Software Design Document (SDD)-UPDATED

1. Introduction

It has been mentioned during the exam that our Design Document was highly low-level and so was too oriented towards the implementation part instead of the general design procedure. Actually this part was totally missing from the document as we have thought that the section titled “General Architecture” dealt with that (but it also was implementation-oriented as it discussed the front-end implementation of our application on Bubble instead of defining a general description of the different components that make up our app, in a framework-independent fashion). Therefore, we do that now as an addition to the SDD:

2. General Design of PoliSAM

In this paragraph we define the different components (modules) of our PoliSAM system while making reference to their functionalities as well:

1. ***Signup Module:***

This component manages the registration of new users that are asked to enter the following inputs:

- *“Username” (such that it is unused by previously registered students, so that every user has his own unique username)*
- *“Email” (such that the email is a Polimi email, in order to make sure that the student is enrolled at the university itself)*
- *“Password”*

The output of this module is the addition of the new student into the current database (ofcourse under the condition of correct inputs), and the sending of a confirmation email when the registration has been successful. This component should be present on the first interface the user comes across.

2. ***Login Module:***

This component enables registered users to use the main functionalities of the app, after inputting the following info:

- *"Email"*
- *"Password"*

This module is also expected to be placed on the first interface, and it takes the registered student to a new second interface, which contains all of the components that will be discussed from now on.

3. ***Help Module:***

This component provides the logged-in user with introductory guidance concerning how he should interpret some of the other modules present on the same interface, which are not apparently straightforward to comprehend directly and easily.

4. ***Options Module:***

The *Options module* allows the user to see some of the attributes associated with the study-areas added to the database. These displayed attributes are:

- *The study-area's name*
- *The study-area's category (either library, classroom or common area)*
- *The study-area's capacity*

- *The number of checked-in occupants in this study-area (which is not necessarily the same as the number of students actually occupying the place as explained in the requirements specification document)*
- *The number of students who have included this study-area as one of their favorites*

In addition to that, this component should inform the student about the availability status of each study-area in a visually intuitive manner (such as with the use of three different colors for: **available**, **not available**, and **maybe available**):

- A place is surely **not available** when the number of checked-in students is equal to its capacity.
- A place is **maybe available** whenever it is a classroom and there are no checked-in people (thus indicating the possibility of an ongoing lecture at the current timing)
- A place is **available** otherwise.

Finally, the user should be able to choose one of the options before performing some of the further operations.

5. ***Filter & Display-All Module:***

The filter component searches for the study-areas that satisfy some assigned search criteria, and displays them with the help of the **Options Module** discussed above. The user can assign the following:

- "*Name of Study-Area*"
- "*Building of Study-Area*"
- "*Category of Study-Area*"
- "*Campus of Study-Area*"

Whereas the **Display-All** component displays all options again.

6. ***Interactive-Map Module:***

This component depends on the Google Maps API to show the student his current location, and enables him to search for all study-areas that are within “x” kilometers away from him, where “x” is assigned by the user himself. The map shows the student these places and allows him to select one of them from there immediately; this module also displays the filtered areas with the **Options Module** (they are given from closest to farthest)**,** discussed earlier.

7. ***Central Module:***

After the user selects one of the options shown to him by the **Options Module** or the **Interactive-Map Module,** all the information (given in the attributes of the **Study-Area** Class in the UML class diagram) corresponding to the chosen study-area is displayed with this component (note that the Schedule attribute is only provided whenever the place is of category classroom). Moreover, with this module, the user performs the essential operations defining most of our application’s specifications. In fact, the central component can itself be considered as containing several **sub-modules** (hence the name central module) such as:

- ***Submit Module:*** *Prompts the student to enter an approximation of the number of occupants (could be in the form of a percentage) he can see in the selected study-area, before he can perform the check-in procedure (assuming he is there and wants to check-in). The latest entered estimation is then displayed (along with the date/time of report) for all student users with the parent-module. Now the user is asked to either check-in or cancel the operation.*
- ***Check-in Module:*** *Checks the user in, and so consequently adds the total number of checked-in students by one (which is displayed with the Parent Module and the Options Module). Moreover, the student is notified that he is currently checked-in in the corresponding place (even if he selects another option from the study-area list to be displayed), and he is now asked to check-out whenever he decides so.*

- **Cancel Module:** *Cancels the check-in operation, and so takes the user back to the Submit Component.*
- **Check-out Module:** *Checks the user out from the place in which he was already checked-in (and so the check-in message disappears), and the total number of checked-in students is decreased by one. Moreover, the user is taken back to the Submit Component again.*
- ***Favorite Module:*** *Allows the user to set the selected study-area as one of his favorites, or remove it from his favorites if he had previously added it. In addition to that, this module gives the user the possibility to display (with the* **Options Module***) the study-areas that are among his favorites, or display the study-areas that are favored by other people. The display of these options is done in descending order of preference (mostly favored can be seen first until the least favored one which is seen last, and all other options that have never been favored by anyone, do not appear in the list)*
- ***Message-Posting Module:*** *This component enables the student user to write messages that correspond to the selected study-area, and when the message is written, the following information appears for all the other users checking this option:*
  - ➢ *Number of messages corresponding to the study-area*
  - ➢ *Username of the student who wrote the message*
  - ➢ *The time at which the student wrote the message*
  - ➢ *The message itself*

Moreover, the student is also able to delete any of his own messages without being able to delete the ones that do not correspond to him. Finally, this module clears all messages that were not written during the same day as a new message being posted currently.

8. ***Logout Module:***

This component doesn't require any user inputs, and it simply exits the student from the main interface and takes him back to the initial interface (the one which contains the **Signup and Login Modules**).

3. Improvement of Previous Diagrams

During the oral exam, there were also some comments made concerning the UML Class diagram (related to the fact that we have not used the exact same attributes in our diagram as those that were used later on during the implementation), and other comments also related to the Finite-State-Machine diagram not capturing the full picture of PoliSAM's integrated functionalities. In this section, we solve the aforementioned issues; notice that the UML diagrams are now placed in the design document instead of the Requirements document since we felt that it would be more suitable to add them as additional tools for describing the high-level design process (***the details of the attributes given below in the different classes of the UML-Class Diagram, will be explained in the implementation part***):

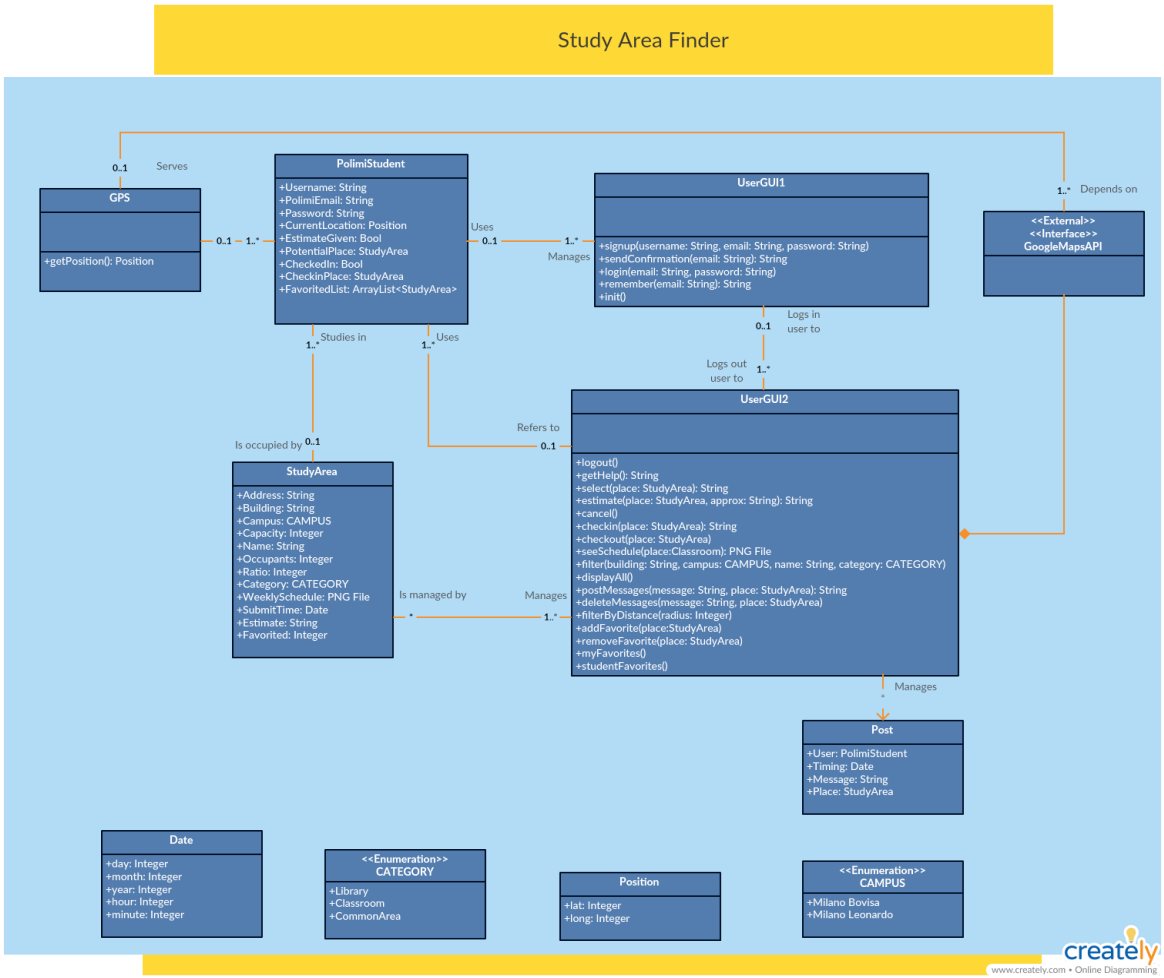


Fig. Updated UML Class Diagram

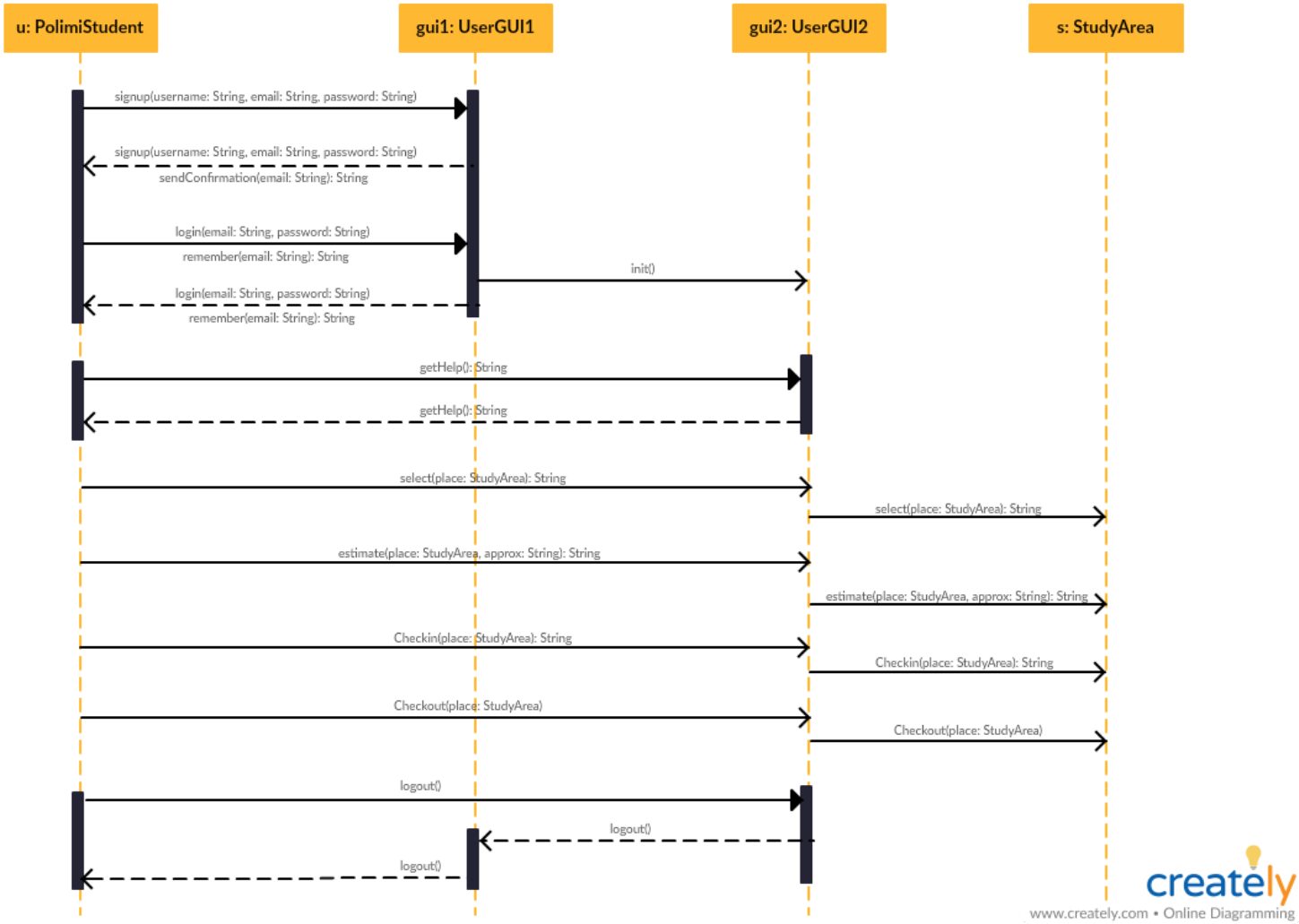


Fig. Updated UML Sequence Diagram

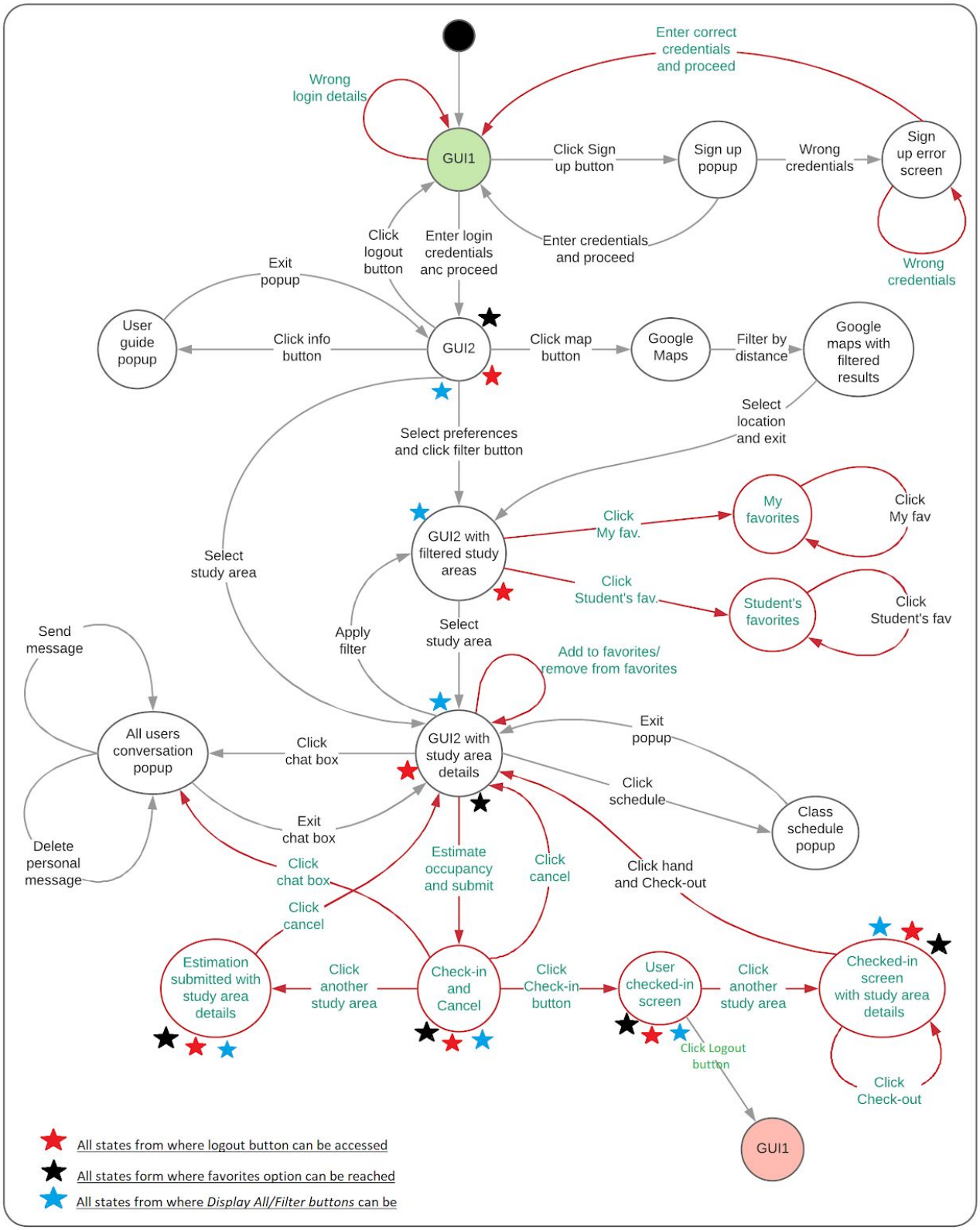


Fig. Updated Finite-State-Machine

4. Implementation

In this part, we do the same thing that was done in the similar section of the previous SDD document, but here we add the related information on the database and workflow changes that were made upon the addition of our two new features (providing a current estimate for the people occupying the study-area and providing the possibility to add places unto a Favorites List as well as perceive students' favorites sorted in order of mostly favored to least favored...)

Database Changes

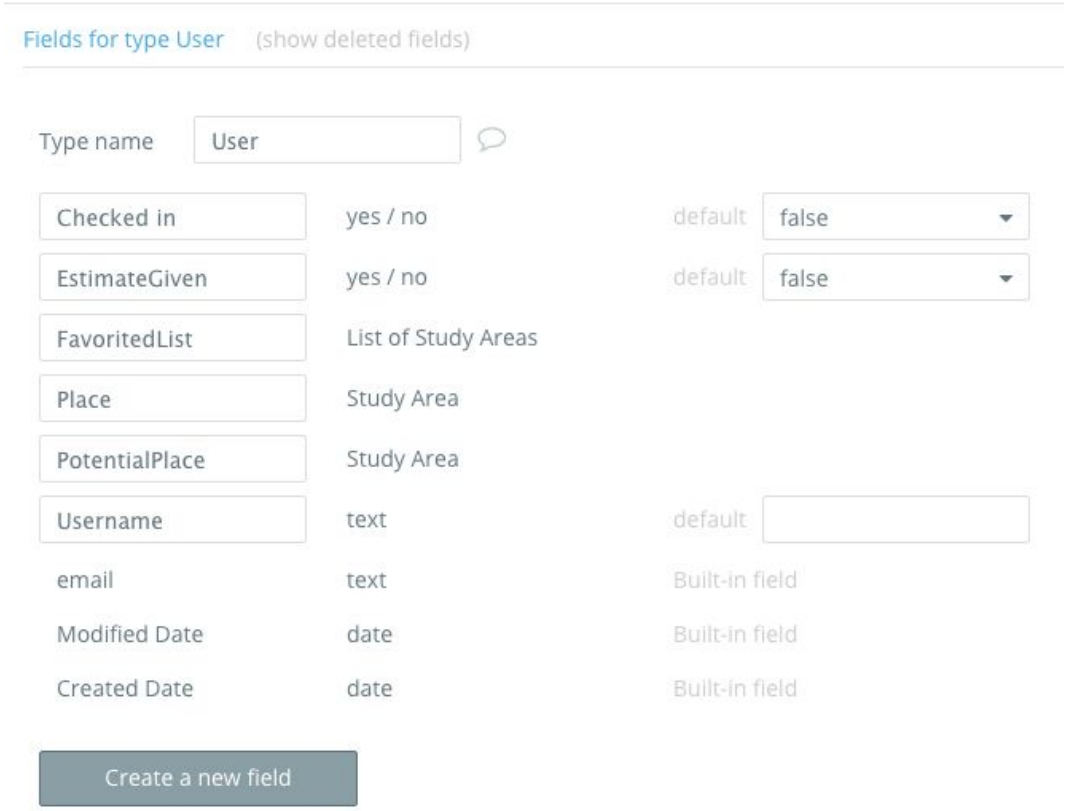
User

The changes that were made in this part were basically in adding three extra fields to the Type **User**:

- The first one was **EstimateGiven** which is of type **boolean**, and it is used to indicate if the user has given an estimate for the number of occupants in the selected place for check-in (ofcourse assuming he is already there), otherwise, the student is not allowed to check-in at all.

- The second addition was **PotentialPlace** which is of type **Study Area**, and it is nothing but the place for which the student has reported an estimate to (before wanting to check-in there)

- The last addition was **FavoritedList** which is of type **List of Study Areas** (this is the first time that we had to use a field of type: List...). It is used to store all the places that the **User** has set as part of his favorites.



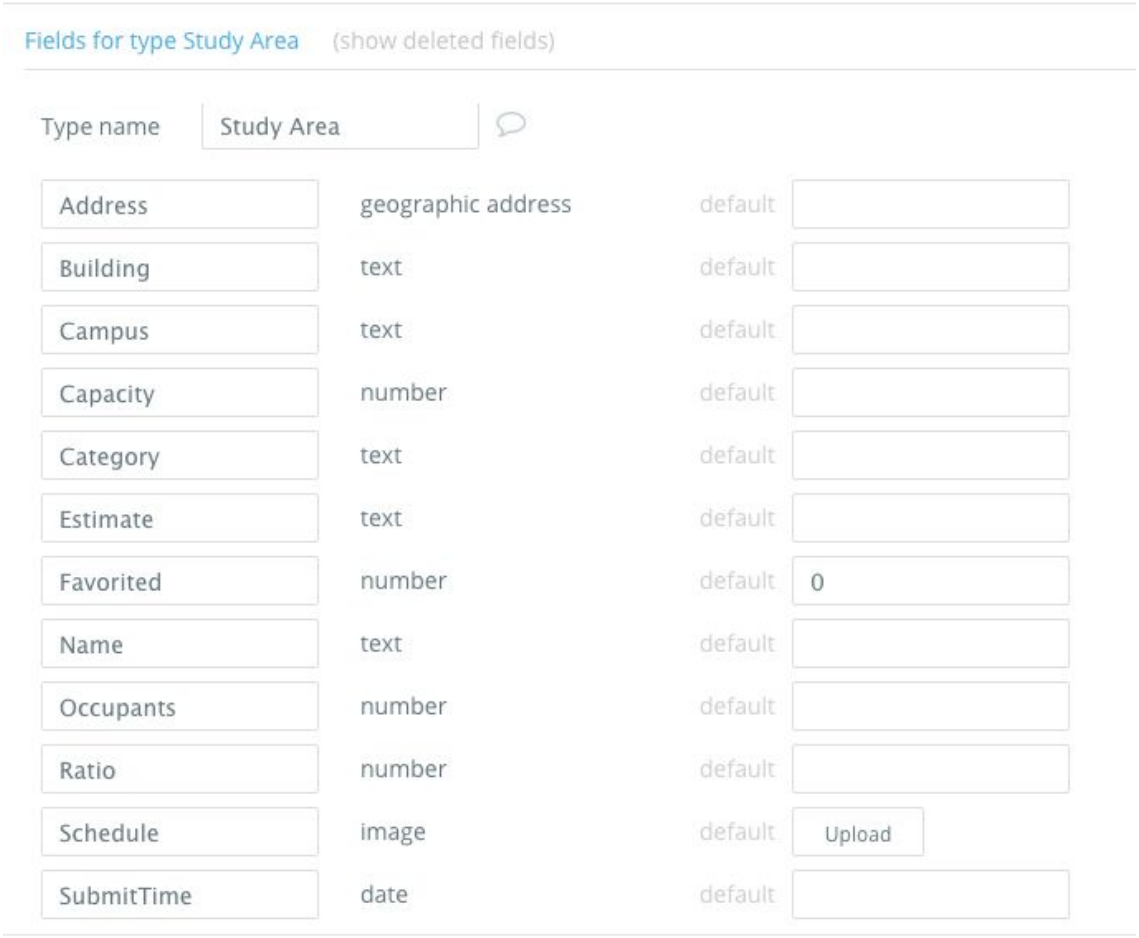
Study Area

The changes that were made in this part were basically in adding three extra fields to the Type **Study Area**:

- The first one was **Estimate** which is of type **text,** and it is used to store the last estimate (for the number of occupants) made by a student for a given place. It is of type text because we prompt the user to select a text option from a dropdown menu that indicates the percentage range of the occupancy.

- The second addition was the field **Favorited** which is of type **number** and it indicates what the number of students who have favored the given study-area is.

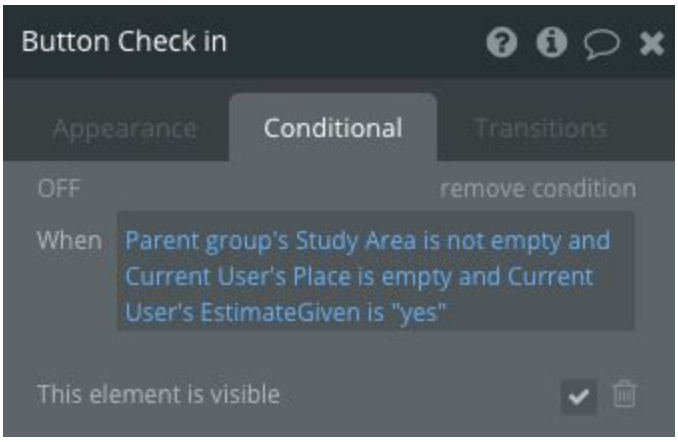
- The last one was **SubmitTime** which is of type **date,** and which simply stores the timing at which the last estimate for the given study-area was made.

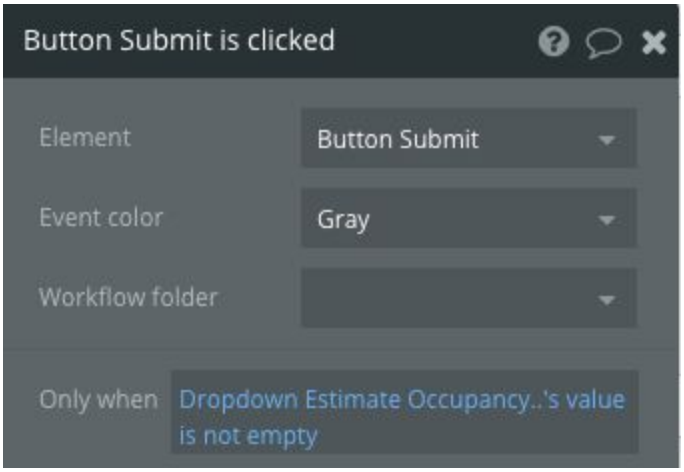


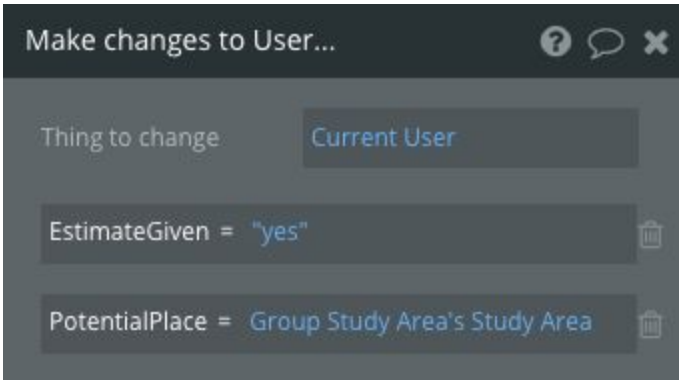
Workflow Changes

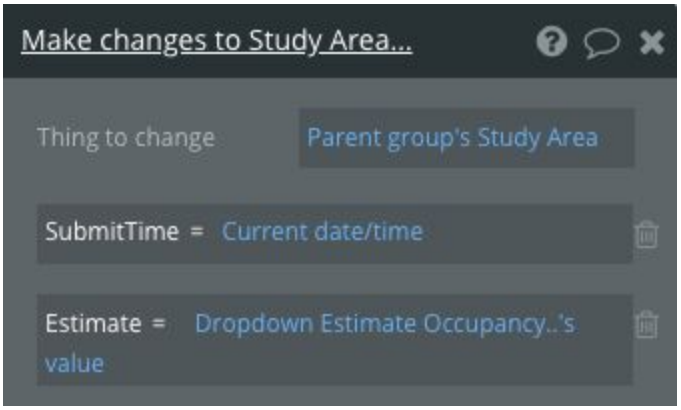
Reporting Estimate

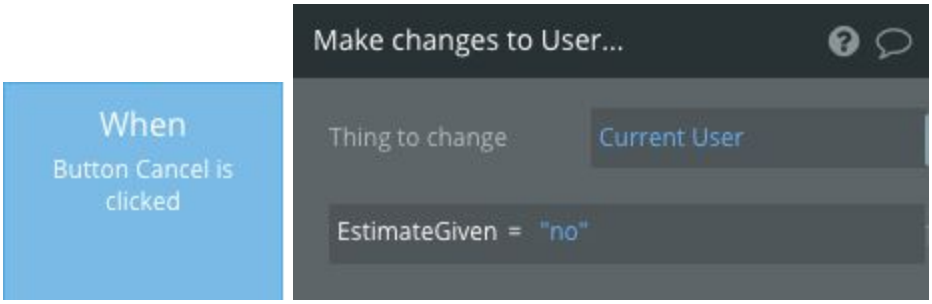
- After the user selects an option in which he would like to check-in, we want to prompt the user to first give an estimate of the occupancy by selecting the choice from a dropdown menu. He can either choose: *"Below 25%", "Between 25% and 50%", "Between 50% and 75%", or "Above 75%".*
- *When the user clicks on the "Submit Button" two things can happen: If the user has not selected an option from the dropdown menu, nothing changes; whereas if the user did select an option, then the field* ***EstimateGiven*** *is changed to "yes", the* ***SubmitTime*** *is recorded as the current date/time, the* ***Estimate*** *is filled with the option from the dropdown menu, and finally, the field* ***PotentialPlace*** *is set to the study-area initially selected by the user. Consequently, this reveals the "Check-in" button and the "Cancel" button. Clicking on the Check-in button was explained in the previous SDD, but Clicking the Cancel button basically resets the field* ***EstimateGiven*** *to "no".*



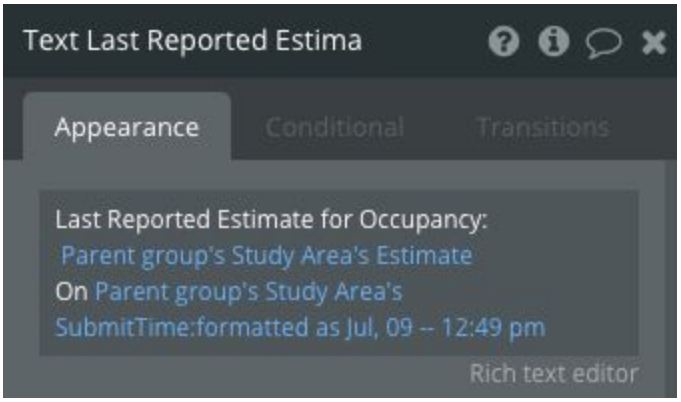

Button Submit is clicked
Element
Button Submit
Event color
Gray
Workflow folder
Only when
Dropdown Estimate Occupancy..'s value is not empty

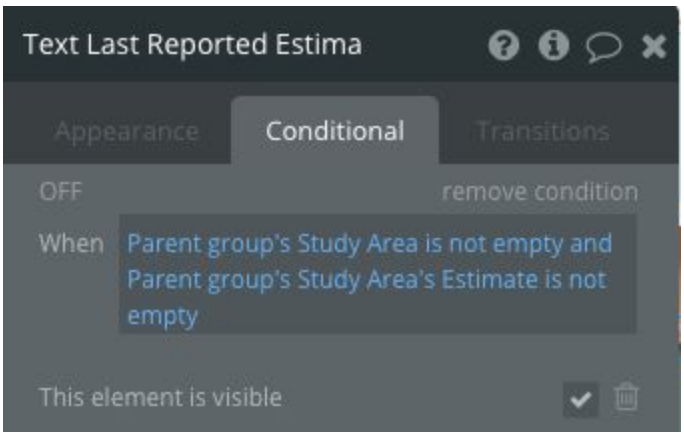

Make changes to User...
Thing to change
Current User
EstimateGiven = "yes"
PotentialPlace = Group Study Area's Study Area


Make changes to Study Area...
Thing to change
Parent group's Study Area
SubmitTime = Current date/time
Estimate = Dropdown Estimate Occupancy..'s value



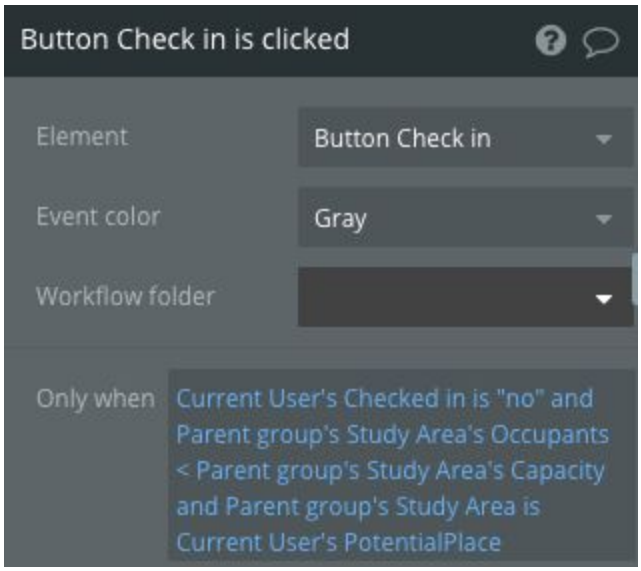
➢ *The following figures show how to display the reported estimate by the user:*



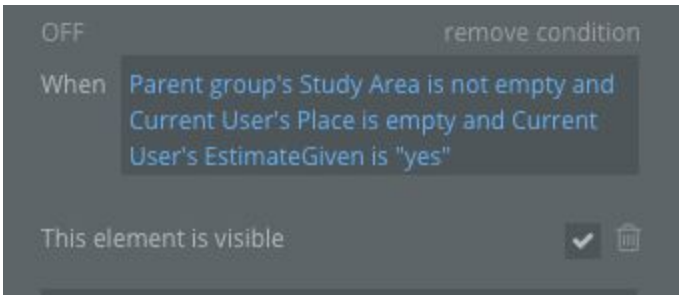


Checking-In Changes

➢ What follows after checking-in remains the same as what was explained in the previous SDD document, however there are some additional changes concerning the conditionals behind allowing the user to check-in in the first place. The conditions are given in the figure below (the same conditions as those mentioned previously but the only addition is in the last statement which indicates that check-in only works when the place in which the user is trying to check-in is already a potential place, meaning that it is a place for which the user has reported an estimate. Because otherwise, the user could report an estimate for a place, and then by mistake could check-in in another one):

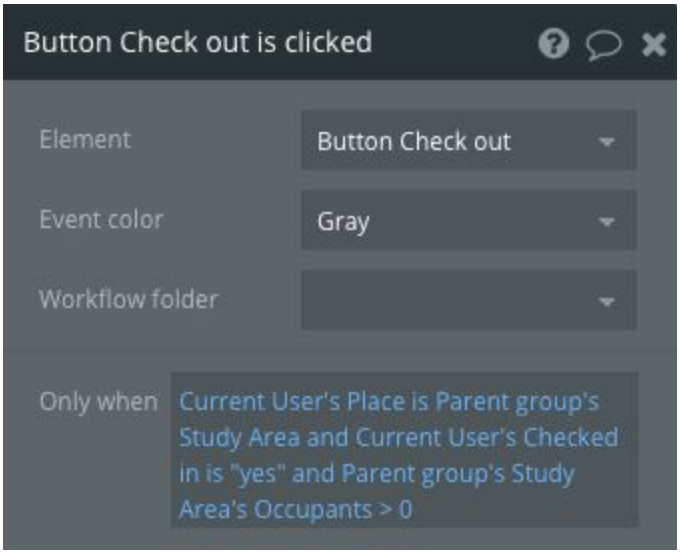


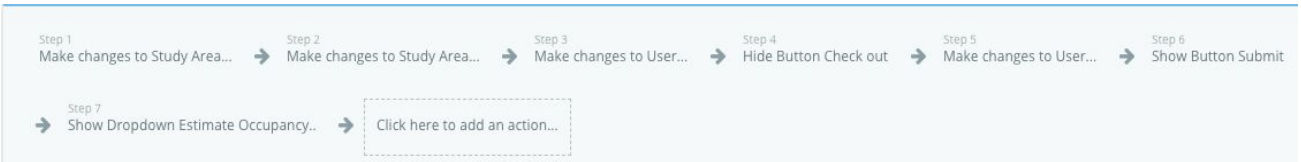
➢ Moreover, since now we have added a cancel button as well, we needed a logic that makes the button visible only when the user has not clicked the check-in button yet and so the field **Place** is still empty.



Checking-Out Changes

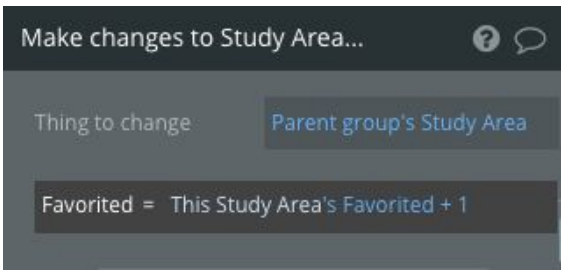
➢ For the check-out procedure, we also keep most of the steps discussed in the previous SDD, but now we also add some additional steps that take care of displaying the Dropdown menu and the Submit button after check-out instead of showing the Check-in button as before (also the field **EstimateGiven** is set back to “no”):

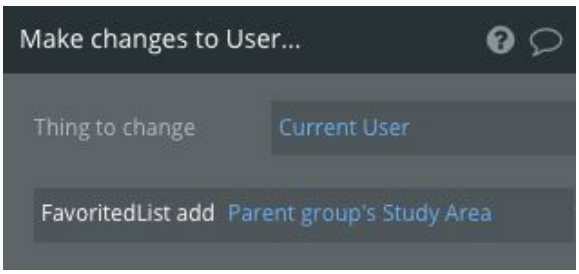




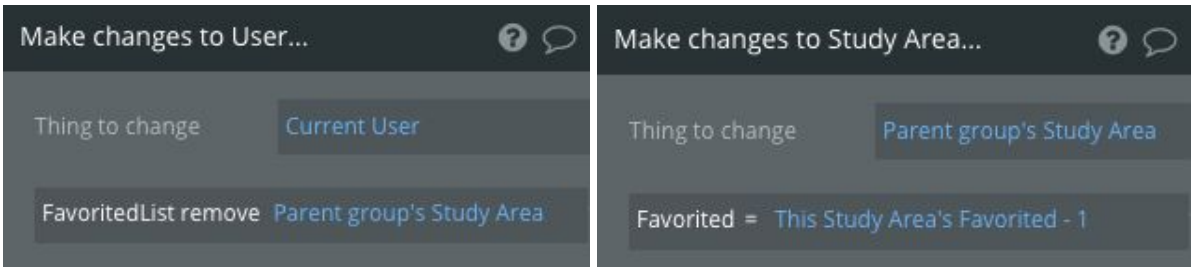
User Favorites

➢ For any study-area option, the user can see an “unfilled-star” icon which he can click in order to turn it into a “filled-star” icon (the opposite can also be done ofcourse). A filled star indicates that the student has set the selected place as one of his favorites, and an unfilled one otherwise. What happens in the back-end when the student clicks on the unfilled star is the following: The field **Favorited** corresponding to the selected study area increases by 1, and the selected place is also added to the field **FavoritedList.**

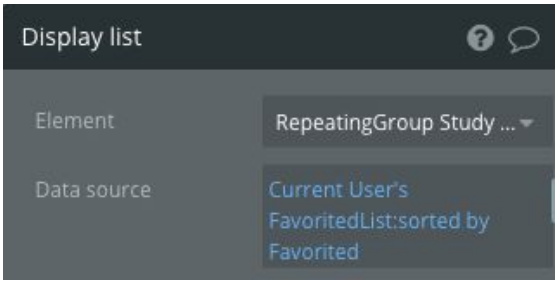




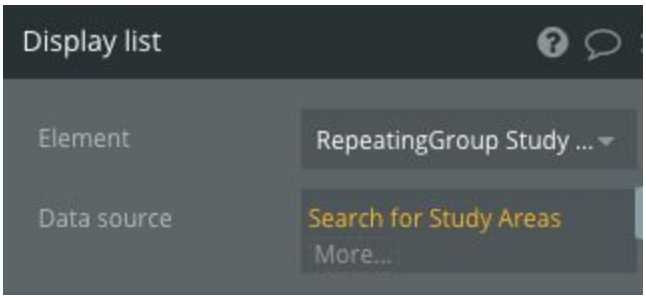
➢ While when the filled star is clicked, **Favorited** is decreased by 1 and the selected place is removed from the **FavoritedList.**

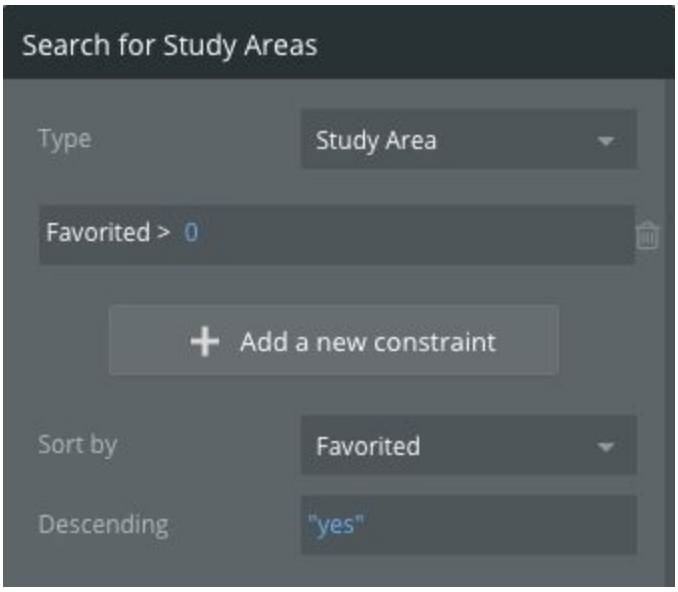


➢ When the “My Favorites” button is clicked, only the study-areas that are present in the current user’s FavoritedList are displayed and when the “Students’ Favorites” button is clicked all the study-areas having at least 1 student who has set it as a favorite, are displayed. They are sorted in descending order of students’ preferences.




When
Button Students' Favorites is clicked


Display list
Element
RepeatingGroup Study ...
Data source
Search for Study Areas
More...


Search for Study Areas
Type
Study Area
Favorited > 0
Add a new constraint
Sort by
Favorited
Descending
"yes"