Software Design Document

SDG-Research-Connector (working title)

Internal name: SRC

Version 0.2

Last updated: 2025-12-03

Product owner: Eva Witesman

Project lead: Matt Cooper

Lead developer: Christian Taylor

Junior developer: Levi Henstrom

Revisions

Version	Primary Author	Description	Date Completed
0.1	Matt Cooper	Initial system design and data modeling	11/14/25
0.2	Matt Cooper	Minor formatting changes	12/3/25

Table of Contents

1. Introduction

1.1 Purpose

This Software Design Document describes the architecture and system design for SRC, a research connecting tool to index papers and profiles from open sources, classify them to UN SDGs (Sustainable Development Goals), and recommend cross-university collaborators with feedback loops. SRC is designed for BYU faculty and staff to make connections on research to further the reach and relevance of the UN's Sustainable Development Goals.

1.2 Scope

This document outlines the implementation details of the SRC Web Application. SRC will have three main features:

- **Semantic Search** of currently available research that connects in some way to one or many SDGs.
- **Tailored Recommendations** for top-k similar authors across institutions using cosine similarity on embeddings, constrained by shared SDGs.
- **Admin and Feedback Systems** to tag corrections and export CSVs or data.

1.3 Definitions, Acronyms and Abbreviations

Acronym	Definition
SRC	SDG Research Connector
SDG	Sustainable Development Goal

1.4 References

Quick start on Supabase with ReactJS

https://supabase.com/docs/guides/getting-started/quickstarts/reactjs

50,000 MAUs based which should be plenty to start
https://supabase.com/docs/guides/platform/manage-your-usage/monthly-active-users-third-party

Setting up SAML with Supabase (BYU SSO)

https://supabase.com/docs/guides/auth/enterprise-sso/auth-sso-saml#managing-saml-20-connections

2. System Overview

2.1 Tech Stack

Frontend

React 18 with Vite

Tailwind CSS 3 for styling

React Router 6 for navigation

Supabase JS Client for database and auth

Recharts for visualizations

Backend

Supabase

- PostgreSQL with pgvector extension
- Auto-generated REST and maybe GraphQL APIs
- Row-level security (RLS)
- Built-in authentication with SSO support via SAML
- Storage buckets for PDFs and exports

ML & Classification

Vercel Python Serverless Functions

Sentence Transformers (all-MiniLM-L6-v2 model)

OSDG classifier for SDG tagging

Hugging Face Transformers library

Analytics

Power BI Pro (direct Postgres connection)

Supabase Studio (database monitoring)

Hosting

Vercel for React frontend and Python serverless functions

Supabase for backend services (free tier)

Developer Tools

VS Code

VS Code Extensions (Jest, Live Server, ESLint, Prettier)

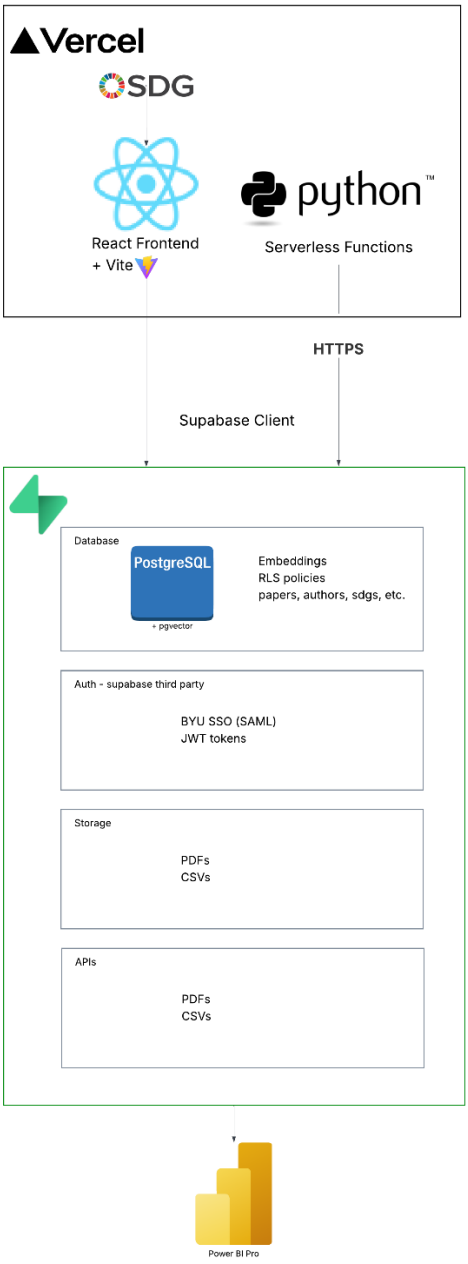
GitHub for version control

GitHub Actions for CI/CD

Supabase CLI for migrations and local development

Postman for API testing

2.2 Design Diagram



3. System Components

3.1 Decomposition Description

Supabase

Project URL: https://lhygdhwhhkpuuakkwfgv.supabase.co

Anon public key:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImxoeWdkaHdoaGtwdXVha2t3Zmd2Iiwicm9sZSI6ImFub24iLCJpYXQiOjE3NjU0MDYyOTksImV4cCI6MjA4MDk4MjI5OX0.Ggu8Ho7cDQnAW91UrUzwr1WszUo84nt8_sPtGJWmGKQ

Connection string:
postgresql://postgres:[YOUR_PASSWORD]@db.lhygdhwhhkpuuakkwfgv.supabase.co:5432/postgres

Vercel

Vercel Team: Matt's projects – hobby

Project Name: src

Framework Preset: Vite

Root Directory: ./

Build Command: npm run build

Output Directory: dist

Environment Variables:
VITE_SUPABASE_URL = https://lhygdhwhhkpuuakkwfgv.supabase.co

VITE_SUPABASE_ANON_KEY =
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImxoeWdkaHdoaGtwdXVha2t3Zmd2Iiwicm9sZSI6ImFub24iLCJpYXQiOjE3NjU0MDYyOTksImV4cCI6MjA4MDk4MjI5OX0.Ggu8Ho7cDQnAW91UrUzwr1WszUo84nt8_sPtGJWmGKQ

3.2 Core Features

MVP functionality

- Search papers by SDG

- View paper details
- Get author recommendations for collaboration
- View recommendations for collaboration

Stretch functionality

- Feedback system to correct SDG classification
- Admin workflows

3.3 User Interfaces (GUI)

3.3.1 Admin Page

3.3.2 User Page

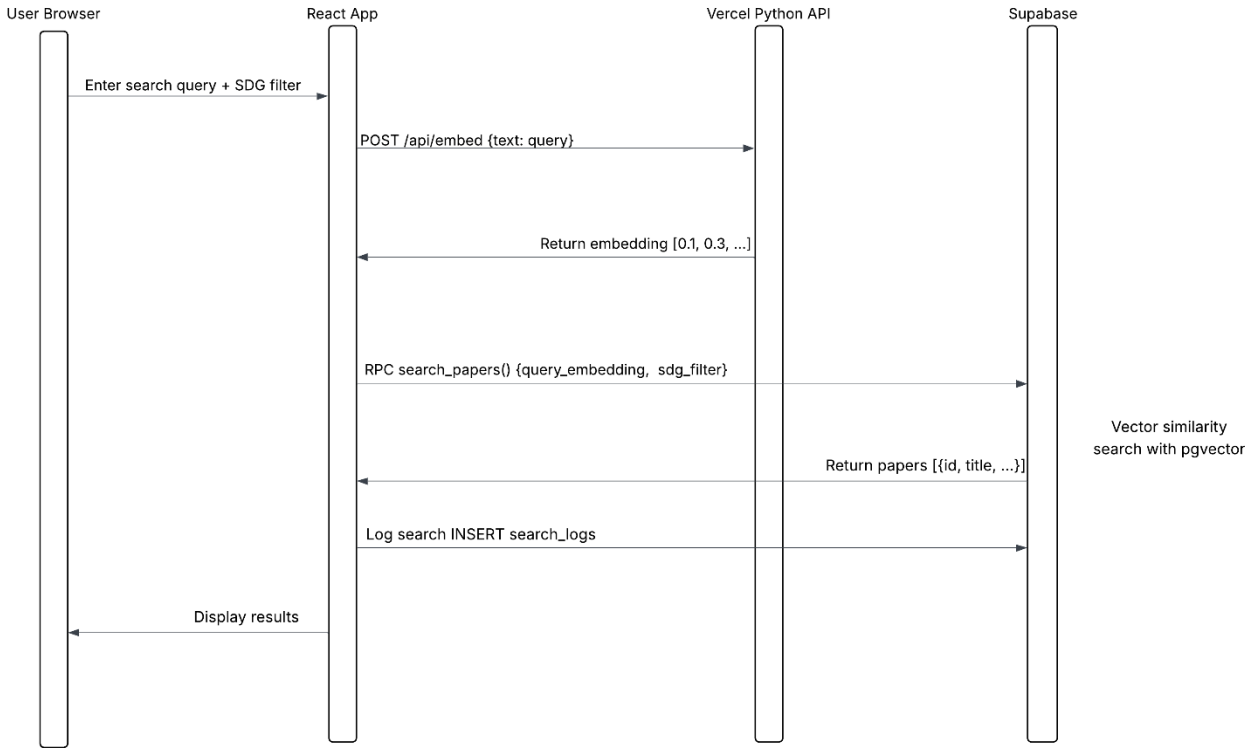
3.3.3 UI Design Description

3.4 Interface Description

4. Detailed Design

4.1 Module Detailed Design

Sequence Diagram – Paper Search Flow



Pseudocode

User enters search query "climate change adaptation" and selects SDG 13

React captures input and SDG filter

- Request embedding from Vercel Python API with query text
- Python API generates embedding using Sentence Transformers
- Python returns embedding vector [0.23, -0.15, 0.87, ...]

React calls Supabase RPC function search_papers()

- Pass query_embedding and sdg_filter=[13]

Supabase executes vector similarity search

- Filter papers WHERE paper_sdgs.sdg_id = 13
- Order by embedding <=> query_embedding (cosine distance)

Return top 20 results

React logs search to search_logs table

React displays results to user

Sequence Diagram - Paper Indexing Flow

Pseudocode

Admin uploads paper metadata via form

Title: "Climate Resilience in Agricultural Systems"

Abstract: "This paper examines..."

DOI: "10.1234/example"

Authors: ["Jane Smith", "John Doe"]

React sends abstract to /api/embed

Python generates embedding using all-MiniLM-L6-v2

Returns 384-dimensional vector

React sends abstract to /api/classify

Python runs OSDG classifier

Returns SDGs: [{id: 2, confidence: 0.87}, {id: 13, confidence: 0.92}]

React inserts paper into Supabase

INSERT INTO papers (title, abstract, doi, embedding)

Get returned paper_id

React inserts SDG mappings

For each SDG with confidence >= 0.7

INSERT INTO paper_sdgs (paper_id, sdg_id, confidence_score)

React inserts author relationships

For each author name

Find or create author in authors table

INSERT INTO author_papers (author_id, paper_id, author_order)

React displays success message

Sequence Diagram - Recommendation Generation Flow

Pseudocode

Every Sunday at 2 AM, Vercel Cron triggers recommendation job

Function fetches all authors from Supabase

SELECT id, name, embedding FROM authors WHERE embedding IS NOT NULL

For each author:

Call Supabase RPC find_similar_authors()

Pass author_id and optional sdg_filter

Supabase calculates cosine similarity (1 - embedding <=> other.embedding)

Supabase filters: only authors with shared SDGs

Supabase returns top 10 similar authors

Delete existing recommendations for this author

DELETE FROM recommendations WHERE source_author_id = author_id

Insert new recommendations

For each similar author:

INSERT INTO recommendations

(source_author_id, target_author_id, similarity_score, shared_sdgs, expires_at)

Set expires_at = NOW() + INTERVAL '7 days'

Function logs completion

Total authors processed: 1,247

Total recommendations generated: 12,470

Execution time: 4.3 minutes

Sequence Diagram - User Authentication Flow

Pseudocode

User clicks "Sign in with BYU" button

React calls supabase.auth.signInWithSSO({provider: 'saml', domain: 'byu.edu'})

Supabase redirects to BYU CAS login page

User enters BYU NetID and password on BYU site

BYU validates credentials

BYU redirects back to app with SAML assertion

Supabase validates SAML token

Supabase creates JWT token with user data

  Token contains: user_id, email, name, institution

React receives JWT and stores in localStorage

React checks if user exists in users table

  SELECT * FROM users WHERE id = auth.uid()

If user doesn't exist:

  INSERT INTO users (id, email, name, institution_id, byu_net_id)

  Values from JWT token

React sets user session

React redirects to dashboard

All future API calls include JWT in Authorization header

Sequence Diagram - View Recommendation Details

Pseudocode

User views their recommendations on dashboard

List shows: Author name, institution, similarity score, shared SDGs

User clicks on recommendation for "Dr. Jane Smith"

React logs the view event

```
INSERT INTO recommendation_views
  (recommendation_id, user_id, viewed_at, action_taken)
```

React fetches author details

```
SELECT authors.*, institutions.name
FROM authors JOIN institutions
WHERE authors.id = target_author_id
```

React fetches author's papers filtered by shared SDGs

```
SELECT papers.* FROM papers
JOIN author_papers ON papers.id = author_papers.paper_id
JOIN paper_sdgs ON papers.id = paper_sdgs.paper_id
WHERE author_papers.author_id = target_author_id
  AND paper_sdgs.sdg_id IN (shared_sdgs_array)
```

React displays profile modal

```
Shows: Name, institution, email, profile link
Shows: "You both work on SDG 3 and SDG 13"
Shows: List of relevant papers
```

User clicks "Contact" button

React updates recommendation_views

```
UPDATE recommendation_views SET action_taken = 'email_sent'
WHERE id = view_id
```

React opens mailto: link

```
To: jane.smith@example.edu
Subject: "Research Collaboration Opportunity - SDG 3"
Body: Pre-filled template with shared research interests
```

File Structure

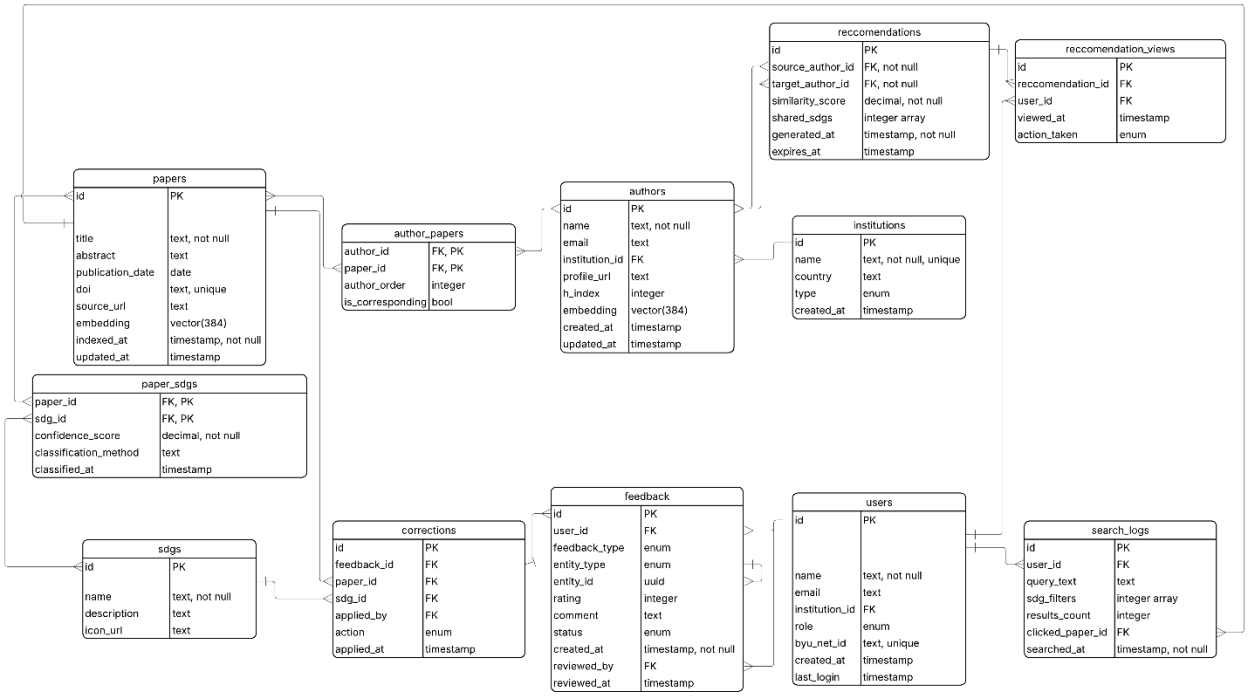
Project Structure

```
src-research/
├──── .github/
│    └──── workflows/
│         └──── ci.yml
├──── api/                  # Vercel serverless functions
│    ├──── embed.py          # Generate embeddings
│    ├──── classify.py       # OSDG classification
│    └──── requirements.txt
├──── supabase/
│    ├──── migrations/
│    │    └──── 001_initial_schema.sql
│    └──── config.toml
├──── src/                  # React app
│    ├──── components/
│    │    ├──── SearchBar.jsx
│    │    ├──── PaperCard.jsx
│    │    ├──── RecommendationList.jsx
│    │    └──── AuthButton.jsx
│    ├──── pages/
│    │    ├──── SearchPage.jsx
│    │    ├──── RecommendationsPage.jsx
│    │    └──── PaperDetailPage.jsx
│    ├──── lib/
```

```
│   │   └── supabase.js      # Supabase client
│   ├── utils/
│   │   └── api.js           # API helpers
│   ├── App.jsx
│   └── main.jsx
├── public/
├── .env.local               # Gitignored
├── .env.example
├── .gitignore
├── index.html
├── package.json
├── vite.config.js
├── tailwind.config.js
├── postcss.config.js
└── README.md
```

4.2 Data Detailed Design

Entity Relationship Diagram



3. Add Component Specifications

Add to section 3.3.3:

```

UI Components

SearchBar Component

- Input field for search query

- Multi-select dropdown for SDG filters

- Search button

- Props: onSearch(query, sdgFilters)

PaperCard Component

- Display: title, abstract (truncated), authors, SDG badges
```

- Click to view details
- Props: paper {id, title, abstract, authors[], sdgs[]}

RecommendationList Component
- List of recommended authors
- Shows: name, institution, similarity score, shared SDGs
- Click to view author details
- Props: recommendations[]

AuthButton Component
- "Sign in with BYU" button
- User profile dropdown when authenticated
- Props: user, onSignIn(), onSignOut()
```

**4. Add API Endpoints Section**

Add to section 3.4:
```
API Endpoints

Vercel Python Functions:
POST /api/embed
  Body: { text: string }
  Returns: { embedding: number[] }
```
```

POST /api/classify

  Body: { text: string }

  Returns: { sdgs: [{id: number, confidence: number}] }

Supabase RPC Functions:

search_papers(query_embedding vector, sdg_filter int[], limit_count int)

  Returns: papers with similarity scores

find_similar_authors(query_author_id uuid, sdg_filter int[], limit_count int)

  Returns: similar authors with scores

```

**5. Add Environment Variables**

Already have this in 3.1, but make it clearer:

```
Required Environment Variables

Frontend (.env.local):

VITE_SUPABASE_URL=https://lhygdhwhhkpuuakkwfgv.supabase.co

VITE_SUPABASE_ANON_KEY=eyJhbGci...

Vercel (set in dashboard):

VITE_SUPABASE_URL

VITE_SUPABASE_ANON_KEY

GitHub Secrets (for CI/CD):

SUPABASE_ACCESS_TOKEN

VERCEL_TOKEN

```

**6. Clarify MVP Scope**

Update section 3.2 to be crystal clear:

```

MVP Features (Must Have)

✓ Search papers by SDG filter

✓ Semantic search with text query

✓ View paper details (title, abstract, authors, SDGs)

✓ BYU SSO authentication

✓ View personalized author recommendations

✓ Click recommendation to see author profile

✓ Analytics logging (search_logs, recommendation_views)

Excluded from MVP (Add Later)

✗ Feedback system

✗ Admin workflows

✗ Bulk paper upload

✗ Email notifications

4.3 RTM

Requirement ID	Requirement Description	Design Component	Test Case #