Coding Temple Project Requirements

The following document will provide clarity on the requirements Coding Temple will be looking for in final projects for successful completion of the course. Each student is required to produce a working application using the technologies that were discussed in class.

The following topics and technologies are acceptable for project completion:

- Frameworks
    - React
    - Flask
- Databases
    - PostGreSQL
    - SQLite
    - Firebase
- A Third party Web API
    - List of Public Web APIs
- Languages
    - Python
    - JavaScript/TypeScript

Project Guidelines:

The following guidelines are a "starting point" for each student's project and should be treated as minimum requirements. Each student has the liberty to expand more on the requirements, but the below details are the bare minimum.

The project is to be built with one of the frameworks discussed in class, should interact with some form of database, and allow for interaction between/integration of a third-party web API.

Functions of a successful application:
The application a student builds should have the ability to Create, Retrieve, Update and Delete data.
Categories available for use regarding the Web APIs:

- **Music**
- **Food & Drink**
- **Health**
- **Weather**
- **Sports & Fitness**
- **News**
- **Books**
- **Art & Design**

Getting Started (Full Stack Project Reqs)

First, start by choosing the type of data you would like to use by picking a Web API from the aforementioned categories. Once you have that information, choose your web framework of choice to build the project in (Flask or React). With that in mind, decide how you would like to design your project. If you need inspiration feel free to use dribbble.com for guidance.

Building and Design Requirements

Each student's project should have a clean layout (doesn't have to be award-winning) that displays the usefulness of the application. **If your application has different areas to showcase, make sure proper navigation is available.** (i.e a user should be able to navigate through the application if needed via a navigation bar).

You may use the following frameworks to help aid in the design process:

- Bootstrap
- Tailwind
- Bulma
- MaterialUI

Functions of a Working Application

Each student's project should have the ability to

- **Gather & Display Data (CRUD)**
    - gather information from a third party Web API (if applicable)
    - gather information from a user(s)
    - Store information gathered from a user(s)
- **Sign up & Sign In/Out**
    - Give the user(s) the ability to update their information
    - Provide a way for a user to sign up for the application
    - Provide a way for a user to sign in to the application
    - Provide a way for a user to sign out of the application
- **Clean User Interface (UI)**
    - Provide a clean and user-friendly User interface

Successfully Completing the Project Requirements

Each student will be required to demonstrate a working application at about 75%+ competition. Any student that can not demonstrate a working knowledge of the subject matter taught in class by demonstrating/presenting to Coding Temple Staff **can not** be considered a graduate of the program. Special exceptions of a 50% or lower completed project ***may*** be extended solely at the discretion of Coding Temple staff on a case-by-case basis.

Data Engineering Project Requirements

If you plan to create a data analysis/engineering project, you will need to extract a dataset from some third-party data vendor. Once you have the data, perform some sort of transformation on it (if the data requires it that is) then Load the data afterward into a data store (PostgreSQL database, Cloud Data Storage, etc [OPTIONAL ONLY IF YOUR DATASET REQUIRES Cloud Usage]). From that point, you will need to analyze the data to tell a story about the data.

How you decide to display the data will be completely up to you. However, some guidelines will be:

- Create a slide deck to show your results (Tell the story of your data - ie: How you accessed it, what you originally thought about the data vs what you found)
- Create a front-end for the display of your data (Note: This may require an API to be created so that your frontend can consume the data to display it.
    - Resources for Showing data in the frontend
        - ChartJS
        - D3JS
        - PlotlyJS