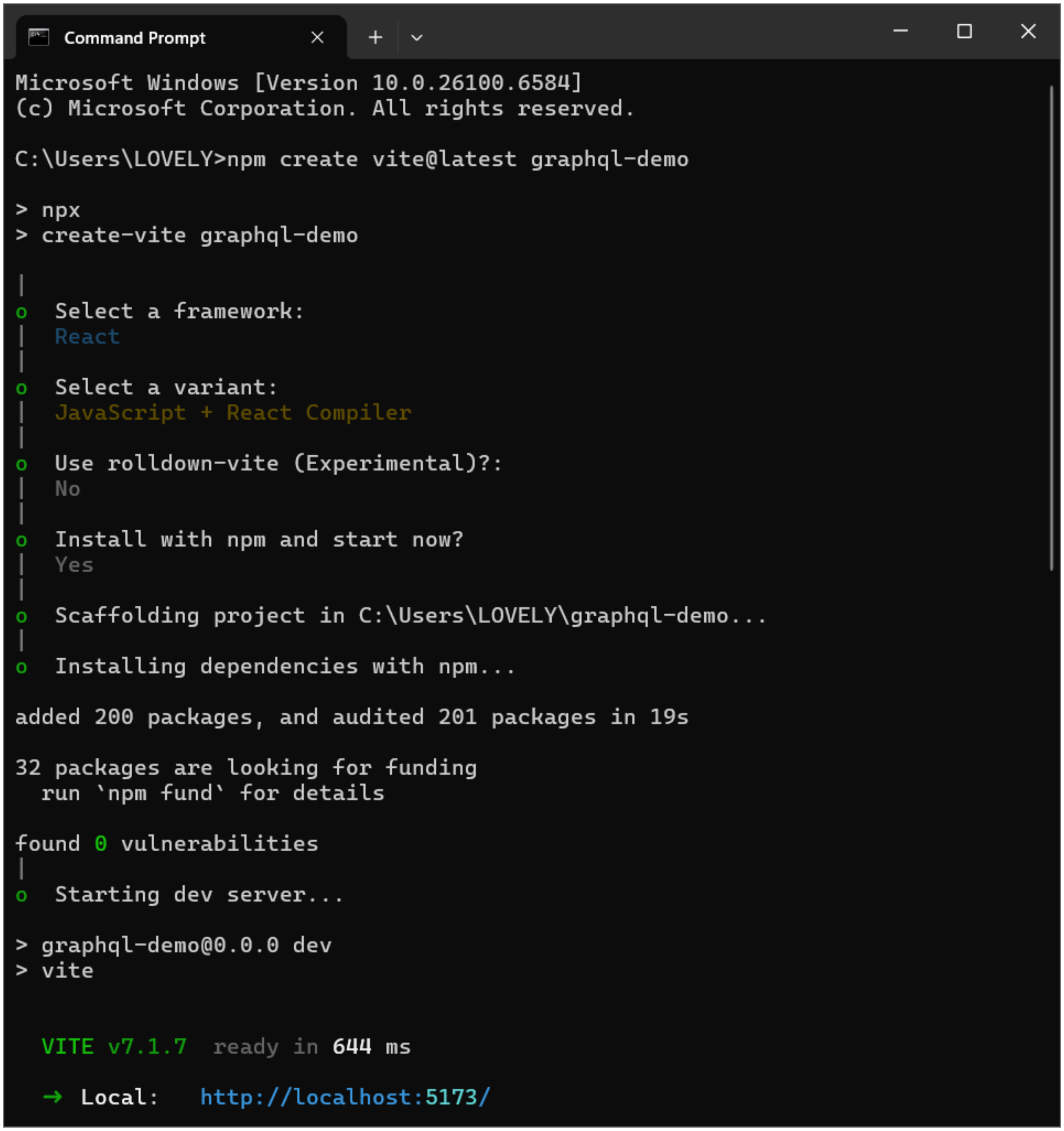
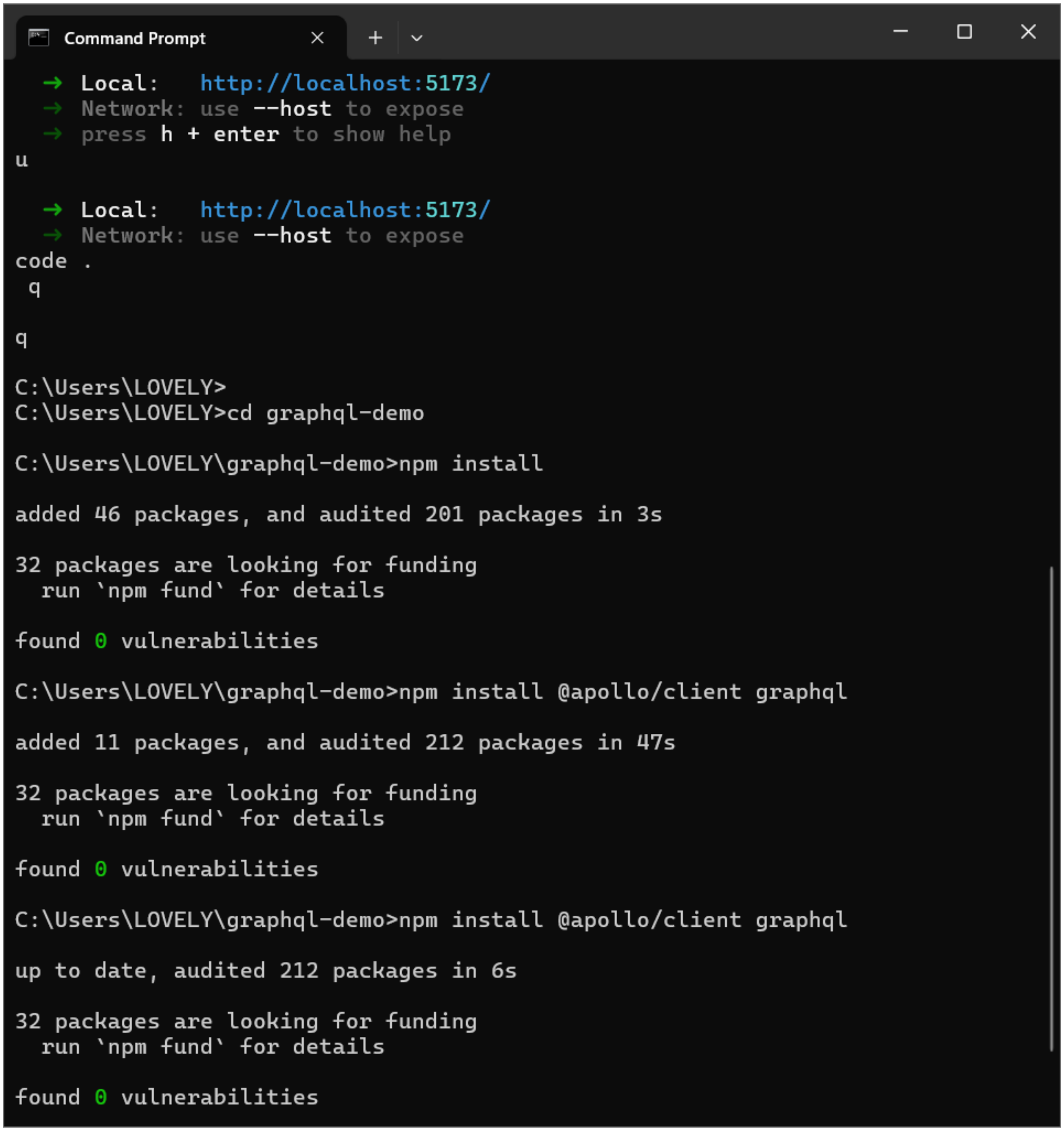
CASTILLANO, LOVELY ANNE O.

COMMAND PROMPT:



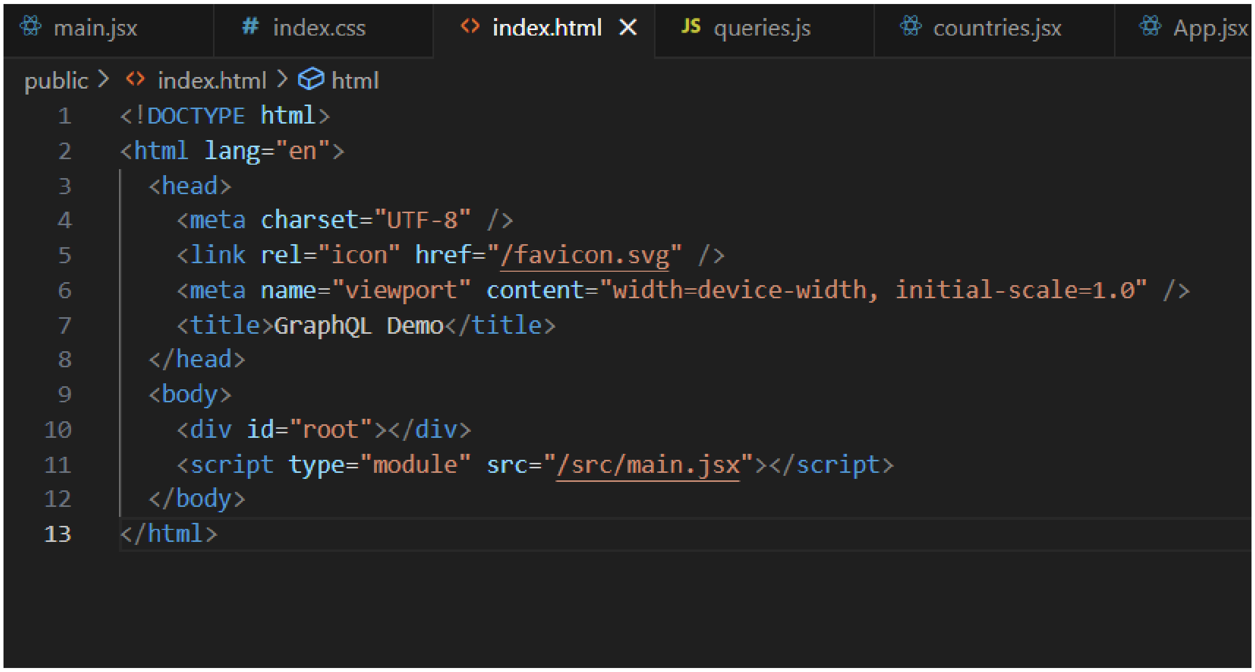
```
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LOVELY>npm create vite@latest graphql-demo

> npx
> create-vite graphql-demo

|
o  Select a framework:
|  React
|
o  Select a variant:
|  JavaScript + React Compiler
|
o  Use rolldown-vite (Experimental)?:
|  No
|
o  Install with npm and start now?
|  Yes
|
o  Scaffolding project in C:\Users\LOVELY\graphql-demo...
|
o  Installing dependencies with npm...

added 200 packages, and audited 201 packages in 19s

32 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
|
o  Starting dev server...

> graphql-demo@0.0.0 dev
> vite


  VITE v7.1.7  ready in 644 ms

  ➜  Local:   http://localhost:5173/
```



```
  → Local:   http://localhost:5173/
  → Network: use --host to expose
  → press h + enter to show help
u

  → Local:   http://localhost:5173/
  → Network: use --host to expose
code .
 q

q

C:\Users\LOVELY>
C:\Users\LOVELY>cd graphql-demo

C:\Users\LOVELY\graphql-demo>npm install

added 46 packages, and audited 201 packages in 3s

32 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Users\LOVELY\graphql-demo>npm install @apollo/client graphql

added 11 packages, and audited 212 packages in 47s

32 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Users\LOVELY\graphql-demo>npm install @apollo/client graphql

up to date, audited 212 packages in 6s

32 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

CODE:

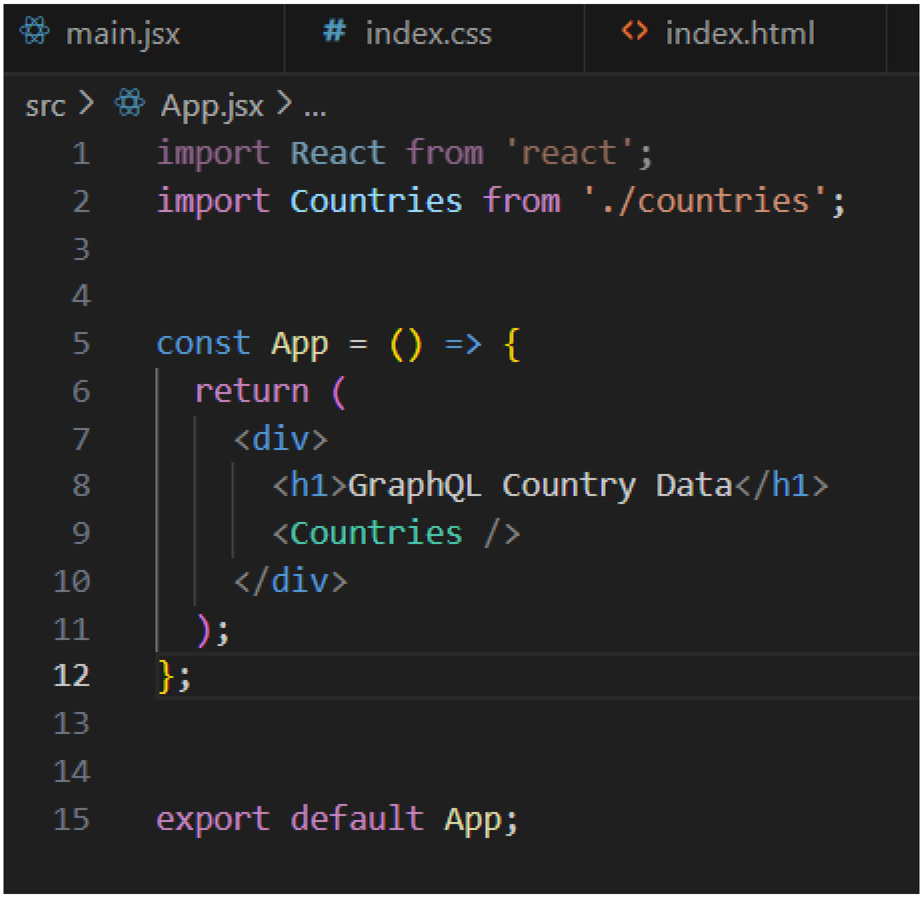
Index.html



```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" href="/favicon.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>GraphQL Demo</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" href="/favicon.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>GraphQL Demo</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

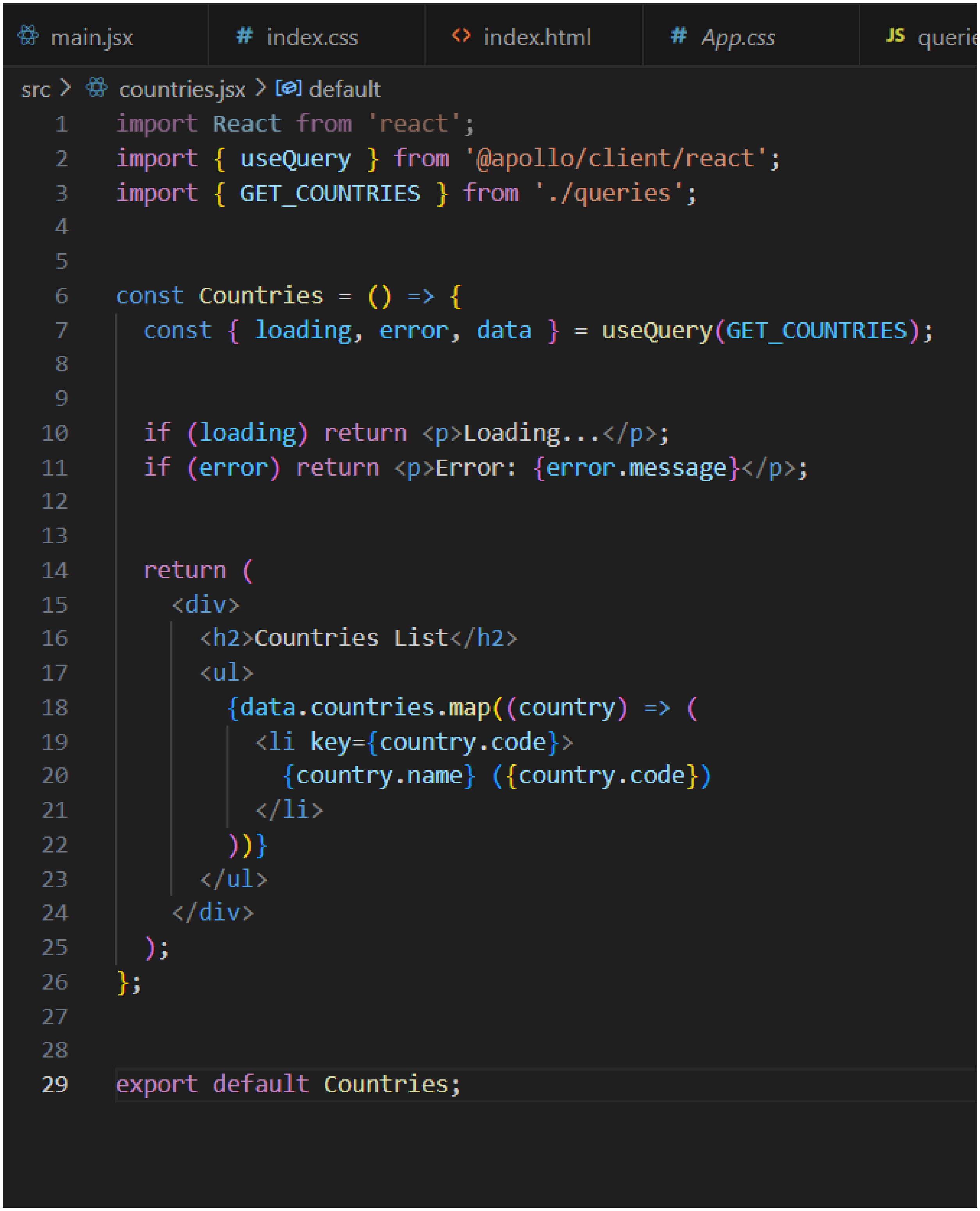
App.jsx



```jsx
import React from 'react';
import Countries from './countries';


const App = () => {
  return (
    <div>
      <h1>GraphQL Country Data</h1>
      <Countries />
    </div>
  );
};


export default App;
```

```jsx
import React from 'react';
import Countries from './countries';


const App = () => {
  return (
    <div>
      <h1>GraphQL Country Data</h1>
      <Countries />
    </div>
  );
};


export default App;
```

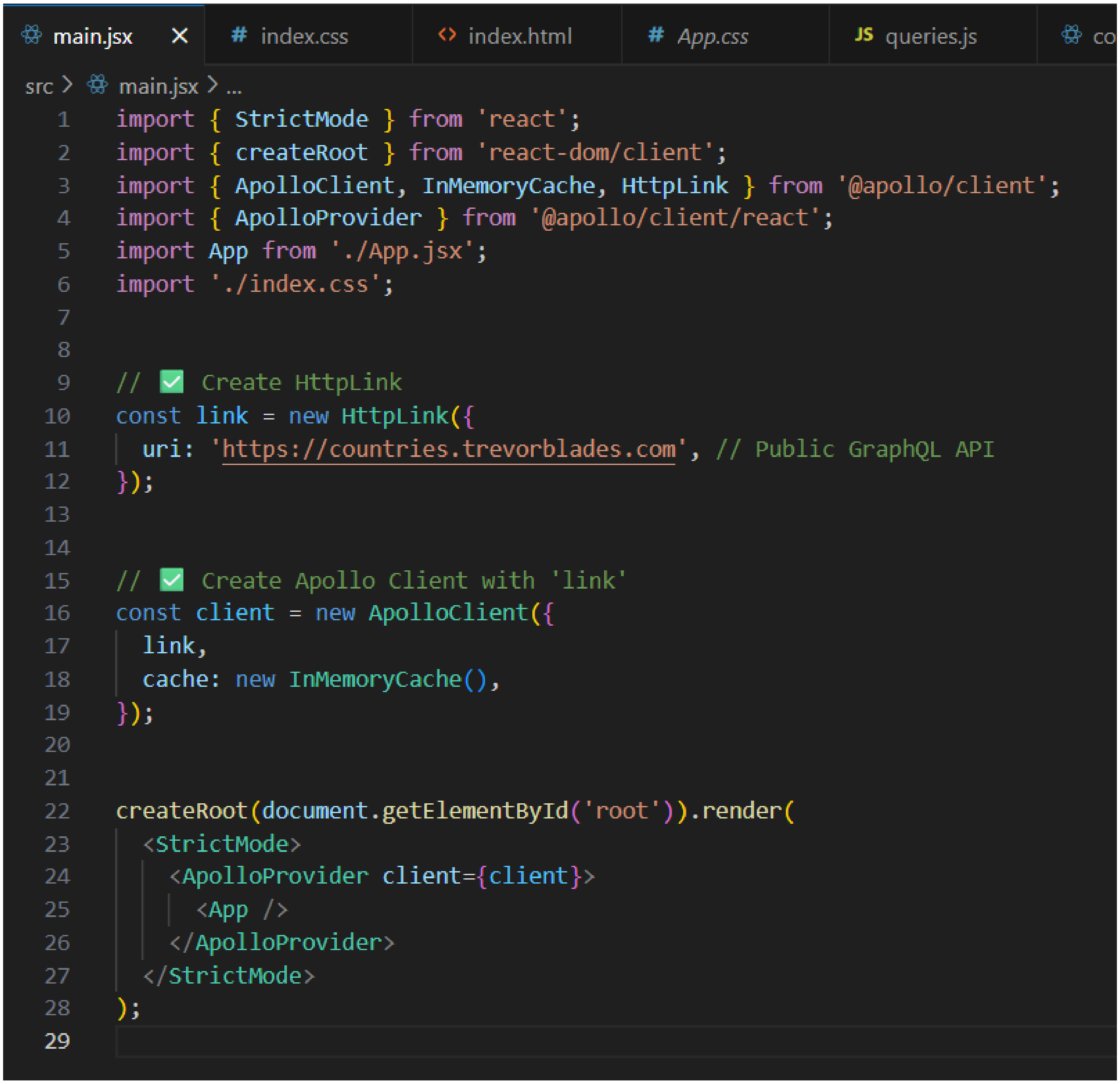
countries.jsx



```jsx
import React from 'react';
import { useQuery } from '@apollo/client/react';
import { GET_COUNTRIES } from './queries';


const Countries = () => {
  const { loading, error, data } = useQuery(GET_COUNTRIES);


  if (loading) return <p>Loading...</p>;
  if (error) return <p>Error: {error.message}</p>;


  return (
    <div>
      <h2>Countries List</h2>
      <ul>
        {data.countries.map((country) => (
          <li key={country.code}>
            {country.name} ({country.code})
          </li>
        ))}
      </ul>
    </div>
  );
};


export default Countries;
```

```
import React from 'react';

import { useQuery } from '@apollo/client/react';
import { GET_COUNTRIES } from './queries';

const Countries = () => {
  const { loading, error, data } = useQuery(GET_COUNTRIES);

  if (loading) return <p>Loading...</p>;
  if (error) return <p>Error: {error.message}</p>;

  return (
    <div>
      <h2>Countries List</h2>
      <ul>
        {data.countries.map((country) => (
          <li key={country.code}>
            {country.name} ({country.code})
          </li>
        ))}
      </ul>
    </div>
  );
};

export default Countries;
```

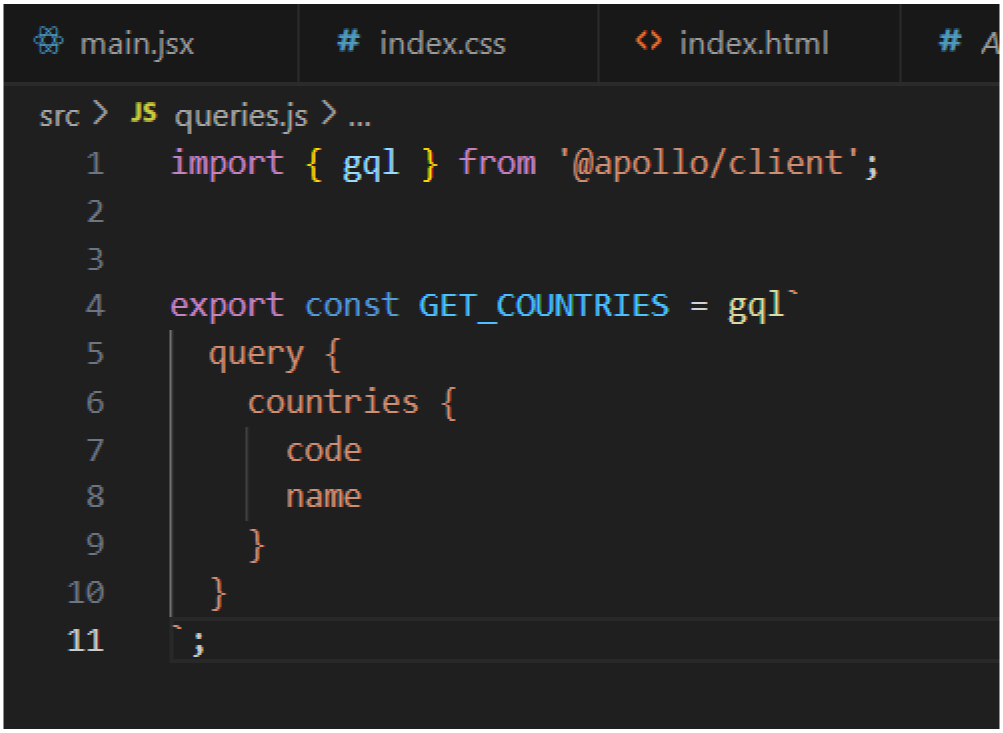
main.jsx



```jsx
import { StrictMode } from 'react';
import { createRoot } from 'react-dom/client';
import { ApolloClient, InMemoryCache, HttpLink } from '@apollo/client';
import { ApolloProvider } from '@apollo/client/react';
import App from './App.jsx';
import './index.css';


// ✅ Create HttpLink
const link = new HttpLink({
  uri: 'https://countries.trevorblades.com', // Public GraphQL API
});


// ✅ Create Apollo Client with 'link'
const client = new ApolloClient({
  link,
  cache: new InMemoryCache(),
});


createRoot(document.getElementById('root')).render(
  <StrictMode>
    <ApolloProvider client={client}>
      <App />
    </ApolloProvider>
  </StrictMode>
);

```

```
import { StrictMode } from 'react';

import { createRoot } from 'react-dom/client';

import { ApolloClient, InMemoryCache, HttpLink } from '@apollo/client';

import { ApolloProvider } from '@apollo/client/react';

import App from './App.jsx';

import './index.css';


const link = new HttpLink({
  uri: 'https://countries.trevorblades.com', // Public GraphQL API
});

const client = new ApolloClient({
  link,
  cache: new InMemoryCache(),
});

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <ApolloProvider client={client}>
      <App />
    </ApolloProvider>
  </StrictMode>
);
```

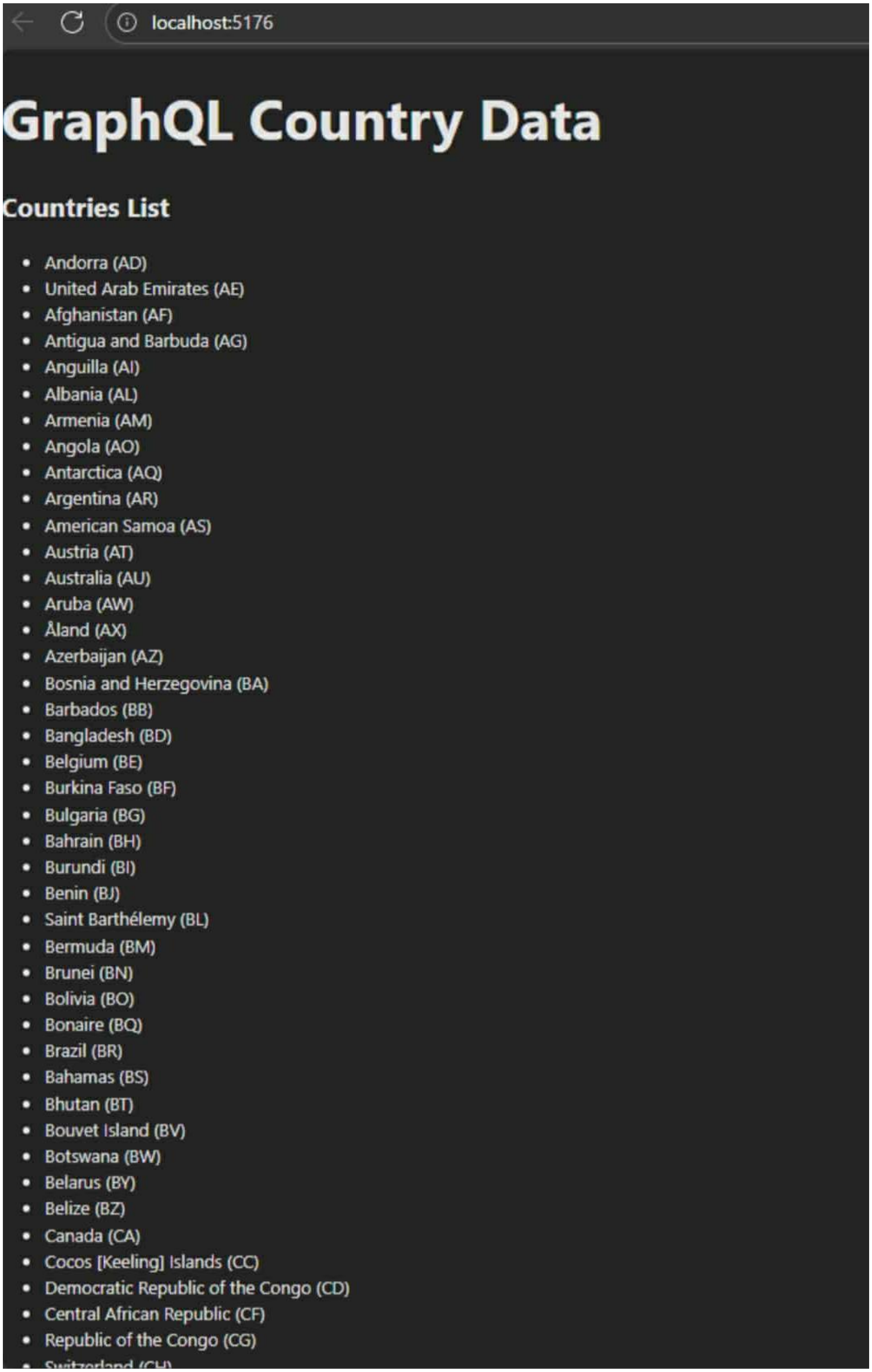
Queries.js



```
import { gql } from '@apollo/client';


export const GET_COUNTRIES = gql`
  query {
    countries {
      code
      name
    }
  }
`;
```

```
import { gql } from '@apollo/client';


export const GET_COUNTRIES = gql`
  query {
    countries {
      code
      name
    }
  }
`;
```

OUTPUT:



GraphQL Country Data

Countries List

- Andorra (AD)
- United Arab Emirates (AE)
- Afghanistan (AF)
- Antigua and Barbuda (AG)
- Anguilla (AI)
- Albania (AL)
- Armenia (AM)
- Angola (AO)
- Antarctica (AQ)
- Argentina (AR)
- American Samoa (AS)
- Austria (AT)
- Australia (AU)
- Aruba (AW)
- Åland (AX)
- Azerbaijan (AZ)
- Bosnia and Herzegovina (BA)
- Barbados (BB)
- Bangladesh (BD)
- Belgium (BE)
- Burkina Faso (BF)
- Bulgaria (BG)
- Bahrain (BH)
- Burundi (BI)
- Benin (BJ)
- Saint Barthélemy (BL)
- Bermuda (BM)
- Brunei (BN)
- Bolivia (BO)
- Bonaire (BQ)
- Brazil (BR)
- Bahamas (BS)
- Bhutan (BT)
- Bouvet Island (BV)
- Botswana (BW)
- Belarus (BY)
- Belize (BZ)
- Canada (CA)
- Cocos [Keeling] Islands (CC)
- Democratic Republic of the Congo (CD)
- Central African Republic (CF)
- Republic of the Congo (CG)
- Switzerland (CH)