COMPUTER SCIENCE & ENGINEERING
TEXAS A&M UNIVERSITY

Project Details

Iteration 2 (21 Oct 2022)

Purpose: Started with developing and integrating the features of the User Story selected for this interaction.

Scrum Master: Xintong Wu

Project Owner: Jesse Phipps

General Members: Pedro Henrique Villar De Figueiredo, Ryan Kafka, & Siddhant Thakur

GitHub: https://github.com/jessefphipps/fighting-aggies-platform

Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2598148

Heroku: https://fightin-aggies.herokuapp.com/

Logistics

Weekly Scrum meetings will occur directly after class on Tues/Thurs, as well as before the submission of any iteration documentation. Additional meetings will be set if needed.

Customer meeting date

- 28th October, 2022 [Friday] - 1:45PM (CST)

User Stories addressed

- Feature - User can log in and out of the application
    - As an A&M Athletics Member
    - I want to setup a login session manager
    - So that I can maintain the privacy of the data uploaded
- Feature - User can upload video files
    - As an A&M Sports Analyst
    - I want to upload video files of football practices / games
    - So that I can get a performance report of my players

Running the app locally on your terminal

Prerequisites

To install ruby (3.1.2) on your machine,

`winget install RubyInstallerTeam.Ruby // for Windows`

or

`sudo apt-get install ruby-full // for Ubuntu`

or

`brew install ruby // for MacOS`

For the video uploading feature, we need ffmpeg CLI to be present on the local machine,

For Windows,

1. Go to the link.
2. Extract the file to where you want it to be and rename it as ffmpeg.
3. Run the Command Prompt as admin and write `setx /m PATH "path\to\ffmpeg\bin;%PATH%"`

For Ubuntu,

`sudo apt install ffmpeg // for Ubuntu`

Running the app

```
git clone https://github.com/jessefphipps/fighting-aggies-platform.git
cd fighting-aggies-platform
bundle install
gem install cucumber
yarn install
rails server
```

Please ensure the ruby version present in your system is the same as the one mentioned in the Gemfile. Just change the Gemfile to your version of ruby if they are not the same.

Running the app on your browser

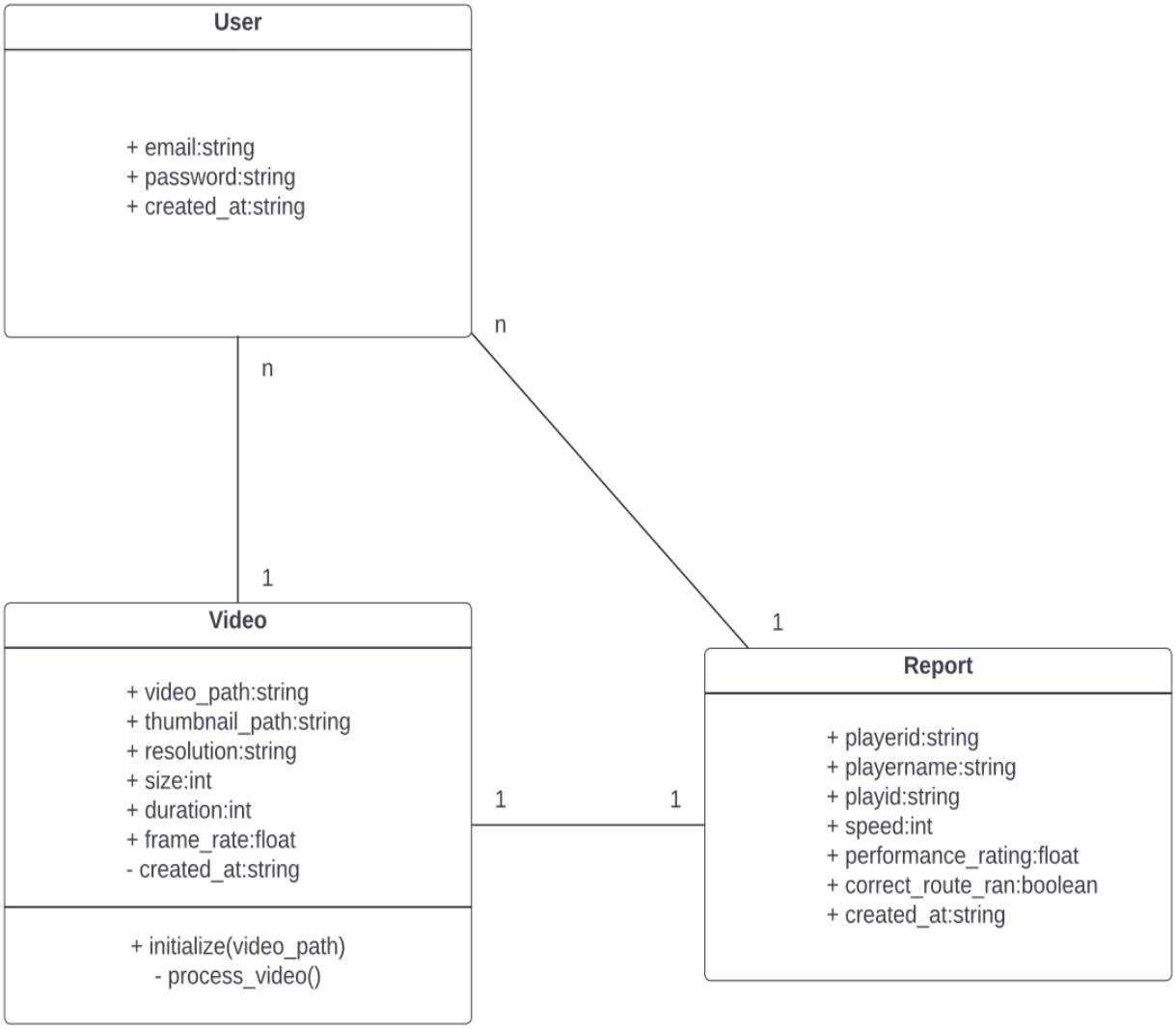
Just follow the link for the Heroku App! Enter the login info to gain access to the dashboard.

Login Info

Email - admin@example.com

Password - 123456

Design Diagram



Work done in Iteration 2

After implementing the user stories locally, we were facing a few dependency issues while trying to deploy the app on other machines and on heroku. So during this iteration, we focused on setting up a common environment to work through these dependency issues, and deploy the user stories (Login and Uploading features) on Heroku. Thereby, completing 2 User Stories in 2 Iterations.

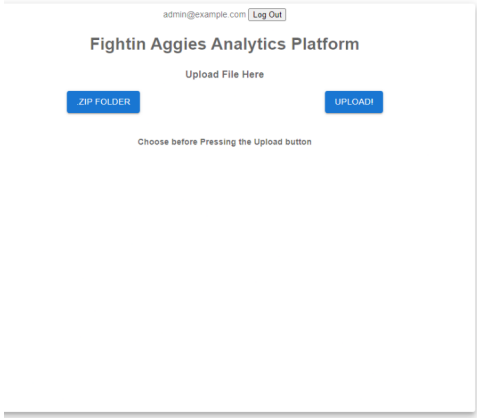
Extension to User Story 1

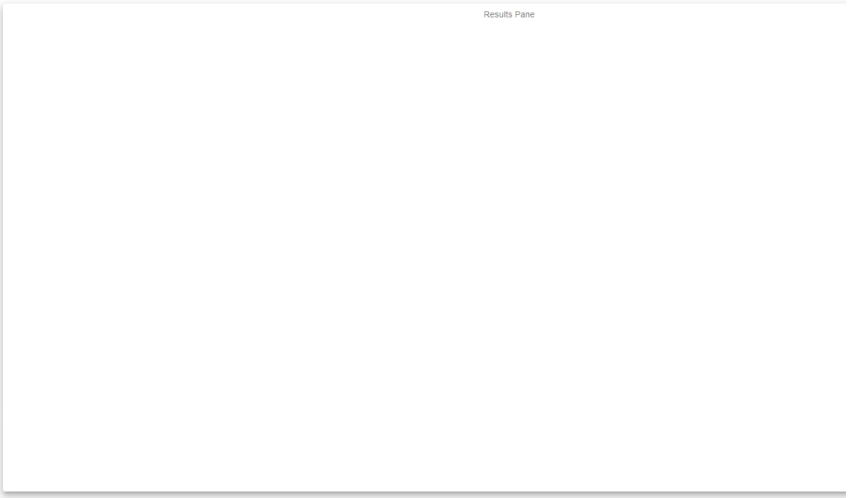
The Login user story was successfully extended to implement Session Management, allowing the user to log in & out of the application. It also allows the application to keep track of who's logged in to the server.

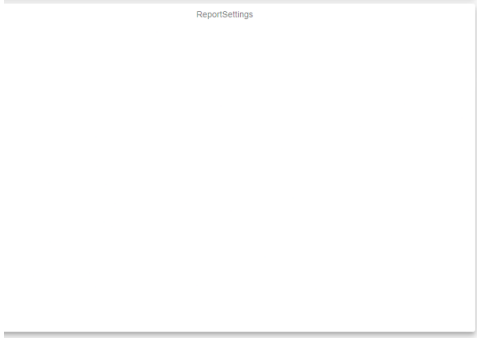
Lo-Fi UI Update

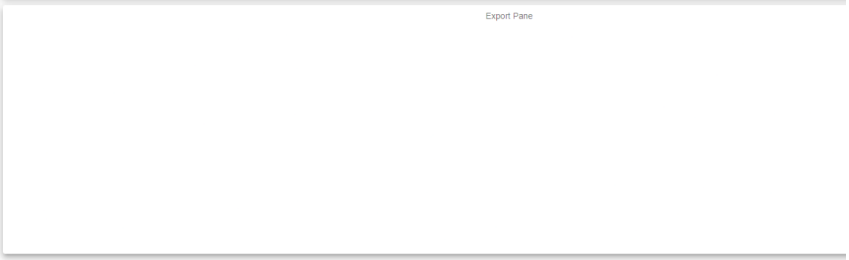
Log In Page

Email Email Address Password Password Log In









Evaluation of code and tests

We have used Cucumber and Minitests to test our front-end and back-end development. Cucumber was used for UI tests on both login and dashboard pages. It was also used for end-to-end testing on the dashboard (front-end + back-end, to see how the user will interact with the system). Minitest was specifically used for the backend Video testing.

Tests for UI Login: Cucumber

- **[bad path 1]** whether the user has provides invalid password → wrong password
- **[bad path 2]** whether the user provides invalid email address → user not found
- **[bad path 3]** whether the user provides a bad email format → improper email format
- **[correct path 1]** when the user clicks the logout button → logged out from session & redirected to login page
- **[correct path 2]** whether the user enters correct email and password combination → redirected to dashboard

Tests for End-to-End File Upload (UI + backend): Cucumber

- **[bad path 1]** when the user logs in to the dashboard → upload button is disabled by default
- **[bad path 2]** when the user uploads a file with incompatible file type → user gets negative feedback: incompatible upload type
- **[bad path 3]** when the user uploads a file which is corrupt → user gets negative feedback: incompatible upload file
- **[correct path 1]** when the user uploads file successfully → user gets positive feedback: file uploaded successfully

Tests for Video Model: Minitests

- **[bad path 1]** when there's no video uploaded → Uploaded file is not a video
- **[bad path 2]** when the format for the video is incorrect → Uploaded file is not a video
- **[bad path 3]** when the video file path is not present → Video file does not exist
- **[bad path 4]** when the video file is corrupted→ Cannot read the video file
- **[correct path 1]** when the video file path is correct for existing and valid sample video
    - **[bad path 1]** when the file does not exist → Video file does not exist
    - **[bad path 2]** when the recorded video path does not match the uploaded video path → Recorded video path is not the same as in db
    - **[bad path 3]** when the thumbnail of the recorded video does not match the thumbnail of the uploaded video → Recorded thumbnail path is not the same as in db
    - **[bad path 4]** when there is no thumbnail generated of the previous upload → Generated thumbnail file does not exist
    - **[bad path 5]** where there is a resolution problem with the recorded video → Recorded resolution does not match sample video
    - **[bad path 6]** where there is a duration problem with the recorded video → Recorded duration does not match sample video

- **[bad path 7]** where there is a frame problem with the recorded video → Recorded frame does not match sample video
- **[bad path 8]** where there is a frame rate problem with the recorded video → Recorded frame rate does not match sample video

Tests for Video Controller: Minitests

- index
  - [correct path 1] controller is able to fetch index → 200, Success
  - [bad path 1] checks if it returns all videos present in DB → Videos coming from index request don't match db's videos
- show
  - [bad path 1] if no id present in get request → 400, No ID was provided
  - [bad path 2] if id is invalid (negative or not in DB) → 400, Video with requested ID not found in DB
  - [bad path 3] if id is valid, but it doesn't match the query → Requested video does not match with query ID
  - [correct path 1] if id is valid → 200, Success
- create
  - [bad path 1] uploading video with no data → 400, No video provided
  - [bad path 2] uploading video with incorrect file format → 400
  - [bad path 3] uploading video with corrupted data → 400
  - [correct path 1] uploading video with everything correct → 201, video json details
- destroy
  - [bad path 1] deleting video with no id → 400, No ID was provided
  - [bad path 2] deleting video with invalid id (negative or not in DB) → 400,Video with requested ID not found in DB
  - [bad path 3] deleting a video with valid id, but it doesn't match the query → Video was not deleted from DB
  - [correct path 1] deleting a video with valid id → 200, Success

Tests for running the app on the browser (Environment testing)

- Environment should be driven by a Selenium webdriver and use Chrome.