Final Project Report

for

Perceptual Video Quality Data Acquisition and Analysis Tool

Version 1.0

Prepared by

California State Los Angeles Senior Student Team

Deanna Thomas (Lead)

Ponaroth Eab

Daniel Ramirez

Nelson Hyunh

George Beltran

Project Advisor

Mark Sargent & Dr. Elaine Kang

Liaison

Harrison L. Hays V

Peshala V. Pahalawatta

Ross Castillo

May 8, 2020

Revision History

Name	Date	Reason For Changes	Version
Initial Draft	2020-5-4		1.0

Table of Contents

1. Introduction

1.1 Project Background

AT&T DirecTV is starting to switch from satellite to streaming. When users were watching television shows and movies through a satellite, video quality wasn’t an issue because the hardware that would be installed in everyone’s house would be the same. However, as more and more users use streaming services, the hardware that they will be watching videos on will be varying. AT&T has to accommodate their video quality according to what device the user will be watching from and how well the internet connection is. For AT&T to figure out the best quality they can distribute, a lab was created to compare videos being displayed in different hardware with different variations in bitrate, frame rate, resolution, etc. One issue with this method is that it’s extremely costly. Tons of money is spent on technology to test videos including, but not limited to, 4k televisions, Amazon Fire Sticks, AppleTVs, and iPads. Another issue is that employees of AT&T who work at the lab may claim a video has a low quality experience for a user but in reality, the user might not notice the drop in quality at all.

1.2 Existing Implementation

The idea for a video quality app was proposed and conceptualized by AT&T to gather and collect data about video quality more accurately and efficiently. Thereby reducing the cost needed to improve streaming quality. A video comparison application was created by the Harvey Mudd College as part of their Clinic Program. The app was able to survey and collect data about what type of video quality users prefered but it was only compatible with iPads, and there were additional issues in concern with the user interface and user experience. A web application was created to act as an admin tools dashboard and allowed the admin to add and delete videos and collections in the comparison app. The admin also had the ability to hide collections from being shown on the app. There were plans to have more features like tag searching but they were left incomplete before the Clinic Program ended.

1.3 Team Implementation

Improvements have been made to the existing video comparison application and the web application. An absolute rating application was created by us to allow a user to rate a single video based on the given knowledge about what both a high quality video and low quality video should look like. Both the video comparison app and the absolute rating app were updated to be compatible with Android, IPhone, IPad, and AppleTV. The web application completed the features that were left incomplete by the Harvey Mudd College and added more features to simplify the experience and to help with data analysis.

1.4. Key Design Principles

A major issue that our liaisons at AT&T noticed with the app before we started working on it was that users wouldn't get any feedback after selecting their choice. Our approach to solve this issue was to give users information that might be intriguing to them. The user would be informed about how their response compared with other users and they will be shown the amount of times they responded similar to a points system. This was designed to encourage users to continue rating videos on the app so more data can be gathered. The web application had a modern and flat design that we decided to keep. The main thing we changed was to have the features of the admin tools dashboard be displayed on a sidebar instead of a tab bar because the tab bar limited the amount of features that can be displayed on to the web app and wasn't future proof. Many of the new features we added were designed to match the original look created by the team before us to make the entire admin tools dashboard cohesive.

1.5 Design Benefits

Both video comparison and absolute rating apps are designed with simplicity so the user has no problem watching and rating videos. These two apps are meant to collect data from users as fast as possible. For the app to be simple and fast we first pre-load the video collections so when the user gets to the video collection page it is already loaded with the different collections. The flow of the applications are linear making the users not have to spend much time on the app.

The web app is designed with many tools that are easily accessed for an admin. Within the web app the admin can make changes to the database making it easy for them to pull up information or delete information. The web app being connected to the API is what makes it gain access to the database.

1.6 Distribution

AT&T plans to use the applications internally. Therefore, the mobile applications will be manually distributed by the liaisons to AT&T employees to collect data.

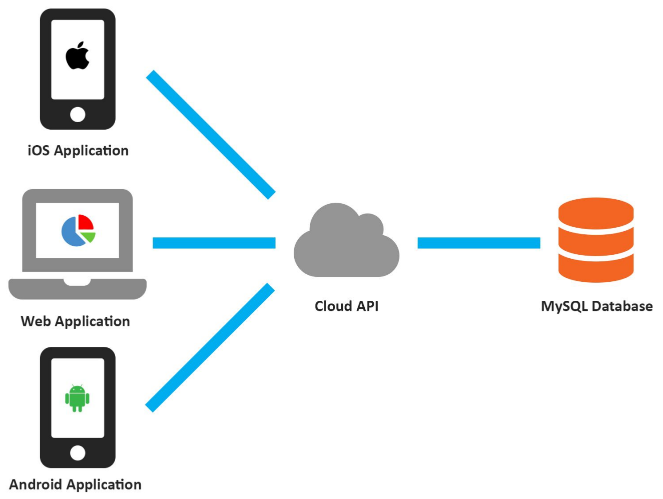
2. Related Works and Technologies

2.1 Existing technologies

We looked into existing libraries for both Video Comparison and Absolute Rating apps. One of the libraries that were used to display the results of the users in a pie chart or bar graph after they have rated a video is “Charts” for iOS and “MPAndroidChart” for Android. The library comes with many different options of charts and graphs that could be animated into the application. For iOS libraries like “Alamo fire”, “Reachability Swift”, and “SD Web Image” were all used to design features and pass data into the application. While for Android libraries such as “RXJava”, “OkHTTP”, “Retrofit”, and “Android Architecture Components” for the same reason as iOS. For the web app different node packages such as “material-ui”, “chart.js”, “react-table”, and “react-csv” were used for web design and downloading functions for the data table.

3. System Architecture

3.1. System Data Flow Diagram



Modules:

1. iOS Application
2. Android Application
3. Web Application
4. Cloud API
5. MySQL Database

3.2 Module Functions

Mobile Applications:

- Side-by-Side video quality rating application allows the user to rate which video was better out of two. One application for iOS and one for Android.
- Absolute rating application allows the user to rate a single video on a 1-5 scale. One application for iOS and one for Android

Web Admin Application:

- Allows the admin to manage videos, collections
- Allows the admin to view and analyze result data

API:

- Provides access to the database with different endpoints for the mobile applications and web admin application

Database:

- Stores videos, collections, users, and score data

3.3 Technologies
The iOS application was designed with the programming language “Swift”. Different libraries like “Alamo fire”, “Reachability Swift”, “Charts”, and “SD Web Image” were all used to design features and pass data into the application.

The Android application was designed using Kotlin. Different libraries such as “RXJava”, “OkHTTP”, “Retrofit”, “Android Architecture Components”, “MPAndroidChart” were all used to design features and pass data into the application. A form of MVVM architecture pattern was used.

The Web application was designed using JavaScript and React.js. Different node packages such as “material-ui”, “chart.js”, “react-table”, and “react-csv” were used for web design and downloading functions for the data table.

The API was designed with Node.js and hosted on cloud via Heroku.

The database was designed with MySQL and managed with phpMyAdmin.

3.4 Development Methods
We used Agile development methods. Each two-week sprint, we met on Monday, met with the liaisons to discuss progress and updates, then tasked out our work on our Trello board, which our faculty advisor and liaisons also had access to. We set goals for the sprint in our “To-Do” column with tasks from our backlog, and closed them out by the end of the sprint, when we worked on the Bi-Weekly reports. We allowed our project to be flexible and change as we worked by continuously integrating our liaison’s ideas into our design.

4. Results and Conclusions

4.1 Results and Conclusions

Our applications are completed and ready to be installed on respective devices. API and database can handle a large amount of users at the same time. Our software should satisfy the requirements to allow AT&T to collect data and accurately determine user preference. Implementation goals and stretch goals have been met. Through our development period, we discovered that streaming a large video file can result in stuttering because of slow internet connection or the device cannot handle the streaming rate.

4.2 Follow Up Work

Because of the short period of time we have to work on the project, we were not able to do a complete unit test on all software that we built. We also were not able to test our applications on every single device out there. The applications look fine on our available equipment. The follow up work would be to perform unit testing and perhaps adjust the user interface to fit some device that may run differently. Also, AT&T have to migrate the API host and the database from ours to theirs.

5. References

Swift.org - Swift programming language documentation (https://swift.org/documentation/)
Kotlin - Kotlin programming language documentation (https://kotlinlang.org/)
Javascript programming language documentation (https://devdocs.io/javascript/)
React.org - React JS documentation (https://reactjs.org/docs/getting-started.html)
Cocoapods for library manager - (https://cocoapods.org/)
Gradle - Build Tool documentation (https://gradle.org/)
Alamofire - Elegant HTTP in Swift (https://github.com/Alamofire/Alamofire)
Reachability.swift - Reachability for iOS (https://github.com/ashleymills/Reachability.swift)
Charts - Chart library for iOS (https://github.com/danielgindi/Charts)
SD Web Image - Image Downloader for iOS (https://github.com/SDWebImage/SDWebImage)
RxJava - Reactive extension for JVM (https://github.com/ReactiveX/RxJava)
OkHTTP - HTTP in Android (https://square.github.io/okhttp/)
Retrofit - Type-safe HTTP for Android (https://square.github.io/retrofit/)
MPAndroidChart - Android chart library (https://github.com/PhilJay/MPAndroidChart)
Node.js - Javascript Runtime Environment (https://nodejs.org/en/)
Material-UI - React UI framework (https://material-ui.com/)
React-table - Table package for React (https://www.npmjs.com/package/react-table)
React-csv - csv file generator for React (https://www.npmjs.com/package/react-csv)
MySQL - Relational database management system (https://www.mysql.com/)
PhpMyadmin - MySQL administration tool (https://www.phpmyadmin.net/)