Tempo

Alex Boyd
Brandon Kelley

Jessie Smith
Andrew Cofano

Yeng Tan
Costin Pirvu

Aim of the document

This document will describe the architectural basis on which our product was founded on and the functionalities provided to effectively resolve the problems we seek to solve.

Overview of the defined system.

The assiduous task of arranging meeting times has always been an issue, be it for a local meeting, a domestic arrangement, or an international conference. Tempo was founded under that very premise: of encompassing calendars across a myriad of scheduling platforms to automatically generate a schedule that works specifically for you and your cohort.

Stakeholders

- Students
- Startups/small businesses
- Dr. Falessi
- Everyone in our group
- Our 308 class
- Software developers
- Google drive calendar users
- Work groups

Operational settings (where is this supposed to work).

- Android dynamic mobile application
- Will be used in a group setting
- Must have internet connection to utilize application

Related systems

Doodle

- Doodle takes more time because the user has to put in when they are available manually, whereas Tempo analyzes the user's schedules and automatically generates results.
- This system (Doodle's) is less convenient, as users will have to cross reference other resources to receive results.
- Something that is good about doodle is that the user can put in when they are available more situationally, meaning they don't have to base it off of what is only on their calendars. Some people do not have every scheduled minute entered into their calendar so Tempo might give non-optimal times.
- Our product will exist on a mobile platform as opposed to the traditional desktop experience that Doodle solely exists on. This platform has the potential to grow integrate into a desktop experience (see Slack and GroupMe).

Outlook

- Outlook is primarily a desktop experience and is not focused on provided a reliable mobile interface. Tempo resolves this issue by primarily being a mobile application. This grants Tempo multi-platform accessibility as well.
- You can only merge and share calendars easily if everyone is using the same calendar product (Outlook). While Tempo will initially only integrate with Google Calendars, it has the potential to support future 3rd Party calendar models (such as Outlook).
- Tempo will be available for non-enterprise use and in more casual settings whereas Outlook is traditionally seen as a business oriented product.

User Stories

1. As a student, I want to sync my class schedule with my work schedule, so that I can be more effective with how I plan my time.
2. As a business executive, I want to arrange meetings with my clients, so that I can be more productive during my working hours.
3. As a teacher, I want to arrange my office hours with students' schedules, so that I can save time.
4. As a student, I want to connect my calendar with my professor's calendar and schedule a meeting time, so that I can get help with my homework.
5. As a professor, I want to see the availability of all of my students, so that I can save time scheduling.
6. As a student, I want to sync my schedule in under a minute with people I just met, so that we can plan a meeting time without manually checking our schedules.
7. As a student, I want to be reminded of an event, so I am not late.
8. As a team leader, I want to monitor my team's individual work schedules, so that I can check that they are productive.
9. As a secretary, I want to edit calendars with information, so that I may keep people's schedules consistent.
10. As a user with a Google account, I want to access my saved calendars, so that I can organize my events in one place.
11. As a team member, I want to send out a reminder to all of my teammates, so that they are aware of an important event they should attend.
12. As a team member, I want to be alerted if I am creating an event that conflicts with another event on the team calendar, so that we do not have overlapping obligations.
13. As a team leader, I want to remove an event from the teammates' calendars, so that I can free up space for possible meeting times.
14. As a team leader, I want to be able to add events on my teammates' calendars, so that they can know when events will be held.
15. As a team leader, I want to add a new member to the group, so that we can include more group member's pre-existing calendar events.
16. As a user, I want to invite all of my Google contacts to a group, so that we can organize events together.
17. As a user, I want to add events to my Google Calendar through the system, so that I can avoid closing the application.
18. As a user, I want to vote on a common meeting time with my group members, so that all my group members' time constraints are taken to into account.

Functional Requirements

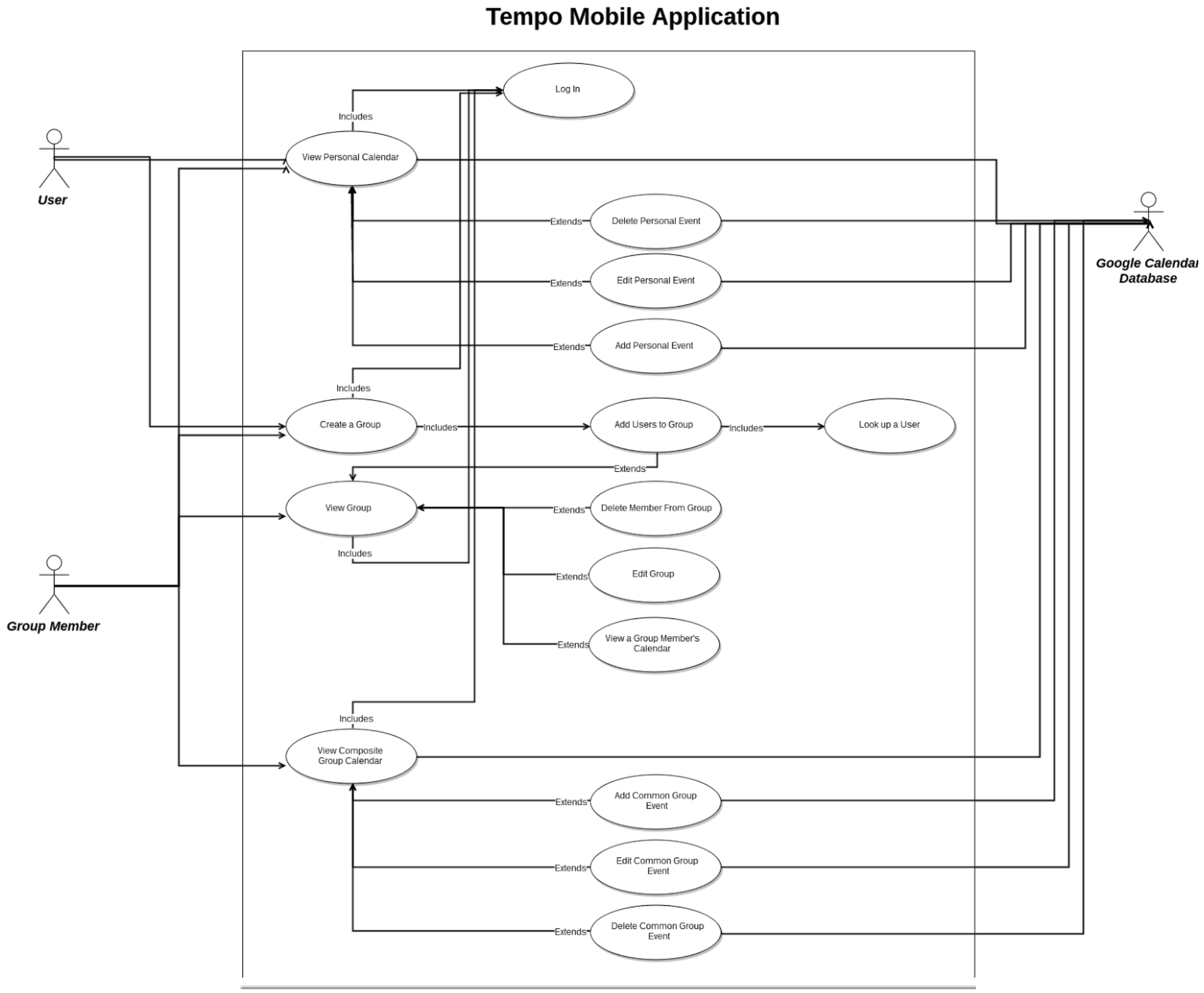
1. The system shall display a user’s calendar information when a user signs in.
2. The system shall allow a user to remain signed into an external email service after closing the application.
3. The system shall display a list of related users when a user search is performed.
4. The system shall display a user’s information when that user is selected from a search.
5. The system shall create a group on user input.
6. The system shall remove a group on user input.
7. The system shall allow a maximum of seven users per group.
8. The system shall allow a group member to add other users to the group.
9. The system shall allow a group member to remove a member from the group.
10. The system shall allow a group member to leave a group.
11. The system shall allow a group member to view the information of other members in the group on user input.
12. The system shall allow a group member to create an event on each group member’s calendar.
13. The system shall provide upcoming event notifications within 5 seconds of launching.
14. The system shall send notifications of upcoming events to the user’s provided email within a 1 minute window of the user specified notification time.
15. The system shall display the events of each user’s calendars as one master calendar.
16. The system shall display meeting times that are available for all group members.
17. The system shall create a common event within 5 minutes of group approval.
18. The system shall update the master calendar within 5 minutes of a group member’s calendar change.
19. The system shall display a user’s personal calendar to another user upon approval.
20. The system shall allow a user to update their personal calendar events.
21. The system shall remove personal calendar events on user input.
22. The system shall add personal calendar events on user input.
23. The system shall send a notification to a user when another user requests to see their personal calendar.
24. The system shall allow a user to assign a category for their event.
25. The system shall allow a user to select an interval for a recurring event.
26. The system shall display the user’s calendar by week, day, or month as specified by the user.
27. The system shall display event details on user input.
28. The system shall allow a user to change the language on user input.
29. The system shall allow a group member to view specific events on another group member’s calendar.
30. The system shall allow a user to sign out of a personal calendar account.

Non-Functional Requirements

1. The system shall be compatible with all Android operating systems at or above API level 14.
2. The system shall depend on calendar service APIs to import calendars.
3. The system shall manage a user’s calendar through the use of that service’s API session support.
4. The system shall adapt to changes from any calendar format.
5. The system shall support the US date style and the European date style.
6. The system shall label different calendar events with the group members associated with it.
7. The system shall reserve a maximum of 50 MB of storage.
8. The system shall have a mean time between crashing of greater than 1 month.
9. The system shall support English and Spanish.
10. The system shall update a calendar event within 2 seconds of a user’s input.
11. The system shall reserve a maximum of 50 MB of RAM.
12. The system shall keep a user’s calendar private.
13. The system shall have a level of security approved with an Extended Validation Certificate on Google Chrome and Safari.
14. The system shall recover from invalid states related to user sessions.
15. The system shall respond to all user input within 2 seconds.
16. The system shall maintain user account privacy as per Google’s Privacy Policy.
17. The system shall go through maintenance every Monday at 4:00 a.m. PST.
18. The system shall allow free access to all users.

Use Cases

Overview diagram



Internal Step Description

<u>Creating a new event:</u>

1. The user requests to use the calendar on the main screen.
2. The system redirects the user to the calendar page and displays a calendar with all the user's events.
3. The user requests to add a new event on the calendar page.

4. The system creates a blank new event.
5. The user provides an event description and the date of the event.
6. The user completes the activity and requests to save the event changes.
7. The system records the new event and displays it on the calendar.

Extensions:

2a. *There are no current events*: The system will display an empty calendar page if there are no events.

5a. *The user wants to exit this use case*: The user indicates to cancel the form and the system terminates the use case.

6a. *The specified new event conflicts with a preexisting event:* The system will notify the user and terminate the use case.

7a. *Recording the new event returns a server error:* The system will notify the user to retry.

<u>Delete Member from Group:</u>

1. The user views one of his/her groups and selects a member to remove.
2. The system asks the user to verify removal of the member.
3. The user verifies removal of the member.
4. The system dissociates this member from the group.
5. The system notifies the user of removal.

Extensions:

1a. *The user is the only member of the group:* the system does not display the ability to remove the user.

2a. *The user does not verify removal of group member:* the system terminates the use case.

3a. *The user wants to exit this use case*: The user indicates to cancel the form and the system terminates the use case.

4a. *The system is unable to dissociate the member:* the system alerts the user of the database error and terminates the use case.

<u>Create a group:</u>

1. The user requests to create a new group.
2. The system provides a blank group to be created with user input.
3. The user provides the group name and adds members to the group.
4. The system stores the group information.
5. The system notifies the group members.

Extensions:

3a. *The user provides a group name that already exists*: The system prompts the user for a different group name.

3b. *The user wants to exit this use case*: The user indicates to cancel the form and the system terminates the use case.

4a. *The system is unavailable*: The system notifies the user and terminates the use case.

<u>View Composite Group Calendar</u>

1. The group member requests to view the composite group calendar.
2. The system provides a blank composite calendar format form.
3. The user specifies the time range and format of the composite calendar.
4. The system merges the individual calendars of all group users in the group.
5. The system provides a calendar view.

6. When the group member indicates they have completed viewing the calendar, the system closes the calendar view.

Extensions:

3a. *The user wants to exit this use case*: The user indicates to cancel the form and the system terminates the use case.

4a. *The Google Calendar Database does not respond:* The user is notified and the system terminates the use case.

Add Personal Event

1. The user views their personal calendar and requests to add an event.
2. The system displays a blank event form.
3. The user provides the start time, end time, and name of the event.
4. When the user indicates the event's information is correct, the system informs the Google Calendar Database of the changes to the user's calendar.
5. The system displays the user's calendar via the use case View Personal Calendar

Extensions:

3a. *The user wants to exit this use case*: The user indicates to cancel the form and the system terminates the use case.

3b. *The user enters a start time greater than the end time:* The system notifies the user and prompts the user to correct the mistake.

4a. *Google Calendar Database does not respond:* The system notifies the user and terminates the use case.

Bugs

Bug #1:

Old Functional Requirement:

The system shall select the time with the most votes from users and display this time to all of the users involved within one minute of the last user submitting their vote.

Bug:

Extend this to two FR's. One for the system selecting the time and one for displaying the notification.

Fixed Functional Requirement(s):

The system shall shall collect votes from users and tally up the event time with the most votes.
The system shall create and save an event for two people to their calendars within one minute of both agreeing to a meeting time.

Bug #2:

Old Functional Requirement:

The system shall reserve a maximum of 50 MB of memory.

Bug:

What kind of memory is "50 MB of memory" referring to? Internal? External? Etc?

Fixed Functional Requirement:
The system shall use a maximum of 50 MB of RAM

Bug #3:
Old Functional Requirement:
The system shall display a list of times that are free in all users' group member's calendars on user input.
Bug:
"a list of times that are free in all users' group calendars" can be misunderstood as non-mutual "free-times"
Fixed Functional Requirement:
The system shall display a list of times that are mutually free in all group member's calendars on user input.

Bug #4:
Old Functional Requirement:
Description: The system shall run on all android devices.
Bug:
- Cannot test "on all android devices"
- Suggestion: remove the word all and possibly add a version, i.e. Windows 7 and up.

Fixed Functional Requirement:
The system shall run on android devices with version Windows 7 and up.
Bug #5:
Old Functional Requirement:
The system shall give a user access to add calendar events on other group members' calendars.
Bug:
The system shall give a user access to add calendar events on other group members' calendars.
Fixed Functional Requirement:
The system shall send a notification to a user when another user requests to see their personal calendar.

Bug #6:
Old Functional Requirement:
The system shall merge a user's group's calendars together into the group's master calendar.
Bug:
"user's group's calendar" is wordy. It can be more concise without saying much.
Fixed Functional Requirement:
The system shall merge a group's calendars together into the group's master calendar.

Bug #7:

Old Functional Requirement:

The system shall allow a user to delete events from their personal calendar instead of a group calendar.

Bug:

You can shorten it to just say "The system shall let the user remove calendar events"

Fixed Functional Requirement:

This functional requirement was already used, we changed it to:
The system shall allow a maximum of seven users per group.

Bug #8:

Old Functional Requirement:

The system shall add and save an event to the user’s Google calendar when the user creates an event on the system.

Bug:

A revision to this requirement would be: The system shall add personal calendar events on user input.

Fixed Functional Requirement:

This requirement is already used, so the revision is: The system shall allow a group member to view specific events on another group member’s calendar.

Bug #9:

Old Functional Requirement:

As a secretary, I want to edit calendars with concurrent information, so that I may keep people’s schedules consistent.

Bug:

First, there is a misspelling of the word "consistent". But also, the use of the word "consistent" may change the meaning of the sentence.

Fixed Functional Requirement:

As a secretary, I want to edit calendars with concurrent information, so that I may keep people's calendars accurate.

Bug #10:

Old Functional Requirement:

The system shall prompt the user to input their Google account information when the program opens, and save their account information to their profile before the user is logged in.

Bug(s):

- Not specified on whether or not the system prompts the user for their google information every time or only if they are not logged in.
- Save login credentials

Fixed Functional Requirement:

The system shall prompt the user to input their Google account information the first time the program opens, and save their login credentials before the user is logged in.

Bug #11:

Old User Story:

As a business executive, I want to automatically arrange meetings with my clients, so that I can be more productive during my working hours.

Bug:

Remove "automatically" from the user story

Fixed User Story:

As a business executive, I want to arrange meetings with my clients, so that I can be more productive during my working hours.

Bug #12:

Old Functional Requirement:

The system shall combine two or more calendars from a Google account into one calendar at the user's request.

Bug:

Maybe change "two or more" to multiple.

Fixed Functional Requirement:

A revision for this requirement is: The system shall display the events of each user's calendars as one master calendar.

However, this requirement has been used. The revision is: The system shall allow a group member to view the other members in the group.

Bug #13:

Old Functional Requirement:

The system shall run on an IOS 8 or above.

Bug:

IOS8 or above implies a different meaning than IOS8 and above. Simply replace 'or' with 'and' to clarify the requirement.

Fixed Functional Requirement:

The system shall run on all systems with IOS 8 and above.

Bug #14:

Old Functional Requirement:

The system shall send an email to remind each user of a meeting before the meeting time, allowing a user to specify the notification time.

Bug:

The requirement only specifies to send reminder 'before the meeting'. This is vague and needs to be replaced with a concrete time frame.

Fixed Functional Requirement:

The system shall send notifications of upcoming events to the user's provided email within a 1 minute window of the user specified notification time.

Bug #15:

Old User Story:

As a student, I want to sync my class schedule with my work schedule automatically, so that I can be more effective with how I plan my time.

Bug:

Remove "automatically" from the user story.

Fixed User Story:

As a student, I want to sync my class schedule with my work schedule, so that I can be more effective with how I plan my time.

Bug #16:

Old Functional Requirement:

The system shall combine the events of each user's calendars into one common calendar to show common free time.

Bug:

The use of the word common in "common free time" is redundant. If it is free time, it is already common.

Fixed Functional Requirement:

The system shall display the events of each user's calendars as one master calendar.

Bug #17:

Old Functional Requirement:

The system shall collect the user's Google account information and connect to the Google API when a user signs in.

Bug:

The FR does not describe a "function", but instead focuses on implementation.

Fixed Functional Requirement:

This requirement is redundant. Using revision: The system shall allow a user to select an interval for a recurring event.

Bug #18:

Old Functional Requirement:

The system shall depend on calendar service APIs to import calendars.

Bug:

Unnecessary information, don’t need to specify the API’s.

Fixed Functional Requirement:

The system shall display a user’s calendar information when that user signs in.