Sprint Week 1

Hello all, and welcome to your first sprint week of the term. As you know, you have two options this time around:

1.) You can pitch any project you would like to me. If I approve it, you can use it as a sprint week project. Just reach out to me on teams with the following information:
    a. What are you doing? (What is your idea, how does it work, what does it do, why is it useful? Just a basic overview of your idea in general.)
    b. What is your exact plan for features you want to finish during the sprint?
2.) You can do my social media project outlined below.

In either case, you will need to submit a Figma mock-up to me of each page you intend to implement for the project. If you are proposing a project, it is important that you get this into me as soon as possible, as it will help me make sure that you guys aren't biting off more than you can chew, since I'll have a much clearer idea from the mock-up of what you're actually trying to do.

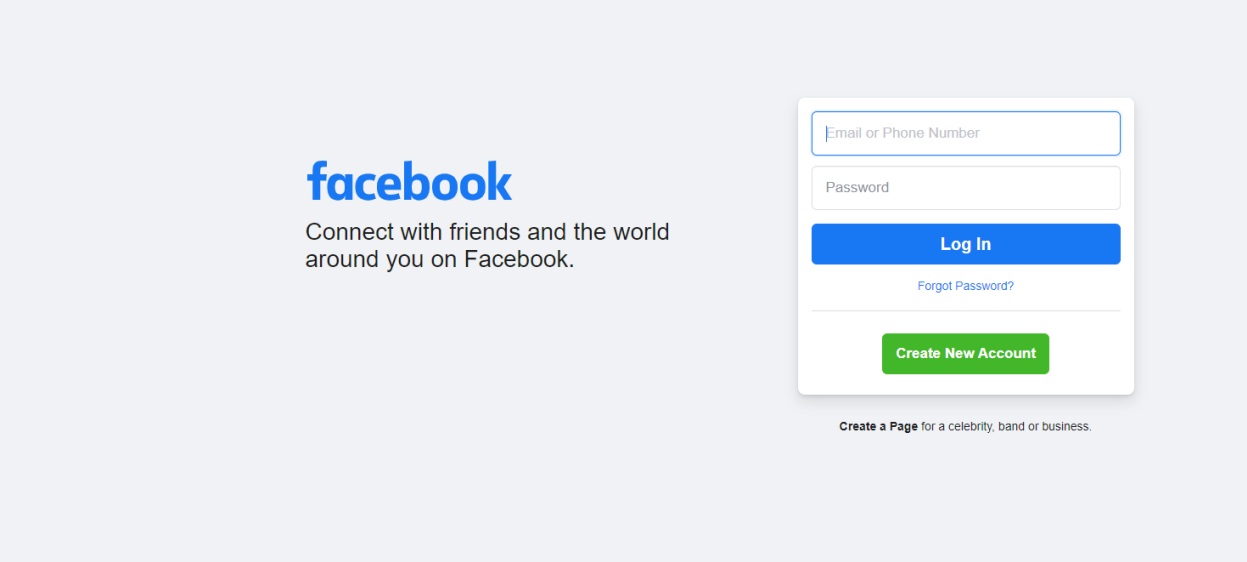
The next section of this document will discuss the details of the social media project but check out the very end of the document for a fun bonus idea for either type of project.

Social Media Project

For this project, you are going to need to create four different linked webpages. It should have a homepage, a login page, a signup page, and the social media feed itself.

Home Page

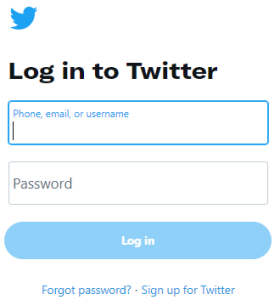
The home page should basically have links for you to sign up for the website, or to sign into your account. Consider the Facebook home page:



It is quite simple, just some basic information, and then it has a link to create a new account, and a place to login. If you want, you can have your login and sign-up on two separate pages, or you can combine the login and the homepage like Facebook does here. Likewise, if you wanted to combine the homepage and the sign-up page, that is okay too.

Login Page

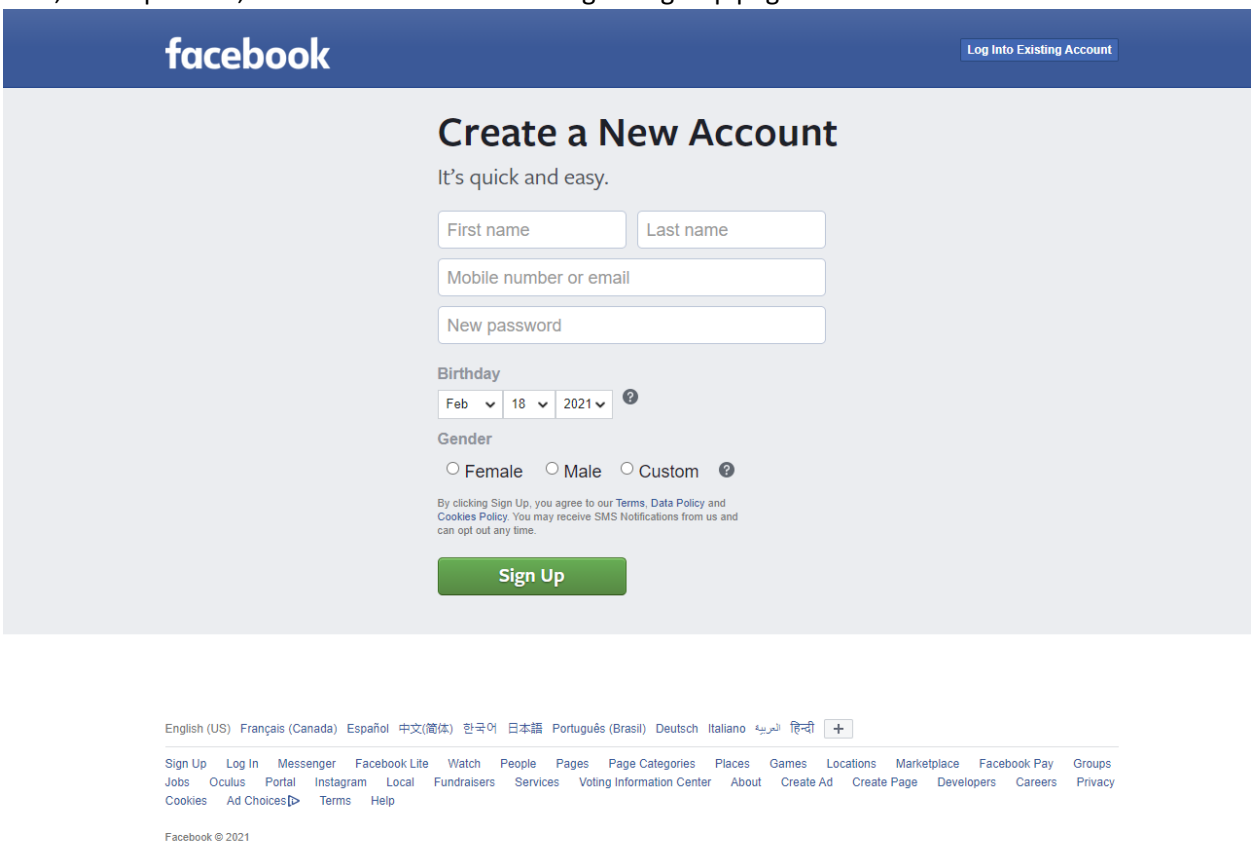
This is just a fake login page, no matter what username and password you enter, it should take you straight to the social media feed page when you click “login.” You should have a link to the sign-up page on the login page as well. You can keep it simple if you’d like, consider twitter’s login page for example:

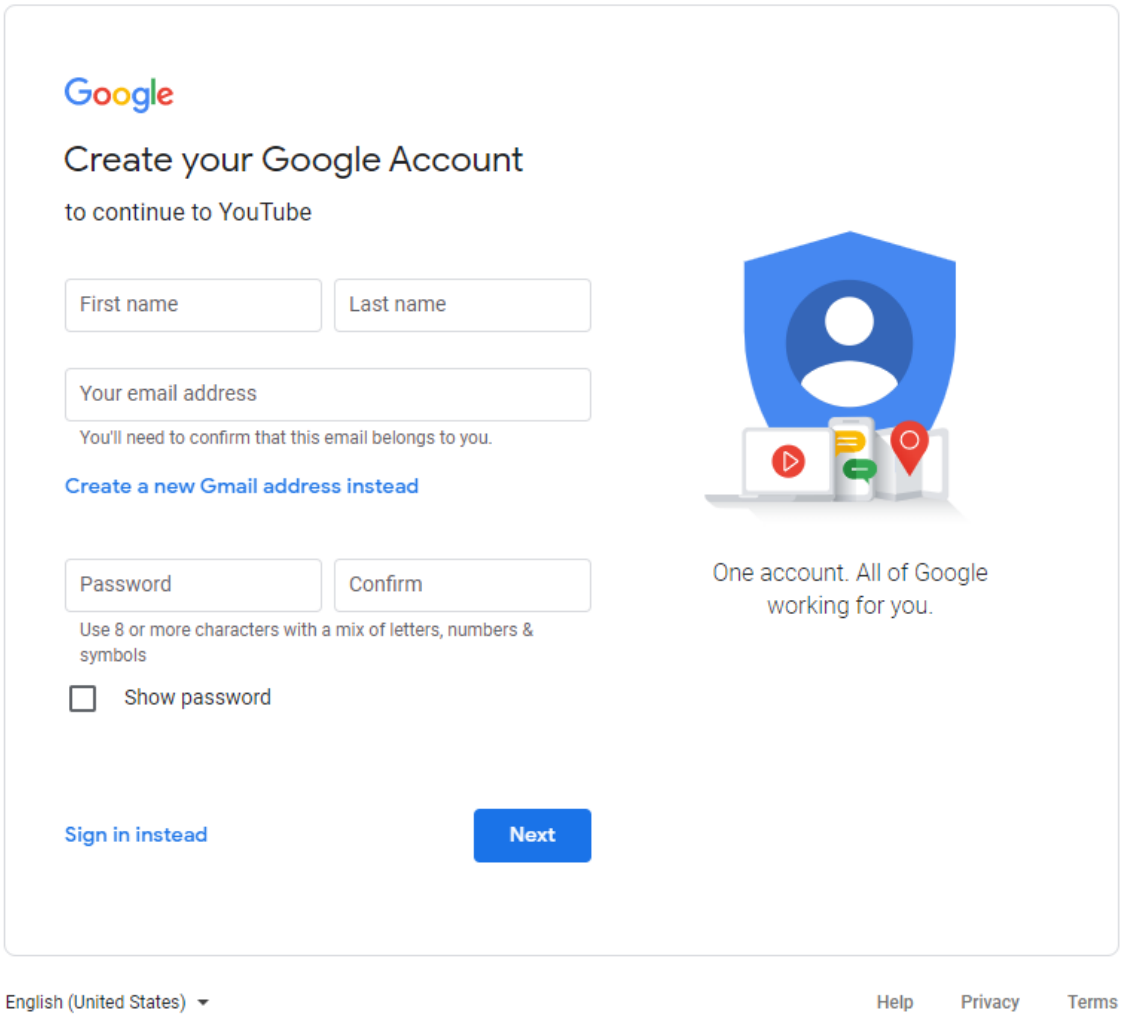


Sign-up Page

The signup page is the first page that has some basic features on it. Again, you can have it accept a bunch of fake information and when you click “create account” it should just take you to the feed page, but on this sign-up page, you should do some validation with JavaScript, which we will talk about below.

First, for inspiration, consider Facebook and Google's sign-up pages:



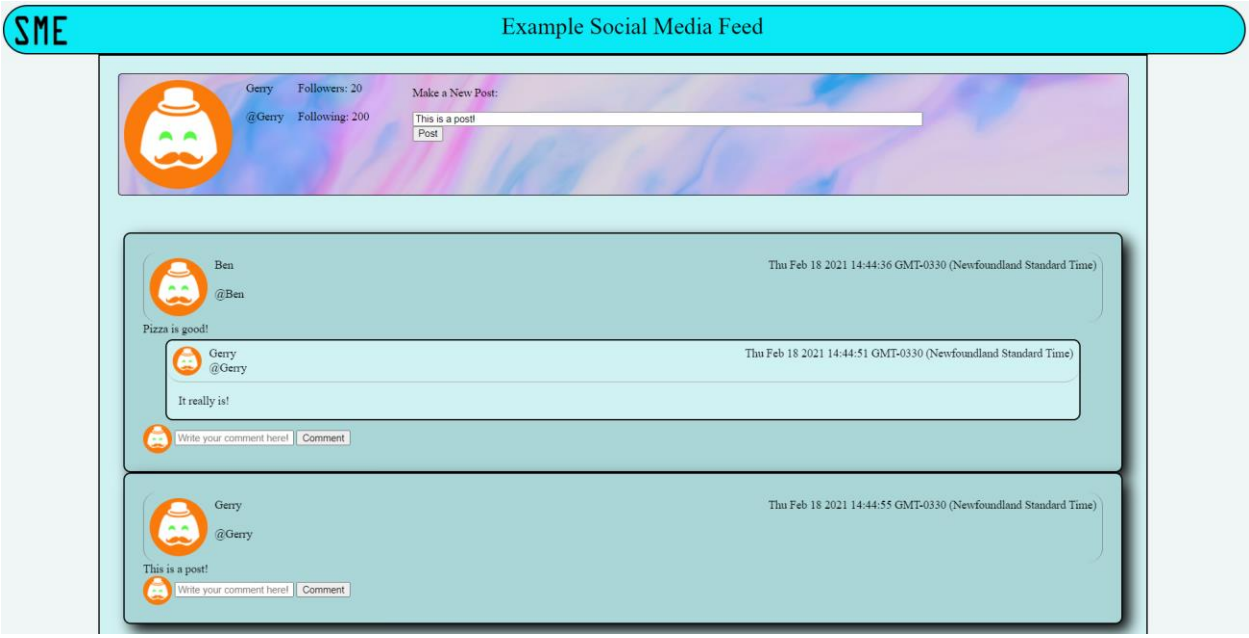


Your sign-up page should ask for, at a minimum, a username and/or email address, a birthday (with drop-down boxes, as shown in the Facebook picture above), a password, and a box to confirm the password. Feel free to add other types of information using, for example, radio buttons like on Facebook, or anything else that you desire. Again, this information will all be ignored, but you should do some basic validation on the sign-up page. Make sure that the password matches what is in the confirm box and ensure that the entire sign-up form actually has some data in each field before allowing the user to continue (for example, make sure they don't leave the username box empty.) If the user has an invalid entry (like if their passwords don't match) you should output an error message to the screen somewhere prompting them to change it.

Your sign-up page should also have a link to your login page somewhere on it (notice Facebook and google both have this option, for google they have "Sign in instead" and Facebook has "Log into an existing account.")

Feed Page

This is the main part of the project, implementing a social media feed of some kind. This could be a feed that has posts like Facebook or twitter, it could be something that lets you post images or videos like Instagram or TikTok, or basically any idea that you would like. Here is an example feed that I showed you in class which uses simple text posts.



The main functionality you want to implement here is as follows: When you type out some text in the input area and hit "post", it creates a new JavaScript element with your post available to be seen in the feed. Importantly, it should also save all the post data in Local Storage such that when you refresh the page, the posts stay showing, and even if you close out of the browser and come back, the posts will still be there. The relevant data, in the case of the example above, was the content of the post (the words that you said on the post), the comments attached to the post and all their associated content, the date of the post, and the name of the person who made the post. To be clear, this means that you will have to re-create the HTML for the posts each time when you load the page by reading the saved post data out of local storage and making the appropriate elements. This process is tricky, so if you need some pointers, be sure to let me know. Put simply, if you can make it so you can write a post on the page, and then close out of the browser and come back to it later and the post was saved and shows up again in the feed, you have succeeded. If you need a refresher on how this all should work from a user perspective, re-watch the meeting recording for February 18th 2021 at 1:00pm.

If you do a text-based posting system like this, I encourage you to add a comment system like the one I have shown above, since this is conventional on websites you would probably like to replicate, like Facebook and Twitter. Be sure to save the comments and restore those too, just like the posts.

Bonus Idea

As you guys know, we can style the same HTML with two different CSS stylesheets. There is a lot that we could potentially do with this. For the first bonus idea, if you do a social media replication study, you could have a drop-down that lets you switch styles between two different types of replicated feeds. For example, one style that is a Facebook clone, and one that is a twitter clone. This is how things like WordPress themes work, for example.

A more generally applicable version of this, though, would simply be to implement a dark-mode and a light-mode on your website, and allow the user to switch between the two via some mechanism on the site (like a drop-down box, or radio buttons, or whatever). Give it a try! If you need help switching between styles, let me know, and I can give you some tips on possible approaches to this.