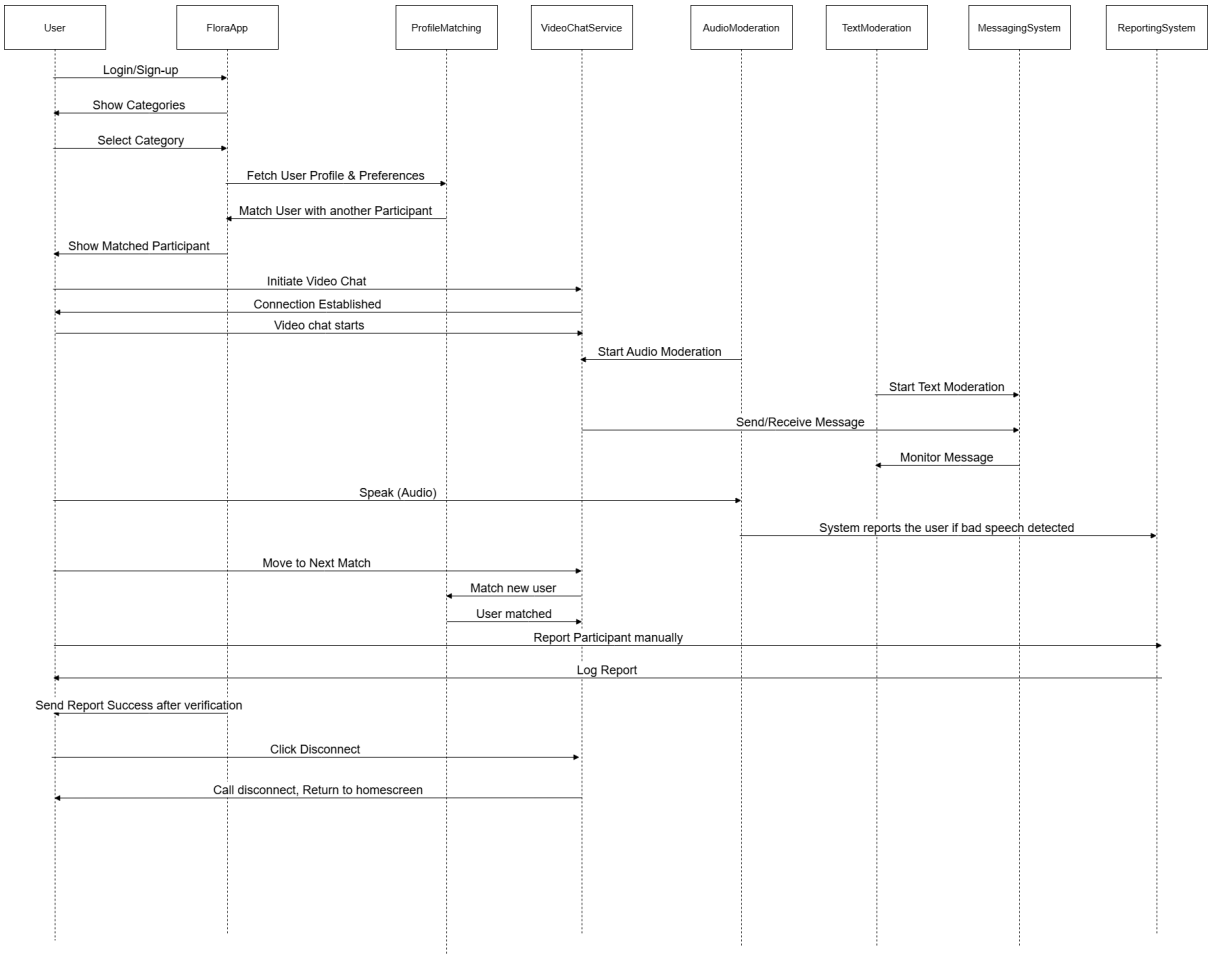

Figure 6.3: Class diagram of Flora's Complete System

6.7 Policies and Tactics

This section covers the decisions we've made during the development of Flora. While these decisions might not shake up the overall architecture, they're still crucial to how things function and how the system is built.

6.7.1 Product Choices

We went all-in with **Python** for the backend because of how well it integrates with AI, which is a big part of Flora's future. We also chose **PostgreSQL** as the database because of its compatibility with **Django** and the way it handles data efficiently, especially when scaling up. When it comes to moderation, we'll be using **PyTorch**. We picked it over other options because it gives us more control and the ability to customize things exactly the way we need for Flora's AI-driven moderation features. For collaborative

coding, we'll use **GitHub**, which makes it easy to roll back if something goes wrong.

6.7.2 Engineering Trade-offs

Hybrid architecture will help in scalability but it will add more complexity in development and maintenance. Other than that, We've decided to keep things as straightforward as possible for now. For example, while automated tools for testing could save time, we'll mostly be writing our own test cases to ensure everything works exactly as intended. **Django**'s built-in testing features are great for this. To avoid overcomplicating package management, we'll stick to either **Poetry** or **Pip**. We'll also use tools like **Black**, **isort**, and **flake8** to keep the code clean and consistent. We don't want our team to waste time dealing with messy code.

6.7.3 Testing and Maintenance

Testing is a big deal for for creating a reliable system, and we'll do it in stages:

- **Unit Testing** will help us test each part of the system on its own. We want to make sure that video chat, messaging, and moderation work exactly as they should.
- **Integration Testing** will check if these modules play well together.
- **Load Testing** will make sure the system can handle a large number of users chatting at the same time without crashing. Of course, This one will be done after deployment with is not the deliverable for this project.

Most of the testing will be done manually i.e Team will be writing test-cases itself, though we're open to using automated tools as needed. We'll also make regular updates to keep things running smoothly.

6.7.4 Project Management and Workflows

We plan to hold **Scrum meetings** to stay on top of things, and **code reviews** will help catch any issues early. For work management, we'll use tools like **Jira** and **Slack** to keep everything organized. Though these steps demand extra time and effort but they'll help us stay on the same page and make sure deadlines are met.

6.7.5 Frontend and Backend Separation

One of our key decisions is keeping the frontend and backend separate. This will give us more flexibility and allow us to work on them independently. **React** powers the frontend for a fast, responsive user experience, while **Django** takes care of everything on the backend. We considered a monolithic architecture where both parts are tightly integrated, but that would limit our ability to make independent updates and

slow down our development cycles. By separating them, we can choose the best technologies for each and enhance overall performance.

6.7.6 System Protocols

The matching system uses user profile data to match people with similar interests in the categories they choose. This ensures a smoother connection process and kicks off the video chat session as soon as a match is made. Meanwhile, moderation will run in the background, monitoring both the video and chat content in real time.

6.7.7 Code Organization

We've kept the structure of the code neat and organized. It's split into clear modules, like frontend, backend, user matching, and moderation. This makes it easier for future developers to jump in and find what they're looking for without getting lost in the code.

Chapter 7 Implementation and Test Cases

This chapter presents the technical implementation details and test cases for Flora, focusing on how its features were developed and validated. Unlike the design phase, this section delves into the backend algorithms and processes not covered earlier, offering insights into the underlying mechanisms driving the platform's functionalities.

7.1 Implementation

This section outlines the progress made in developing Flora and provides a detailed breakdown of its prototype and the core functionalities implemented. It focuses on the algorithms, technologies, and procedural approaches adopted to achieve the platform's goals.

7.1.1 Authentication

The authentication system for Flora is implemented using Django Rest Framework (DRF) and Djoser.

7.1.1.1 Signup Process

- Users provide their email and password to create an account.
- A verification email with a unique tokenized link is sent.
- The account is activated only when the user clicks the verification link, which is created using Django's token generator.

7.1.1.2 Login Process

- After email verification, users can log in using their email and password.
- If the credentials are correct, Djoser generates a JWT token for authentication.
- The token is used for subsequent requests, eliminating the need for the user to log in again.

7.1.1.3 Forgot Password

- Users can request a password reset link by submitting their email address.
- A reset token is generated using Django's `PasswordResetTokenGenerator` and sent.
- The user can reset their password using this token.

7.1.2 Profile Settings

The profile settings allow the user to manage their personal information and account details. This option will only be available for logged in user.

7.1.2.1 Edit Profile

- Users can edit their profile information, such as their name, username, and bio.
- The email cannot be changed through the profile settings, as it is tied to the user's login credentials and email verification.
- All changes to the profile are saved and updated in the database upon submission.

7.1.2.2 Change Password

- Users can change their password through the change password section in settings.
- They must first provide their old password, which is verified using Django's `check_password`.
- Afterward, users can enter a new password, which is hashed and securely stored in the database.

7.1.2.3 Delete/Deactivate Account

- Users have the option to delete or deactivate their account directly from the profile settings.
- If they click on "Delete Account", there account will be deleted, they'll be signed out. Database will mark the user's record and deleted
- Deactivating the account disables the user's access to the platform without deleting their data, allowing them to reactivate their account in the future.

7.1.3 Implementation of User Matching in Flora

7.1.3.1 Random Connections with Waiting List

- When a new user signs up for Flora, and click on a certain category, they are added to a waiting list based on their selected category.
- If another user joins the same category, the system randomly connects the two users from the waiting list.
- This random matching functionality is the current implementation of user connections in the app.

7.1.4 Video Calling

The video calling feature allows users to engage in real-time video and audio communication using WebRTC technology. Once the connection is established, users are able to interact with each other through live video and audio streams.

7.1.4.1 Call Connection and Video Streaming

- Once the call is initiated, WebRTC establishes a peer-to-peer connection between the two users.
- WebRTC handles the transmission of both video and audio streams between the users, ensuring low-latency communication.
- The system continuously adjusts the video and audio quality based on network conditions to provide the best possible experience for both participants.

7.1.4.2 In-Call User Options

- During the video call, users can control their video and audio settings:
  - **Turn Video On/Off**: Users can choose to enable or disable their video feed at any point during the call.
  - **Mute/Unmute Audio**: Users have the option to mute or unmute their microphone as needed.
- Additionally, users can exit the call or move to the next call after ending the current session.

7.1.5 Text Messaging During Video Chat

The text messaging feature allows users to send and receive messages while on a video call.

- The side panel is integrated into the video call interface, providing space for users to type and send messages without interrupting the video feed.
- A WebSocket connection is used to send and receive messages in real-time.
- The chat messages are stored temporarily on the server during the call and are displayed in the side panel for both users. These chat will not be stored permanantly on database

7.1.6 Report User

Report feature ensures that users can flag behavior that violates platform guidelines, helping maintain a safe and respectful environment.

7.1.6.1 Manual Reporting

- During the video call, users can click the ”Report User” button available in the call interface.
- Once clicked, a popup appears asking the user to specify the reason for their report (e.g., offensive language, inappropriate behavior, harassment).
- The user selects or types the reason for reporting, and the information is captured by the backend.

7.1.6.2 Session Storage

- After the report is submitted, the video call session, including the report details, is stored in the database for further review.
- The session is saved temporarily, and all relevant data, including the reported reason, is logged for future reference.

7.1.7 Real-Time Audio Toxicity Detection

The real-time audio toxicity detection feature ensures a safe environment by identifying and moderating toxic language during video calls.

- Audio streams are captured via WebRTC and processed to detect toxicity in real-time, supporting mixed-language (Urdu-English) communication.
- The system uses pre-trained models (e.g., Wav2Vec for audio and XLM-RoBERTa for text) to analyze audio and transcribed speech, flagging toxic content based on a predefined threshold.
- If toxicity is detected, a warning is displayed to the offending user, and severe cases may mute the audio or end the call, with notifications sent to both users (see Notifications).
- Toxic incidents are logged in PostgreSQL and linked to the ”Report User” feature for further review (see History Pages).

7.1.8 Notifications

The notifications feature provides users with real-time updates about their actions and interactions on the platform, enhancing user engagement and awareness.

- Notifications are triggered by user actions such as signing up, changing passwords, editing profiles, initiating/ending calls, reporting users, or receiving moderation actions.
- The Django backend generates notification messages using predefined templates (e.g., ”Your password has been successfully changed” or ”Your report against [user] has been sent to admins”),

with Celery handling asynchronous delivery.

- The notifications page lists all recent updates with timestamps, allowing users to scroll through their activity history.

7.1.9 History Pages

The history pages feature provides users with records of their past interactions and reports, promoting transparency and accountability.

7.1.9.1 Call History

- After each call ends, details like the matched user's username, duration, date, and category are logged in PostgreSQL.
- A notification confirms the call's addition to the history (see Notifications).
- The call history page on the React frontend displays a list of past calls, accessible via a REST API.

7.1.9.2 Report History

- When a user submits a report, the reported user's ID, reason, and date are logged in PostgreSQL, with a notification sent (see Notifications).
- Moderation outcomes (e.g., warning issued) are updated in the report record.
- The report history page lists all reports and actions taken, fetched via a REST API on the React frontend.

7.2 Test case Design and description

This section outlines the test cases designed for various features of the Flora system. These test cases validate the core functionalities and ensure system reliability. Common attributes such as input constraints, environment setup, and procedural steps are covered.

Table 7.1: Registration Test Case

User Registration Component			
User Authentication Module			
Test Case ID:	*001*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Registration*
Revision History:	*None*		
Objective:	*To verify that a new user can successfully register with valid credentials*		
Product/Ver/ Module:	*Flora/1.0/Authentication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*User has internet access and can reach the registration page*		
Pre-Requisite:	*The app is accessible and the registration page is available*		
Step No.	**Execution description**	**Procedure result**	
1	Navigate to the registration page	Registration page is displayed	
2	Enter valid email: test@example.com	Email is accepted	
3	Enter valid password: Test@123	Password is accepted	
4	Click the ”Register” button	System shows success message and redirects	
Comments: Email verification link is sent to the provided email address			
Passed			

Table 7.2: User Login Test Case

User Login Component			
User Authentication Module			
Test Case ID:	*002*	**QA Test Engineer:**	*Suhaib Rashid*
Test case Version:	*1.0*	**Reviewed By:**	*Zaid Asif*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*User Login*
Revision History:	*None*		
Objective:	*To verify that a registered user can successfully log in to the system*		
Product/Ver/ Module:	*Flora/1.0/Authentication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*User is already registered with valid credentials*		
Pre-Requisite:	*User has a verified account in the system*		
Step No.	**Execution description**	**Procedure result**	
1	Navigate to the login page	Login page is displayed	
2	Enter registered email: test@example.com	Email is accepted	
3	Enter correct password: Test@123	Password is accepted	
4	Click the ”Login” button	System validates credentials	
Comments: User should be able to access all authorized features after successful login			
Passed			

Table 7.3: User Logout Test Case

User Logout Component			
User Authentication Module			
Test Case ID:	*003*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*User Logout*
Revision History:	*None*		
Objective:	*To verify that a logged-in user can successfully log out of the system*		
Product/Ver/ Module:	*Flora/1.0/Authentication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*User is currently logged in to the system*		
Pre-Requisite:	*User has an active session and can access the navigation bar*		
Step No.	**Execution description**	**Procedure result**	
1	Navigate to any page with the logout button	Page is displayed	
2	Click the ”Logout” button in the top-right corner	System clears JWT token and session data	
3	System redirects to login page	Login page is displayed	
4	Attempt to use browser back button	System prevents access to pages	
Comments: User should not be able to access authenticated features after logout			
Passed			

Table 7.4: User Reporting Feature Test Case

User Reporting Component			
Content Moderation Module			
Test Case ID:	*004*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*User Reporting*
Revision History:	*None*		
Objective:	*To verify that users can report inappropriate behavior during chat sessions*		
Product/Ver/ Module:	*Flora/1.0/Content Moderation*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*Reporting system is integrated and moderator review process is established*		
Pre-Requisite:	*User is in an active chat session with another user*		
Step No.	**Execution description**	**Procedure result**	
1	User clicks the ’Report’ button in the chat interface	Report dialog box appears with options	
2	User selects report category (e.g., harassment)	Category is selected	
3	User adds details in comment field	Text is accepted in the description field	
4	User submits the report	System confirms submission	
5	Check moderator panel	The report appears for review	
Comments: System should capture chat transcript and related metadata when report is submitted			
Passed			

Table 7.5: Text Messaging Test Case

Text Messaging Component			
Communication Module			
Test Case ID:	*005*	**QA Test Engineer:**	*Muhammad Suhaib Rashid*
Test case Version:	*1.0*	**Reviewed By:**	*Zaid Asif*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Text Messaging*
Revision History:	*None*		
Objective:	*To verify real-time text messaging functionality during video calls*		
Product/Ver/ Module:	*Flora/1.0/Communication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*WebSocket connection is stable*		
Pre-Requisite:	*Two users are in an active video call (Test Case 009)*		
Step No.	**Execution description**	**Procedure result**	
1	User A opens chat panel during call	Messaging sidebar expands with input field	
2	User A types ”Hello, can you hear me?”	Text appears in input field with typing indicators	
3	User A sends message	Message appears in both users’ chat	
4	User B replies with ”Yes, loud and clear”	Response appears in User A’s panel	
5	Test special characters (&, é,)	All characters render correctly for both users	
Comments: Message history should persist for entire call duration			
Passed			

Table 7.6: Frontpage Display Test Case

Frontpage Display Component			
User Interface Module			
Test Case ID:	*006*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Muhammad Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Frontpage Display*
Revision History:	*None*		
Objective:	*To verify that authenticated users can access and navigate the frontpage*		
Product/Ver/ Module:	*Flora/1.0/User Interface*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*User is logged in (Test Case 002)*		
Pre-Requisite:	*Active user session exists*		
Step No.	**Execution description**	**Procedure result**	
1	User logs in successfully	System redirects to personalized frontpage	
2	Verify all UI components load	Navigation bar, category selection panel, and quick-access buttons appear	
3	Click ”Notifications” section	System redirects to notifications page	
4	Return to frontpage via navigation bar	Frontpage reloads with consistent layout	
Comments: All clickable elements should respond within 500ms			
Passed			

Table 7.7: Video Call Page Test Case

Video Call Component			
Communication Module			
Test Case ID:	*007*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Video Call Page*
Revision History:	*None*		
Objective:	*To verify matched users can establish video calls*		
Product/Ver/ Module:	*Flora/1.0/Communication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*Both users have granted camera/microphone permissions*		
Pre-Requisite:	*Two users are matched via category selection*		
Step No.	**Execution description**	**Procedure result**	
1	Both users accept match notification	Video call interface loads for both parties	
2	Verify local video feed	Self-view displays live camera feed with 2s latency	
3	Verify remote video feed	Partner's video appears with 3s latency and 15fps	
4	Test microphone with speech	Audio transmits clearly without echo	
5	Click "End Call" button	Call terminates and system returns to frontpage	
Comments: WebRTC connection should maintain stability for 10min calls			
Passed			

Table 7.8: Email Verification Test Case

Email Verification Component			
Authentication Module			
Test Case ID:	*008*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Email Verification*
Revision History:	*None*		
Objective:	*To verify users can complete email verification after registration*		
Product/Ver/ Module:	*Flora/1.0/Authentication*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*Test email account is accessible*		
Pre-Requisite:	*User completed registration (Test Case 001)*		
Step No.	**Execution description**	**Procedure result**	
1	Check test email inbox for verification message	Email arrives with valid link	
2	Click verification link in email	System redirects to success page	
3	Attempt to log in with verified account	System grants full access to features	
4	Resend verification email via UI	New email arrives with working link	
Comments: Links should expire after 24 hours for security			
Passed			

Table 7.9: Profile Matching Test Case

Profile Matching Component			
Matching Algorithm Module			
Test Case ID:	*009*	**QA Test Engineer:**	*Suhaib Rashid*
Test case Version:	*1.0*	**Reviewed By:**	*Zaid Asif*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Profile Matching*
Revision History:	*None*		
Objective:	*To verify the system suggests compatible matches based on profile data*		
Product/Ver/ Module:	*Flora/1.0/Matching Algorithm*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*Database contains multiple user profiles with varied interests*		
Pre-Requisite:	*Test user has completed profile (Test Case 012)*		
Step No.	**Execution description**	**Procedure result**	
1	Navigate to ”Find Matches” section	System displays match suggestions within 5 seconds	
2	Verify match compatibility scores	Scores (70-100%) reflect shared interests from profile data	
3	Filter matches by specific interest	System updates list showing only relevant profiles	
4	View match profile details	Complete profile information displays correctly	
Comments: Algorithm should prioritize active users in suggestions			
Passed			

Table 7.10: Update Profile Test Case

Update Profile Component			
User Profile Module			
Test Case ID:	*010*	**QA Test Engineer:**	*Zaid Asif*
Test case Version:	*1.0*	**Reviewed By:**	*Suhaib Rashid*
Test Date:	*15/04/2025*	**Use Case Reference(s):**	*Update Profile*
Revision History:	*None*		
Objective:	*To verify users can modify and save profile information*		
Product/Ver/ Module:	*Flora/1.0/User Profile*		
Environment:	*Chrome and Edge browsers on Windows/MacOS*		
Assumptions:	*User has edit permissions for their profile*		
Pre-Requisite:	*User is logged in (Test Case 002)*		
Step No.	**Execution description**	**Procedure result**	
1	Navigate to ”My Profile” page	Profile editor loads with current data	
2	Update bio field with new text (50 chars)	Text field accepts input and shows character count	
3	Upload new profile picture (JPG ¡2MB)	Image uploads and thumbnail updates	
4	Add new interest tag ”Photography”	Tag appears in profile and saves to database	
5	Click ”Save Changes”	System confirms update and reflects changes	
Comments: Changes should propagate to match suggestions within 1 hour			
Passed			

Chapter 8 Experimental Results and Discussion

In this chapter we give a detailed overview of the experiments we did and the results we were able to achieve. We will explain how fusing audio encodings with text encodings for classification improved our results as compared to only text based classification. We also give reasons why text as a standalone modality does not work in out case.We used the standard classification metrics i.e Accuracy, Precision, Recall and F1 score to evaluate our models. We provide results for all the models and combination of models fine-tuned during our experimentation.

8.1 Evaluation Metrics

Here is a brief overview of all the evaluation metrics that have been used to evaluate our experiments.

8.1.1 Accuracy

The accuracy of a classification model is an important metric to evaluate its overall performance. It is calculated by dividing the total number of correctly predicted instances by the total number of instances in the dataset. The formula for accuracy is given as:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \tag{8.1}$$

8.1.2 Precision

Precision measures a model's ability to correctly identify positive outcomes from the predictions it has made. It reflects how many of the predicted positive cases are actually correct. The formula for precision is as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{8.2}$$

8.1.3 Recall

Recall measures the model's ability to correctly identify all relevant instances that actually belong to the positive class. It is calculated as the ratio of the number of true positive predictions to the sum of true positives and false negatives:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{8.3}$$

8.1.4 F1 Score

The F1 score is the harmonic mean of precision and recall. It provides a balance between the two metrics The formula for the F1 score is:

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8.4}$$

8.1.5 Evaluation for Speech-to-Text Models

One of the reasons we opted for multi-modal classification was that ASR models perform poorly on the real-world audios, especially the ones which are seen in case of normal casual conversations where there is an overlap between utterances very often. As [17] conducted experiments on different ASRs and found Whisper to be the best for conversational scenarios, we also decided to evaluate the outputs of different variants of Whisper (small, medium and large) and Seamlessm4t for out data. But as writing texts for almost 20 hours of audio data is a tedious task we only annotated 10% of the entire data. We randomly sampled 10% audios from each category and asked our annotators to write text for each of the audio samples. Then we used WER metric to compare outputs of all models mentioned above. Our results found out that Whisper-large-v3 outperforms all other Whisper variants and Seamlessm4t by a large margin. But this does not solve the problem as the transcriptions by Whisper-large-v3 were also full of errors.

8.2 Experimantal Setup

The audio dataset collected was transcribed to text using Whisper-Large-v3 and Seamless-m4t-v2-large. The text was obtained for both toxic and non-toxic classes. Then we fine-tuned XLM-Roberta and Multilingual BERT for the Binary Classification task at hand. We trained XLM-Roberta for 15 epochs using AdamW optimizer with learning rate of 1e-4. Multilingual BERT was trained using Adam optimizer for 11 epochs with a learning rate of 1e-4. The batch size used was 32 for both models. Then we encode audios using Wav2Vec and removed the classification layer from fine tuned XLM-Roberta and Multilingual BERT to obtain text encodings and fused both vectors. These were then trained on classification task using Feed Foward Neural Network. For current experiment we used 2 hidden layers with size 1024 and 512 respectively with ReLU as activation. This architecture was trained using SGD optimizer with a learning late of 0.01.

8.3 Results

The results have been documented in the order they were obtained in. As we used samples from the Religious part first (Toxic and Non-Toxic) for Human Evaluation first we document that at the very start. Then as the any models were fine-tuned, the experiments were evaluated using the above mentioned metric i.e Accuracy, Precision, Recall and F1 score. This extensive evaluation was done to test what combination or standalone models work best for Urdu-English conversational audio classification.

8.3.1 Speech-to-Text Results

Whisper-v3-large outperfomed Seamless-m4t-Large and other Whisper variants by a significant margin. We observed that Seamless-m4t only works well for audios that are made in a controlled environment. In case of any noise or overlap in conversation it just starts to repeat words. Whisper-v3-Large gave WER score of 32.06 which is significantly better than other models.

8.3.2 Text Based Classification Results

XLM-Roberta-Base Model performed better than Multilingual BERT. The evaluation metrics and their respective results and given in Table 8.1.

Table 8.1: Performance metrics for XLM-Roberta and Multilingual-BERT

Models	Accuracy	Precision	Recall	F1 Score
XLM-Roberta	0.803	0.806	0.793	0.799
Multilingual-BERT	0.755	0.863	0.603	0.710

The figures below show how the loss reduced during training for both models.

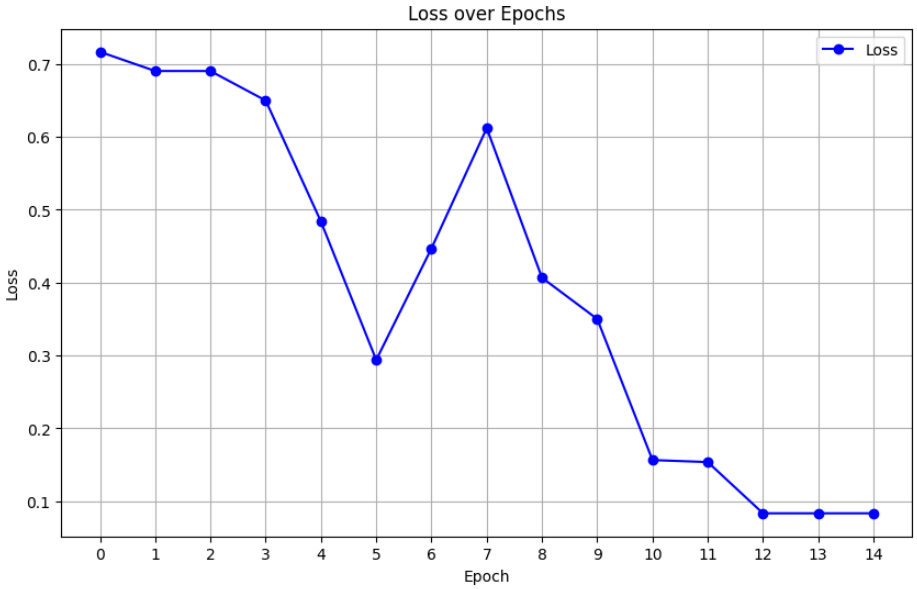

Figure 8.1: XLM-Roberta Training Loss Graph

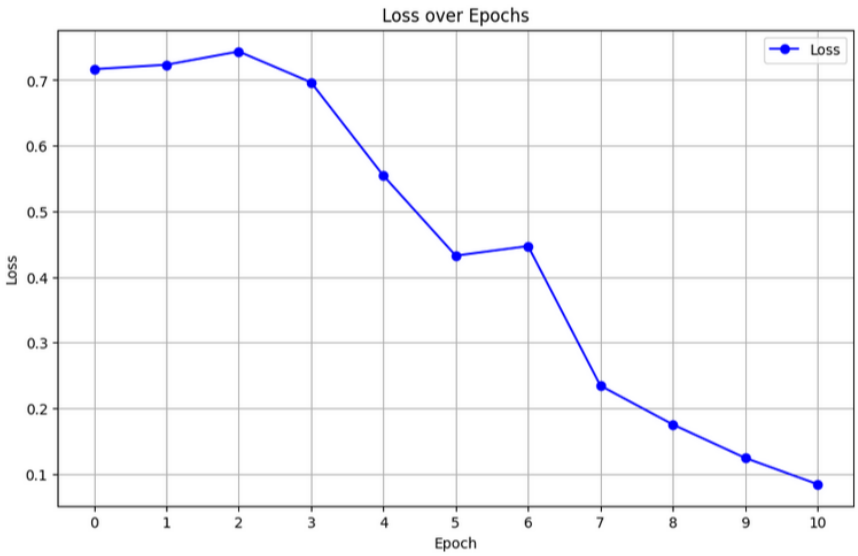

Figure 8.2: Multilingual BERT Training Loss Graph

8.3.3 Audio Based Classification Results

Resnet-50 applied on the spectrograms obtained from the audios gave results quite close to what text-based classifiers were able to achieve, highlighting the importance of audio in improving toxicity detection systems.

Table 8.2: Performance metrics for Resnet-50

Models	Accuracy	Precision	Recall	F1 Score
Resnet-50	0.8059	0.8058	0.8059	0.8055

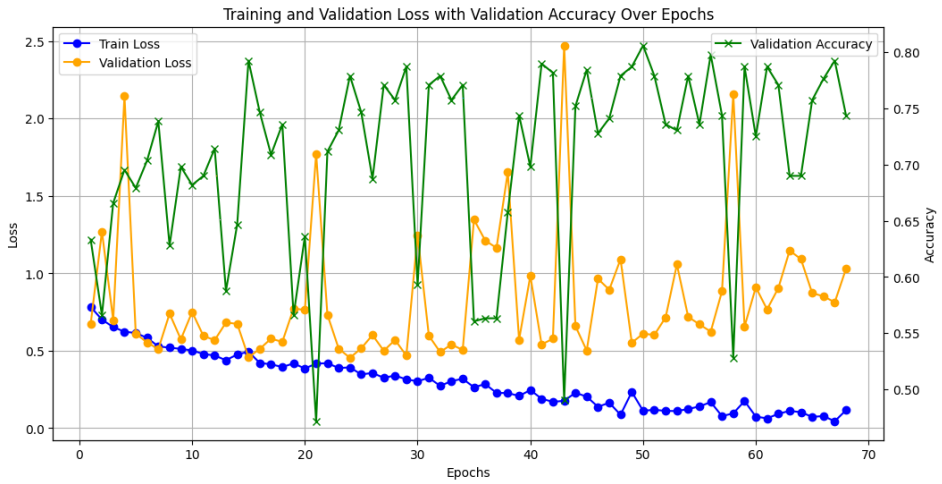

Figure 8.3: Resnet-50 Training and Validation Loss Graph

8.3.4 Classification Results after Fusion of Modalities

As the the above results show that although text based models seem to be performing quite well considering the not so good performance of ASR models, there is still a room for improvement. To do that we added audio as a modality aiding in classification. The results of the combined models are given in Table 8.2.

Table 8.3: Performance metrics for (XLM-Roberta and Multilingual-BERT) + Wav2Vec

Models	Accuracy	Precision	Recall	F1 Score
XLM-Roberta + Wav2Vec	0.872	0.886	0.853	0.869
Multilingual-BERT + Wav2Vec	0.845	0.832	0.811	0.821

These results clearly show that audio can be used to improve classification performance in Toxic Language Detection tasks.

8.4 Conclusion

To conclude we not only show that audio as a contributing factor can improve classification accuracy but we also curated a benchmark dataset for evaluating Toxic Language and Hate Speech Detection models in real world scenarios where data is not always clean and noise free. A lot of research has been done to find the models that work best for a low resource language like Urdu. We believe with further improvement in the dataset and further augmentation the results obtained can be significantly improved making AI more usable in real world scenarios.

Chapter 9 Conclusions

9.1 Conclusion

The development of Flora is still ongoing, and significant progress has been made during FYP-1. We have successfully implemented core features like user authentication, user matching, profile settings, video calling, and text messaging during video chats. These features provide a functional base for the platform and have been tested to ensure proper operation.

One of the main challenges we faced was the collection of audio data for toxicity detection in Urdu, as there were no pre-existing datasets available online. To overcome this, we had to collect our own data, which took considerable time and effort but was crucial for the project's progress.

While we managed to implement several key features, the integration of the real-time toxicity detection model during video calls could not be completed in FYP-1, since we were collecting data too. This functionality will be developed in FYP-2.

9.2 Future Work

For FYP-2, the main objectives include the completion of the toxic speech detection system, particularly the integration of the machine learning model for real-time detection during video calls. After it's completed, thorough testing of the system will be done

In Future, we will also enhance the platform by incorporating dynamic categories that will change daily based on trending topics, keeping the platform fresh and engaging for users. Furthermore, once the platform is operational, we plan to collect authentic user data (with proper consent) to refine our system and further enhance its capabilities.

In conclusion, FYP-1 has successfully laid the groundwork for Flora, and the focus for FYP-2 will be on implementing the remaining features, ensuring that the platform becomes more functional and engaging for users, while addressing the issues related to toxicity detection and content moderation.

Bibliography

[1] M. Costa-jussà, M. Meglioli, P. Andrews, D. Dale, P. Hansanti, E. Kalbassi, A. Mourachko, C. Ropers, and C. Wood, "MuTox: Universal MUltilingual audio-based TOXicity dataset and zero-shot detector," in *Findings of the Association for Computational Linguistics ACL 2024* (L.-W. Ku, A. Martins, and V. Srikumar, eds.), (Bangkok, Thailand and virtual meeting), pp. 5725–5734, Association for Computational Linguistics, Aug. 2024.

[2] M. Das, R. Raj, P. Saha, B. Mathew, M. Gupta, and A. Mukherjee, "Hatemm: A multi-modal dataset for hate video classification," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 17, pp. 1014–1023, 2023.

[3] H. Wang, T. R. Yang, U. Naseem, and R. K.-W. Lee, "Multihateclip: A multilingual benchmark dataset for hateful video detection on youtube and bilibili," *arXiv preprint arXiv:2408.03468*, 2024.

[4] V. Gupta, R. Sharon, R. Sawhney, and D. Mukherjee, "Adima: Abuse detection in multilingual audio," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6172–6176, IEEE, 2022.

[5] K. Maity, P. Sangeetha, S. Saha, and P. Bhattacharyya, "Toxvidlm: A multimodal framework for toxicity detection in code-mixed videos," in *Findings of the Association for Computational Linguistics ACL 2024*, pp. 11130–11142, 2024.

[6] M. Yousefi and D. Emmanouilidou, "Audio-based toxic language classification using self-attentive convolutional neural network," in *2021 29th European Signal Processing Conference (EUSIPCO)*, pp. 11–15, IEEE, 2021.

[7] F. T. Boishakhi, P. C. Shill, and M. G. R. Alam, "Multi-modal hate speech detection using machine learning," in *2021 IEEE International Conference on Big Data (Big Data)*, pp. 4496–4499, IEEE, 2021.

[8] M. Ibañez, R. Sapinit, L. A. Reyes, M. Hussien, J. M. Imperial, and R. Rodriguez, "Audio-based

hate speech classification from online short-form videos," in *2021 International Conference on Asian Language Processing (IALP)*, pp. 72–77, IEEE, 2021.

[9] S. Ghosh, S. Lepcha, S. Sakshi, R. R. Shah, and S. Umesh, "Detoxy: A large-scale multimodal dataset for toxicity classification in spoken utterances," in *Interspeech 2022*, pp. 5185–5189, 2022.

[10] H. Rizwan, M. H. Shakeel, and A. Karim, "Hate-speech and offensive language detection in Roman Urdu," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (B. Webber, T. Cohn, Y. He, and Y. Liu, eds.), (Online), pp. 2512–2522, Association for Computational Linguistics, Nov. 2020.

[11] A. Rana and S. Jha, "Emotion based hate speech detection using multimodal learning," *arXiv preprint arXiv:2202.06218*, 2022.

[12] M. Atanu, R. Gargi, B. Amit, D. Indranil, and N. Sudip, "Attentive fusion: A transformer-based approach to multimodal hate speech detection," in *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*, pp. 720–728, 2023.

[13] J. Liu, M. K. Nandwana, J. Pylkkönen, H. Heikinheimo, and M. McGuire, "Enhancing multilingual voice toxicity detection with speech-text alignment," *arXiv preprint arXiv:2406.10325*, 2024.

[14] P.-A. Duquenne, H. Schwenk, and B. Sagot, "Sonar: sentence-level multimodal and language-agnostic representations," *arXiv e-prints*, pp. arXiv–2308, 2023.

[15] S. Latif, A. Qayyum, M. Usman, and J. Qadir, "Cross lingual speech emotion recognition: Urdu vs. western languages," in *2018 International conference on frontiers of information technology (FIT)*, pp. 88–93, IEEE, 2018.

[16] S. Arif, A. H. Azeemi, A. A. Raza, and A. Athar, "Generalists vs. specialists: Evaluating large language models for urdu," *2024.emnlp.org/program/accepted_findings*, 2024.

[17] S. Arif, A. J. Khan, M. Abbas, A. A. Raza, and A. Athar, "Wer we stand: Benchmarking urdu asr models," *arXiv preprint arXiv:2409.11252*, 2024.

[18] M. Bilal, A. Khan, S. Jan, S. Musa, and S. Ali, "Roman urdu hate speech detection using transformer-based model for cyber security applications," *Sensors*, vol. 23, no. 8, 2023.

[19] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12449–12460, 2020.

[20] D. Alexey, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv: 2010.11929*, 2020.