Getting Started with React

In this discussion we will be downloading Node and creating a simple webpage using React and React components.

Task 1: Download Node

1. Visit https://nodejs.org/en/download/package-manager and download and install the latest version of Node for your operating system
    a. Feel free to use any of the download methods: package manager, prebuilt installer, etc.
2. After installing Node, confirm this has been done successfully by using the commands **node -v** and **npm -v** in your terminal

Task 2: Initialize a React project (terminal)

If you would prefer to use a GUI, skip to the next version of this task

1. In your terminal, navigate to the location you would like to initialize a new project (e.g. **/cs391/labs**)
2. Use the command **npm create vite@latest** ***project-name*** to create a new project with the name of your choice
3. Select to utilize the React framework with Typescript

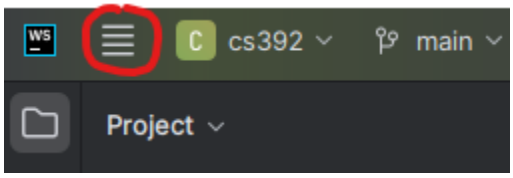
```
$ npm create vite@latest react-discussion
Need to install the following packages:
create-vite@5.5.2
Ok to proceed? (y) y
√ Select a framework: » React
√ Select a variant: » TypeScript
```

4. Enter the root of your project using the following command **cd** ***project-name***
5. Use the command **npm install** to install any necessary dependencies
6. You can now use the command **npm run dev** to see the default state of the project at http://localhost:5173
7. This should automatically hot-reload when you make changes, so you don't need to constantly re-run the project

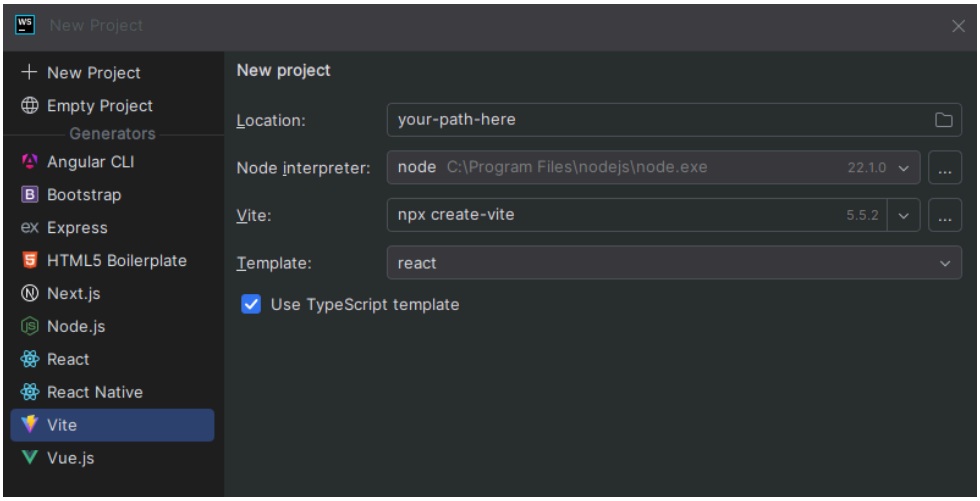
Task 2: Initialize a React project (WebStorm)

If you have already initialized your React project, skip to the next task. If you would prefer to use the terminal, return to the previous version of this task

1. Select the horizontal bars in the top left corner, then hover over **File**



2. From the dropdown, select **New**, then **Project…**
3. Select **Vite** from the column on the left and configure the following: the desired path/location of the project, the template, and select **Use TypeScript template**



4. After your have configured everything, click **Create** and open the newly created project
5. If prompted in the bottom right, select **Run ‘npm install’**, otherwise, open the builtin terminal and run **npm install** to install any necessary dependencies
6. You can now click the play button at the top of the screen or use the command **npm run dev** to see the default state of the project at http://localhost:5173
7. This should automatically hot-reload when you make changes, so you don’t need to constantly re-run the project

Task 3: Edit the displayed content

1. If you haven't already, open the project in your preferred editor and select the file **src/App.tsx**
    a. Here you can see a function called **App** being defined and exported. This function returns the content shown on the screen.
2. Delete all of the content in this file. We are going to rewrite this file to display our desired content

```
import './App.css'

function App() {

  return (
    <div>
      <h2>Welcome to my first React application</h2>
      <div>
        <h4>Look at my custom content!</h4>
      </div>
    </div>
  )
}

export default App
```

3. You should now see these changes in the content at http://localhost:5173!

Task 4: Create a simple component

Components allow us to define structures so that we don't need to repeatedly type large amounts of the same code. This makes our code much easier to read, and, more importantly, it makes it much easier to make changes to the code. When the component changes, all instances of it change as well. In this demonstration, we will be creating a component to show each member of the course staff, however, feel free to make a component for anything you want (e.g. movies, books, tv shows, pokémon, courses, programming languages, cars, etc.)

1. Create a new directory with the path **src/components** and then create a new file in that directory ending in **.tsx**

that reflects the content you will display using this component

2. Below is our example of a component, **CourseStaff** (components should be named using PascalCase). Name your component appropriately. By convention, the filename should be the same as the main component it exports/defines but using spinal-case/kebab-case.
    a. In this file we first define the input type. Each member of the course staff has a name, title, and a rating. Name and define this type appropriately based on what you've chosen. (i.e. if you chose cars, you may want to include year, make, model, distance driven, etc. if you chose pokémon you may want to include name, type, evolution stage, etc.) Properties should be name using camelCase
    b. We then define our component (a function) as taking in an object with the properties we defined above. We declare that this object is of the above type
    c. Finally, we return what we want this component to display with some inline React CSS styling. The styling is optional.
        i. Using braces, **{}**, indicates that we want to use the value of a variable instead of what was literally typed. Content within the tags that is not surrounded by braces is interpreted literally

```tsx
type CourseStaffProps = {
  name: string;
  title: string;
  rating: number;
}

export function CourseStaff({name, title, rating}: CourseStaffProps) {
  return (
    <div style={{padding: "0.25rem", margin: "1rem", backgroundColor: "lightsteelblue"}}>
      <h3>{name}</h3>
      <h4>{title}</h4>
      <p>{rating}/100</p>
    </div>
  )
}
```

Task 5: Import and use this new component

1. Go back to the file **src/App.tsx**
2. At the top of the file, import your newly created component with a line similar to **import { *YourComponent* } from "./components/*your-component*.tsx"**
3. Next, call your component with appropriate data, similar to the below example
    a. We call the component and fill out each of the required field that we defined for it

```
import './App.css'
import { CourseStaff } from "./components/course-staff"

function App() {

  return (
    <div>
      <h2>The amazing CS391 course staff!</h2>
      <div>
        <CourseStaff name="Taymaz" title="Professor" rating={92} />
        <CourseStaff name="Jeffrey" title="Teaching Assistant" rating={90} />
        <CourseStaff name="Sadiq" title="Course Assistant" rating={90} />
        <CourseStaff name="Ale" title="Course Assistant" rating={90} />
      </div>
    </div>
  )
}

export default App
```

4. You should now see the declared content and components at http://localhost:5173