Speak Guard - Development of a multi-platform Application for Speech Analysis

Final Report

Start Date: 01/10/2023

by

Michelle Lau
11917662

Under the supervision of

Dr. Horst Eidenberger

Media Informatics and Visual Computing UE 033 521
University of Technology Vienna

Contents

1 Project Description **1**
1.1 Use Case 1
1.2 Task 1
1.3 Class Diagram 1
1.4 Timeline 2

2 Implementation **3**
2.1 System environment 3
2.2 Libraries 3
2.3 Bugs 3
2.4 Implementation Notes 4
2.5 Potential for Further Development 4

1 Project Description

Speak Guard is a Flutter mobile application that analyzes the user's speech through AI technology provided by Google. This application aims to transcribe spoken words in real time and then classify the transcript using AI, providing the user with insights into their speech patterns.

1.1 Use Case

The primary use case for this application involves individual people aiming to improve their public speaking or communication skills by taking advantage of AI technology to analyse their speech patterns. Users can record their speech directly within the app, which then transcribes their words in real-time. This immediate feedback allows users to see and observe what they have said.

The app then uses Vertex AI provided as one of the Google Cloud APIs to classify the transcript, which offers an analysis of the user's use of language and potential areas of improvement. This could be particularly beneficial for professionals such as teachers or comedians who rely heavily on communication in their roles or any individual who strives to improve their language.

1.2 Task

The primary objective of this thesis is the development of a mobile application using the Flutter framework. It is designed with the purpose of analysing recordings provided by the user including detecting vulgar speech with elements such as sexism, hate and racism. It is specifically noted that third-party libraries should be used wherever feasible, to take advantage of existing technologies and so that the wheel does not have to be reinvented.

In the planning phases of this application, it was determined that cloud services offer a good solution to our requirements. These services host pre-trained AI models that are capable of performing speech analysis and categorizing text passages. By utilizing these AI models, the app can efficiently evaluate speech for vulgarity and negative elements, providing users with immediate feedback on their language use.

1.3 Class Diagram

The class diagram provided below illustrates a potential workflow of the application from start to finish. The process begins with the 'Start' phase, where user is immediately lead to the 'Login' phase, an important step for ensuring secure access to the Google Cloud API. Upon successful authentication, the user is then directed to the 'Main Page', serving as the homepage with navigation to other tabs such as 'Settings' or 'History'. The 'Recording' phase follows after the user presses a button 'Record your speech', where users can start recording their speech. Once it is complete, the application transitions to the 'Analyse Speech' phase, where the recorded speech is processed, transcribed and sent to the Google Cloud API for analysis. Upon receiving a response from the API, the app leads the user to the 'Report with Result' phase, where they are presented with an analysis and transcript of their speech. This diagram illustrates the logical progression through the application and provides a visual representation of the user's journey from start to finish.

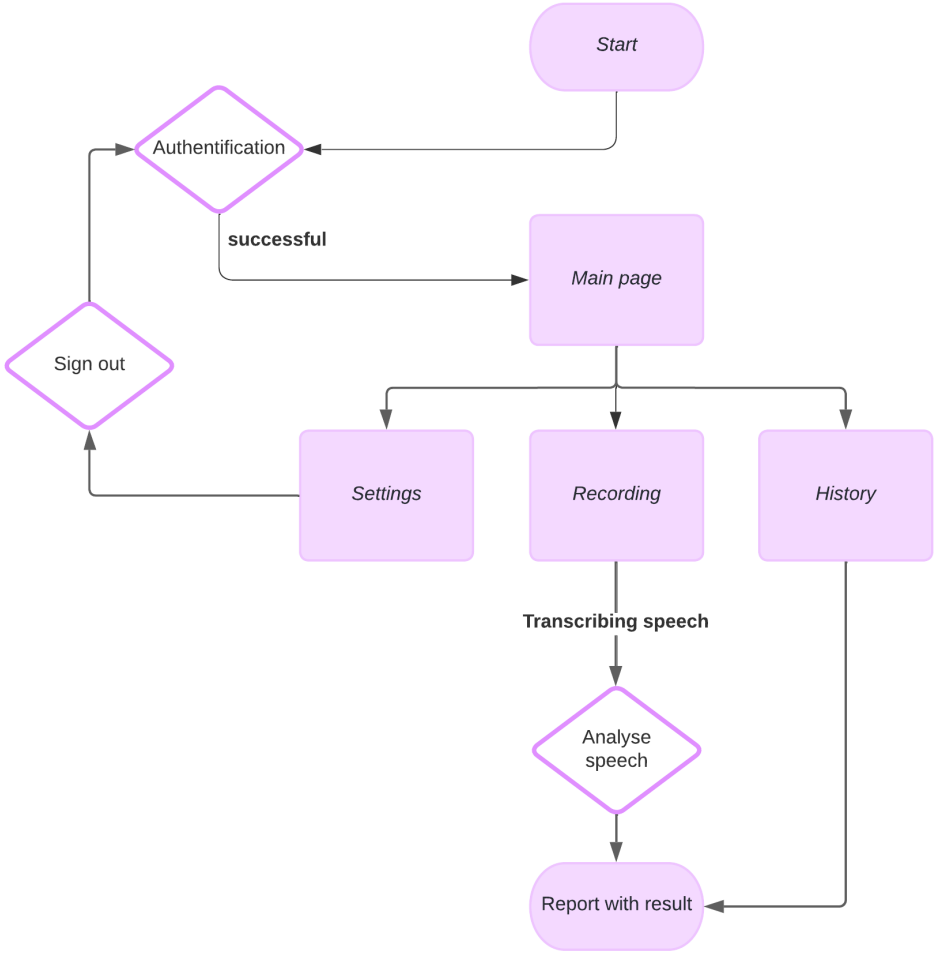

Figure 1: This is a simple diagram that showcases the workflow of the app.

1.4 Timeline

The development timeline for Speak Guard was sectioned into different phases for a structured progress from concept to completion.

- By end-November 2023: A frontend mockup prototype was developed with AdobeXD and suitable technologies for the project were evaluated.
- By end-November 2023: During this period the focused was on deepening the understanding of chosen technologies and starting frontend development based on the mockup.
- By end-December 2023: Implementation started with the aim to complete the frontend and user interface. The Text-to-Speech (TTS) functionality and the user interface were also finished and functional.
- By end-January 2024: Cloud services were slowly starting to get integrated for advanced features and documentations were prepared.

- By end-February 2024: We finished integrating cloud services for advanced features, and conducted some testing and polishing the app for optimal user experience.

This timeline highlights the approach taken during the development of Speak Guard.

2 Implementation

2.1 System environment

The system environment for the Speak Guard project was carefully selected in order to establish efficient development. The application was built using Flutter, a wide-spread and powerful framework that supports the development of applications for mobile, web, and desktop from a single codebase. This choice was made during the planning of the development process to ensure a consistent user experience across different platforms. Various libraries offered by Flutter including speech-to-text among many others were used. Google Cloud services, particularly its VertexAI APIs, were integrated to handle speech analysis. The development and testing phases primarily utilized an Android Emulator (Pixel 3a API 34, extension level 7, x86_64). This setup, combined with Dart as the programming language and multiple third-party libraries for additional functionality, helped achieving the project's objectives of analyzing speech for elements of vulgarity, sexism, hate, and racism.

2.2 Libraries

- speech_to_text (*speech_to_text 6.6.0*, n.d.)
- flex_color_scheme (*flex_color_scheme: ^7.3.1*, n.d.)
- path_provider (*path_provider: ^2.1.2*, n.d.)
- file_picker (*file_picker: ^6.1.1*, n.d.)
- firebase_core (*firebase_core: ^2.25.4*, n.d.)
- firebase_storage (*firebase_storage: ^11.6.5*, n.d.)
- google_sign_in (*google_sign_in: ^6.2.1*, n.d.)
- googleapis (*googleapis: ^12.0.0*, n.d.)
- googleapis_auth (*googleapis_auth: ^1.4.1*, n.d.)
- http (*http: ^1.2.0*, n.d.)

2.3 Bugs

As this project is part of an academic thesis, it has undergone limited testing, primarily on an Android emulator (Pixel 3a API 34, extension level 7, x86_64). During the development, several bugs were identified. These include occasional inaccuracies in speech transcription, particularly in noisy environments or with fast-speaking users, which may potentially affect the result of the analysis. Therefore, a confidence value for the conversion from speech to text was implemented and the user can observe any fluctuations in the value. During the development, some integration issues such as delays or errors when using the AI analysis services were also observed. This is likely due to memory management issues within the app. While efforts have been made to fix these problems, it is important for users to be aware of these known bugs.

2.4 Implementation Notes

In the current implementation of Speak Guard, there exists a history tab that displays past recordings and their analysis results. It is important to note that this feature is not yet connected to an actual database. Instead, the entries seen within the history tab are hard-coded examples designed to illustrate reports. This decision was made to prioritize developing the core objectives and functionalities of the application, which is focusing on real-time speech recording and analysis. Further development would be required to integrate a dynamic history tracking system.

2.5 Potential for Further Development

There is a lot of potential for further development of Speak Guard. A significant improvement would be the implementation of a dynamic database for the history tab. This would enable users to access a record of their past recordings.

Additionally, an introduction to stricter security measures could safeguard user data more efficiently, ensuring that any kind of sensitive information or recordings remain hidden and protected against unauthorized access.

Another area for improvement is the depth of speech analysis. Currently, the application only categorizes speech into predefined categories without pinpointing the exact segments that are deemed vulgar or offensive. Future developments could involve a more thorough analysis, offering users detailed feedback on exactly which parts of their speech were considered vulgar and why. This could include highlighting specific words within the transcript and providing suggestions for possible alternatives. Such improvements would increase the educational value of Speak Guard significantly. Illustrating how to use a reference.

References

file_picker: ^6.1.1 (n.d.). https://pub.dev/packages/file_picker. Accessed: 14.02.2024.

firebase_core: ^2.25.4 (n.d.). https://pub.dev/packages/firebase_core. Accessed: 14.02.2024.

firebase_storage: ^11.6.5 (n.d.). https://pub.dev/packages/firebase_storage. Accessed: 14.02.2024.

flex_color_scheme: ^7.3.1 (n.d.). https://pub.dev/packages/flex_color_scheme. Accessed: 14.02.2024.

googleapis: ^12.0.0 (n.d.). https://pub.dev/packages/googleapis. Accessed: 14.02.2024.

googleapis_auth: ^1.4.1 (n.d.). https://pub.dev/packages/googleapis_auth. Accessed: 14.02.2024.

google_sign_in: ^6.2.1 (n.d.). https://pub.dev/packages/google_sign_in. Accessed: 14.02.2024.

http: ^1.2.0 (n.d.). https://pub.dev/packages/http. Accessed: 14.02.2024.

path_provider: ^2.1.2 (n.d.). https://pub.dev/packages/path_provider. Accessed: 14.02.2024.

speech_to_text 6.6.0 (n.d.). https://pub.dev/packages/speech_to_text. Accessed: 14.02.2024.