

AJC Map Application Technical Document

Version <1.2>

**Sophia Fisher, Samarth Bhat,
Deja Martin, Nathan Jacob**

Table of Contents

1 Introduction.....	3
1.1 Overview.....	3
1.2 Goals.....	3
1.3 Application Framework.....	3
1.4 Source Code.....	4
2 Design Layout.....	5
2.1 Map Design – Laptop.....	5
2.2 Map Design – Phone.....	6
3 Transitioning Control.....	7
3.1 Transitioning Tasks – UT Team.....	7
3.2 Transitioning Tasks – AJC Team.....	7
3.3 Suggested Implementation.....	7
4 Instructions.....	8
4.1 Setting up Your Local Repository.....	8
4.2 Adding a New Location Marker.....	12
4.3 Setting Up Your Google Maps API Key.....	16
5 Integration Suggestions.....	24
5.1 Website Integration – Linking the Map.....	24
5.2 Website Integration – Directly Adding Map.....	24
6 Maintenance.....	26
6.1 Maintaining the Map Application.....	26

1. INTRODUCTION

1.1 Overview

The AJC Map Application was created to serve both long-term Japanese residents living in Austin and short-term Japanese travelers visiting the city. This tool allows users to explore a map of Austin, displaying markers that signify recommended sights and locations shared by members of the Japanese Austin community. By providing these recommendations, we aim to strengthen the sense of connection and community among Japanese residents and visitors in Austin.

1.2 Goals

Our primary goal was to create an interactive map application that showcases various recommended spots within Austin. Users can narrow their search by filtering locations based on different categories or selecting tags such as “**Kid-Friendly**” or “**Dogs Allowed.**”

We also wanted the application to visually match the AJC website by using a similar color scheme, fonts, and imagery, which would help maintain visual cohesion if the application is integrated into the site. Additionally, we made the application accessible to both Japanese and English speakers by designing the text to support both languages.

1.3 Application Framework

The application was built using **HTML**, **CSS**, and **JavaScript**, with the **React** framework providing a modular approach to organizing different components and efficiently updating data in response to user interactions.

To streamline development, we incorporated **Material UI components** for features like checkboxes and dropdown menus, as these ready-made components saved time compared to building them from scratch. For the interactive map visualization, we used the **Google Maps Platform API (Maps JavaScript API)** to seamlessly integrate map functionality.

The application is deployed using **GitHub Pages**, allowing for easy hosting and public access to the latest version of the app.

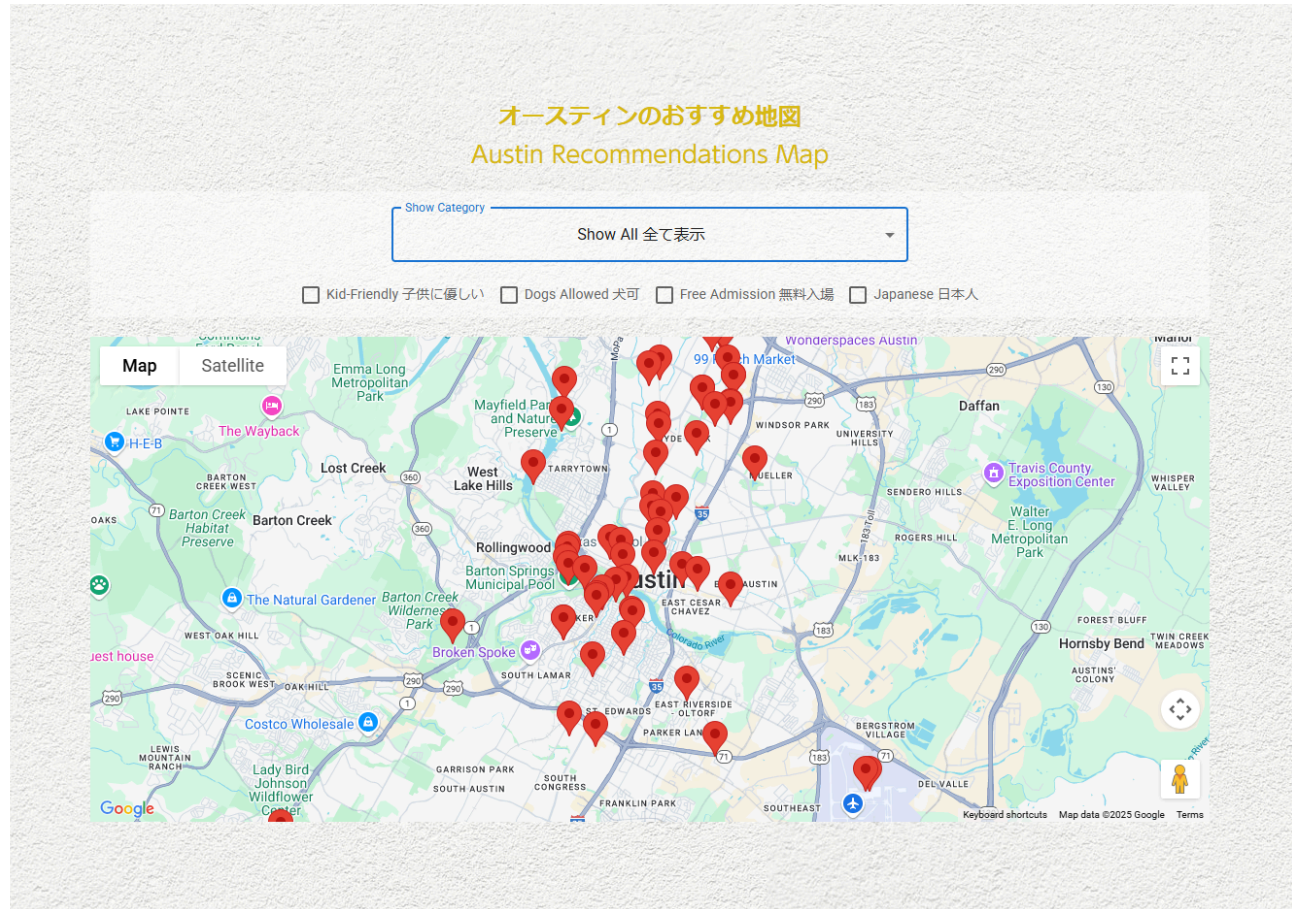
1.4 SOURCE CODE

The main files of importance to the application are: **App.js**, **App.css**, **index.html**, and **places.json**.

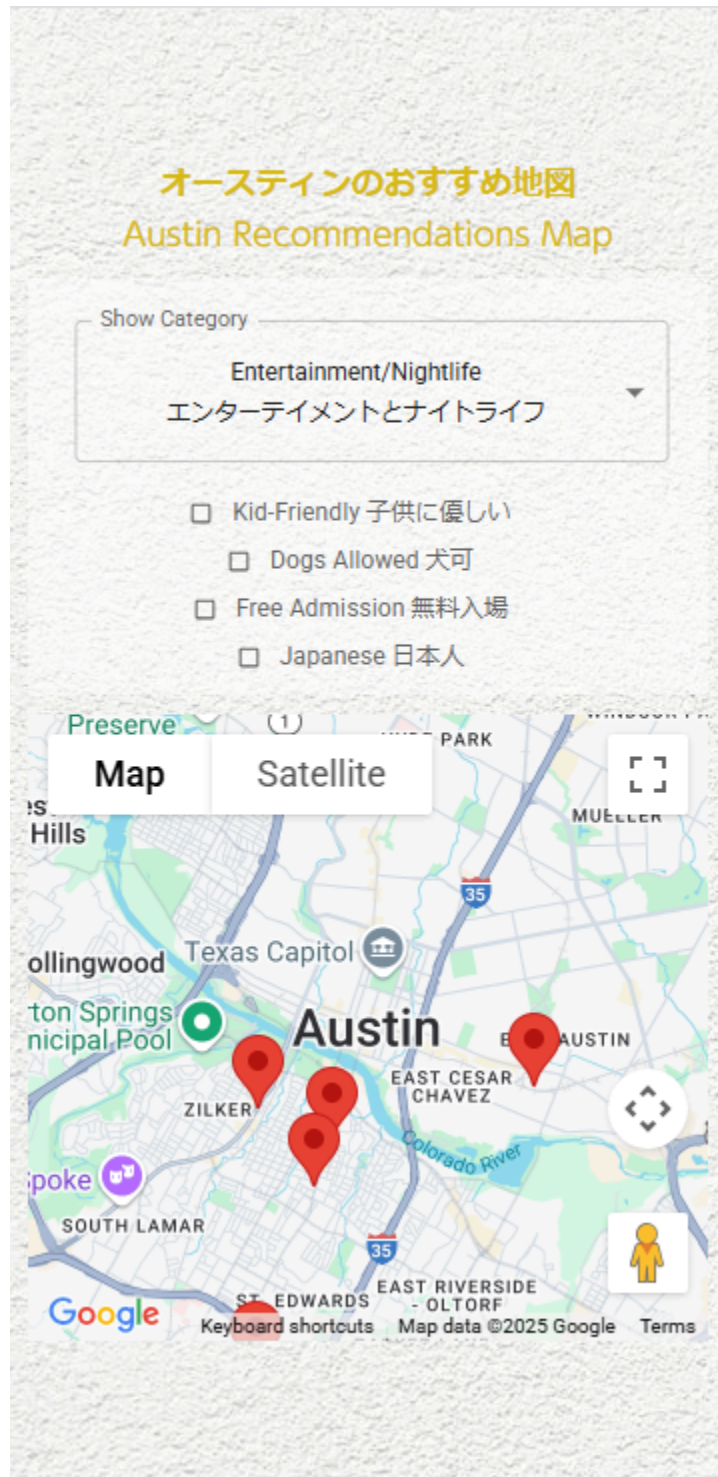
- **App.js:** Defines the HTML structure of the application and contains the JavaScript logic to handle user input and update the application controls.
- **App.css:** Provides styling for all components defined in **App.js**, ensuring a consistent and visually appealing user interface.
- **index.html:** Initializes the APIs used by the application, including the **Google Maps API**.
- **places.json:** Contains the data for all location markers, including their fields and attributes.

2. DESIGN LAYOUT

2.1 Map Design – Laptop



2.2 Map Design – Phone



3. TRANSITIONING CONTROL

3.1 Transitioning Tasks – UT Team

- ✓ Add the AJC IT team members to the GitHub repository as Admins.
- ✓ Add all dependencies to the `package.json`.
- ✓ Create a `setup.sh` file that runs all necessary commands for setup.
- ✓ Add a `README.md` to the repository with clear instructions for modifying the source code.
- ✓ Implement a GitHub Action `.yaml` file to automatically redeploy the application when changes are pushed.

3.2 Transitioning Tasks – AJC Team

- ☐ Accept Github repository invite.
- ☐ Create AJC Google Maps API Key.
- ☐ Update source code to use the new API key.
- ☐ Deploy changes to ensure the application uses the new API key.
- ☐ Add link to the map application on AJC website.

3.3 Suggested Implementation

To transition control of the application from our development team at UT to your AJC organization, we've created a set of instructions to guide you through the process. First, we recommend following the instructions for setting up your local repository. This will allow you to gain access to the source code and ensure that you can make changes. Next, we suggest adding a new location marker using the provided instructions. This will confirm that you can successfully modify the code and deploy changes to the active URL.

Once you're comfortable with making changes, proceed to set up your own Google Maps API key. Currently, the application is using one of our personal Google Maps Platform accounts, and we need to transfer that responsibility to your organization. By the end of this process, the source code will be securely transitioned to AJC.

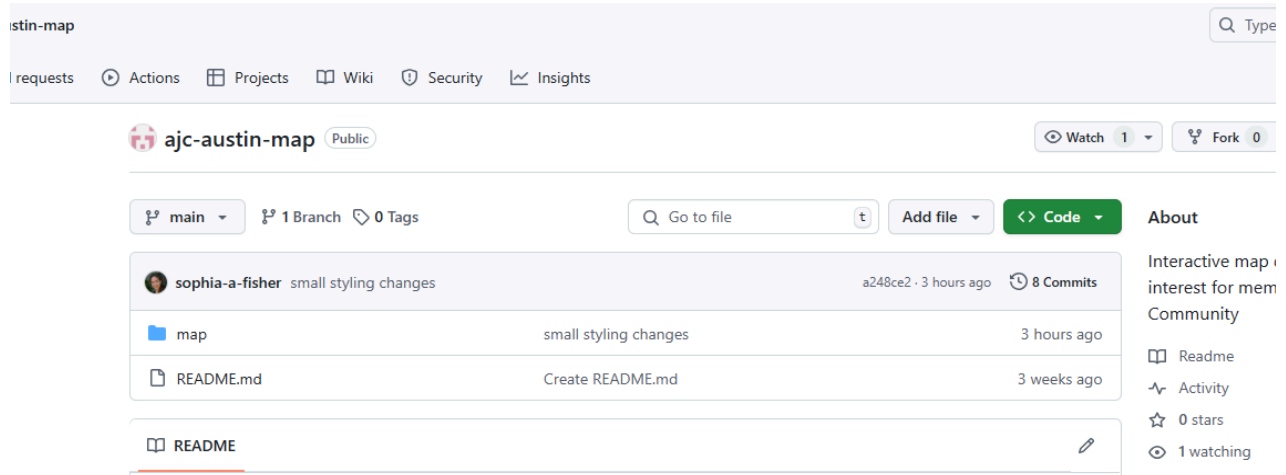
Finally, we've included an outline with instructions on how to integrate the map application into your website. You can either link to the application or directly embed the component.

4. INSTRUCTIONS

4.1 Setting Up Your Local Repository

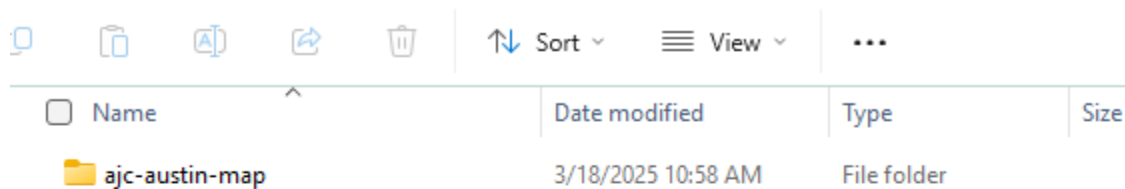
1) Navigate to the GitHub Repository:

- Go to <https://github.com/sumearthb/ajc-austin-map>.



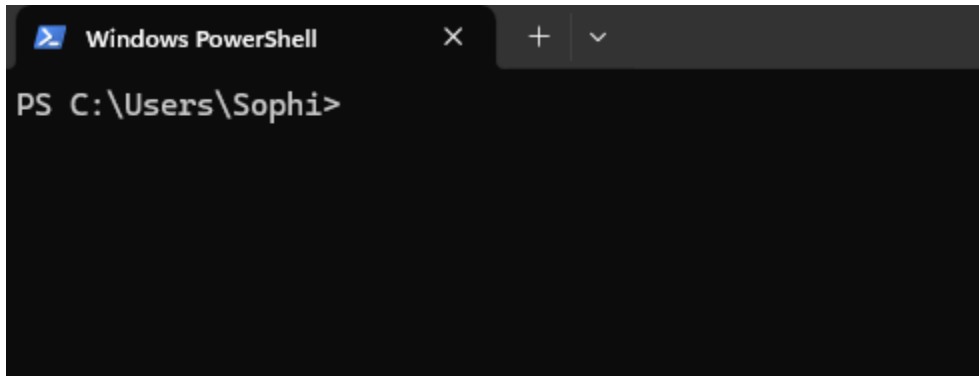
2) Create a New Folder:

- Create a new folder on your computer to store this project.



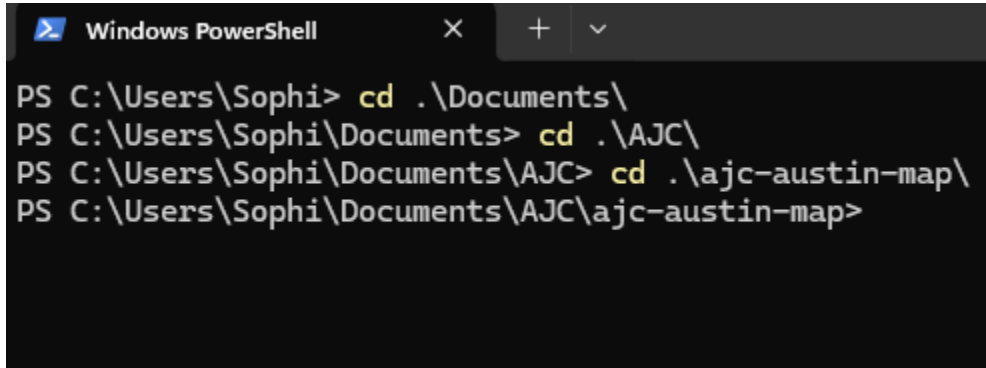
3) Open Windows PowerShell:

- Launch **Windows PowerShell**.



4) Navigate to the New Folder:

- In PowerShell, navigate to the folder you just created using the `cd` command.

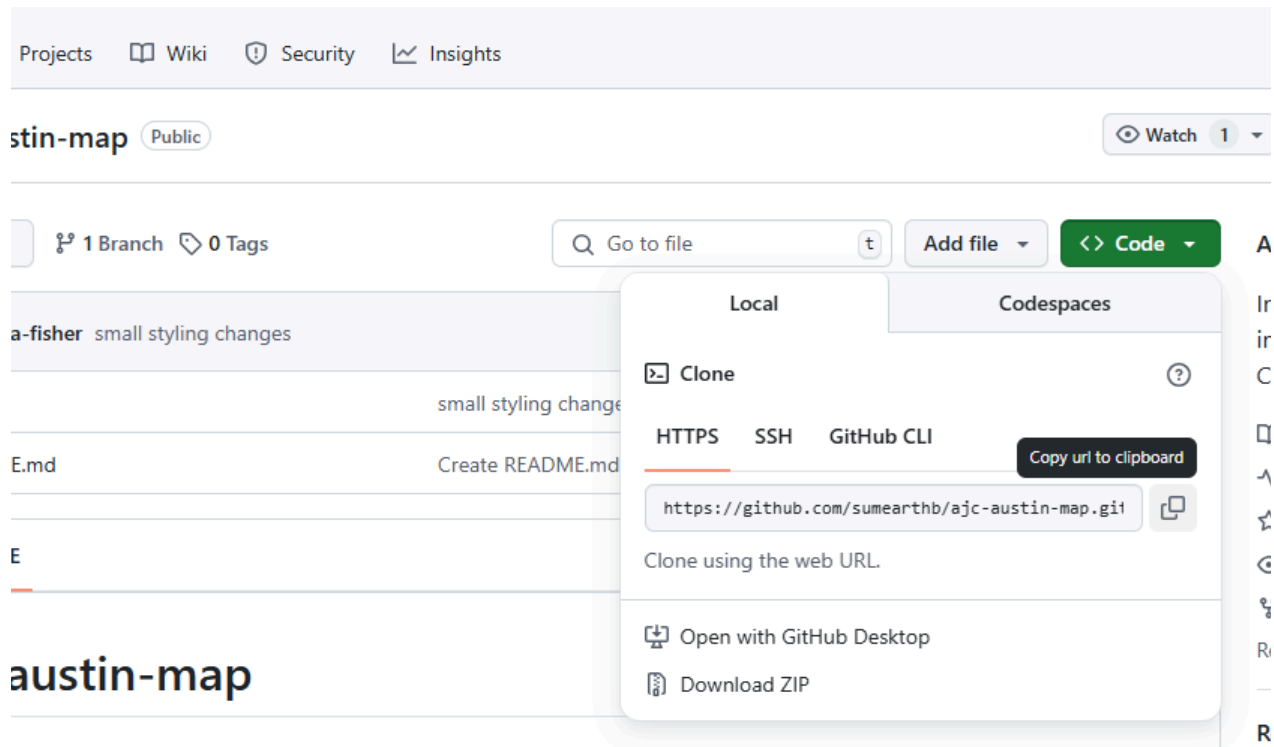


5) Clone the Repository:

- On the GitHub repo page, click the green **Code** button.
- Copy the URL listed under **Local HTTPS**.
- In PowerShell, type the following command and press Enter:

`git clone <paste the copied URL> .`

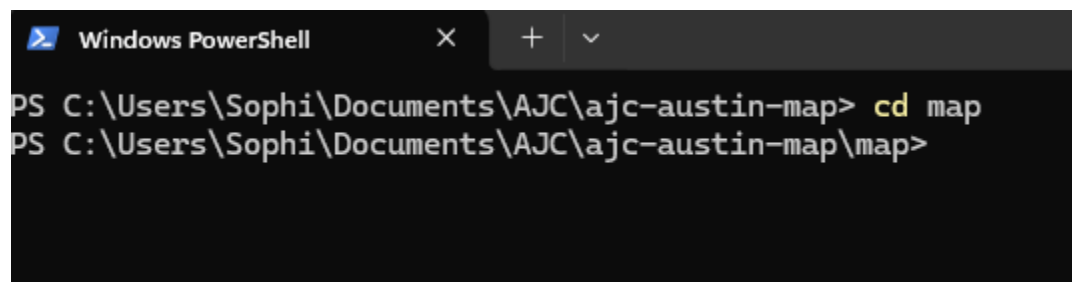
- This will download all the source files to your local machine.



```
git clone https://github.com/sumearthb/ajc-austin-map.git .
```

6) Navigate to the **map** Folder:

- In PowerShell, change the directory to the **map** folder inside the project:



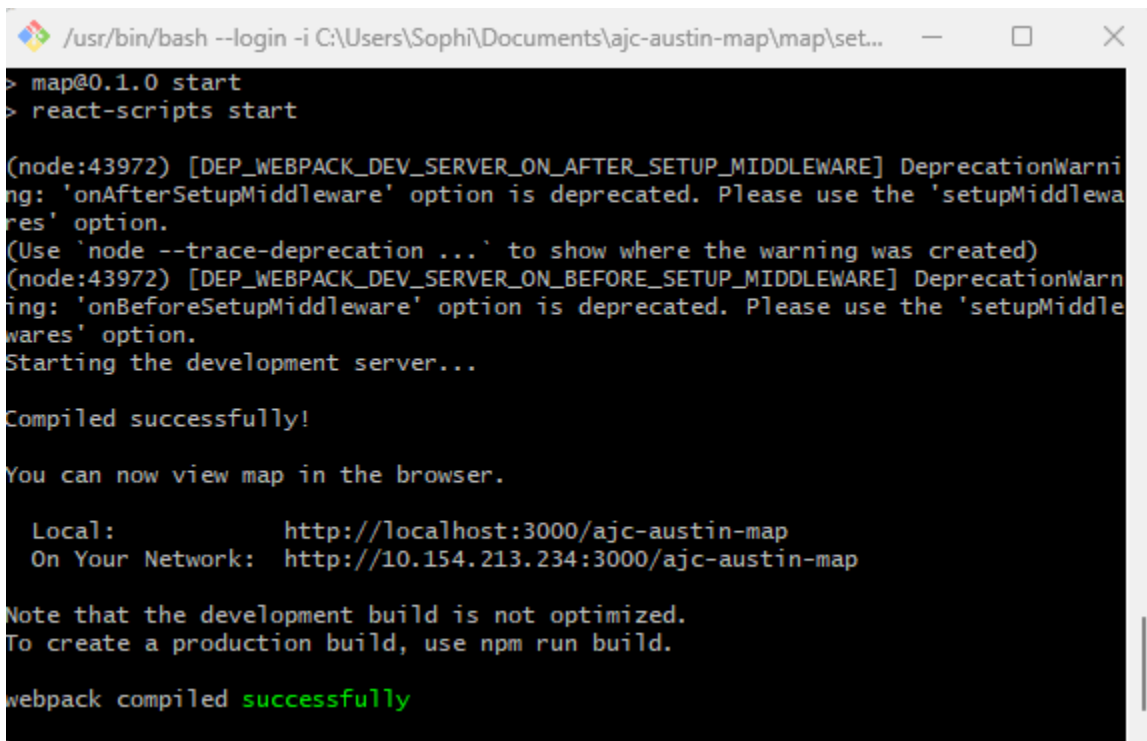
7) Run Setup Script:

- Run the setup script by typing the following command:

```
./setup.sh
```

- This will install all necessary dependencies and open a development window where you can view any changes made to the application.
- **This might take a little while, so please be patient!**

```
C:\Users\Sophi\Documents\ajc-austin-map\map> ./setup.sh
C:\Users\Sophi\Documents\ajc-austin-map\map> |
```

A screenshot of a terminal window with a title bar that reads "/usr/bin/bash --login -i C:\Users\Sophi\Documents\ajc-austin-map\map\set...". The terminal content shows the execution of the setup script. It starts with "map@0.1.0 start" and "react-scripts start". There are two deprecation warnings from Webpack Dev Server regarding "onAfterSetupMiddleware" and "onBeforeSetupMiddleware" options. The script then says "Starting the development server...", "Compiled successfully!", and "You can now view map in the browser.". It provides two URLs: "Local: http://localhost:3000/ajc-austin-map" and "On Your Network: http://10.154.213.234:3000/ajc-austin-map". A note states "Note that the development build is not optimized. To create a production build, use npm run build." Finally, it shows "webpack compiled successfully" in green text.

```
> map@0.1.0 start
> react-scripts start

(node:43972) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:43972) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...

Compiled successfully!

You can now view map in the browser.

  Local:            http://localhost:3000/ajc-austin-map
  On Your Network:  http://10.154.213.234:3000/ajc-austin-map

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

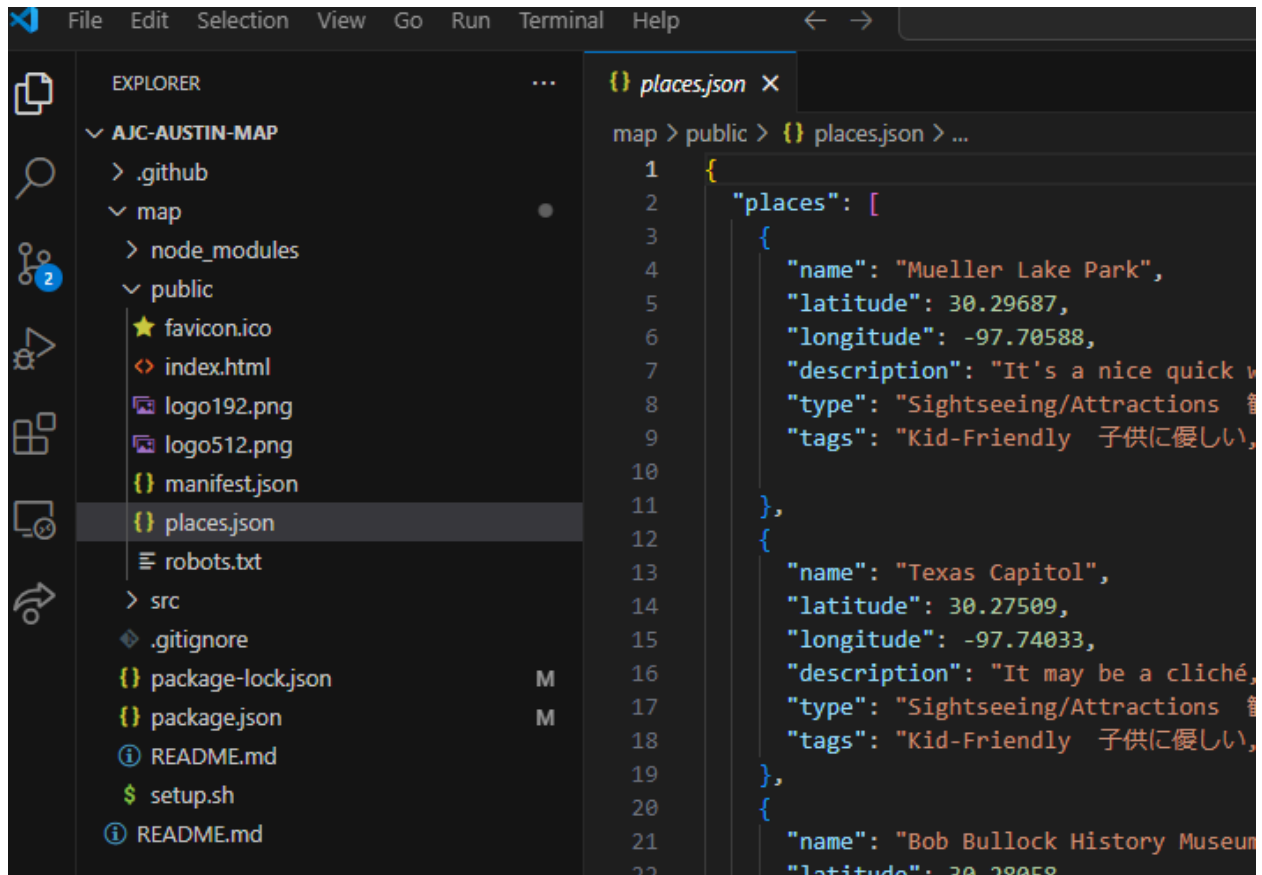
8) Modify the Source Code:

- Once this setup is complete, you're ready to make changes to the source code!

4.2 Adding a New Location Marker

1) Open the Project in Your IDE:

- Open the project folder in your preferred IDE (e.g., Visual Studio Code).



2) Navigate to **places.json**:

- In the file explorer within your IDE, locate and open the **places.json** file. This file contains the data for all recommendation locations.

3) Add Your New Location:

- To add a new location, follow the same structure as the existing entries in the file. Each location should include fields like name, description, coordinates, and tags.

```

668 | {
669 |   "name": "Daiso",
670 |   "latitude": 30.23263,
671 |   "longitude": -97.82054,
672 |   "description": "Japanese variety-store chain featuring J
673 |   "type": "Shopping 買い物",
674 |   "tags": "Kid-Friendly 子供に優しい, Japanese 日本人"
675 | }

```

4) Save Your Changes:

- Once you've added your location, save the changes to `places.json`.

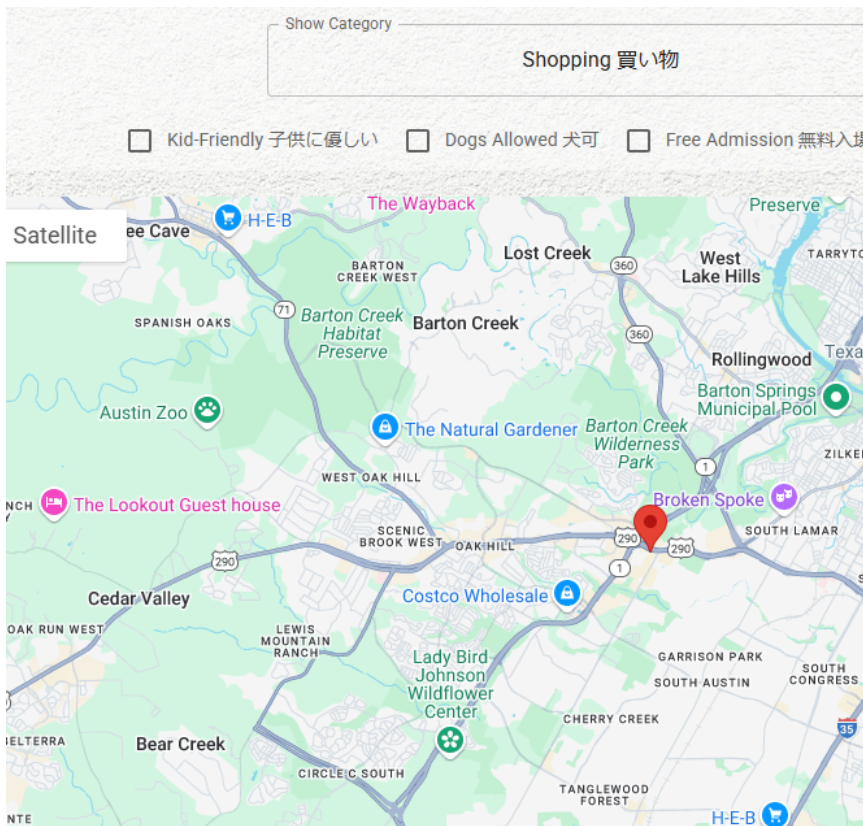
5) View the Location on the Map:

- Run the application locally using the development server (via `./setup.sh` as described earlier or by running `npm start`).
- You should now see your newly added location on the map!

```

p\ajc-austin-map\map> npm start

```



6) Stage the Changes:

- In the terminal, navigate to the project folder (if you're not already there).
- Stage your changes by typing the following command and pressing Enter:

`git add .`

```
Documents\ajc-austin-map\map> git add .  
Documents\ajc-austin-map\map>
```

7) Commit the Changes:

- Now, commit your changes with a meaningful message. You can use:

`git commit -m "Added new recommendation location"`

```
PS C:\Users\Sophi\Documents\ajc-austin-map\map> git commit -m "added a new location"  
[main 917a567] added a new location  
3 files changed, 107 insertions(+), 21 deletions(-)  
PS C:\Users\Sophi\Documents\ajc-austin-map\map>
```

8) Push the Changes to GitHub:

- To upload your changes to the remote GitHub repository, type the following command:

`git push`

```
PS C:\Users\Sophi\Documents\ajc-austin-map\map> git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 2.61 KiB | 533.00 KiB/s, done.
Total 7 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To https://github.com/sumearthb/ajc-austin-map.git
 7551e70..917a567  main -> main
PS C:\Users\Sophi\Documents\ajc-austin-map\map>
```

9) Redeploy the Application:

- To ensure the working link for the application gets updated, you need to redeploy.
- First, build the application by typing:

```
npm run build
```

- Once the build process is complete, deploy the application using:

```
npm run deploy
```

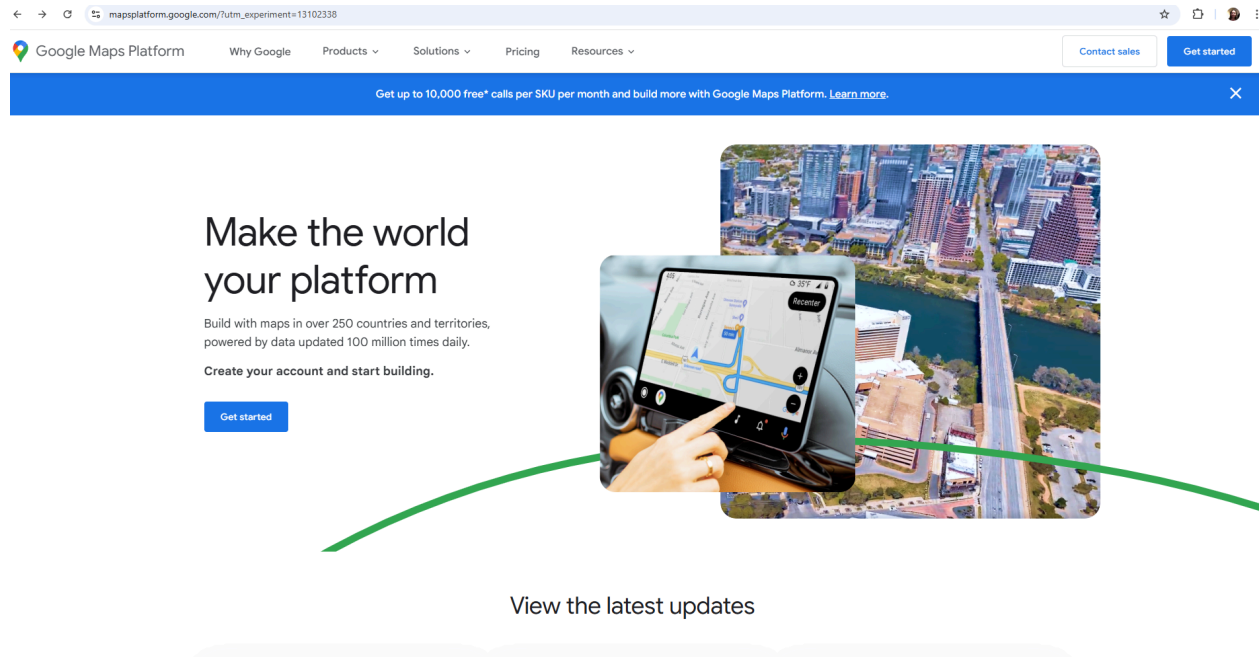
```
\ajc-austin-map\map> npm run build
```

```
p\ajc-austin-map\map> npm run deploy
```

4.3 Setting Up Your Google Maps API Key

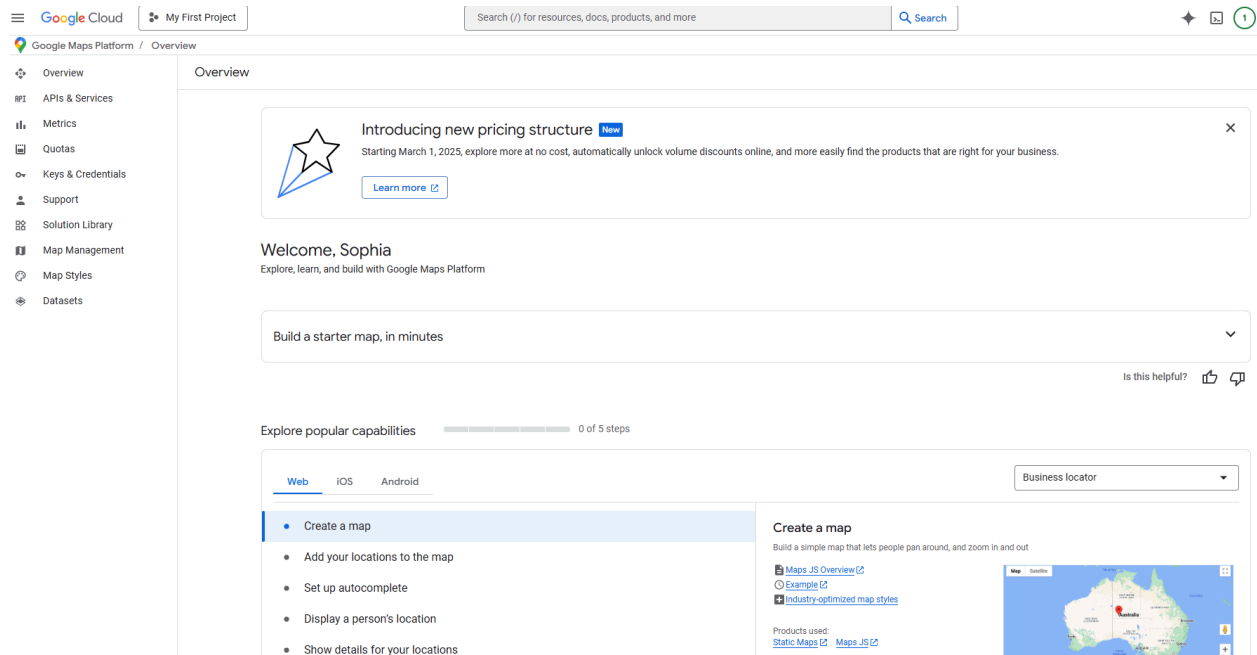
1) Navigate to Google Maps Platform:

- Go to the Google Maps Platform website:
https://mapsplatform.google.com/?utm_experiment=13102338.



2) Click the Get Started Button:

- On the Google Maps Platform page, click the **Get Started** button.

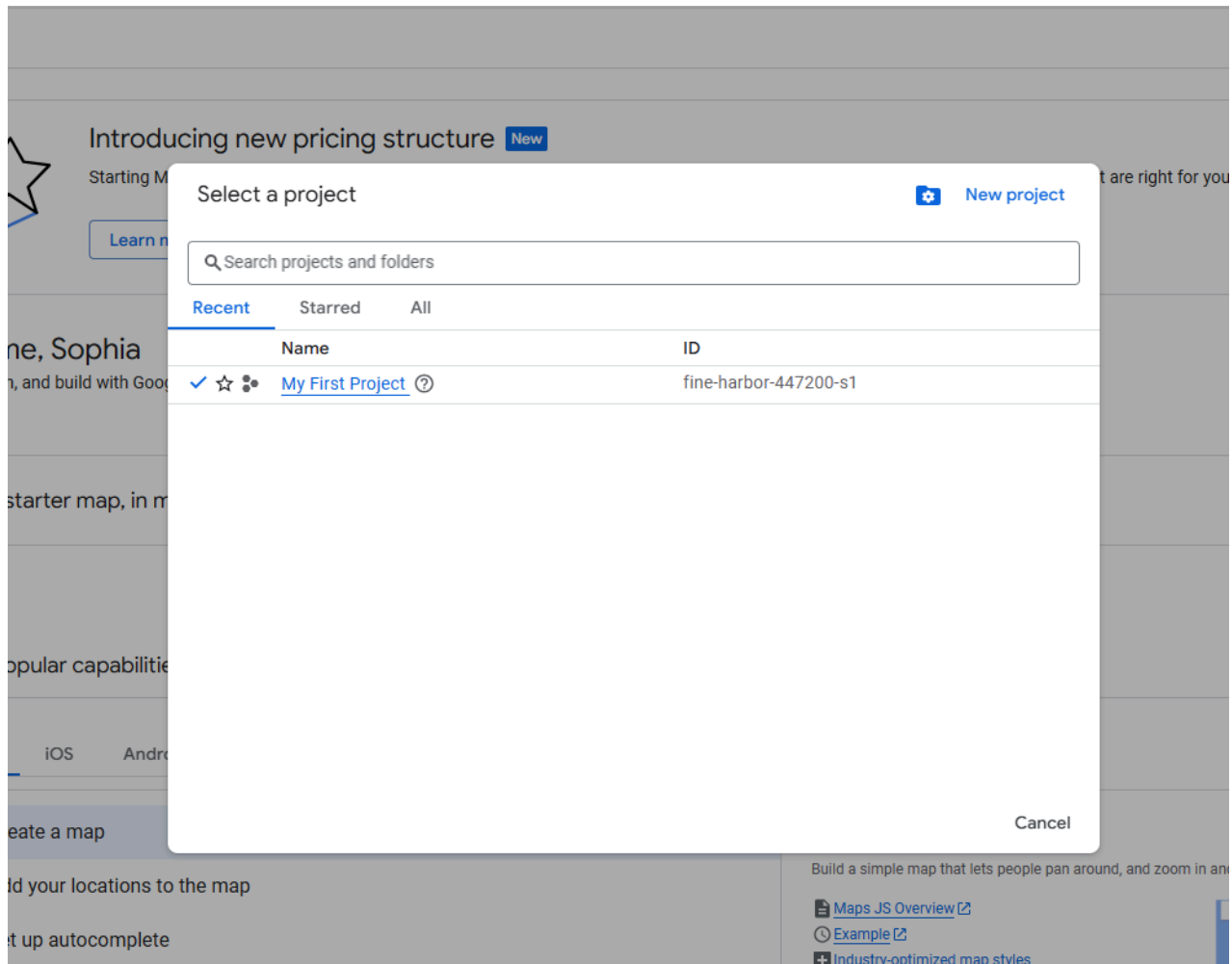


3) Create a New Account (if you don't have one):

- If you don't already have an account, you will be prompted to create one.
- The setup process will require you to enter payment information.
- **Google Maps API guarantees access of up to 10K calls per SKU at no cost per month**, so it is unlikely the AJC organization will pay anything

4) Create a New Project (if you don't have one):

- If you don't already have a project, you will need to create one by following the prompts.



5) Enable the Maps JavaScript API:

- Under the **APIs & Services** tab, click the **Enable** button for the **Maps JavaScript API**.

Google Cloud My First Project Search (/) for resources, docs, products, and more Search

Google Maps Platform / APIs & Services

Overview
APIs & Services
Metrics
Quotas
Keys & Credentials
Support
Solution Library
Map Management
Map Styles
Datasets

APIs & Services

Discover what tools you need to transform your maps and location-based experiences.

PRODUCTS: Maps Places Routes Environment TIERS: Essentials Pro Enterprise STATUS: Enable

Air Quality API Enable Provides air quality data for a specific location with a resolution of 500 x 500 meters Environment Guides	Solar API Enable Advanced imagery and insights to create solar proposals and designs Environment Guides	Poll Provides pol of 1x1km Environment
Maps Datasets API Enable Use your own geospatial data with Google Maps Platform APIs Maps Guides	Street View Publish API Enable Publishes 360 photos to Google Maps, along with position, orientation, and connectivity metadata.... Maps Guides	Ma 2D, 3D and \$ visualizator Maps
Maps Embed API Enable Make places easily discoverable with interactive Google Maps. Maps Guides	Maps JavaScript API Disable Maps for your website Maps Keys Metrics Guides	Ma Maps for yo Maps
Maps SDK for iOS Enable Maps for your native iOS app.	Maps Static API Enable Simple, embeddable map image with minimal code.	Stre Real-world i

6) Create API Credentials:

- Go to the **Keys & Credentials** tab and click the **Create Credentials** button.
- Click the **Show Key** button to copy your unique API key.

**Protect API Keys in Places API (New)****New**

Secure your Places API (New) API Keys with Firebase App Check! Prevent unauthorized access and keep the corresponding platform to find the integration steps.

[Places \(New\) JavaScript](#)[Places \(New\) Android](#)[Places \(New\) iOS](#)**Credentials compatible with this API**

To view all credentials visit [Credentials in APIs & Services](#)



Remember to configure the OAuth consent screen with information about your application.

API Keys

Name	Creation date	Restrictions	↑
------	---------------	--------------	---

**Protect API Keys in Places API (New)****New**

Secure your Places API (New) API Keys with Firebase App Check! Prevent unauthorized access and keep your API Keys safe. Only your trusted apps and websites will have access. Click on the corresponding platform to find the integration steps.

[Places \(New\) JavaScript](#)[Places \(New\) Android](#)[Places \(New\) iOS](#)**Credentials compatible with this API**

To view all credentials visit [Credentials in APIs & Services](#)



Remember to configure the OAuth consent screen with information about your application.

[Configure consent screen](#)**API Keys**

Name	Creation date	Restrictions	↑	Actions
▲ API key 2	Mar 23, 2025	—		Show key ⋮
▲ API key 1	Jan 7, 2025	—		Show key ⋮

7) Restrict the API Key:

- Click on your API key's name to manage the key settings.

- Under **API restrictions**, select **Restrict Key** and choose **Maps JavaScript API** as the only supported API.
- If you are deploying the application using **GitHub Pages**, select **Websites** under **Application restrictions** and add the URL of the application as well as the development server URL:
<https://sumearthb.github.io/ajc-austin-map/>.
<http://localhost:3000/ajc-austin-map>

API APIs & Services

Enabled APIs & services
 Library
 Credentials
 OAuth consent screen
 Page usage agreements

Application restrictions

☐ None
☒ Websites
☐ IP addresses
☐ Android apps
☐ iOS apps

Website restrictions

Restrict key usage requests to the specified websites.

If left blank, your API key will accept requests from any website.

Add

Filter Enter property name or value

<input type="checkbox"/>	Status	Website	Edit
<input type="checkbox"/>		http://localhost:3000/ajc-austin-map	
<input type="checkbox"/>		https://sumearthb.github.io/ajc-austin-map/	

API restrictions

☐ Don't restrict key
This key can call any API
☒ Restrict key

1 API

Selected APIs:
Maps JavaScript API

8) Update **index.html** with Your API Key:

- In your project folder, open the **public/index.html** file.
- Find line 32, which references the Google Maps API.
- Replace the text after **key=** and before **&callback=** with your unique API key.

```
user's mobile device or desktop. See https://developers.google.com/web/fundam
-->
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
<!--
Notice the use of %PUBLIC_URL% in the tags above.
It will be replaced with the URL of the `public` folder during the build.
Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
work correctly both with client-side routing and a non-root public URL.
Learn how to configure a non-root public URL by running `npm run build`.
-->
<title>React App</title>
</head>
<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
  <script src="https://maps.googleapis.com/maps/api/js?key=INSERTHERE&callback=ir
  <!--
This HTML file is a template.
If you open it directly in the browser, you will see an empty page.
```

9) Test the Application:

- To check if the application is working, either open the development server or type the following command in the terminal:

npm start

- This should open a development window displaying the application. If everything is set up correctly, the map should load with your new API key.

10) Redeploy the Application:

- Follow steps 6-9 from *4.2 Adding a New Location Marker* to push your changes to the Github repository and redeploy the application.
- Once complete the application URL should be updated to use your API key.

5. INTEGRATION SUGGESTIONS

5.1 Website Integration – Linking the Map

We are deploying the application through GitHub Pages. Since the repository is already hosted on GitHub and deployment via GitHub Pages is free, we believe this is the best option for hosting the application.

To deploy the application, start by cloning the repository to your local machine using the steps outlined above. Once you've made any necessary changes (replacing the API Key), run the following commands to build and deploy the application:

```
npm run build
```

```
npm run deploy
```

After completing these steps, the updated application will be accessible via the link:
<https://sumearthb.github.io/ajc-austin-map>.

5.2 Website Integration – Directly Adding Map

If the AJC website's codebase does not yet have access to the necessary npm libraries, you will need to run `npm install` in your project's root directory. Additionally, if the Material UI components used in the map application are not already available, you'll need to install them by running:

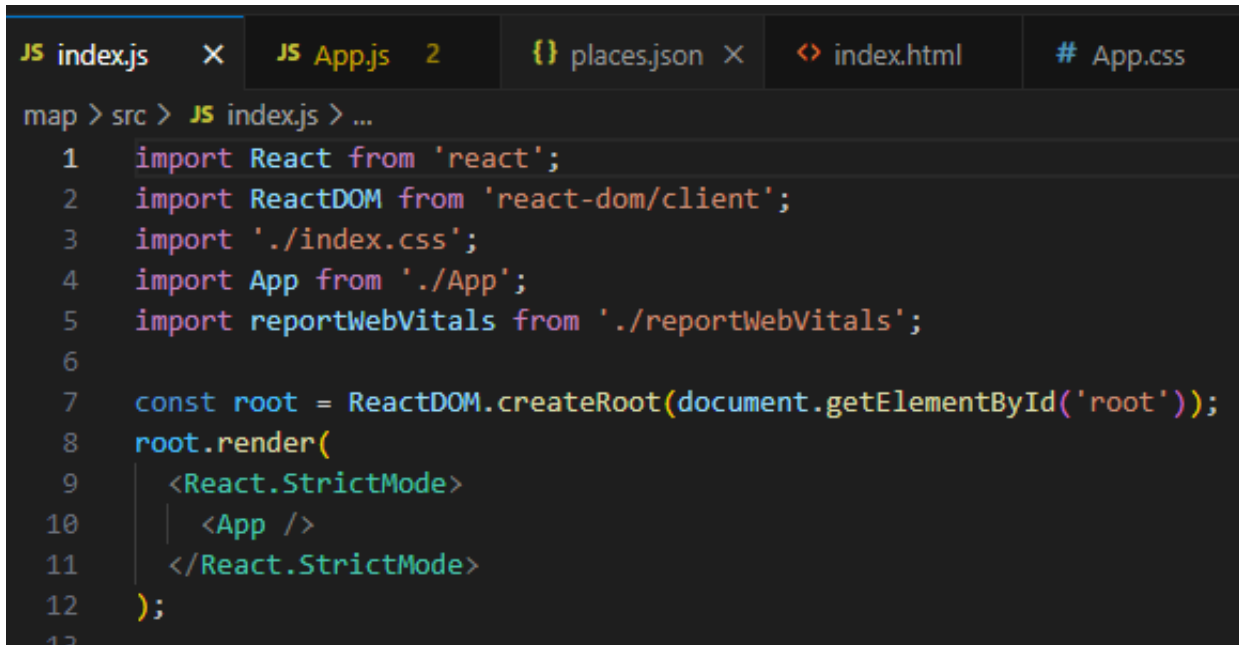
```
npm install @mui/material  
npm install @emotion/react @emotion/styled
```

All other required dependencies for the map application are listed in the `package.json` file under the `dependencies` section. You will similarly need to add them.

Next, if the AJC website does not currently use the Google Cloud API, you'll need to follow the steps outlined above to create a new project and generate an API key, ensuring that the **Maps JavaScript API** is enabled for your project.

Once you've set up the necessary dependencies and API key, you'll need to integrate the map application into your website's directory. Copy the files from the GitHub repository into your

project. If there are files with the same name already in your directory, you may need to rename them. The map application's entry point is located in `index.js`, with the map being rendered on line 10.

A screenshot of a code editor with a dark theme. The top of the editor shows a tab bar with five tabs: 'JS index.js' (active), 'JS App.js 2', '{} places.json', '<> index.html', and '# App.css'. Below the tabs, the breadcrumb path reads 'map > src > JS index.js > ...'. The main editor area displays the following JavaScript code:

```
1  import React from 'react';
2  import ReactDOM from 'react-dom/client';
3  import './index.css';
4  import App from './App';
5  import reportWebVitals from './reportWebVitals';
6
7  const root = ReactDOM.createRoot(document.getElementById('root'));
8  root.render(
9    <React.StrictMode>
10     <App />
11   </React.StrictMode>
12 );
13
```

Instead of keeping the map application in `index.js`, you will likely want to place the `<App />` component wherever you'd like it to appear within your website's HTML structure. To do this, you will need to copy the import statements from `index.js` into the appropriate file in your project.

As mentioned earlier, you will also need to insert your unique Google Maps API key into `index.html`. Alternatively, if you prefer, you can migrate that line of code into existing files in your project's source code where it is more appropriate.

6. MAINTENANCE

6.1 Maintaining the Map Application

To ensure the application remains deployed, there are a few important things to keep in mind. The repository must remain public, and it cannot be deleted or renamed.

If you need to add more locations to the map or make other changes to the application, follow these steps: clone the repository to your local machine, make the necessary changes, push those changes, and then redeploy the application.