CSE22: Introduction to Programming
Lab1

1 Introduction

In this lab you will become acquainted with the CodeSync development environment, you will explore the structure of a proper web application, and you will create some good looking web interfaces.

2 CodeSync

If you are reading this, you are most probably reading it in CodeSync, which means you have found your way here. CodeSync is divided into four major areas, most of which are self-explanatory. On the left sidebar there should be a listing of the files and folders in the project, which is similar to any other file explorer utility. The left sidebar, shows a list of participants who are currently logged into the project, at the top, and a chat message interface at the bottom, which participants can use to send messages to the group. Along the bottom area of the interface is a terminal window, which will be used to compile and run our programs, and finally, the majority of the area in the middle of the screen is where the files are displayed.

Depending on your access level in a given project, you may or may not be able to edit the files or use the terminal. In class demos, your access level is read-only, so you can look but you can not make any modifications. For your lab projects, you have a higher access level so you are allowed to do anything in the project, until the deadline, at which point the system automatically demotes your access level and you will no longer be able to make any modifications. That is why it is imperative that you complete your labs on time.

3 The Project Structure

If you look at the sidebar, you will see that our C++ project is made up of a number of folders. These are `static`, and `templates`. The `static` folder is where we put all our assets, such as pictures, JavaScript, and CSS files (JavaScript and CSS are not a part of this lab, but just know that they go in `static`). The `templates` folder is where we put all our HTML pages. Any project will typically be made up of many HTML pages, so they are all in `templates`.

Finally, there is a Python file called `app.py`, and this is what drives our web applications. It is a simple web server that allows web browsers from anywhere on the internet to connect to it, and right now it just serves up a single web page called `index.html`. We will constantly be adding new functionality to our web servers, but for now that's a good start.

4 Running the code in CodeSync

To start running the project, go to the terminal and type

```
pip install -r requirements.txt
```

and press Enter. This will install all the project dependencies on your workspace. You only need to do this once and then everything will be installed. Once that is done, you can run your server, by typing:

```
python app.py
```

If the server launches successfully, then you should see some output in your terminal, which looks something like this:

```
* Serving Flask app "app" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:9000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 319-935-144
```

If it is running, then you will be able to visit your URL and interact with the page. To find your URL, simply look at the green stripe at the bottom of your terminal. On the right corner of it there is some text of the form `terminal-[num]-[key]`, for example one of our class demos looks like `terminal-660-5df99d5c`. We are interested in the `[num]` portion, which is between the two dashes. In the class demo example, the `[num]` is 660.

Once you know your project number, you can form your url, which is always of the form:

```
https://app-[num].ucm.kyrilov.xyz
```

So the class demo url is `https://app-660.ucm.kyrilov.xyz`.

The last thing you need to know is how to stop your server from running. To do that, click on your terminal in CodeSync to make sure it is selected. Then press Ctrl+C. If you have successfully stopped it, your terminal will print out `^C$` on its last line.

The first think you should do is install your dependencies according to the instructions above, then run your server and try to visit your URL. Make sure you are comfortable with that. Then make some minor change to your HTML page (like change the h1 heading to something else), and refresh your page to see if the changes took effect. If they did not, you can try to rerun your server by stopping it and starting it again. Also viewing the page you are developing in private/incognito mode is better as the browser does not try to cache as much in this mode.

5 Online Resources

We will be using resources from W3 Schools quite extensively for our interface designs. If you are looking to brush up your HTML skills, go to the HTML section of W3 schools. For this lab, we are mostly interested in the Bootstrap section of. As in the class demo, when you find a bootstrap component you want to use, simply copy and paste the HTML in you own page. Then do your modifications to make it fit in and test to see if your page looks like what you expected it to look like.

6 Tasks

Your task for this lab is to create a webpage that represents or promotes you or something from your community that you are passionate about. I do not want to see a replication of the demo page we did in class. I want to see your designs.

Requirements

- You must have more than one page
- You must have a navigation menu on top of each page, which shows the page that is currently active
- Your interface has to look good, which means you are using Bootstrap (no Craigslist lookalikes)
- Youp pages have to have real content, including text, pictures, etc.

This lab assignment is out of 100 points. The due date is Monday September 21, 202 at 11:59 pm.