ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Project Report

Version 2

CENG 408

Innovative System Design and Development II

202418

LieWave

Melih TAŞKIN
202111070
Eray YIKAR
202011074
Çağrı BAŞARAN
202011015
Onur GÜVEN
201611028
Berkay ÜĞE
202011055

Advisor: *Hayri Sever*

Table of Contents

Abstract

The **"LieWave"** project focuses on identifying deception by analyzing speech features using machine learning. It extracts key acoustic and emotional patterns from audio to classify speech as truthful or deceptive. The system integrates a user-friendly interface, a backend for data processing, and a secure database for storage. By leveraging modern tools like Librosa and TensorFlow, it ensures accuracy and scalability. This report provides an overview of the system's design and its potential applications in areas such as security and research.

Key words:

Artificial intelligence (92431), Computer Software (92408), Cloud Computing (92413)

Introduction

Deception detection has become a critical area of research, with applications in security, law enforcement, and psychological analysis. The **"LieWave"** project aims to develop an automated system for detecting deceptive behavior through speech analysis. Unlike traditional lie detection methods, this approach leverages advanced speech processing and machine learning to analyze acoustic, prosodic, and emotional features from audio.

The system is designed to provide a scalable, user-friendly solution that classifies speech as truthful or deceptive. It combines audio preprocessing, feature extraction, and machine learning models, such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM), to achieve accurate results.

This project highlights the potential of integrating technology with behavioral analysis to improve the efficiency and reliability of lie detection. The following report details the system's design, architecture, and evaluation, demonstrating its value in real-world applications.

Literature Search

1. Introduction

In security, law, and psychology; truth verification is crucial process. Conventional techniques use devices such as the lie detector setting screen people for physiological reactions. But these approaches are highly error-prone and invasive by nature, which often receive backlash. However, with the advance in technology, new speech processing techniques are an emerging alternative for lie detection. Techniques that Measure AP, RP, or EM of Speech During Deception These techniques measure acoustic (e.g., pitch, speaking rate), prosodic (e.g., rhythm, emphasis) and emotional (e.g., stress, anxiety) alterations in speech pattern during a lie.

The fusion of machine learning and deep learning algorithms in this field has shown promise in improving detection accuracy. However, challenges such as limited datasets, language and cultural differences, and difficulties in testing under realistic conditions limit their applicability. The purpose of this study is to review existing work on lie detection using language processing techniques, evaluate the strengths and weaknesses of current approaches, and provide guidance for future research.

2. Speech Processing Techniques for Lie Detection

2.1 Analysis of Acoustic Features

Acoustic features are critical for detecting changes in speech that may indicate deception. These include fundamental frequency (pitch), formant frequency, intensity, and speech rate.

- Ekman [1] found that fundamental frequency increases under stress, a common indicator of deception. This finding was critical in linking stress-induced acoustic changes to lie detection.

- J. Hirschberg et al. [2] showed that deceptive people often change their speech rate, either speeding up due to nervousness or slowing down due to cognitive load. Machine learning models trained on these features achieved significant improvements in accuracy.

Further research showed that changes in formant frequency and changes in voice intensity may further improve lie detection. These findings highlight the importance of acoustic analysis in detecting speech patterns associated with deception.

2.2 Examination of Prosodic Features

Prosodic analysis focuses on the rhythm, intonation, and stress in speech. These features can provide insights into the speaker's emotional and cognitive states, which are often disturbed during deception.

- E. Shriberg et al. [3] observed longer pauses, irregular rhythms, and unnatural intonation patterns in deceptive speech. Such deviations from normal prosody indicate the cognitive and emotional burden associated with lying.
- Prosodic features can also complement acoustic features to provide richer datasets for machine learning models. For example, combining pitch changes with pause duration can significantly improve the accuracy of deception detection systems.

2.3 Emotion Analysis

Emotion recognition in speech has been shown to be a key component of lie detection. Emotions such as stress, anxiety, and fear often accompany deceptive behavior and are therefore valuable indicators.

- V. Pérez-Rosas, R. Mihalcea, and A. Narvaez. [4] used emotion recognition algorithms to detect deception with over 80% accuracy. Their work highlighted the effectiveness of analyzing vocal expressions of stress and anxiety.
- Advanced emotion analysis systems can now detect subtle changes in pitch and intensity that are usually imperceptible to human listeners but are highly suggestive of deception.

3. Machine Learning and Deep Learning Methods

3.1 Machine Learning Models

In general, machine learning algorithms have heavily been used to classify features from speech using automatic analysis in lie detection. Commonly used are the Support Vector Machines, Decision Trees, and Naive Bayes.

- In the work done by C. Fuller, D. Biros, and R. Wilson. [5], an SVM model analyzing speech features was able to realize an accuracy rate of 85%. The critical point taken from this study was that the selection of acoustic and prosodic features for speech deception detection must be properly done for better results. Various Decision Tree-based models have also achieved promising results when finding deception patterns by combining them with feature selection techniques for model training optimization.

Despite all these successes, these traditional models usually perform poorly with more complex data. Due to this limitation, a deep learning approach has started to be adopted, which is capable of processing higher volumes of data more effectively.

3.2 Deep Learning Techniques

Deep learning models allow high-dimensional feature analyses because of their architecture; thus, models like CNN, RNN, and LSTM have started a revolution in deception detection.

- S. I. Levitan, M. An, and J. Hirschberg. [6] used an LSTM model on the speech segments as time series for 90% accuracy. Again, this outlined the strengths of deep learning for finding deception through the dynamic speech features. Different researchers, while extracting the spatial features from spectrograms, which are visual representations of speech signals, employed CNNs. These models showed a lot of promise for spotting acoustic and prosodic patterns connected to deception.

Deep learning techniques also provide transfer learning whereby pre-trained models could be fine-tuned for specific tasks. This has been one of the helpful features of the models, given that one of the major challenges to lie detection research involves the scarcity of labeled datasets.

3.3 Datasets

The development and evaluation of these lie detection systems depend on datasets. Several datasets have been created specifically for this purpose:

- **Columbia SRI-Colorado Corpus (CSC)**: It contains annotated **speech data for prosodic and acoustic analysis.** This dataset is a benchmark dataset for lie detection models.
- **Deceptive Speech Corpus**: There is also speech samples that are labeled as truthful or deceptive, so it is a foundational resource for the machine learning experiments.
- **Real-life Trial Corpus**: It consists of speech data from real court room scenarios, where valuable insights of application of lie detection techniques in the real world are available.

However, these datasets have permitted great leaps forward in the field, but lack sufficient size and diversity to promote model generalization. To further develop our results, future research should involve building larger and more varied dataset such that it covers more languages, cultures, and contexts.[7]

4. Limitations in the Literature

4.1 Data Scarcity

Unfortunately, robust lie detection models are limited by the rarity of available labeled datasets. Traditional datasets are small and not diverse which makes them not suitable for training and testing machine learning algorithms [8].

4.2 Linguistic and Cultural Variations

Language and culture play a major role in speech feature variation, which poses challenges towards developing languages and culture independent models. For instance, intonation and pitch patterns easily vary dramatically across tonal and non tonal languages and are problematic when trying to apply lie detection techniques to a wide array of populations.

4.3 Real-World Usage

In controlled environments, lie detection systems often perform well, but rarely so in real-life. In fact, background noise, overlapping speech and variability of the speakers as well can significantly affect model performance [9].

4.4 Future Research Areas

The challenges identified in the literature highlight the need for innovative approaches to advance the field:

1. **Multimodal Systems:** By combining speech or facial expressions, gestures, or physiological signals, it is possible to improve detection accuracy. These modalities require advanced data fusion techniques and real time processing capabilities to be integrated.
2. **Real-Time Analysis:** For practical applications, we need to be able to develop systems that are able to do real time lie detection.This is because we need to optimize algorithms for speed and scalability, while keeping accuracy.
3. **Transfer Learning:** It can also help alleviate the impact of scarce datasets and shorten training time by leveraging pre-trained models in similar domains, for instance linguistic modelling as in emotion recognition ; or modelling based on high dimensional data spaces like speech synthesis.

5. Conclusion

In this literature review, a systematic analysis of lie detection through speech processing techniques has been carried out. This study explores the use of speech derived acoustic, prosodic, emotional features with the use of machine learning and deep learning methods with the aim of increasing lie detection accuracy.

However, these advances are not sufficient, as challenges remain, including data paucity, cultural and linguistic variations, and the challenge in deploying such techniques in real world scenarios. To address these limitations, novel approaches must be developed: multimodal systems, transfer learning, and diverse, high quality datasets.

Finally, we conclude that speech processing techniques present a respectable route to lie detection. Nevertheless, more research is needed to realize the full potential of these directionalities. This work lays the groundwork for future research by providing a guide to shortcomings and advances in the field.

References

[1] P. Ekman, *Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage*, 4th ed., W.W. Norton, 2009.

[2] J. Hirschberg et al., "Prosody and deception detection," *Proceedings of Speech Prosody 2004*, 2004, pp. 123-132.
DOI: 10.21437/SpeechProsody.2004-12

[3] E. Shriberg et al., "Acoustic-prosodic indicators of deception in speech: Cross-corpus evaluation," *Proceedings of IEEE ICASSP 2012*, 2012, pp. 4829-4832. [4] V. Pérez-Rosas, R. Mihalcea, and A. Narvaez, "Cross-cultural deception detection," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 440-450.

[5] C. Fuller, D. Biros, and R. Wilson, "Decision support for deception detection: A machine learning approach," *Decision Support Systems*, vol. 46, no. 3, pp. 673-684, 2009.
DOI: 10.1016/j.dss.2008.11.007

[6] S. I. Levitan, M. An, and J. Hirschberg, "Acoustic-prosodic and lexical cues to deception and trust: Deciphering variation across gender, personality, and culture," *Proceedings of Interspeech 2018*, 2018, pp. 409-413.

[7] A. Vrij, *Detecting Lies and Deceit: Pitfalls and Opportunities*, 2nd ed., Wiley, 2008.
Link: https://www.wiley.com/en-us/Detecting+Lies+and+Deceit%3A+Pitfalls+and+Opportunities%2C+2nd+Edition-p-9780470516249

[8] T. Baltrušaitis, P. Robinson, and L.-P. Morency, "OpenFace: An open source facial behavior analysis toolkit," *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1-10.
DOI: 10.1109/WACV.2016.7477553

[9] A. Kumar, S. G. Koolagudi, and K. S. Rao, "Speech emotion recognition using deep

learning," *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 116-121.

Software Requirements Specification

1. Introduction

1.1 Purpose of this Document

This Software Requirements Specification (SRS) document is designed to serve as a comprehensive guide for the development of a lie detection system using speech processing techniques. The system employs advanced machine learning and deep learning methodologies to analyze speech features such as acoustic, prosodic, and emotional patterns for detecting deception. The purpose of this document is to outline the functional and technical requirements of the project, providing a detailed framework to guide the development team in creating an efficient and accurate lie detection system. This document will ensure all stakeholders have a clear understanding of the project objectives, constraints, and deliverables while navigating the complexities of speech analysis and machine learning integration.

1.2 Scope of The Project

The primary focus of this project is to develop a system capable of detecting deception by analyzing speech patterns. The system utilizes machine learning algorithms (e.g., Support Vector Machines, Decision Trees) and deep learning techniques (e.g., CNN, LSTM) to extract and process features from speech, such as pitch, intonation, and emotional cues.

The project involves creating a robust pipeline for speech preprocessing, feature extraction, and model training, allowing the system to analyze speech data accurately and efficiently. The application will target various domains, including security, law enforcement, and psychological research, providing a non-invasive alternative to traditional lie detection methods. The software will support real-time analysis, leveraging pre-trained models for high accuracy and scalability.

Developed in Python, the project will integrate key libraries and tools such as Librosa for audio analysis and TensorFlow/PyTorch for machine learning and deep learning tasks. The deliverables include a web-based user interface for input and output interaction, datasets for training and testing, and model evaluation reports to ensure transparency and reliability.

---

2. General Description

The project entails developing a speech-based lie detection system that leverages cutting-edge speech processing techniques and AI models. The system will analyze acoustic (e.g., pitch, speech rate), prosodic (e.g., rhythm, intonation), and emotional (e.g., stress, anxiety) features in speech to detect deceptive behavior.

Machine learning models, including both traditional (e.g., SVM, Decision Trees) and advanced deep learning architectures (e.g., CNN, LSTM), will be employed to achieve high detection accuracy. Speech data will be preprocessed to extract key features, which will then be fed into models for training and evaluation.

A web-based user interface will allow users to upload audio recordings and view the results of lie detection analysis in real-time. The system will also include visualization tools to display key patterns and insights derived from speech data. By incorporating datasets such as the Columbia SRI-Colorado Corpus and the Deceptive Speech Corpus, the project aims to ensure robust model training and testing.

Developed on Python, the system will be deployable across multiple platforms and environments, ensuring scalability and versatility for real-world applications.

2.1 Glossary

Term	Definition
AI (Artificial Intelligence)	A branch of computer science focused on creating systems capable of performing tasks requiring human intelligence.
ML (Machine Learning)	A subset of AI where algorithms learn patterns from data to make predictions or decisions.
DL (Deep Learning)	A subset of ML that uses neural networks with many layers to analyze complex patterns in data.
Speech Processing	Techniques used to analyze and process speech signals for various applications, such as recognition or analysis.
Acoustic Features	Characteristics of sound, such as pitch, intensity, and formant frequency, used in speech analysis.
Prosodic Features	Speech attributes related to rhythm, intonation, and stress, providing insights into cognitive and emotional states.
Emotion Recognition	The process of identifying emotions in speech or other forms of data through analysis and modeling.
LSTM (Long Short-Term Memory)	A type of recurrent neural network (RNN) that is effective in processing sequential data, such as speech signals.
SVM (Support Vector Machine)	A machine learning algorithm used for classification and regression tasks by finding an optimal decision boundary.
Librosa	A Python library for analyzing and extracting features from audio signals
TensorFlow/PyTorch	Popular frameworks for implementing and training machine learning and deep learning models.

Dataset	A collection of data samples used for training and testing machine learning models.
User	An individual interacting with the lie detection system, typically providing audio input for analysis.

2.2 User Characteristics

- Technical Proficiency: Users are not required to have in-depth knowledge of machine learning or speech processing. However, familiarity with basic computing concepts and the ability to upload audio files to the system is necessary.
- Professional Background: The primary users of this system may include professionals in law enforcement, psychological research, and security. These users may not have technical expertise but are expected to interpret the system's results based on provided insights.
- Training Requirements: Minimal training is required for users to interact with the web-based interface. The system will include a help guide and tooltips for user assistance.
- Problem-Solving Skills: Users should be capable of applying the system's outputs to their specific contexts, such as investigations, behavioral studies, or risk assessments.
- Adaptability: Users must be open to adopting this new technology as a supplementary tool for detecting deception, which may involve integrating it with existing workflows.

2.3 Overview of Functional Requirements

- Speech Analysis: The system will analyze audio input files to extract relevant acoustic, prosodic, and emotional features.
- Lie Detection: Employ machine learning and deep learning models to classify speech as deceptive or truthful based on the extracted features.
- Real-Time Processing: Enable real-time or near-real-time analysis for scenarios requiring immediate results, using pre-trained models.
- User Interface: Provide a user-friendly web interface for audio input, displaying results, and accessing visualization tools that explain the analysis.
- Visualization Tools: Include graphical representations of speech patterns, highlighting features contributing to deception detection (e.g., pitch variations, stress markers).
- Model Customization: Allow for updates or retraining of models with new datasets to improve system accuracy and adapt to evolving requirements.
- Data Security: Ensure that user data, including uploaded audio files, is handled securely and complies with relevant privacy standards.

2.4 General Constraints and Assumptions

Constraints:

1. **Hardware Requirements**: The system's performance may depend on the server's computational capabilities for running deep learning models and processing speech data.

2. **Dataset Availability:** The quality and diversity of training datasets directly influence the system's accuracy and reliability.
3. **Real-Time Limitations:** Real-time processing may be constrained by the size and complexity of the audio files and the computational resources available.
4. **Internet Dependency:** The system requires a stable internet connection for users to access the web interface and perform analyses.

Assumptions:

1. **User Proficiency:** It is assumed that users will have basic computer skills and access to devices capable of recording or uploading audio files.
2. **Speech Quality:** The audio input provided will be of sufficient quality (e.g., clear speech with minimal background noise) to enable effective analysis.
3. **Ethical Use:** It is assumed that the system will be used ethically and responsibly within legal and organizational guidelines.
4. **Model Accuracy:** The system's accuracy is dependent on the training data and model optimization, assuming that the provided datasets are comprehensive and unbiased.

---

3. Specific Requirements

3.1 Interface Requirements

3.1.1 User Interface

- **Real-Time Data Visualization:**
  Implement a user-friendly interface to visualize real-time analysis results, including acoustic, prosodic, and emotional features detected in the speech input.
- **Interactive Dashboard:**
  Provide a dashboard displaying analysis metrics such as deception probability, emotional states, and acoustic feature changes, allowing users to monitor the system's performance and insights.
- **Audio Upload Interface:**
  Design a simple audio file upload interface for users to input their recordings for analysis.

3.1.2 Hardware Interface

- **Computing Power:**
  Ensure adequate computing resources, including high-performance CPUs and GPUs, to process complex speech features and run machine learning models efficiently.
- **Audio Input Devices:**
  Support for standard audio recording devices such as microphones for real-time audio capture.

- **Peripheral Support:**
  Compatibility with basic peripherals like monitors and input devices (keyboard, mouse) for interacting with the user interface.

3.1.3 Software Interface

- **Python Environment:**
  Compatibility with Python and necessary libraries (e.g., Librosa, TensorFlow, PyTorch) for audio processing and model execution.
- **Model Integration:**
  Seamless integration of machine learning models (SVM, CNN, LSTM) for feature extraction and lie detection analysis.
- **Database Support:**
  Integration with a database system (e.g., PostgreSQL) for storing analysis results and logs.
- **Operating System Support:**
  Ensure system compatibility with major operating systems like Ubuntu and Windows.

3.1.4 Communication Interfaces

- **Data Exchange Protocols:**
  Implement secure and efficient data exchange protocols for audio files and model outputs.
- **API Integration:**
  Develop APIs for connecting the system with external services or other software components for data sharing and model updates.
- **Real-Time Communication Support:**
  Ensure communication channels can handle real-time audio processing and results generation with minimal latency.

3.2 Detailed Description of Functional Requirements

3.2.1 Template for Describing Functional Requirements

For each functional requirement of the lie detection system, the following template will be used:

- **Requirement Name:** A clear and concise name for the requirement.
- **Description:** A detailed description of the requirement, explaining its purpose within the system.
- **Priority:** The level of importance of the requirement (e.g., High, Medium, Low).
- **Inputs:** The inputs required for the requirement to function (e.g., audio files, user commands).
- **Processing Steps:** A step-by-step explanation of how the requirement will be processed within the system.

- **Outputs:** The expected outputs or results from the requirement (e.g., deception probability, emotional state analysis).
- **Dependencies:** Any other system components or requirements that this requirement relies on to function properly.

3.3 Non-Functional Requirements

1. Performance

- The system must process speech data and generate analysis results within 1000 milliseconds of receiving input, ensuring real-time feedback with latency not exceeding 2000 milliseconds under normal operating conditions.

2. Reliability

- The system must maintain a 99% uptime and demonstrate consistent performance when handling varied speech inputs and concurrent user requests, supporting up to 100 simultaneous users without degradation.

3. Usability

- The user interface must have a System Usability Scale (SUS) score of 80 or higher, enabling non-technical users to perform key tasks (e.g., uploading speech files, viewing results) within 3 clicks or less.

4. Scalability

- The system must scale to handle a 50% increase in data load (e.g., from 100 GB to 150 GB) and support up to 100 concurrent users with response times remaining under 1500 milliseconds.

5. Maintainability

- The system architecture must be modular, allowing for independent updates to components without affecting the rest of the system. All code must include unit tests with at least 90% coverage and detailed documentation of key modules.

6. Security

- Implement data encryption using AES-256 for storage and TLS 1.3 for transmission. The system must enforce multi-factor authentication (MFA) and log all access attempts, retaining logs for a minimum of 90 days.

7. Compatibility

- The system must be compatible with at least 90% of commonly used hardware and software configurations, including Windows, macOS, Linux, and a variety of microphones (e.g., USB and analog).

8. Portability

- Trained models must be exportable in standard formats (e.g., ONNX or TensorFlow SavedModel) and deployable on different platforms (e.g., cloud environments, edge devices) with no more than 10% performance degradation.

9. Compliance

- The system must comply with GDPR and CCPA regulations for data privacy, and adhere to ethical AI guidelines as defined by ISO/IEC 22989 (AI ethical frameworks).

10. Accessibility

- Ensure conformance to WCAG 2.1 Level AA guidelines, including features like real-time captioning for speech analysis results and keyboard navigation for all system functionalities.

---

4. Analysis-UML

4.1 Use Cases

4.1.1 Use Case Diagram

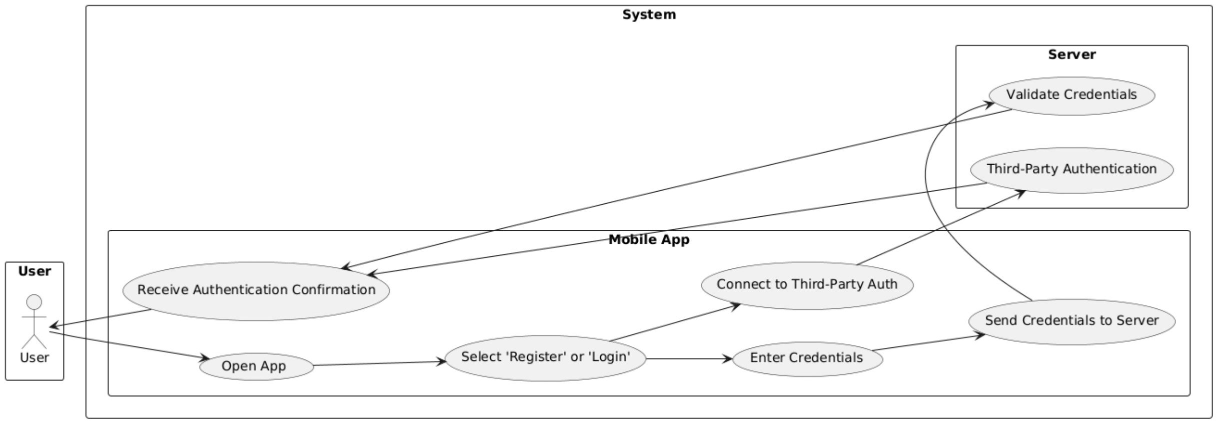

System
Server
Validate Credentials
Third-Party Authentication
Mobile App
User
User
Receive Authentication Confirmation
Connect to Third-Party Auth
Send Credentials to Server
Open App
Select 'Register' or 'Login'
Enter Credentials

Diagram 1 - Use Case 1 Diagram

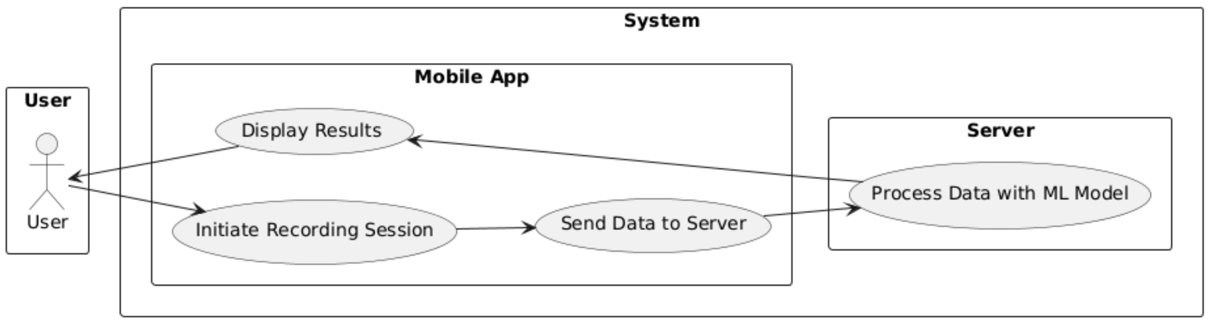


Diagram 2 - Use Case 2 Diagram

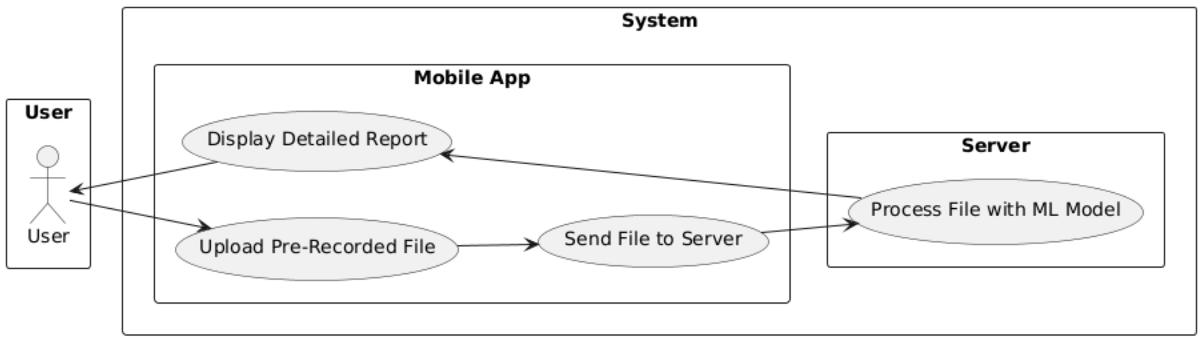


Diagram 3 - Use Case 3 Diagram

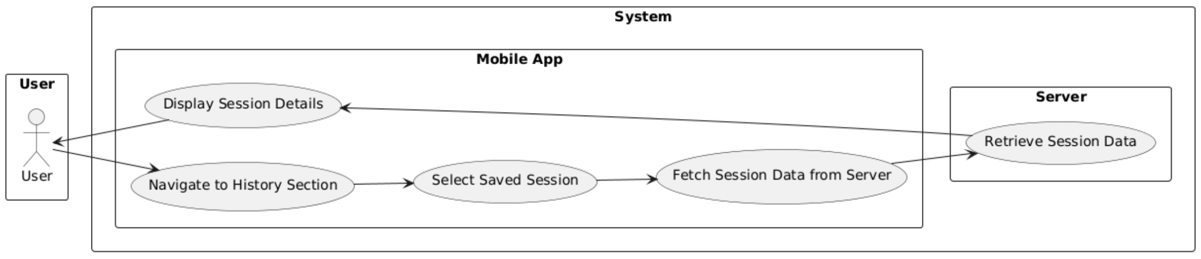


Diagram 4 - Use Case 4 Diagram

4.1.2 Describe Use Cases

Field	**Description**
Use Case Number	Use Case 1
Use Case Name	User Registration and Authentication
Actor	User
Description	Users create an account or log in to access the app.
Precondition	The user must have the app installed and an active internet connection.
Scenario	1. User opens the app.
	2. Selects "Register" or "Login."
	3. Enters credentials (email/phone, password) or uses third-party authentication (e.g., Google, Apple ID).

	4. The app sends credentials to the server for validation or connects to a third-party authentication service.
	5. The server or third-party provider verifies the credentials.
	6. The app receives a confirmation of successful authentication.
Postcondition	The user is authenticated and granted access to app features.
Exceptions	Invalid credentials entered, third-party authentication unavailable, or server issues prevent successful authentication.

Field	**Description**
Use Case Number	Use Case 2
Use Case Name	Real-Time Lie Detection
Actor	User
Description	The user records or streams audio/video, and the app sends the data to the server for lie detection.
Precondition	The user must grant microphone/camera permissions. The device must be connected to the server.
Scenario	1. The user initiates a recording session.
	2. The app sends the recorded data to the server.
	3. The server processes the data using the machine learning model.
	4. The app displays the result: "Truth," "Lie," or "Uncertain."
Postcondition	Results are displayed to the user.
Exceptions	Microphone or camera permissions not granted, Server connectivity issues, Errors during the server-side ML model processing.

Field	**Description**
Use Case Number	Use Case 3
Use Case Name	Analysis of Recorded Conversations
Actor	User
Description	The user uploads a pre-recorded audio or video file for analysis.
Precondition	The file format is supported. The device is connected to the server.
Scenario	1. The user uploads a file.
	2. The app sends the uploaded data to the server.

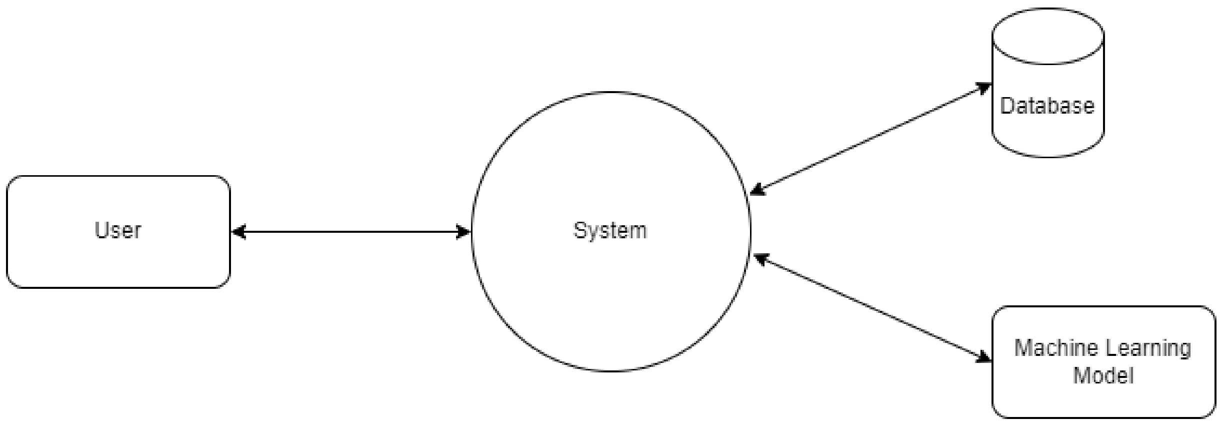
	3. The server processes the data using the machine learning model.
	4. The app provides a detailed report on potential lies and truthfulness.
Postcondition	Results are displayed to the user.
Exceptions	Unsupported file format, Server connectivity issues, Errors during the server-side ML model processing.

Field	Description
Use Case Number	Use Case 4
Use Case Name	View History and Reports
Actor	User
Description	Users can view past lie detection reports and session summaries.
Precondition	The user must be authenticated, The user must have prior saved sessions available in the app.
Scenario	1. The user navigates to the "History" section.
	2. The user selects a session from the list of saved sessions.
	3. The app displays detailed session information.
Postcondition	The user gains insights and can review the analysis of previous sessions.
Exceptions	No saved sessions available for the user, Errors retrieving session data from the server.

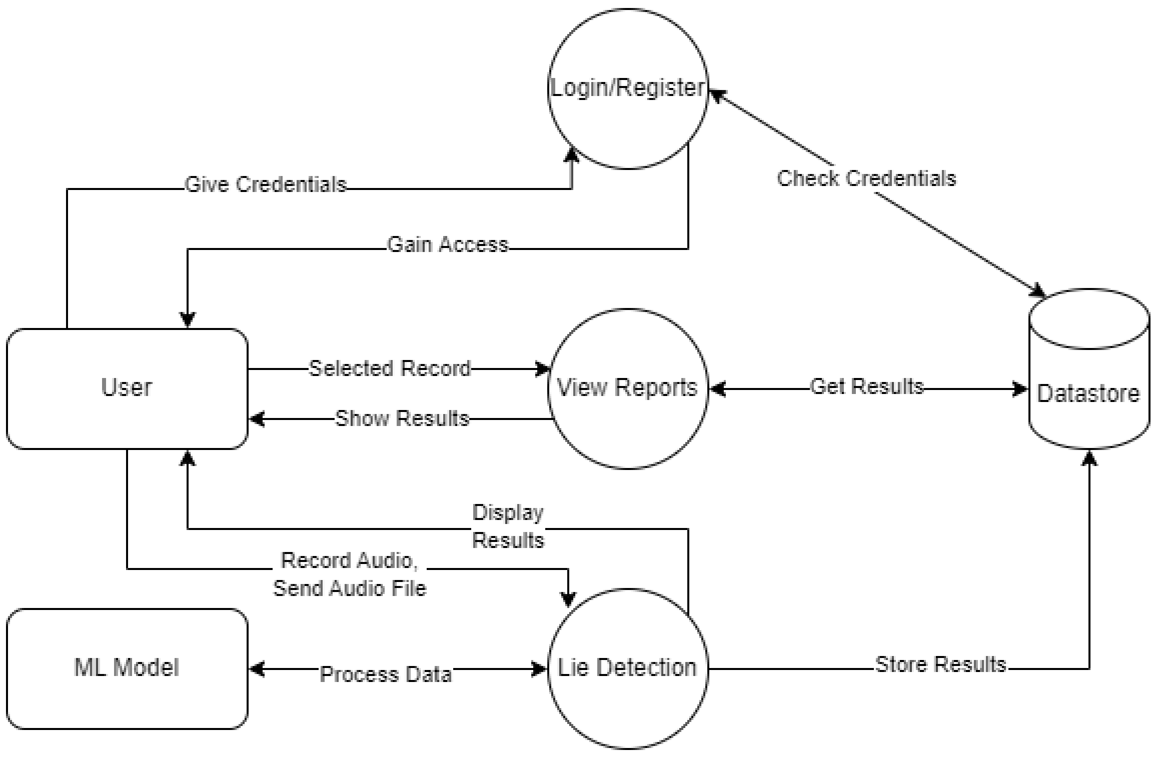
4.2 Functional Modeling (DFD)

4.2.1 DFD Diagrams

Level-0 Diagram



Level-1 Diagram



5. Conclusion

The "Lie Detection Using Speech Processing Techniques" project aims to provide a fast, accurate, and non-invasive method for detecting deception through speech analysis. By utilizing advanced AI and speech processing technologies, the system offers practical applications in fields like law enforcement, security, and psychological research.

This project focuses on creating a reliable and user-friendly tool that can analyze speech patterns in real time, providing clear and actionable results. With its ability to adapt to new data and evolving needs, the system has the potential to become a valuable resource for professionals seeking a deeper understanding of truth and trust in various contexts.

References and Resources

- **Librosa Documentation**
  https://librosa.org/
  A Python library for audio and speech feature extraction.
- **TensorFlow Documentation**
  https://www.tensorflow.org/
  A popular deep learning framework used for implementing and training machine learning models.
- **PyTorch Documentation**
  https://pytorch.org/
  A flexible and efficient deep learning framework for building neural networks.
- **Columbia SRI-Colorado Corpus**
  A dataset of speech samples widely used for research in speech processing and emotion recognition.
- **Deceptive Speech Corpus**
  A specialized dataset for studying speech-based deception detection.

Software Design Document

1. Introduction

This document provides a detailed description of the software design for the "Lie Detection Using Speech Processing Techniques" project. The system focuses on analyzing acoustic, prosodic, and emotional features to detect deception. This document elaborates on the project's purpose, scope, and components while emphasizing technical details and architectural decisions.

The project offers a roadmap to guide the development team and stakeholders in managing the complexities of integrating speech analysis and machine learning. It combines advanced technology and software architecture to provide an effective solution for detecting deceptive behavior.

1.1 Purpose

The purpose of this document is to detail the design and architectural choices of the "Lie Detection Using Speech Processing Techniques" project and provide guidance throughout the development process. The system focuses on the integration of components, the use of technical tools, and the establishment of a scalable infrastructure. This document addresses the following key objectives:

- Establishing a comprehensive framework for system design.
- Providing a clear roadmap for the engineering team.
- Aligning stakeholders’ understanding of the project.

1.2 Definitions

This section defines key technical terms and concepts frequently used in the document:

Term	Definition
Lie Detection	The process of identifying deception by analyzing speech features.
Speech Processing	Techniques used to analyze and extract features from speech signals.
CNN (Convolutional Neural Network)	A deep learning model used to analyze structured data such as speech.
LSTM (Long Short-Term Memory)	A type of neural network effective for sequential data like audio.
Librosa	A Python library for audio analysis and feature extraction.
TensorFlow/PyTorch	Frameworks used for building and training machine learning models.
Deceptive Speech Corpus	A dataset designed for research in speech-based deception detection.

---

2. System Overview

This project is designed to detect deceptive behavior by analyzing speech features. The system combines advanced speech processing techniques and machine learning methodologies to provide an effective solution.

2.1 System Components

The system consists of the following key components:

- **Speech Preprocessing Module:**
  - Uses Librosa for audio cleaning, segmentation, and extraction of acoustic features.
- **Machine Learning Module:**
  - Analyzes extracted features using traditional ML models (SVM) and deep learning architectures (CNN, LSTM).
- **Mobile-Based Interface:**
  - Provides a user-friendly platform for uploading audio, analyzing results, and offering graphical visualizations.
- **Visualization Tools:**
  - Offers graphical representations of speech patterns (e.g., pitch fluctuations, stress markers) contributing to the analysis.

2.2 System Interaction
The system components interact as follows:

1. **Audio Input:** Users upload audio files or provide real-time recordings.
2. **Preprocessing:** Audio files are cleaned, and key features are extracted after noise removal.
3. **Model Analysis:** Extracted features are analyzed using models to classify results as "truthful" or "deceptive."
4. **Result Display:** Analysis results are presented with visualized reports.

2.3 Key Technologies and Tools
- **Programming Language:** Python, Dart
- **Libraries:** Librosa, TensorFlow, PyTorch
- **Databases:** PostgreSQL for storing analysis results.
- **Frameworks:** Flutter (Mobile), Flask (backend), TensorFlow Serving (for model deployment).
- **Deployment:** Docker for platform-independent and portable infrastructure.

2.4 System Objectives
- Provide accurate and fast solutions for lie detection.
- Prioritize data privacy and user confidentiality.
- Ensure a scalable infrastructure for multiple users and increased data loads.

---

3. System Design
This section provides a comprehensive description of the system architecture, its components, and how they interact to fulfill the functional requirements outlined in the SRS.

3.1 Architectural Design

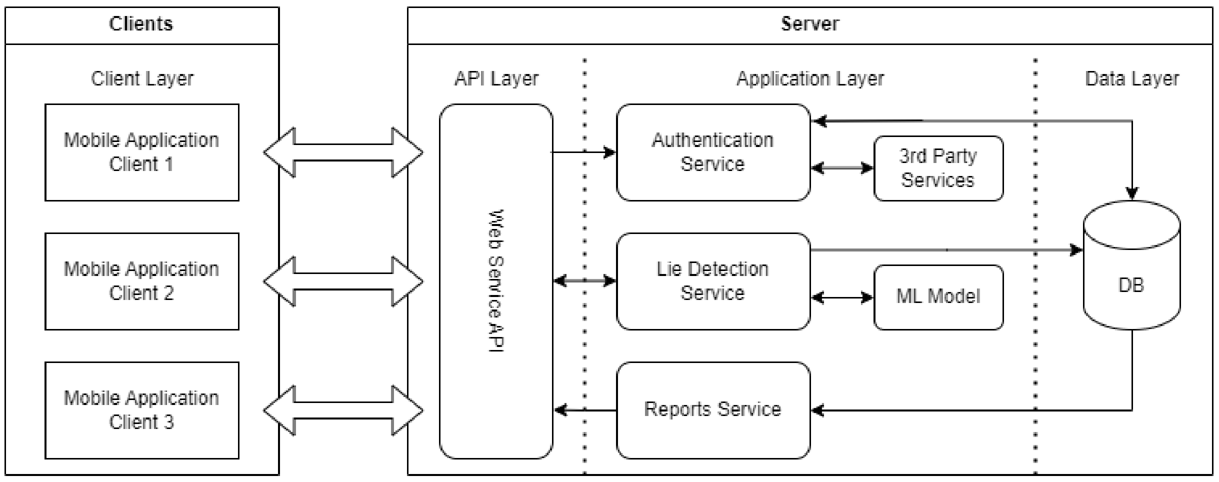
3.1.1 System Architecture Overview
The system is composed of three main components:

1. **Mobile App (Frontend)**:
   - Provides an interface for users to interact with the system.
   - Allows users to upload audio files, record real-time audio, and view analysis results.
2. **Server (Backend)**:

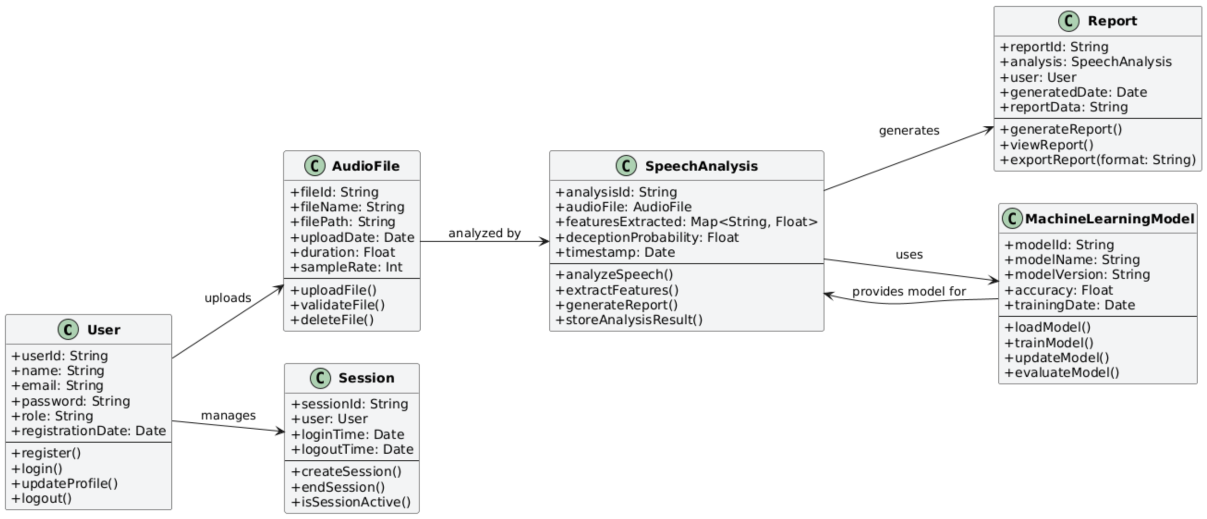
- Acts as the intermediary between the mobile app and the machine learning model.
- Handles user authentication, audio file processing, and data storage.
- Invokes the machine learning model for analysis and returns results to the frontend.

3. **Machine Learning Model**:
    - Hosted on the server, trained to analyze speech patterns and classify them as truthful or deceptive.
    - Processes audio features such as pitch, prosody, and emotional markers to generate results.
4. **Database**:
    - Stores user data, audio files, and analysis results securely.

3.1.2 Architecture Diagram

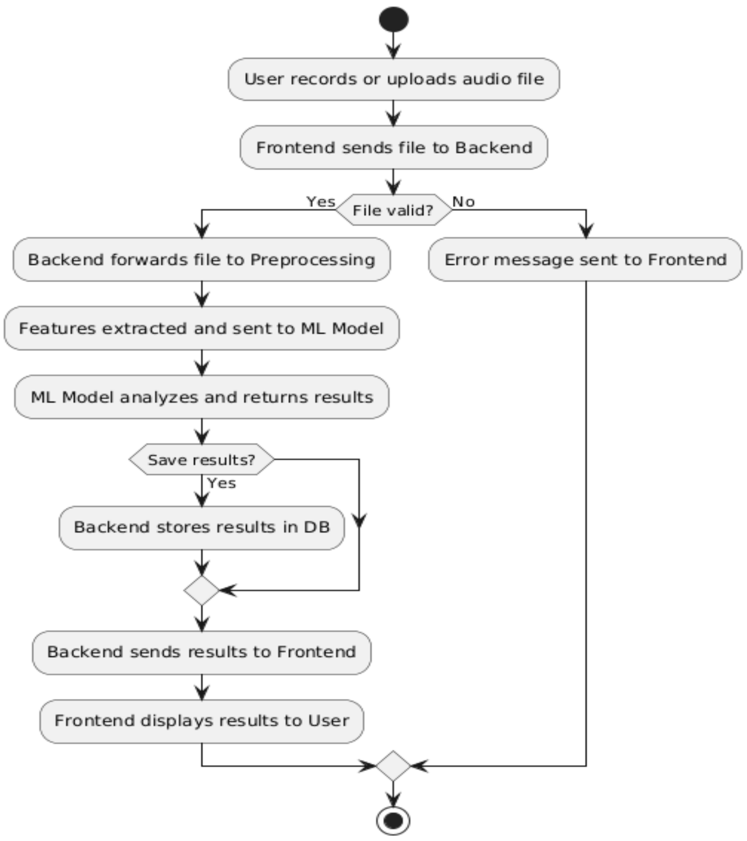


3.2 Class Diagram



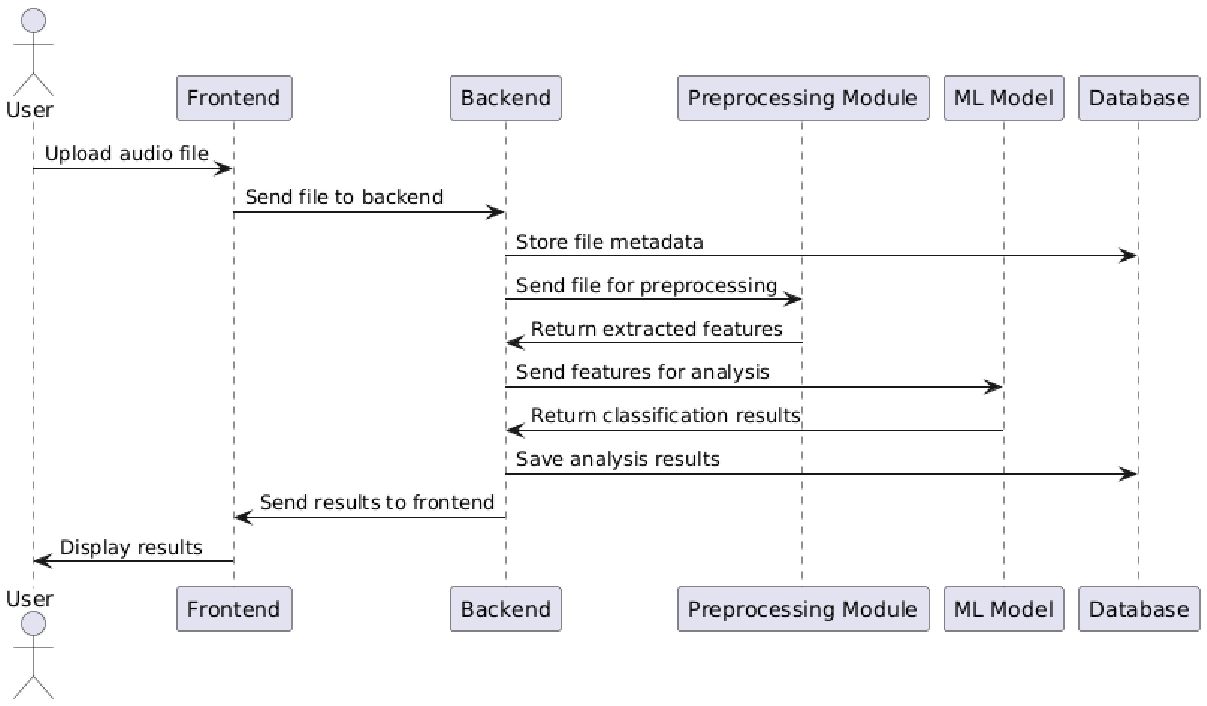
3.3 System Modelling

3.3.1 Lie Detection



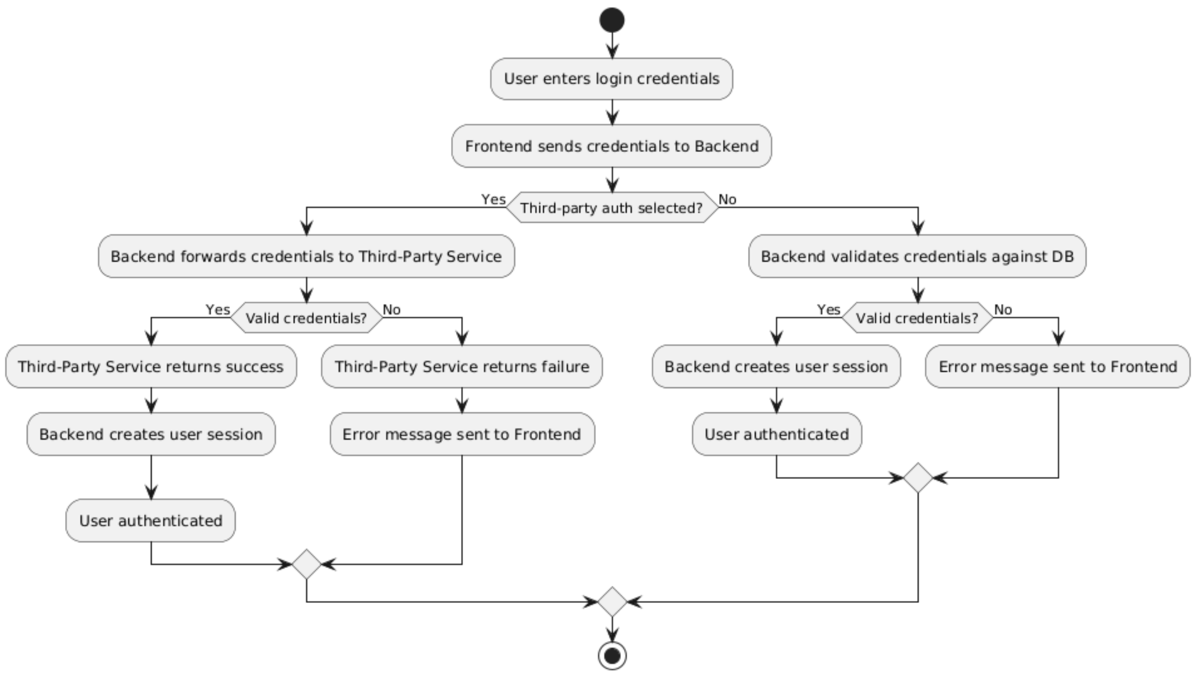
User records or uploads audio file
Frontend sends file to Backend
Yes
File valid?
No
Backend forwards file to Preprocessing
Error message sent to Frontend
Features extracted and sent to ML Model
ML Model analyzes and returns results
Save results?
Yes
Backend stores results in DB
Backend sends results to Frontend
Frontend displays results to User

Activity Diagram 1



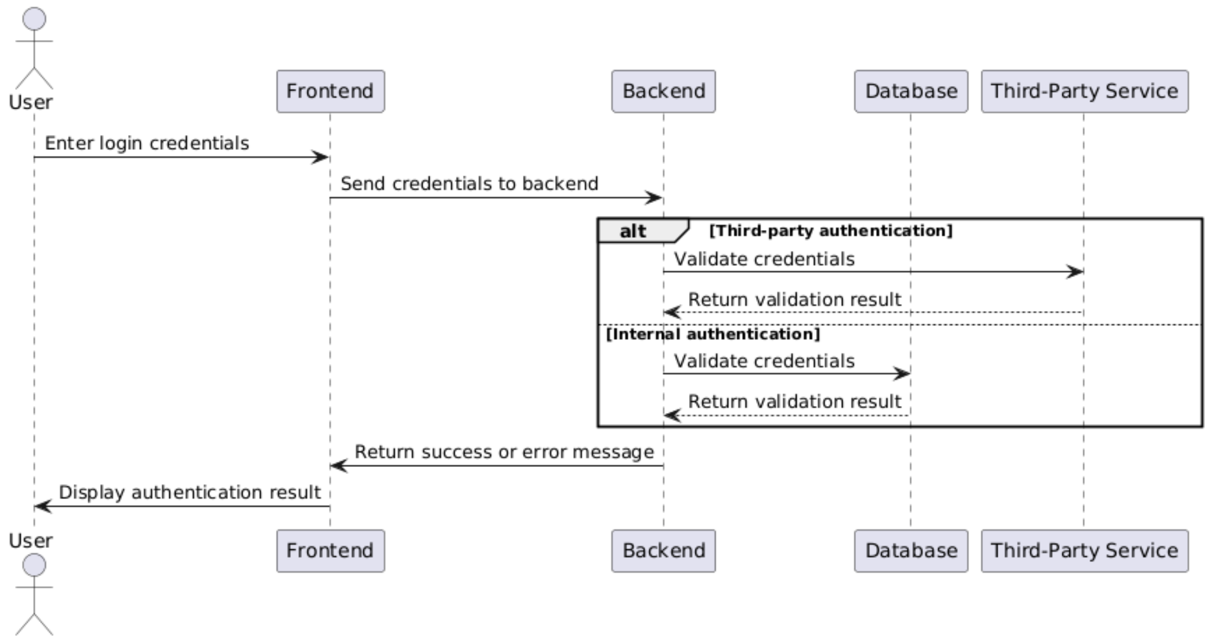
Sequence Diagram 1

3.3.2 Authentication



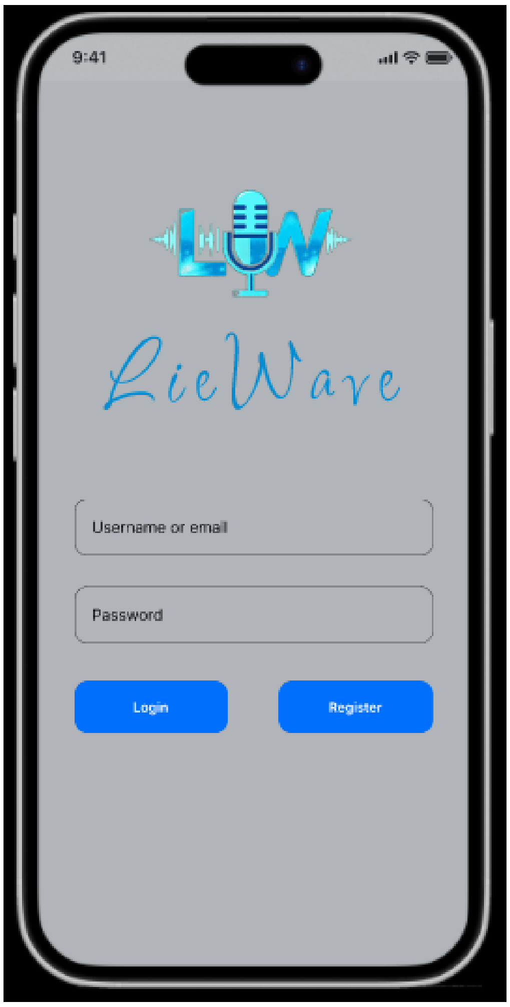
User enters login credentials
Frontend sends credentials to Backend
Third-party auth selected?
Yes
No
Backend forwards credentials to Third-Party Service
Valid credentials?
Yes
No
Third-Party Service returns success
Backend creates user session
User authenticated
Third-Party Service returns failure
Error message sent to Frontend
Backend validates credentials against DB
Valid credentials?
Yes
No
Backend creates user session
User authenticated
Error message sent to Frontend

Activity Diagram 2

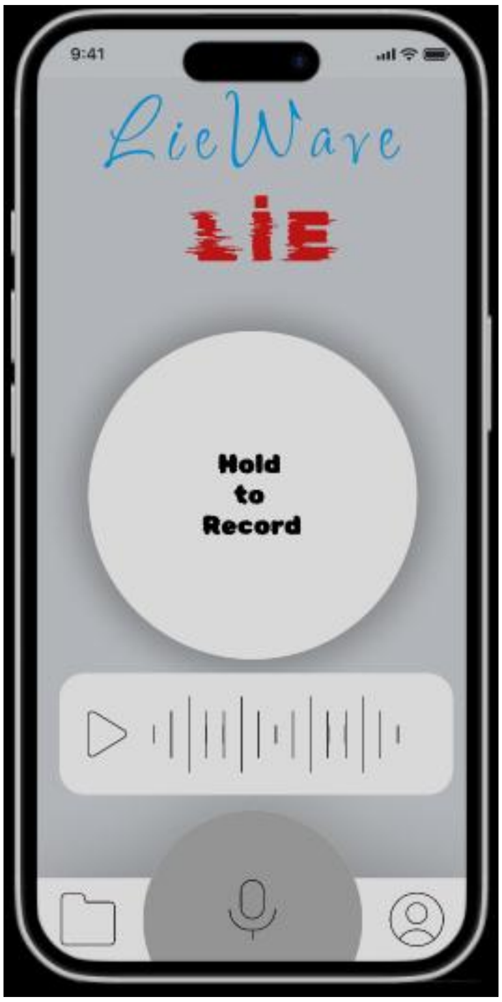


Sequence Diagram 2

4. Interface Design



Login Screen



Lie Detection Screen

Test Plan Document

1. Introduction

1.1 Version Control

Version No	Description of Changes	Date
1.0	First Version	May 21, 2025

1.2 Overview

This document describes the test plan for **LieWave**, a deception detection app that uses Google Firebase Authentication, speech recording/upload, Flask API-based lie prediction, and playback of previous analyses. It covers functional testing of user login via Google, recording and uploading audio, sending audio paths to the Flask API, and correctly displaying predictions.

1.3 Scope

Includes:

- Testing Google-based user authentication.
- Testing audio recording, upload, and format validation.
- Testing communication with the lie detection API.
- Testing UI feedback and result display with color-coded predictions.
- Testing playback of previous recordings with associated predictions.

1.4 Terminology

Acronym	Definition
LG	Google Login
UR	Upload Recording
RT	Real-Time Lie Detection
HP	Historical Playback
API	Application Programming Interface

---

2. Features To Be Tested

2.1 Google Login (LG)

- Authenticate users using Firebase Google Sign-In.
- Validate login success and failure flows.

2.2 Upload Recording (UR)

- Record audio (AAC format).
- Upload audio files to Firebase Storage.
- Validate upload success and error handling.

2.3 Real-Time Detection (RT)

- Record audio, send file path to Flask API.
- Receive prediction response with confidence.
- Display prediction with color codes (“truth” in green, “lie” in red).
- Handle API errors and edge cases gracefully.

2.4 Historical Playback (HP)

- List previously analyzed recordings with predictions.
- Playback audio files.
- Show prediction and confidence per playback item.

---

3. Features Not Be Tested

- Firebase Authentication internals (Google auth handled by Firebase).
- Model training and machine learning algorithm internals.
- Network layer reliability outside app/API scope.

---

4. Pass/Fail Criteria

Pass Criteria

- Correct behavior with no crashes.
- UI responsiveness and proper state updates.
- Correct display of prediction results and confidence.
- Successful uploads and API calls.

Fail Criteria

- App crashes, freezes, or UI blocks.
- Incorrect or missing prediction displays.
- Upload failures without error messaging.

Exit Criteria

- 100% of high and medium priority tests executed.
- At least 95% tests passed.
- Critical bugs resolved.

---

5. References

- LieWave Project Report
- Software Requirements Specification (SRS)
- Software Design Document (SDD)

---

6. Test Design Specifications

6.1 Google Login (LG)

TC ID	Requirement	Priority	Scenario Description
LG.GG.01	1.1	High	User logs in successfully via Google Sign-In
LG.GG.02	1.1	Medium	User cancels Google login flow
LG.GG.03	1.1	Medium	Login fails due to no internet connection

---

6.2 Upload Recording (UR)

TC ID	Requirement	Priority	Scenario Description
UR.REC.01	2.1	High	Record audio for minimum 10 seconds
UR.REC.02	2.1	Medium	Attempt to record audio less than 10 seconds (should warn and reject)
UR.UP.01	2.2	High	Upload valid audio file (AAC) to Firebase
UR.UP.02	2.2	Medium	Upload fails due to network error (show error)

6.3 Real-Time Lie Detection (RT)

TC ID	Requirement	Priority	Scenario Description
RT.API.01	3.1	High	Send uploaded audio file path to Flask API
RT.API.02	3.1	High	Receive prediction response with confidence
RT.DP.01	3.2	High	Display prediction with “truth” as green and “lie” as red
RT.DP.02	3.2	Medium	Show appropriate error message on API failure

---

6.4 Historical Playback (HP)

TC ID	Requirement	Priority	Scenario Description
HP.LS.01	4.1	Medium	List past predictions associated with user
HP.PB.01	4.1	Medium	Play audio recording from history list
HP.DC.01	4.1	Medium	Display prediction and confidence for selected item

---

7. Detailed Test Cases

7.1 LG.GG.01 – Google Login Success

- **Purpose:** Verify user can authenticate with Google successfully
- **Requirement:** 1.1
- **Priority:** High
- **Preconditions:** User has valid Google account
- **Test Steps:**
    1. Launch app.
    2. Tap “Sign in with Google”.
    3. Complete Google authentication flow.
- **Expected Result:** User is authenticated and navigated to Home screen.
- **Cleanup:** Log out.

7.2 UR.REC.01 – Record Audio (≥ 10 sec)

- **Purpose:** Ensure audio recorder captures at least 10 seconds of audio
- **Requirement:** 2.1
- **Priority:** High
- **Preconditions:** User is logged in and on HomePage
- **Test Steps:**
    1. Tap “Start Recording”.
    2. Record audio for at least 10 seconds.
    3. Stop recording.
- **Expected Result:** Audio file saved locally, file path stored for upload.
- **Cleanup:** None.

7.3 UR.UP.01 – Upload Audio File

- **Purpose:** Validate audio file upload to Firebase Storage
- **Requirement:** 2.2
- **Priority:** High
- **Preconditions:** Audio file recorded and available
- **Test Steps:**
    1. Trigger upload function.
    2. Confirm upload completes without error.
- **Expected Result:** File is uploaded successfully and URL is retrieved.

- **Cleanup:** None.

7.4 RT.DP.01 – Display Prediction Result with Colors

- **Purpose:** Verify UI shows prediction and confidence, color-coded by result
- **Requirement:** 3.2
- **Priority:** High
- **Preconditions:** API returns valid prediction (truth or lie)
- **Test Steps:**
    1. Upload audio and receive API response.
    2. Observe prediction displayed on screen.
- **Expected Result:**
    1. Prediction “truth” text is green
    2. Prediction “lie” text is red
    3. Confidence percentage shown below prediction
- **Cleanup:** Clear prediction data.

7.5 HP.PB.01 – Playback Historical Audio

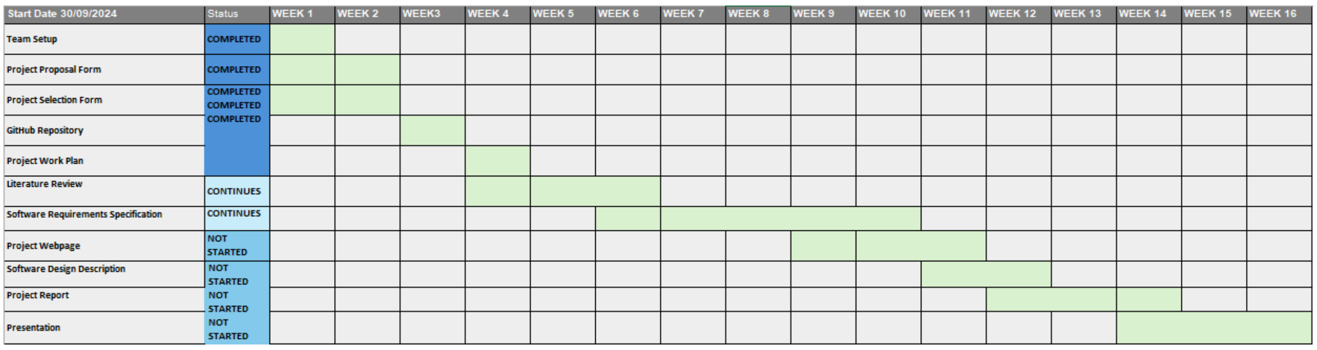
- **Purpose:** Verify playback of previously analyzed recordings
- **Requirement:** 4.1
- **Priority:** Medium
- **Preconditions:** User has historical recordings available
- **Test Steps:**
    1. Open history list screen.
    2. Select a recording.
    3. Tap play.
- **Expected Result:** Audio plays smoothly, UI reflects playback status.
- **Cleanup:** Stop playback.

---

8. Test Results

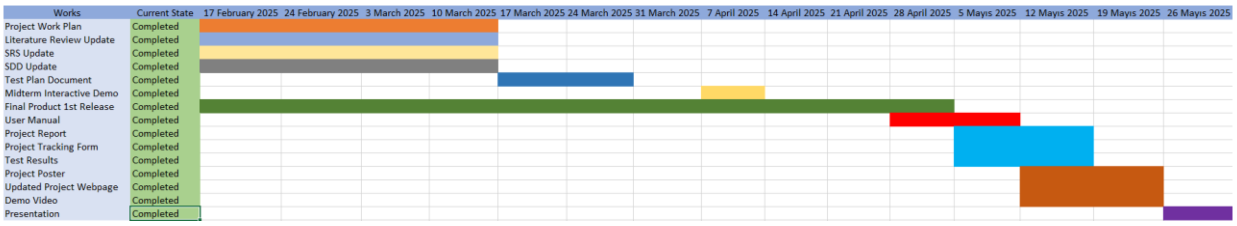
TC ID	Test Description	Date	Result	Remarks
LG.GG.01	Google Login Success	2025-05-20	Pass	Login successful, user navigated home
LG.GG.02	Google Login Cancel	2025-05-20	Pass	Cancel login returned to login screen
LG.GG.03	Login Failure (No Internet)	2025-05-21	Pass	Proper error shown for no connectivity
UR.REC.01	Record Audio (≥ 10 sec)	2025-05-21	Pass	Audio recorded and saved successfully
UR.REC.02	Record Audio (< 10 sec) Reject	2025-05-21	Pass	Warning shown, recording stopped
UR.UP.01	Upload Audio File	2025-05-22	Pass	File uploaded to Firebase without error
UR.UP.02	Upload Fail Network Error	2025-05-22	Pass	Upload error message displayed correctly
RT.API.01	API Request Success	2025-05-22	Pass	API response received as expected
RT.API.02	API Failure Handling	2025-05-22	Pass	Failure message displayed
RT.DP.01	Prediction Result Display with Colors	2025-05-22	Pass	“Truth” shown in green, “Lie” in red
HP.LS.01	List Historical Predictions	2025-05-23	Pass	History displayed correctly
HP.PB.01	Playback Historical Audio	2025-05-23	Pass	Playback successful
HP.DC.01	Display Prediction/Confidence	2025-05-23	Pass	Data shown with appropriate UI styles

Work Plan 407



Start Date 30/09/2024	Status	WEEK 1	WEEK 2	WEEK3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10	WEEK 11	WEEK 12	WEEK 13	WEEK 14	WEEK 15	WEEK 16
Team Setup	COMPLETED	🟩															
Project Proposal Form	COMPLETED	🟩	🟩														
Project Selection Form	COMPLETED COMPLETED	🟩	🟩														
GitHub Repository	COMPLETED			🟩													
Project Work Plan					🟩												
Literature Review	CONTINUES				🟩	🟩	🟩										
Software Requirements Specification	CONTINUES						🟩	🟩	🟩	🟩	🟩						
Project Webpage	NOT STARTED									🟩	🟩	🟩					
Software Design Description	NOT STARTED											🟩	🟩				
Project Report	NOT STARTED												🟩	🟩	🟩		
Presentation	NOT STARTED														🟩	🟩	🟩

Work Plan 408



Works	Current State	17 February 2025	24 February 2025	3 March 2025	10 March 2025	17 March 2025	24 March 2025	31 March 2025	7 April 2025	14 April 2025	21 April 2025	28 April 2025	5 Mayıs 2025	12 Mayıs 2025	19 Mayıs 2025	26 Mayıs 2025
Project Work Plan	Completed	🟧	🟧	🟧	🟧											
Literature Review Update	Completed	🟦	🟦	🟦	🟦											
SRS Update	Completed	🟨	🟨	🟨	🟨											
SDD Update	Completed	⬛	⬛	⬛	⬛											
Test Plan Document	Completed					🟦	🟦									
Midterm Interactive Demo	Completed								🟨							
Final Product 1st Release	Completed	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩				
User Manual	Completed											🟥	🟥			
Project Report	Completed												🟦	🟦		
Project Tracking Form	Completed												🟦	🟦		
Test Results	Completed												🟦	🟦		
Project Poster	Completed													🟫	🟫	
Updated Project Webpage	Completed													🟫	🟫	
Demo Video	Completed													🟫	🟫	
Presentation	Completed															🟪

Conclusion

The **"LieWave"** project demonstrates the potential of using advanced speech analysis and machine learning to detect deception. By extracting and analyzing acoustic, prosodic, and emotional features from audio, the system provides an efficient and scalable solution for classifying speech as truthful or deceptive.

The integration of deep learning models like CNN and LSTM, along with traditional methods such as SVM, ensures robust and accurate performance. A user-friendly interface, secure backend, and scalable database enhance the system's usability and reliability.

This project offers valuable applications in fields such as security, forensics, and psychological research, providing a technological alternative to traditional lie detection methods. The results and insights gained from this project lay the foundation for further improvements and advancements in speech-based deception detection systems.