In this lab, you will use Flask and other Python libraries to allow users to browse and manage resources tracked by a .csv file.

You will have the entire class period on the 22th of April, and half to work on the lab. Assignments will be due before Friday midnight the 26th of April. You can choose to work on the lab individually or with your classmates, but each person must turn in their own work.

### Project Description

Create a Flask application that allows users to interact with a set of data, either stored in a .csv file or from a web-based API.

### Deliverables

No matter what option you choose, you will submit a Flask application that includes:

* Index and show routes with appropriate templates, allowing users to view a collection or individual items from your chosen data set.

### Process

Follow this approach to start addressing your chosen problem statement. Feel free, as you begin exploring your data, to develop different questions than the ones provided. Let the data guide your analysis.

| **Option** | **Process** |
| --- | --- |
| Option No. 1: Locally-stored .csv "database" | 1. Decide what kind of data you want the application to manage. Examples include budget items, work request tickets, investment performance, book reviews, and more. 2. Create the .csv file with specific columns and a first row or two as "seed" data for your application to test while developing. 3. Create your index route and template that shows all items in the .csv file on the web page, including links to each individual item's details page. 4. Complete the show route that displays the details of each item in the .csv 5. Include a create route that allows users to add new data to the file. This should accept a POST request with the new data. |
| Option No. 2: API browsing application | 1. Choose an API to work with from common lists of free APIs. Make sure the API has easily accessible routes for your "index" and "show" routes of a specific resource. 2. Create functions that successfully use the requests library to make valid API requests and return the relevant data from the response. Don't worry about Flask yet, just make sure the API requests work as intended. 3. Set up your Flask index route and template to display the collection of data, including links to individual resources 4. Include query parameters in your index route that allow users to perform a more specific search for data matching their parameters 5. Complete your show route that allows users to view details of each individual item in the API results |