

Senior Design Document

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

Liaison

Harrison L. Hays V

Peshala V. Pahalawatta

Ross Castillo

March 22, 2020

Revision History

Name	Date	Reason For Changes	Version
Initial Draft	2019-11-29		1.0
Revision	2019-12-8	Additional information	2.0
Revision	2020-3-22	Updated	3.0
Revision	2020-4-9	Updated	4.0

Revision History.....	2
Table of Contents.....	3
1. Introduction.....	4
1.1. Purpose.....	4
1.2. Document Conventions.....	4
1.3. Intended Audience and Reading Suggestions.....	4
1.4. System Overview.....	4
2. Design Considerations.....	5
2.1. Assumptions and dependencies.....	5
2.2. General Constraints.....	5
2.3. Goals and Guidelines.....	5
2.4. Development Methods.....	6
3. Architectural Strategies.....	7
4. System Architecture.....	8
4.1. Level 0	8
5. Policies and Tactics.....	9
5.1. Specific Products Used.....	9
5.2. Requirements Traceability.....	9
5.3. Testing the software.....	9
5.4. Guidelines and conventions.....	9
5.5. Building the Code.....	9
6. Database Design	10
7. User Interface.....	12
7.1. iOS/tvOS User Interface	12
7.2. Android/AndroidTV User Interface	14
7.3. Web App User Interface	15
7.4. Mobile Application Data Flow	21
8. Requirements Status	22
9. Glossary.....	27

1. Introduction

1.1 Purpose

The purpose of this document is to identify the software requirements for the iOS app, database and administration tool web application. The iOS app collects data from users and sends it to the database. Then, the administration tool can communicate with the database to view the data, alter, or export it.

1.2 Document Conventions

All text for the document will be single-spaced, size 12pt, and Calibri font; however, the headers will be size 24pt, Calibri font, and bolded. Bullet points are used to list or outline specifications for ease of readability.

1.3 Intended Audience and Reading Suggestions

This document is intended for the AT&T engineers who will be utilizing the system, as well as for any other engineer who may expand or otherwise work on it.

1.4 System Overview

The system contains a MySQL database which stores video rating data. A Node.js server provides API endpoints to send, receive, and modify data. A web application is used as an admin tool in which the data can be analyzed and visualized, and, in addition, video data can be added and modified. Lastly, Android and iOS apps are created to collect the data by allowing users to rate the quality of the videos.

2. Design Considerations

- Compatibility with multiple mobile devices
- Compatibility with multiple web browsers
- Communication between mobile app, web application and database
- Display will vary and change slightly depending on the platform and screen sizes being used.
- Swift programming language and libraries.
- React JS and React libraries.

2.1 Assumptions and Dependencies

This software was intended for internal AT&T use. Thus, we had a lot of flexibility in terms of design limitations and constraints. For example, in order to avoid adding advanced encryption techniques to log-in to the admin portal, the liaison was satisfied with simply running the React application tool on his local machine. In addition, we did not need to worry about preparing the application for widespread distribution, such as via the Apple Store or Google Play store. The engineers frequently gave feedback regarding new feature ideas for the system, but we have not gotten and do not anticipate to receive major modification requests to the system.

2.2 General Constraints

Because the software was intended for internal use, we did not have significant constraints. However, in terms of collecting data, AT&T only was licenced to use the video clips in a laboratory setting. Therefore, we would not be able to distribute the application to get more data.

2.3 Goals and Guidelines

The system had already been worked on for one year by seniors at Harvey Mudd college. Thus, a lot of the basic parts were completed. However, we needed to make several modifications.

- The liaisons were not convinced that the NoSQL database was the best way to store and represent their data. Therefore, we decided to migrate to a MySQL database.
- The liaisons described a lot of “friction” in the existing iOS application which made it difficult to use and discouraged people from participating in rating videos
- The liaisons were interested in adding an Android app, which would run on Android TV devices as well be easily distributable to internal AT&T employees.
- The liaisons wanted a way to easily view and analyze their data, as well as be able to prove or disprove key hypotheses that they had about video quality
- We also wanted it to be easier for the liaisons to manage their videos and video collections without having to directly access the database.

- We wanted better code quality by implementing design patterns as well as implementing unit testing and continuous integration.

2.4 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 integrating our liaison’s ideas into our design.

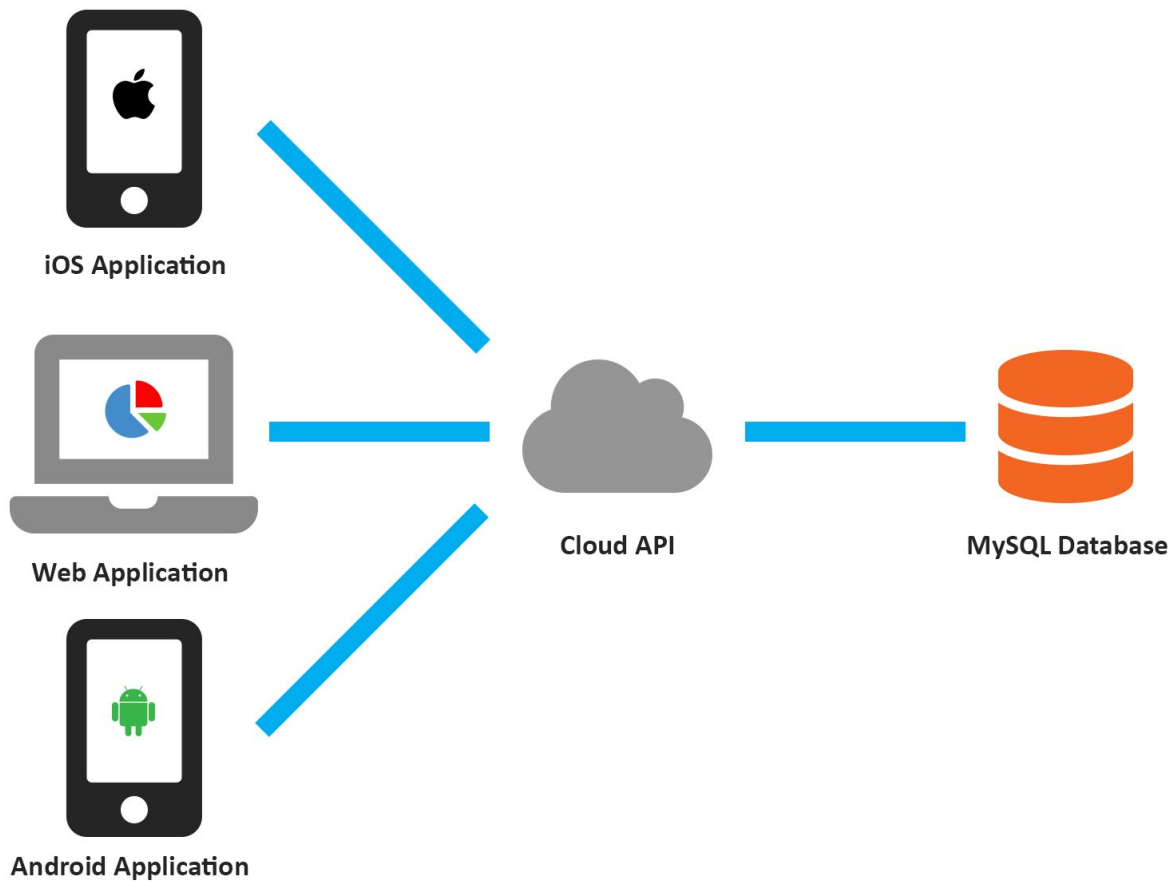
3. Architectural Strategies

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”, “MPCharts” 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 React.js. The API was designed with Node.js.

4. System Architecture



Modules:

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

5. Policies and Tactics

5.1 Choice of which specific products used

All iOS and tvOS applications were developed on Xcode using Swift.

Different libraries like “Alamofire”, “Reachability”, “Charts”, and “SD Web Image” were all used to design features and pass data into the application.

The Android application was developed on IntelliJ using Kotlin. Kotlin has been declared Android’s official development language as opposed to Java as of 2019 and was therefore chosen for this project.

The web application was developed using React.JS.

The cloud API was developed with Node.js and hosted using Heroku.

The database was migrated from Firebase (Non-relational) to MySQL.

5.2 Plans for ensuring requirements traceability

We have used an Agile development system and have been organizing tasks as items on our Trello Board. All tasks are assigned and can be viewed per sprint on Trello after they have been tested and merged.

5.3 Plans for testing the software

To generate data, we will be using our mobile applications as well as having other people use the application. By using the product, we can thoroughly test it to ensure we are getting the correct data before handing off to the liaisons.

The cloud API will use unit testing to ensure that the correct data is being sent to and retrieved from the database.

5.4 Coding guidelines and conventions

Code is managed via private repositories on GitHub. Everytime a task is completed, a pull request is made for the subsequent branch. After a brief code review, the code is merged.

5.5 How to build and/or generate the system's deliverables (how to compile, link, load, etc.)

An apple device is needed to build and run the iOS/tvOS applications.

To run the Android app, Android Studio must be installed.

To run the Web App, npm must be installed. It can be run by using the command “npm start”.

To run the API, npm must be installed. It can be run by using the command “npm start”.

6. Database Design

The database for this system is a relational database with MySQL.

Tables:

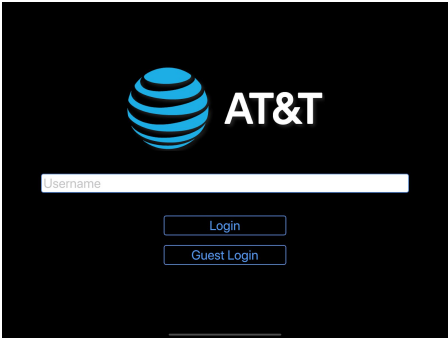
- collections
 - id - int(32) - auto increment - primary key
 - name - varchar(255)
 - thumbnail - text
 - hidden - tinyint(1)
- config
 - id - int(32) - auto increment - primary key
 - setting - varchar(255)
 - value - varchar(255)
- scores
 - id - int(32) - auto increment - primary key
 - user_name - varchar(255)
 - first_video - varchar(255)
 - second_video - varchar(255)
 - score - int(2)
 - device - varchar(255)
 - mirroring - tinyint(1)
 - brightness - double
 - rewatch - tinyint(32)
 - notes - text
- scores_absolute
 - id - int(32) - auto increment - primary key
 - user_name - varchar(255)
 - video - varchar(255)score - int(2)
 - score - int(2)
 - device - varchar(255)
 - mirroring - tinyint(1)
 - brightness - double
 - rewatch - tinyint(32)
 - notes - text
- tutorial
 - id - int(32) - auto increment - primary key
 - url - text
 - expected_rating - tinyint(2)
- users

- id - int(32) - auto increment - primary key
 - name - varchar(255)
 - score - int(32)
- users_absolute
 - id - int(32) - auto increment - primary key
 - name - varchar(255)
 - score - int(32)
- videos
 - id - int(32) - auto increment - primary key
 - bitrate - int(32)
 - codec - varchar(32)
 - color_space - varchar(32)
 - content - varchar(255)
 - eotf - varchar(32)
 - frame_rate - int(32)
 - original_resolution_height - int(32)
 - original_resolution_width - int(32)
 - output_resolution_height - int(32)
 - output_resolution_width - int(32)
 - url - text
 - name - varchar(255)
 - tags - text

7. User Interface

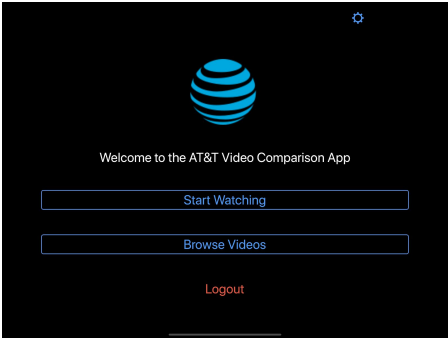
7.1 iOS/tvOS Overview of User Interface

I.



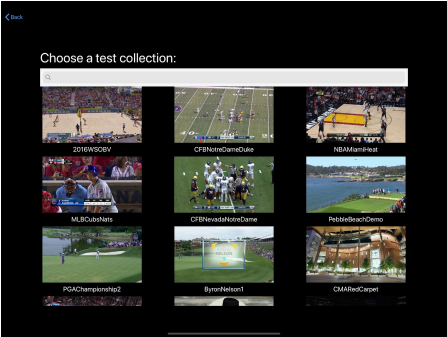
The login screen features the AT&T logo at the top. Below it is a text input field labeled "Username". At the bottom, there are two buttons: "Login" and "Guest Login".

II.



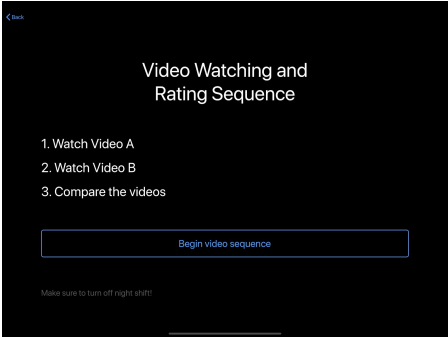
The welcome screen displays the AT&T logo and the text "Welcome to the AT&T Video Comparison App". It includes three buttons: "Start Watching", "Browse Videos", and "Logout".

III.




This screen prompts the user to "Choose a test collection:". It shows a grid of nine video thumbnails with titles: "2016WSDBV", "CFBHomeGameDuke", "NBAMainstreet", "MLBCubaHats", "CFBNewadelaideGame", "RebbedBeachDemo", "PGAChampioned", "BeverlyHillz1", and "GMHillCapeen".

IV.



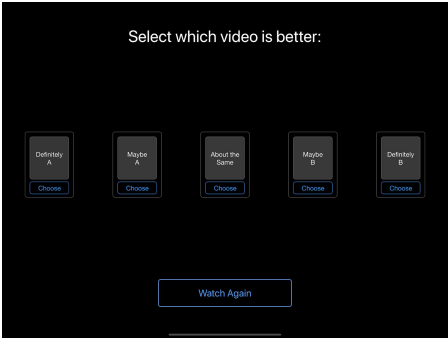
The screen is titled "Video Watching and Rating Sequence". It lists three steps: "1. Watch Video A", "2. Watch Video B", and "3. Compare the videos". A "Begin video sequence" button is at the bottom. A small note at the bottom says "Make sure to turn off night shift".

V.



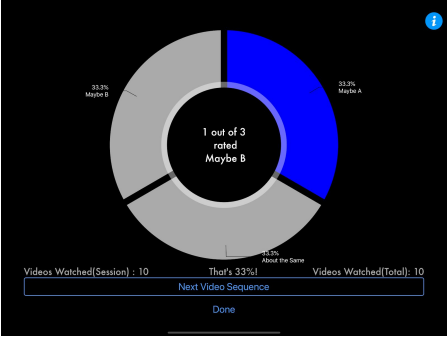
A screenshot of a basketball game in progress. The court is visible with players in white and dark jerseys. The scoreboard at the bottom shows "ROCKETS 24" and "HEAT 27".

VI.




The screen asks the user to "Select which video is better:". It presents five options: "Definitely A", "Maybe A", "About the Same", "Maybe B", and "Definitely B". Each option has a "Choose" button. A "Watch Again" button is at the bottom.

VII.



The screen shows a donut chart with the text "1 out of 3 rated Maybe B". It also displays "Videos Watched(Session): 10" and "Videos Watched(Total): 10". A progress bar at the bottom indicates the session is "Done".

VIII.



This screen compares technical specifications for Video A and Video B. The data is as follows:

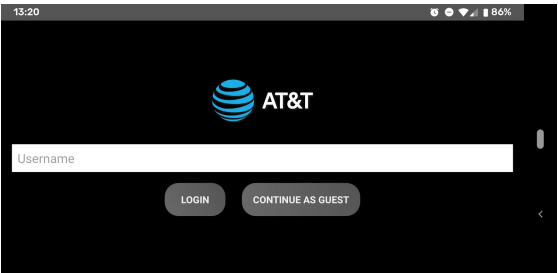
Key	Video A	Video B
output_resolution_width	640	1920
output_resolution_height	360	1080
color_space	bt709	bt709
codec	HEVC	HEVC
bitrate	850	3500
frame_rate	29	60

- I. Login Screen (Allows for user login or guest login)
- II. Decision Screen (Allows user to choose from Start Watching or Browse Videos)
- III. Collections Screen (Displays all collections available)
- IV. Video Sequence Screen (Displays instructions for rating videos)
- V. Video Player Screen (Plays both Video A and Video B)
- VI. Rating Screen (Allows users to selected a rating)
- VII. Data Screen (Displays data on all results between videos watched using a pie chart.
Allows users to watch another set of videos, return to Decision Screen, or see the video properties of the two videos.)
- VIII. Video Properties Screen (Displays the properties of the two videos watched.)

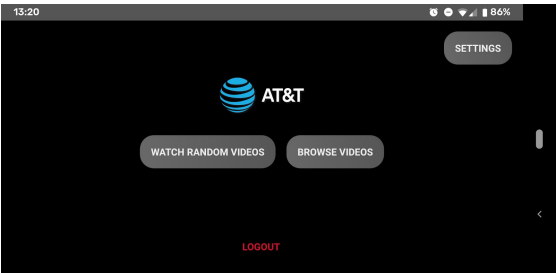
When the user opens any of the iOS/tvOS applications, they will see a screen that contains a textbox and two buttons for logging into the application with a username and logging into the application as a guest user. The next screen will give the user a choice to press one of two buttons, one to view random videos and another to search through all collections. If the random video button is pressed, the user will watch two randomly selected videos from a randomly selected collection. If the searched video button is pressed, the user will be displayed a list of collections to choose from. After the user chooses a collection, the user will view two randomly selected videos from that collection. Next, the user will watch the two videos retrieved and then decide which video, in their opinion, was higher quality. After the user submits their rating, they will be shown a graph that displays the percentages of how everyone else rated the two videos and where the user falls in that graph. There will also be three buttons on this screen for displaying video details, starting a new video sequence, or going back to the main menu. If the user decides to display video details, a new screen will be shown with the properties of each video such as fps, bitrate, and resolution.

7.1.2 Android Overview of User Interface

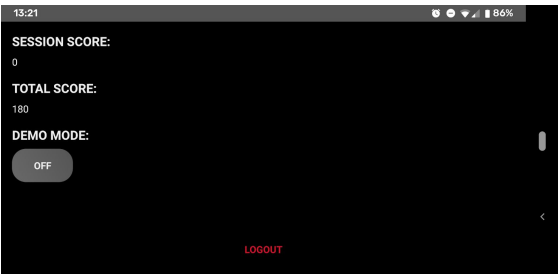
I.



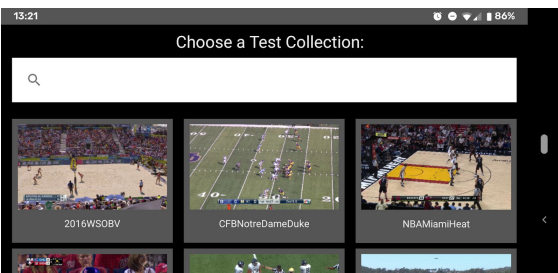
II.



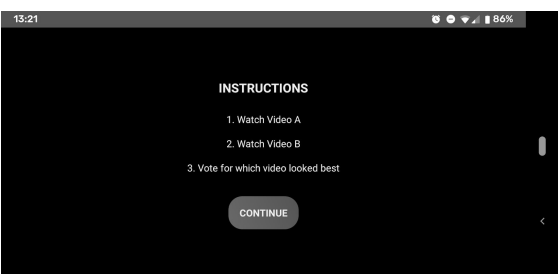
III.




IV.



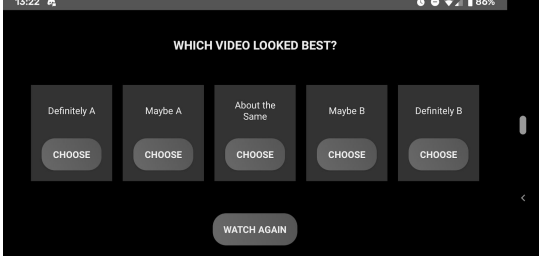
V.



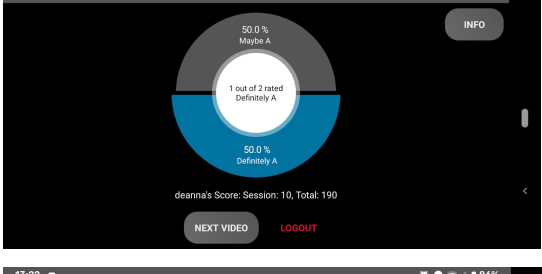
VI.



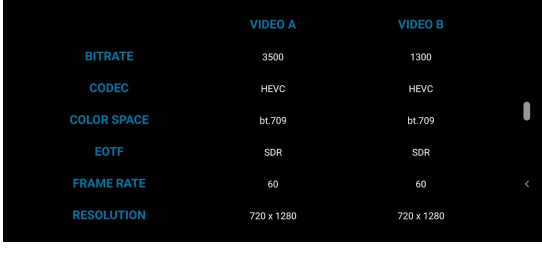
VII.



VIII.



IX.

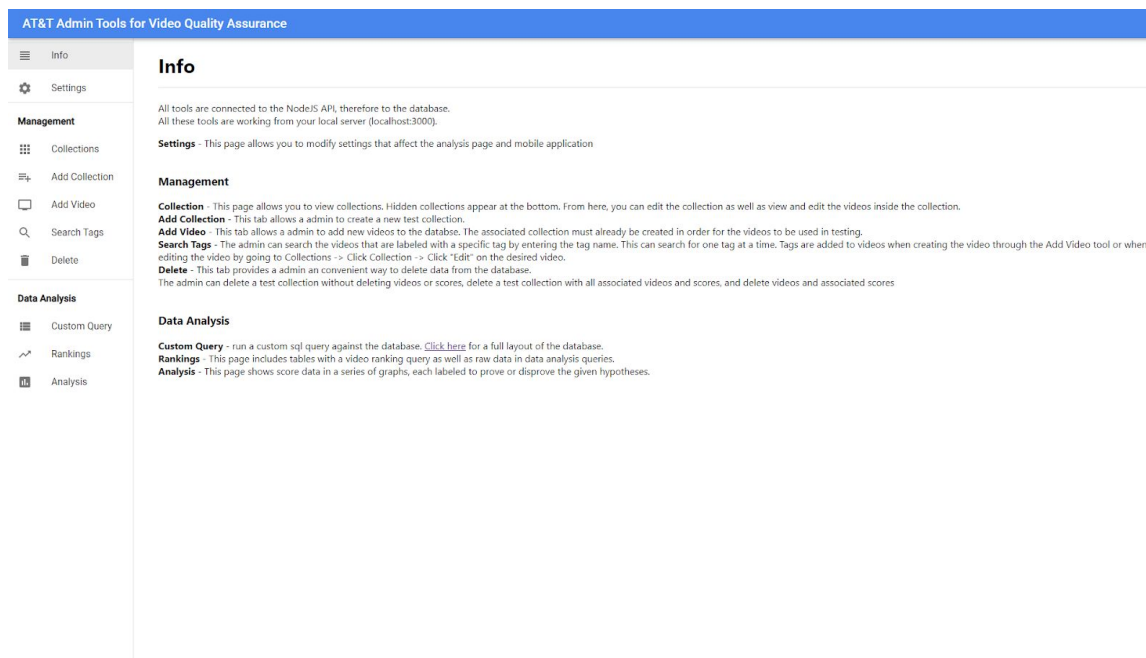


	VIDEO A	VIDEO B
BITRATE	3500	1300
CODEC	HEVC	HEVC
COLOR SPACE	bt.709	bt.709
EOTF	SDR	SDR
FRAME RATE	60	60
RESOLUTION	720 x 1280	720 x 1280

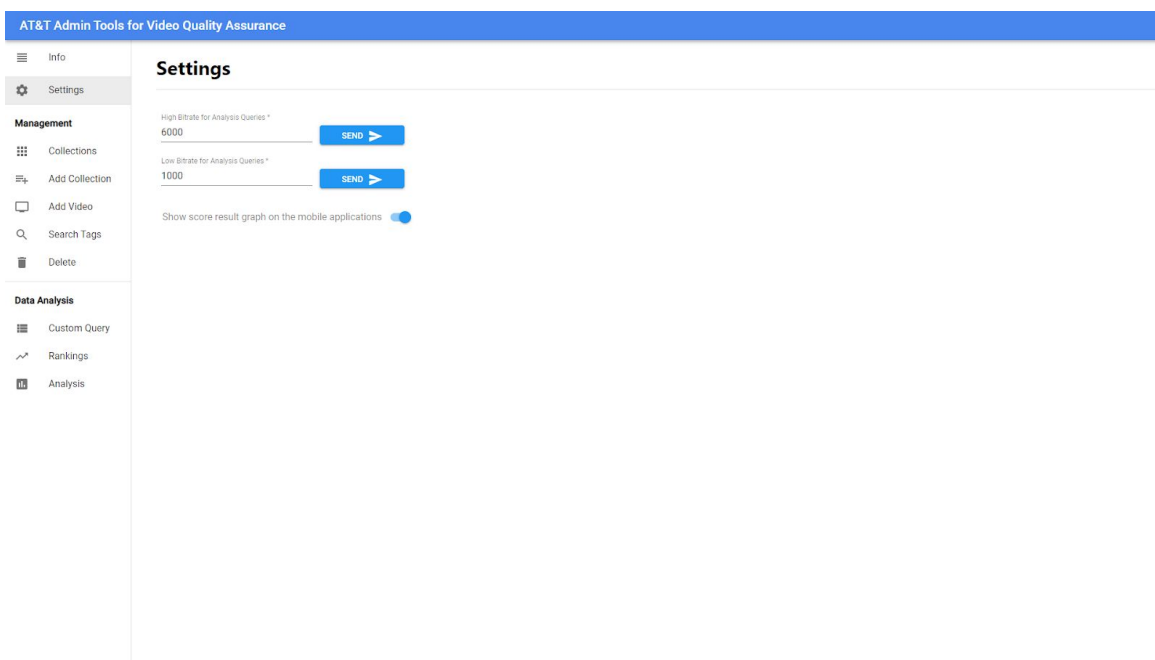
- I. Login Screen
- II. Decision Screen
- III. Settings Screen
- IV. Browse Videos Screen
- V. Instructions Screen
- VI. Video Viewer Screen
- VII. Rating Screen
- VIII. Results Screen
- IX. Video Details Screen

When the user opens the mobile application, they will see a screen that contains a textbox and two buttons for logging into the application with a username and logging into the application as a guest user. The next screen will give the user a choice to press one of three buttons, one to view random videos and another to view a searched video. In addition, they can access a settings page to view their user info or turn on “Demo Mode”, which will disable sending scores to the database for testing or demo purposes. If the random video button is pressed, the user will watch two randomly selected videos from a randomly selected collection. If the searched video button is pressed, the user will be displayed a list of collections to choose from. After the user chooses a collection, the user will view two randomly selected videos from that collection. When the user watches two videos from both options, the user will be shown a screen to rate each video and compare which one is better than the other video according to what they have seen by pressing one of five buttons. After a button is pressed, the user will be shown a pie graph that displays the percentages of how everyone else rated the two videos and where the user falls in that graph. There will be a button for the user to press so they can view the video properties for both videos. Two more buttons will also be on the graph screen to allow the user to either watch videos again or exit the application.

7.1.3 Web Overview of User Interface



VIII.



IX.

X.

AT&T Admin Tools for Video Quality Assurance

Info
Settings

Management

Collections
Add Collection
Add Video
Search Tags
Delete

Data Analysis

Custom Query
Rankings
Analysis

Collections

XI.

AT&T Admin Tools for Video Quality Assurance

Info
Settings

Management

Collections
Add Collection
Add Video
Search Tags
Delete

Data Analysis

Custom Query
Rankings
Analysis

Add a Collection

Name of Test Collection *

Note: Collection name cannot be edited after adding.

Hide Collection? ☐

Thumbnail URL *

SEND

XII.

AT&T Admin Tools for Video Quality Assurance

Info

Settings

Management

Collections

Add Collection

Add Video

Search Tags

Delete

Data Analysis

Custom Query

Rankings

Analysis

Add a New Video

Video general details

Video Name*

Video Type

Collection

Video URL *

Insert URL from AT&T source

Video resolution details

Video original width *

Video original height *

Output Resolution Width *

Output Resolution Height *

Width only

Height only

Video settings details

Codec *

Color Space *

Video bitrate *

Video framerate *

SDR = bt.709, HDR = bt.2020

Bitrate should be in kilobytes

Precise the videos framerate (fps)

Video Tags

Tags *

Insert tags separated by commas

SEND

XIII.

AT&T Admin Tools for Video Quality Assurance

Info

Settings

Management

Collections

Add Collection

Add Video

Search Tags

Delete

Data Analysis

Custom Query

Rankings

Analysis

Tag Searcher

Tag Name

Name of Tag to Search *

Name

Original Resolution

Output Resolution

Framerate

Bitrate

Tags

SEARCH TAGGED VIDEOS

XIV.

AT&T Admin Tools for Video Quality Assurance

Info

Settings

Management

Collections

Add Collection

Add Video

Search Tags

Delete

Data Analysis

Custom Query

Rankings

Analysis

Deletion Tool

Delete a Collection

Choose a collection...

DELETE COLLECTION

DELETE COLLECTION, VIDEOS, AND SCORES

Delete a Video

Choose a collection...

Choose a video...

DELETE VIDEO

DELETE VIDEO AND SCORES

XV.

AT&T Admin Tools for Video Quality Assurance

Info

Settings

Management

Collections

Add Collection

Add Video

Search Tags

Delete

Data Analysis

Custom Query

Rankings

Analysis

Custom Query

Query *

SELECT * FROM videos

BUN

VIEW DATABASE

DOWNLOAD

Shareable URL: http://localhost:3000/customQuery/query-SELECT%20*%20FROM%20Videos

id	bitrate	codec	color_space	content	eof	frame_rate	original_resol...	original_resol...	output_resol...	output_resol...	url	name	tags
1	5300	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	720	1280	http://cdntes...	UHD_2016W...	test.football...
2	8000	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	1080	1920	http://cdntes...	UHD_2016W...	
3	3500	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	720	1280	http://cdntes...	UHD_2016W...	
4	2200	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	720	1280	http://cdntes...	UHD_2016W...	
5	850	HEVC	bt.709	2016WSOBV	SDR	29	1080	1920	720	1280	http://cdntes...	UHD_2016W...	
6	3500	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	1080	1920	http://cdntes...	UHD_2016W...	
7	5300	HEVC	bt.709	2016WSOBV	SDR	60	1080	1920	1080	1920	http://cdntes...	UHD_2016W...	
8	850	HEVC	bt.709	2016WSOBV	SDR	29	1080	1920	360	640	http://cdntes...	UHD_2016W...	
9	1300	HEVC	bt.709	2016WSOBV	SDR	29	1080	1920	540	960	http://cdntes...	UHD_2016W...	
10	1300	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	720	1280	http://cdntes...	UHD_ABlue...	
11	2000	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	720	1280	http://cdntes...	UHD_ABlue...	
12	3000	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	720	1280	http://cdntes...	UHD_ABlue...	
13	4500	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	720	1280	http://cdntes...	UHD_ABlue...	
14	850	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	720	1280	http://cdntes...	UHD_ABlue...	
15	3000	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	1080	1920	http://cdntes...	UHD_ABlue...	
16	4500	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	1080	1920	http://cdntes...	UHD_ABlue...	
17	7000	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	1080	1920	http://cdntes...	UHD_ABlue...	
18	1300	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	540	960	http://cdntes...	UHD_ABlue...	
19	850	AVC	bt.709	ABlueMoon	SDR	24	1080	1920	360	640	http://cdntes...	UHD_ABlue...	
20	1300	HEVC	bt.709	AdinserLoop1	SDR	29	1080	1920	720	1280	http://cdntes...	UHD_Adinser...	

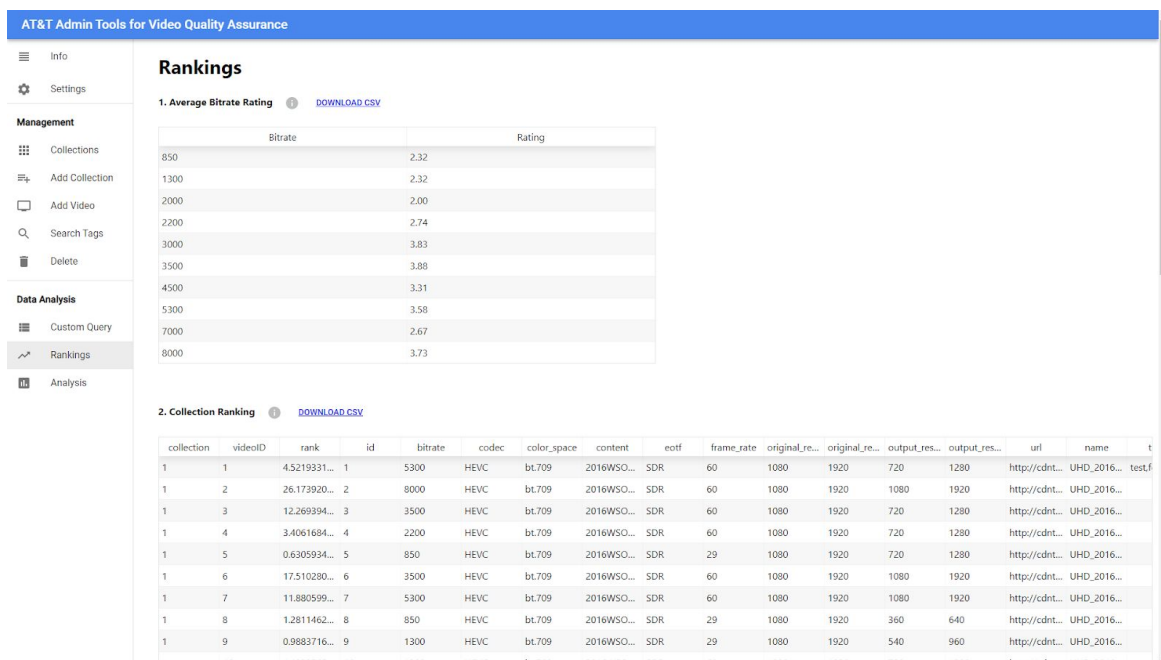
Previous

Page 1 of 19

20 rows

Next

XVI.



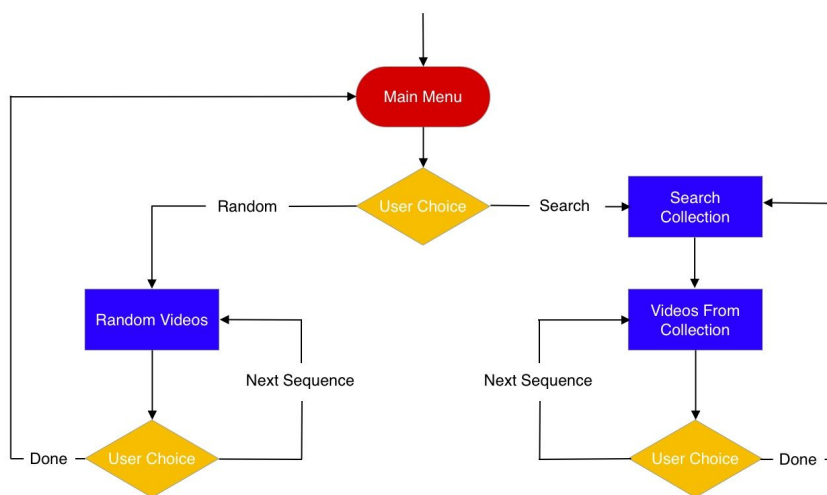
XVII.



- IX. Info Page
- X. Settings Page
- XI. Collections
- XII. Add Collection
- XIII. Add Video
- XIV. Search Tags
- XV. Delete
- XVI. Custom Query
- XVII. Rankings
- XVIII. Analysis

The user will be informed about what everything does on the user interface of the web application by the initial info page containing a list explaining how each component functions. An application bar will be displayed at the top of the window with the name of the application. There will be a sidebar on the screen that will direct users to pages that are listed on it when the user clicks on the name of the page.

7.2 User Interface Flow Model for Android and iOS



8. Requirements Status

Below is a list of our planned requirements for the system. Bold text indicates that the requirement was completed. Red test indicates that the requirement was not completed.

1. Mobile Application
 - a. Login
 - i. **The app shall accept a username for login**
 - ii. **The login shall not be secured and shall not require a password**
 - iii. **The app shall allow the user to continue as a guest**
 - iv. **The app shall call the API to get the user's information upon login**
 - b. Rating Choice
 - i. **The app shall allow the user to choose between watching a random video and browsing a list of available videos**
 - ii. **The app shall allow the user to logout**
 - c. Random Video Flow
 - i. **The app shall query the API to get the random collection and the two random videos from that collection**
 - ii. **The app shall show the videos in the following order:**
 1. **The app shall show the user instructions for rating the videos**
 2. **The app shall show the first randomly selected video**
 3. **The app shall prompt the user to watch the second video**
 4. **The app shall show the second randomly selected video**
 - d. Browse Video Flow
 - i. **The app shall query the API to get all collections and display them for the user to select**
 - ii. **After selecting a collection, the app shall query the API to get two random videos from that collection**
 - iii. **The app shall show the videos in the following order:**
 1. **The app shall show the user instructions for rating the videos**
 2. **The app shall show the first randomly selected video**
 3. **The app shall prompt the user to watch the second video**
 4. **The app shall show the second randomly selected video**
 - e. Video Rating
 - i. **The app shall show a rating screen which allows the user to select which video was better with the options Definitely Video A, Maybe Video A, About the Same, Maybe Video B, and Definitely Video B. The user must select a rating to proceed**
 - ii. **The app shall allow the user to rewatch the videos**

- iii. The app shall send the user's chosen rating to the API. It must include the username, videos rated, device type, if mirrored, screen brightness, and rewatch count

f. Video Results

- i. The app must query the API to check the show_results setting.
- ii. If the show_results setting is ON, the app shall show the current rating data for that video in a chart
- iii. if the show_results setting is OFF, the app shall only display a "Thank you for rating" message
- iv. The user shall be able to select "Next Video" to watch another random video
- v. The user shall be able to logout from the application
- vi. The user shall be able to view the video's technical details by clicking an "info" button

2. Absolute Rating Mobile Application

- a. The app shall have all the same login requirements as #1.a above
- b. The app shall have all the same rating choice requirements as #1.b above
- c. The app shall show a tutorial before rating videos with the following user flow:
 - i. The user will be shown a first video and instructed that this should be considered "Excellent"
 - ii. The user will be shown a second video and instructed that this should be considered "Good"
 - iii. The user will be shown a final video and instructed that this should be considered "Poor"

d. Random Video Flow

- i. The app shall query the API to get the random collection a single random video from that collection
- ii. The app shall show the videos in the following order:
 - 1. The app shall show the user instructions for rating the video
 - 2. The app shall show the randomly selected video

e. Browse Video Flow

- i. The app shall query the API to get all collections and display them for the user to select
- ii. After selecting a collection, the app shall query the API to get a single random video from that collection
- iii. The app shall show the videos in the following order:
 - 1. The app shall show the user instructions for rating the videos
 - 2. The app shall show the randomly selected video

f. Video Rating

- i. The app shall show a rating screen which allows the user to select the video rating with the options Excellent, Very Good, Good, Fair, and Poor. The user must select a rating to proceed
 - ii. The app shall allow the user to rewatch the video
 - iii. The app shall send the user's chosen rating to the API. It must include the username, video rated, device type, if mirrored, screen brightness, and rewatch count
 - g. The app shall have all the same video rating requirements as #1.f above
- 3. Web Application
 - a. Video Management
 - i. The user shall be able to view all collections
 - ii. The user shall be able to edit all collections
 - 1. The user shall be able to hide a specific collection
 - iii. The user shall be able to delete a collection
 - iv. The user shall be able to delete a collection along with all associated videos and scores
 - v. The user shall be prompted to confirm before making any deletions
 - 1. It was determined that only delete actions required confirmation.
 - vi. The user shall be able to view all videos for each collection
 - vii. The user shall be able to edit all videos
 - viii. The user shall be able to delete a video
 - ix. The user shall be able to delete a video along with all associated scores
 - b. Settings
 - i. The user shall be able to view the system settings
 - ii. The user shall be able to toggle the system settings
 - c. Data Analysis
 - i. The user shall be able to run a custom query on the database and return the result
 - ii. The user shall be able to view rankings of videos for each question
 - iii. The user shall be able to view graphs which display data to prove/disprove the following hypotheses:
 - 1. Higher framerate is preferable to higher resolution if bitrate is constant.
 - 2. Bits per pixel ($\text{bitrate} / \text{height} * \text{width} * \text{framerate}$) is a good way to predict user preference.
 - 3. Users can't tell the difference between low bitrate (< 1Mbps AVC) variants.
 - 4. Users can't tell the different between high bitrate (> 6Mbps AVC) variants.

5. HEVC doesn't provide noticeably better quality than AVC at low bitrates (< 1Mbps)
6. Frame rate is still preferable to resolution on HEVC encoded content.
7. The HLS recommended 1.5x bitrate ABR ladder always provides noticeable VQ improvement between variants.

4. API

- a. The user shall be able to send a query to get data from the database:
 - i. **Send a score**
 - ii. **Get all scores**
 - iii. **Get a list of all collections**
 - iv. **Get a single random collection**
 - v. **Get two random videos from the collection**
 - vi. **Get collection details from collection ID**
 - vii. **Get video details from video ID**
 - viii. **Login a user and get user data**
 - ix. **Add points to a user**
 - x. **Add a collection**
 - xi. **Edit a collection**
 - xii. **Delete a collection**
 - xiii. **Add a video**
 - xiv. **Edit a video**
 - xv. **Delete a video**
 - xvi. **Get config/settings**
 - xvii. **Update config/settings**
 - xviii. **Run a custom query which has been sanitized only to allow SELECT***
 - xix. **Get video rankings for each collection**
 - xx. **Get average score per AVC video**
 - xxi. **Compare ratings for HEVC and AVC codecs**
 - xxii. **Average scores per bitrate**
 - xxiii. **Average scores per bit-per-pixel ratio**
 - xxiv. **Average scores per Framerate/Resolution pair**
 - xxv. **Average scores per Framerate/Resolution pair for HEVC videos only**

5. Database

- a. **The database shall be relational and use MySQL**
- b. **The database shall store the following information:**
 - i. **Users**
 1. **Table for Rating App**
 2. **Table for Absolute Rating App**

- ii. **Video ratings**
 - 1. **Table for Rating App**
 - 2. **Table for Absolute Rating App**
- iii. **Videos**
- iv. **Collections**
- v. **Absolute Rating tutorial videos**

9. Glossary

SRS	Software Requirement Specification
SDS	Software Design Specification
js	Java Script
Android OS	A mobile operating system developed by Google
iOS	A mobile operating system developed by Apple
MacOS	Macintosh Operating System
JDK	Java Development Kit
App	Application
OS	Operating System
GUI	Graphic User Interface
MMI	Main Menu Interface
API	Application Program Interface
HEVC	High Efficiency Video Coding
SQL	Structured Query Language
HTTP	Hyper-Text Transfer Protocol
AVC	Advanced Video Coding
MB	Megabyte
GB	Gigabyte
RAM	Random Access Memory
NPM	Node Package Manager
URL	Uniform Resource Locator