Cloud Computing

UI Design Document

Course: AD440 Winter Quarter 2016

Group members:

1- Kianoush Moradian

2- Ksenia Ivantsova

3- Avi Herbstman

Professor: Dr. Paul Z. Wu

Company name: Agitare Technologies - Toddy Mladenov

March 22, 2016

Contents

[Technologies: 3](#_Toc446418826)

[Development Mythology 3](#_Toc446418827)

[General Scenario (Abstract): 3](#_Toc446418828)

[Target Audience: 4](#_Toc446418829)

[App Architecture: 4](#_Toc446418830)

[Modular Angular File Structure 5](#_Toc446418831)

[CSS file structure: 6](#_Toc446418832)

[Font: 6](#_Toc446418833)

[JS Folder: 7](#_Toc446418834)

[Testing 8](#_Toc446418835)

[Twitter Login Page 9](#_Toc446418836)

[Display Image 9](#_Toc446418837)

[Upload Image 10](#_Toc446418838)

[Update Tags 10](#_Toc446418839)

[Steps to Upload an Image 10](#_Toc446418840)

[Clear Agile Example: 11](#_Toc446418841)

[Functional Specifications for Get, Post, Delete, Put Directives: 11](#_Toc446418842)

[Implementation and Coding: 13](#_Toc446418843)

# Technologies:

1. AngularJS

2. JQuery / AJAX

3. Google API AJAX

4. Bootstrap

5. Angular Factory

6. HTML, CSS, JavaScript

7. Twitter API, OAuth

# Development Mythology

We chose agile software development to develop our project because it is a methodology for the creative process that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product. Agile software development focuses on keeping code simple, testing often, and delivering functional bits of the application as soon as they're ready. The goal of ASD is to build upon small client-approved parts as the project progresses, as opposed to delivering one large application at the end of the project.

# General Scenario (Abstract):

We created the UI for an Application to dynamically interact with database to allow user to interact with the image list in the following ways:

• Login with Twitter Account

• Ability to upload images with tags from local machine

• Display images in a list with tags associated with each image

• Image displayed in response to selection

• Ability to edit/add tags to existing images

• Search mechanism for images

This is an interactive application which allows users to log in with a twitter account. Once logged in, they are shown a display page with a list of first 20 images in their personal account. User can search for an image with tags for display. User can Upload new images from local directory and add tags to that image. User can also update image tags.

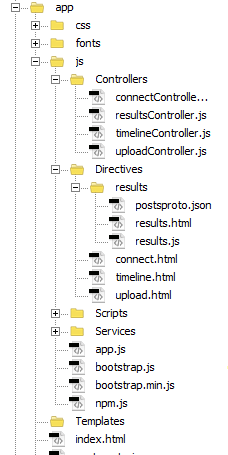
# Target Audience:

The target audience for this app would include personal use for easy filing or in a business setting to keep track of various documents and files in an easily searchable infrastructure.

# App Architecture:

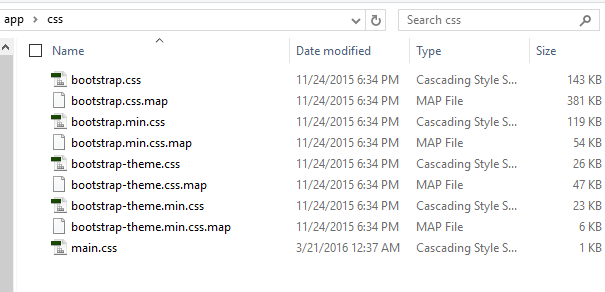
Our image view/upload application would have three main activities branching from the Twitter log in page. Users are then directed to the image display page. User then can go to the upload page where they can select an image to upload and tags. The User can also choose to update an existing image’s tags.

# Modular Angular File Structure



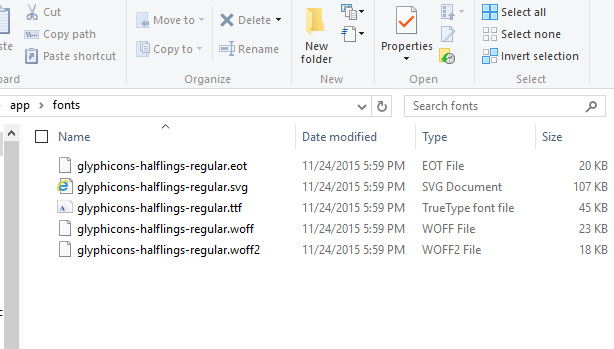
# CSS file structure:

UI design benefits from bootstrap styling and also one customized “main.css” file which has been made by our team.



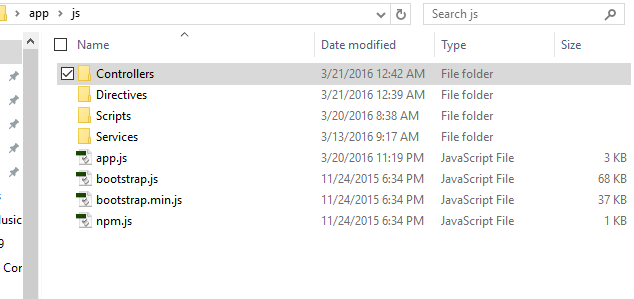
# Font:

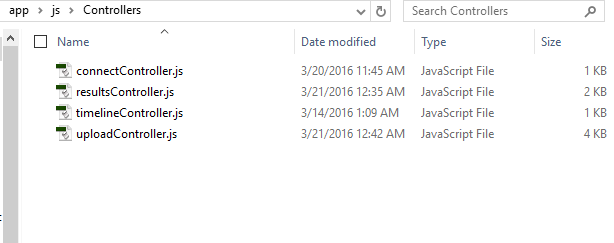
Our preferred font for the UI is “glyphicons-halflings.regular”. We archived all font faces inside Fonts folder.

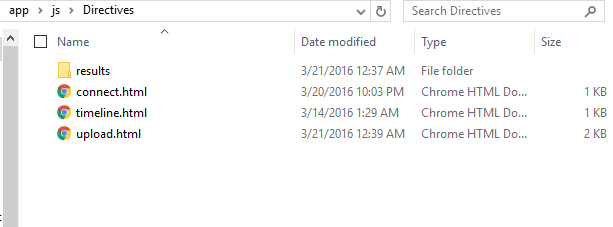


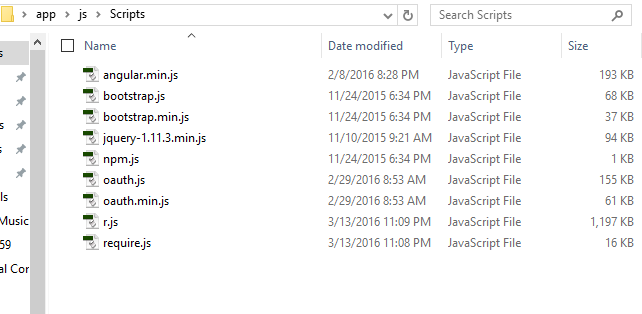
# JS Folder:

JS contains Controllers, Directives, Scripts, and Services. Also we added our app.js and required bootstrap JavaScript file in it.







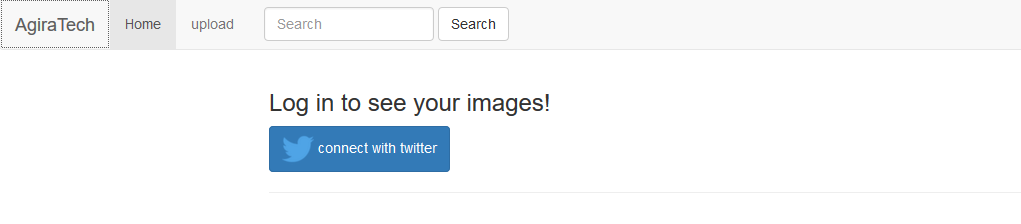


# Testing

We used testing methods to insure that various JSON files are displayed properly. We also created several PHP file handler to see whether our forms work or not.   
We had to engage to some backend coding to make sure things were happening.

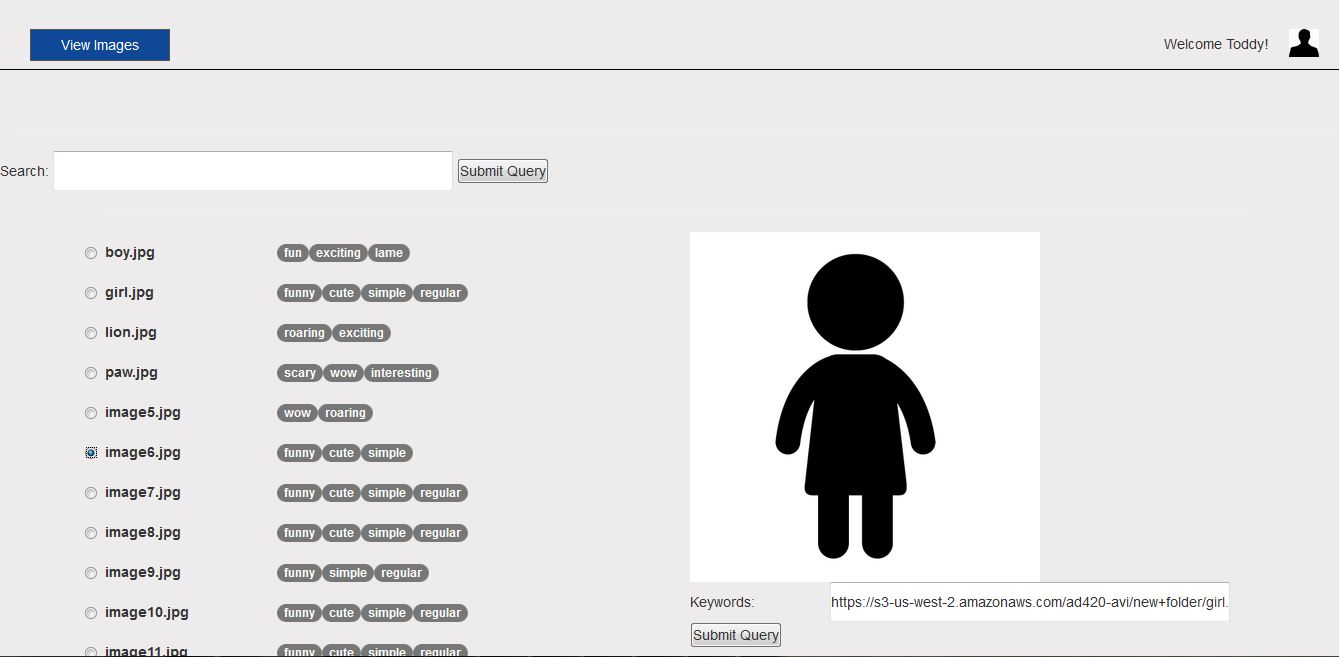
# Twitter Login Page

User is redirected to Twitter’s Login page.



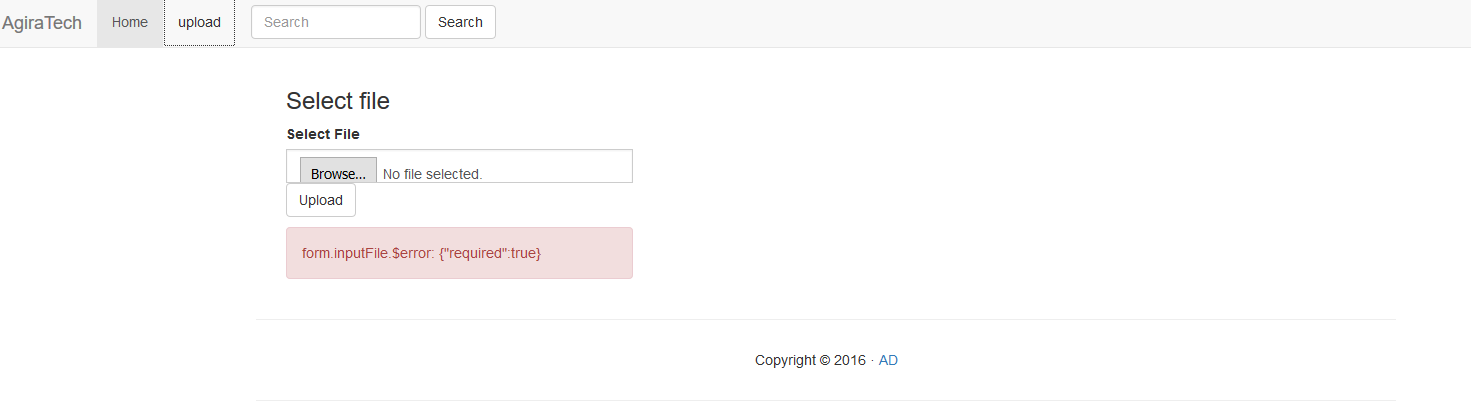
# Display Image

Images List is on left with ability to select Image for display.



# Upload Image

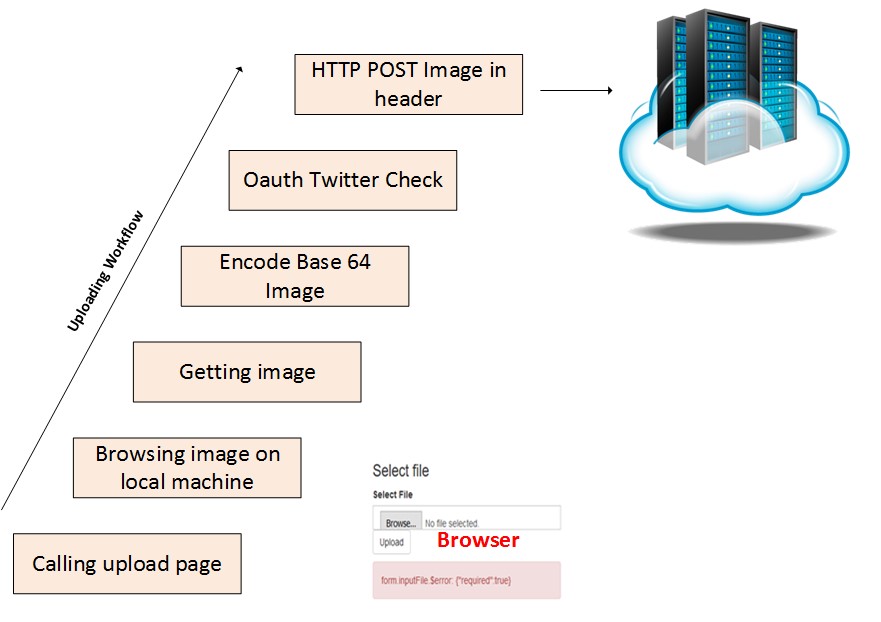
User is able to upload image and add tags.



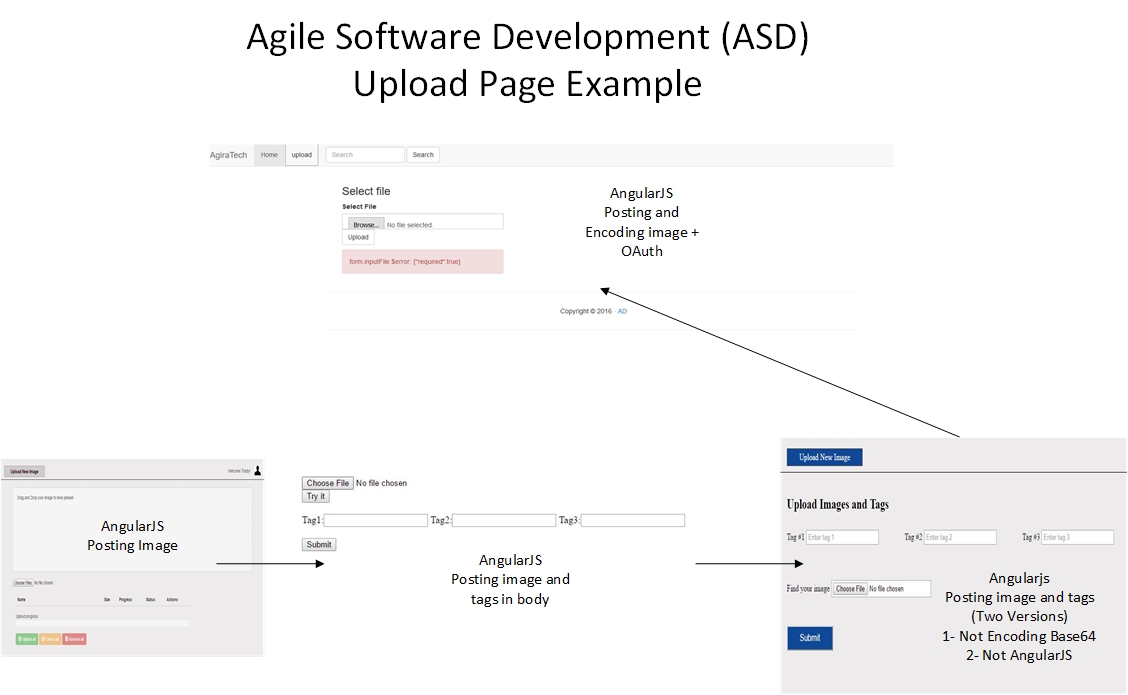
# Update Tags

User is now able to edit/add tags to existing image

# Steps to Upload an Image



# Clear Agile Example:



# Functional Specifications for Get, Post, Delete, Put Directives:

/getImages (Get)

\* timestamp - timestamp (string, ex.2016-02-010T19:47:04.009533)

\* tags [] - list of tags (list[string])

\* username - username from twitter (string)

\* oAuth token - oauth token from twitter (string)

\* oAuth secret - oauth secret from twitter (string)

/uploadImage(Post)

\* tags [] - list of tags (list[string])

\* filename - name of file (string)

\* blob - blob stream (blob)

\* username - username from twitter (string)

\* oAuth token - oauth token from twitter (string)

\* oAuth secret - oauth secret from twitter (string)

/deleteImage (Delete)

\* image url - image url to be deleted (string)

\* username - username from twitter (string)

\* oAuth token - oauth token from twitter (string)

\* oAuth secret - oauth secret from twitter (string)

/updateTags (Put)

\* image url - image url to be deleted (string)

\* tags [] - list of tags (list[string])

\* username - username from twitter (string)

\* oAuth token - oauth token from twitter (string)

\* oAuth secret - oauth secret from twitter (string)

# Implementation and Coding:

Main file: Here is the list of main files use in project

* Index.html
* Results.html
* App.js
* Bootstrap.css
* Upload.html
* Update.html