Part 1: Getting Started with Pythonanywhere

1. Please watch [“Lab Recorded Video”](https://psu.mediaspace.kaltura.com/media/Jeremy+Sean+Gamble%27s+Zoom+Meeting/1_i5ksctgl) either before or while completing the lab
2. Follow this link and sign up for a free account through Pythonanywhere: <https://www.pythonanywhere.com/>
3. Click the web tab and familiarize yourself with this page. The green reload button must be pressed every time you update your code to populate the website with it. The hyperlink above is the link to your website.
4. Click the link above, which should display a page that says something along the lines of “Hello from flask!”. This is your running front end that we will be connecting to a back-end database.
5. Go back, and now click the files tab, and navigate to the /myfiles directory. Open the flash\_app.py file in a new tab.
6. The file you just opened is where you will code your flask app that will connect your front and back-ends together, but for now is blank. We will come back to this app many times, so leave it open for now.

Part 2: Creating a Back-End and Connecting it to the Flask App

1. Go back to your other tab, and now click the consoles tab.
2. Click the MySQL link under start a new console and create a database \*\*NOTE: Make sure to save the password and database name created with the MySQL database so you can connect to it later. \*\*
3. Either create an example table or use some of the code from your back-end part of the course project to populate the database.
4. Open your flask app tab and write the code for connecting to the MySQL database and creating a cursor object using the information you wrote down from the MySQL console to connect.

Part 3: Creating Pages for the Front-End and Connecting it to the Flask App

1. Go back to files, then again to /mysite, and create a new directory called templates.
2. In here, create two html files, one called home.html, and another called display.html.
3. In the home.html file, create a button that can be interacted that will call the /display route.
4. Go back, and edit the display.html file this time, and create a basic display that will show the data from certain tables and have a form and button that can take input. These will be used to send data back to the database.
5. Go back to the flask app and create a route called /, that just returns the template for home.html. This will set it so that when anyone clicks on the site, it loads the home.html page by default.
6. Now create another route, this time called /display. This route will need both the GET and POST methods, as well as an if statement for what the route should do depending on which method is being used. If the method is POST, take the data from the form and insert it into the database using the cursor object, then redirect back the /display route. If the method is GET, fetch some data from the database and display it. \*\*NOTE: The data that you chose to display MUST be the same data you are adding to this way we can see that the database updates on the back-end, then gets pulled again by the front-end. \*\*

Tasks to do:

1. Either creating your own new database, or what you have started for the final project, please create a MySQL database that houses some tables to use for this lab.
2. Create a flask app that will connect to your MySQL database and can feed data to front end pages as well and send data back to your MySQL database.
3. Create a few front-end pages that will be connected to through the flask app that will populate with and send data to your MySQL database.

Deliverables:

1. Provide evidence of your choice that shows you have successfully ran your code, and that the webpage successfully shows the data you chose to use and can it can be updated. Here are some examples of evidence you could provide:
   1. Screenshots of steps you have taken with your name or userid shown on the captured screen.
   2. Recorded session with webcam and microphone on and computer screen shared.
   3. Any other technique of your choice provided you have received approval from the instructor.
2. Upload the codes used for the MySQL server, the flask app, as well as the front-end html files.