ACM SIGITE 2023 Workshop

Full-Stack Development – Mobile App

October 13, 2023

Developed by Sam Chung, Ph.D.

**HOS01: Cloud Development Environment**

School of Technology & Computing (STC)

City University of Seattle (CityU)

**Overview:**

React Native is a framework for building mobile applications using JavaScript and React. In this guide, we will walk through the process of setting up a React Native development environment with GitHub Codespaces, Expo, and Expo Go. The guide includes step-by-step instructions on how to set up GitHub Codespaces, install Expo Go, and create a basic React Native app. This guide is designed for students who are new to React Native development and want to learn how to use Expo Go and GitHub Codespaces to create mobile applications.

**Before You Start**

* Screenshots may be different from your environment.
* The directory path shown in screenshots may be different from yours.
* Version numbers may not match with the most current version at the time of writing. If given the option to choose between stable release (long-term support) or most recent, please select the **stable release** rather than the beta-testing version.
* There might be subtle discrepancies along with the steps. Please **use your best judgment** while going through this cookbook-style tutorial to complete each step.
* If you are not familiar with terminal, command line, and bash scripts, check out this video: <https://youtu.be/Dp7uw9c6QH8>
* All the steps and concepts in this tutorial are from the textbook, so if you encounter problems in this tutorial, please **try to read and compare the textbook to solve the problem**. If you still can’t solve the problem, please feel free to contact your course TA.
* **Avoid copy-pasting code from the book or the GitHub repository**. Instead, type out the code yourself. Resort to copy-pasting only when you are stuck and find that things are not working as expected.
* Some steps may not be explained in detail. If you are not sure what to do:

1. Consult the resources from the course.
2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

* Section 1: Accessing GitHub Codespaces.
* Section 2: Setting Up Expo account and Downloading Expo Go on my Smartphone.
* Section 3: Creating my first mobile app.
* Section 4: Updating my first mobile app.
* Section 5: Pushing your work to GitHub.

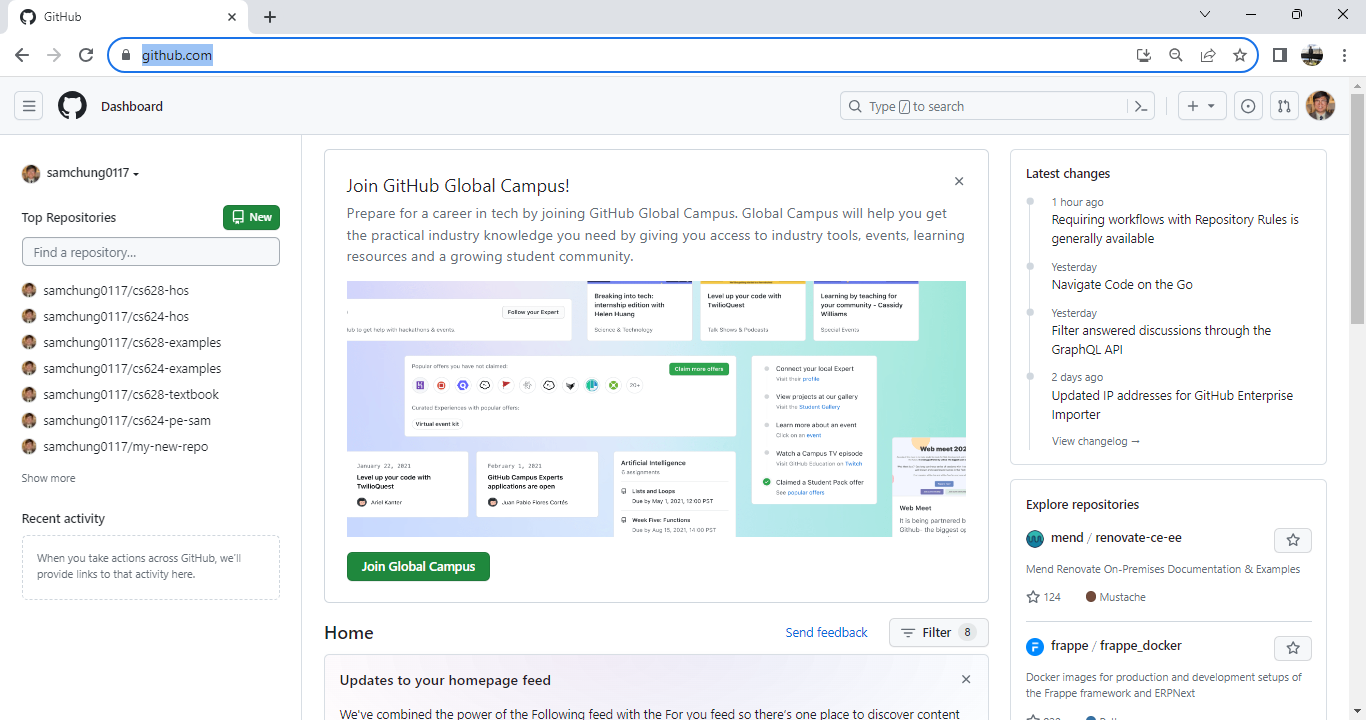
**Mobile App Development Practices**

* Visit [ACM SIGITE 2023 Examples](https://github.com/samchung0117/sigite2023-example).
  + Visit the README.md file.
  + Find examples for your practices.
* Section 6: Using <Text> and <View> components.
* Section 7: Creating a function component.
* Section 8: Creating a class component.
* Section 9: Using <ScrollView>, <Image>, and <TextInput> components.

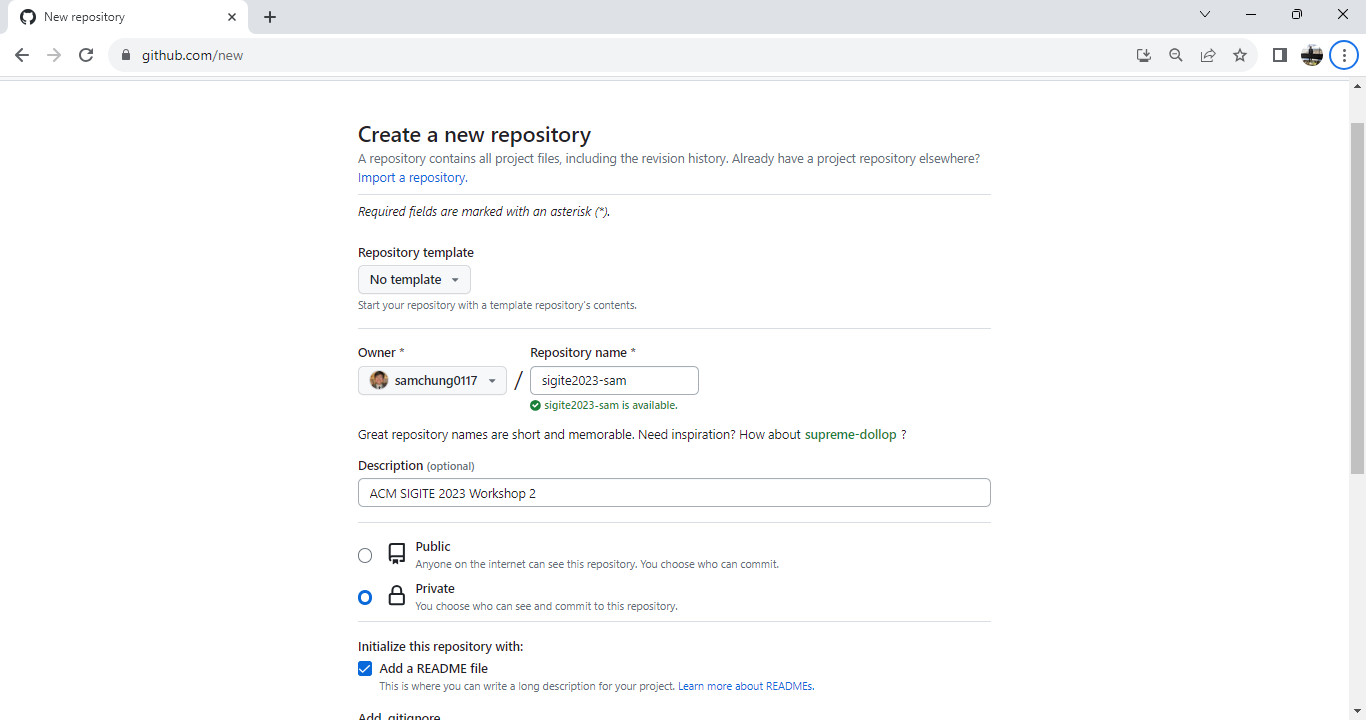
**Section 1: Accessing GitHub Codespaces.**

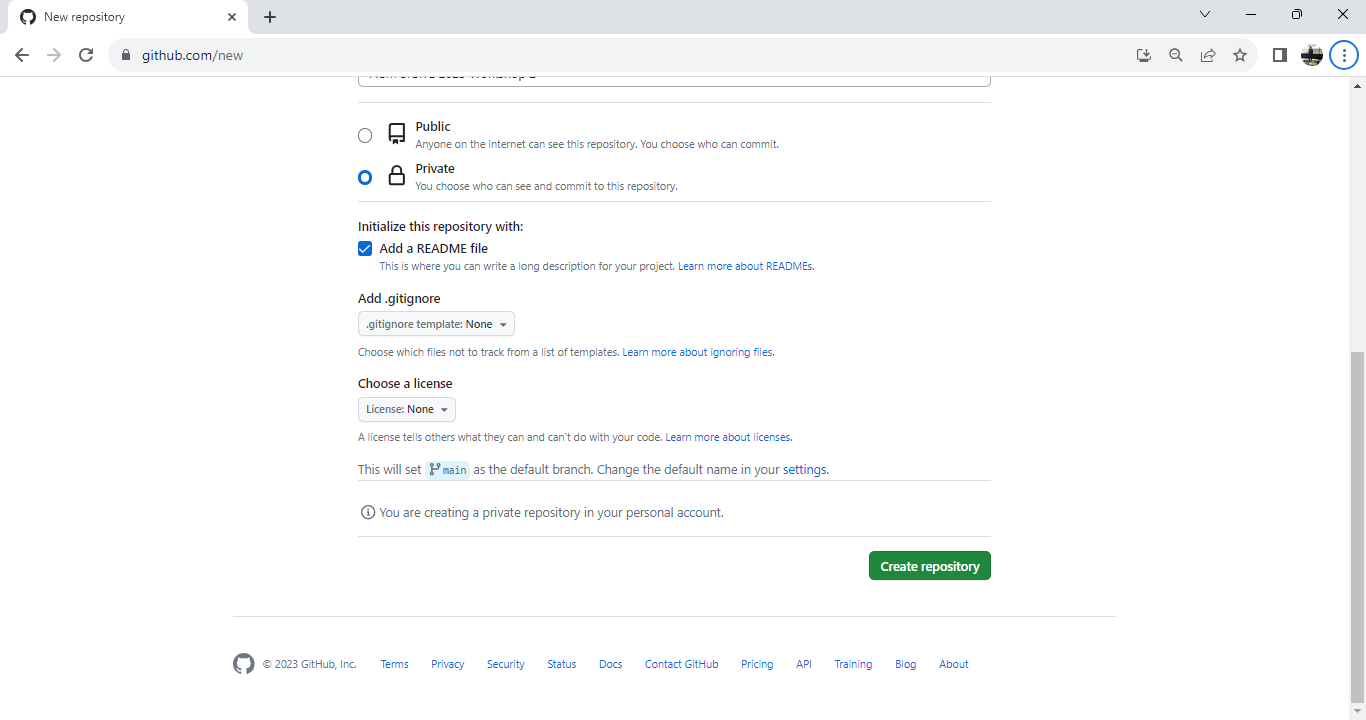
GitHub Codespaces are online cloud-based development environments that allow you to easily write, run, and debug your code. It is fully integrated with your GitHub repository and provides a seamless experience for developers. It would help if you had a GitHub account and an active internet connection to access the Codespaces environment.

1. Visit your GitHub.



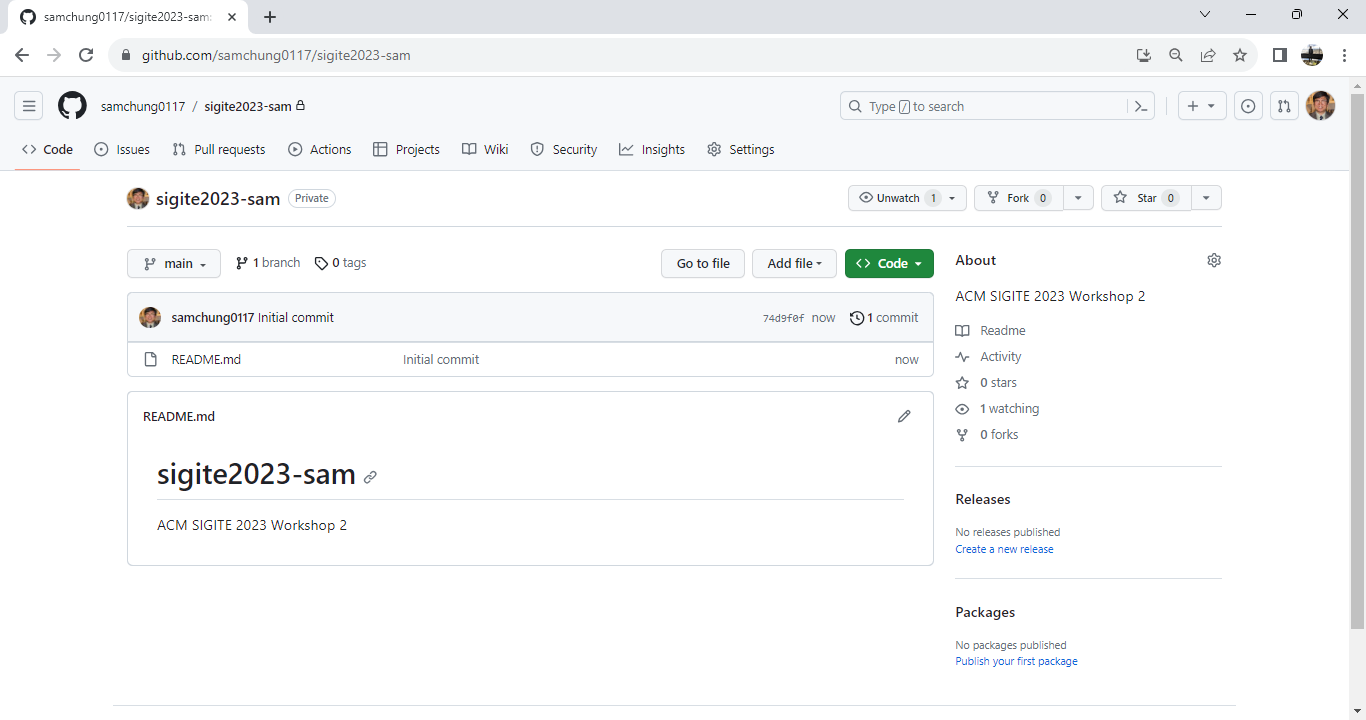
1. Create a new repository like “sigite2023-your\_name.”
   1. Repository name: “sigite2023-your\_name”
   2. Description: “ACM SIGITE 2023 Workshop 2”
   3. Type: Private
   4. Initialize this repository with “Add a README file.”
   5. Press the button “Create repository.”

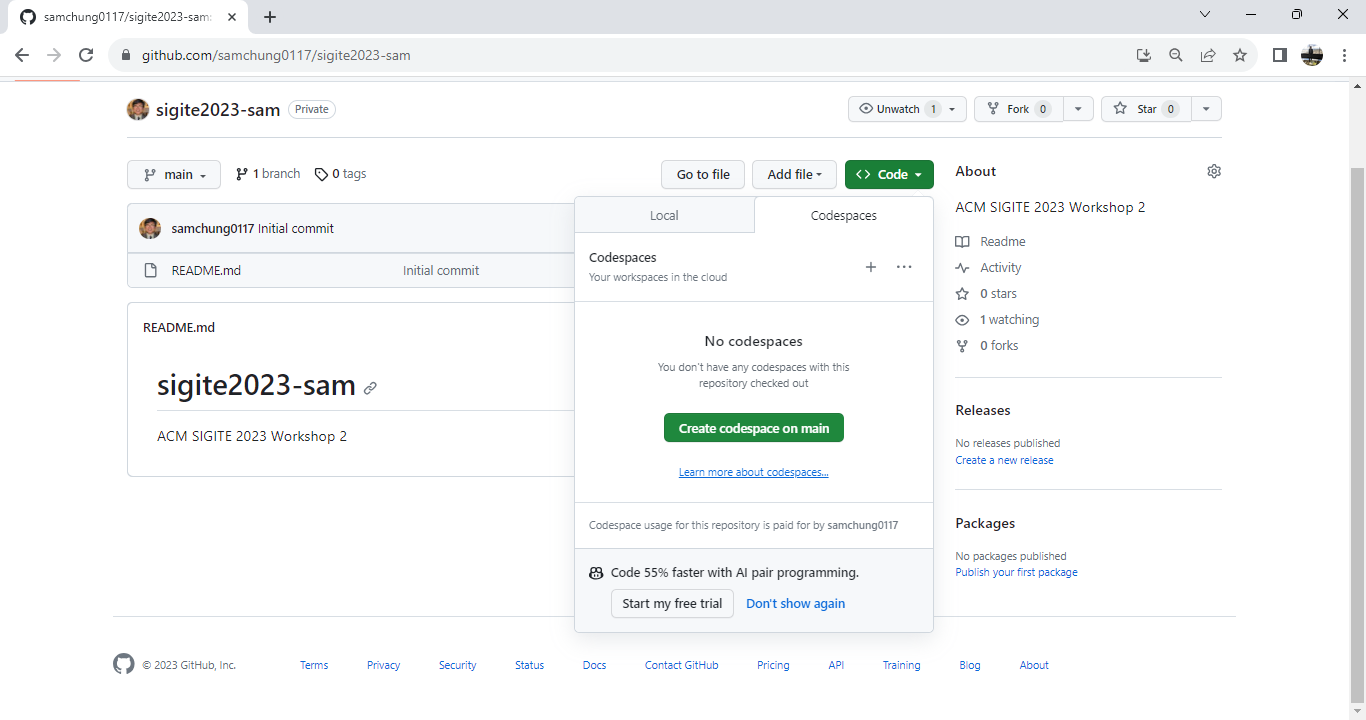




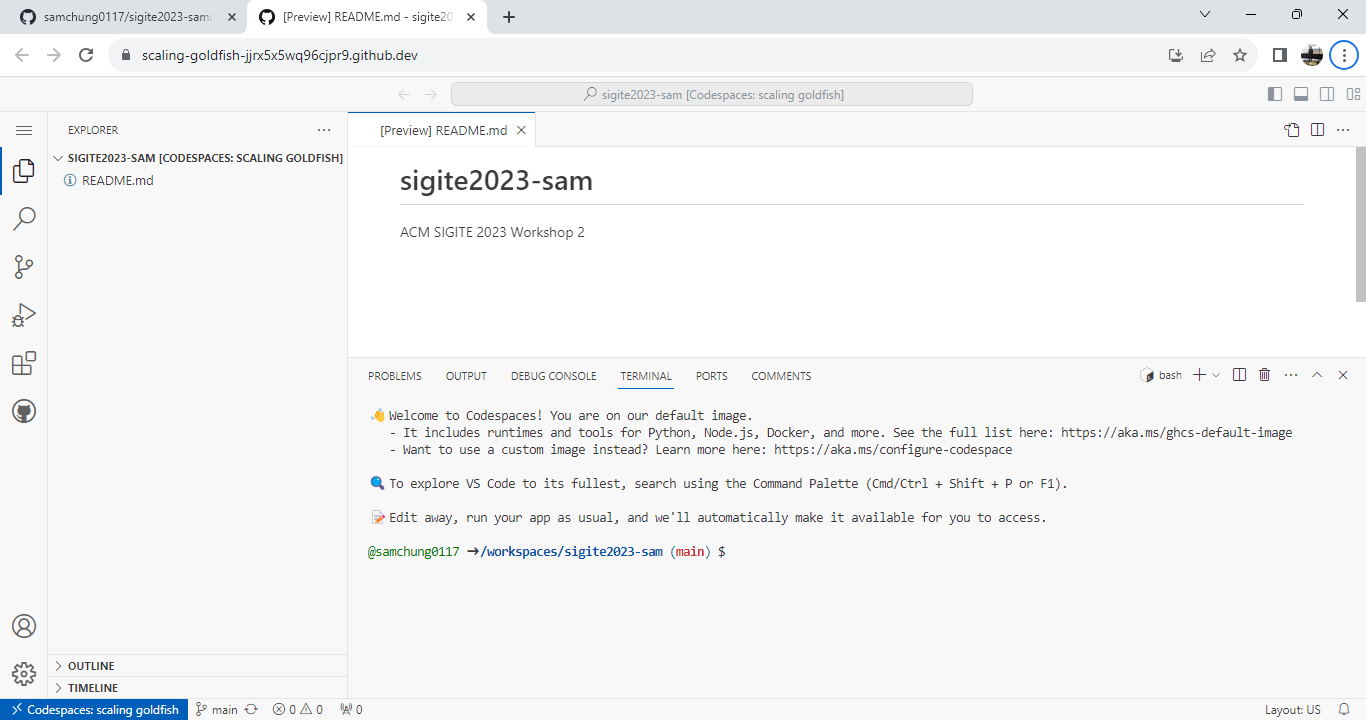
1. In the top-right corner, click on the “<>Code” drop-down menu.

Select "Create a codespace on main."





1. Wait for the Codespaces environment to load. Once loaded, you can access the terminal, file explorer, and other tools to start working on your project.  
     
   Once your codespace is created, the template repository will be automatically cloned into it. Now you can run the application and launch it in a browser which we call cloud IDE (Integrated Development Environment). We use Visual Studio Code for the web.

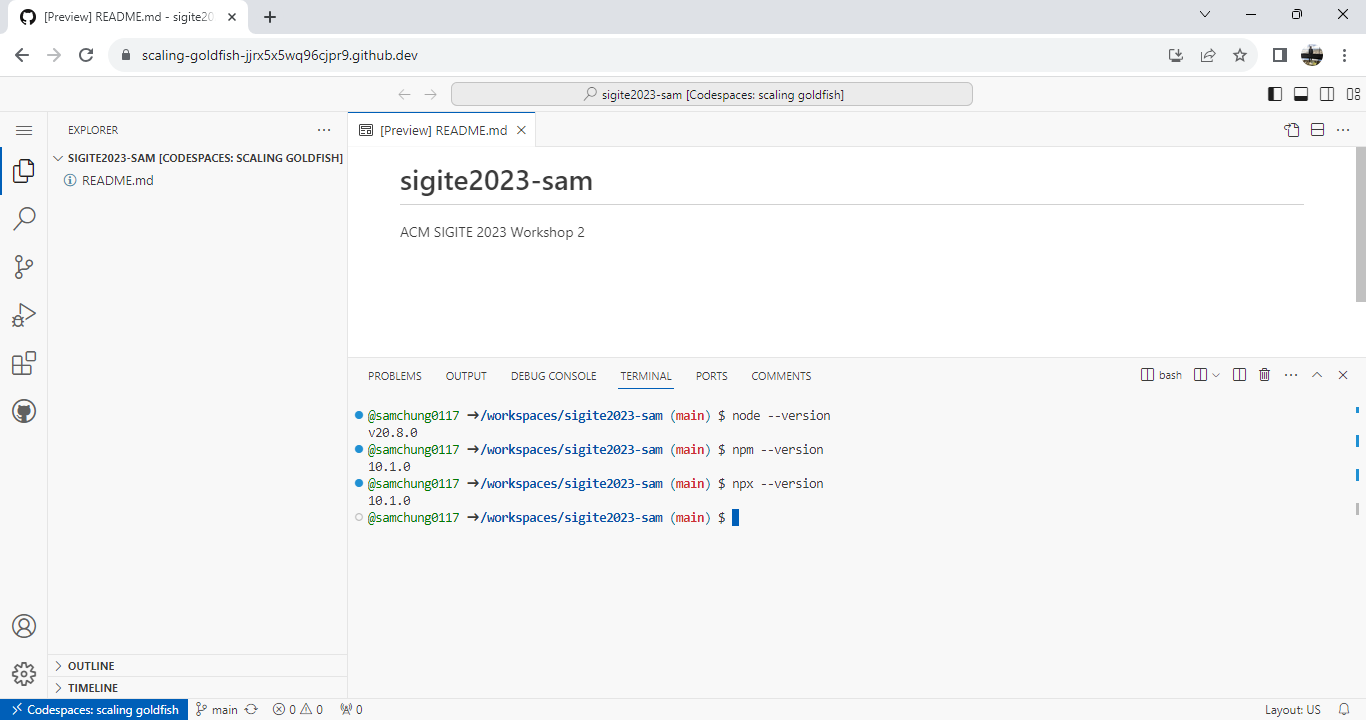


1. Open the terminal if you cannot see the terminal. (Press “~” and “ctl” together.)
2. Test the Node environment – node, npm, and npx  
   Assume that you are under the current working directory for HOSs.  
     
   “By default, the container for your codespace has many languages and runtimes, including Node.js, JavaScript, and Typescript. It also includes a common set of tools, such as nvm, npm, yarn, git, wget, rsync, openssh, and nano.” (<https://docs.github.com/en/codespaces/setting-up-your-project-for-codespaces/adding-a-dev-container-configuration/setting-up-your-nodejs-project-for-codespaces>)

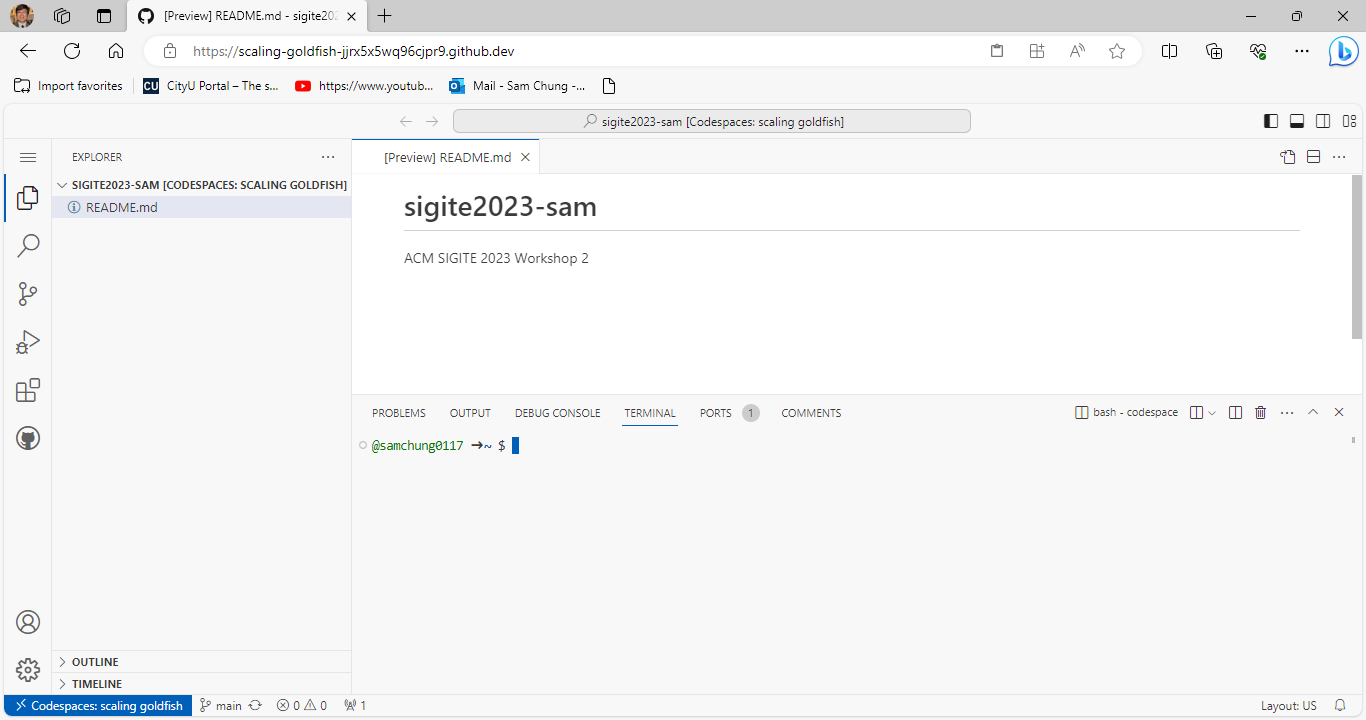
“Node or Node.js is an asynchronous, event-driven, open-source, cross-platform JavaScript runtime environment.” (<https://nodejs.org/en/>)

“The command npm is used to download JavaScript packages from Node Package Manager, and npx is used to execute JavaScript packages downloaded this way.” (<https://www.geeksforgeeks.org/what-are-the-differences-between-npm-and-npx/>)

By typing the following version check commands, you can confirm that your Codesapces have node, npm, and npx.



1. Clear your screen by typing the “clear” Bash shell command.

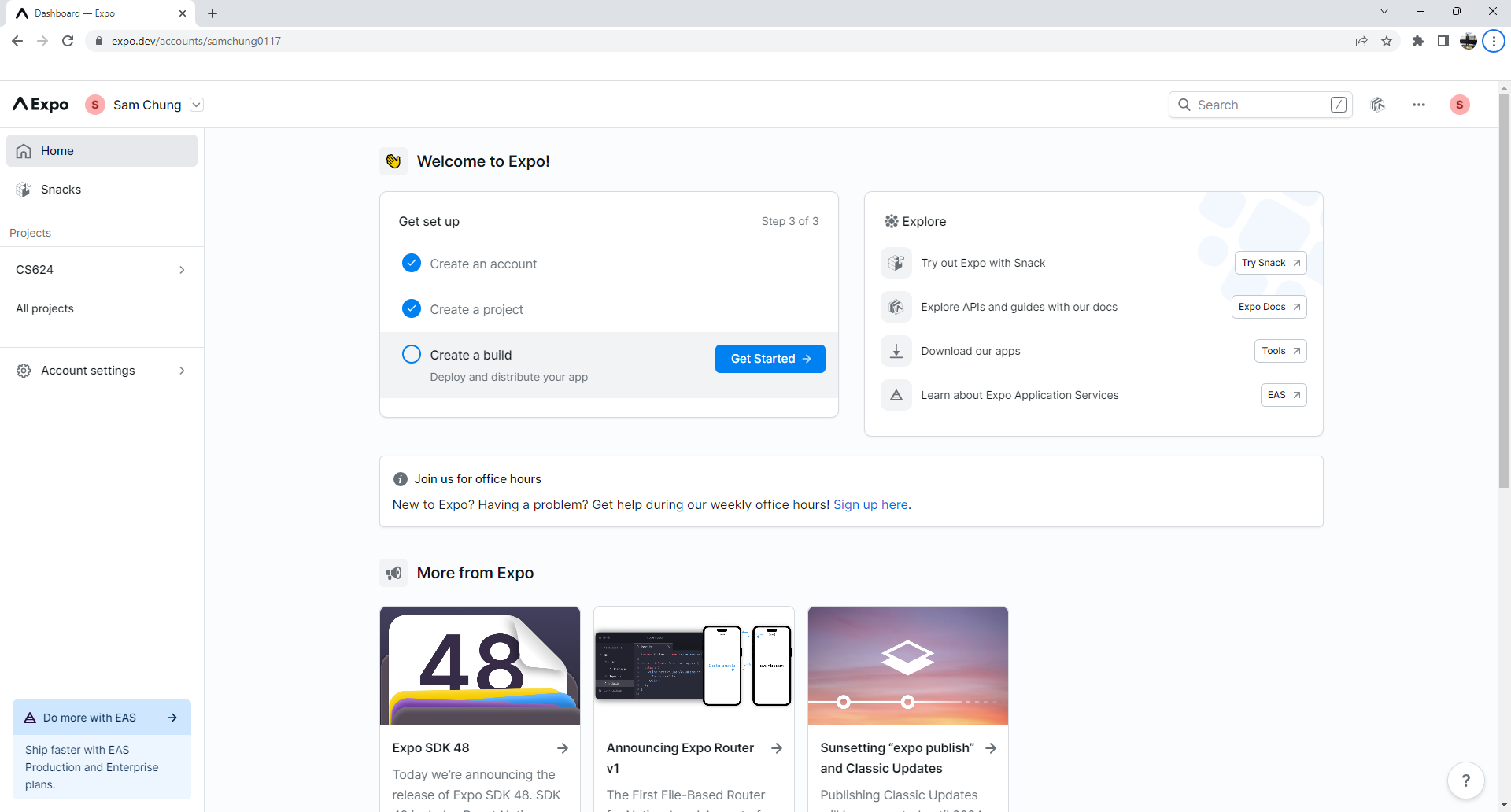


1. **CHALLENGE**: What is the purpose of this section? i.e., why did you practice this section?

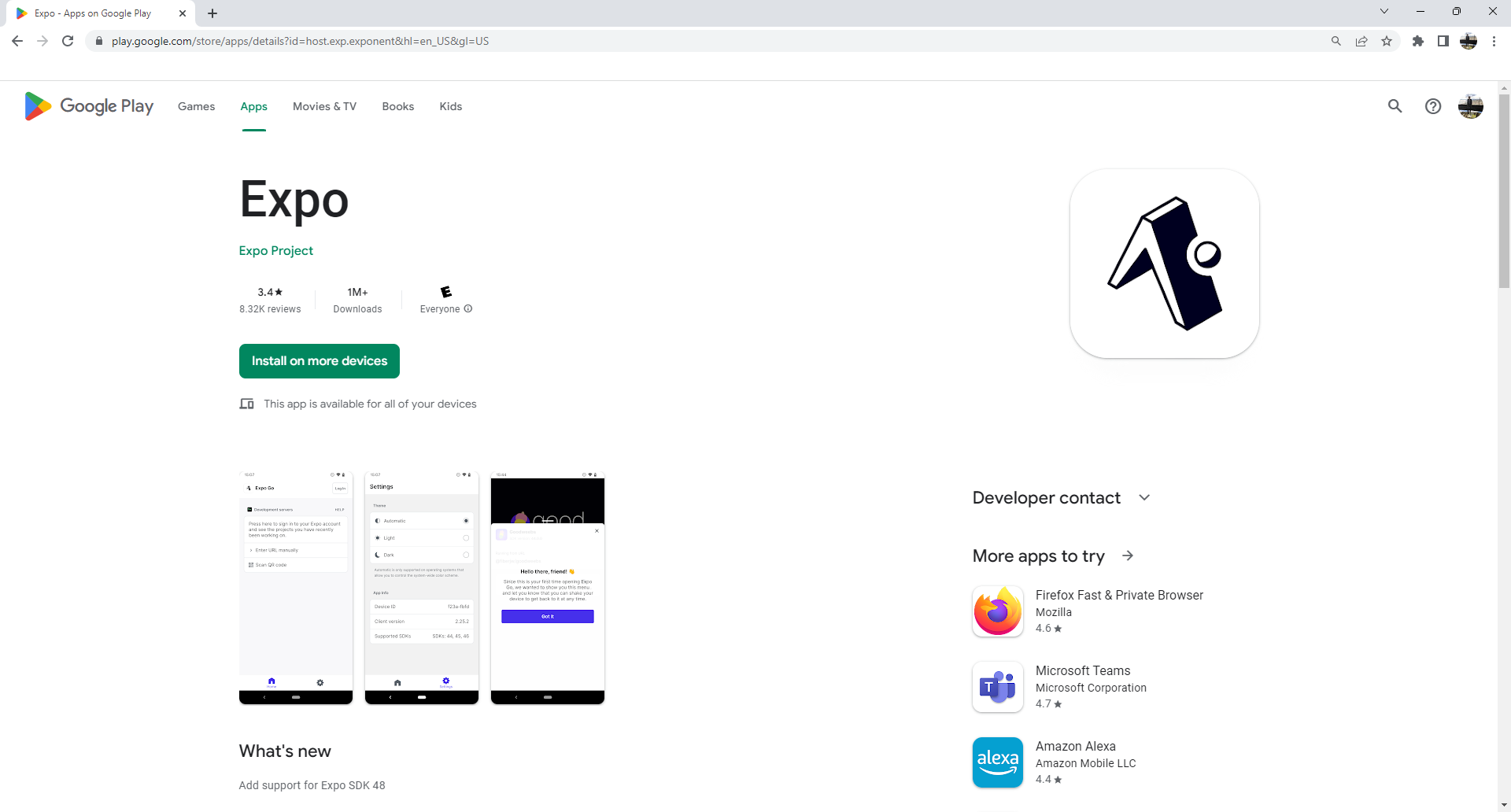
**Section 2: Setting Up Expo account and Downloading Expo Go on my Smartphone.**

1. Creating my Expo account  
   “Expo is an open-source platform for making universal native apps for Android, iOS, and the web with JavaScript and React.” (<https://docs.expo.dev/introduction/faq/> )

If you do not have an Expo account, sign up for an account in Expo [here.](https://expo.dev/)   
**Remember the credentials. We need them for all the exercises.**   
  
If you log into your Expo account, you can see a screen like the one below.



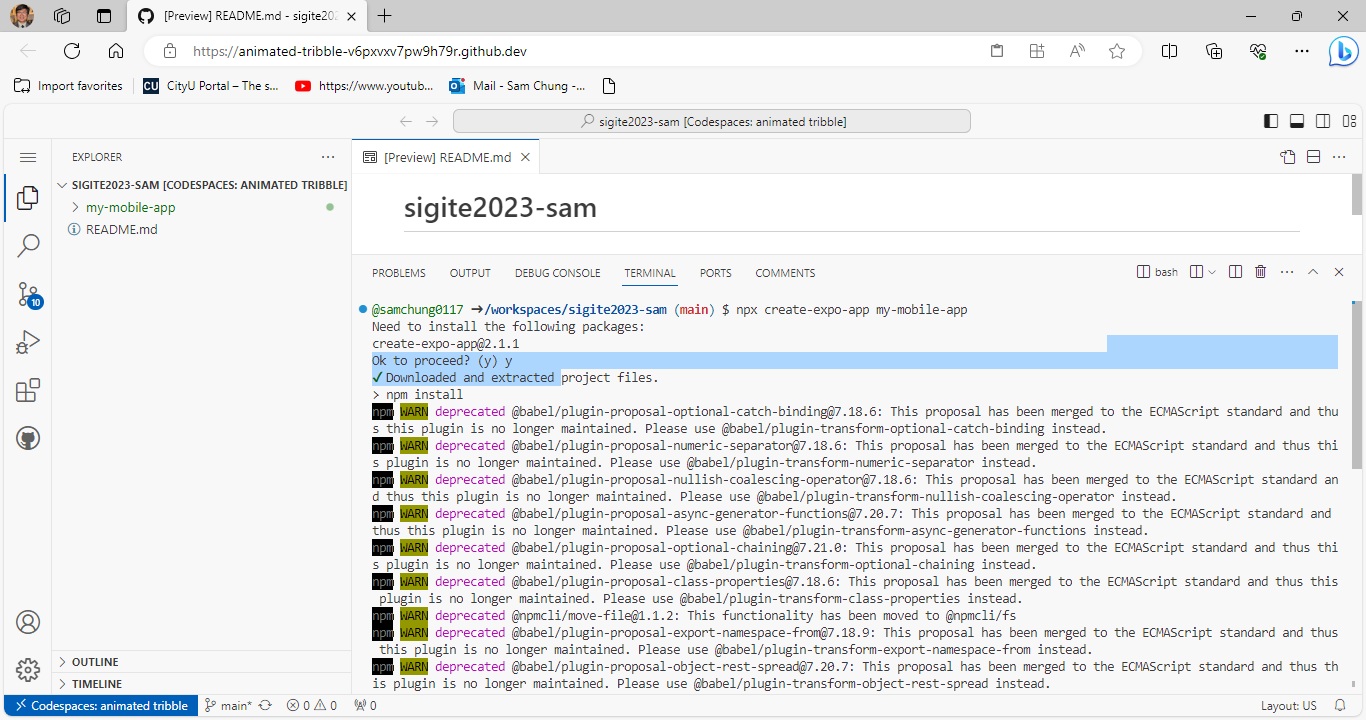
1. Downloading Expo Go on my smartphone  
   “Expo Go is a free, open-source client for testing React Native apps on Android and iOS without needing to build anything locally.” (<https://docs.expo.dev/workflow/expo-go/>)  
     
   Download the "Expo Go" app from the App Store or Google Play Store.



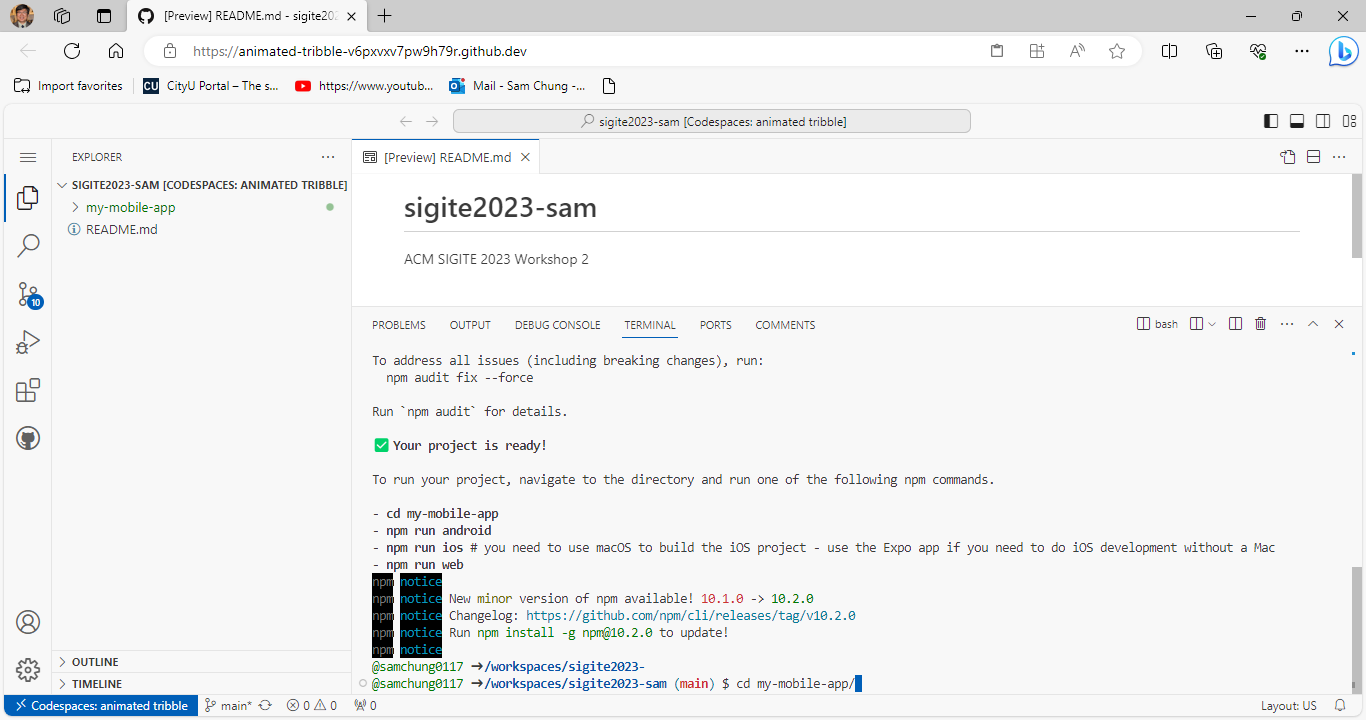
1. **CHALLENGE**: What is the purpose of this section? i.e., why did you practice this section?

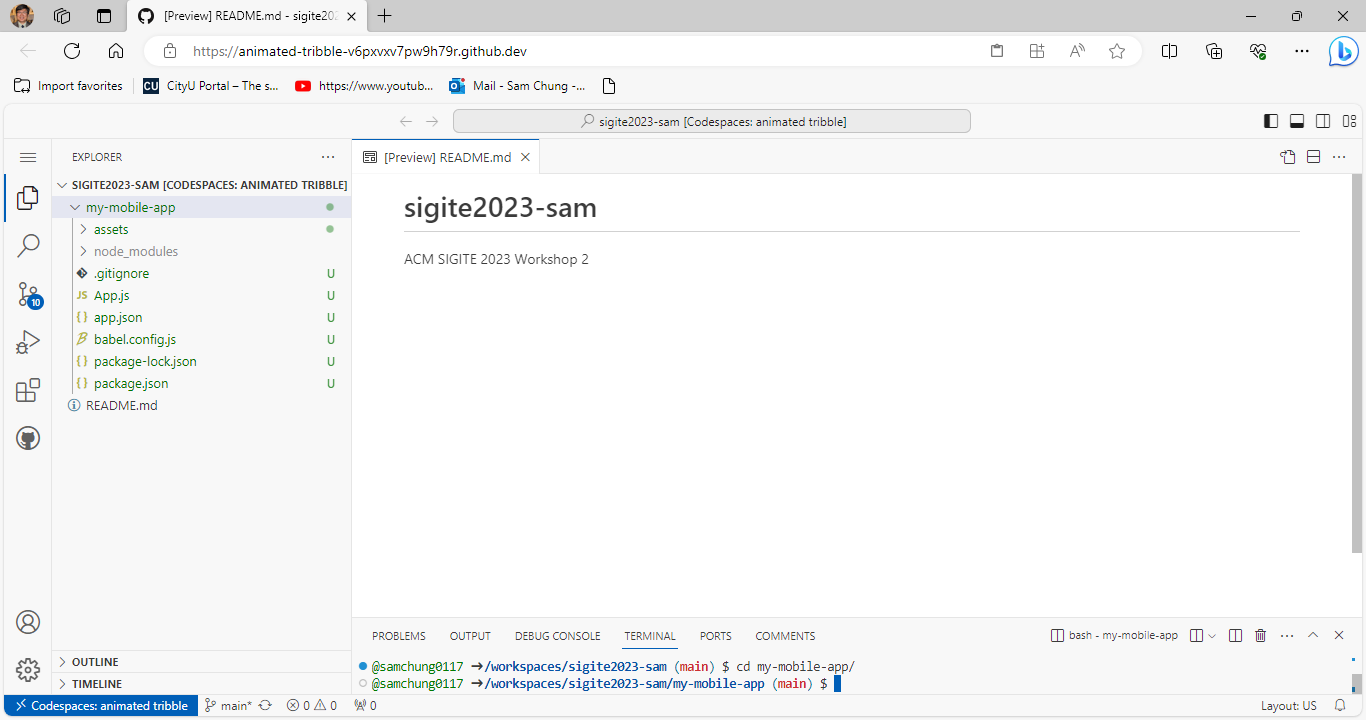
**Section 3: Creating my first mobile app.**

1. Run the command “**npx create-expo-app my-mobile-app”** to create a new Expo project.  
   It requires to install the following package: create-expo-app@2.1.1

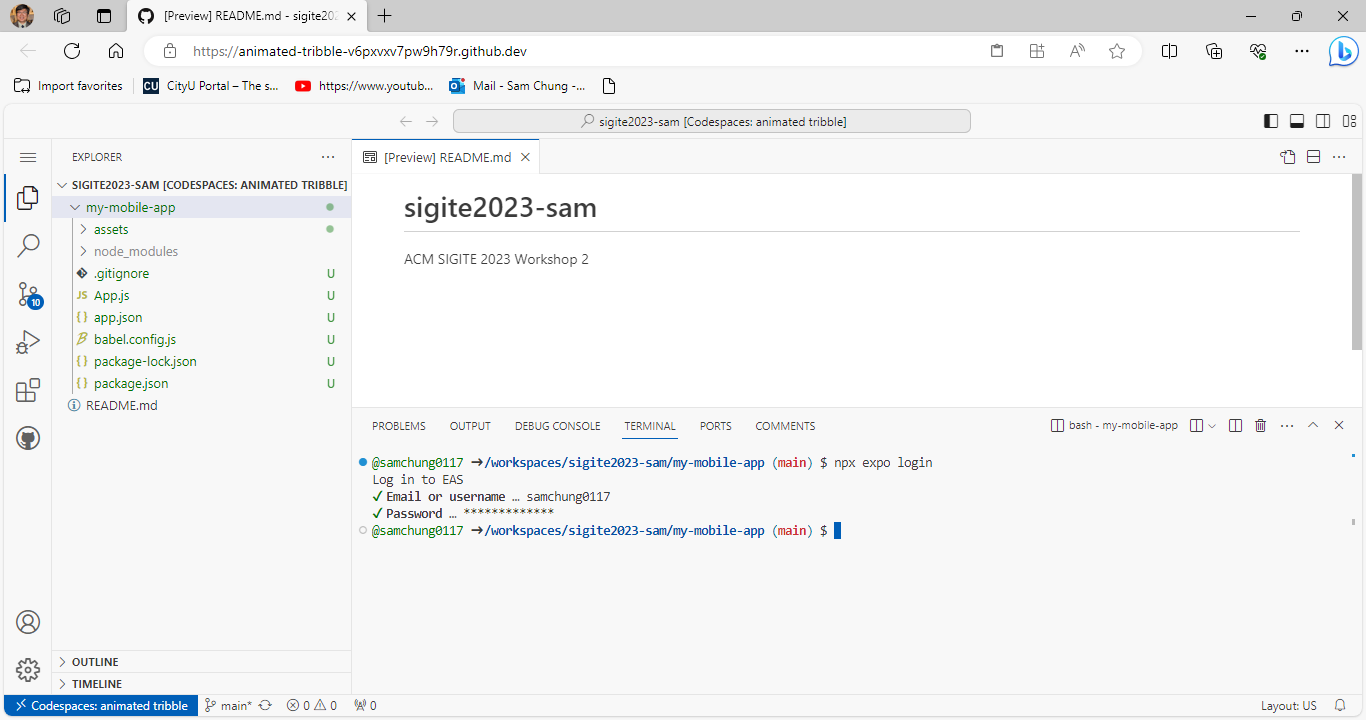


1. Navigate to the project directory.  
   **cd my-mobile-app**

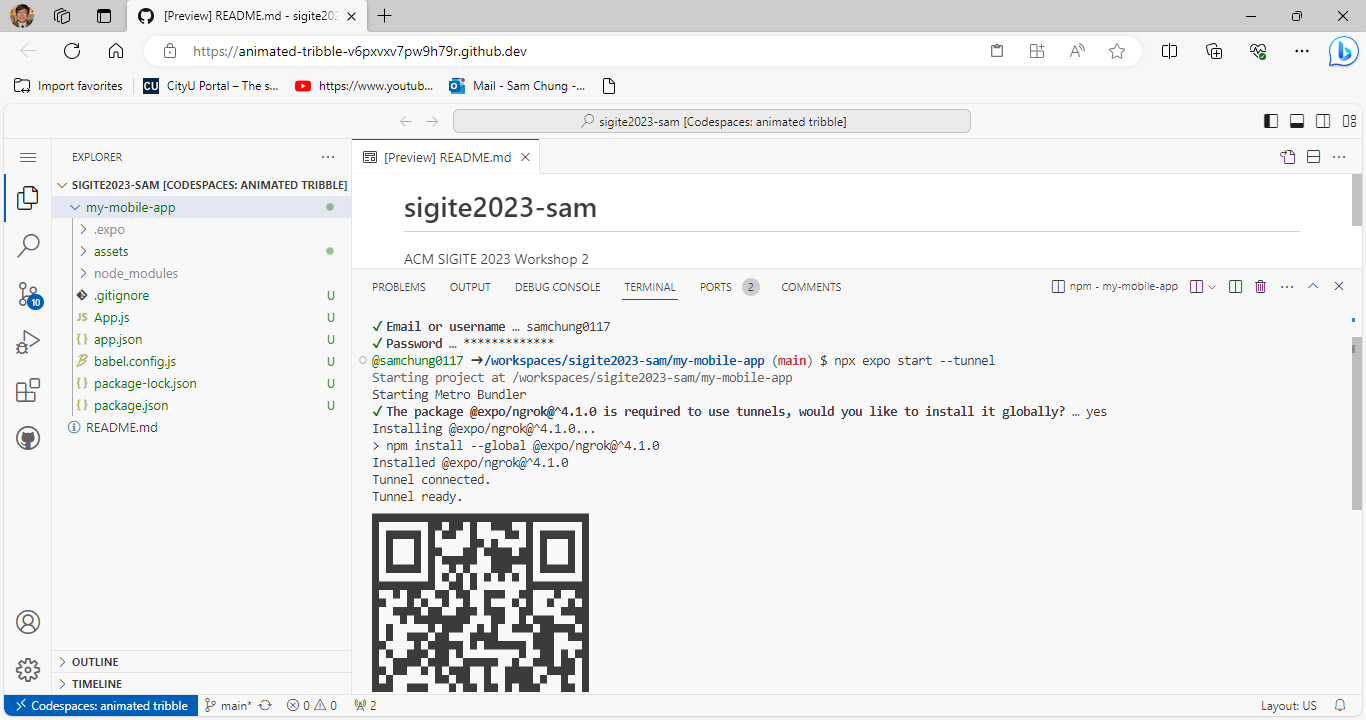




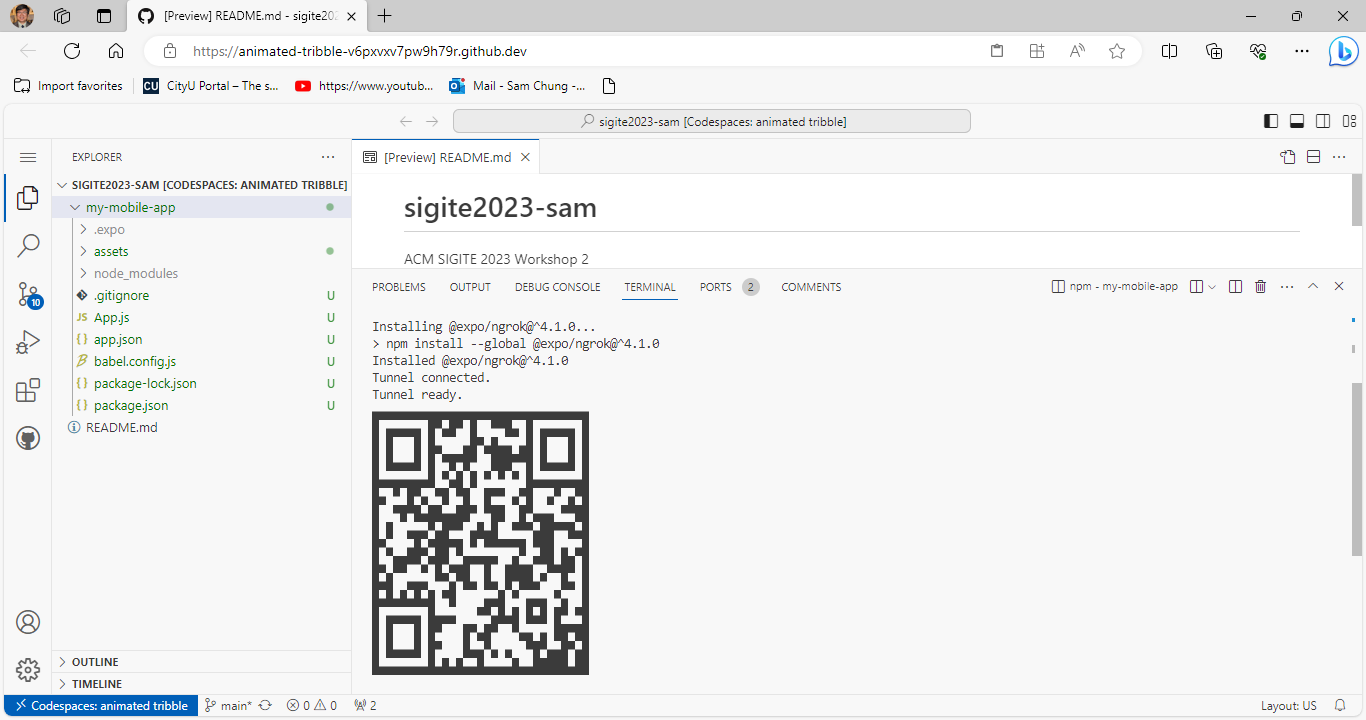
1. Check whether you logged into your Expo account to use Expo Go on your smartphone.  
   In the terminal, log into Expo using the command “**npx** **expo login**.”  
   When prompted, enter your credentials if you did not log in already.



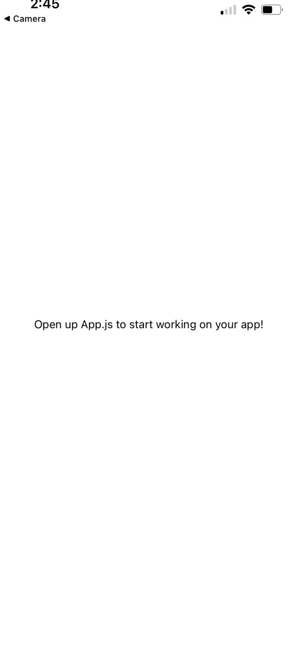
1. Run the command “**npx expo start –tunnel”** to start the development server.  
   **npx expo start --tunnel**



1. Preview the app in the Expo Go client app on your mobile device by scanning the QR code with Expo Go App (Android) or the Camera app (iOS).



1. Each smartphone provides a feature to take a screenshot.  
   The captured screenshot is shown below.

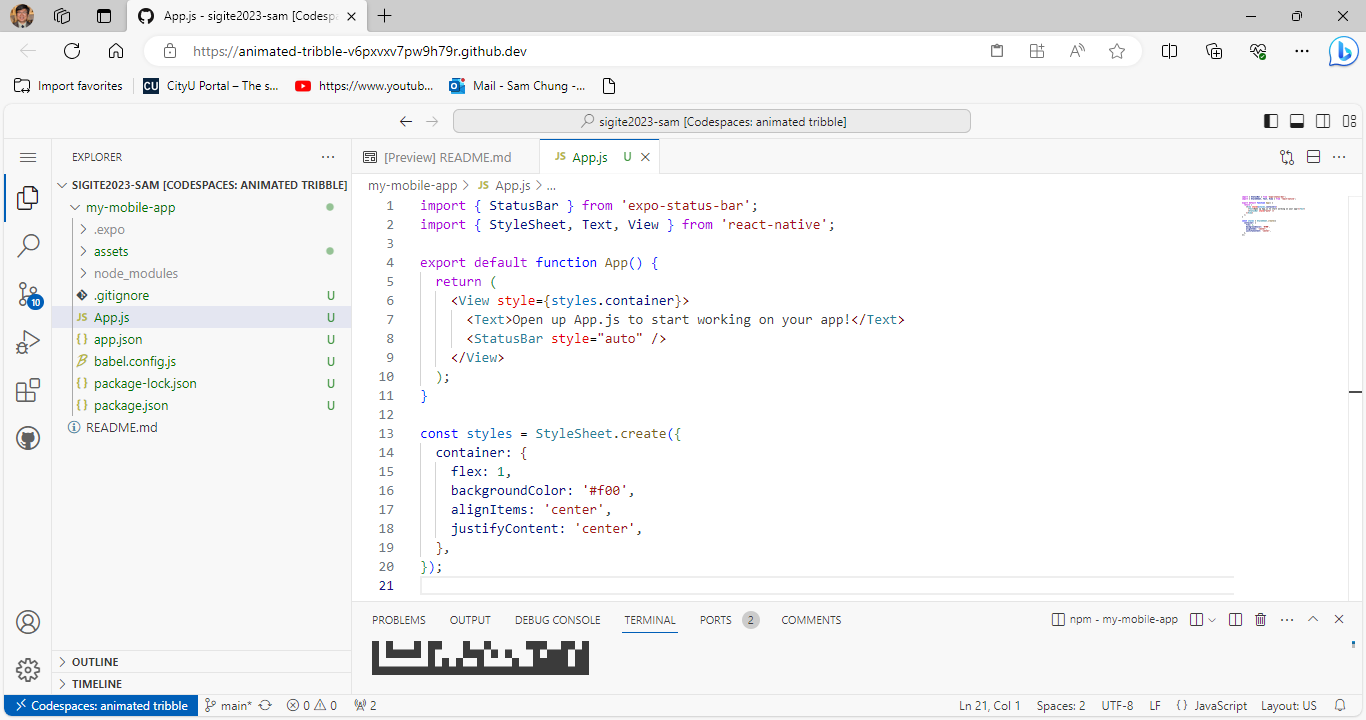


1. Use ctrl + C to stop the server.
2. Close the Expo Go app on your mobile device.
3. **CHALLENGE**: What is the purpose of this section? i.e., why did you practice this section?

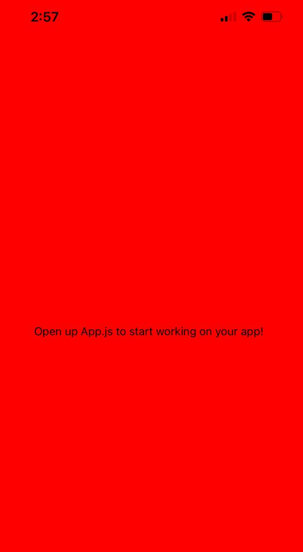
**Section 4: Updating my first mobile app.**

Steps to change the background color of the app:

1. Open the **app.js** file in the GitHub Codespaces environment.
2. In the **styles** object, locate the **container** property.
3. Change the **backgroundColor** value to your desired color (e.g., **'#** **f00’**).



1. If the server is running, you can see the update immediately.



Steps to verify the changes:

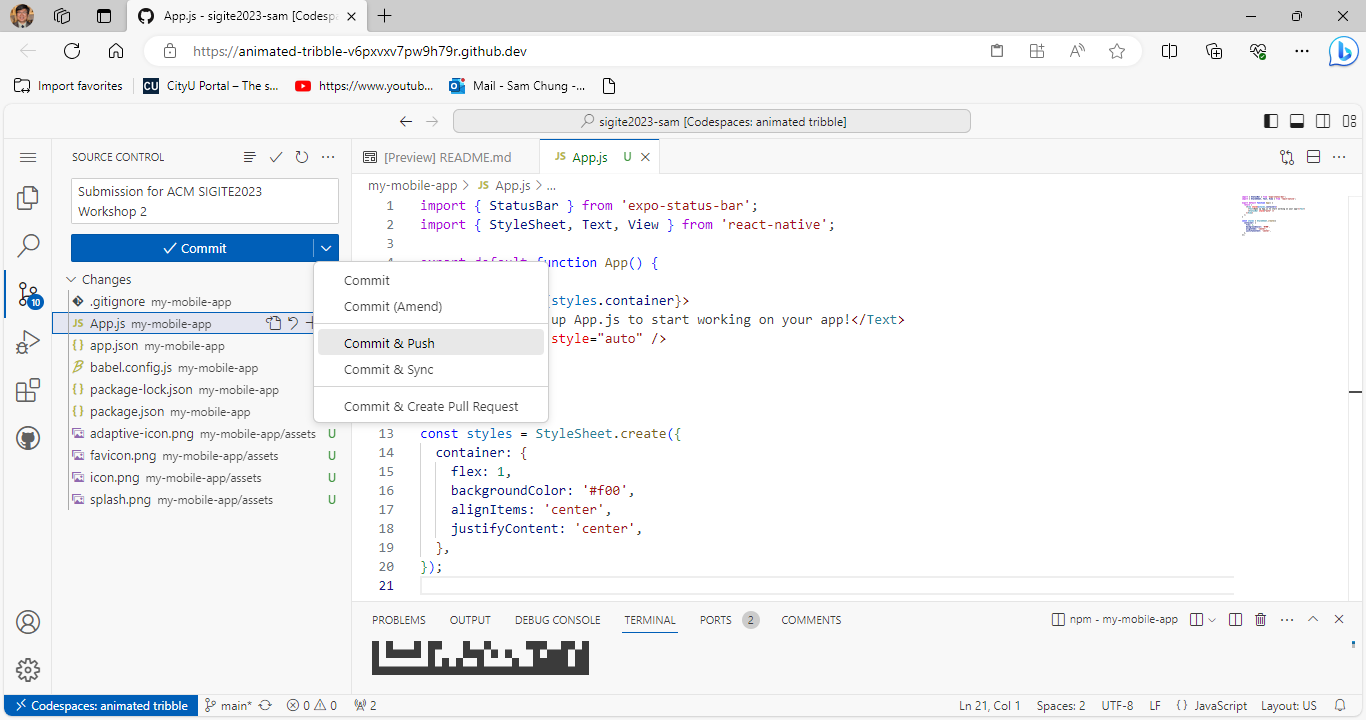
1. Open the terminal in the GitHub Codespaces environment.
2. Type “**npx** **expo start –tunnel”** and press Enter to start the expo development server.
3. Wait for the development server to load and show the QR code.
4. Open the "Expo Go" app from your mobile device.
5. Scan the QR code shown in the terminal with the "Expo Go" app or the Camera app (iOS).
6. Wait for the app to load on the mobile device.
7. Verify that the background color of the app has been changed to the desired color.

NOTE: Loading the app for the first time may take some time, and you may need to click Reload JS if you receive an error due to slow connections. In fact, if you use this method, the Codespaces and your phone may be disconnected due to inactivity, so if you see Disconnected from the Metro server, shake your phone to bring up the Expo devtool menu and click Reload.

1. **CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?**

**Section 5: Pushing your work to GitHub.**

* 1. Go to Source Control on your GitHub codespace and observe the pending changes.
  2. Type the Message for your changes in the Message box on the top. For example, “**Submission for ACM SIGITE2023 Workshop 2.**”
  3. Click on the dropdown beside the commit button and select **Commit & Push** to update the changes to your repository main branch.
  4. Select **Yes** when prompted.



1. **CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?**

**Mobile App Development Practices**

* Visit [ACM SIGITE 2023 Examples](https://github.com/samchung0117/sigite2023-example).
  + Visit the README.md file.
  + Find examples for your practices.

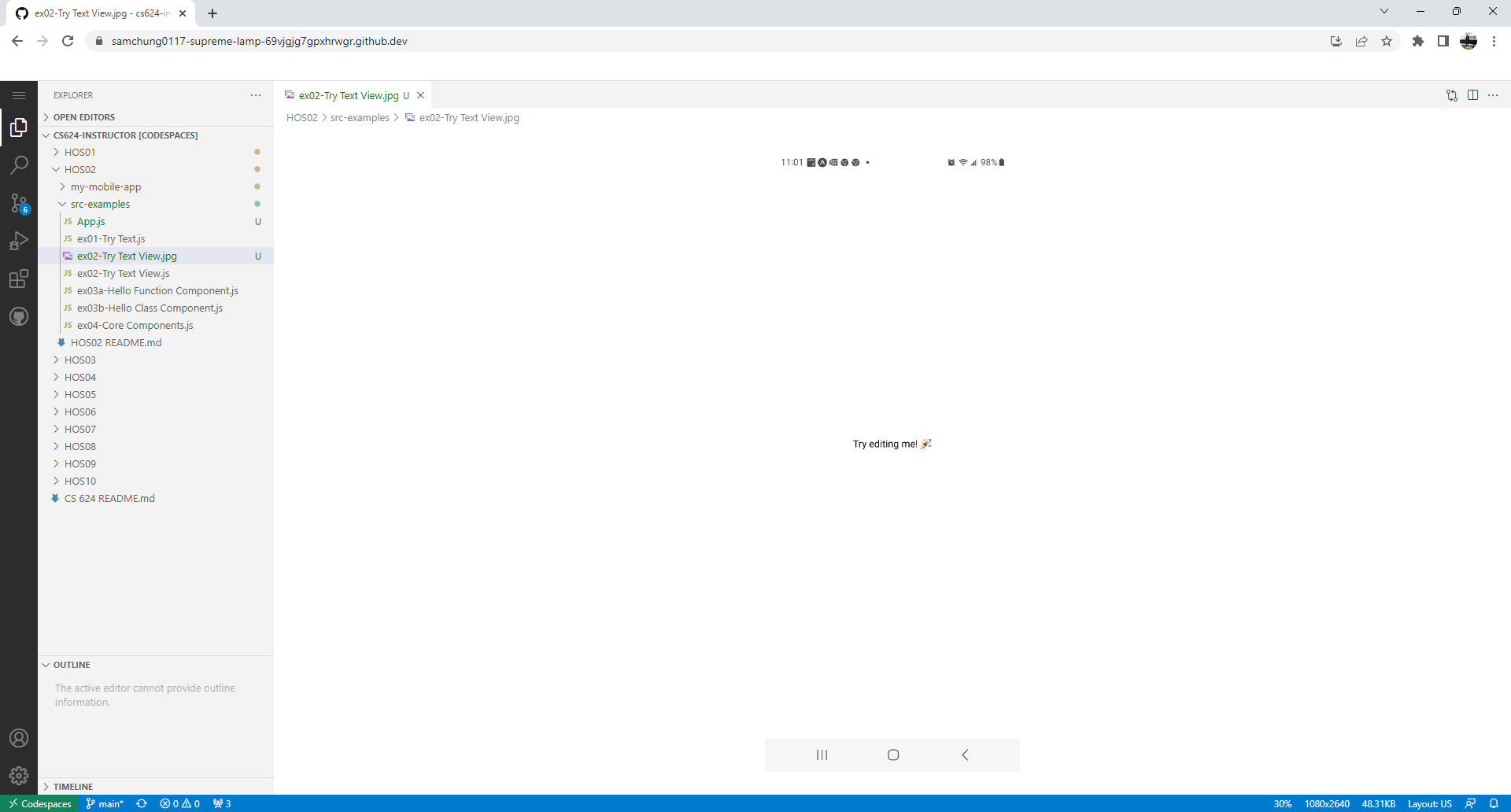
**Section 6: Using <Text> and <View> components.** The code imports two components **Text** and **View** from the **react-native** library. The **App** function is a component that returns a **View** component with a defined inline **style** property, which sets the **flex** value to 1 and centers the content both vertically and horizontally. Within the **View** component, there is a **Text** component displaying the text "Try editing me!". The **App** component is exported as the default export.

1. Open the **app.js** file in the GitHub Codespaces environment.
2. Replace the contents of **app.js** with the following code.  
   PLEASE type the source codes step by step to learn React Native programming.



1. Save the changes to the file.
2. Steps to verify the changes:

* Since the expo client is already connected, hit “r” button in the terminal to reload the app.
* Observe that (scanning the QR code again) the content is now properly visible at the center of the screen.
* Capture the screenshot (section03-Try Text View.jpg) from the mobile. Each smartphone provides a feature to take screenshot.
* Create “src-examples” under the “hos02” directory.
* Drag and drop the screenshot under “hos02/src-examples” directory.



**A picture containing chart

Description automatically generated**

1. Push your work to GitHub.
2. CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?

**Section 7: Creating a functional component.**

This code imports the **Text** and **View** components from the **react-native** library. It then creates a functional component **App** which returns a **View** component that takes up all available space (**flex: 1**) and is centered both horizontally and vertically (**justifyContent: 'center', alignItems: 'center'**). Within this **View** component, there is a **Text** component that says "Hello, world. Functional Component!". Finally, the **App** component is exported as the default export. In this case, we use an arrow function.

1. Open the **app.js** file in the GitHub Codespaces environment.
2. Replace the contents of **app.js** with the following code through your typing.



1. Save the changes to the file.
2. Steps to verify the changes:

* Since the expo client is already connected, hit “r” button in the terminal to reload the app.
* Observe that the text has changed.
* Capture the screenshot (section04-Functional Component.jpg) from the mobile and save it under “hos02/src-examples” directory.

Graphical user interface, text

Description automatically generated

1. Push your work to GitHub.
2. CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?

**Section 8: Creating a class component.**

This code is a React Native class component, defined by the "**class App extends Component**" syntax. The component renders a "**View**" element with a set of styles applied to it. The style aligns the content in the center of the view both horizontally and vertically. Within the view is a "**Text**" element displaying the message "Hello, world Class Component!".

1. Open the **app.js** file in the GitHub Codespaces environment.
2. Replace the contents of **app.js** with the following code,



1. Save the changes to the file.
2. Steps to verify the changes:

* Since the expo client is already connected, hit “r” button in the terminal to reload the app.
* Observe that the text has changed and now we are using class component instead of function component.
* Capture the screenshot (section05-Class Component.jpg) from the mobile and save it under “hos02/src-examples” directory.

Text, letter

Description automatically generated

1. Push your work to GitHub.
2. CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?

**Section 9: Using <ScrollView>, <Image>, and <TextInput> components.**

The code imports the necessary components from React Native library. Then it defines a stateless component 'App' using the arrow function that returns a ScrollView component. The ScrollView component has its **contentContainerStyle** property set to center the contents within the view. The ScrollView component includes 3 components, a Text component with a string 'Some text', a View component with a Text component and an Image component, and a TextInput component for user input. The Image component's source is set to an image URL, and its style is set to have a width of 200 and height of 200. You can see the cat image at <https://reactnative.dev/docs/assets/p_cat2.png>. The TextInput component has its height set to 40 and a border with color 'gray' and width 1. The component is then exported as default.

1. Open the **app.js** file in the GitHub Codespaces environment.
2. Replace the contents of **app.js** with the following code.



1. Save the changes to the file.
2. Steps to verify the changes:

* Since the expo client is already connected, hit “r” button in the terminal to reload the app.
* Observe that the text has changed and there is an image displayed using the Image component.
* Also, there is a TextInput component which allows the users to type in some text.
* Capture the screenshot (section06-Core Components.jpg) from the mobile and save it under “hos02/src-examples” directory. Each smartphone provides a feature to take screenshot. For example,

Text

Description automatically generated with medium confidence

1. Push your work to GitHub.
2. CHALLENGE: What is the purpose of this section? i.e., why did you practice this section?