Project Details

Iteration 5 (2 Dec 2022)

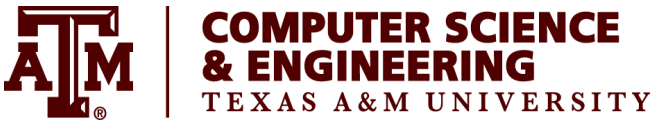
Purpose:	Started with developing and integrating the features of the User Story selected for this interaction.
Scrum Master:	Jesse Phipps
Project Owner:	Siddhant Thakur
General Members:	Ryan Kafka, Xintong Wu, & Pedro Henrique Villar De Figueiredo
GitHub:	https://github.com/jessefphipps/fighting-aggies-platform
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

- ??? ??????????



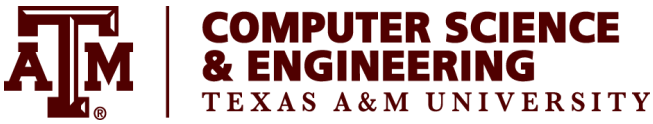
User Stories addressed

- Feature - User can see summarized findings compiled from hundreds of plays
    - As a Texas A&M Athletics Data Analyst
    - I want to compile performance results across many plays
    - So that I can view summarized data describing the effectiveness of plays, players, and the like

- Feature - User can sort relevant data
    - As a Texas A&M Athletics Data Analyst
    - I want to be able to sort the data I am viewing
    - So that I can easily view the performance information that is the most informative

- Feature - User can view color coded data
    - As a Texas A&M Data Analyst
    - I want my data to be color coded to indicate when there is a particularly low performance rating
    - So that I can quickly detect weaknesses in the respective category

- Feature - User can export results as an excel file containing the summarized data displayed on the dashboard
    - As a Texas A&M Athletics Data Analyst
    - I want to be able to export the performance information into a formatted excel file
    - So that I can easily view, edit, and share the data outside of the web application



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

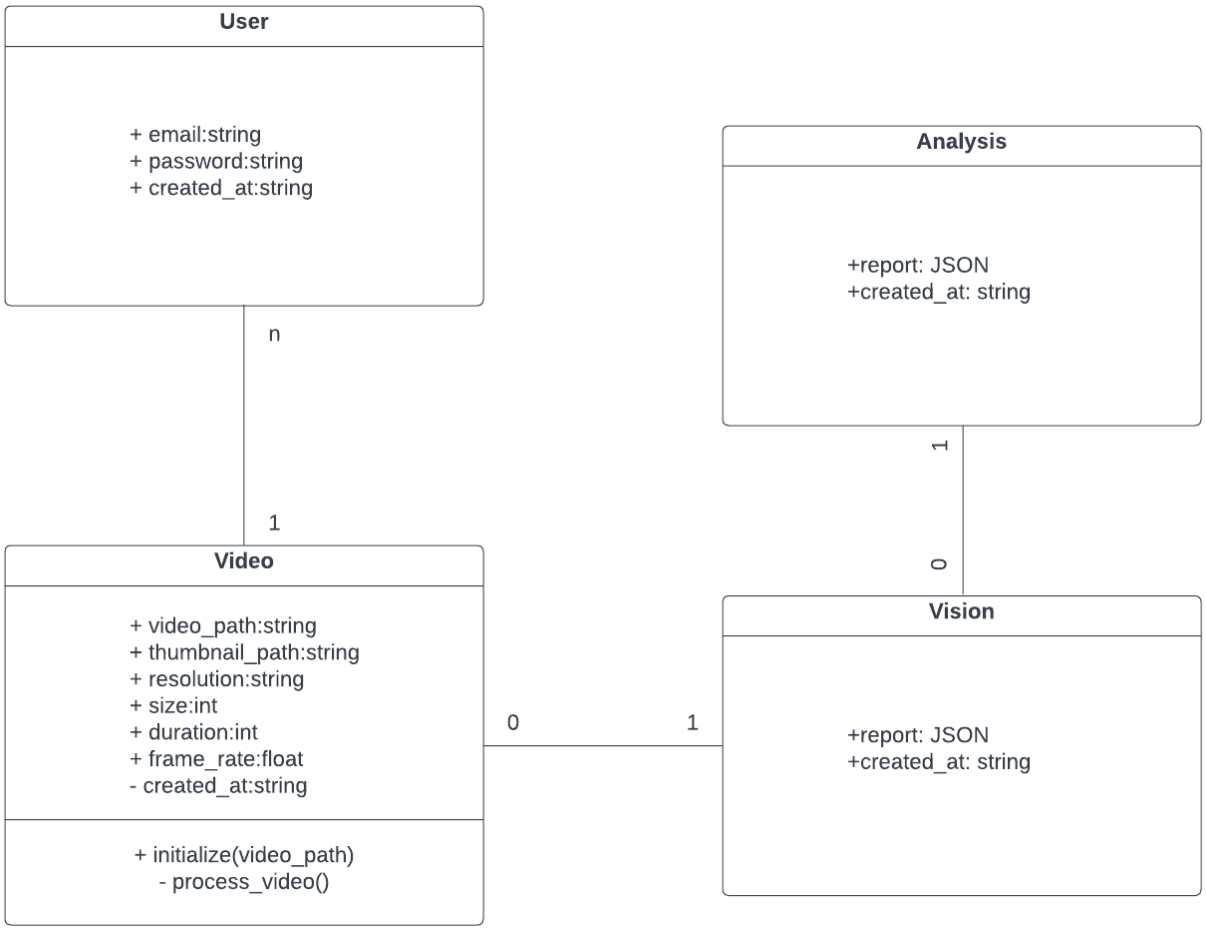
Just follow the link for the Heroku App! Enter the login info to gain access to the dashboard.

Login Info

Email - admin@example.com

Password - 123456

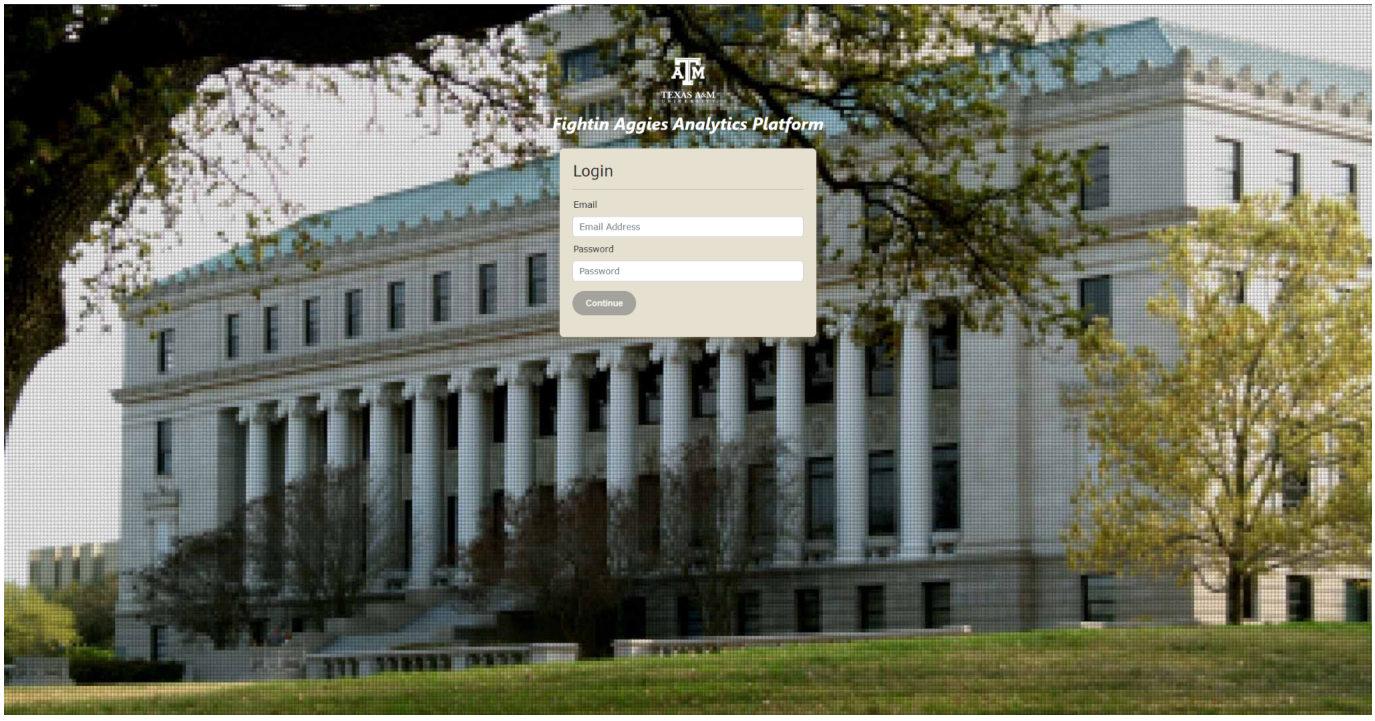
Design Diagram

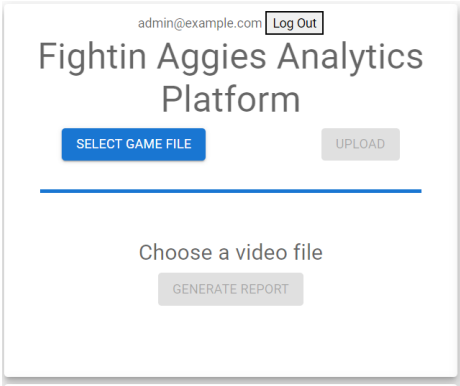


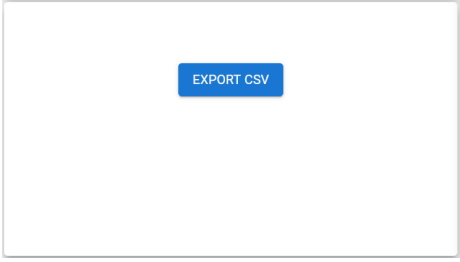
Work done in Iteration 5

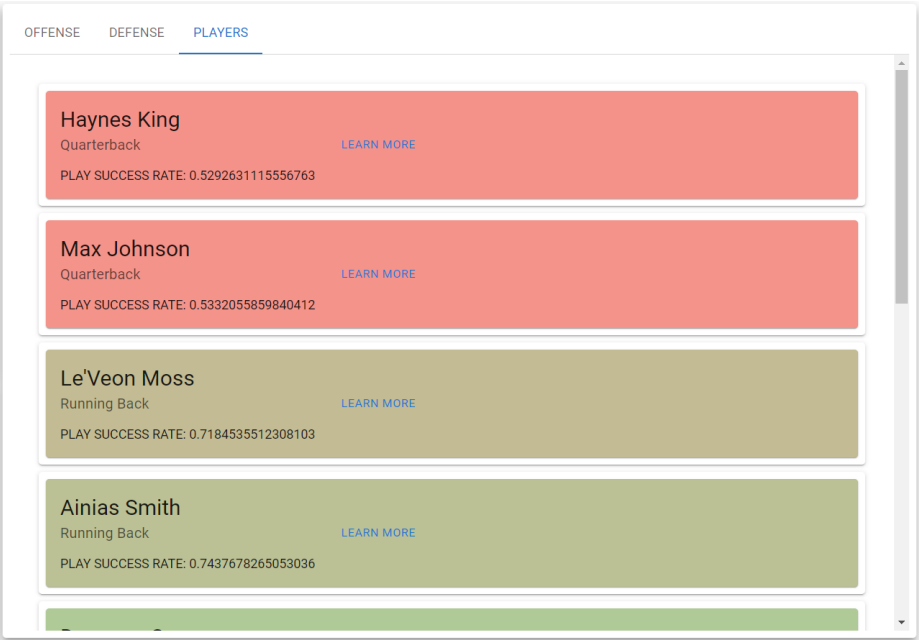
In this iteration, we make sure that we allow the coaches and the players of the Aggie football team to be able to view their performance as well as export the data generated from the vision model to create their own visualizations for better understanding. We also add our own features in the visualization dashboard by enabling sorting of players and color coding their tabs based on that player's performance allowing the coaches and the staff to make better decisions in-game. Lastly, we improved the layout of our application to be more dynamic. This allows for elements/panes to better react to changes in both the UI and screen size.

Lo-Fi UI Update









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 report generation (UI): Cucumber

- **[generate button disabled 1]** when the user first accesses the dashboard, the report generation button should be disabled
- **[generate button disabled 2]** when the user has selected a file but not successfully completed an upload, the button should be disabled
- **[success]** when the user successfully uploads a file and all other tests pass, the report is generated and the results pane is populated

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 Vision Controller: Minitests

- index
  - **[correct path 1]** controller is able to fetch index → 200, Success
  - **[bad path 1]** checks if it returns all raw vision data present in DB → Raw vision data coming from index request don't match db's raw vision data

- 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, Raw vision data with requested ID not found in DB
    - **[bad path 3]** if id is valid, but it doesn't match the query → Requested raw vision data does not match with query ID
    - **[correct path 1]** if id is valid → 200, Success
- create
    - **[bad path 1]** generating raw vision data with no data → 400, No raw data received
    - **[bad path 2]** generating raw vision data with video id not present in the database → 400
    - **[bad path 3]** generating two raw vision data for same video id → 400
    - **[correct path 1]** generating raw vision data with correct video id→ 201, raw vision data json details
- destroy
    - **[bad path 1]** deleting raw vision data with no id → 400, No ID was provided
    - **[bad path 2]** deleting raw vision data with invalid id (negative or not in DB) → 400, Raw Vision data with requested ID not found in DB
    - **[correct path 1]** deleting a video with valid id also destroys raw vision data→ 200, Success

Tests for Analysis Controller: Minitests

- index
    - **[correct path 1]** controller is able to fetch index → 200, Success
    - **[bad path 1]** checks if it returns all analyses present in DB → Analyses coming from index request don't match db's analyses
- 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, Analysis with requested ID not found in DB
    - **[bad path 3]** if id is valid, but it doesn't match the query → Requested analysis does not match with query ID
    - **[correct path 1]** if id is valid → 200, Success
- create
    - **[bad path 1]** generating analysis with no data → 400, No analysis done
    - **[bad path 2]** generating analysis with video id not present in the database → 400
    - **[bad path 3]** generating two analyses for same video id → 400
    - **[bad path 4]** generating analysis with correct video id but null raw vision data → 400
    - **[correct path 1]** generating analysis with correct video id and correct raw vision data→ 201, analysis json details

- destroy
    - **[bad path 1]** deleting analysis with no id → 400, No ID was provided
    - **[bad path 2]** deleting analysis with invalid id (negative or not in DB) → 400, Analysis with requested ID not found in DB
    - **[correct path 1]** deleting a video with valid id also destroys analysis → 200, Success

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

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