CS 696A Full-Stack Entrepreneurs App Development - Lab 5

Prof. Arya Boudaie - Fall 2025
Pace University

OBJECTIVE

In this lab, you will create a NextJS application that creates interactions based on a public API, and then work on optimizing suboptimal useEffect patterns.

SUBMISSION INSTRUCTIONS

You will work on this assignment in your assigned groups. Everyone must contribute in some manner to the final application. Your group's submission should be a word document with the following sections:

- Link to your application repository.
- Link to demo video or deployment link.
- Group Reflection
- Individual Team Member Reflection.
- useEffect Replacement Exercises

STEP 1: API VISUALIZATION APPLICATION

For this step, you will be creating your own NextJS application. Only one person will be creating the initial application and Git repository, but it's suggested everyone does this together. Initialize it by doing the following:

1. Check that you have npm and node installed on your command line by running npm −v and node −v in your command line. If you do not have it installed, install it, see instructions here.
2. Initialize your NextJS application by running npx create−next−app@latest in the folder you want this application to live in. Give the application a name, and answer the defaults for all the questions unless you have strong opinions about any of them. If you do not know

TypeScript or Tailwind you may not want to use them, but this could be good practice.

3. The command will create an application with many files. Some important ones:
   (a) .gitignore - list of files that will not get added to git.
   (b) package.json - configuration for file, you can add libraries here to install them.
   (c) package−lock.json - resolved library versions after installation.
   (d) README.md - Instructions for running the application.
   (e) public/ - Folder with public assets (e.g. images).
   (f) app/ - Application logic folders.
   (g) app/page.tsx or jsx - Entry point to your application.
4. At the top of app/page.tsx, add the following line: "use client"; - this will tell Next to render these components on the browser and allow you to use browser APIs. In future classes you will learn about server vs client rendering.
5. Create a components/ folder in the root directory; this will hold all the components you need to make in the rest of your application.
6. Push this application to GitHub, and add all your team members as collaborators in the repository settings.

Now that you have the application set up, you are ready for the core assignment. Your team's job is to find an interesting free and public API similar to the Pokemon API we looked at in class, and create interesting interactions using them. The API endpoints should be GET requests to retrieve the public data; you should not be POSTing any data to the API. Each team member needs to have their own interaction on the website, so make sure the API is extensive enough to have multiple interactions. You can help each other with the interactions (e.g. make commits to fix bugs, styling, etc) but each team member should own an interaction. The only other requirement is that your application must use a component library like MUI, shadcn, etc... If you find an API related to your final project, that is fine, but it does not have to be. Please post the API you are using in the class Discord so other teams do not use the same one.

To find example APIs, you can search online for "Free APIs". Some example listings:

1. Free APIs
2. Public APIs
3. Reddit Thread
4. API List
5. Best Weather API Reddit Thread

Tips:

1. For CSS, you can either use CSS Modules, regular CSS, Tailwind, or any other CSS library you

can find. I suggest putting some time into making these interactions look nice as practice.
2. Try using useEffect first for querying the API, and then use a library for managing the query like tanstack query.
3. If your API requires you to sign up and gives you an API key, you can store that in the .env file starting with the NEXT_PUBLIC_ prefix (this is needed for the variable to be visible to the client), and then reference it in your code by using process.env.NEXT_PUBLIC_YOURVAR in the client code. Note that this isn't very secure; we will learn about where these secrets should live in future classes. Note that the .env file is in the gitignore, so this will not be on Github; this is good as you don't want to leak this code. Every API will be different in how they want you to put this token in the API request, but most will want it as authorization headers in the fetch.
4. In the most basic version, your app/page.tsx file can just contain the different components all on the page. If you want to be fancier, you can put them in Tabs (e.g. with MUI or shadcn).
5. Many governments, museums, public institutions, etc have APIs. If there's one that interests you, look for their APi in specific.
6. If you want to add a library from npm, run npm i <pkg name> −−save to install the new package, which will update package.json. Push the change to Github so your teammates can install the package too with npm i.

Once you are done, you can either make a video demo of the application running locally, or you can attempt to deploy it to a service like Netlify or Cloudflare workers.

REFLECTION

As a team, write a paragraph or so answering the following questions:

1. What API did you use, any reason why?
2. What component library did you use and why?
3. Did you use any other packages?

Each member should write their own answer for these questions:

1. What interaction did you create?
2. In the components for your interaction, what exists as state, and what exists as props? Why did you do it like that?
3. What did you find easy, and what did you find difficult about implementing this interaction?
4. Are there any bugs in your system?

You might want to move some of this to the README file, so people know more about the app you created.

USEEFFECT REPLACEMENT CHALLENGES

This one will also be done as a team - I suggest you all work on these together or split the exercises up appropriately.

First, go over the article "You might not need an effect" - many of the examples were covered in class but it is definitely worth a read.

At the end, there are some challenges where they ask you to replace a useEffect with a better code example. Write the code for them in your browser (you don't have to submit it), then in your own words in the submission, explain why the useEffect solution was suboptimal and how you optimized it. Be concise in explaining how you did it. Avoid clicking the button for the solution unless you are completely stuck; I am looking for unique solutions.

AI ATTESTATION

Include one of the following statements at the top:

- I attest that I did NOT use ChatGPT or any other automated writing system for ANY portion of this assignment.
- I acknowledge using ChatGPT [or some other system; name it] for the following tasks: [list them]. Some surprising benefits of using GPT include the following: [list them]. I also encountered a few unanticipated challenges: [list them].