

## Source

This document lists the external dependencies used by the project and explains how each dependency is used. It also includes the link to the repository.

## Repository

- GitHub repository: <https://github.com/manthranatarajan/CS4352-DesignProject-GlobalLink>

## Core runtime / build

- Node.js
  - URL: <https://nodejs.org/>
  - How it is used: Provides the runtime for development tools (npm) and the **react-scripts** dev server used to run and build the React app.
- npm (bundled with Node.js)
  - URL: <https://www.npmjs.com/>
  - How it is used: Installs JavaScript dependencies (**npm install**) and runs scripts (**npm start**, **npm build**).

## Front-end framework

- React
  - Package: **react** (and **react-dom**)
  - URL: <https://reactjs.org/>
  - How it is used: The UI is built as a React single-page application. Components in **src/** are React components that render the UI and handle navigation and state.
- react-router-dom
  - Package: **react-router-dom**
  - URL: <https://reactrouter.com/>
  - How it is used: Client-side routing between pages (**/**, **/signup**, **/login**, **/jobs**, **/user-search**, **/profile/:name**, etc.). Defined in **src/App.js** using **BrowserRouter**, **Routes**, and **Route**.

## Build tooling and CSS

- Create React App (**react-scripts**)
  - Package: **react-scripts**
  - URL: <https://create-react-app.dev/>
  - How it is used: Provides the development server (**npm start**), build pipeline (**npm run build**), and test runner used to run the project during development.
- Tailwind CSS

- Package: `tailwindcss`
- URL: <https://tailwindcss.com/>
- How it is used: Utility-first CSS framework used in JSX class names to style components. Configuration file: `tailwind.config.js`.
- PostCSS / Autoprefixer
  - Packages: `postcss`, `autoprefixer`
  - URL: <https://postcss.org/> and <https://github.com/postcss/autoprefixer>
  - How it is used: PostCSS processes Tailwind's output and adds vendor prefixes as needed during build.

## Testing libraries (dev / optional)

- `@testing-library/react`, `@testing-library/jest-dom`, `@testing-library/user-event`
  - URL: <https://testing-library.com/docs/react-testing-library/intro/>
  - How they are used: Unit and integration test helpers for React components (tests present under `src/` if added).

## Other libraries

- `web-vitals`
  - Package: `web-vitals`
  - URL: <https://github.com/GoogleChrome/web-vitals>
  - How it is used: Optional performance metrics collector included in CRA template (`reportWebVitals.js`).

## Where data is stored (project-specific)

- `public/users.json`
  - File path: `public/users.json`
  - How it is used: Static sample list of user names used by `src/pages/UserSearch.jsx` as one source for search results.
- Browser `localStorage`
  - How it is used: The project uses browser `localStorage` as a simple mock backend. Examples:
    - \* `src/App.js` seeds mock profiles into `localStorage` using keys like `Jane Doe_profile`.
    - \* `src/pages/SignupPage.jsx` saves new user profiles under `<username>_profile` (JSON string) and also sets `current_user`.
    - \* `src/pages/UserSearch.jsx` scans `localStorage` keys ending in `_profile` to include created accounts in search results.

## Exact versions (from `package.json`)

- `react`: `^19.2.0`

- react-dom: ^19.2.0
- react-router-dom: ^6.14.1
- react-scripts: ^5.0.1
- tailwindcss: ^3.4.18
- postcss: ^8.5.6
- autoprefixer: ^10.4.21
- testing libraries: versions present in `package.json` `devDependencies` / `dependencies`

## How to obtain the SDKs / packages

- Install Node.js (includes npm) from <https://nodejs.org/> and run in the project root:

`npm install`

- To run the development server:

`npm start`

- To build a production bundle:

`npm run build`

## Notes on usage in this project

- Because this is a prototype, the project uses `localStorage` for persistence. This is intended for development/demo purposes only; do not rely on it for secure or production storage.
- Passwords are stored in plaintext in `localStorage` in the current prototype. If you need stronger security, add a backend and never store plaintext passwords.

END OF SOURCE