HTN FE Challenge Writeup

Name: Aryaman Singh
Stack: React (TypeScript) & GraphQL
Libraries Used: Apollo, react-modal, Lodash, Moment, Classnames

Hey! This is my submission for the Hack the North Frontend challenge 😃 In case you have any issues running the project, please contact me at:

Email: aryaman.singh.2012@gmail.com

Discord: PuffyPanda#1588

Running the project:

This is a React app that you can run using npm. At the base directory, run:

`npm install`

To run the app:

`npm start`

The Node version I'm running is `16.13.1`, but it should also work on `v14`. I've tested this on Mac and Linux, and on Firefox and Chromium browsers.

Section 1

How did you plan out the structure and design of it?

I like to visualize the structure of a project before I start working on it. I used **FigJam** to keep track of all of the information I was finding and what I was working on, which you can find here: https://www.figma.com/file/PW3cSDhEGxcpFxTW7Atp3p/HTN-FE-Challenge

The main things I wanted to keep track of were:

1. The requirements of the project, and any associated questions/issues I had
2. Any relevant documentation, links, or colors I wanted to keep on-hand
3. A very basic Kanban board for tasks

For the design, I also spent a bit of time prototyping on **Figma** to figure out how certain elements in my app would work together in terms of layout or colors.

How did you decide on what tools you've used?

- I have quite a bit of experience with **React** and TypeScript, and I believe a similar tech stack is used by the HTN team, which is why I decided to use this stack. React offers enough scaffolding to create a modern app easily while also giving you a lot of flexibility in how you want your app to look, while TypeScript helps you avoid mistakes that can haunt you in the future. (In other words, it helps with future scalability)
  - I used `create-react-app` to start since it's industry standard and I have experience with it.
  - In terms of Libraries, the main one I used was `react-modal` for the Event modals, since it offered the right balance of simplicity and functionality for this project. I've used MaterialUI Modals in the past, but I felt that it was easier to customize the styles of react-modal.
- I decided to use GraphQL and, specifically, **Apollo** to query the API. I'm very comfortable with GraphQL. Especially for TypeScript projects, GraphQL lets you easily control what data you are going to receive to reduce the amount of edge cases you have.
- I used Figma and FigJam for some basic designing and prototyping.

Did you encounter any problems? If so, how did you solve them?

One of the main problems that stumped me was how I was going to get Related Events to work. Writing down the list of event names was easy enough, but I wanted the user to be able to *open* the related events. I first attempted to solve it by creating links for each event and redirecting to that URL, but the design ended up being too complex for me given the time limitation.

I ended up realizing that my original design for the `EventBoxes` could be repurposed for the related events in the modal as well. With a little bit of code tweaking, I was able to re-use my existing code to add this functionality. This also had the added benefit of creating a “stack” for the user to open related events.

Are there any areas of your code that you’re particularly proud of or want to point out?

1. One of the main things I wanted to ensure was that my design would be **responsive.** I started by designing for phones and then seeing how that design would scale for larger screen sizes. I’m very happy with how usable my website is on phones, especially since a large number of people check schedules on their phones (at least for in-person Hackathons).
   This *does* come at the cost of the desktop design being a little bit boring, but that is something that can be worked on later. Going from a desktop design to a simpler phone design is much harder.

2. I spent some time early-on figuring out the color scheme of the app by color-picking parts of Hack the North’s current website and putting them through coolors.co to generate a color palette. I think my website's elements and background look visually consistent and appealing, and relatively on-brand with the HTN style. There are also different design cues for the various event types.

3. My `EventBox` component is scalable and self-sufficient, in the sense that it can be added recursively or in other elements and still function as intended. For instance, we could easily add “Conflicting Events” to the modal, similar to Related Events, or embed this EventBox on a blog post advertising a particular event.

Section 2

How would you extend the functionality of the app to scale with thousands of hackers?

1. One of the things I would do is refactor the GraphQL Query. Currently, all events and their details are loaded on initial page-load, and then are passed on to each component. This might cause issues as we scale the number of events the API returns. I would fix this by reducing the data that is requested in the useQuery on the main page (`App.tsx`) and shifting individual event detail requests to when the Event is opened (`EventBox/index.tsx`)
2. I would also improve the Login page by adding options to reset passwords and create accounts.

Would you add more features and performance metrics?

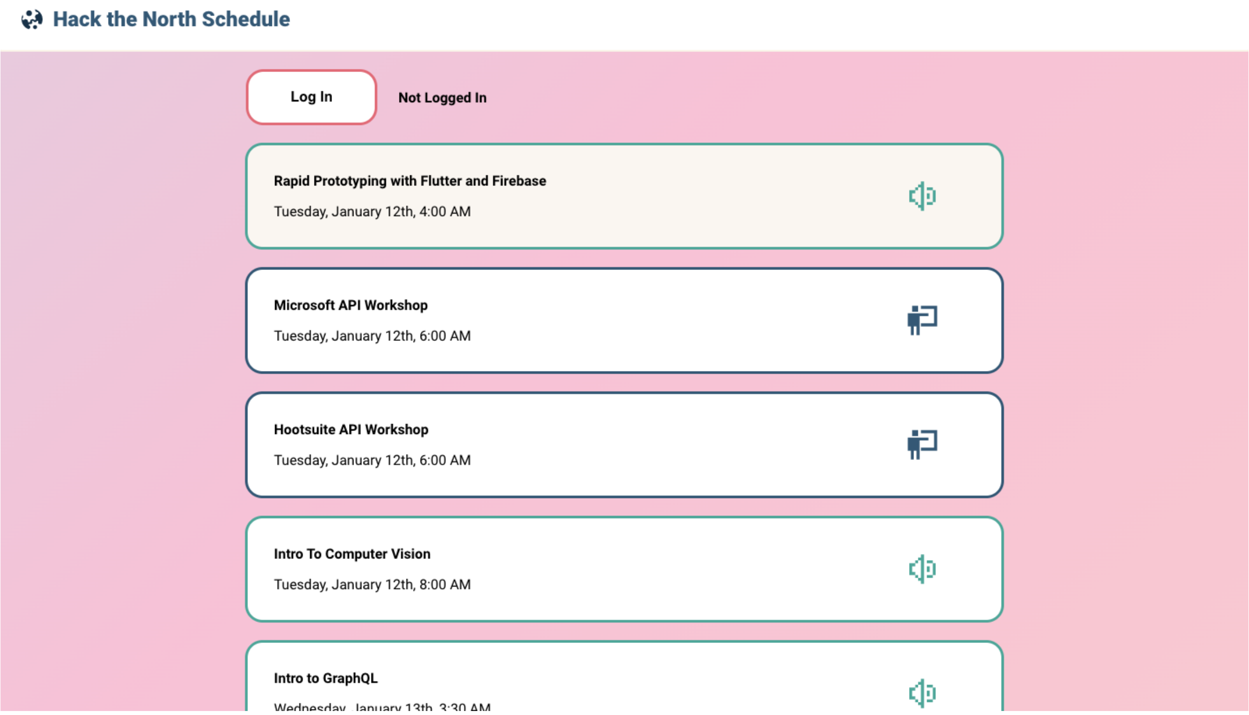
1. I believe **Filters** would be an awesome addition to the app, especially as we had more and more events and EventTypes. It should be a straightforward addition since Lodash has the `_.pickBy(...)` function which can filter the events array depending on the EventType, time of day etc.
    a. Searching for events would be another feature to look into, which can also use Lodash's `pickBy` function.
2. It would also be nice to create a bookmark option for events the hacker would like to attend,. They can be associated with the account of the hacker to synchronize between devices.

Section 3

Any other thoughts you have (not limited to the previous questions).

This was a very interesting project to work on, and I had a lot of fun trying to come up with a functional prototype. Though I think there are a few missed opportunities and improvements that can be made to it, my main focus was getting an accessible, responsive, and usable prototype up and running. 😎

Image Gallery



Hack the North Schedule

Log In

Not Logged In

Rapid Prototyping with Flutter and Firebase

Tuesday, January 12th, 4:00 AM

Microsoft API Workshop

Tuesday, January 12th, 6:00 AM

Hootsuite API Workshop

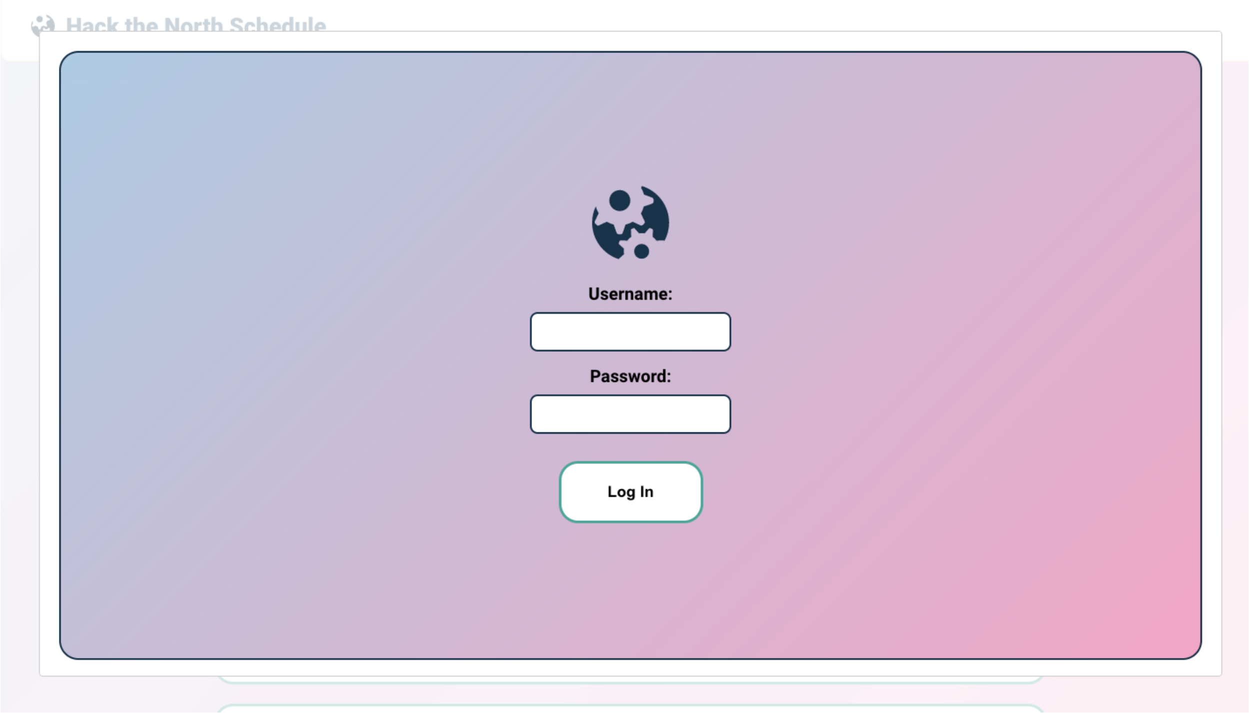
Tuesday, January 12th, 6:00 AM

Intro To Computer Vision

Tuesday, January 12th, 8:00 AM

Intro to GraphQL

Wednesday, January 13th, 3:30 AM

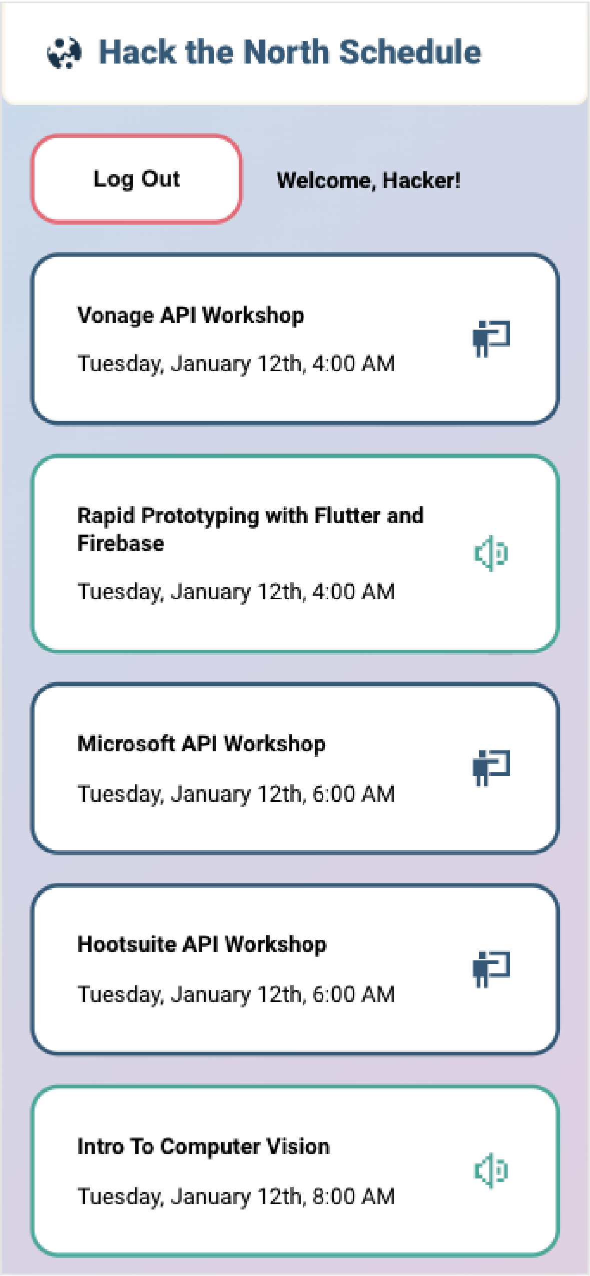


Hack the North Schedule

Username:

Password:

Log In



Hack the North Schedule

Welcome, Hacker!

Vonage API Workshop

Tuesday, January 12th, 4:00 AM

Rapid Prototyping with Flutter and Firebase

Tuesday, January 12th, 4:00 AM

Microsoft API Workshop

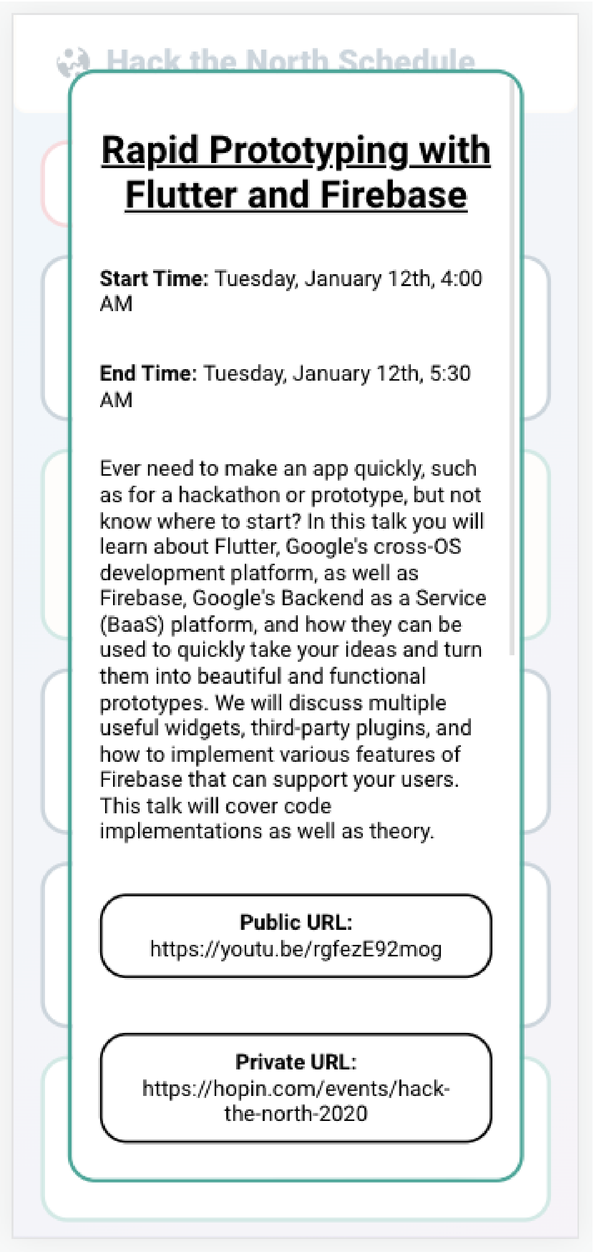
Tuesday, January 12th, 6:00 AM

Hootsuite API Workshop

Tuesday, January 12th, 6:00 AM

Intro To Computer Vision

Tuesday, January 12th, 8:00 AM



Rapid Prototyping with Flutter and Firebase

Start Time: Tuesday, January 12th, 4:00 AM

End Time: Tuesday, January 12th, 5:30 AM

Ever need to make an app quickly, such as for a hackathon or prototype, but not know where to start? In this talk you will learn about Flutter, Google's cross-OS development platform, as well as Firebase, Google's Backend as a Service (BaaS) platform, and how they can be used to quickly take your ideas and turn them into beautiful and functional prototypes. We will discuss multiple useful widgets, third-party plugins, and how to implement various features of Firebase that can support your users. This talk will cover code implementations as well as theory.

Public URL: https://youtu.be/rgfezE92mog

Private URL: https://hopin.com/events/hack-the-north-2020