ReadMe — GlobalLink (UI Prototype)

Overview

This repository contains a React-based UI prototype for a user/job search experience. The UI includes: - Welcome page - Login page - Signup (multi-step) page that saves a profile to `localStorage` - Job Feed and Job Preferences pages - User Search that merges `public/users.json` and profiles saved in `localStorage` - User Profile viewer for individual profiles

Quick start (requirements)

- OS: Windows (instructions below use PowerShell)
- Node.js and npm installed (https://nodejs.org/)
- Recommended browser: Chrome or Edge (latest versions). The app is a web SPA and works in modern browsers.

Install and run

1. Open PowerShell and change to the project directory (where `package.json` is located):

```
cd C:\Users\manth\Documents\GitHub\hand-controlled-media-player\CS4352-DesignProject-UI
npm install
npm start
```

2. The development server (Create React App) will start. Open `http://localhost:3000` in your browser.

Test user credentials (pre-seeded)

The app seeds several mock users in `localStorage` for testing. Use these example usernames and passwords to login from the Login page:

- Username: `Jane Doe` — Password: `12`
- Username: `John Paul` — Password: `34`
- Username: `Alice Land` — Password: `56`
- Username: `Bob Stark` — Password: `78`
- Username: `Charlie Brown` — Password: `90`

Notes about credentials and profiles

- Profiles are stored under the key `<username>_profile` in `localStorage` as a JSON string; `UserSearch` reads keys that end with `_profile` and includes them in search results.
- `Signup` stores the currently signed-in user under `current_user` and writes a profile object to `<username>_profile`.

- The signup flow also writes to the `username` key (older code saved password to this key; new code may overwrite it with the profile JSON). This is inconsistent but currently works for the prototype.

How the UI works (user flows)

- Welcome page (`/`): entry point with navigation to `Login` and `Signup`.
- Login (`/login`): checks `localStorage.getItem(username) === password` (legacy behavior), and then sets `localStorage.current_user` to the username and routes to `/jobs`.
- Signup (`/signup`): multi-step wizard that collects Basic Info, Work Experience, Education, Citizenship, Additional Info, Review & Submit. On submit, the profile is saved to `localStorage` under `<username>_profile` and the app navigates to `/job-preferences`.
- User Search (`/user-search`): fetches `public/users.json` (static) and also adds any localStorage entries with keys ending in `_profile` to the search results. Results are deduplicated case-insensitively.
- Profile (`/profile/:name`): opens a profile view for the selected user. The code first tries `localStorage.getItem(name)` and then falls back to `localStorage.getItem(name + '_profile')` for compatibility.

Where to inspect data in the browser

Open the browser DevTools → Application (Storage) → Local Storage → `http://localhost:3000`. You will see entries like: - `current_user` - `Jane Doe_profile` (JSON string) - other `_profile` keys

Open the static JSON directly while the dev server runs: - `http://localhost:3000/users.json`

Limitations and known issues

- Persistence: `localStorage` is used for demo only. Data is stored in the user's browser and will be lost if cleared.
- Security: Passwords are stored in plaintext (legacy behavior). This is not secure — do not use in production.
- Key inconsistency: The project currently mixes storing credentials and profiles under the same `username` key in some places; this can cause confusion.

END OF README