# Course Project

Due: 12/9/22 @ 11:59pm

---

**Content Covered**
All of it

---

## Description

In this project you will build a small web application to visualize a dataset. You have the freedom to choose a dataset that is meaningful to you (either personally, or to your research, or to your field), as well as the visualizations that make sense for your dataset. You are also expected to incorporate User Interface (UI) components in the front end of the web app, so that the user can interact with the app to affect the visualizations.

The application will be complete with a **JavaScript** visualization in the **front-end** of the application, and a **Python**-based web server **back-end** that provides cleaned and filtered data to the front end.

Because each student's project is unique to them, there is no autograding available. The project is worth 30 points overall, with weight given as shown below.

---

## Front-End Visualization

Your web application must produce (**at least**) two distinct plot.ly visualizations - neither can be a bar graph, line graph, or scatter plot. See https://plot.ly/javascript/ for options. You may have additional visualizations, which can be of any type.

---

## Front-End user controls

Your web application must incorporate some form of user input in the front end.

One input must be sent to the back-end web server where it is used to select, constrain, filter, manipulate **the data** being visualized in some way. For example, if your chosen data set spans multiple years, allow a user to select a specific year to focus on. This must involve at least one POST request to your server.

The other input may be sent to the back-end web server, or it may be used in the front-end, to affect some change in the visualization. For example, if your visualization has two axes, x and y, clicking a button could interchange the data and titles for the x and y axes. This need not involve a POST request, though it may. It could be handled entirely within the front-end.

# Back-End Data Processing

The back end of the web application must be able to download a fresh copy of the data from its original data source, and then cache the data locally in a SQLite database or a CSV file (but not both).

All data processing (e.g. cleaning of data [e.g. removing data elements lacking a relevant property], filtering [e.g. selecting data for a specific year], computing derived values from the raw data [e.g. computing the number of data entries]) must happen in the back end.

---

# Submission

Submissions will be made to AutoGrader as normal. As soon as the deadline passes, I will take whatever your most recent submission is, and grade it manually.

---

# Grading Process

During class time during the last four days of classes students will give short (~6 minute) presentations on their projects. The presentations should not only showcase the web app code and its execution, but also talk a little bit about why this data is important/interesting to you and what the visualization shows. **Your grade will be based on your final submitted code [24 points] and your presentation [6 points].**

# Code Point Breakdown

## Front-End [ 12 points ]

### HTML [ 4 points ]

- Basic HTML structure
- <script> tags for code/library retrieval
-  tags for graphs
- <body onload="..."> 
- Input elements

### JavaScript [ 8 points ]

- AJAX requests (GET and POST)
- Callback function(s)
- Visualization with plot.ly

## Back-End [ 12 points ]

### Python Web Server [ 4 points ]

- Correct 'bottle' routes/request handling

### Python Application Code [ 8 points ]

- Data retrieval from JSON source
- Local data caching (csv or SQLite)
- Data cleaning and processing