Use Cases

for

Tempus

**Version 0.2a**

**Prepared by**

**The Tempus Terriers (TTT) – Team 5**

**Michael Yuja, Calvin Liang, Kathleen McKay, Thomas Hsu**

**March 24, 2016**

Use Case List

|  |  |  |
| --- | --- | --- |
| ***ID*** | ***Primary Actor*** | ***Use Case Title*** |
| UC0 | Team Leader | Team/Meeting Creation |
| UC1 | Individual Student | Independent Study Place |
| UC2 | Team Leader | Meeting Creation |
| UC3 | Team Member | Team Member Perspective |
| UC4 | Meeting Initiator | Single Meeting Creation |

Use Case 0

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC0 | | |
| Use Case Name: | Meeting Creation | | |
| Created By: | All Team Members | Last Updated By: | Calvin, Michael, Kathleen |
| Date Created: | Feb. 16, 2016 | Date Last Updated: | Mar. 20, 2016 |

|  |  |
| --- | --- |
| Actors: | Team Leader + Team Members |
| Description: | Team leader is looking for a time and place for his team to meet. The output of the app will be a meeting time and place that is consistent with everyone’s schedules. |
| Trigger: | The team has a group project for which they need to meet. |
| Preconditions: | * + 1. All team members are BU students.     2. Their entire schedule is on Google Cal.     3. All calendar events have location data.     4. All team members have created an account on the site. |
| Postconditions: | * + - 1. Save meeting to team calendars.       2. Our server also saves the meeting time. |
| Normal Flow: | The primary actor is making a team in the app and sending the invites to the rest of the team. First, the primary actor fills out a form that specifies team name, description of the purpose of the team, emails of people who will be in the team and the permissions each person has in managing the team, and project timeline/milestones. The primary actor verifies the information and the app will send out the invites. All specified members receive the invites for the team and accept them. The primary actor creates a meeting by filling out a form that specifies the name and description of the meeting, the period the meeting needs to occur, cut off times for sleeping, place specifications (food, ambiance, access to these locations, etc.), the duration of the meeting, who is invited to the meeting, minimum number of people to attend, and the time selection deadline. The app suggests a meeting time and place to the primary actor. The actor then ‘vetoes’ any times that are inconvenient for the team. Email gets sent to everyone with updated times and places. Everyone selects all the times they are available by the deadline, and whichever time and place has the majority of people available will be the meeting time. The meeting is set and sent to everyone’s calendar. |
| Alternative Flows: | UC0.1: No possible meeting time for all members. System will select a time when the maximum amount of members can attend.  UC0.2: At least one member is not responding to the meeting invite. Person not responding is reminded to answer. Team leader is notified that the member is not responding and given the option to decrease minimum number of participants. If the cutoff time is reached the meeting will be set (or not) regardless of the non-response.  UC0.3: The members already know when and where they will be meeting, and the team creator opts out of using our algorithm in the meeting-creation process. The rest of the app functions the same way. |
| Exceptions: | One of our APIs is down or not accessible. The system would put up an error message and ask the user to check back later. |
| Includes: | Not currently applicable |
| Priority: | Top Priority |
| Frequency of Use: | Once per Team |
| Business Rules: | Time Algorithm: Starts with all time in the meeting period as available. Eliminates sleeping time. Goes through each person’s schedule and eliminates times the person is no longer available. When all schedules have been scanned it checks the currently available times for a slot big enough to fit the specified duration. The output are all the possible timeslots for meeting.  Place Algorithm: For each timeslot, take all the possible locations for meeting and all the locations of where people were immediately before the meeting time based on their calendars, and find the euclidean distance for each location. Whichever location has the least distance will be the best location to meet at. |
| Special Requirements: | Reliability: Crashes should not affect the state of the use case. |
| Assumptions: | Assume team coordinator is willing to take on the task of being responsible for interacting with the app. |
| Notes and Issues: | Some of these fields are subject to change pending further use case development. |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Group | 02/16/16 | Initial Commit | 0.1a |
| Michael, Kathleen, Calvin | 03/20/16 | Updating for suggested changes by the Professor | 0.2a |

Use Case 1

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC1 | | |
| Use Case Name: | Independent Study Place | | |
| Created By: | Calvin Liang | Last Updated By: | Calvin Liang |
| Date Created: | March 20, 2016 | Date Last Updated: | March 20, 2016 |

|  |  |
| --- | --- |
| Actors: | Individual Student |
| Description: | A student is looking to find a good place to do some independent studying and will use the app to do so. |
| Trigger: | Student has an exam coming up or wants to do homework |
| Preconditions: | 1. The individual is a BU student 2. Permission for Google Maps to access location services on the individual’s computer is turned on 3. The individual already has an account |
| Postconditions: | 1. The individual leaves a review + rating for the place they studied at 2. The individual can ‘favorite’ places they liked |
| Normal Flow: | The individual opens the app and selects “Individual” on the home page. The app will take the individual’s current location and provide a map with places nearby. We will implement a filtering system where the user can select places based on his/her preferences (ie. place is quiet, you can buy food, etc.). After studying, the individual can then choose to leave a review.. |
| Alternative Flows: | If the individual arrives at the location they selected and it’s either full or the individual does not like it, s/he can select the “Find Another Place” option and the app will display other closeby places. |
| Exceptions: | One of our APIs is down or not accessible. The system would put up an error message and ask the user to check back later. |
| Includes: | N/A |
| Priority: | Low Priority |
| Frequency of Use: | Multiple times based on when the individual wants to do work or study by him/herself |
| Business Rules: | Place Algorithm: Finds nearby locations (by calculating the euclidean distance) based on the individual’s preferences. |
| Special Requirements: | Reliability: Crashes should not affect the state of the use case. |
| Assumptions: | Assume location services is turned on on the individual’s device(s) where the web app is being accessed. |
| Notes and Issues: | Some of these fields are subject to change pending further use case development. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Calvin Liang | 3/20/16 | Initial Commit | 1.1a |
|  |  |  |  |

Use Case 2

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC2 | | |
| Use Case Name: | Meeting Creation | | |
| Created By: | Kathleen McKay | Last Updated By: | Kathleen McKay |
| Date Created: | 3/20/16 | Date Last Updated: | 3/20/16 |

|  |  |
| --- | --- |
| Actors: | Team Leader, Team Members |
| Description: | Team leader is looking for a time and place for his team to meet. The output of the app will be a meeting time and place that is consistent with everyone’s schedules. |
| Trigger: | The team has a group project for which they need to meet. |
| Preconditions: | 1. All team members are BU students. 2. Their entire schedule is on Google Cal. 3. All calendar events have location data. 4. All team members have created an account on the site. 5. The team has already been created and all members have joined. |
| Postconditions: | 1. Save meeting to team calendars. 2. Our server also saves the meeting time. |
| Normal Flow: | The primary actor creates a meeting by filling out a form that specifies the name and description of the meeting, the period the meeting needs to occur, cut off times for sleeping, place specifications (food, ambiance, access to these locations, etc.), the duration of the meeting, who is invited to the meeting, minimum number of people to attend, and the time selection deadline. The app suggests a meeting time and place to the primary actor. The actor then ‘vetoes’ any times that are inconvenient for the team. Email gets sent to everyone with updated times and places. Everyone selects all the times they are available by the deadline, and whichever time and place has the majority of people available will be the meeting time. The meeting is set and sent to everyone’s calendar. |
| Alternative Flows: | UC0.1: No possible meeting time for all members. System will select a time when the maximum amount of members can attend.  UC0.2: At least one member is not responding to the meeting invite. Person not responding is reminded to answer. Team leader is notified that the member is not responding and given the option to decrease minimum number of participants. If the cutoff time is reached the meeting will be set (or not) regardless of the non-response.  UC0.3: The members already know when and where they will be meeting, and the team creator opts out of using our algorithm in the meeting-creation process. The rest of the app functions the same way. |
| Exceptions: | One of our APIs is down or not accessible. The system would put up an error message and ask the user to check back later. |
| Includes: | Not currently applicable |
| Priority: | High |
| Frequency of Use: | 1+ |
| Business Rules: | Time Algorithm: Starts with all time in the meeting period as available. Eliminates sleeping time. Goes through each person’s schedule and eliminates times the person is no longer available. When all schedules have been scanned it checks the currently available times for a slot big enough to fit the specified duration. The output are all the possible timeslots for meeting.  Place Algorithm: For each timeslot, take all the possible locations for meeting and all the locations of where people were immediately before the meeting time based on their calendars, and find the euclidean distance for each location. Whichever location has the least distance will be the best location to meet at. |
| Special Requirements: | Reliability: Crashes should not affect the state of the use case. |
| Assumptions: | Assume team coordinator is willing to take on the task of being responsible for interacting with the app. |
| Notes and Issues: | Some of these fields are subject to change pending further use case development. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Kathleen McKay | 3/20/16 | Creation | 2.1a |
|  |  |  |  |

Use Case 3

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC3 | | |
| Use Case Name: | Team Member Perspective | | |
| Created By: | Michael Yuja | Last Updated By: |  |
| Date Created: | March 20, 2016 | Date Last Updated: | March 20, 2016 |

|  |  |
| --- | --- |
| Actors: | Team member |
| Description: | This is someone who is going to be part of a team, but will not take part in the creation process. This actor will only create an account for him/herself, and select meeting times that he/she is available. |
| Trigger: | The member has joined a team that will need to meet. |
| Preconditions: | 1. The actor needs to have all his schedule on Google Calendars. 2. All events have location data. 3. The actor is a BU student. |
| Postconditions: | 1. Save meeting time to calendar. |
| Normal Flow: | Member opens the app and starts to create an account. Member registers to the app with a Google account. The member can then see any pending team/meeting invites that someone sent to that address. The actor can accept/reject all the invites. If the member received an invite for a meeting, the member must select the times that he/she is good to meet at. Once a majority is reached for a time/place selection, the meeting will be added to the actor’s calendar. |
| Alternative Flows: | 0.1a: The member already has an account. The member receives any invitation notification through email and accesses the app through there. The rest of the process is the same as the normal flow from hereon. |
| Exceptions: | One of our APIs is down or not accessible. The system would put up an error message and ask the user to check back later. |
| Includes: | Not Currently Applicable |
| Priority: | Top |
| Frequency of Use: | Once for each meeting (after creating account) |
| Business Rules: | Time Algorithm: Starts with all time in the meeting period as available. Eliminates sleeping time. Goes through each person’s schedule and eliminates times the person is no longer available. When all schedules have been scanned it checks the currently available times for a slot big enough to fit the specified duration. The output are all the possible timeslots for meeting.  Place Algorithm: For each timeslot, take all the possible locations for meeting and all the locations of where people were immediately before the meeting time based on their calendars, and find the euclidean distance for each location. Whichever location has the least distance will be the best location to meet at. |
| Special Requirements: | Reliability: Crashes should not affect the state of the use case. |
| Assumptions: | Team leader is in charge of setting meetings and invites to all members. |
| Notes and Issues: |  |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Michael Yuja | March 20, 2016 | Initial commit() | 0.1 |
|  |  |  |  |

Use Case 4

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC4 | | |
| Use Case Name: | Single Meeting Creation | | |
| Created By: | Kathleen McKay | Last Updated By: | Kathleen McKay |
| Date Created: | Mar. 20, 2016 | Date Last Updated: | Mar. 20, 2016 |

|  |  |
| --- | --- |
| Actors: | Meeting Initiator, Meeting Participants |
| Description: | The Meeting Initiator wants to meet with another user(s) only once without creating a team. The output of the app will be a meeting time and place that is consistent with everyone’s schedules. |
| Trigger: | Some number of users want to meet and work together. |
| Preconditions: | 1. All users are BU students. 2. Their entire schedule is on Google Cal. 3. All calendar events have location data. 4. All users have created an account on the site. |
| Postconditions: | 1. Meeting time is saved to the users’ Google Cals. |
| Normal Flow: | The primary actor creates a meeting by filling out a form that specifies the name and description of the meeting, the period the meeting needs to occur, cut off times for sleeping, place specifications (food, ambiance, access to these locations, etc.), the duration of the meeting, who is invited to the meeting, minimum number of people to attend, and the time selection deadline. The app suggests a meeting time and place to the primary actor. The actor then ‘vetoes’ any times that are inconvenient for the team. Email gets sent to everyone with updated times and places. Everyone selects all the times they are available by the deadline, and whichever time and place has the majority of people available will be the meeting time. The meeting is set and sent to everyone’s calendar. |
| Alternative Flows: | UC0.1: No possible meeting time for all members. System will select a time when the maximum amount of members can attend.  UC0.2: At least one member is not responding to the meeting invite. Person not responding is reminded to answer. Team leader is notified that the member is not responding and given the option to decrease minimum number of participants. If the cutoff time is reached the meeting will be set (or not) regardless of the non-response.  UC0.3: The members already know when and where they will be meeting, and the team creator opts out of using our algorithm in the meeting-creation process. The rest of the app functions the same way. |
| Exceptions: | One of our APIs is down or not accessible. The system would put up an error message and ask the user to check back later. |
| Includes: | Not currently applicable. |
| Priority: | Medium |
| Frequency of Use: | Infrequent |
| Business Rules: | Time Algorithm: Starts with all time in the meeting period as available. Eliminates sleeping time. Goes through each person’s schedule and eliminates times the person is no longer available. When all schedules have been scanned it checks the currently available times for a slot big enough to fit the specified duration. The output are all the possible timeslots for meeting.  Place Algorithm: For each timeslot, take all the possible locations for meeting and all the locations of where people were immediately before the meeting time based on their calendars, and find the euclidean distance for each location. Whichever location has the least distance will be the best location to meet at. |
| Special Requirements: | Reliability: Crashes should not affect the state of the use case. |
| Assumptions: | Assume team coordinator is willing to take on the task of being responsible for interacting with the app. |
| Notes and Issues: | Some of these fields are subject to change pending further use case development. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Kathleen McKay | 3/20/16 | Creation | 4.1a |
|  |  |  |  |