Books

Objectives

- Become more comfortable with Python.
- Gain experience with Flask.
- Learn to use SQL to interact with databases.

Overview

In this project, you'll build a book review website. Users will be able to register for your website and then log in using their username and password. Once they log in, they will be able to search for books, leave reviews for individual books, and see the reviews made by other people. Finally, users will be able to query for book details and book reviews programmatically via your website's API.

Getting Started

GitHub Classroom

We'll again use GitHub Classroom to distribute projects and collect submissions. To begin Project 1:

1. Click here (https://classroom.github.com/a/VizLIS7B) to go to the GitHub Classroom page for starting the assignment.
2. Click the green "Accept this assignment" button. This will create a GitHub repository for your project. Recall that a git repository is just a location where your code will be stored and which can be used to keep track of changes you make to your code over time.
3. Click on the link that follows "Your assignment has been created here", which will direct you to the GitHub repository page for your project. It may take a few seconds for GitHub to finish creating your repository.
4. In the upper-right corner of the repository page, click the "Fork" button, and then (if prompted) click on your username. This will create a fork of your project repository, a version of the repository that belongs to your GitHub account.
5. Now, you should be looking at a GitHub repository titled **username/project1-username**, where **username** is your GitHub username. This will be the repository to which you will push all of your code while working on your project. When working on the project, do not directly push to the **Web-2020/project1-username** repository: always push your code to your **username/project1-username** repository.

PostgreSQL

For this project, you'll need to set up a PostgreSQL database to use with our application. You can set up PostgreSQL locally on your own computer, or use a database hosted by any online web hosting service.

Python and Flask

1. First, make sure you install a copy of Python (https://www.python.org/downloads/). For this course, you should be using Python version 3.6 or higher.
2. You'll also need to install `pip`. If you downloaded Python from Python's website, you likely already have `pip` installed (you can check by running `pip` in a terminal window). If you don't have it installed, be sure to install it (https://pip.pypa.io/en/stable/installing/) before moving on!

To try running your first Flask application:

1. Clone your **username/project1-username** repository from GitHub (note: this is NOT your **Web-2020/project1-username** repository).
2. In a terminal window, navigate into your `project1` directory.
3. Run `pip3 install -r requirements.txt` in your terminal window to make sure that all of the necessary Python packages (Flask and SQLAlchemy, for instance) are installed.
4. Set the environment variable `FLASK_APP` to be `application.py`. On a Mac or on Linux, the command to do this is `export FLASK_APP=application.py`. On Windows, the command is instead `set FLASK_APP=application.py`. You may optionally want to set the environment variable `FLASK_DEBUG` to `1`, which will activate Flask's debugger and will automatically reload your web application whenever you save a change to a file.
5. Set the environment variable `DATABASE_URL` to be the `URI` of your database.
6. Run `flask run` to start up your Flask application.
7. If you navigate to the URL provided by `flask`, you should see the text "`Project 1: TODO`"!

Requirements

Alright, it's time to actually build your web application! Here are the requirements:

- **Registration:** Users should be able to register for your website, providing (at minimum) a username and password.
- **Login:** Users, once registered, should be able to log in to your website with their username and password.
- **Logout:** Logged in users should be able to log out of the site.
- **Import:** Provided for you in this project is a file called books.csv, which is a spreadsheet in CSV format of 5000 different books. Each one has an ISBN nubmer, a title, an author, and a publication year. In a Python file called import.py separate from your web application, write a program that will take the books and import them into your PostgreSQL database. You will first need to decide what table(s) to create, what columns those tables should have, and how they should relate to one another. Run this program by running python3 import.py to import the books into your database, and submit this program with the rest of your project code.
- **Search:** Once a user has logged in, they should be taken to a page where they can search for a book. Users should be able to type in the ISBN number of a book, the title of a book, or the author of a book. After performing the search, your website should display a list of possible matching results, or some sort of message if there were no matches. If the user typed in only part of a title, ISBN, or author name, your search page should find matches for those as well!
- **Book Page:** When users click on a book from the results of the search page, they should be taken to a book page, with details about the book: its title, author, publication year, ISBN number, and any reviews that users have left for the book on your website.

- **Review Submission:** On the book page, users should be able to submit a review: consisting of a rating on a scale of 1 to 5, as well as a text component to the review where the user can write their opinion about a book. Users should not be able to submit multiple reviews for the same book.
- **API Access:** If users make a GET request to your website's /api/ route, where is an ISBN number, your website should return a JSON response containing the book's title, author, publication date, ISBN number, and review count. The resulting JSON should follow the format:

  ```
  {
      "title": "Memory",
      "author": "Doug Lloyd",
      "year": 2015,
      "isbn": "1632168146",
      "review_count": 28
  }
  ```

  If the requested ISBN number isn't in your database, your website should return a 404 error.
- You can use raw SQL commands (as via SQLAlchemy's execute method) in order to make database queries. And you can also use the SQLAlchemy ORM (if familiar with it) for this project.
- In README.md , include a short writeup describing your project, what's contained in each file, a list of all of the tables in your database and what column names (and data types) are in each column, and (optionally) any other additional information the staff should know about your project.
- If you've added any Python packages that need to be installed in order to run your web application, be sure to add them to requirements.txt !

Beyond these requirements, the design, look, and feel of the website are up to you! You're also welcome to add additional features to your website, so long as you meet the requirements laid out in the above specification!

How to Submit

1. Go to the GitHub page for your **username/project1-username** repository (note: this is different from the **Web-2020/project1-username** repository).
2. On the right side of the screen, click the Pull request button.
3. Make sure that the "base fork" is `Web-2020/project1-username` , and the "head fork" is `username/project1-username` .
4. Click "Create pull request".
5. On the next page, click the "Create pull request" button again.