Setting up a database with Flask

Initializing and running the application

- Flask Application Setup: A Flask application is an instance of the Flask class, which acts as the central object for configuring routes, databases, and other application-specific settings. This will be created with a `create_app` function in the `__init__.py` file.
- Handling the Instance Folder:
  - The instance folder is used for storing local data that should not be version-controlled (e.g., configuration secrets, database files).
  - Ensure the existence of the instance folder using `os.makedirs(app.instance_path)`.
- Running the Application:
  - Run the application with `flask --app flaskr run --debug` from the terminal, enabling debug mode for live updates and interactive debugging.
  - Access the application by visiting `http://127.0.0.1:5000/` in a web browser to see the home page.

Creating database

- Database Connection:
  - Establish a database connection using the `sqlite3` module, typically tied to individual web requests in Flask applications.
  - Utilize Flask's `g` and `current_app` to store and manage the database connection per request.
- Creating and Closing Database Connections:
  - `get_db` function to create and reuse a database connection stored in `g` for the duration of a request.
  - `close_db` function to close the database connection at the end of a request, registered to Flask's teardown process.
- Defining the Database Schema:
  - Use SQL commands to define tables and columns, demonstrated with `user` and `post` tables.
  - The schema includes essential fields for users and posts, with relationships defined via foreign keys.

- Initializing the Database:
    - Implement `init_db` to run schema SQL commands, creating a clean database structure.
    - A CLI command (`init-db`) is created using `click.command()` for initializing the database, facilitating easy database setup via the command line.
    - Call separate function `populate_db` to populate the database with data from the csv
- Registering with the Application:
    - `create_app` function to register database-related teardown and CLI commands with the Flask application instance.
    - Ensures that database initialization and connection management are integrated into the Flask application lifecycle.
- Running Initialization Command:
    - Instructions for using the `flask --app flaskr init-db` command to initialize the database, creating the SQLite file within the application's instance folder.
    - This should only need to be run once

Initializing and running application:
https://flask.palletsprojects.com/en/3.0.x/tutorial/factory/
Creating database:
https://flask.palletsprojects.com/en/3.0.x/tutorial/database/