TIME PIES
Use Case Specification

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:
Co, Patricia Kelly D.
Otsuka, Kenneth T.
Rubio, Mary Jane T.

In partial fulfillment of academic requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2014-2015

Revision Control

History Revision:

Revision Date	***Person Responsible***	***Version Number***	***Modification***
09/26/14	Co, Patricia Kelly Otsuka, Kenneth Rubio, Mary Jane	1.0	Initial Document; Version 1.0

Use-Case Name: 1.0 View To-do List

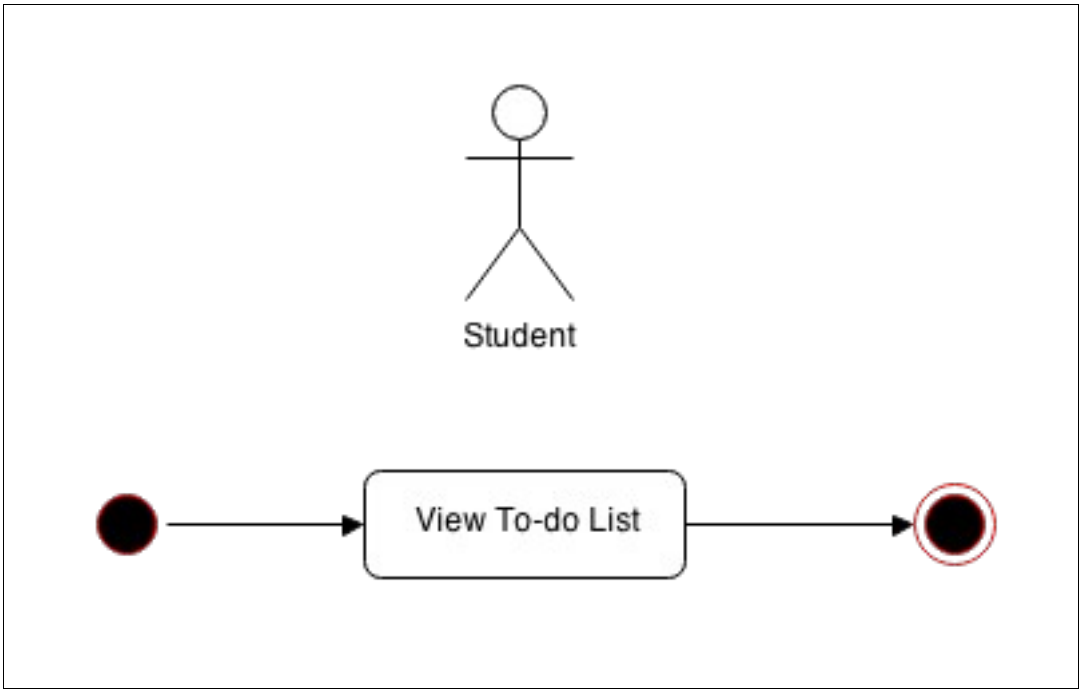
Description: The list stores the tasks listed by the user. View To-do List shows a column of tasks and a column of dates. Each row contains a task name and corresponding schedule.

Preconditions: NONE

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student views the To-do List	1. Student presses the View To-do List button. 2. The application shows the list of tasks inputted in the To-do List, if the list is empty then an empty/blank list is shown.

Activity Diagram of the Flow of Events:



Postcondition: NONE

Relationships: 2.0 Sort List, 3.0 Edit Cell, 4.0 Add Task to List, 5.0 Delete Task from List

Special Requirements: NONE

Use-Case Name: 2.0 Sort List

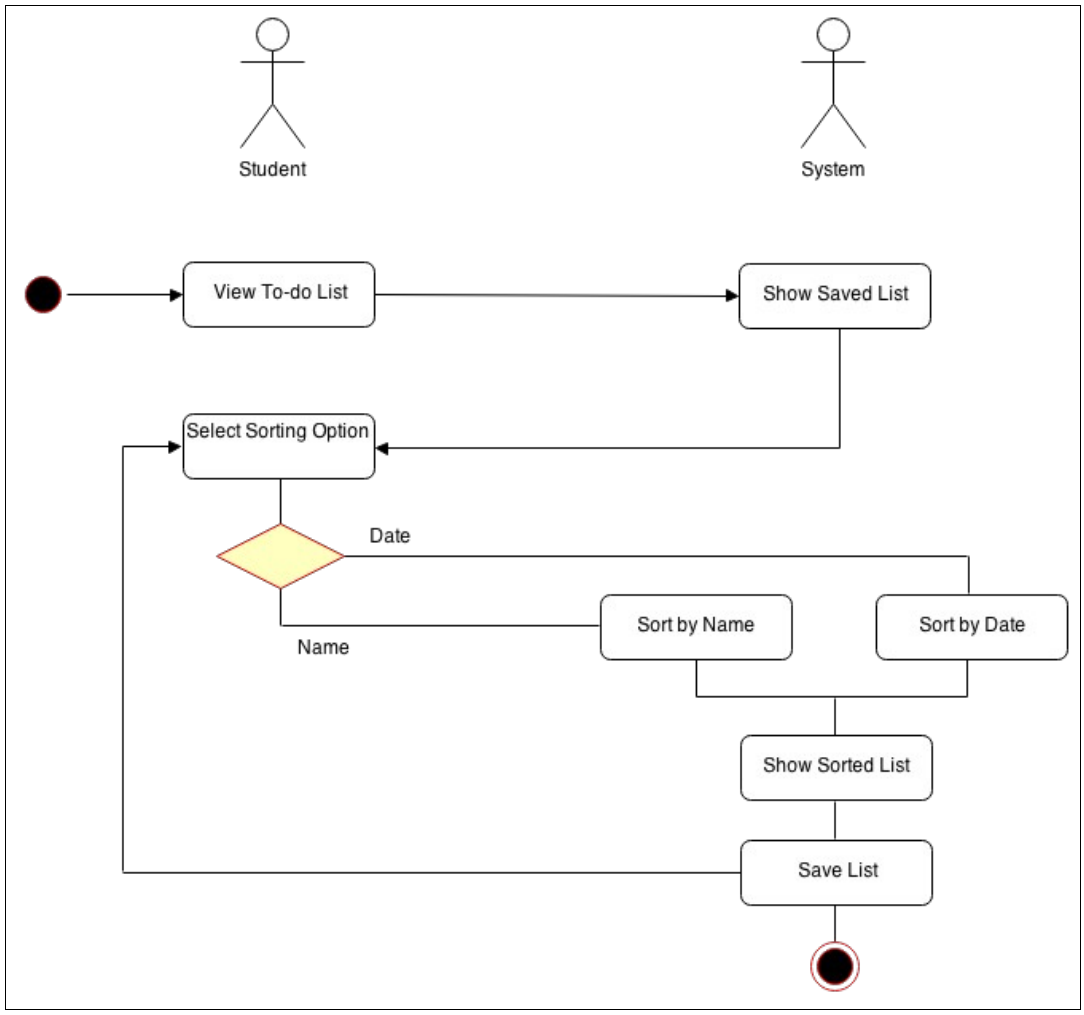
Description: By clicking the column header, the list can be sorted by task name and by date in alphabetical and chronological order, respectively. When sorted by date, tasks having the same date are arranged alphabetically. Tasks without a set date are grouped together alphabetically and placed at the bottom of the list upon sorting.

Preconditions: 1.0 View To-do List

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student chooses to sort	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User sorts the list. 4. The list is sorted. 5. The list is saved.
Scenario 2 The list is sorted by name at first usage	1. Student presses the View To-do List button. 2. The application shows an empty list. 3. By default, the list is sorted by name alphabetically.
Scenario 3 Student chooses to sort by date	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User sorts the list by date chronologically. If two tasks have the same date, they are sorted by name alphabetically. 4. The list is sorted by date. 5. The list is saved.
Scenario 4 Student chooses to sort by name	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User sorts the list by name alphabetically. 4. The list is sorted by name. 5. The list is saved.

Activity Diagram of the Flow of Events:



Postcondition: List is sorted (either by name or by date).

Relationships: 1.0 View To-do List, 3.0 Edit Cell, 4.0 Add Task to List, 5.0 Delete Task from List

Special Requirements: NONE

Use-Case Name: 3.0 Edit Cell

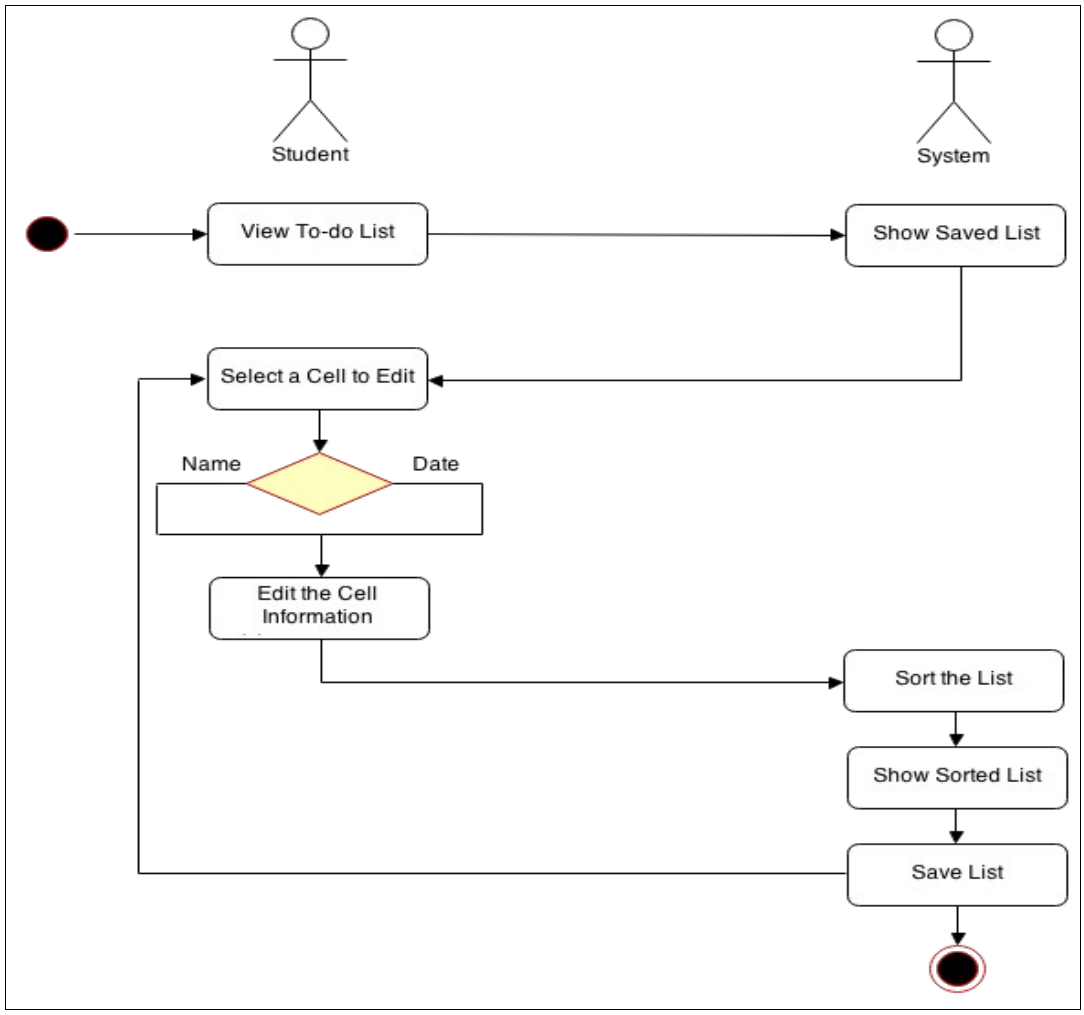
Description: The list is a table with two columns and a definite number of rows. It can be edited by editing its cells one at a time. The user can change the tasks' names and dates. After a cell is modified, the list is resorted.

Preconditions: 1.0 View To-do List, the list is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student edits a cell	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User clicks a cell to edit. 4. The information contained within the cell is edited. 5. The application sorts the list. 6. The list is sorted. 7. The list is saved.
Scenario 2 Student edits a task name cell	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User clicks a task name cell to edit. 4. The information contained within the cell is edited. The cell cannot be empty. 5. The application sorts the list. 6. The list is sorted. 7. The list is saved.
Scenario 2 Student edits a task date cell	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User clicks a task date cell to edit. 4. The information contained within the cell is edited. 5. The application sorts the list. 6. The list is sorted. 7. The list is saved.

Activity Diagram of the Flow of Events:



Postcondition: List is sorted (either by name or by date), the information within the cell has been modified.

Relationships: 1.0 View To-do List, 2.0 Sort List

Special Requirements: NONE

Use-Case Name: 4.0 Add Task to List

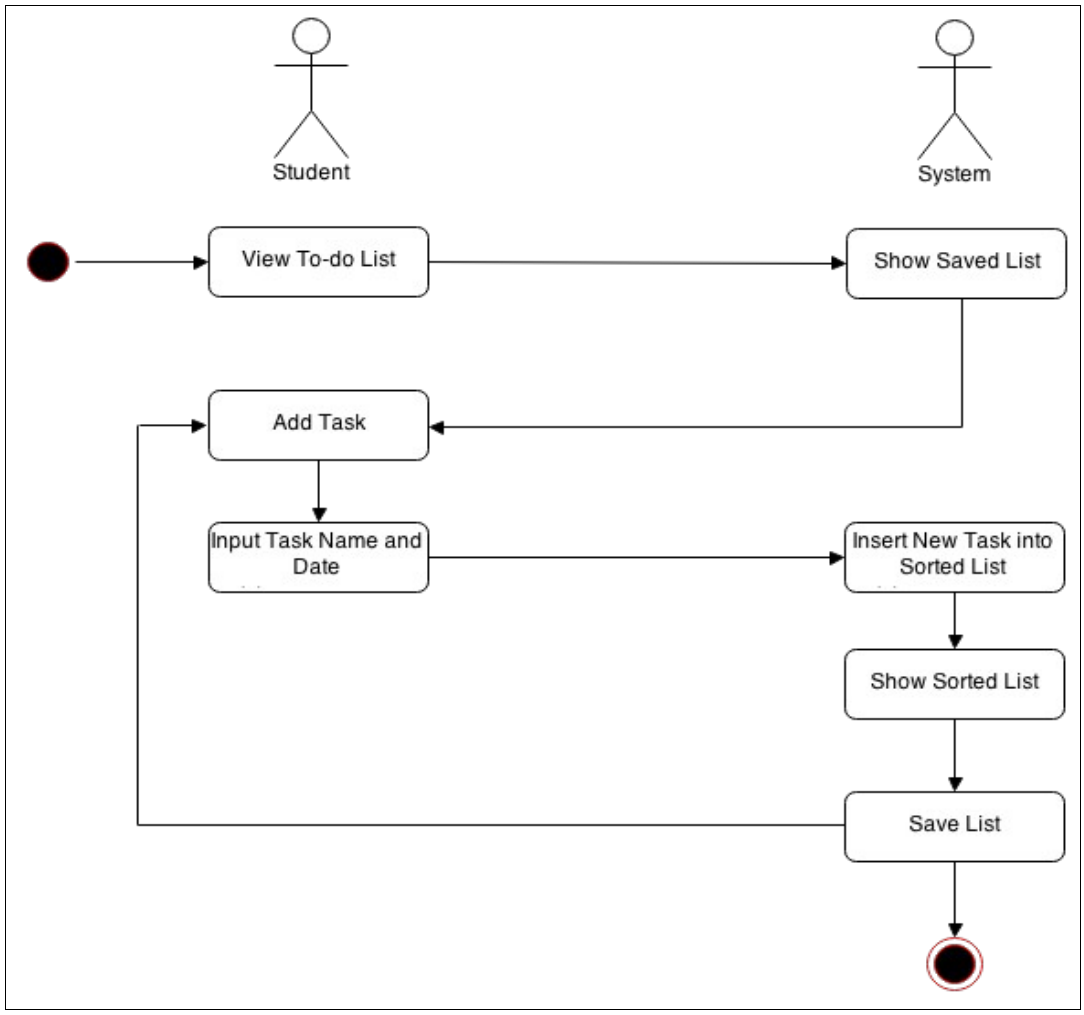
Description: A task is added by inputting its name/label. The user may or may not input a scheduled date. A new row containing the entry is inserted into the list based on how the list is currently sorted.

Preconditions: 1.0 View To-do List

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student adds a task to the list	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User clicks on Add Task button. 4. User inputs task name and date. (Date is optional) 5. A new row containing the entry is inserted into the list based on the list's current sorting option. 6. The list is sorted. 7. The list is saved.

Activity Diagram of the Flow of Events:



Postcondition: List is still sorted (either by name or by date) after the new task is added.

Relationships: 1.0 View To-do List, 2.0 Sort List

Special Requirements: NONE

Use-Case Name: 5.0 Delete Task from List

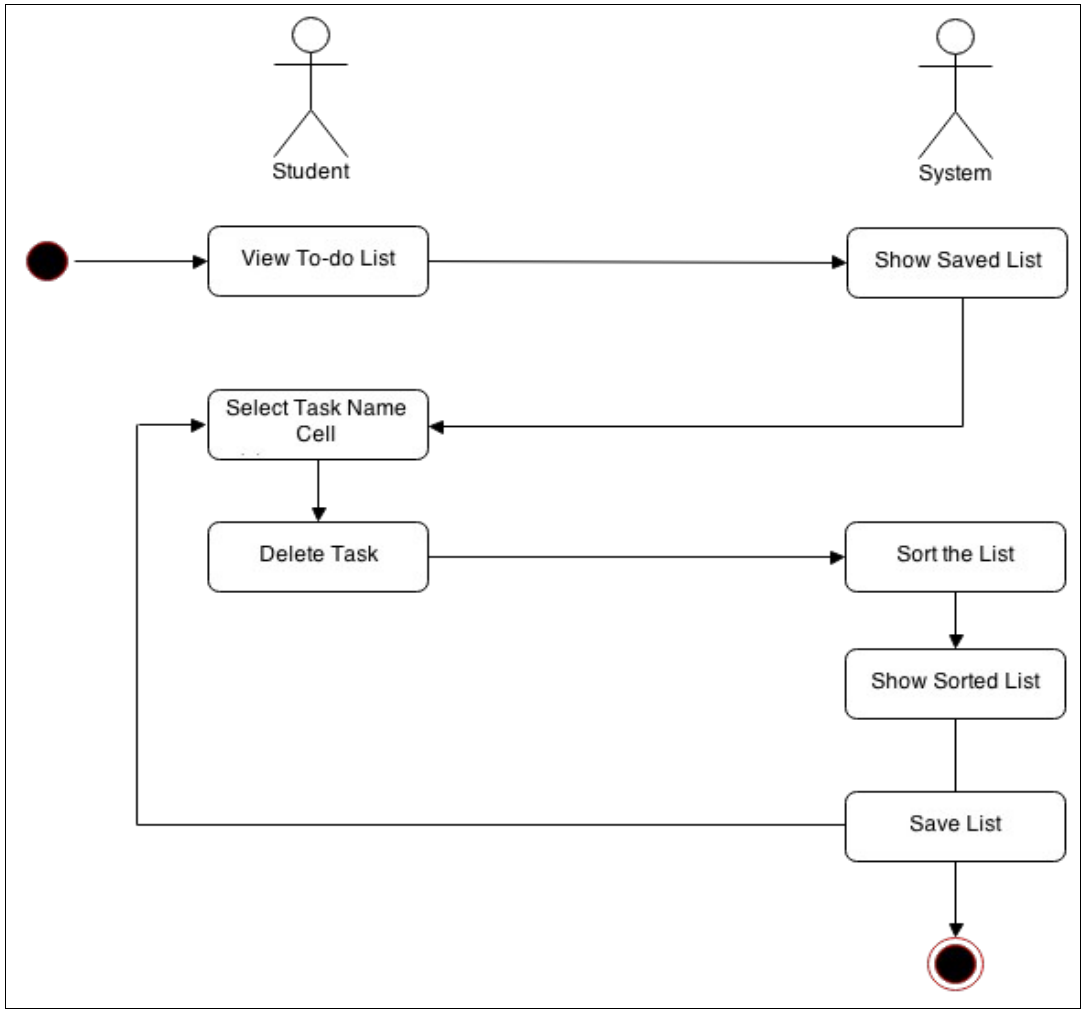
Description: The user can choose to remove a task from the list. A task is deleted by deleting the cell containing the task name. This removes the entire row from the list. If a task in the list is added to the pie, once it is done, the task is automatically deleted from the list.

Preconditions: 1.0 View To-do List, the list is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student deletes a task from list	1. Student presses the View To-do List button. 2. The application shows the saved list. 3. User clicks on a task name cell from the list. 4. The entry is deleted from the list. 5. The application sorts the list. 6. The list is sorted. 7. The list is saved.

Activity Diagram of the Flow of Events:



Postcondition: List is still sorted (either by name or by date) after the new task is added.

Relationships: 1.0 View To-do List, 2.0 Sort List

Special Requirements: NONE

Use-Case Name: 6.0 Set Time

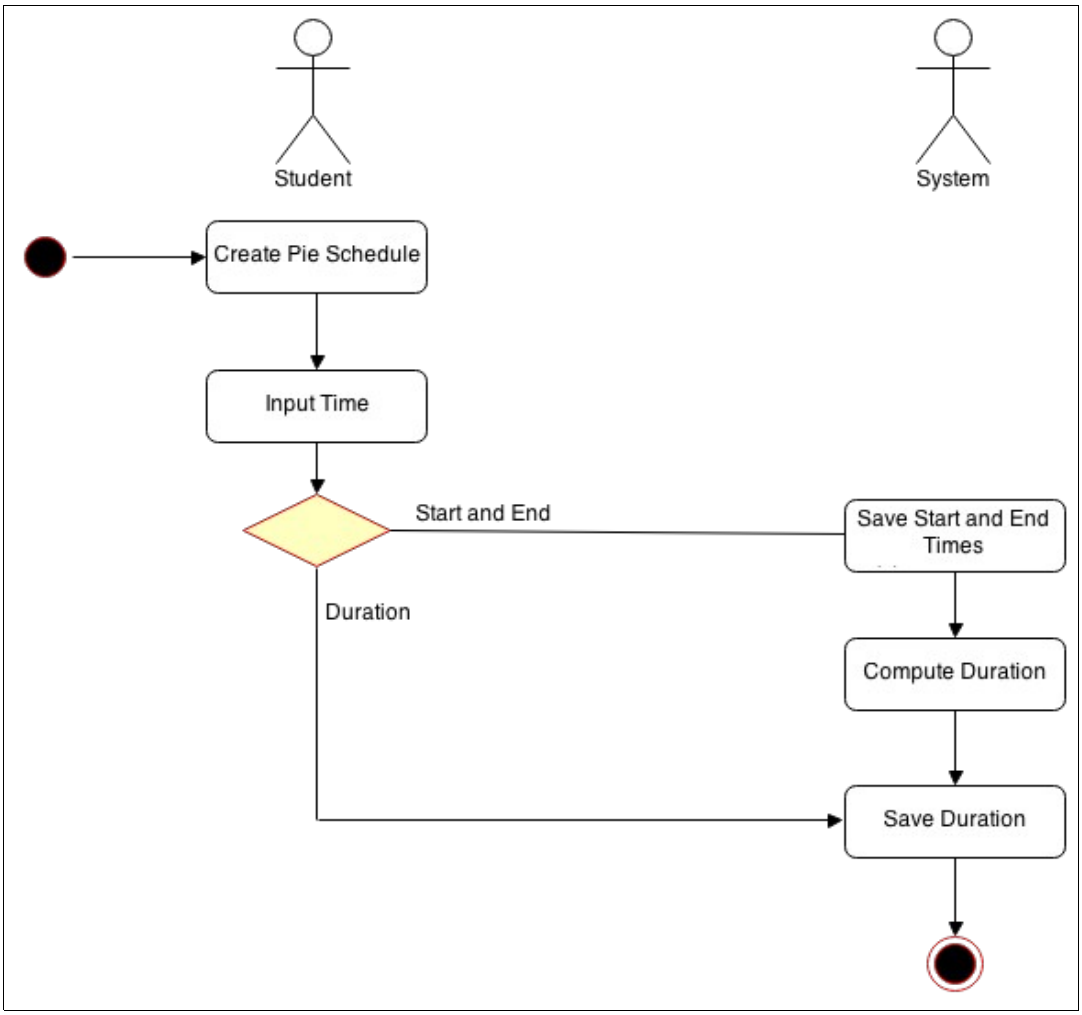
Description: The time frame of the pie is set by the user. The input can be either the duration or the start and end times expected to accomplish all the tasks in the schedule. Once the user has started the timer of the pie, the application will run during the given time.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student sets the time of the new pie	1. Student creates a pie schedule. Pie must not be empty. 2. User clicks the Set Time button. 3. User inputs the time frame for the entire pie schedule. The input is either the duration or the start and end times. 4. If the input time is the start and end times the application computes for duration. 5. Save the duration and the start and end times (if available).

Activity Diagram of the Flow of Events:



Postcondition: NONE

Relationships: 7.0 Create Pie Schedule

Special Requirements: NONE

Use-Case Name: 7.0 Create Pie Schedule

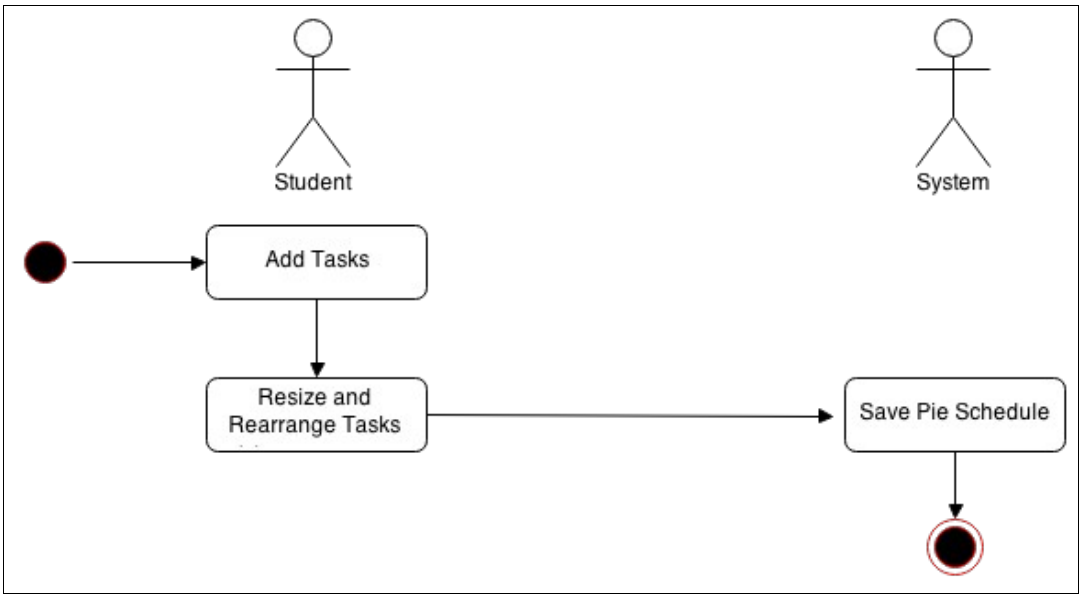
Description: This is the main operation of the application. The pie schedule is manually created by the user by adding, arranging, allotting time, and deleting. This is linked to the To-do List.

Preconditions: The pie schedule is empty.

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student creates pie schedule	1. Student adds tasks to the empty pie. 2. Student resizes and rearranges the tasks. 3. The application saves the generated pie schedule.

Activity Diagram of the Flow of Events:



Postcondition: A pie schedule is created.

Relationships: 6.0 Set Time, 7.1 Add Task/Slice, 7.2 Arrange Task/Slice, 7.3 Resize Slices, 7.4 Delete Task/Slice

Special Requirements: NONE

Use-Case Name: 7.1 Add Task/Slice

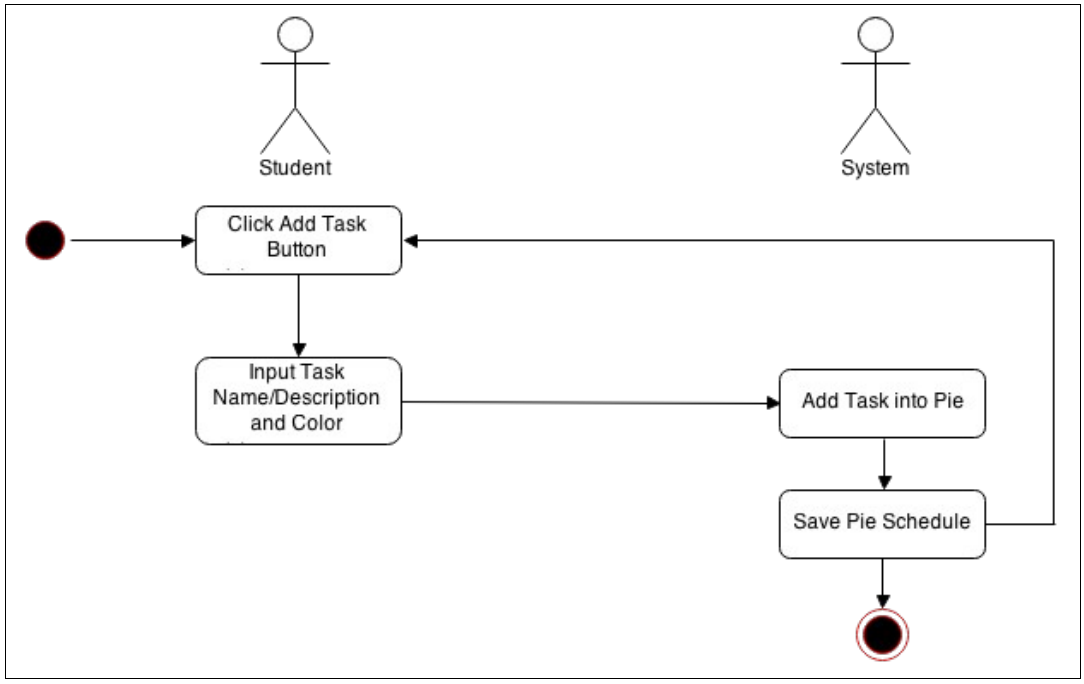
Description: A task is added by inputting its name and choosing a color. It is added to the pie schedule as a slice with the chosen color. The user can add up to 18 tasks only. The user can also select tasks from the To-do list. The list will show only the tasks scheduled on the same date the user creates the pie.

Preconditions: NONE

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student adds a task	1. Student clicks Add Task button. 2. Student inputs task name/description and chooses color. Input can be selected from the To-do List. 3. A slice, with the chosen color, representing the task is added into the pie. 4 The pie is saved.

Activity Diagram of the Flow of Events:



Postcondition: The pie is modified.

Relationships: 7.0 Create Pie Schedule

Special Requirements: Can add up to 18 tasks only. Cannot be done after resizing.

Use-Case Name: 7.2 Arrange Task/Slice

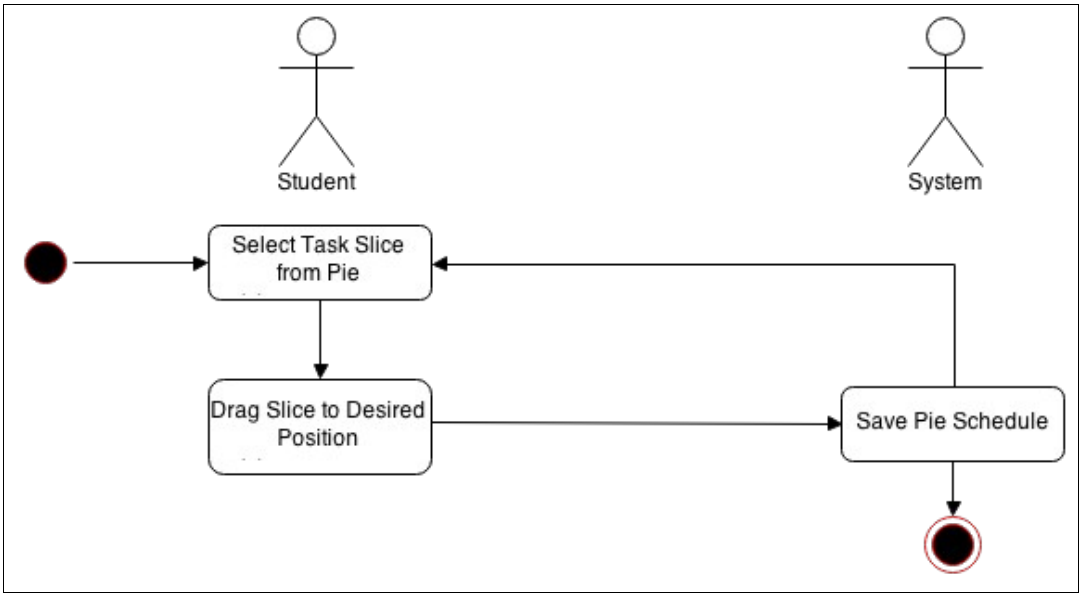
Description: The tasks can be easily rearranged. The slices are moved by dragging them clockwise or counterclockwise along the pie. This can be done anytime when creating the pie schedule.

Preconditions: 7.0 Create Pie Schedule, the pie schedule is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student moves a task	1. Student presses down on the task's slice. 2. Student drags the slice to the desired position. 3. The pie is saved.

Activity Diagram of the Flow of Events:



Postcondition: The pie is modified.

Relationships: 7.0 Create Pie Schedule, 11.0 Edit Pie Schedule

Special Requirements: Can add up to 18 tasks only.

Use-Case Name: 7.3 Resize Slices

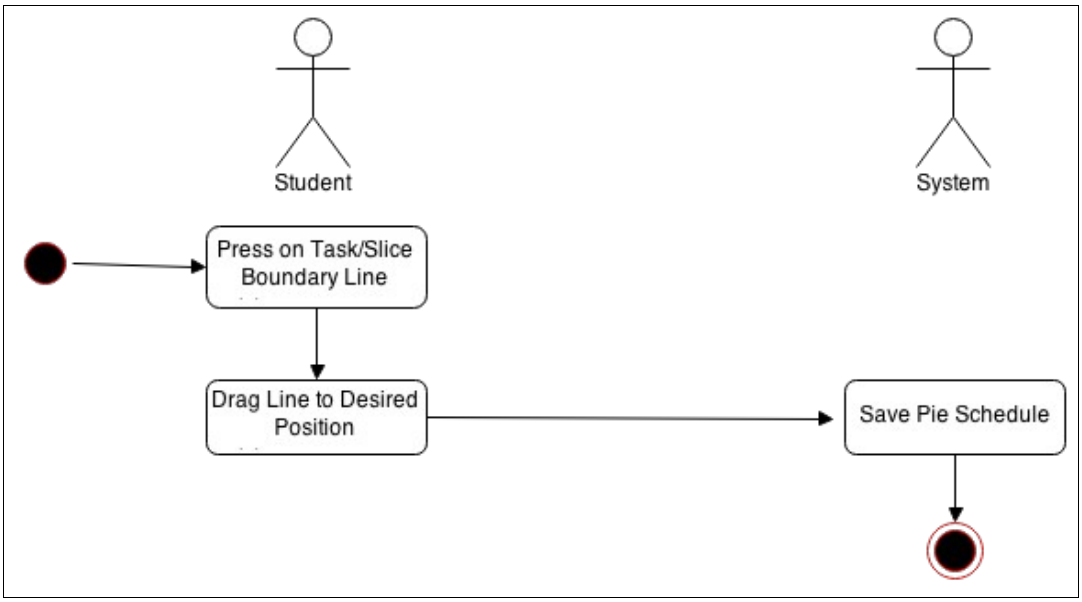
Description: Only after all the tasks have been added can the user change their sizes. Once resizing has occurred, adding a task will be impossible. Slices are resized by dragging their boundary lines. The size of a slice is used to compute for its corresponding task's allotted time. The ratio of a task's duration to the time frame of the whole pie is equal to the ratio of the slice's area to the pie's.

Preconditions: 7.0 Create Pie Schedule, the pie schedule is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student resizes a slice of pie	1. Student presses down on either of the boundary lines of the task's slice. 2. Student drags the line to the desired position. 3. The pie is saved.

Activity Diagram of the Flow of Events:



Postcondition: The pie is modified.

Relationships: 7.0 Create Pie Schedule, 11.0 Edit Pie Schedule

Special Requirements: A slice can be resized to at least 10 degrees.

Use-Case Name: 7.4 Delete Task/Slice

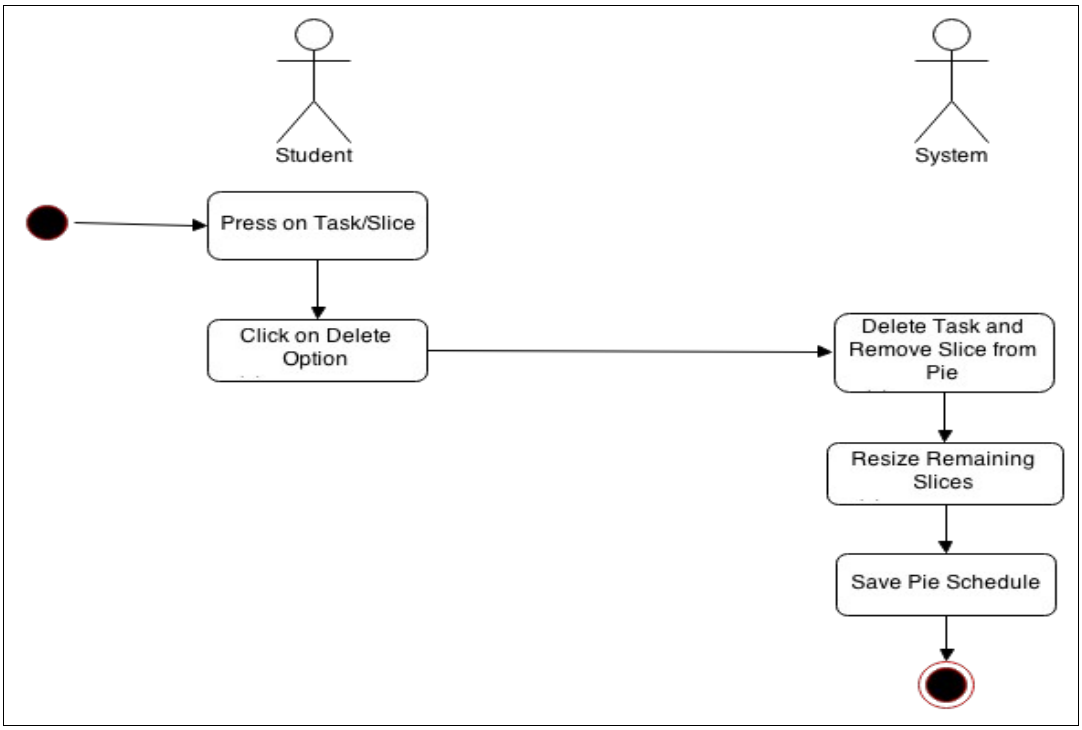
Description: Tasks can be deleted one at a time. Since there is no edit task function, the alternative is to delete and add. When a task is deleted, the pie adjusts such that the area previously occupied by the deleted slice is distributed equally among the slices.

Preconditions: 7.0 Create Pie Schedule, the pie schedule is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student deletes a slice of pie	1. Student presses on the task's slice. 2. Student clicks delete option. 3. Task is deleted and the slice is removed from the pie. 4. The slices are resized wherein the area of the deleted slice is distributed equally among the other slices. 5. The pie is saved.

Activity Diagram of the Flow of Events:



Postcondition: The pie is modified.

Relationships: 7.0 Create Pie Schedule

Special Requirements: NONE

Use-Case Name: 8.0 Check Task

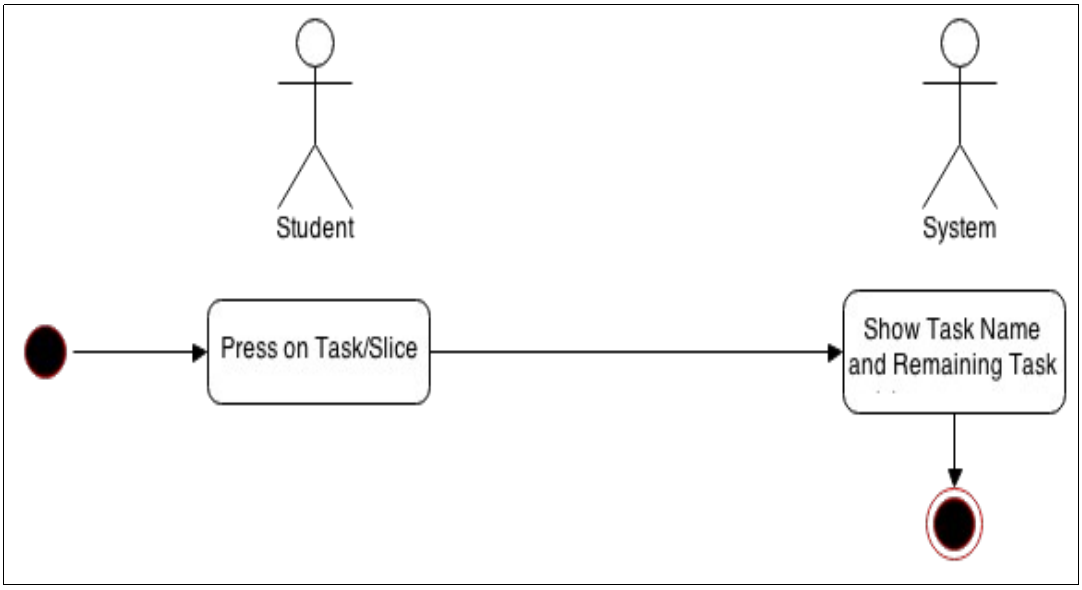
Description: The user can view the detail of a slice by clicking it. This will show the task and its remaining allotted time. This can be done anytime.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student views a task in the pie	1. Student presses on the task's slice. 2. The application shows the name of the task and the remaining time allotted to it.

Activity Diagram of the Flow of Events:



Postcondition: NONE

Relationships: 7.0 Create Pie Schedule

Special Requirements: NONE

Use-Case Name: 9.0 Reset

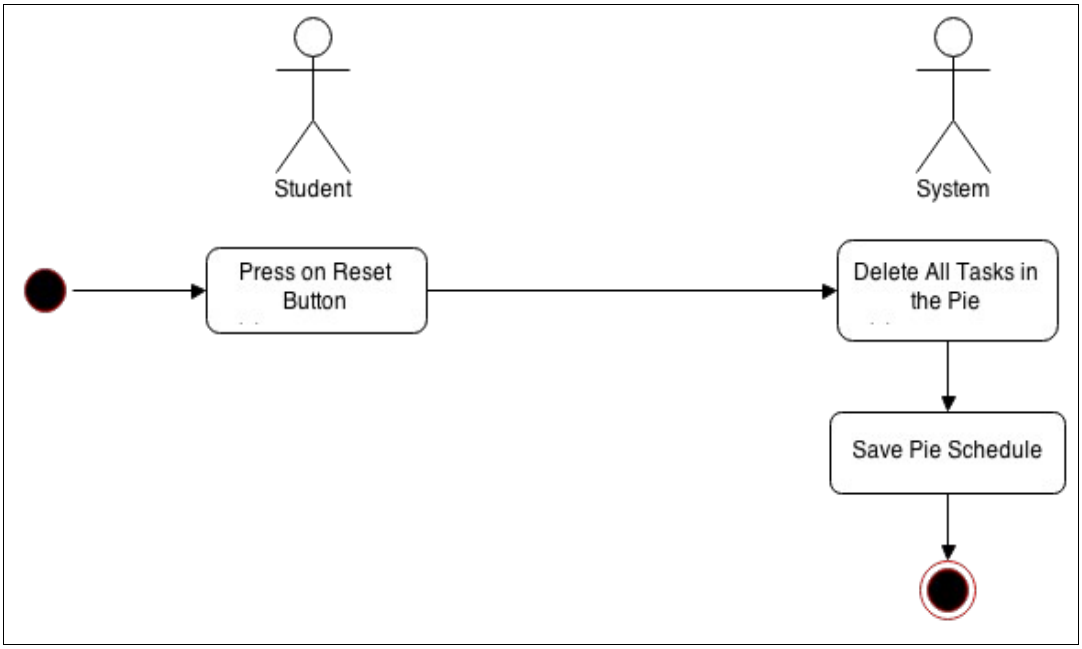
Description: This deletes all the tasks resulting to an empty circle. It is useful when the user wants to add tasks after resizing. This can be done anytime.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student resets the pie schedule	1. Student presses Reset button. 2. The application deletes all tasks in the pie, resulting into an empty pie. 3. The pie is saved.

Activity Diagram of the Flow of Events:



Postcondition: The pie is empty.

Relationships: 7.0 Create Pie Schedule

Special Requirements: NONE

Use-Case Name: 10.0 Start

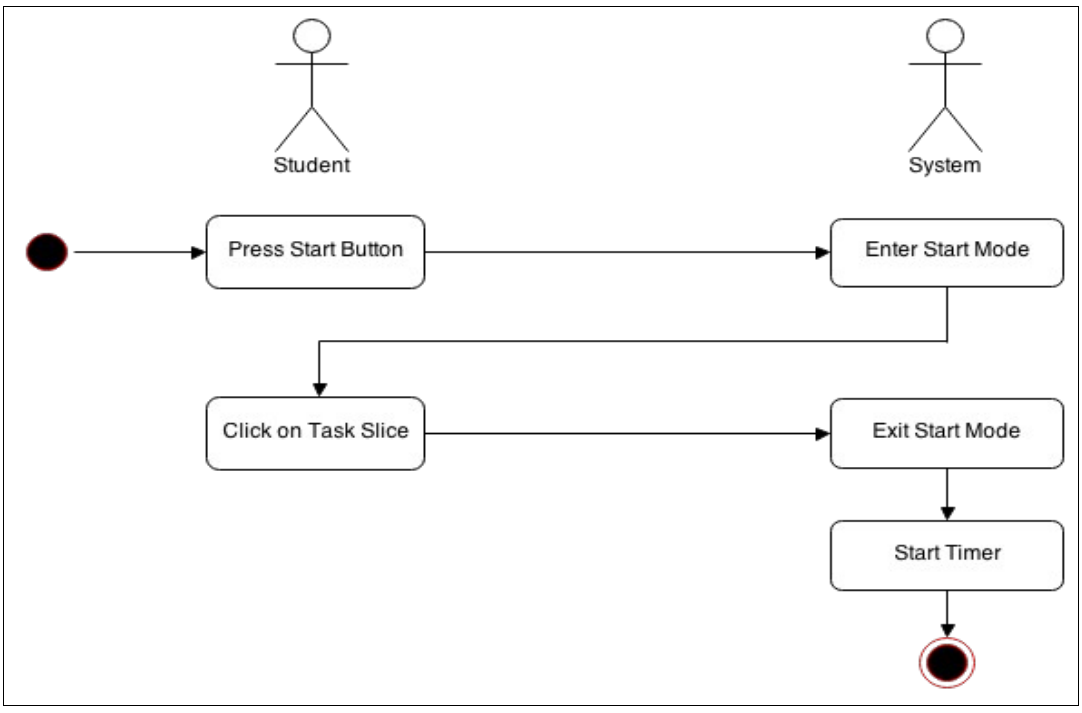
Description: This starts the timer of the pie. Also, the user picks from which task to start. Once the start button is pressed, the pie schedule can only be resized and rearranged during pause.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student starts the pie	1. Student presses Start button. 2. When Start is clicked, student must press on a slice. While in start mode, the user can only click on the pie. Timer won't start unless a task is selected. 3. After a task has been chosen, it will exit start mode and the timer will start. The application goes through the pie in a clockwise direction.

Activity Diagram of the Flow of Events:



Postcondition: The pie timer starts running.

Relationships: 7.0 Create Pie Schedule

Special Requirements: NONE

Use-Case Name: 11.0 Respond to Alarm

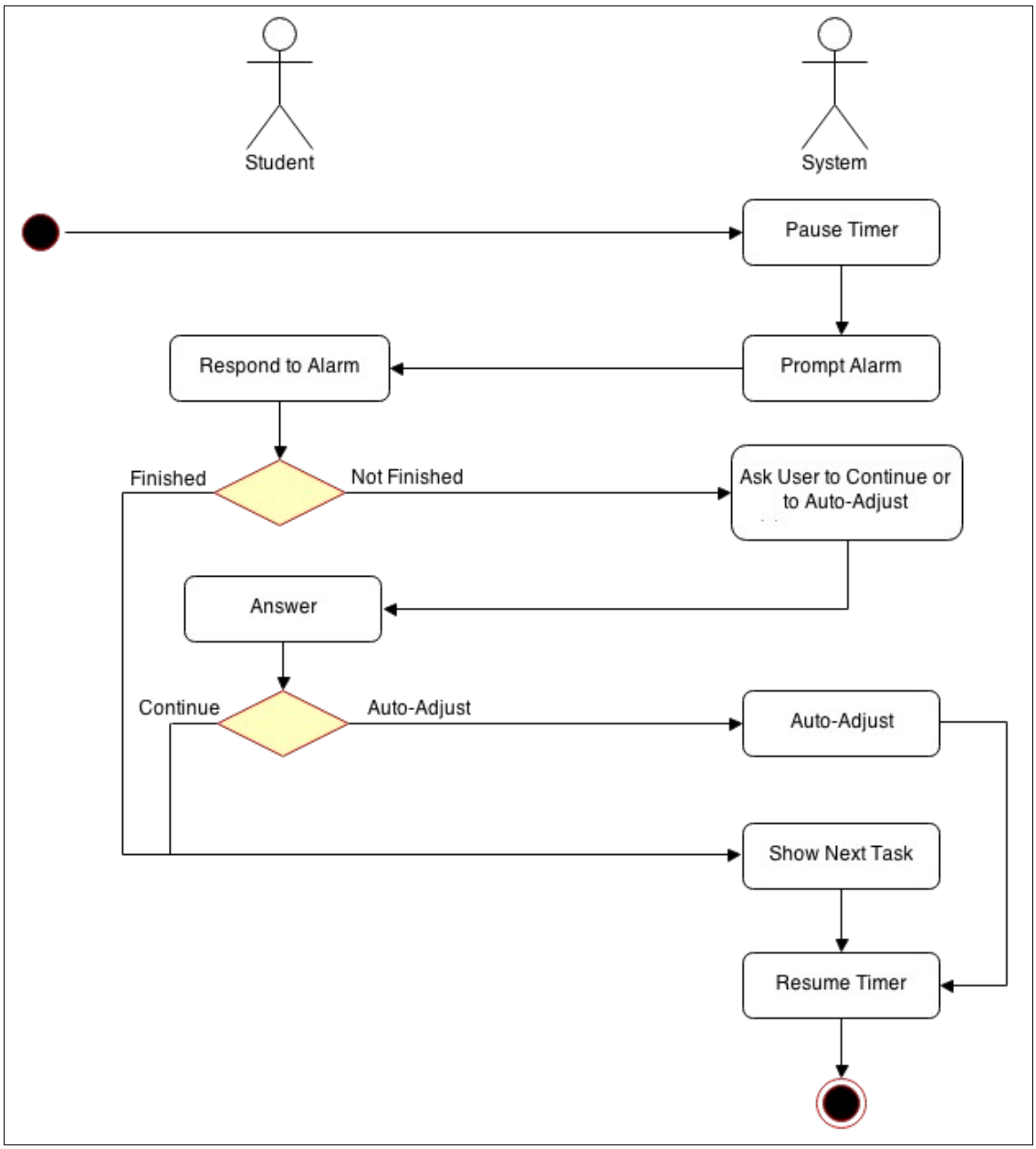
Description: The application will alarm once for a few seconds whenever a task runs out of time. At the same time, it will ask the user if the task is finished or not. If yes, the next task will be shown and the timer will resume. Otherwise, the user will have the option to auto-adjust.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty,10.0 Start, the pie timer is running

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student responds to an alarm	1. The applications pauses the timer when a task runs out of time. 2. It prompts an alarm. 3. Student replies if the task is done or not. 4. The application resumes running the timer of the pie.
Scenario 2 Student responds to an alarm when the task is not finished	1. The applications pauses the timer when a task runs out of time. 2. It prompts an alarm. 3. Student replies that the task is not finished. 4. Student might opt to extend the time by using auto-adjust, otherwise the application will show the next task. 5. The application resumes running the timer of the pie.
Scenario 3 Student responds to an alarm when the task is finished	1. The applications pauses the timer when a task runs out of time. 2. It prompts an alarm. 3.. Student replies that the task is finished. 4. The application shows the next task. 6. The application resumes running the timer of the pie.

Activity Diagram of the Flow of Events:



Postcondition: The pie may be modified.

Relationships: 7.0 Create Pie Schedule, 10.0 Start

Special Requirements: NONE

Use-Case Name: 12.0 Adjust Time

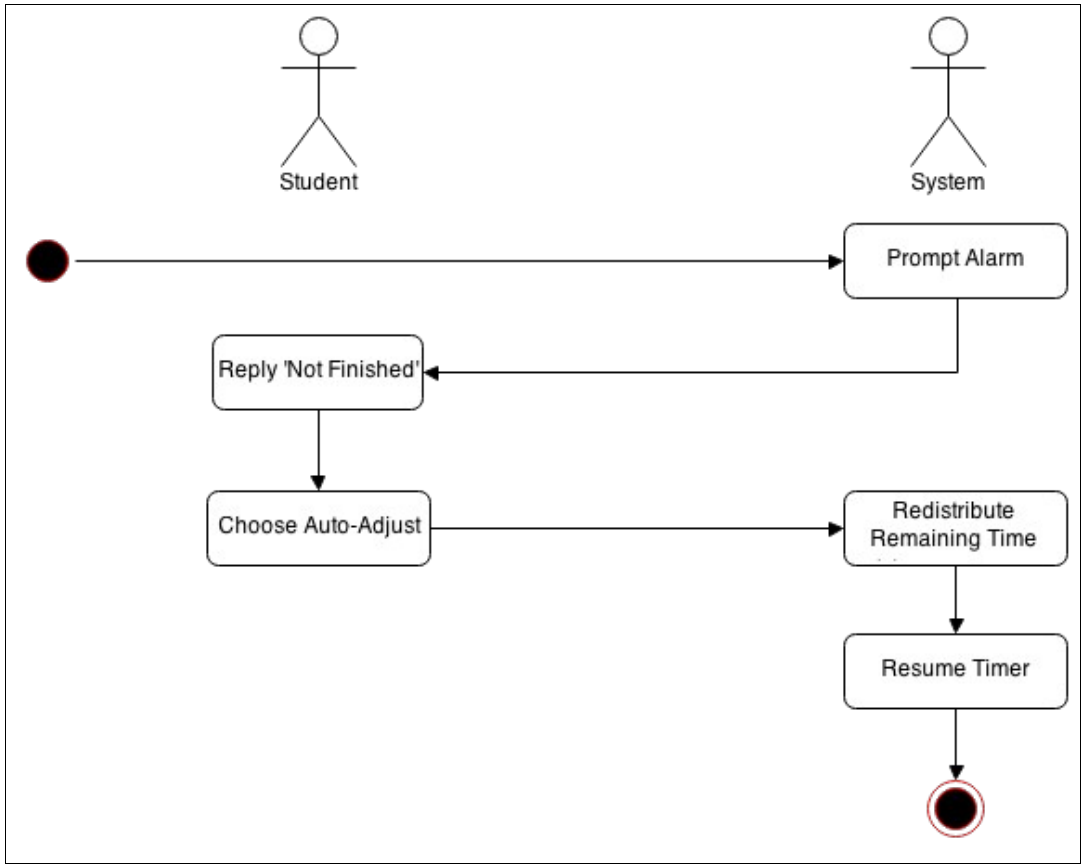
Description: When the time allotted for a task runs out and the user is not yet done, he/she has the option to let the application redistribute the remaining time among the unfinished tasks. If the user chooses to do so, the Adjust Time will recalculate the durations.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty, 11.0 Respond to Alarm

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student uses auto-adjust	1. The application prompts an alarm. 2. Student replies that the task is not finished. 3. Student opt to extend the time by using auto-adjust. 4. The application redistributes the remaining time among the unfinished tasks. 5. The application resumes running the timer of the pie.

Activity Diagram of the Flow of Events:



Postcondition: The pie is modified.

Relationships: 7.0 Create Pie Schedule, 11.0 Respond to Alarm, 14.0 Pause/Unpause

Special Requirements: NONE

Use-Case Name: 13.0 Pause/Unpause

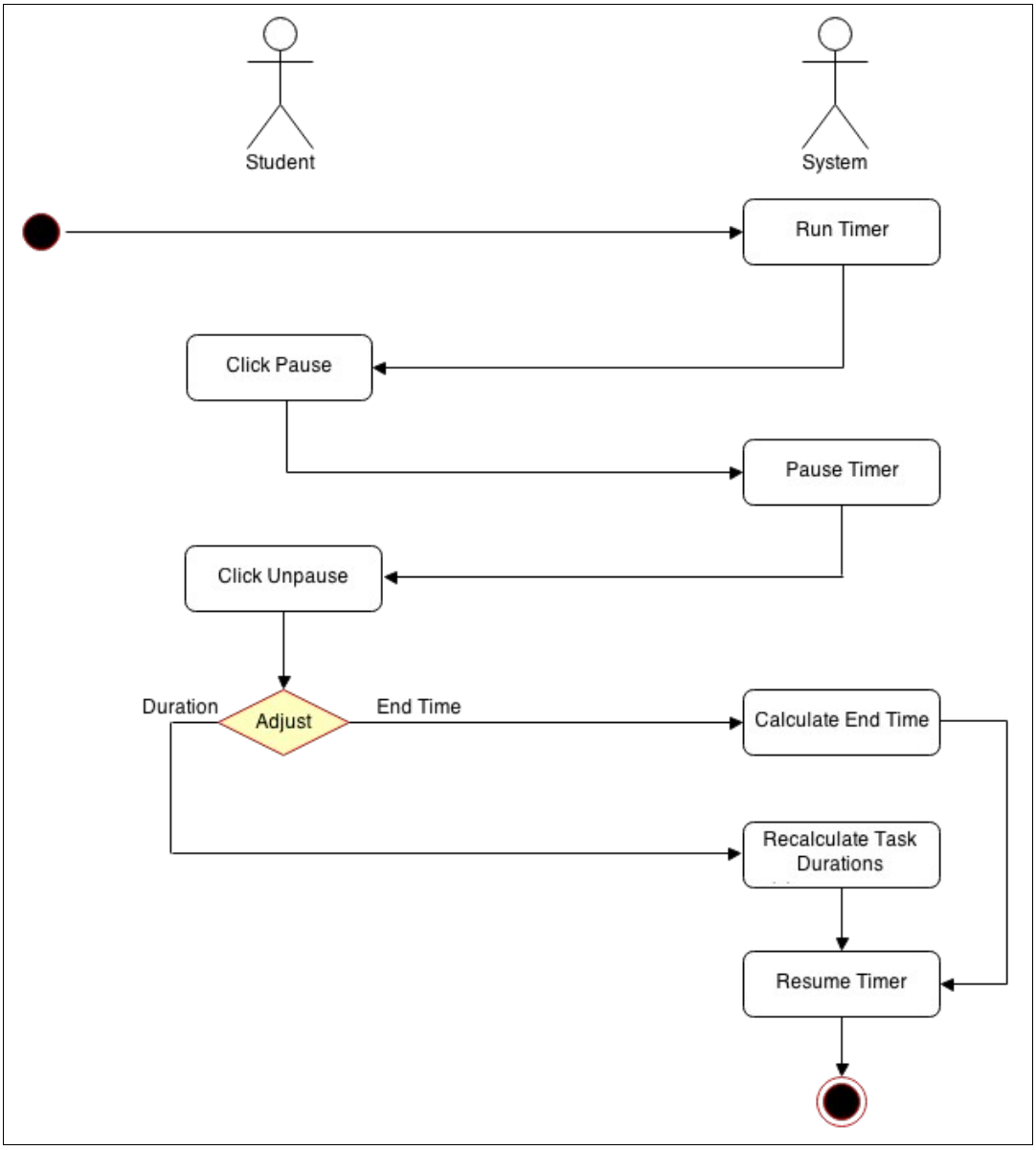
Description: The Pause Timer function enables the user to pause the timer for the schedule. Its purpose is to make it convenient for the user when necessary to tend to things not included in the pie. The Unpause Timer function resumes the timer. In setting the time, if the input was start and end times instead of duration, the user will be given two options: move end time or recalculate the durations for the unfinished tasks. Selecting move end time changes the end time by adding the duration of the pause. While choosing recalculate the durations results to Adjust Time.

Preconditions: 7.0 Create Pie Schedule, the pie is not empty, Timer is running

Flow of Events:

Scenario Name	***Description***
Scenario 1 (Basic Flow) Student pauses/unpauses timer	1. Timer is running. 2. Student clicks on the Pause button. 3. Timer pauses. 4. Student clicks on the Unpause button. 5. Student chooses between adjust the end time or durations. 6. The application calculates which is chosen. 7. Timer resumes.
Scenario 2 Student pauses timer	1. Timer is running. 2. Student clicks on the Pause button. 3. Timer pauses.
Scenario 3 Student unpauses timer, chooses to move end time	1. Timer is running. 2. Student clicks on the Pause button. 3. Timer pauses. 4. Student clicks on the Unpause button. 5. Student chooses to adjust the end time. 6. The application calculates the new end time for the entire pie schedule. 7. Timer resumes.
Scenario 4 Student unpauses timer, chooses to adjust durations	1. Timer is running. 2. Student clicks on the Pause button. 3. Timer pauses. 4. Student clicks on the Unpause button. 5. Student chooses to adjust durations. 6. The application calculates the new durations for the unfinished tasks. 7. Timer resumes.

Activity Diagram of the Flow of Events:



Postcondition: The pie timer stops or the pie timer resumes.

Relationships: 7.0 Create Pie Schedule, 10.0 Start

Special Requirements: NONE