

Comprehensive Guide: Managing Subdomains and URL Redirect in a React + Express URL Shortener App

🌟 Overview

You're building a URL shortener app that:

- Uses **React** for frontend
 - Uses **Express/Node** for backend
 - Shortened URLs look like `http://url.localhost:5173/Of1kRrL8`
 - Your app handles redirection and subdomain-based routing
-

🐘 Objective

- ☒ Redirect short links via frontend (e.g., `url.localhost:5173/Of1kRrL8`)
 - ☒ Separate subdomains: one for main app (`www.localhost`) and one for shortened links (`url.localhost`)
 - ☒ Dynamically load different routers depending on subdomain
-

🐛 Subdomain Detection Logic

`/utils/helper.ts`

```
export const getSubDomain = (hostname) => {
  const cleanHost = hostname.split(":")[0]; // Removes port
  const parts = cleanHost.split(".");
  const isLocalhost = parts.slice(-1)[0] === "localhost";
  const sliceTill = isLocalhost ? -1 : -2;
  return parts.slice(0, sliceTill).join(".");
};

export const getApps = () => {
  const subdomain = getSubDomain(window.location.hostname);
  const mainApp = subDomainList.find((app) => app.main);
  if (subdomain === "") return mainApp.app;
  const matchedApp = subDomainList.find((app) => app.subDomain ===
subdomain);
  return matchedApp ? matchedApp.app : mainApp.app;
};
```

`/constants/subDomainList.ts`

```
import AppRouter from '../AppRouter';
import { subDomainRouter } from '../AppRouter';
```

```
export const subDomainList = [
  { subDomain: "www", app: AppRouter, main: true },
  { subDomain: "url1", app: subDomainRouter, main: false }
];
```

App Entry: App.jsx

```
import { getApps } from './utils/helper';

function App() {
  const CurrentApp = getApps();
  return <CurrentApp />; // Don't wrap in another <Router>
}

export default App;
```

Main App Router: AppRouter.js

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import LandingPage from './components/LandingPage';
import RegisterPage from './components/RegisterPage';
import LoginPage from './components/LoginPage';
import DashBoardLayout from './components/Dashboard/DashBoardLayout';

const AppRouter = () => (
  <Router>
    <Routes>
      <Route path="/" element={<LandingPage />} />
      <Route path="/register" element={<RegisterPage />} />
      <Route path="/login" element={<LoginPage />} />
      <Route path="/dashboard" element={<DashBoardLayout />} />
    </Routes>
  </Router>
);

export default AppRouter;
```

Subdomain Router: subDomainRouter

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import ShortenUrlPage from './components/ShortenUrlPage';

export const subDomainRouter = () => (
```

```

<Router>
  <Routes>
    <Route path="/:url" element={<ShortenUrlPage />} />
  </Routes>
</Router>
);

```

Redirection Component: ShortenUrlPage.jsx

```

import { useParams } from 'react-router-dom';
import { useEffect } from 'react';

const ShortenUrlPage = () => {
  const { url } = useParams();

  useEffect(() => {
    if (url) {
      window.location.href = `${import.meta.env.VITE_BACKEND_URL}${url}`;
    }
  }, [url]);

  return <p>Redirecting...</p>;
};

export default ShortenUrlPage;

```

Localhost Testing



To make subdomains like `url.localhost` work:

1. Edit your `/etc/hosts` file (or `C:\Windows\System32\drivers\etc\hosts` on Windows)
2. Add:

```

127.0.0.1 url.localhost
127.0.0.1 www.localhost

```

1. Restart dev server & test:
2. `http://www.localhost:5173`  Main app
3. `http://url.localhost:5173/0f1kRrL8`  Redirect

Final Recap

- Subdomain logic detects which router to load
- `subDomainRouter` handles short URL redirection

- `AppRouter` handles full app UI
- Host aliasing is essential on `localhost`

Let me know if you want deployment setup for real domains too!