CSC309: Programming on the Web

Toronto Fitness Club: part 2

Due date: Thu Dec 8, 2022 22:00:00 | 00:00:00:00 remaining

Overview

Please find the general information about the project, features, interviews, mentor sessions, rules, etc. on the part 1 handout. All rules and restrictions apply to this part as well.

In this part, you are to implement the React frontend of TFC and connect it to your backend server. The deliverable is a fully-functional website that is shippable, presentable, self-explanatory, smooth, and beautiful.

Deadline

This part is due on **Thursday, Dec 8, 2022 at 10:00 PM**. Late submissions of x hours (starting from 1) will deduct following percentage from your group's grades.

$$p = 0.0002x^{3.06}$$

Notes

Course Links

Schedule	Google Calendar
Announcement - Q&A	Piazza
Submissions	Markus
Informal chat	Discord
Lab sessions	Zoom

The interviews will simulate the real-world delivery of your application to the business owner. That is, the business owner is looking for a **complete website** that has a **very smooth and bug-free UX** and a **beautiful UI**. Pay utmost attention to the UX, where users should be able to navigate around without any problem or any need for instructions from someone else. You certainly remember websites that you left before a minute because you could not find a page you where looking for, or parts of a button was not clickable because of a simple frontend bug. You certainly did not have any empathy for the potentially countless hours the backend and frontend engineers spent on building that website and whether other parts worked well or not. You left the website, and that was the loss of a potential customer for that business.

In our interviews, we will not be as harsh as mentioned above, but you should expect that the marking scheme is different from a classic project where the only important thing is that the code somehow *works*. You will lose marks is your website is incomplete (lacks some functionalities) or has major or minor UI/UX issues. Smooth UX is also about single-page experience, front-end validations, in-place error-handling, asynchronous requests, and clean re-renders; all of which offered by React.

This course is not about UI design; so you can freely choose any UI templates that is available outside. However, you must adhere to the common sense when it comes to your UI. Your UI must give a good first impression to users. Exploring your website should be delightful for users. A basic UI that everyone can tell it is the default of a framework (e.g., bootstrap) is **not** delightful to explore.

Your website should include pages. That is, even though you are building a single-page application, you should change the browser's URL accordingly so that users will be able to navigate through different places with the back/forward button or re-visit a specific page through its URL. Please refer to lecture 11 to find out about pages in React.

Different sections of your website must be reachable by a navigation bar. A user dropdown to see/edit profile, manage subscriptions, and logout is also required. The user must **never** need to manually change the URL (except for accessing the admin panel). Besides those requirements, the exact content of pages and the choice of first page (or landing page) is up to you. We recommend that your first page includes the search functionality, and also includes (or links) to the subscription plans.

Please make sure to check PF notes on Piazza every day for updates and clarifications

Pre-populated database

Unlike last part, you should push your database/media files to your repository. The reason is pre-existing data improves your presentation a lot. The business owner certainly does not want to navigate a blank website that contains only a couple of studios. To showcase your project, it is suggested that you include at least 10-20 studios with their full information (location, classes, images, amenities, coaches, etc.) Obviously, your strings, images, phone numbers, etc. cannot be random sequences of characters/bytes. You can enter data manually, or use online datasets (recommended), or even crawl similar websites.

Note 1: The data does not have to be original or genuine. For example, if you show the phone number (647) 643-6710, it does not have to be a real phone number associated with a gym. However, a blank number or something like (111) 111-1111 is obviously unacceptable.

Note 2: Make sure that your database/media is not too big that it would take a lot of time to clone/load.

Pagination

As mentioned in part 1 handout, every API that returns a list of objects should be paginated. (e.g., list of studios, upcoming class schedule, past payments). However, depending on your database size, you should choose your page size so that the TA can test that you implemented pagination correctly. For example, a page size of 200 when you only have 100 gyms means that the TA should create 100 new studios to check this feature, which is impossible. You can implement pagination via regular buttons (like google search on Desktops) or through infinite scrolls. Also, different endpoints can have different page sizes.

What to submit

You must push both your frontend and backend servers to your repository. Push two scripts as well: startup.sh and run.sh. The **startup** script should run all preparations such as creating the virtual env, pip installing all packages, migrations, npm install, etc. The **run** script should run both backend and frontend servers. The TA will clone your application and work with your frontend interface like an ordinary user and will check all the required features. It should be a fully-functional website where users interact without any help from other people. This time, there is no need for pushing documentation or Postman collections.

Note 1: You can very make any improvements you want to your submissions for the previous phases. You can even write them from scratch (not recommended because there is not that much time). We will not look at what you submitted for those phases and will only assess the files under the PF folder.

Note 2: Before submitting, make sure that your startup and run scripts work well on the virtual machines given to you. The TAs will run your application on clean instances of that virtual machine. If you have worked with another operating system, it is your absolute responsibility to double-check that your project works fine within the virtual machine as well.

Comments

Captain WiFi ★★★★★

I've always dreamed of being a web developer. Now my dream has come true, but ... I just wish the free healthcare sessions were actually offered by TFC; I really need it.

108 | Reply

Write a comment

Your email address will not be published. Required fields are marked *

Review*

Name*

Email*

✓ Notify me of new posts by email

SUBMIT