Schedule Optimizer
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:
Cavan, Antonio D.
Ramos, John Matthew G.
See, Engelberg Jeremy T.

In partial fulfillment of academic requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2019-2020

Unique Reference:

The documents are stored in the https://github.com/DarkLuminosity/Schedule-Optimizer/tree/master/02-Requirements%20Engineering/Project%20Deliverables referenced with ScheduleOptimizer-Use Case 2-Indicate Subject.pdf

Document Purpose:

The purpose of this document is to identify and present the different scenarios that may occur within the use case presented.

Target Audience:

The target audience would be University of the Philippines-Diliman undergraduate students.

Revision Control:

Revision Date	*Person Responsible*	*Version Number*	*Modification*
09/19/19	Engelberg See	1.0	Prepare Initial Document.
09/21/19	Engelberg See	2.0	Added Flow of Events and Activity Diagram
09/23/19	Antonio Cavan	3.0	Added ER Diagram and Relationships

***Use-Case Name*:** Indicate Subject

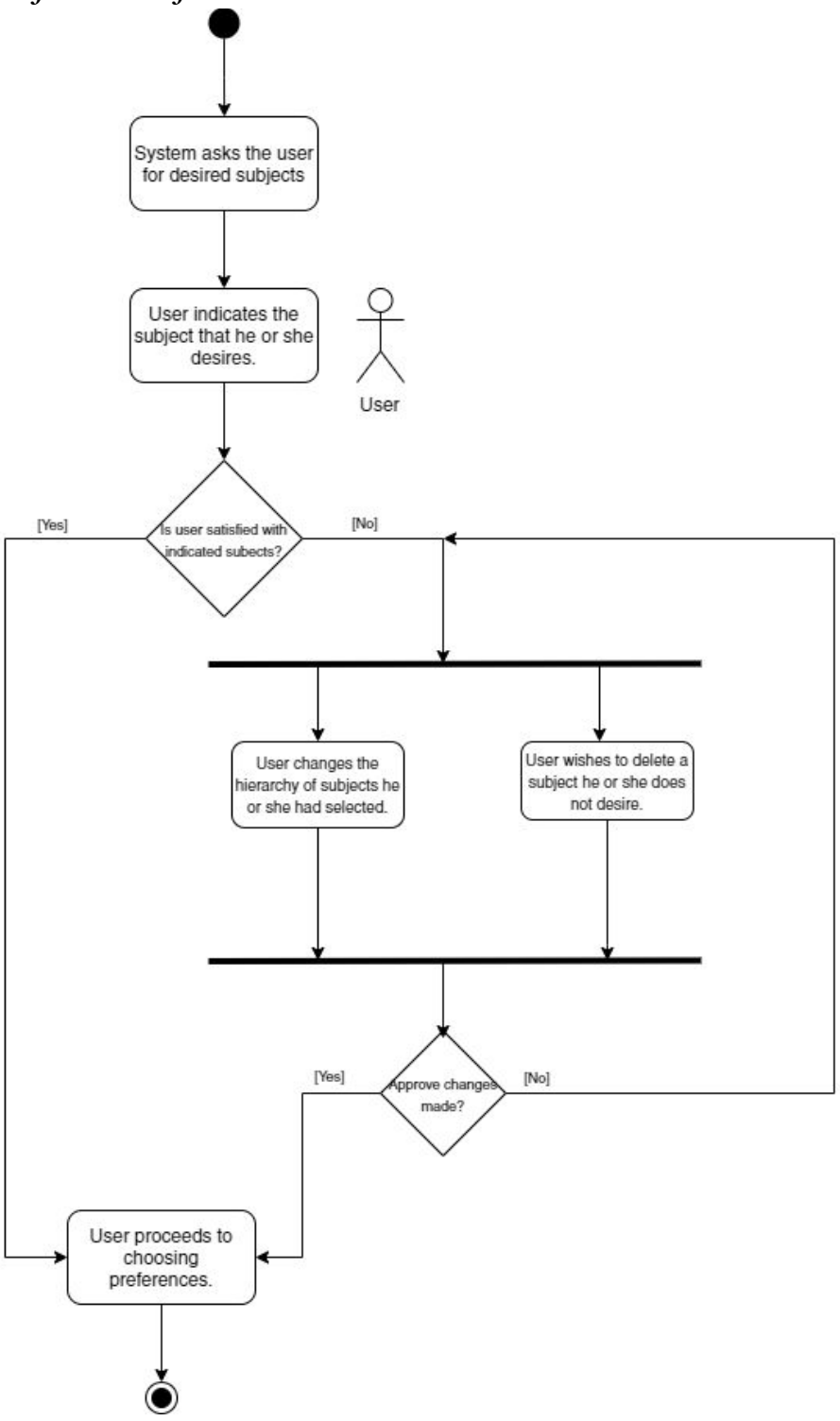
Description: Indicate Subject is a use case in which the user would select, edit or delete the subjects that would be used for the succeeding use cases. It would contain a list of subjects being offered in UP Diliman wherein he or she would input the different subjects that he or she would want. The user could also modify the hierarchy of subjects that they have chosen to prioritize which subjects are the main points of their schedule.

Preconditions: The only precondition to this use case is that the user should already have the subjects he or she will be taking in mind.

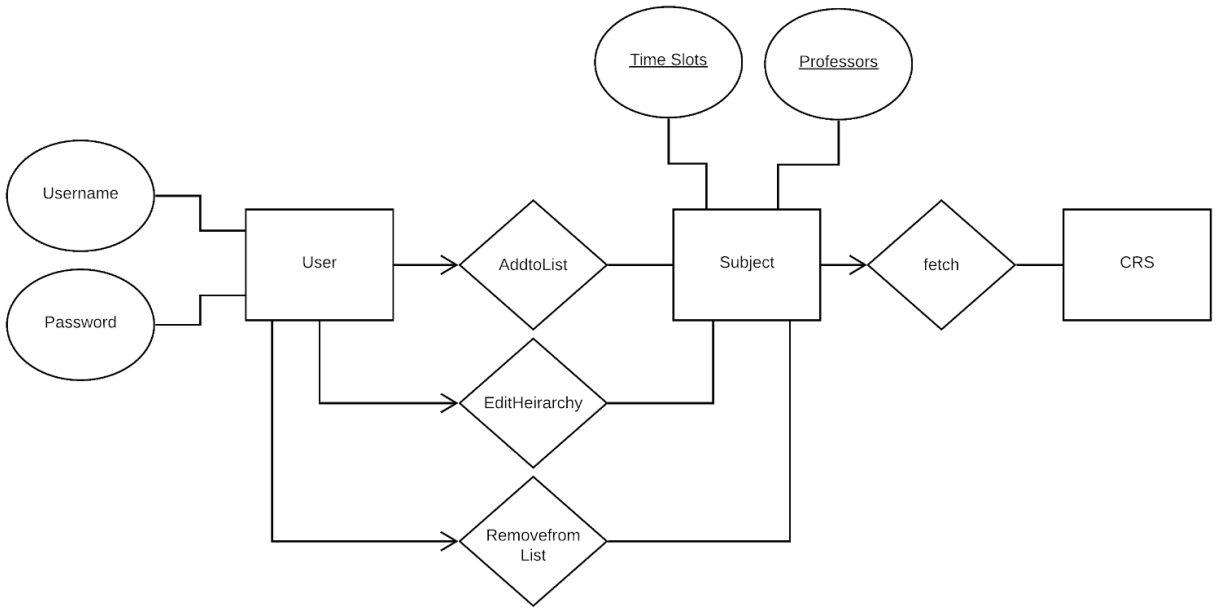
Flow of Events:

Scenario Name	*Description*
Scenario 1 (Basic Flow) User wishes to indicate the subjects that he or she would want to see on his or her schedule	1. The system asks a list of subjects from the user. 2. User chooses the subjects that are relevant to him/her. 3. Once done, the user would then proceed to the next step to indicate his or her preferences.
Scenario 2 (Alternative Flow) User wishes to edit the hierarchy of the subjects that they had selected.	1. The system would provide an option to edit the existing subjects that had been selected by the user. 2. The user would then indicate the hierarchy of the importance of each subject that he or she had selected. 3. The system would then show a list of the updated hierarchy of the subjects that the user had selected. 4. A pop-up box will appear if the user had approved of the changes made. 5. If the user is satisfied with the hierarchy of the subjects indicated, he or she would then proceed to the next step to indicate his or her preferences.
Scenario 3 (Alternative Flow) User wishes to delete a subject that they had misclicked.	1. The system would provide an option to delete the existing subjects that had been selected by the user. 2. The user would delete the subjects that he or she does not want to be on his or her optimized schedule. 3. The system would then show a list of the updated hierarchy of the subjects that the user had selected. 4. A pop-up box will appear if the user had finalized the changes made to the list of subjects. 5. If the user is satisfied with the hierarchy of the subjects indicated, he or she would then proceed to the next step to indicate his or her preferences.

Activity Diagram of the Flow of Events:



Other Diagram: Entity Relationship Diagram



Postcondition: NONE

Relationships:

AddtoList: strong-entity relationship between User and Subject
- adds a subject to the list of subjects preferred by the user

EditHeirarchy: strong-entity relationship between User and Subject
- changes the heirarchy of a subject

RemovefromList: strong-entity relationship between User and Subject
- removes a subject from the list

fetch: strong-entity relationship between Subject and CRS
- gets the subject from the CRS Database

Special Requirements:

NONE