Audio Recognition App

Ethan Knop, Zachary Vanscoit, Shaun Lai

CS 435 001 Software Engineering

12/03/2019

Table of Contents

Abstract

Introduction
- Background
- Problem
- Motivation
- Project Scope
- Development Model
- Report Outline

Project Requirements
- Functional & Non-Functional
- Use Case Diagram
- Sequence Diagram
- Class Diagram

Project Implementation
- Sprint Discussion
- Software Architecture
- Screenshots & Figures
- Test Cases

Conclusion & Future Enhancements

References

Appendices

Abstract

Our project Audio Recognition App is a cross-platform application designed to provide a simplistic way of classifying sounds provided by the microphone of the device using interchangeable machine learning models.

Introduction

Background

Our experience mainly resides in working in web development and front-end design. The suggestion of being able to deploy an app before the presentation of this project meant it needed to be of a manageable size for the students partaking in the endeavor. Many ideas were cycled through but the idea of a mobile app that uses a web managed interface meant we could be in a familiar environment while being able to produce something within time. The libraries that eventually became decided on were P5JS because 1 of the members had extensive experience using processing 3 with Java, and a new library called ML5JS which was built for P5JS meaning easy integration. There are tons of text to speech applications out there that try to emulate human speech but not too many that do the reverse.

Problem

Sounds vary by some degree, especially speech from person to person. Human dialect is such a variance that even causes us to be confused as to what the intended word was. This will cause errors as we don't have access to every type of way to say a singular word, but we can use an easy way to train models ahead of time and just load them into the application through a reference link. Mobile phones do not have the strongest processing abilities to train entire models of machine learning data either. To get around this problem we turned to Google's Teachable Machine to train the models for us, and allow for anyone to switch the model, and then reload it in-app.

Motivation

The motivation came from the discovery of ML5JS which is a library that integrates easily with another easy to implement library called P5JS. As explained these are web application libraries meaning they need to run in a web browser. This is where Cordova comes into play as it simulates a web browser for a base phone application.

Project Scope

This project was initially aimed to be a translation application but with time constraints, we came up with a simple audio recognition app that can be further tweaked into any other application in the future. This application collects users

voice input and by using machine learning models, it will determine what the user has said and outputting the results to the user.

Development Model

We utilized Agile for the flexibility of requirements it allows and for the frequent iterations of development.

Report Outline

This report

Project Requirements

Our requirements were initially brainstormed the first few meetings and as work progressed we refined our requirements to what they are currently.

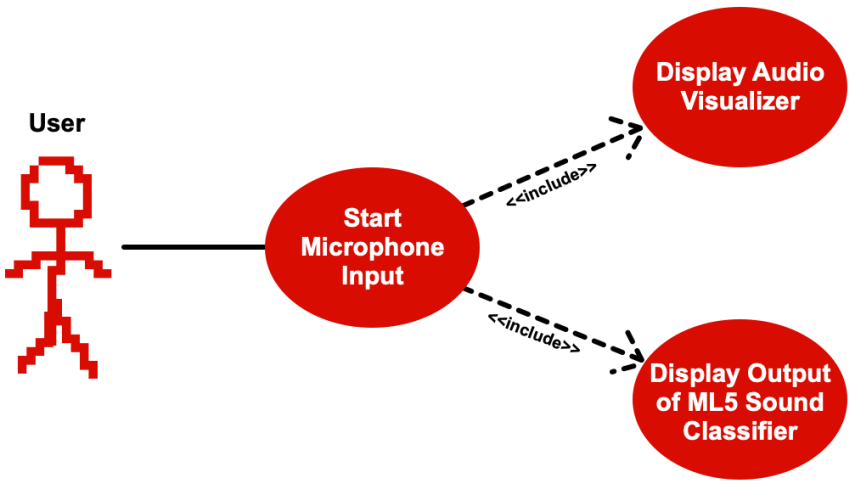
Non-functional User Requirements:

- Classify sounds from microphone input
- Display visualizer of microphone input
- Display HTML and CSS interface
- Allow use cross-platform

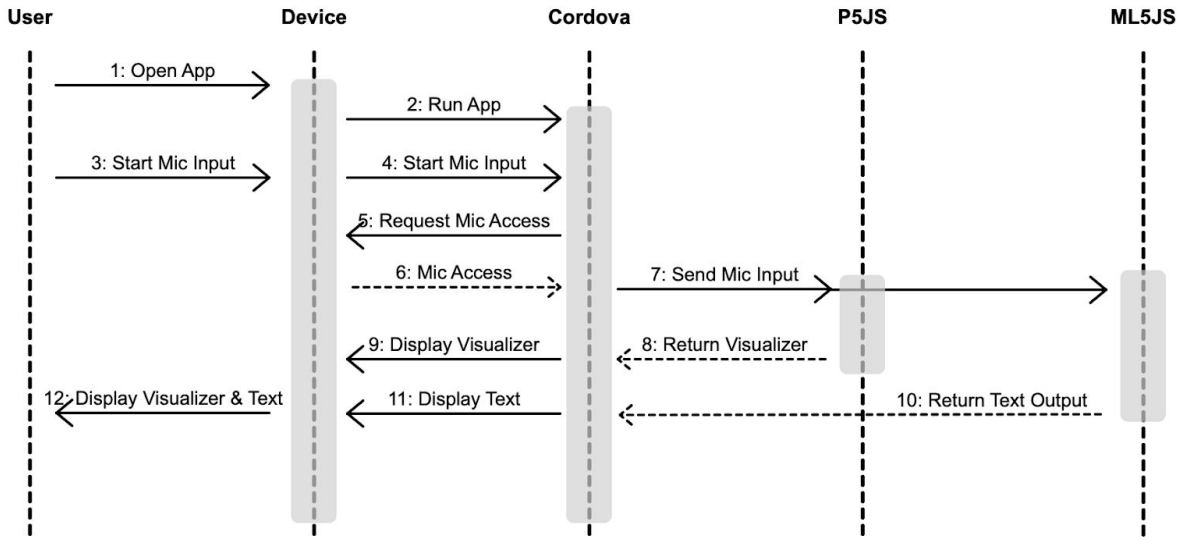
Functional System Requirements:

- Use ML5JS for sound classification
- Use P5JS for visualizer
- Build upon Apache Cordova for cross-platform HTML CSS & JS apps

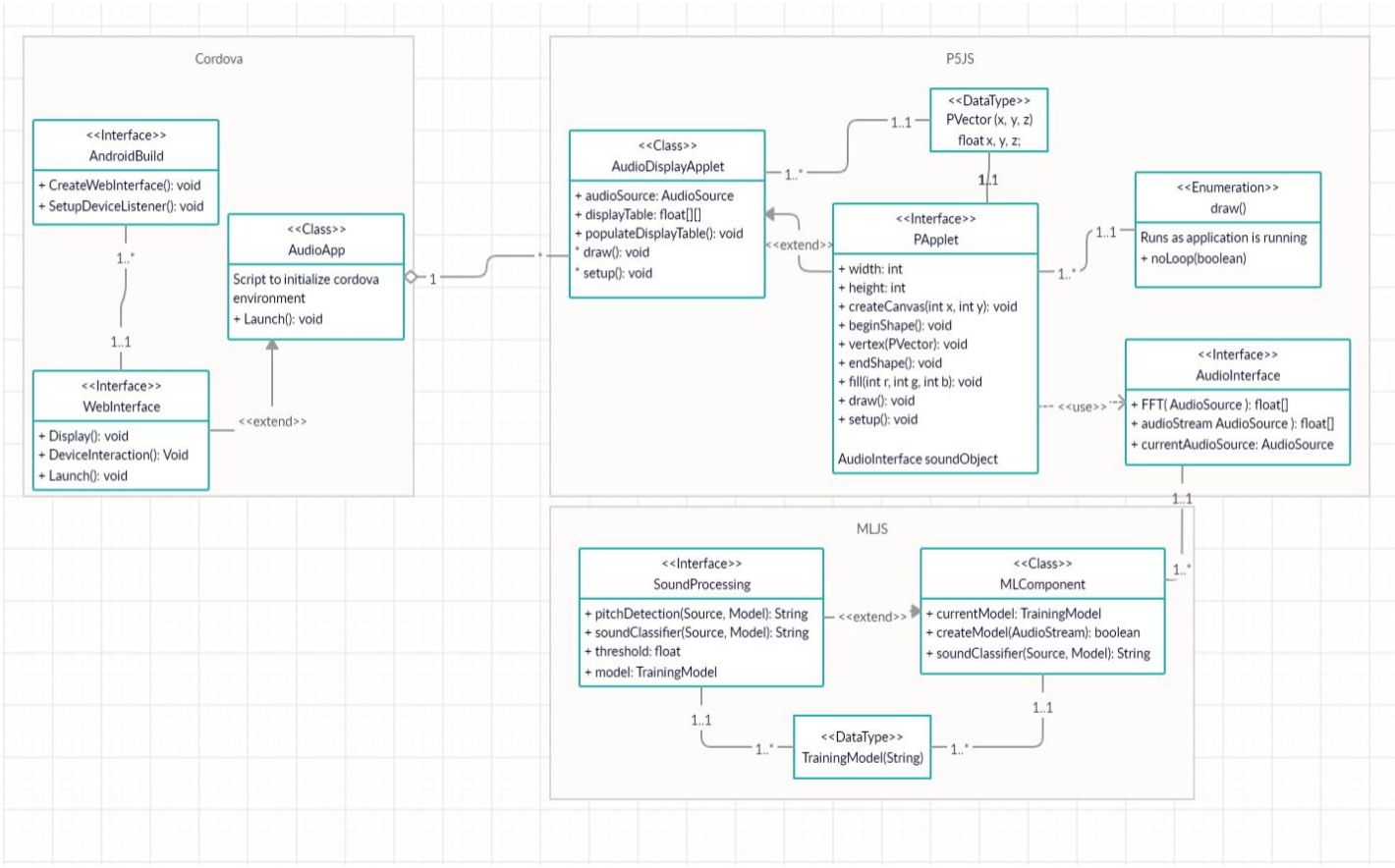
Use Case Diagram:



Sequence Diagram:



Class Diagram:



Project Implementation

Sprint 1:

We developed the initial idea of an app in which speech is translated into text which is then translated into a different language. After meeting and discussing we decided it would at least use Apache Cordova and develop a UI with HTML and CSS. We would use ML5JS to translate speech into text and P5JS to display an audio visualizer.

The sprint ended with a working UI and a test visualizer saved in a git repository for easier collaboration. Cordova was successfully implemented and the app could be run on an android device as well as a desktop browser which were both tested. A use-case diagram, sequence diagram, and class diagram were created but needed refinement.

Sprint 2:

The requirement that the app should translate text into a different language was discarded and our focus shifted to simply developing a machine learning model to classify a small range of speech and other sounds.

By the end of the sprint we had successfully created a machine learning model which could identify a small selection of words. The p5 audio visualizer was also completed, and both the visualizer and output of the ml5 model were implemented into the user interface using a combination of HTML, CSS, and Javascript.

Sprint 3:

At the end of sprint 3 we have completed the app and updated our sequence diagram, class diagram, and use-case diagram, and slightly modified our test cases. Our app works well in the desktop browser but when running as a Cordova application on a mobile device the app fails to secure access to the microphone of the device.

Software Architecture

Model-View Controller

Model: ML5JS Audio Classifier, P5JS Visualizer

View: HTML CSS & Javascript UI

Controller: Cordova

Screenshots and figures

Test cases

Test Priority (Low/Medium/High): Medium	**Test Designed date:** 11/14/2019
Module Name: User Interface	**Test Executed by:** Ethan
Test Title: Screen center	**Test Execution date:** 11/14/2019
Description: Ensure UI elements remain in center of any screen size.	
Pre-conditions: New variable to display ratio of UI element positions to viewport size of Chrome	
Dependencies: Javascript, CSS, HTML, Google Chrome	
Step 1 **Test Steps** 1 **Test Data** Width of viewport, width of UI element, UI element position on page **Expected Result** The position of the UI element from the center is the UI's position on page plus half of the UI element's width, this value should be equal to half of the viewport width. **Actual Result** The UI element's position measured from its center is equal to half of the viewport's width.	

Status (Pass/Fail) Pass **Notes** In the linked javascript file, I wrote code so that every time the viewport's size is changed it will print to the console the value of comparing the element's position on the page plus half the value of its width to half the value of the viewport width, and it evaluated to equal every time.

Test Priority (Low/Medium/High): Medium	**Test Designed date:** 9/11/2019
Module Name: Audio Visualizer	**Test Executed by:** Shaun
Test Title: Design a working Audio Visualizer	**Test Execution date:** 12/11/2019
Description: Design a simple and visually appealing audio visualizer.	
Pre-conditions: Must detect voice input	
Dependencies: P5JS, Javascript, CSS, HTML, Google Chrome	
Step Test Steps Test Data Expected Result Actual Result Status (Pass/Fail) Notes 1 2 Provide voice input properly. Detect Voice input Audio input and audio visualizer working Audio input works but not the visualizer Fail Research Detect Voice input Audio input and audio visualizer working Provide voice input properly. Both audio input and audio visualizer work Pass Try to make audio visualizer more appealing 3 Provide voice input Audio Threshold Must not surpass 100 Decibels Frequency bands are capped at 100 Decibels Pass Test case on audio input (shouldn't be more than a certain threshold)	

Conclusion & Future Enhancements

This project has given us the opportunity to work with different tools that we have never used before such as p5.js, lm5js ,apache cordova and also different machine learning models. With this project, it has provided us a glance of what the real world

software engineers are actually doing with developing a product. Teamwork is a huge role in this project and without it, we wouldn't be able to complete this project in a timely manner. We are developers that strive to provide users with a complete, fun and useful application.

Future enhancements of the application would be adding a few options where users will be able to enter their own word of choice for the application to recognize as the application currently only recognizes a few key words. Could possibly use our current application as a template and convert it into a game where if users mentions certain words correctly, they earn points.

References

- L. McCarthy, C. Reas and B. Fry, Make: Getting started with p5.js. San Francisco, CA: Maker Media, 2016.
- L. McCarthy, "home | p5.js", P5js.org, 2019. [Online]. Available: https://p5js.org/. [Accessed: 12- Oct- 2019].
- D. Shiffman, W. Li and J. Lee, "ml5js·Friendly Machine Learning For The Web", Ml5js.org, 2019. [Online]. Available: https://ml5js.org/. [Accessed: 13- Oct- 2019].
- "Apache Cordova", Cordova.apache.org, 2019. [Online]. Available: https://cordova.apache.org/. [Accessed: 13- Oct- 2019].

Appendices