

Calendar Event and Task List Manager



Prepared by Team Thundercats

Shubhangi Rakhonde

David Schechter

Zayd Hammoudeh

Table of Contents

1. Application Name	1
2. Application Description	1
2.1. Application Purpose.....	1
2.2. Application Overview	1
3. Functional Requirements.....	1
3.1. General Functional Requirements	1
3.2. Functional Requirements Related to Events and Tasks.....	1
3.3. Functional Requirements Related to Event and Task Priority.....	2
3.4. Functional Requirements Related to Synchronous Calendar Events	2
3.5. Functional Requirements Related to Asynchronous Task Completion.....	3
4. Nonfunctional Requirements.....	3
4.1. General Nonfunctional Requirements.....	3
4.2. Nonfunctional Requirements Related to Calendar Events	3
4.3. Nonfunctional Interface Requirements	3
4.4. Nonfunctional Security Requirements	5
5. Use Cases	6
5.1. Log in User	6
5.2. Add a Third Party Calendar	7
5.3. Sort Asynchronous Tasks	8
5.4. Mark Asynchronous Tasks Completed.....	9
5.5. Create Calendar Event.....	10

1. Application Name

Calendar Event and Task List Manager

2. Application Description

This section provides an overview of the application, including its intended purpose.

2.1. Application Purpose

This application simplifies the management (e.g. viewing, modifying, prioritizing, etc.) of a user's daily calendar as well as the user's to-do/task list by integrating these items into a single, cohesive interface.

2.2. Application Overview

It is common for a person to have multiple, distinct, daily calendars stored on different, disconnected platforms. For example, a business professional may have a work calendar on his/her company's corporate network, a personal calendar as part of his/her Google account, and a social calendar on Facebook. Managing and visualizing these disparate calendars can be cumbersome and difficult; this application simplifies this otherwise burdensome task by integrating all of a user's different calendars into a unified calendar where the user can visualize and modify all of his/her calendars using a single, cohesive interface.

In addition to scheduled events, meetings, and appointments, an individual usually must also complete a set of tasks, chores, errands, etc. These tasks may be professional, personal, or social. This application also integrates the ability to create and manage the user's tasks in the form of an advanced "to-do list".

By juxtaposing in a single interface an individual's calendar with the tasks s/he must perform, a user is able to easily visualize and prioritize all of his/her daily activities. Moreover, the application supports the ability to provide the user with alerts (e.g. SMS text message or email) to serve as event or task reminders. Therefore, this application's integrated approach helps prevent the inefficiencies and issues (e.g. belated completion of tasks) associated with what for most is an unstructured system to daily activity management.

3. Functional Requirements

This section reviews the event and task manager's functional features. The requirements have been organized into different categories based on the requirement's logical role in the application.

3.1. General Functional Requirements

- A. Before accessing this tool's features, the user must enter his/her username and password correctly into the application.
- B. If a user fails to enter valid login credentials, the application shall display an error message instructing the user to enter valid credentials or to create a new account.

3.2. Functional Requirements Related to Events and Tasks

- A. The application shall allow two types of user items; the application shall track these differing types separately since there is limited overlap in their nature.
 - i. The first item type is a user calendar event, which is "synchronous" in nature in that such an event is associated with a specific time and date. Examples of synchronous calendar events

- would be a user's: doctor's appointment, business meeting, birthday party, etc.; note that all of these synchronous events occur at specific times.
- ii. The second item type is a user task, which is "asynchronous" in that the user has significant flexibility regarding the time the task can be performed/completed. For instance, a possible asynchronous task could be to cut the grass or to go shopping since the user can do them when s/he sees fit or has time. In contrast, if the user only had access to a lawnmower from 2-3pm on a particular Saturday, mowing the grass could turn from being asynchronous to synchronous.
- B. The application shall allow the user to create notifications regarding certain events. These notifications will be sent by the application to the user at user specified times.

3.3. Functional Requirements Related to Event and Task Priority

- A. Users shall be able to specify a priority for each synchronous calendar event and asynchronous task.
- B. When the user clicks the "Sort" button on the interface, a drop down menu shall appear listing the different criteria (e.g. task priority, task creation date, and alphabetically by task name) by which the asynchronous tasks may be sorted.
- C. If the user clicks on a sorting criterion in the sort drop down menu, the application shall close the drop down menu and update the displayed ordering of the asynchronous tasks to match the user selected criteria.
- D. If the user clicks in a part of interface other than the sort drop down menu while that menu is being displayed, the application shall close the drop down menu and not change the order of the asynchronous tasks.

3.4. Functional Requirements Related to Synchronous Calendar Events

- A. The user shall be able to integrate, view, and modify their synchronous calendar events from other third-party applications/services.
- B. The user shall be able to create, view, and modify synchronous calendar events that were not imported from a third-party application.
- C. If when the user attempts to create a synchronous calendar event, s/he fails to enter all of the required information, the application shall alert the user of the missing information and then allow the user to enter the missing information before s/he tries to create the event again.
- D. The user shall be able to specify that a synchronous calendar event occurs at a specified regular interval (e.g. once a month, once a week, every Tuesday and Thursday, etc.), and the application must automatically replicate such an event, based off the user specified time recurrence profile.
- E. The application must support the ability to create and appropriately display multiple synchronous calendar events that overlap in time (e.g. two separate meetings scheduled at exactly the same time).
- F. A description is an optional field for a synchronous calendar event. If when creating or editing a synchronous event the user tries to add a description, the form shall lengthen automatically to make space for the description text the user is adding.

3.5. Functional Requirements Related to Asynchronous Task Completion

- A. The application must support the ability to categorize asynchronous tasks as either uncompleted or completed.
- B. Once an asynchronous task has been completed, the user shall be able to mark it as “Completed” by clicking a check box next to the task. Upon an asynchronous task’s completion, the application must automatically remove the task from the set of uncompleted tasks and include it in the set of completed tasks.
- C. Once a task has been marked as completed, the application must allow the user to specify the task’s completion time. This completion time can be either the current time, or the user shall be able to specify another time.

4. Nonfunctional Requirements

The following are a set of nonfunctional requirements for the event and task manager. The requirements have been categorized depending on each one’s logical type.

4.1. General Nonfunctional Requirements

- A. The application shall be accessible to the user through a web browser. At the minimum, the application needs to support Google Chrome since it is the most widely used browser with support for other browsers prioritized based on their user base size.
- B. As described in the section entitled “Functional Requirements Related to Events and Tasks”, the application can send the user notifications. Types of notifications that shall be supported include but are not necessarily limited to: SMS text messages, browser pop-ups, and emails.

4.2. Nonfunctional Requirements Related to Calendar Events

- A. The user shall be able to import synchronous calendar events from at least the following set of third-party platforms: Google Calendar, Facebook, and Apple’s calendar application.
- B. The user specifiable attributes for a calendar event shall be:
 - i. Event name
 - ii. Event time and date
 - iii. Event description (if any)
 - iv. Invitee List (if any)
 - v. Event recurrence (e.g. once a week, every Tuesday/Thursday - if any)

4.3. Nonfunctional Interface Requirements

- A. The application shall display synchronous calendar events and asynchronous tasks in a side-by side two panel view as shown in figure 1.
- B. Tasks and events with different priorities shall be color coded in the user interface. The color scheme used must make higher priority events and tasks appear more prominently than their lower priority counterparts.

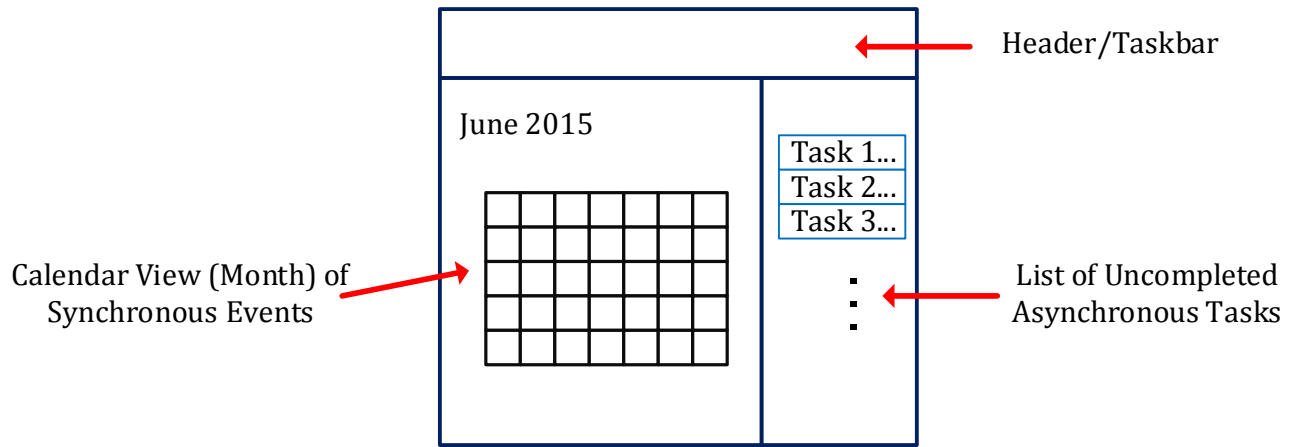


Figure 1 - Basic Structure of the User Interface

Name:	<input type="text"/>
Time:	<input type="text"/>
Description: (Optional)	
Priority: ★★★★★	<input type="button" value="Share"/> <input type="button" value="Create"/>

Figure 2 - Synchronous Event Creation Form Model

Name:	<input type="text"/>
Time:	<input type="text"/>
Description: (Optional)	
<input type="text"/>	
Priority: ★★★★★	<input type="button" value="Share"/> <input type="button" value="Create"/>

Figure 3 - Synchronous Event Creation Form Model with Expanded Description Entry Form

- C. The rating system used to prioritize tasks shall be on a scale of zero to five stars, with zero stars being the lowest priority and five stars being the highest priority.

- D. When creating a synchronous calendar event, the application form shall resemble the structure shown in figure 2.
- E. As detailed in the section entitled “Functional Requirements Related to Synchronous Calendar Events”, when the user attempts to add a description, the interface lengthens to make space for the event description text. When the form lengths to allow the user to enter a description, the interface shall resemble the event description field shown in figure 3.

4.4. Nonfunctional Security Requirements

- A. The application must encrypt all of the user’s data including his/her events, tasks, username, password, and the login credentials for third party tools whose calendar events are accessed by this application.
- B. The application shall keep each user’s data separate and unviewable by other users.

5. Use Cases

This section summarizes a set of uses cases the event and task manager must support. They have been included on separate pages for improved readability.

5.1. Log in User

Use Case Name: Log in User

Goal: The user logs into the application so that s/he can access his/her account information and use the application.

User Action	System Action
The user attempts to access the application in the browser.	The application prompts the user to enter his/her username and password.
The user enters login credentials and submits it to the system.	The application matches the user specified login credentials to a known set of credentials. The application gives the user access to the corresponding account's data and opens the application's home interface.
The user begins using the application.	

Use Case Variation #5.1.1– Invalid Login Information

User Action	System Action
The user attempts to access the application in the browser.	The application prompts the user to enter his/her username and password.
The user enters login credentials and submits it to the application.	The application fails to matches the user specified login credentials to a known set of credentials. The application prompts the user that the specified login is invalid and allows the user to enter new login information.
The user enters another set of login credentials and submits it to the system.	The application matches the new user specified login credentials to a known set of credentials. The application gives the user access to the corresponding account's data and opens the application's home interface.
The user begins using the application.	

Note: The incorrect login cycle could have been repeated multiple times. For the sake of brevity and conciseness, we provide only one variation with the knowledge it could be duplicated n times.

5.2. Add a Third Party Calendar

Use Case Name: Add a Third Party Calendar

Goal: The user adds a calendar hosted on another platform to this application for unified management.

User Action	System Action
The user clicks the button to add a third party calendar to their account.	The application displays a list of the supported third party calendars (e.g. Google, Facebook, Apple) and asks the user which type of account s/he would like to add.
The user selects the type of account calendar (e.g. Google) s/he wants to add.	The application prompts the user to enter the login information for an account of the user specified type.
The user enters the login information for the account s/he wants to add and submits it to the system.	The application successfully verifies the login information with the third party provider. The application alerts the user that the account was successfully added and syncs the data from the third party into the user's account.

Use Case Variation #5.2.1 – Invalid Third-Party Login Information

User Action	System Action
The user clicks the button to add a third party calendar to their account.	The application displays a list of the supported third party calendars (e.g. Google, Facebook, Apple) and asks the user which type of account s/he would like to add.
The user selects the type of account calendar (e.g. Google) s/he wants to add.	The application prompts the user to enter the login information for an account of the user specified type.
The user enters the login information for the account s/he wants to add and submits it to the system.	The application fails to successfully verify the login information with the third party provider. The application prompts the user that the third-party login credentials were invalid and allows the user to enter credentials again.
The user enters another set of login credentials for the account s/he wants to add and submits it to the system.	The application successfully verifies the new login information with the third party provider. The application alerts the user that the account was successfully added and syncs the data from the third party into the user's account.

Note: The incorrect third-party login cycle could have been repeated multiple times. For the sake of brevity and conciseness, we provide only one variation with the knowledge it could be duplicated n times.

5.3. Sort Asynchronous Tasks

Use Case Name: Sort Asynchronous Tasks

Goal: The user views the asynchronous tasks ordered according to a user specified sort criteria.

User Action	System Action
In the asynchronous task viewer pane, the user clicks the "Sort" button.	The application drops down a menu showing the available sorting criteria (e.g. alphabetical by name, priority, task creation date, etc.).
The user selects the sorting criteria according to which s/he would like to see the tasks ordered.	The application hides the sorting criteria drop down menu. The application sorts the tasks according to the user specified criteria. The application updates the user's view with the tasks appropriately sorted.

Use Case Variation #5.3.1 – User Cancels Sort Operation

User Action	System Action
In the asynchronous task viewer pane, the user clicks the "Sort" button.	The application drops down a menu showing the available sorting criteria (e.g. alphabetical by name, priority, task creation date).
The user clicks in an area of the interface other than the newly opened drop down list.	The application hides the sorting criteria drop down menu and does not change the remainder of the user's view.

5.4. Mark Asynchronous Tasks Completed

Use Case Name: Mark Asynchronous Tasks Completed

Goal: The user updates an asynchronous task in the application to be marked as completed.

User Action	System Action
The user clicks on the task s/he wants to mark completed.	The application displays the task information, including a check box the user can check to mark the task completed.
The user clicks the "Completed" check box.	The application displays the message, "Are you sure you want to mark this task as completed?" The available options are "Yes" and "No".
The user clicks "Yes".	The application prompts the user to enter a task completion time with the default being the current time.
The user enters the task completion time and clicks the "Finish" button to continue.	The application marks the task completed at the user specified time. The application returns to the task view menu and removes the task from the list of uncompleted tasks and adds it to the list of completed tasks.

Use Case Variation #5.4.1 – User Cancels Task Completed Operation

User Action	System Action
The user clicks on the task s/he wants to mark completed.	The application displays the task information, including a check box the user can check to mark the task completed.
The user clicks the "Completed" check box.	The application displays the message, "Are you sure you want to mark this task as completed?" The available options are "Yes" and "No".
The user clicks "No".	The mark asynchronous task completed operation terminates. The application returns the user to the task view menu, without updating the task.

5.5. Create Calendar Event

Use Case Name: Create Calendar Event

Goal: The user creates a new calendar event for his/her account.

User Action	System Action
The user clicks on the button to create a new calendar event.	The application displays the create calendar event form similar to what is shown in figure 2.
The user enters the calendar event information and clicks the “Create” button.	The application verifies the calendar event information is complete. The application adds the new calendar event to the user’s account data and returns to the event calendar view with the new calendar event displayed.

Use Case Variation #5.5.1 – Missing Calendar Event Form Information

User Action	System Action
The user clicks on the button to create a new calendar event.	The application displays the create calendar event form similar to what is shown in figure 2.
The user enters calendar event information and clicks the “Create” button.	The application determines the calendar event information is incomplete. The application prompts the user that a specific field(s) in the form is missing and allows the user to enter event the information.
The user enters all of the missing, required calendar event information and clicks the “Create” button.	The application verifies the calendar event information is complete. The application adds the new calendar event to the user’s account data and returns to the event calendar view with the new calendar event displayed.

Note: The action for the missing calendar event information could have been repeated multiple times. For the sake of brevity and conciseness, we provide only one variation with the knowledge it could be duplicated n times.

Use Case Variation #5.5.2 – Optional Event Description Used

User Action	System Action
The user clicks on the button to create a new calendar event.	The application displays the create calendar event form similar to what is shown in figure 2.
While adding event information, the user clicks on the “Description” field to enter an event description.	The create event form lengthens to allow the user to enter an event description as shown in figure 3.
The user enters the calendar event information and clicks the “Create” button.	The application verifies the calendar event information is complete. The application adds the new calendar event to the user’s account data and returns to the event calendar view with the new calendar event displayed.

