Wentworth Institute of Technology
COMP4960 – Software Engineering

Software Requirements Specification for SyncedUp

Version 1.1
Finalized on 04-07-2024

Prepared By
Team: JTNF, Project Group 2
Jillian Desmond, desmondj2@wit.edu
Thomas Michael, michaelt@wit.edu
Nilay Patel, pateln19@wit.edu
Fabio Tran, tranf@wit.edu
Instructor: Dr. Gyllinsky

GitHub Link
https://github.com/teachingworkshops/comp4960-spring-2024-project-group-2-syncup.git

Submitted on 04-07-2024

Contents

Revision History 3
1. Introduction 4
1.1. Problem statements 4
1.2. Proposed solution 4
1.3. Novelty 4
1.4. Software In Use Examples 4
2. Requirements 5
2.1. Functional requirements 5
2.2. Non-functional requirements 5
2.3. Other requirements 6
2.4. Use Case Tabular Diagrams 6
3. Appendix I 9
3.1. Comparison to Existing Products 9
3.1.1. Apple Calendar 9
3.1.2. Google Calendar 10
4. Appendix II 11
4.1. Definitions 11
4.2. References 11

Revision History

Date	Version	Description	Author
02-29-2024	1.0	Initial draft	Jillian Desmond, Thomas Michael, Nilay Patel, Fabio Tran
03-14-2024	1.1	Minor revision to requirements	Fabio Tran
04-07-2024	1.2	Fixing minor formatting issues and adding onto definitions sections	Fabio Tran

1. Introduction

1.1. Problem statements

- Absence of Centralized Planning: Due to busy and dynamic schedules, friends and family members have trouble planning social gatherings with one another and often struggle to accommodate everyone's availability with a lack of a centralized source of information.
- Inefficient Communication: Without centralized planning, reaching out to members individually leads to excess time and effort.
- Overlapping of Events: Struggles in coordination could lead to mistakes and the creation of events that overlap with one another or situations where invitees are unable to attend.

1.2. Proposed solution

- An application that people can upload data about their schedule and availability. Other people in one's social network can join and provide their data as well which will be analyzed by the application to suggest recommended dates and times to establish event plans.

1.3. Novelty

- The application will have a continuous calendar where users can enter and update their availability as needed.
- The application will allow people to be able to join social networks and see when others are available for potential plans.
- The application will make recommendations to users with regards to date and time for events.
- The application will create and save events into a digital calendar at the request of users.

1.4. Software In Use Examples

- User A will log in to the application and enter data into their digital calendar about their availability. For example, this user might fill in days they already have existing plans. User B and other users will also enter their data. User A and User B can add each other to their social network to be able to view each other's availability. The application will analyze User A and User B's data to suggest recommended meetup times. If accepted by all parties, this plan should be created and added to the virtual calendar.
- User A will log into the application and invite or accept friend requests from other Users. Users can create friend groups or social networks to organize their social relationships.

2. Requirements

2.1. Functional requirements

2.1.1 Upon starting, the system shall be able to generate a new user account.

2.1.2. Upon starting, the system shall be able to login to an existing user account.

2.1.3. Once a user is logged in, the system shall be able to delete that account.

2.1.4. The system shall be able to connect users to multiple groups.

2.1.5. The system shall have a tab that displays all the groups users are connected to.

2.1.6. Upon selection of a group, the system shall display the availability of each member.

2.1.7. Once logged in, the system shall allow users to set their availability. The system shall offer availability to be set as repeating (daily, weekly, monthly, yearly) or one-time.

2.1.8. Within a group, the system shall have a button for planning a time to meet.

2.1.9. Upon selecting the button 2.1.9, the system shall suggest times that every group member is available within a specified time frame. If there is no time window when everyone is available, then the system will suggest alternatives with as many members as possible.

2.1.10. When producing meting times 2.1.10, the system shall order these time windows earliest to latest.

2.1.11. The system shall allow users to send friend requests to other users.

2.1.12. Upon the recipient user accepting the friend request 2.1.12, the system will make a connection between both users.

2.1.13. When two users have a connection 2.1.13, the system shall give both users a button to view the other’s availability.

2.1.14. The system shall provide a button to view the availability of every member of the group.

2.1.15 The GUI shall display a friends list and a group list upon user request.

2.1.16 Within each group, the GUI shall display a list of all users in the group.

2.1.17 The GUI shall display an error message when a user attempts to send a friend request to a user that does not exist. It should also give the user the option to retry.

2.2. Non-functional requirements

2.2.1. The system shall store login data via database to be queried for comparison with user input in order to authenticate user.

2.2.2. The system shall remove user login data when a user deletes their account.

2.2.3. Upon account deletion, the system shall delete availability data and reflect that on the UI.

2.2.4. The system database will link user account data to groups as one-to-many mapping.

2.2.5. The system should provide real-time user calendar updates of the group the user is viewing or when viewing availability of another user.

2.2.6. The method for suggesting meeting times within a group will use a greedy activity selection algorithm.

2.2.7. The system shall be able to handle 1,000 requests/queries at once.

2.2.8. The system shall respond to any user input within 1 second.

2.2.9. API integration for backend and front-end interactions. Consecutive requests are needed to ensure any changes to data stored in the database are shown on UI.

2.2.10. The system shall be available to use and interact with 24/7, with maintenance exceptions.

2.3. Other requirements

2.3.1. The system will use Java as the primary programming language.

2.3.2. The system will use Solid as its data management platform.

2.3.3. The system will use Inrupt PodSpaces and Inrupt's Enterprise Solid Server as its third party provider for Solid Pods

2.3.4. UI should have intuitive navigation, where a user can utilize all functionalities without a tutorial and within a short time frame.

2.4. Use Case Tabular Diagrams

Number	1
System	SyncUp
Name	Set Availability
Primary Actors	User
Descriptions	A user can set availability in the system for any day of the year. The user can make that availability repeat daily, weekly, monthly, or yearly.
Preconditions	The user has created an account.
Post-conditions	One or more days have availability assigned to them.
Trigger	User initiates the set availability process
Basic Flow	1. User chooses to set availability 2. Week Calendar pops up 3. User selects a day 4. Input window pops up. 5. User inputs the timeframe they are available for that day. 6. User saves input
Alternate Flow	A1.1. User chooses to set availability A1.2.1. Week Calendar pops up A1.2.2. User changes the view to a monthly calendar A1.3. User selects a day A1.4. Input window pops up.

	A1.5. User inputs the timeframe they are available for that day. A1.6. User saves input
Exception Flow	E 1.1 (2, 3, 4, 5) User decides to cancel the process.
Extensions	5a. User opts for the time window to repeat (options are daily, weekly, monthly, yearly) 6a. User selects another day (repeats steps 3-6, alternate flows and exception flows still valid).

Number	2
System	SyncUp
Name	Send friend request
Primary Actors	User A, User B
Descriptions	User A sends a friend request to User B
Preconditions	The user has created an account.
Post-conditions	User A and User B are connected as “friends”.
Trigger	User initiates send friend request process
Basic Flow	1. User A searches User B by username. 2. User A sends a friend request 3. User B receives request and chooses to accept
Alternate Flow	
Exception Flow	E1.1. (1) There is no user associated with that username E1.2. No results populate from that search E2.1. (3) User B denies the friend request E2.2. Friend request disappears from User Bs display
Extensions	

Number	3
System	SyncUp
Name	Join Group
Primary Actors	User, Group
Descriptions	A user can join multiple groups.
Preconditions	1. The user has created an account. 2. The group exists.
Post-conditions	User is a member of the group.
Trigger	User wants to join a group.
Basic Flow	1. User requests to join a group. 2. A notification is sent to each group member. 3. Any member of the group approves the user request.
Alternate Flow	A1.1. A member of the group sends the user an invite. A1.2. A notification is sent to the user. A1.3. The user accepts the invite.
Exception Flow	E1.1 (3) A member of the group rejects the request to join. E2.1 (A1.3) The user declines the invite.
Extensions	

Number	4
System	SyncUp
Name	Create Event

Primary Actors	User
Descriptions	Create an event within a team availability
Preconditions	1. The user has joined a group 2. The members of the group have added their availability
Post-conditions	Each team member has a new event blocked off on their calendars
Trigger	User wants to plan a time to meet
Basic Flow	1. User selects the option to find a time to meet 2. System searches for date(s) and time(s) that all group members are available. 3. System returns meeting time options. 4. User selects one 5. Event is added to the team members' calendars as unavailable.
Alternate Flow	A1.3. System could not find a time that all team members are available A1.4. System asks the user if they would like to find a meeting time with not all team members A1.5. User selects option to continue without all team members A1.6. System returns a meeting time with as many team members as possible. A1.7. User approves meeting time, and the meeting is added to the selected team members' calendars
Exception Flow	E1.1 (A1.5.) User declines to continue.
Extensions	

3. Appendix I

3.1. Comparison to Existing Products

3.1.1. Apple Calendar

- Apple offers “Calendar” which is a calendar app built into the operating system of nearly all of their electronic products such as iPhone, iPad, and iMacs (Apple).
- This app offers a virtual calendar where users can create events on specific dates and times.
- Apple Calendar is infinite so users can scroll and view dates days or years ahead into the future or into the past.
- Apple Calendar supports embedded GPS data for event location purposes, URL support for related material, and notifications feature to remind users about the event.
- Apple has iCloud, a cloud storage solution, that when integrated with their native [1] Calendar app, can allow a user to check their virtual calendar on any of their devices connected to the same iCloud account.
- Users can also share their apple Calendar with friends and family through iCloud.
- While this app is integrated seamlessly with Apple devices, users on Windows or Androids will have to use utilize workarounds to access Apple Calendar (Calendar).
- Shared calendars have permission levels of “View Only”, “Can Edit”, or “Can Edit & Share”.
- Users are not able to directly see friends and families’ calendars unless a specific calendar has been shared between all parties.
- Apple Calendar does not have a suggestion feature to propose potential plans for all invitees to review and accept/reject.

- Apple Calendar does not have a feature that recommends a date and time for friends and family to create new events that do not have conflicting timings.
- Apple Calendar has a public calendar feature but only the original creator of the calendar can make changes to its events and sharing permissions.
- Apple Calendar has many features geared towards personal event creation with added abilities to share for other people to view and edit. However, it lacks features for invitees to provide their input on their own availability.

3.1.2. Google Calendar

- Compared to other calendar applications, Google Calendar also offers a virtual calendar that gives users the ability to create events and add information such as date, time, and location (Google).
- Google Calendar is designed for work organizations rather than friends and family.
- Google Calendar users can subscribe to other user's calendars to view events on their calendars (Lau).
- Users can set days and times for when they are offline [2].
- Google Calendar has a search feature for users to search key words to find created events on their calendar.
- Google Calendar has a focus time feature that users can use to notify external users that a person cannot be bothered at a certain day and time.
- Google Calendar offers some analytics to notify users of what they are spending their time on which is useful in business to identify how time is spent.
- Google Calendar has support to embed its calendars onto website.
- Like Apple Calendar, there is no permission level to provide suggestions only but there are options to allow people to read or edit only.
- While Apple Calendar is designed for family and friends, Google Calendar is designed to be used in a work or business setting.

4. Appendix II

4.1. Definitions

- [1] Native: A native app is software designed for a specific operating system or device.
- [2] Offline: In the context of a work environment, offline means a person or system is not available to be contacted and accessed.
- [3] Solid: is a specification that lets individuals and groups store their data securely in decentralized data stores called Pods.
- [4] Pods: are like secure web servers for data where owners' control which people and applications can access it.

4.2. References

- Apple Inc. (n.d.). *ICloud User Guide*. Apple. Retrieved February 25, 2024, from https://support.apple.com/guide/icloud/what-you-can-do-with-icloud-and-calendar-mm15eb200ab4/icloud
- Calendar (2023, May 2). *Apple Calendar Guide: Everything You Need to Know About iCal*. Retrieved February 25, 2024, from https://www.calendar.com/apple-calendar/
- Google LLC (n.d.). *Google Calendar Help*. Google. Retrieved February 25, 2024, from https://support.google.com/calendar/?hl=en#topic=10509740
- Lau, J. (2023, November 27). *9 Google Calendar features you should start using now*. Zapier. Retrieved February 25, 2024, from https://zapier.com/blog/google-calendar-schedule/