OAuthLoginApp – Technical Documentation

Overview

OAuthLoginApp is a full-stack demo application that demonstrates Google OAuth 2.0 login using a React frontend and a Node.js + Express backend. It uses Passport.js for authentication and express-session for session management.

Architecture

- **Frontend:** React (client/)
- **Backend:** Node.js + Express (server/)
- **Authentication:** Auth0

Features

- Google Login integration
- Session-based authentication
- Displays logged-in user's name and profile picture
- Logout functionality
- Cross-origin support for frontend/backend on different ports

---

Folder Structure OAuthLoginApp--main/ client/ - React frontend server/ # Node.js + Express - Backend

---

Setup Instructions Prerequisites

- Node.js v14+
- Google Cloud project with OAuth 2.0 Client ID/Secret
- Add test user (Gmail) in Google OAuth consent screen

Installation

1. **Clone the repository:**

```
git clone https://github.com/yourusername/OAuthLoginApp.git
cd OAuthLoginApp--main
```

2. **Set up the backend:**

```
cd server
npm install
```

Create a .env file with your Google OAuth credentials and session secret

3. **Set up the frontend:**

```
cd ../client
npm install
```

4. **Start both servers:** Backend: npm start (in server/) Frontend: npm start (in client/)

Environment Variables Create a .env file in the server/ directory with

```
GOOGLE_CLIENT_ID=your-google-client-id
GOOGLE_CLIENT_SECRET=your-google-client-secret
GOOGLE_CALLBACK_URL=http://localhost:5000/auth/google/callback
SESSION_SECRET=your-session-secret
CLIENT_URL=http://localhost:3000
```

Authentication Flow

- User clicks "Login with Google" in the React app.
- Frontend redirects to backend /auth/google endpoint.
- Backend uses Passport.js to initiate Google OAuth flow.
- On success, Google redirects back to backend /auth/google/callback.
- Backend creates a session and redirects to frontend.
- Frontend fetches user info from backend and displays it.
- Logout destroys the session on the backend.

Backend Server (server/index.js)

- Uses Express for HTTP server.
- Uses CORS to allow requests from http://localhost:3000 with credentials.
- Loads environment variables from .env.
- Listens on port from process.env.PORT or 5000.

Demo Flow

- Visit http://localhost:3000
- Click "Login with Google"
- Authenticate with Google
- See your name and profile photo
- Click "Logout" to end session