Member Data Manager (Project Y)

Version: 1.0.0

Repository: Dev-Api-Org/Project-Y-Member-Manager

Tech Stack: Microsoft Power Platform, Azure SQL, Power BI, Python

1. Executive Summary

The **Member Data Manager** is a bespoke operational tool developed to support the **Project Y** initiative.

Historically, Project Y relied on decentralized spreadsheets to track member enrollment and progress, leading to data redundancy and version control issues. This solution replaces those manual processes with a centralized, cloud-based application. It provides a secure interface for administration to manage the full member lifecycle—from initial enrollment and team assignment to project scoring and career track progression.

2. Development Team: Dev-Api-Org

This solution was engineered by the **Dev-Api-Org** team, leveraging a full-stack approach to low-code development. Responsibilities were distributed as follows:

- **Dingaan** | *Database Architecture & Business Intelligence*

    - Designed the normalized **Azure SQL Database** schema to ensure data integrity.

    - Developed the **Power BI Dashboards** to visualize key performance indicators (KPIs) and program metrics.

- **Olifile** | *Data Engineering & Pipeline*

    - Architected the **Data Pipeline** (ETL) to prepare operational data for analytical reporting.

    - Assisted with the initial database configuration and provisioning.

    - Managed the generation and population of **dummy data** to facilitate robust testing during the development phase.

- **Seward** | *Application Development (Power Apps)*

    - Led the design and implementation of the **Power Apps User Interface (UI)**.

- o   Managed the connectivity layer, ensuring the app securely communicates with the Azure SQL backend.

- o   Implemented client-side validation and navigation logic.

3. Development Methodology

The project adhered to **Agile methodologies** to ensure rapid delivery and flexibility.

- **Sprints:** The development lifecycle was divided into **one-week sprints**. This allowed for rapid feedback loops between the developers and project stakeholders.

- **Version Control: GitHub** was utilized as the central source of truth. The team used version control to manage the .msapp solution files and SQL scripts, ensuring that database schema changes and UI updates were synchronized without overwriting collaborative work.

4. Functional User Stories

The application features are mapped to specific Agile user stories, organized by the functional modules (Screens) within the app.

Program Management User Stories

Epic 1: Member Management

Implementation: membersPage

ID	User Story	Acceptance Criteria
US-01	As an Admin, I want to register new members so that they can participate in the program.	• Form captures First Name, Last Name, Email, and Enrollment Date. • System prevents duplicate email entries. • Data is successfully written to the Members table.
US-02	As an Admin, I want to search and filter member records so that I can quickly update contact details.	• Search bar filters the gallery by name dynamically. • Selecting a member populates the edit form.

Epic 2: Team Coordination

Implementation: teamsPage

ID	User Story	Acceptance Criteria
US-03	As an Admin, I want to create new teams so that members can be grouped for projects.	• Ability to define a Team Name and assign a Mentor. • Team is created in the Teams table.
US-04	As an Admin, I want to assign members to teams so that workload is distributed evenly.	• Interface allows selection of a member and a target team. • Updates the Team_Members association table.

Epic 3: Project & Progress Tracking

Implementation: projectsPage & careerTrackPage

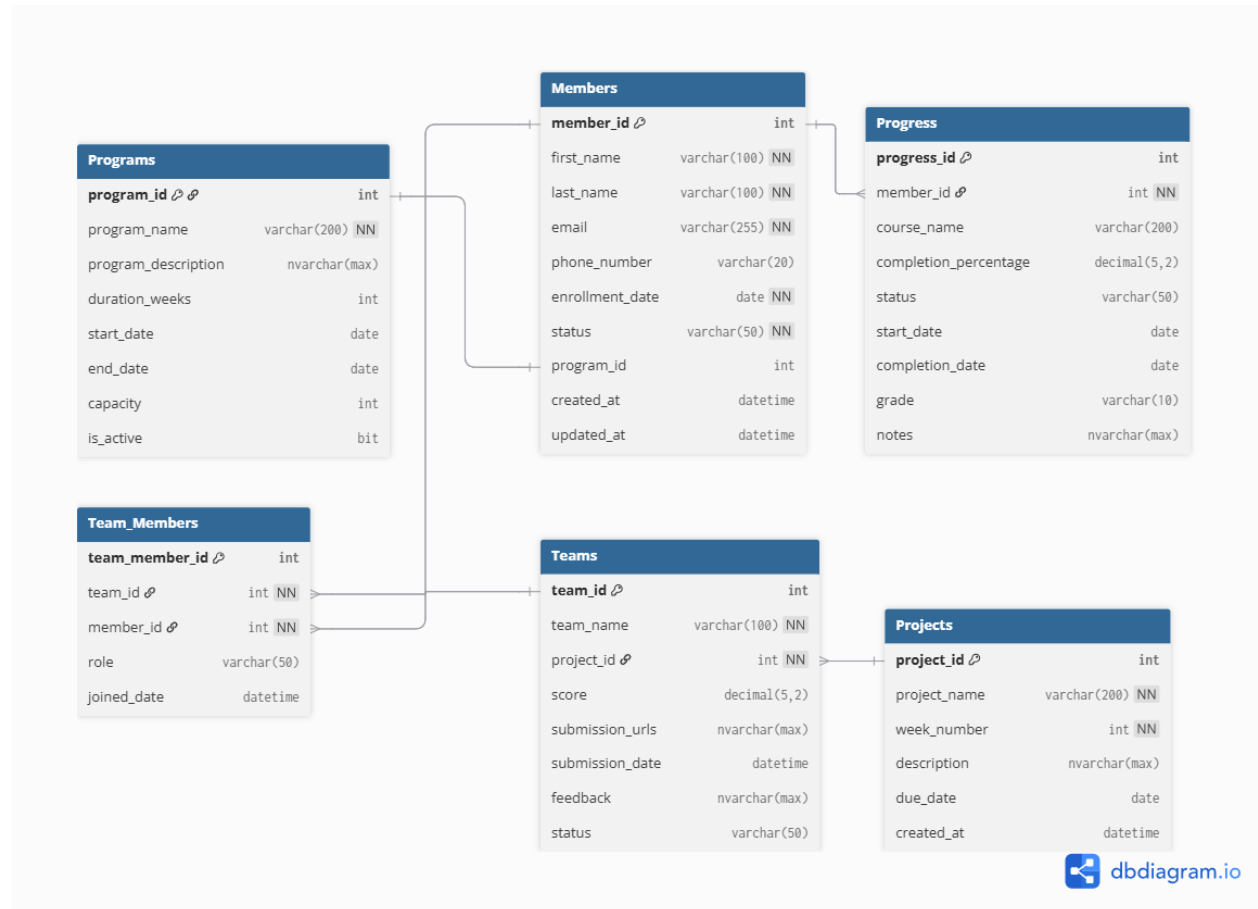| ID	User Story	Acceptance Criteria
US-05	As an Admin, I want to record project submissions so that team performance can be evaluated.	• Input fields for Project Name, Submission Date, and File Link. • Record is linked to the specific Team ID.
US-06	As an Admin, I want to input scores and feedback so that members know how they are performing.	• Admin can update the Score and Feedback columns for a specific project submission.
US-07	As an Admin, I want to view a member's career track progress so that I can identify at-risk participants.	• Visual indicator (bar/gauge) showing % complete based on completed modules.

5. Technical Architecture

The Stack

1. **Frontend:** Microsoft Power Apps (Canvas App).

2. **Backend:** Azure SQL Database (PaaS).

3. **Analytics:** Microsoft Power BI

Database Schema

The solution utilizes a relational database model hosted on Azure. The core entities include Members, Teams, Projects, and Programs.

6. Setup & Installation

Option A: End User Access

For general administrators utilizing the tool:

1. Navigate to the **Power Apps Portal** or use the link provided by IT.

2. Login with your Project Y organizational credentials.

3. Launch the **Member Data Manager** app.

Option B: Developer Deployment (from GitHub)

For developers looking to extend or debug the application:

1. **Prerequisites:**

   o   Access to a Power Platform Environment.

   o   Access to the Project Y Azure SQL Instance.

   o   **CRITICAL:** Your IP address must be whitelisted on the Azure SQL Server firewall to establish a connection.

2. **Importing the App:**

   o   Download the Project Y - Member Data Membership App.msapp file from this repository.

   o   Go to make.powerapps.com.

   o   Click **Apps** > **Import Canvas App**.

   o   Upload the .msapp file.

3. **Configuration:**

   o   Upon first load, the app will prompt for Data Connections.

   o   Select the **Azure SQL** connector.

   o   Enter the Server Name and Database Name provided in the config.json (or secure team notes).

   o   Save and Publish the app.

7. Future Roadmap

- **Mobile Offline Mode:** Enabling field data entry without an active internet connection.

- **Automated Alerts:** Power Automate flows to send email reminders for project deadlines.

- **LMS Integration:** API connectors to pull grade data directly from the Learning Management System.