|  |
| --- |
| DESIGN USE CASES |



FLEX: SCHEDULING APPLICATION

|  |  |  |  |
| --- | --- | --- | --- |
| SAN DIEGO SOFTWARE  February 12, 2017   |  |  |  | | --- | --- | --- | | **Orr Yakobi**, Project Manager  **Aidan Keogh**, Software Dev Lead  **Amanda Smith**, UI Specialist | **Anand Kumar**, Senior System Analyst  **Oliver Ganguli**, Database Specialist  **Steven Lay**, Business Analyst | **Tal Ben-Ari**, Software Architect  **Abram Trinh**, Quality Assurance  **Matt Olsen**, Database Specialist | |

# Overview

## User Account Management (UA)

1. Log In
2. Log Out

## Task List Management (TM)

1. Create Task
2. Modify Task
3. Delete Task
4. Task Completion
5. Task Scheduling
6. Conflict Resolution

## Task View (TV)

1. View Tasks
2. Sort Tasks
3. Filter Tasks
4. Search for Task

## Calendar View (CV)

1. View Calendar
2. Search For Event
3. Task Details

## Settings Page (SP)

1. View Settings
2. Update Settings

## User Support (US)

1. Access the Help Page
2. Contact Customer Support

# 

# References

## Task/Event Fields (REF 01):

1. Description
2. Priority
3. Date
4. Time
5. Location
6. Duration
7. Recurring Task/Not Recurring Task
8. Flexibility

## Calendar Fields (REF 02):

1. Description
2. Color
3. Priority

## Database Models (REF 03): (directory : sds-webapp/server/models/)

1. User
2. Group
3. Calendar
4. Task

## HTML pages (REF 04):

1. userAuth.html
2. calendarWeeklyView.html
3. createTask.html
4. createCalendar.html
5. modifyTask.html
6. modifyCalendar.html
7. userSettings.html
8. userSupport.html

## Google JSON fields (GOOG 01):

<https://developers.google.com/google-apps/calendar/v3/reference/events>

# Design Use Case Variable Key

**Priority Level:**

|  |  |
| --- | --- |
| **1** | Core Functionality, Critical to application |
| **2** | Extended Functionality, Noncritical to application but of value |
| **3** | Future Extension, Noncritical to application and of low importance |

**Status:**

|  |  |
| --- | --- |
| **Implemented** | Use case is implemented into application and is functioning. |
| **Incompleted** | Use case is in the progress of being implemented into the application. |
| **Delayed** | Use case’s development is put on hold. |

**Test-phase status:**

|  |  |
| --- | --- |
| **Completed** | Test Phase complete and use case is functional. |
| **In Progress** | Test Phase is in the process of being tested |
| **Incompleted** | Test phase is not being tested. |

# User Sign In [UA 01]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to log into the application. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The user shall be able to sign in via their Google Account using the OAuth 2.0 protocol and view their information. |
| USER GOALS | The user wants to view their personalized tasks and events. |
| DEPENDENT USE CASES | * (none) |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Complete |
| PRE-CONDITIONS | * The user has opened the webpage and is at the Sign in Page. * The user is not currently signed in |
| POST-CONDITIONS | * The system shall redirect the user to the main screen. * The user shall be signed into the website and be able to view their information. |
| TRIGGER | The user selects the Sign in button. |
| WORKFLOW | 1. The user visits the homepage (with the URL ‘/’). 2. The backend prepares the homepage information for the client. During server-side rendering, the server looks for a cookie with the request but does not find one, so it prerenders the sign in dialog. 3. The frontend attaches its library hooks to the prerendered content. 4. The frontend looks for an authentication cookie and does not find one. 5. The user clicks the signin button, which redirects them to ‘/auth/google’. 6. The backend hands control to the Passport library with a Google OAuth 2.0 Strategy and the necessary credentials. 7. Passport redirects the user to Google’s OAuth account sign in webpage, and awaits a response from Google. 8. Google redirects the user to the callback URL provided by Passport, ‘/auth/google/callback’. 9. The backend prepares the homepage information for the client again. This time, during server-side rendering, the server finds a cookie with the request, so it deserializes the authentication token and prerenders the Calendar Weekly View module with the user’s information. 10. The frontend attaches its library hooks to the prerendered content. 11. The frontend finds an authentication cookie and stores it in a new instance of an Authentication module, which will be used in future AJAX requests. |
| ALTERNATE PATHS | 1. The user visits the homepage (with the URL ‘/’). 2. The backend receives a valid authentication cookie. The system proceeds from step 9 of the default workflow. |

# 

# 

# User Sign Out [UA 02]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to sign out of the application. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The user shall be able to sign out of their account. The system shall preserve all changes to the user’s data. |
| USER GOALS | The user is finished using the application and wants to sign out to preserve security. |
| DEPENDENT USE CASES | * UA 01 - User Sign In |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user is currently signed in. * The user is at at any page with a Navigation Bar. |
| POST-CONDITIONS | * The system shall preserve all modifications made by the user. * The system shall redirect the user to the Sign in page. |
| TRIGGER | The user wants to switch accounts or sign out for security reasons. |
| WORKFLOW | 1. The user visits any valid URL (see “REF 04”). 2. The backend prerenders the page and sends it to the browser. 3. The frontend attaches its library hooks to the prerendered content. 4. The user clicks the “Sign Out” button in the top right corner of the application. 5. The frontend deletes the authentication token from the cookie. 6. The frontend destroys the Authentication instance. 7. The frontend renders the sign in dialog. |
| ALTERNATE PATHS | 1. The user visits any valid URL (see “REF 04”). 2. The backend looks for a cookie in the request but does not find one. It prerenders the sign in dialog and sends it to the browser. 3. The frontend attaches its library hooks to the prerendered content. |

# 

# Create Task [TM 01]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to create a task. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The user shall be able to create new tasks which will be integrated into their calendar. |
| USER GOALS | The user has and wants to complete a task within the constraints of their existing schedule. |
| DEPENDENT USE CASES | * UA 01 - User Sign In * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall be on Task View. * The user has a task in mind to create. |
| POST-CONDITIONS | * The system shall update the relevant calendar with the new task. |
| TRIGGER | The user wants to add a new responsibility to their schedule. |
| WORKFLOW | 1. The user shall press on the “New Task” button. 2. The frontend shall present an add task dialog to the user. 3. The user shall fill in the necessary task details. 4. The backend shall add the task to the task list’s table in the database. 5. The backend shall add the task to the calendar’s table in the database. |

# 

# 

# Modify Task [TM 02]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to modify a task. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The user shall be able to modify the details of their task |
| USER GOALS | The user wants to update the details of their task so that the details are up-to-date. |
| DEPENDENT USE CASES | * TD 01 - Creating Tasks * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall have a task. * The user shall be on the Tasks page. |
| POST-CONDITIONS | * The system shall modify the task to the calendar and task list. * The system shall display an updated view of their task list. |
| TRIGGER | The user wants to change the details in a previously created task. |
| WORKFLOW | 1. The user shall have selected a task. 2. The frontend shall enable the task fields to be modified. 3. The user shall change any field they choose. 4. The backend shall overwrite all the fields of the task in the database table containing the task fields. 5. The frontend shall refresh the page. |

# 

# 

# Delete Task [TM 03]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to delete a task. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The user will have tasks deleted from their calendar and task list. |
| USER GOALS | The user wants to remove tasks to keep an up-to-date calendar and task list. |
| DEPENDENT USE CASES | * TM 01 - Creating Tasks * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall be on the task page. * The user shall have tasks in their calendar and task list. |
| POST-CONDITIONS | * The system shall remove the task from the calendar and task list. * The system shall display an updated view of their calendar and task list. |
| TRIGGER | The user wants to remove a task from their calendar and task list. |
| WORKFLOW | 1. The frontend shall display the list of the tasks to be completed. 2. The user shall select a task they wish to delete. 3. The user shall click the trashcan button displayed on the right corner of the task. 4. The backend shall remove the task from the task list’s table in the database. 5. The frontend shall redirect the user back to the task list. |
| ALTERNATE PATHS | 1. The user does not want to delete the task from the task list and calendar after they have clicked the trashcan button. 2. The user shall be on Step 2. 3. The user shall click on the red ‘x’ button. 4. The backend shall redirect the user back to the task list. |

# 

# Task Completion [TM 04]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to mark a task as completed. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The user no longer sees the completed task in their upcoming schedule. |
| USER GOALS | The user wants to mark one of their task as completed. |
| DEPENDENT USE CASES | * TM 01 - Creating Tasks * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall be on the task page. * The user shall have completed tasks task list. |
| POST-CONDITIONS | * The system shall remove the completed task from the schedule. * The system shall display an updated task list. |
| TRIGGER | The user wants to mark a task done so that they can have an uncluttered view of unfinished tasks. |
| WORKFLOW | 1. The frontend shall display a list of tasks that are scheduled. 2. The user shall click the checkbox button next to the tasks they have completed and no longer wish to be rescheduled. 3. The backend shall delete tasks selected from task list in the task list’s table in the database. 4. The frontend shall render the user’s calendar with the uncompleted items. |
| ALTERNATE PATHS | None |

# 

# 

# Task Scheduling [TM 05]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to have tasks scheduled into their calendar. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The user’s selected tasks are scheduled into the calendar. |
| USER GOALS | The user wants to have tasks from their tasklist scheduled into their calendar. |
| DEPENDENT USE CASES | * TD 01 - Creating Tasks * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall have tasks on their task list. |
| POST-CONDITIONS | * The system shall update the calendar to reflect the new schedule. |
| TRIGGER | The user wants their list of tasks to be scheduled into their schedule. |
| WORKFLOW | 1. The frontend shall display the task list with all the user’s tasks. 2. The user shall click on the “Schedule Tasks” button. 3. The backend shall use the optimization algorithm to find where to add the selected tasks to the calendar. 4. The backend shall insert the tasks into the calendar. 5. The backend shall update the calendar information with the added tasks. 6. The frontend shall refresh the page. |
| ALTERNATE PATHS | 1. The user selects a task that conflicts with an existing event on the calendar.    1. The user shall complete the TM06 Workflow. 2. The user wants to deselects a task that they want to be added to their calendar.    1. The user shall deselect the task selected.    2. The frontend will display the task they selected without a marking. |

# 

# View Tasks [TV 01]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to view their list of tasks. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The task list shall display tasks that have been created. |
| USER GOALS | The user wants to view their task items that they have created. |
| DEPENDENT USE CASES | * User Sign In [UA 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 1 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall be viewing the “My Tasks” page. |
| POST-CONDITIONS | * The frontend shall display the user’s list of tasks. |
| TRIGGER | The user wants to get an overview of their tasks. |
| WORKFLOW | 1. The frontend shall request the user’s task information from the backend. 2. The backend shall load all task information from the database and the Google Calendar JSON objects. |
| ALTERNATE PATHS | none |

# 

# Sort Tasks By Priority [TV 02]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to sort the task list by user-defined priority. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The task list shall display tasks that have been ordered by priority. |
| USER GOALS | To view their task items in order of their priority. |
| DEPENDENT USE CASES | * User Sign In [UA 01] * Create Task [TM 01] * View Tasks [TV 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 3 * Status: Delayed * Test-phase status: Delayed |
| PRE-CONDITIONS | * The user shall be viewing the “Task View” page * The user shall have existing tasks |
| POST-CONDITIONS | * The “Task View” page shall display all tasks in order of priority. |
| TRIGGER | The user wants to view their tasks in order of importance. |
| WORKFLOW | 1. The user shall click the “Sort By Priority” button. 2. The backend shall retrieve all of the task objects from the database. 3. The frontend shall sort the tasks by their priorities. 4. The frontend shall display the newly sorted tasks. |
| ALTERNATE PATHS | 1. (none) |

# 

# Filter Tasks[TV 03]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to filter the task list to only show tasks from a specific calendar. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The task list shall only display tasks on a specific calendar. |
| USER GOALS | To view only tasks that are on a specific calendar. |
| DEPENDENT USE CASES | * User Sign In [UA 01] * Create Task [TM 01] * View Tasks [TV 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 2 * Status: Cancelled * Test-phase status: Cancelled |
| PRE-CONDITIONS | * The user shall be on the task view page. |
| POST-CONDITIONS | * The task list shall only display tasks from the specified calendar. |
| TRIGGER | The user wants to see the tasks on a specific calendar. |
| WORKFLOW | 1. The user shall click the button “Filter by calendar.” 2. The backend shall retrieve the user’s calendar list in the user’s settings object from the database. 3. The frontend shall render the user’s calendar list. 4. The user shall click on the calendar they wish to filter by. 5. The frontend shall hide all tasks that do not belong to the selected calendar. |
| ALTERNATE PATHS | 1. The user decides not to filter by calendar.    1. The user shall complete steps 1-3.    2. The user shall click the button “Cancel.”    3. The frontend shall close the display showing the user’s calendars. |

# Task Search [TV 04]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to search the task list to find specific tasks. |
| ACTOR | The webpage user. |
| DESIRED OUTCOME | The task list will only display tasks that the user searched for. |
| USER GOALS | To find a specific task in their task list. |
| DEPENDENT USE CASES | * User Sign In [UA 01] * Creating Tasks [TM 01] * View Tasks [TV 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 3 * Status: Cancelled * Test-phase status: Cancelled |
| PRE-CONDITIONS | * The user shall be on the task view page. |
| POST-CONDITIONS | * The task list will only display tasks containing the searched string. |
| TRIGGER | The user wants to find a task. |
| WORKFLOW | 1. The user shall click on the search bar. 2. The user shall enter the name of their task. 3. The frontend shall hide all tasks whose “ID” fields do not contain the entered string. |
| ALTERNATE PATHS | 1. The entered string does not match any tasks.    1. The user shall complete steps 1-3.    2. The frontend shall not find any tasks whose names contain the entered name.    3. The frontend shall display the message “Task not found.” |

# 

# 

# 

# 

# View Settings [SP 01]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to view all of the settings available to them. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The task list will only display the current setting. |
| USER GOALS | To view the current settings of their calendar. |
| DEPENDENT USE CASES | * User Sign In [UA 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 2 * Status: Cancelled * Test-phase status: Cancelled |
| PRE-CONDITIONS | * The user shall have created an account. * The user shall be logged into the web application. |
| POST-CONDITIONS | * The system shall redirect the user to the settings page. |
| TRIGGER | The user wants to better understand how their events are scheduled. |
| WORKFLOW | 1. The user shall click on the “settings” icon. 2. The frontend shall redirect the user to the “Settings” page. 3. The backend shall retrieve the user’s current settings. 4. The frontend shall display the user’s current settings on the “Settings” page. |
| ALTERNATE PATHS | None |

# 

# 

# Change Settings [SP 02]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to change one or more of their settings. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The user shall be able add modifications to their current settings. |
| USER GOALS | To change their settings for our web application. |
| DEPENDENT USE CASES | * User Sign In [UA 01] |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 2 * Status: Cancelled * Test-phase status: Cancelled |
| PRE-CONDITIONS | * The user shall have created an account |
| POST-CONDITIONS | * The system shall update the user’s setting attributes. * The system shall alert the user that settings has been updated. |
| TRIGGER | The user wants to change how their events are scheduled. |
| WORKFLOW | 1. The user shall click on the “setting” icon. 2. The frontend shall redirect the user to the “Settings” page. 3. The backend shall retrieve the user’s current settings. 4. The frontend shall display the user’s current settings on the “Settings” page. 5. The user shall make changes to their settings accordingly. 6. The user shall click the “save” button. 7. The frontend shall send the form data to the backend. 8. The backend shall overwrite all the fields of the user’s settings in the corresponding database table. 9. The frontend alerts the user that changes are successfully made. |
| ALTERNATE PATHS | 1. The user decides not to make any changes to their settings    1. The user shall click “cancel”    2. The system shall redirect the user back to the page they were originally at. |

# 

# Accessing The Help Page [US 01]

|  |  |
| --- | --- |
| DESCRIPTION | Outlines the ability of the user to access documentation intended to assist the user. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The user shall be able to know more about what the application does. |
| USER GOALS | The user wants to know more about what the application does or learn how to use the application correctly. |
| DEPENDENT USE CASES | * TV 01 - View Tasks |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 2 * Status: Implemented * Test-phase status: Completed |
| PRE-CONDITIONS | * The user shall be viewing either the task page. |
| POST-CONDITIONS | * The system shall display the help page. |
| TRIGGER | The user wants to find out more about the application. |
| WORKFLOW | 1. The user shall click on the “help” button. 2. The frontend shall display the help page (REF 04). |
| ALTERNATE PATHS | None. |

# Contacting Customer Support [US 02]

|  |  |
| --- | --- |
| DESCRIPTION | This use case outlines the ability of the user to contact customer support. |
| ACTOR | The webpage user |
| DESIRED OUTCOME | The user shall be able to learn more about the application and resolve any issues by contacting customer support. |
| USER GOALS | The user wants to inquire more about the application and/or fix any problems they encounter. |
| DEPENDENT USE CASES | * UA 01 - User Sign In * US 01 - Accessing the Help Page |
| INVOLVED REQUIREMENTS | * (To be determined upon the completion of development) |
| DETAILS | * Priority: 3 * Status: Incomplete * Test-phase status: Incomplete |
| PRE-CONDITIONS | * The user shall be viewing userSupport.html (REF 04). |
| POST-CONDITIONS | * The user shall be in contact with customer support. |
| TRIGGER | The user wants to resolve a conflict or inquire about the application. |
| WORKFLOW | 1. The user shall click “Contact Us” 2. The frontend shall display text fields for the user to enter their name, email, and message for customer support. 3. The user shall enter the desired information into the text fields. 4. The user shall click submit 5. The backend shall compose an email using the user’s entered name, email, and message. 6. The backend shall send the email to customer support. 7. The frontend shall display a confirmation message informing the user that the message has been sent. |
| ALTERNATE PATHS | 1. The user shall decide not to send a message to customer support    1. At any point during the workflow steps 1-3, the user shall choose not to click submit and instead navigate to another webpage. |