Li Help

CSC 131 - 04 Computer Software Engineering

Software Requirement Specification (SRS)

12 October, Thursday

Jonathon Delemos

Andrew Lawson

Majd Hameed

Kaden Bettencourt

Paul Arnett

Table of Contents

1. Introduction
    1.1. Project Overview
    1.2. Project Scope
    1.3. Target Audience
2. Use Cases
    2.1. Student Signup
    2.2. Minor Signup
    2.3. Admin
3. Functional Requirements
    3.1. FR 1.0
    3.2. FR 2.0
    3.3. FR 3.0
    3.4. FR 4.0
    3.5. FR 4.1
    3.6. FR 5.0
    3.7. FR 6.0
    3.8. FR 7.0
4. Sequence Diagrams
    4.1. Logging In
    4.2. Admin Creation
    4.3. Database Schema
    4.4. UML Class Diagram
    4.5. Technologies and Tools
    4.6. Development Enviroment
    4.7. Design Prototype
5. Risks and Mitigations
6. Conclusion

1.0 Introduction:

What impact could we create in our communities and the world if every able individual dedicated just one hour a week to volunteer work? LiHelp believes volunteering can give members of the community a sense of purpose and help foster skill development among community members. That's why LiHelp is working to create a website that allows users to quickly apply for volunteer projects at Sac State. There are many events that occur throughout the year, and most of these events need volunteers. Our main objective at LiHelp is to help the community by creating a portal through which dedicated individuals can submit applications for volunteer positions and better serve their community. Specifically, LiHelp will focus on automating the application process. Throughout this document we will highlight the many features our team is implementing.

1.1 Project Overview:

Our development team is working to create a portal through which students and community members can apply for volunteer positions at Sac State events. The current schema of the website is to have a sign in page that allows users to sign into their Google account using a Google API, then quickly fill out an additional form to gather more information that will be necessary to automate the application process. Once this information has been gathered, the user can then rapidly apply for any number of volunteer positions currently available.

1.2 Project Scope:

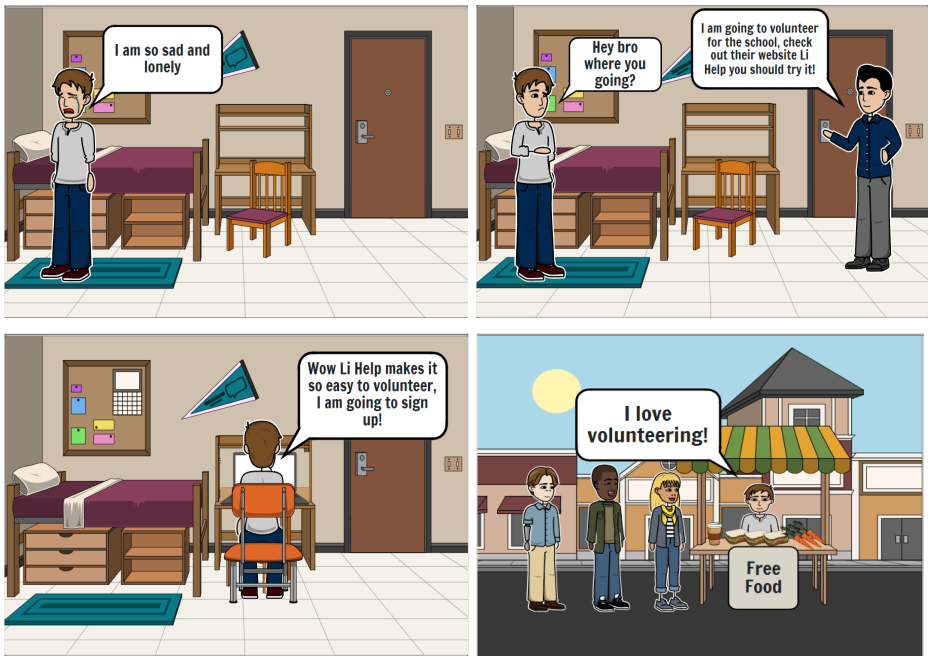
The scope of the project includes all project elements, milestones, assets, activities, and limitations our team faces. Simplistically, our goal is to create a website linked with a database. The website can store information in the database and retrieve it for auto-completion form purposes. As a team, we are working tirelessly to reach a state of full functionality. Every feature listed within our functional requirements/non-functional requirements is within our abilities and we plan to deliver everything ahead of schedule. Our team has yet to encounter something that has proved challenging.

1.3 Target Audience:

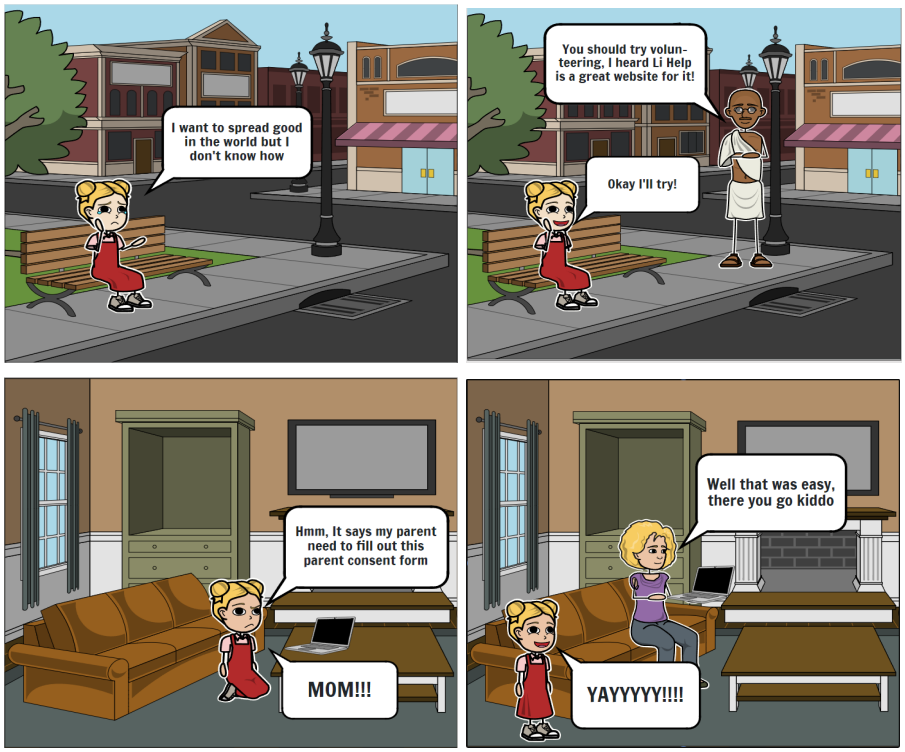
The target audience our team would like to reach is the greater downtown community. Ideally, any individual who would like to find a sense of purpose volunteering would be a candidate for our program. We would like to target all individuals who would like to improve their lives through volunteering.

2.0 Use Cases:

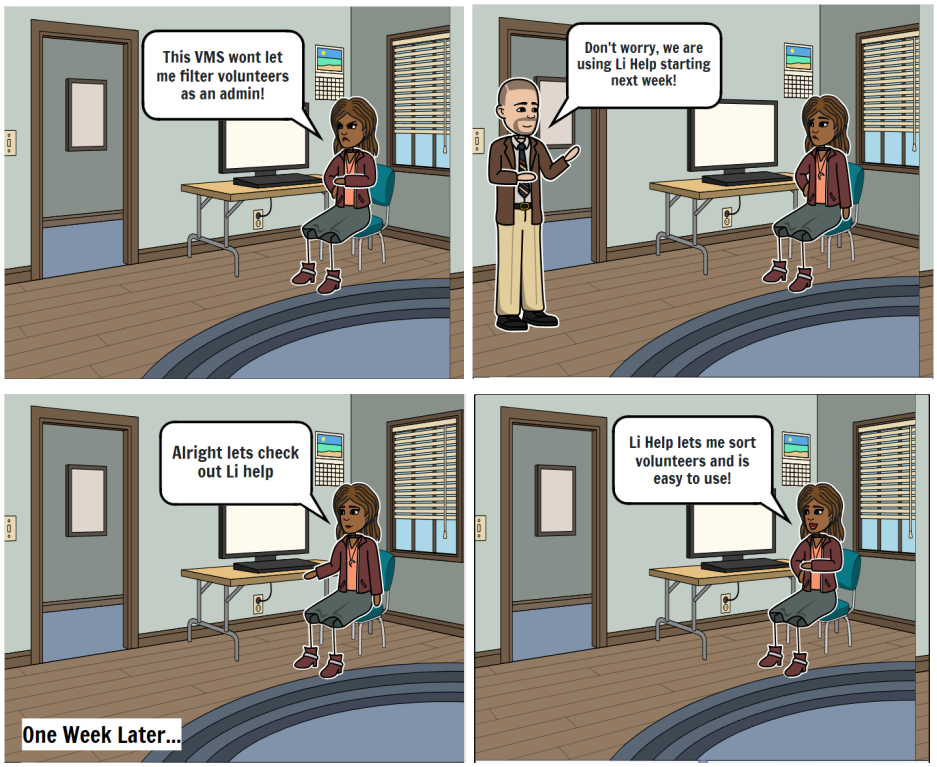
2.1 Student Signup



2.2 Minor Signup



2.2 Admin



3.0 Functional Requirements:

FR 1.0: User authentication.

- ➔ Purpose: To allow users to access their Li Help account
- ➔ Inputs: User clicks on “Continue with Google” button, and selects their google account in the popup
- ➔ Processing: The user inputs their google information. Google checks the account credentials and verifies it. Once it is verified, Google sends an authentication token back to the system
- ➔ Outputs: If the user is successful, they are redirected to the part of Li Help that they have been granted access to.

FR 2.0: Users are able to submit and update personal information.

- ➔ Purpose: To allow users to access Li Help.
- ➔ Inputs: Email, first name, middle name, last name, phone number, emergency contact (first name, last name, phone number), event position, birthday.
- ➔ Processing: The user will be presented with a form to complete, each field is required. The process will not be complete until each field passes a test that the requirement has been completed.
- ➔ Outputs: If successful they will be redirected to the featured events page. Otherwise, an exception as to why the account wasn’t created will be shown so the user can make corrections and complete creating their account.

FR 3.0: The featured event page will update daily with upcoming events and their details. (Risk)

- ➔ Purpose: Update the website with relevant events.
- ➔ Inputs: The website with the available volunteer opportunities.
- ➔ Processing: The website with the volunteer opportunities will be scanned for available events and their details. Information taken includes the name of the event, location, and date.
- ➔ Output: Displays each event with its associated details organized chronologically for users to browse.

FR 4.0: This requirement focuses on allowing coordinators/the career center to view applicants and accept volunteers.

- ➔ Purpose: Allow coordinators to accept volunteers and fill available slots.

- ➔ Inputs: Pressing the button that either accepts the candidate or declines the candidate.
- ➔ Processing: The system finds the record associated with the user account and grants them the status chosen by the admin in the status field.
- ➔ Output: The candidate accept or decline buttons disappear, and a message will be displayed in the table where the volunteer information was shown. The message will be "<user's full name> has been accepted/rejected."

FR 4.1: When coordinators/career center accept or decline an application an email is sent to the user. (Risk)

- ➔ Purpose: Sending the email to the user after the coordinator has made their decision.
- ➔ Inputs: The volunteer's status for an event.
- ➔ Processing: Taking the status and putting that into an automated message that is then emailed to the user's email.
- ➔ Output: The user will receive an email containing the status of whether they'll be volunteering at an event or not. If they are accepted the email will contain the time when they're expected to volunteer and their role.

FR 5.0: Users will be able to apply for events with the press of a button.

- ➔ Purpose: To volunteer at Sac State events.
- ➔ Inputs: Submitting the volunteer form.
- ➔ Processing: Retrieving the form from the database, and storing the form in the event's collection.
- ➔ Output: The button that allows the user to volunteer for an event will become grayed out showing that the user has successfully submitted their form.

FR 6.0: When applying, if the user is underage a pop-up form will appear. The form is a parent permission form that the parent/guardian must fill out.

- ➔ Purpose: To prevent users from applying when underage.
- ➔ Inputs: Parent consent form.
- ➔ Processing: Store parent consent form in database.
- ➔ Output: Include parent consent form in volunteer application form.

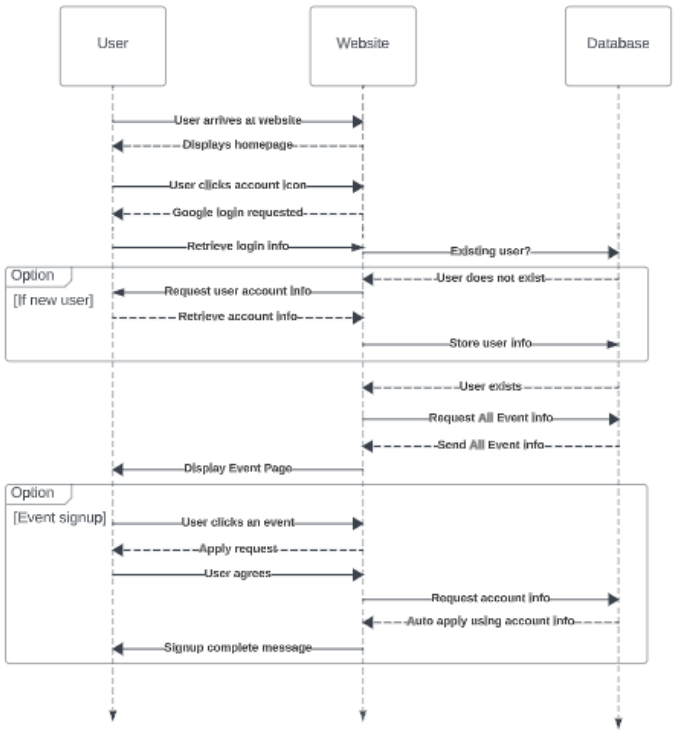
FR 7.0: Admin will be able to create new events (updated)

- ➔ Purpose: To update the website with current opportunities
- ➔ Inputs: Event details: Name, date, place, time, available positions, type of positions

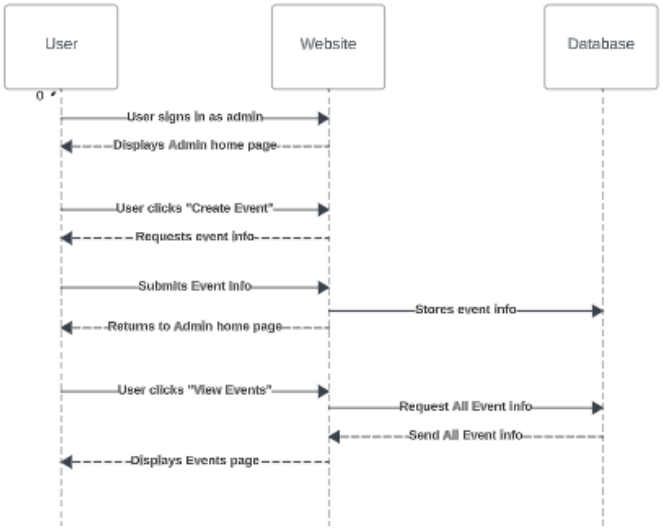
- ➔ Processing: After giving the details for the event the event will be stored in the event collection DB. This database will be read to display the featured event page users will see.
- ➔ Outputs: The website's featured event page will display the events created by the admin and organize them my most recent (top to bottom). A confirmation message will be shown to the admin stating the event was created successfully.

4.0 Sequence Diagrams:

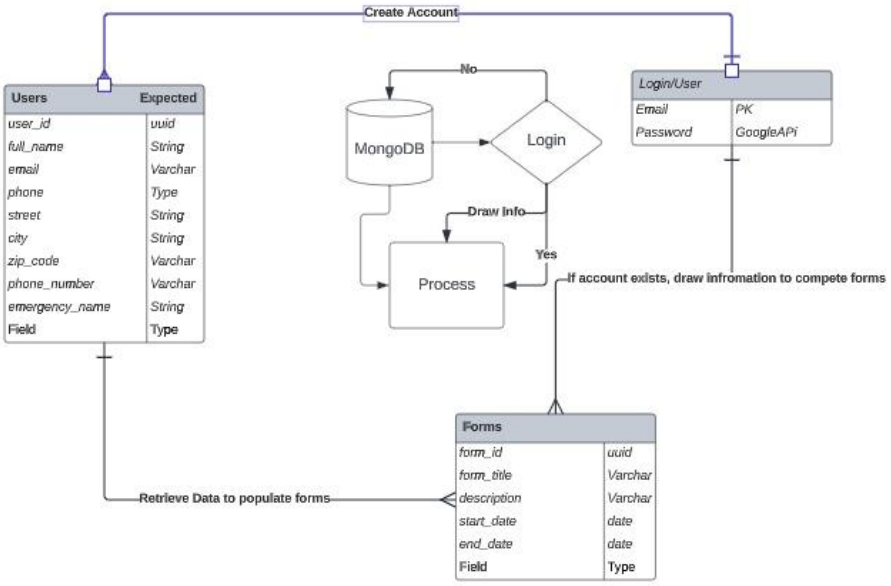
4.1 Logging in and Signing up for an Event:



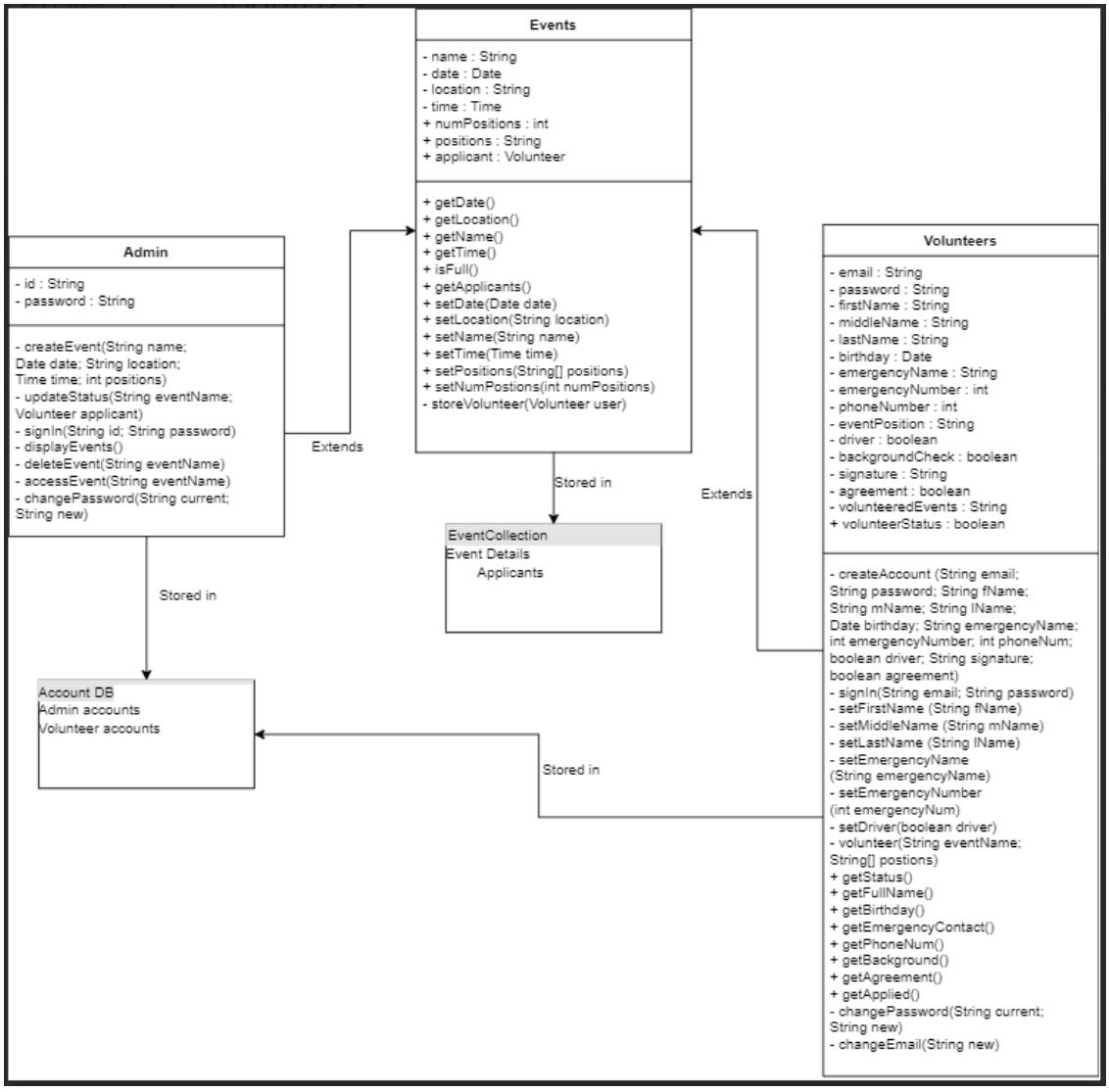
4.2 Admin creating and viewing events:



4.3 Database Schema:



4.4 UML Class Diagram:



4.5 Technologies and Tools:

- HTML, CSS and JavaScript will be used in tandem to develop the frontend.
- MongoDB will be our database technology and will be used to store user information after they fill out the form
- Express.js will be used to connect MongoDB with our JavaScript code
- Google API will be used for logging in so that all of the users info will be connected to their gmail.

4.6 Development Environment:

All of our development will be done using Visual Studio Code with various extensions, including Live Share, which allows us to collaborate on the same files within VS Code.

4.7 Design Prototype:

1. Figma design prototype:

   Link to figma prototype

2. Design Patterns:

5.0 Risks and Mitigations:

Some possible risks the team has been debating and discussing whether the goal is achievable or cohesive include updating the featured event page by scratching an existing source that displays the available events and its details. Another risk is how the software will communicate with the users once the coordinator has responded to their application to volunteer.

Addressing the risk related to updating the featured events page by scratching the details from another source there are two main risks. One, actually finding a source for these events that's dependable and updated frequently enough. Two, actually programming the process of scratching a page and extracting only the information desired is way too complicated and hard to achieve with all other requirements that also need to be completed. In order to mitigate the risk, the team has discussed a new way to update the featured event page that's much more cohesive with the software and keeping everything local to Li Help. The idea is to have coordinators, using an admin login, be able to create their own events. This will allow coordinators to be more active and involved in the process and will continue to post new events to the featured event page for volunteers to apply for.

Addressing the risk connected to communicating the change of status with the volunteers the team also agreed that it would be rather incohesive still. Although users provide their email address, it's also true that everyone checks their emails differently and having to find out whether a user is volunteering outside the website feels disconnected. In order to mitigate this risk the team has discussed implementing a page that includes the vents applied for by each user. This would allow users to log in and view the events they've applied for with their respective status listed as pending, accepted, or declined. Overall, this again, would allow all information and interaction to be done through the Li Help website without needing to access other sources making the software more cohesive.

6.0 Conclusion:

LiHelp wants to make volunteering easy and improve our community. Throughout this document we addressed the functional and non-functional requirements and how they operate. We showcased our final Figma product and how it correlates to the use cases. Finally, we believe with the tools our team has created; everyone will be able to quickly apply for volunteering events at Sac State. We want to make it easy for you to be the change you want to see in the world. Volunteer with us using LiHelp to create a better brighter world and join the other dedicated volunteers to start making an impact today.

Team members Contribution:

Jonathon Delemos	20%
Andrew Lawson	20%
Majd Hameed	20%
Kaden Bettencourt	20%
Paul Arnett	20%