***ImageMetrics***

**End goal:**

A tool to upload, analyze, and manipulate image exif data. This tool should be able to handle both JPG and JPEG type formats, and produce exif data in similar ways. A lack of exif data should be handled gracefully, and users should be informed if such a situation exists. This tool should allow the user to easily upload and analyze data about images, and time permitting, more features will be added.

**Specific Requirements:**

* Obtaining exif data
  + User uploads own image
  + Automatically pull data from image
  + Display metadata
  + Handling bad files appropriately
* Data manipulation
  + Search through data for specific entries
    - Camera type
    - Location
    - Date
    - Name
    - Etc
  + Editing data \*
  + Display popular/common data of bulk files \*
  + Export data in a PDF or similar file
  + Display location where photo was taken on a map \*
  + Sort photos by a metric (such as date taken) \*
    - Handle creation of new files under the desired metric
* GUI
  + Navigation to upload files
  + Display pertinent data, if available
  + Data visualization(bar charts, histogram, etc) \*
  + Saving edited data to the file \*

\* Items annotated with an asterisk indicate “wishlist” items and will be added as time permits.

**User Requirements:**

* Any computer with a browser
  + FireFox
  + Chrome
  + Edge
  + Safari

**Internal Tools/Language Requirements**

* Python3
  + Pillow Library - Manipulating exif data
  + Flask - GUI
  + NumPy
  + Matplotlib
* HTML / CSS
* Git/Github
* Jira

**Scenarios**

***Scenario One: Base Application Requirement***

* A user uploads an image from their machine
* The application takes the image and analyzes it for exif data
* The data is presented to the user
  + If data exists, display it in a clean, table-like manner
  + If no data exists (or a bad file is given), inform the user that data could not be retrieved in a helpful manner.

***Scenario Two: Covid Test Photo***

* An employee sends their boss photos of a positive covid test.
  + Boss is skeptical
* The boss uploads said photos
* Using the data, they can determine if the photos were actually taken by the employee (consider date).
* The boss may be able to perform a reverse image search using the data provided

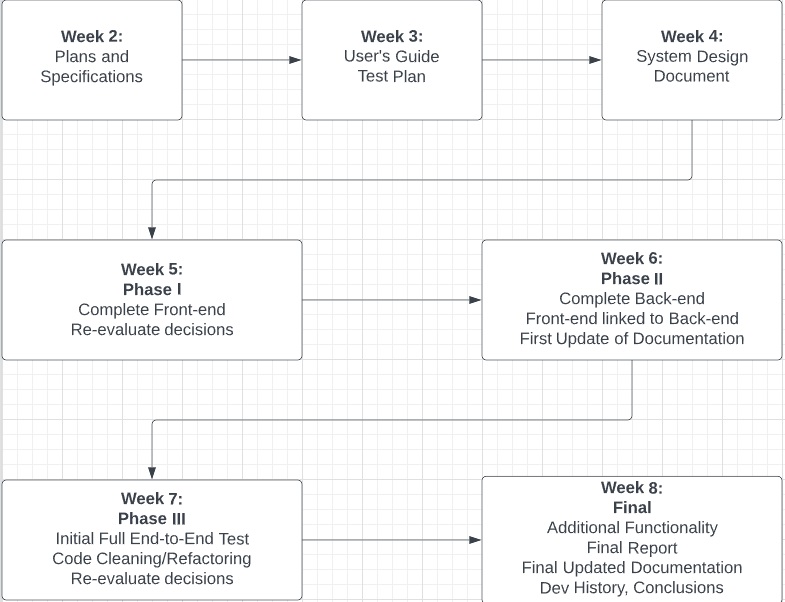
***Scenario Three: Photo ID/Driver License***

* Someone sends a user an image of their ID for -reasons-.
  + This photo, for some reason, is required to be scanned from a specific machine
* The user uploads the photo of the ID
* The user can determine if the photo was acquired from a phone image, scanner, etc.
* The user can determine if the photo was taken before/after a specific date as well

***Scenario Four: Analyzing Bulk Data - Wishlist features***

* A user uploads bulk images and would like to know the most popular aperture settings for similar photos
* The user would like to see a bar chart for this analyzation
* The application takes the files, and analyzes the data of each to find commonalities
* The chart is displayed to the user annotating the metrics that are shared between multiple files

**Schedule and Milestones**

****