# Project Documentation: Report Management System

# 1. Project Overview

## 1.1. Problem Statement

The current manual and decentralized methods of sharing reports within organizations lead to scattered information, ineffective communication, and delays in decision-making. This project aims to address these issues by developing a centralized, web-based platform for the efficient management, storage, and sharing of reports.

## 1.2. Objectives

* To create a unified, centralized platform for the uploading, secure storage, and retrieval of reports.
* To enhance communication and collaboration through features like commenting and granular access control.
* To minimize delays in the dissemination of reports, facilitating quicker decision-making.
* To enable users to create data visualizations from structured report data to gain insights.

# 2. Functional Specification

## 2.1. User Roles & Permissions

The system will feature a tiered role-based access control model to manage user privileges and responsibilities.

|  |  |
| --- | --- |
| Role | Description |
| **Superadmin** | The highest-level role. Responsible for creating and managing different organizations on the platform. |
| **Admin** | Manages users and groups within a single organization. Can invite new users and assign permissions. |
| **User** | The standard user role. Can upload, view, and share reports based on assigned permissions. |

Access to individual reports is further controlled by specific permission levels:

|  |  |
| --- | --- |
| Permission Level | Description |
| **Viewer** | Can view a report. |
| **Commenter** | Can view and add comments to a report. |
| **Editor** | Can view, comment on, and edit a report. |
| **Owner** | Full control over a report, including editing, sharing, and deleting. |

## 2.2. Authentication Flows

### 2.2.1. Initial Admin Authentication (New Organization)

A Superadmin will manually create the first Admin account for a new organization. This new Admin receives an email invitation to set their password and log in for the first time.

### 2.2.2. Admin-Invited User Authentication

1. An Admin logs in and navigates to the "Manage Users" section.
2. The Admin enters the new user's email address and assigns their role (e.g., "User").
3. The system generates a single-use invitation token and sends an email to the user with a registration link.
4. The new user clicks the link, is prompted to set their password, and is automatically added to the organization.

### 2.2.3. Forgot Password Flow

1. The user clicks "Forgot Password" on the login page and enters their email address.
2. If the email exists, the system sends a secure, time-limited password reset token to their email.
3. The user clicks the link, which directs them to a password reset page to enter a new password. The token is then invalidated.

## 2.3. Core Features

### 2.3.1. Multi-Tenancy & Data Isolation

* The system will operate as a single instance serving multiple organizations.
* Every relevant database record (Users, Reports, Groups, Dashboards) **must** be associated with an organization\_id.
* All data queries from the front-end **must** be scoped to the user's organization\_id to ensure strict data isolation.

### 2.3.2. Report Management

* Users can upload various file types (.docx, .pdf, .xlsx, etc.).
* Reports are stored in Supabase Storage.
* The UI will have a folder-based file browser for intuitive organization.
* Report sharing is managed via a permissions modal, where Owners and Admins can assign access to individuals or groups.

### 2.3.3. Group-Based Access Control

* Admins can create and manage user groups (e.g., "Marketing Team," "Finance Department").
* Permissions can be assigned to an entire group, simplifying access management for teams.

### 2.3.4. Data Visualization

* A dedicated dashboard builder will allow users to create visuals from structured data (.csv, .xlsx).
* The builder will be a single-page application with a drag-and-drop interface.
* It will support a variety of chart types (bar charts, line graphs, pie charts) and provide customization options.
* The dashboard configuration will be saved in the database as a JSON object, separate from the source report.

# 3. Technical Specification

## 3.1. Technology Stack

* **Frontend:** React
* **Backend:** Node.js
* **Database, Auth & Storage:** Supabase

## 3.2. Database Schema

* **organizations**
  + organization\_id (PK, UUID)
  + organization\_name (TEXT)
* **users**
  + user\_id (PK, UUID, from Supabase Auth)
  + full\_name (TEXT)
  + email (TEXT)
  + role\_id (FK, INTEGER, from roles)
  + organization\_id (FK, UUID, from organizations)
* **roles**
  + role\_id (PK, INTEGER)
  + role\_name (TEXT, e.g., 'Superadmin', 'Admin', 'User')
* **groups**
  + group\_id (PK, UUID)
  + group\_name (TEXT)
  + organization\_id (FK, UUID, from organizations)
* **reports**
  + report\_id (PK, UUID)
  + file\_name (TEXT)
  + file\_path (TEXT, path in Supabase Storage)
  + owner\_id (FK, UUID, from users)
  + organization\_id (FK, UUID, from organizations)
  + permissions (JSONB, stores user/group permission levels)
* **dashboards**
  + dashboard\_id (PK, UUID)
  + dashboard\_name (TEXT)
  + owner\_id (FK, UUID, from users)
  + organization\_id (FK, UUID, from organizations)
  + source\_reports (ARRAY of UUIDs)
  + layout\_config (JSONB, stores chart types, data mappings, and positions)

## 3.3. API Specification

The following are key API endpoints. The base URL is api/v1/.

|  |  |  |
| --- | --- | --- |
| **Endpoint** | **Method** | **Description** |
| /auth/login | POST | Authenticates a user. |
| /auth/invite-user | POST | Admin invites a new user. |
| /auth/register | POST | User completes registration with a token. |
| /reports/upload | POST | Uploads a new report file. |
| /reports/{report\_id} | GET | Retrieves a single report. |
| /reports/{report\_id}/permissions | POST | Updates report permissions. |
| /reports/data-sources | GET | Fetches reports with structured data. |
| /dashboards/create | POST | Saves a new dashboard configuration. |
| /dashboards/{dashboard\_id} | GET | Retrieves a dashboard's configuration. |

# 4. UI/UX Design

The user interface will be based on the provided Figma designs.

* The designs for the Admin and User dashboards, report management, and analytics are approved.
* The final implementation will include refinements for the Groups management UI and a separate dashboard for the Superadmin role, as previously discussed.
* The mobile UI will be a simplified, view-only experience for reports and dashboards, and will not include the dashboard builder.

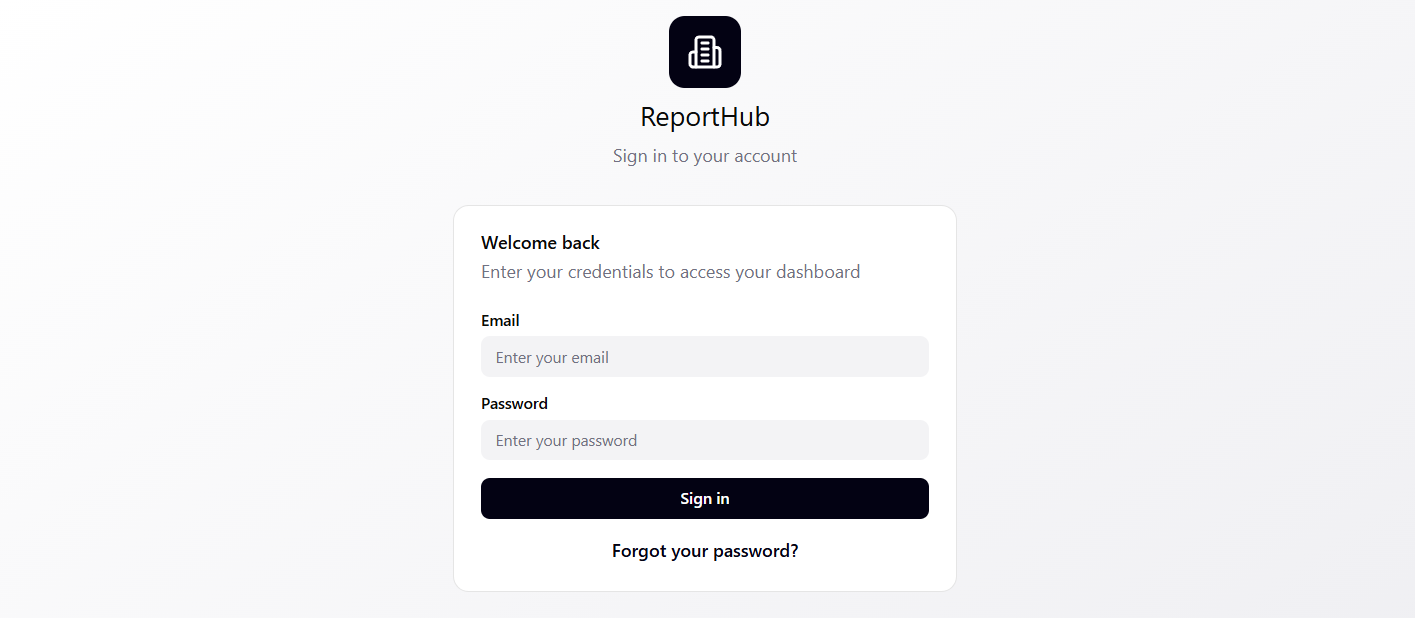


Figure 1 Authentication Screen

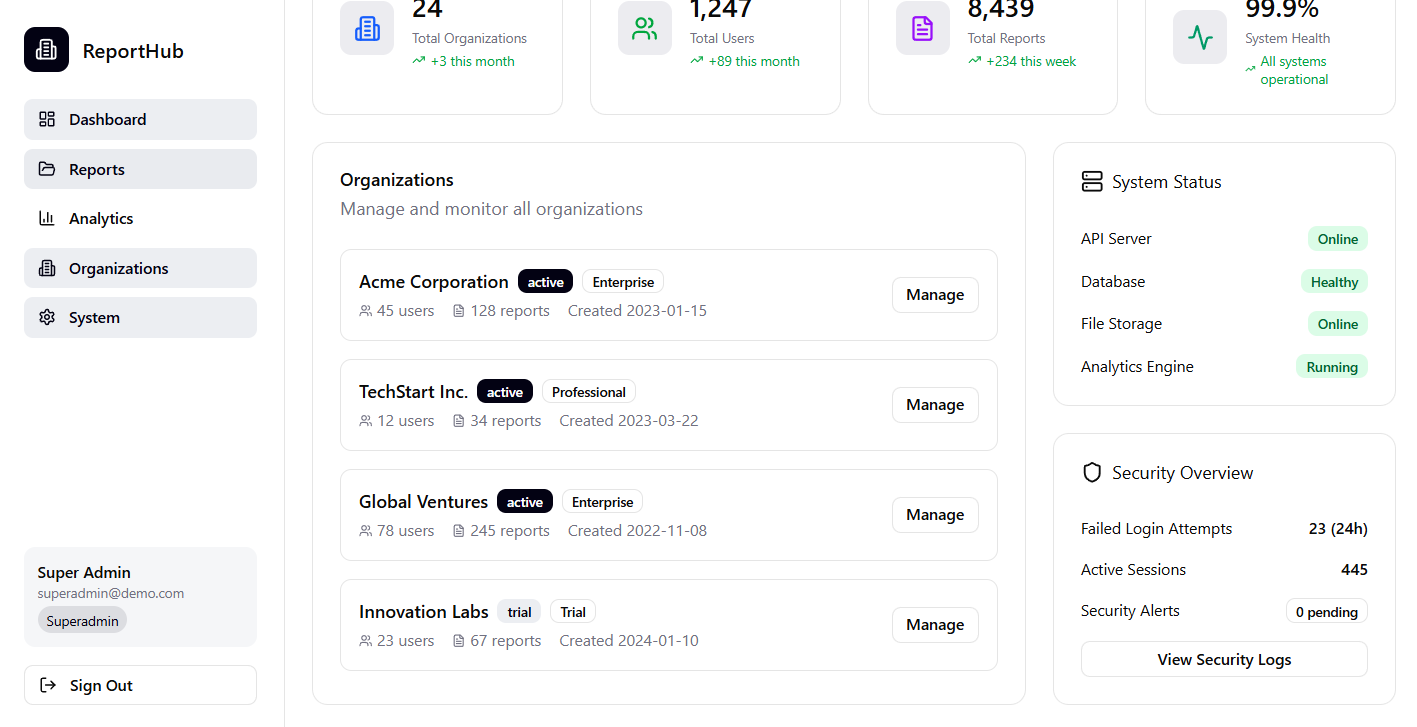


Figure 2 System Overview (super admin)

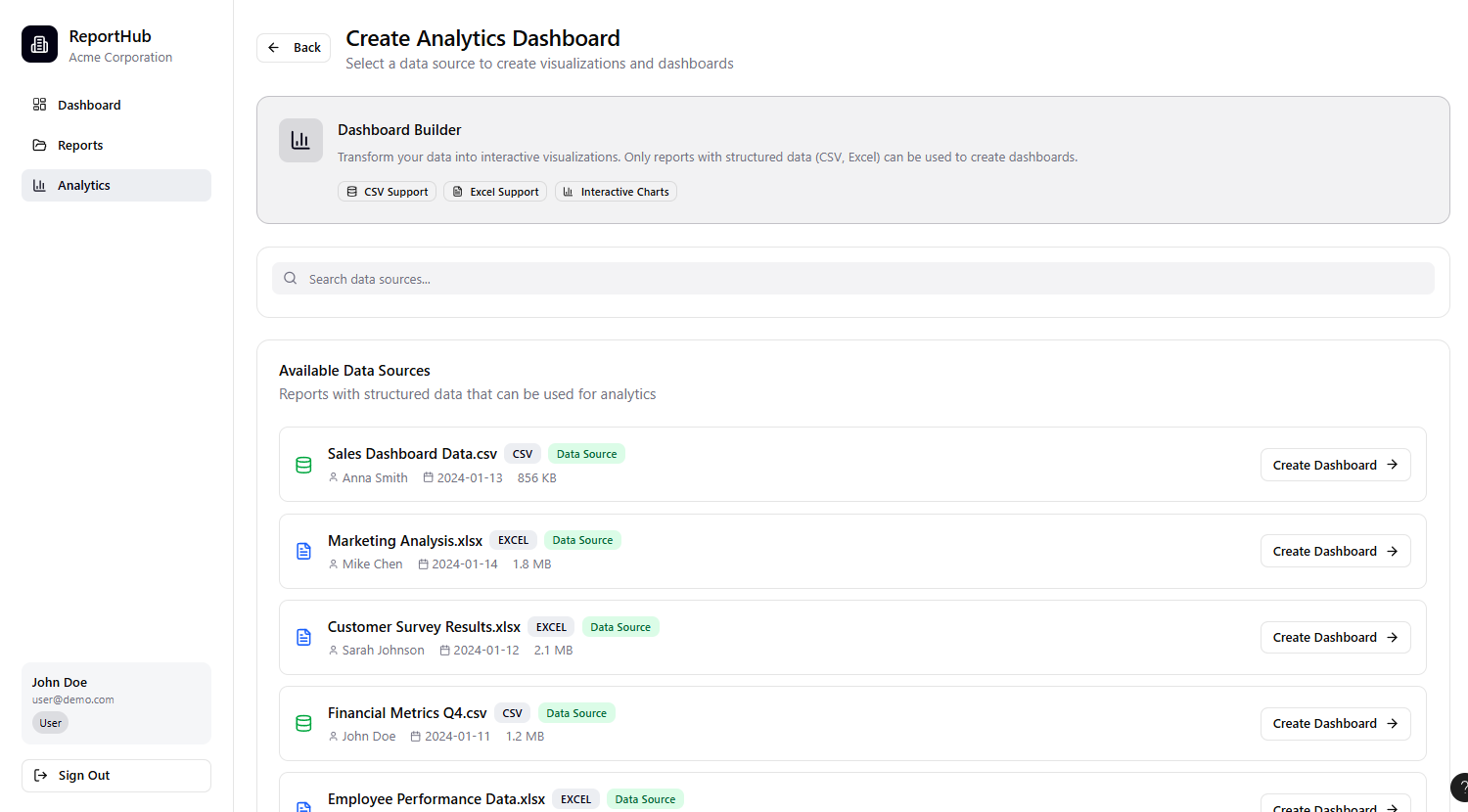


Figure 3 Analytics Page (users)

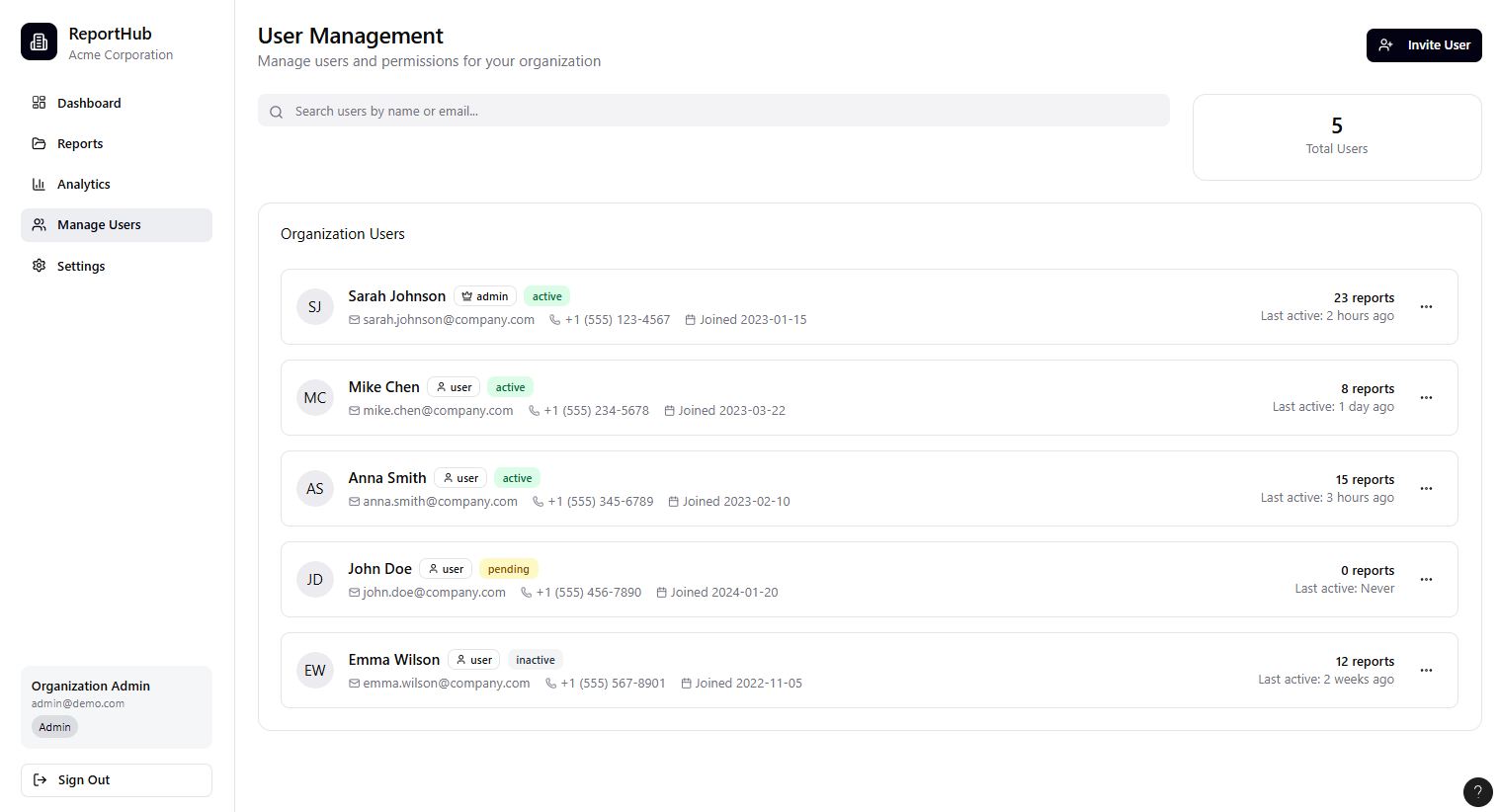


Figure 4 User Management

# 5. Project Roadmap (Agile Methodology)

## Sprint 1: Core Functionality (Weeks 1-3)

* **Objective:** Establish core authentication and basic file management.
* **Tasks:**
  + Set up the Supabase project and database.
  + Implement organizations and users tables.
  + Build user login, registration (with invitation), and password reset flows.
  + Implement file upload and retrieval API endpoints and UI.

## Sprint 2: Collaboration & Permissions (Weeks 4-6)

* **Objective:** Implement security and collaboration features.
* **Tasks:**
  + Refine the database schema to include groups and the permissions JSONB field.
  + Develop the "Manage Users" and "Groups" UI for Admins.
  + Implement the report sharing modal and its API.
  + Build the Admin dashboard.

## Sprint 3: Data Visualization (Weeks 7-10)

* **Objective:** Develop the dashboard builder.
* **Tasks:**
  + Build the Dashboard Builder UI.
  + Implement backend logic for data extraction from structured files.
  + Create API endpoints for saving and retrieving dashboard configurations.

## Sprint 4: Mobile & Refinements (Weeks 11-12+)

* **Objective:** Optimize the application and address final refinements.
* **Tasks:**
  + Ensure the web app is responsive for mobile and tablet use.
  + Build the view-only mobile UI.
  + Implement the Superadmin dashboard and its features.
  + Integrate any final security settings and notifications.