Stack of books, blackboard and pencils in holderProject 1

**HELLO, WEB!**

# DESCRIPTION

Welcome to your first project for CS490!

In this project, you will set up the main toolchain for the class and use it to build and deploy a simple web app that shows information about your favorite food or recipe and quotes, with content dynamically generated from external services.

# SUBMISSION INFORMATION

**When**This project must be completed (including filling out the Google form) before **September 29th, 2020 at 11:59pm.**

**How**

You can hand in this project by filling out this Google Form:

<https://forms.gle/HCyq7hqmUjSULLF1A>

# Purpose

**Get set up with the tools**

The primary goal of this project is to set you up with a common web development toolchain, and one that we will be using for the first half of the class. In particular, the main tools we will be using include: Flask (in Python), Git and GitHub, Heroku, and AWS Cloud9. Mastering the basics using these technologies will not only be required to complete the rest of the course, it will also enable you to hack on and build more projects of your own.

**Get familiar with external APIs**

The secondary goal is to get you familiar with using external APIs. Specifically, we will be combining the Twitter and Spoonacular REST APIs in this project. A large portion of modern software engineering is building on and reusing others’ work, and in this course, you will of course do the very same!

# WHAT YOU NEED TO DO

**Create an AWS Cloud9 workspace**

To start developing, you’ll first need to log into your AWS Educate Cloud9 IDE. If you have not yet set this up, please do this immediately by following the directions on Canvas.

**Track your work in Git**

This will be needed both for deploying your app as well as submitting your code. We will be doing a crash course in class but check out these tutorials for more: <https://try.github.io/>, <http://learngitbranching.js.org/>.

**Build your web app**

Be creative when building this project! Twitter is a huge corpus of text, and Spoonacular, of recipes. Both their APIs provide a ton of filters; play with them to get a consistent set of quotes and recipes that will match a theme (e.g. “chana masala”, “sweet potatoes”, or “dessert”). Have fun! Facebook employees may see your project, so make something interesting!

**Minimum Viable Product**

**To earn 100% of the technical points on this project, your project must satisfy the following criteria:**

* Is written in Flask (<https://damyanon.net/getting-started-with-flask-on-cloud9/>, <http://flask.pocoo.org/docs/0.12/quickstart/>)
* Fetches quotes dynamically using Twitter’s timeline or search REST APIs (<https://dev.twitter.com/rest/public>, <https://dev.twitter.com/rest/public/search>)
  + Documentation on authentication: <https://dev.twitter.com/oauth/application-only>
  + For documentation on OAuth authentication over HTTP, check out <http://docs.python-requests.org/en/latest/user/authentication/#oauth-1-authentication>.
  + You may also use a wrapper library like <http://www.tweepy.org/>.
* Fetches recipe data dynamically using Spoonacular’s REST API (<https://spoonacular.com/food-api/docs>)
  + Help on getting started: <https://spoonacular.com/food-api/docs#Authentication>. Make sure you follow directions to “authenticate“ via an API key)
* Passes in the correct URL and parameter!
  + You should use the HTTP API python library “requests”.
* Stores and references API keys, secrets, and login credentials using Heroku config vars (<https://devcenter.heroku.com/articles/config-vars>)
  + Make sure they’re not checked into code!!
  + See <http://softwareengineering.stackexchange.com/a/182074> for more information.
* Uses the endpoint “/” (the main one) to serve a page where dynamic quotes and recipe information are shown
  + Quotes and recipe information must be fetched via the above APIs, not stored in advance, and must be random for at least 7 page loads.
  + Quote and recipe information pairings must form consistent theme across 7 different page loads. (See below for details.)
  + All quote and recipe information as described on rubric must appear on the page.
  + Attribution link to recipe (directs to original website (not spoonacular) must appear somewhere on page.
  + Quotes and attribution links must be legible for at least 7 page loads; you may need to adjust things such as font family, font size, text color, text location, and CSS3 filters like background brightness and blur to meet this.
  + If you use a background image, it must be easily seen (stretch image if needed). Feel free to hardcode dimension numbers.

**Stretch Features**

**The following features are strictly optional, and will count for no credit until you satisfy all of the above requirements.**

Stretch features will be assessed for extra credit points, and there is no cap to the number of points you may earn through stretch features. These are merely suggestions of features to implement, but you can earn extra credit through implementation of any additional features. In fact, thinking outside the box and implementing useful features not mentioned in this spec is probably more likely to earn more extra credit than just implementing the below features! Please feel free to run your ideas of additional features by the instructor, TA, and classmates by posting in #project1.

Some ideas of stretch features may include:

1. [Backend] Implementing search
   1. [Basic] User has the ability to enter a dish in a search field (for example, "sweet potato casserole"). Once they press enter, the site will use relevant results from Twitter and Spoonacular to display a recipe/photo/tweets about that dish.
   2. [Intermediate] User has the ability to start typing letters into a search field (for example, "sw"). As the user types, autofill answers will populate suggestions of dishes (for example, "sweet potato casserole", "swedish meatballs") that the user may click on.
   3. [Advanced] User has the ability to enter the name of an ingredient in a search field (for example, "chickpeas"). Once they press enter, the site will search Spoonacular for dishes with that ingredient and display one randomly (for example, "chana masala").
2. [Frontend] Carousel
   1. [Basic] Pictures and tweets will swap out every few seconds for the dish. There is no animation.
   2. [Intermediate] Pictures and tweets will swap out every few seconds for the dish using a fade in/fade out animation.
   3. [Advanced] Multiple pictures are laid in a collage view, and multiple tweets are featured as a word cloud.
3. Your suggestions here! Anything goes!

**Deploy your app using Heroku**

This will launch your app to the world. See documentation here: <https://devcenter.heroku.com/articles/git>. You are encouraged to choose an app name that fits your theme, but you may stick with defaults if you can’t come up with anything.

**Submit your code via GitHub**

You’ll need a GitHub-hosted repository to turn in your code; see documentation at <https://help.github.com/articles/create-a-repo/>. Ensure that:

[TODO]

* Owner is the course organization (<https://github.com/msu-sp20-cosc458>)
* Name is project1-<your-email>, where <your-email> should be replaced with your email (sr66 should be project1-sr66, for instance)

You are required to incrementally keep GitHub up to date, but you only need to have 3 pushes to GitHub via Git for the first project. Nonetheless, I recommend practicing more than the minimum, as this is common practice.

**Document your work**

Your code submission must include a file named README.md at the top level. The file should be in Markdown (<https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>), and must answer the following questions:

* What are at least 3 issues you encountered with your project? How did you fix them?
* What are known problems, if any, with your project?
* What would you do to improve your project in the future?

# HOW YOU WILL BE EVALUATED

[TODO]

As outlined in the course syllabus, this project will be worth 150 points, 10% of your final grade. Those points break down as follows:

* 5 points: setup/upkeep
  + 3 points: GitHub repository has at least 3 substantial, spaced-out pushes from Git at least 5 days apart, each with descriptive commit messages detailing what was changed
  + 2 points: GitHub repository is named by your email (project1-sr66)
* 45 points: web app functionality
  + 3 points: server-side component of web app is written in Flask in Python
  + 2 points: quotes are dynamically generated via Twitter’s REST API per load
  + 15 points: recipe data are dynamically generated via Spoonacular REST API per load
  + 2 points: API keys are configured via config vars and not checked into code
  + 1 point: quotes are not filtered based on user, OR quotes are filtered based on user and user variety rule is met
  + 1 point: quotes page is located at /, or the root
  + 5 points: quote and recipe pairings are different across 7 page loads
  + 4 points: quote and recipe pairings form consistent theme across 7 different loads
  + 2 points: recipe title appears on page
  + 2 points: recipe serving/prep time appears on page
  + 2 points: image of recipe appears on page
  + 2 points: ingredients of recipe appears on page
  + 2 points: link to recipe page on appears on page
  + 2 points: Any and all links are legible for at least 7 page loads
* 10 points: Heroku deployment
  + 10 points: web app is deployed via Heroku
* 10 points: Code styling (these points can be made up. See below.)
  + 10 points: code is styled well and organized
* 50 points: Code explanation (meeting with instructor/TA)
  + 50 points: Explain your code (e.g. if 5 lines/sections are deleted for 10 points each, explain what they did and why you need them)
* 30 points: documentation
  + 5 points: README.md is in the root of the repository on GitHub
  + 5 points: Theme is explained in readme file
  + 5 points: Acknowledgement of known problems, if they exist, in readme file
  + 15 points: Description of 3+ issues and how issues were solved in readme file

**\*\*\*Submitted projects that do not run, are private, or not in the org folder will be late (20% off)\*\*\***

# COMMON PITFALLS

**Can’t access Cloud9 app in browser (Oops page)**

Chances are, you will need to bind to port 8080 in your Flask configuration (<https://damyanon.net/getting-started-with-flask-on-cloud9/>). You may also want to make sure you are running your app (by running **python name\_of\_app\_file.py** in your terminal)

**Made some changes and saved in Cloud9, but they’re not showing up on the preview**

Chances are, you will need to restart your webserver. If you don’t want to have to do this every time you may run your app in debug mode (<http://stackoverflow.com/a/17322636>). If that doesn’t work, your browser may be caching your app. Hard refresh your preview tab (not your AWS tab).

**Renamed a project and now Heroku is broken**

Heroku app names are stored in two places; the first is in their database so they can show you your apps dashboard, and the second is in your local Git repository as one of the remotes. If these two don’t match, then chances are things will be broken. Please read through this for details on how to fix the problem: <https://devcenter.heroku.com/articles/renaming-apps>

**[Errno 98] Address already in use when I try to start my web server**

Web servers all need to listen on the same port (port 80), so if you see this, then chances are you already have another web server running in a tab somewhere. You can only have one at a time! If you can’t find another one running somewhere, chances are it may be running in another tab and/or browser. The first answer at <https://stackoverflow.com/questions/19071512/socket-error-errno-48-address-already-in-use> may be helpful.

**Tried loading my deployed app on Heroku but the page is blank/an error page**

You’ll need to debug this by looking at the app logs from Heroku. To view the logs, simply ‘cd’ into your web app’s directory in AWS (yes, AWS), and run heroku logs --tail in your Cloud9 environment (this command is also seen on your Heroku page that errored out).

# ADDITIONAL QUESTIONS AND CLARIFICATIONS

If you need help on the assignment, or are stuck on any part of it, don’t hesitate to discuss with fellow students on Slack in #project1. We will monitor these channels and provide additional help if needed.

If you would like clarification on any part of this document, please contact us by posting in #project1. Depending on the nature of your clarification we may update this document.