P03-T2 Mgmt Sys NP

# Software Design Document (SDD)

CS 4850
Fall 2025
Professor Perry
9/14/2025

## Contents

# 1. Introduction and Overview

The Georgia Laws of Life Essay is a nonprofit organization facing a major problem. The problem is that they have many spreadsheets for keeping track of records and data, but these spreadsheets are majorly disorganized. An update to one spreadsheet requires that they go in and fix the other spreadsheets they have just to keep all the spreadsheets updated. They are looking for a technological solution that is cost effective while being user friendly, to help build a bridge between all the spreadsheets. Our team will look at the current Software as a Service Customer relationship management (CRM) and project management too
to help build a bridge between all the spreadsheets. Our team will look at the current Software as a Service Customer relationship management (CRM) and project management tools and provide them a solution that not only fits their needs right now as an organization but will also allow them to expand and grow the software as they continue to grow in size and personnel.

# 2. Design Considerations

## *2.1. Assumptions and Dependencies*

The following assumptions can be made about the software:

- The software will have integration with Google workspace

The industry partner uses Google workspace to do all their current work. The software that is chosen as the best option for their project will be under the assumption that Google workspace has been integrated into the software and there will be a way to grab data from Google Spreadsheets and be able to transfer this data into the software for them to see. Any method of data transferring will be allowed, but the preferred method will be the Google Docs API to transfer data to the CRM or Project Management Software.

- The user will access the software using either Windows 10 or 11 operating systems

The industry partners do not use Apple Mac and have up to date Windows upgrades. All the software that is looked at for this solution will be able to run on Windows 10 or Windows 11. This will ensure quality control as if the software works on one Windows system during testing, then replicating it to work on other Windows systems will be the goal to determine if the software is functional.

- The system will be regularly updated and logged into by the End User

The chosen software product will require that updates and changes be made by the End User. To deem the software as the optimal solution, the End User will be made aware that the success of the software is determined by the ability to be active and response to it. For example, for every change in the database that the customer will want to make, they will need to be active and update the database to reflect the change they want to make. Not doing so will harm their experience as the software will no longer have the most up to date database that the industry partner is looking for

## 2.2. General Constraints

- Hardware or software environment

  One of the constraints that we faced on the project is the software environment. Since the industry partner has specificized what software they expect to use, we cannot create a custom product that fully meets the needs of the partner. We instead need to rely on the listed software as a service product that they are expecting to see. These services are limited to what they provide and may not provide everything that the partner needs. The final product may be split across multiple services, to fulfill the needs of the industry partners. We also may be limited in the hardware as some of these services may only be available on Windows operating systems and will have trouble running on Linux or Mac operating systems.

- End-user environment

The end-user environment needs to be designed in a way that is user friendly so that anyone is able to use the product. It needs to have a user interface that can be easily learned and trained to be used by anyone.

- Security requirements (or other such regulations)

Certain security standards like SOC 2 and ISO 27001 need to be considered when looking at Software as a Service. Since we cannot implement security measures on our end, we need to rely on the Software we are using to have this security. This is important for the nonprofit as they have sensitive data that will be an issue if it gets leaked from a breach. By focusing on finding software that is designed with security in mind, we can ensure that we have software that practices the best methods to ensure that the data stored by the industry partner is secure.

- Performance requirements

The software needs to be able to perform for the maximum number of users who use the software at the same time. The maximum number of users according to industry partners is currently 3. The software needs to have performance to be able to run at the busiest time, when all 3 users are on the software at the same time trying to do all the work on the software. There should not be any major performance delay when users log on to the software when others are working on the software. New screens should be loaded promptly and when they do not, they should display a message saying that it is taking longer than usual.

When it comes to updating the database that is planned, small changes to the database should happen quickly and updates should show in a timely manner. Larger changes should take longer but should be scalable to smaller changes. A larger change can take longer to update, but it should be done in a reasonable timeframe that is linear in time to making a small change. The software should be able to identify when a small change is happening and when a large change is happening and be able to allocate any resources necessary to make the change in an appropriate timeframe.

- Network communications

Using software as a service means that we need a consistent and reliable network connection. Whereas some software that is custom made could be used offline, these services are often connected to the internet and use some sort of cloud computing, which is why internet connection is important. Without a constant internet connection, the service may not be available or lack performance due to a poor internet connection.

## *2.3. Development Methods*

The Development Methods for the project will be an Agile approach with UML-based modeling. When looking at and determining the best approach for development, this is the best method for the project given that we are looking at multiple CRM or Project Management tools and need a methodology that would allow us to shift from one SaaS to another if something was not working. Agile approach of doing development in increments and then looking at feedback will allow us to work on a SaaS for a while and if it is not working for the industry partners' standards, quickly shift to another approach. UML modeling is important because we need to clearly define who is doing what to the system and how they should interact with the system. By creating a proper UML model, it will help visualize the roles that each End User will have and how they interact with the system.

Other methods that were not considered for this project include the Waterfall and Spiral Model. The Waterfall method was not used as it was outdated and does not give the flexibility needed to do this project. For example, the waterfall does not allow for an easy way of going back once you have completed a phase, which means that we could have run into the issue of choosing a software that was not going to work and have no easy way of returning back to the previous phase.

The Spiral Model had the opposite issue when looking at it. Since the spiral model places an emphasis on risk analysis, we felt that we would spend too much time researching the best Software as a Service to use and not have enough time to implement it. This prevented us from choosing this model as the best option and led us to choose the Agile approach as the ideal approach to doing this project.

## 3. Architectural Strategies

The primary architectural strategy is to leverage configured SaaS over a custom build.

- **Decision:** To utilize an existing, cloud-based SaaS platform for CRM, donor, and project management functionality.

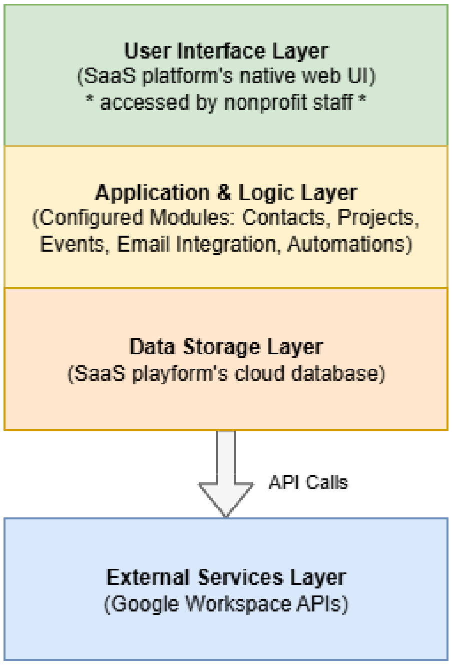
- **Reasoning:** This strategy directly satisfies the core constraints of $0 cost and no custom development. It provides an immediately available, supported, and scalable foundation that can be tailored to the sponsor's needs through configuration.

- **Alternatives Considered:**
  - **Custom-Built Database Application:** Rejected due to prohibitive development cost, long-term maintenance burden, and explicit project scope exclusion.
  - **Enhanced Google Sheets Setup:** Rejected as it does not fundamentally solve the problem of disconnected data and lacks native support for complex entity relationships, robust access controls, and automated workflows.

- **Key Mechanisms:** The architecture will rely on the native data models, relationship modules, automation rules, and API endpoints of the chosen SaaS platform.

# 4. System Architecture

The system architecture is conceptualized as a layered SaaS model, centered around the configured platform:


User Interface Layer
(SaaS platform's native web UI)
* accessed by nonprofit staff *
Application & Logic Layer
(Configured Modules: Contacts, Projects, Events, Email Integration, Automations)
Data Storage Layer
(SaaS playform's cloud database)
API Calls
External Services Layer
(Google Workspace APIs)

The overall project is decomposed into three logical subsystems:

1. Platform Evaluation & Selection: The process for choosing the right SaaS tool
2. Data Model & Configuration: The design and setup of data structures within the chosen platform
3. Integration & Automation: The design of data flow between the platform and Google Workspace

# 5. Detailed System Design

## *5.1. Platform Evaluation & Selection Subsystem*

- **Classification:** Process / Methodology
- **Definition:** A structured framework for comparing and scoring shortlisted SaaS platforms against a weighted set of criteria derived from sponsor requirements.
- **Constraints:** Evaluation is limited to features available in freemium tiers and publicly documented API capabilities.
- **Composition:** Criteria include Cost (30% weight), Google Workspace Integration (25%), Entity Relationship Flexibility (20%), Ease of Use (15%), and Feature Set (10%).
- **Processing:**
  - Research and shortlist platforms (Zoho CRM, HubSpot CRM, Airtable).

    - Score each platform (0-5) on each criterion.
    - Calculate total weighted score for each platform.
    - Select the platform with the highest score for POC development.
- **Interface/Exports: Platform Evaluation Matrix** (Deliverable: A spreadsheet documenting scores and rationale).

### *5.2. Data Model & Configuration Subsystem*

- **Classification:** Configured SaaS Application Modules
- **Definition:** The designed and implemented data structure within the selected SaaS platform that represents the sponsor's entities and processes.
- **Composition:** Configured modules will include:
    - **Contacts:** To manage School Contacts, Rotary Members, Donors.
    - **Companies:** To represent Schools and Rotary Clubs.
    - **Deals/Pipelines:** To represent the Essay Contest lifecycle (e.g., Submitted, Judged, Won).
    - **Projects/Events:** To manage contest cycles and special events.
- **Resources:** Freemium account on the selected SaaS platform. Simulated data sets.
- **Uses/Interactions:** Users will interact with these modules via the platform's UI to view, create, edit, and relate records.
- **Interface/Exports: A configured, live POC environment. User Guide** documenting the data model and navigation.

### *5.3. Integration & Automation Subsystem*

- **Classification:** API-based Design Specification
- **Definition:** The blueprint for connecting the SaaS platform to Google Workspace to facilitate data import/export.
- **Constraints:** Limited by the API call limits and features of the freemium tier.
- **Processing:**
    - Design a standardized Google Sheet template for bulk data entry.
    - Document the manual process for exporting data from the SaaS platform to Sheets for reporting.
    - If supported, design a simple automation (e.g., using native platform automation or Zapier) to add a new Contact from a Google Form submission.
- **Interface/Exports: Designed Google Sheet Templates. Integration Guide** documenting the step-by-step processes for data transfer.

## 6. Glossary

**CRM (Customer Relationship Management)** - system for managing a company's interactions with current and potential customers

**SaaS (Software as a Service)** - software licensing and delivery model where software is accessed online via a subscription

**Freemium** - business model where basic features are provided for free, while advanced features must be paid for

**POC (Proof-of-Concept)** - prototype designed to demonstrate the feasibility and potential of a proposed solution

**API (Application Programming Interface)** - set of rules that allows different software applications to communicate with each other

**Google Workspace** - suite of cloud-based productivity and collaboration tools developed by Google

# 7. Bibliography

Georgia Laws of Life Essay Contest. (n.d.). Retrieved from https://georgialawsoflife.org/

HubSpot CRM. (n.d.). Retrieved from https://www.hubspot.com/products/crm

Zoho CRM. (n.d.). Retrieved from https://www.zoho.com/crm/

Airtable. (n.d.). Retrieved from https://www.airtable.com/

Google Workspace API Documentation. (n.d.). Retrieved from https://developers.google.com/workspace