**Event and Task List Manager**



**Prepared by Team Thundercats**

David Smith

Shubhangi Rakhonde

David Schechter

Zayd Hammoudeh

# Table of Contents

[Table of Contents i](#_Toc410509417)

[1. Application Name 1](#_Toc410509418)

[2. Application Description 1](#_Toc410509419)

[2.1. Application Purpose 1](#_Toc410509420)

[2.2. Application Overview 1](#_Toc410509421)

[3. Functional Requirements 1](#_Toc410509422)

[3.1. General Functional Requirements 1](#_Toc410509423)

[3.2. Functional Requirements Related to Events and Tasks 1](#_Toc410509424)

[3.3. Functional Requirements Related to Event and Task Priority 2](#_Toc410509425)

[3.4. Functional Requirements Related to Synchronous Calendar Events 2](#_Toc410509426)

[3.5. Functional Requirements Related to Asynchronous Task Completion 2](#_Toc410509427)

[4. Nonfunctional Requirements 3](#_Toc410509428)

[4.1. Security Related Nonfunctional Requirements 4](#_Toc410509429)

# Application Name

Event and Task List Manager

# Application Description

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

## 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.

## 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 specific events, meetings, and appointments an individual has scheduled, a user usually must also complete a set of tasks as well. These tasks may be professional, personal, social, etc. This application also integrates the ability to create and manage the user’s tasks in the form of a “to-do list”.

By juxtaposing in a single interface a user’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 will support the ability to provide user alerts via SMS text message or email to serve as event reminders. Such a system streamlines and simplifies a process which today is often managed in an unorganized manner. The application’s integrated approach helps prevent the inefficiencies and issues (e.g. belated completion of tasks) associated with what is otherwise an unstructured system.

# Functional Requirements

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

## General Functional Requirements

1. Before accessing this tool’s features, the user must enter his/her username and password correctly into the application.
2. 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.

## Functional Requirements Related to Events and Tasks

1. 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.
   1. The first type is a user event, which is “synchronous” in nature in that such an event is associated with a specific time and date on the calendar. Examples of synchronous events would be a user’s: doctor’s appointment, business meeting, and birthday party; note that all of these synchronous events occur at specific times.
   2. The second 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 they see fit. In contrast, if the user only had access to the lawnmower from 2-3pm on Saturday, mowing the grass could turn from being asynchronous to synchronous.
2. The application shall allow the user to create notifications regarding certain events. These notifications will be sent by the application to the user at specified times.

## Functional Requirements Related to Event and Task Priority

1. For each synchronous calendar event and asynchronous tasks, the application shall allow the user to specify or change the item’s priority on a scale of zero to five (stars), with zero being the lowest priority and five being the highest priority.
2. Tasks and events with different priorities shall be color coded in the system. The color scheme used must make higher priority events and tasks appear more prominently than their lower priority counterparts.

## Functional Requirements Related to Synchronous Calendar Events

1. The application shall allow the user the user to integrate their synchronous calendar information from other applications/services, including but not necessarily limited to: Google Calendar, Facebook, and Apple’s calendar application.
2. The application must support the ability to synchronous calendar events that were not imported from another application (e.g. Facebook). When creating such an event, the application must support the ability for the user to specify the following information:
   1. Event name
   2. Event time and date
   3. Event description (if any)
   4. Invitee List (if any)
   5. Event recurrence (e.g. once a week, every Tuesday/Thursday - if any)
3. 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).

## Functional Requirements Related to Asynchronous Task Completion

1. The application must support the ability to categorize asynchronous tasks as either uncompleted or completed.
2. 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.
3. Once a task has been marked as completed, the application must allow the user to specify the task completion time. This completion time can be either the current time, or the application shall allow the user to specify another time.
4. Upon an asynchronous task’s completion, the application must automatically remove the task from the set of uncompleted tasks and instead include it in the set of completed tasks.

# Nonfunctional Requirements

The following are set of nonfunctional requirements for the event and task manager. They have been categorized depending on the nonfunctional requirement’s logical role.

## Nonfunctional Requirements

1. 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 most used browser with support in other browsers prioritized based on their user base size.
2. As described in section “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.

## Nonfunctional Interface Requirements

1. The application shall display synchronous events and asynchronous tasks in a side-by side two panel view as shown in figure 1.



Figure – Basic Structure of the User Interface

1. When creating a synchronous event, the application form shall resemble the structure shown in figure 2.



Figure – Synchronous Event Creation Form Model

1. As described in section “Functional Requirements Related to Synchronous Calendar Events”, the event description field is optional. If the user clicks on the description field shown in figure 3, the form shall lengthen to allow the user to enter an event description.



Figure – Synchronous Event Creation Form Model with Expanded Description Entry Form

1. The application’s “Share” button (shown in figures 2 and 3) shall open to another form to allow the user share the event with others via email or third party platforms (e.g. Facebook).

## Nonfunctional Security Requirements

1. 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 is synced by this application.
2. The application shall keep each user’s data separate and unviewable by other users.