Google Sheets Data Integration Application

Overview

This application is designed to integrate Google Sheets with a custom backend to fetch and manipulate spreadsheet data via an OAuth-based server-to-server integration. The backend is built using Scala, Http4s for the REST API, and optionally Cats Effect for handling asynchronous processing.

The application enables users to authenticate via Google OAuth and fetch data from Google Sheets. The data can be displayed on the frontend (built with React), allowing users to interact with multiple sheets, including pagination for displaying sheets and fetching the entire sheet's data.

System Architecture

The system is designed as a client-server application with the following components:

- Frontend: A React-based web application that allows users to view the available Google Sheets and select them to fetch data.
- Backend: A Scala-based API that integrates with Google Sheets API to fetch spreadsheet data. The backend uses Http4s for building REST endpoints, with Cats Effect for asynchronous processing.

Data Flow:

1. User authenticates via Google OAuth.
2. Frontend requests available Google Sheets from the backend.
3. Backend interacts with Google Sheets API to fetch and return the data.
4. Frontend displays the available sheets and allows fetching data from selected sheets.

OAuth Integration

OAuth Flow

Google OAuth is used to authenticate the user and obtain an access token that allows the backend to access Google Sheets API. The steps are as follows:

1. OAuth Consent Screen: The user is prompted to log in and grant permissions to access their Google Sheets.
2. Access Token Retrieval: After authentication, an access token is returned, which is stored locally and passed to the backend for API requests.

Backend Implementation

- The backend uses an OAuth client to generate the authentication URL and redirect the user to Google's OAuth consent screen.
- After successful authentication, the user is redirected back to the application with an access token, which is then used to make authorized API requests to Google Sheets.

Token Handling
Once the user is authenticated and the access token is obtained, it is stored in the application. The access token is passed in the Authorization header for each subsequent request to Google Sheets API.

Backend Development

Fetching Google Sheets
Once authenticated, the backend can fetch a list of Google Sheets accessible to the user. The /spreadsheets/:accessToken endpoint is designed to retrieve the list of sheets.

Fetching Sheet Data
The backend also provides functionality to fetch data from a specific sheet. The /spreadsheet/:accessToken/:spreadsheetId/data endpoint fetches data from a specific sheet in the spreadsheet and stores in locally hosted PostgreSQL database.

Frontend Development

Fetching Spreadsheets List
The frontend displays the available spreadsheets fetched from the backend. It uses React hooks to manage state and fetch data.

Pagination
To improve the user experience, pagination was added to handle cases where the user has many spreadsheets. The frontend fetches sheets in batches of 10.