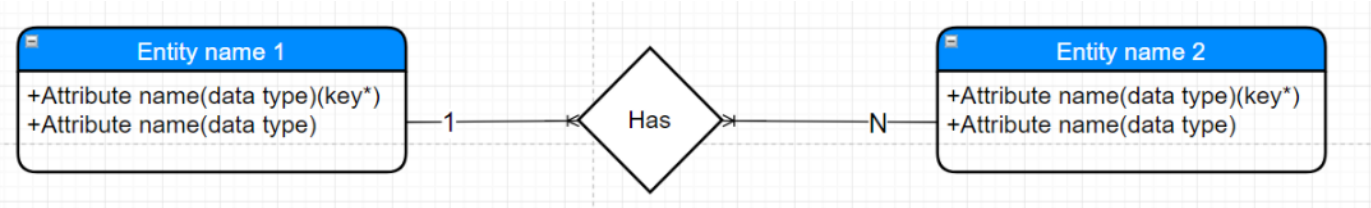
Criterion B

Contents

Database design

Entity relationship diagrams

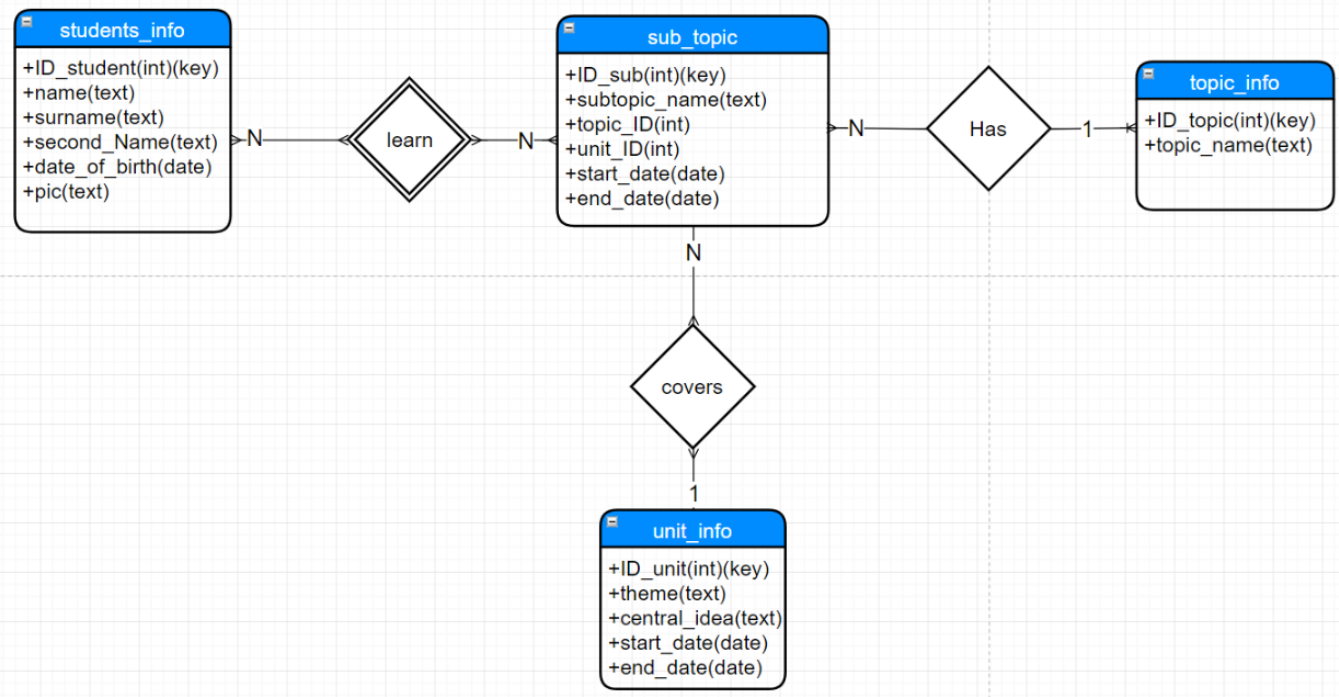
How to read diagrams



Word “key” indicates whether the field is primary key or not

In the example above, the relationship is 1 to many (1 to N). In other words, single “Entity name 1” instance/record can have multiple “Entity name 2” instances/records.

Main entity relationship diagram

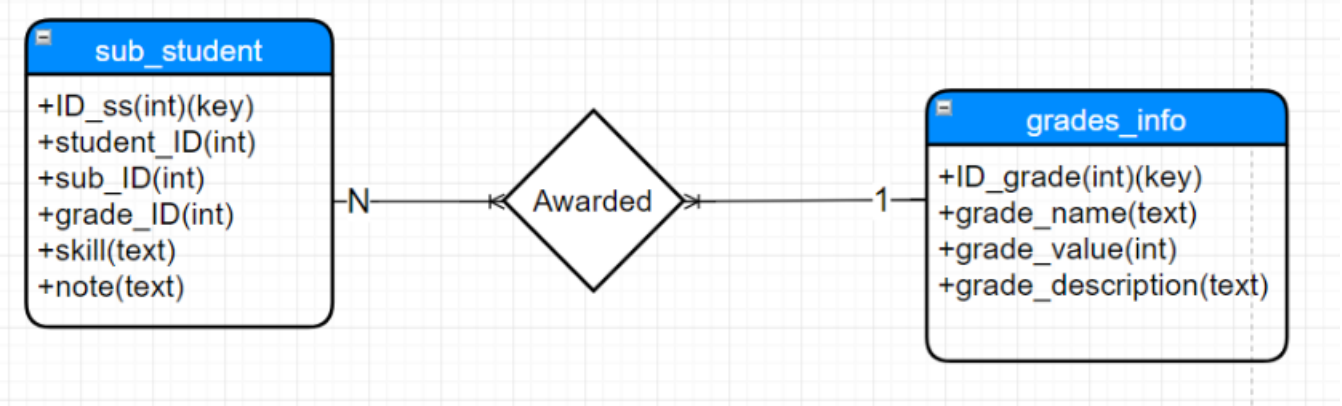


List of relationships in the entity diagram with explanations:

- “topic_info” and “sub_topic” → 1 to many, i.e. 1 topic can have multiple subtopics
- “unit_info” and “sub_topic” → 1 to many, i.e. 1 unit can cover multiple subtopics
- “students_info” and “sub_topic” → many to many, i.e. multiple students can learn multiple subtopics

Extension of the ERD

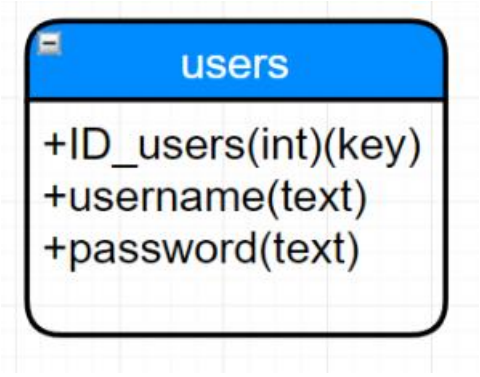
Since there is a “many to many” to relationship between entities “students_info” and “sub_topics”, a new entity needs to be created. This entity is needed to store information about students learning subtopics. In other words, to store remarks. (Further in the document word “remark” and “record of “sub_student” have the same meaning)



In the extension of the ER diagram, there are 2 new entities created, “sub_student” and “grades_info”. “sub_student” is the product of many to many relationship, while “grades_info” is an entity to store information about grades (see *Overview* for more information) The relationship between them is 1 to many, i.e. a single grade can be awarded per remark.

Entity needed for security

One more entity is needed to ensure security of the system. This entity has to store a usernames and passwords needed to be entered in order to access the system. This entity is, however, not connected to any other entities. Consequently, it was not needed to be included in the main ER diagram.



Tables needed to create according to ER diagram and their description

students_info

Field name	Data type	Description	Other details
ID_students	Integer	Uniquely identifies student	Primary key, autoincremented
name	Text	Stores name of a student	
surname	Text	Stores surname of a student	
second_Name	Text	Stores second name of a student	Can be null
date_of_birth	Date	Stores date of birth of a student	Follows yyyy/mm/dd format
pic	Text	Stores location of a picture for a student	

topic_info

Field name	Data type	Description	Other details
ID_topic	Integer	Uniquely identifies topic	Primary key, autoincremented
topic_name	Text	Stores name of a topic	

unit_info

Field name	Data type	Description	Other details
ID_unit	Integer	Uniquely identifies unit	Primary key, autoincremented
theme	Text	Stores name of a theme for a unit	
central_idea	Text	Stores name of a central idea for a unit	
strart_date	Date	Stores date when unit starts	Follows yyyy/mm/dd format
end_date	Date	Stores date when unit ends	Follows yyyy/mm/dd format

sub_topic

Field name	Data type	Description	Other details
ID_sub	Integer	Uniquely identifies subtopic	Primary key, autoincremented
subtopic_name	Text	Stores name of a subtopic	
unit_ID	Integer	Stores ID of the unit to which subtopic belongs	Foreign key
topic_ID	Integer	Stores ID of the topic to which subtopic belongs	Foreign key
start_date	Date	Stores date when subtopic starts	Follows yyyy/mm/dd format

end_date	Date	Stores date when subtopic ends	Follows yyyy/mm/dd format

sub_student

Field name	Data type	Description	Other details
ID_ss	Integer	Uniquely identifies remark	Primary key, autoincremented
sub_ID	Integer	Stores ID of the subtopic to which remark was written for	Foreign key
student_ID	Integer	Stores ID of the student which remark was written for	Foreign key
grade_ID	Integer	Stores ID of the grade	Foreign key
skill	Text	Stores name of a skill which student learnt in subtopic	
note	Text	Stores note for teacher	Can be null
date_added	Date	Stores date when remark was added or edited last time	Follows yyyy/mm/dd format

grades_info

Field name	Data type	Description	Other details
ID_grade	Integer	Uniquely identifies grade	Primary key, autoincremented
grade_name	Text	Stores name of a grade	
grade_value	Integer	Stores value of a grade out 5	
description	Text	Stores description of a grade	Can be null

Users

Field name	Data type	Description	Other details
ID_user	Integer	Uniquely identifies user	Primary key, autoincremented
username	Text	Stores username	
password	Text	Stores password of a user	

Actions to be developed

- Add data
- Delete data
- Edit data

- View data

Flowcharts representing actions for adding information to database

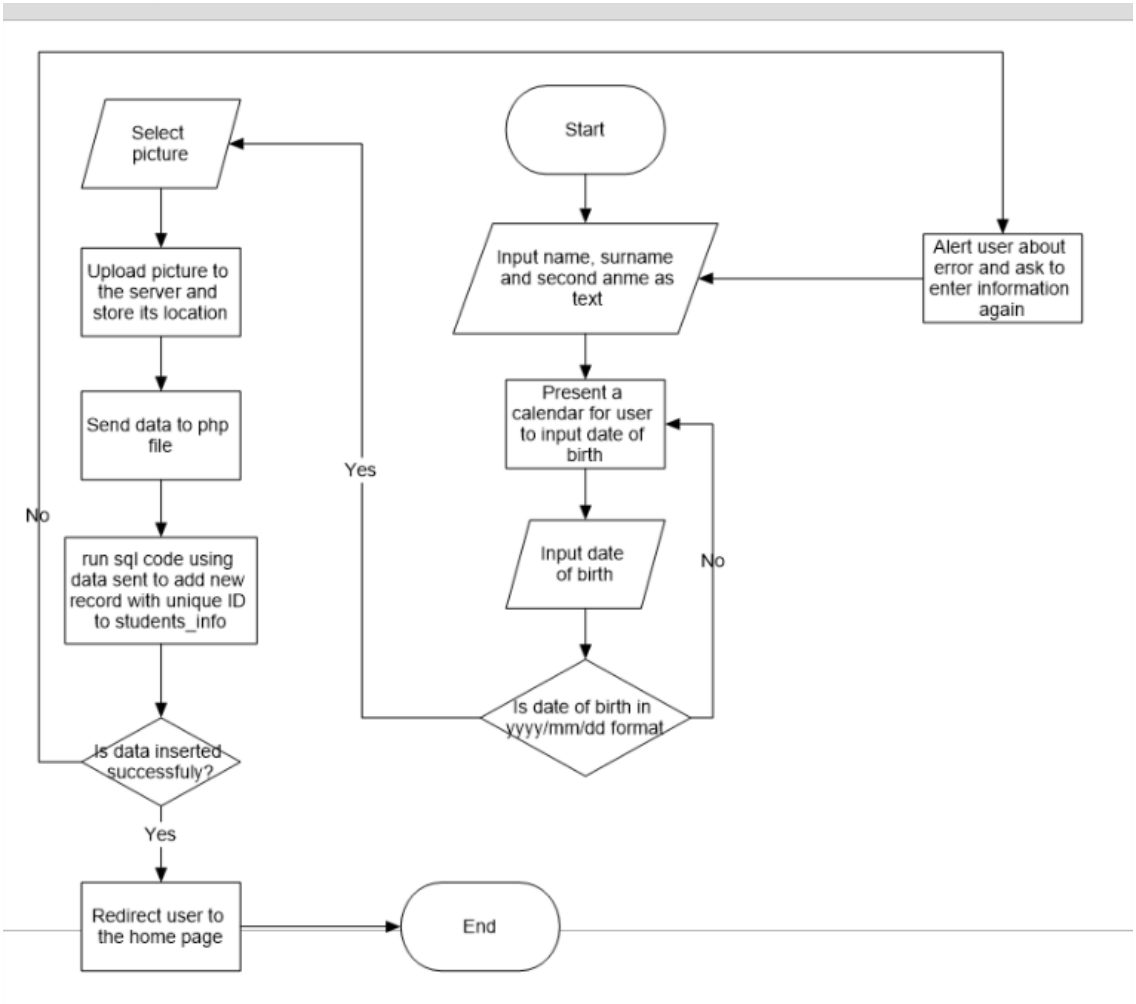


Figure 1 Add student information

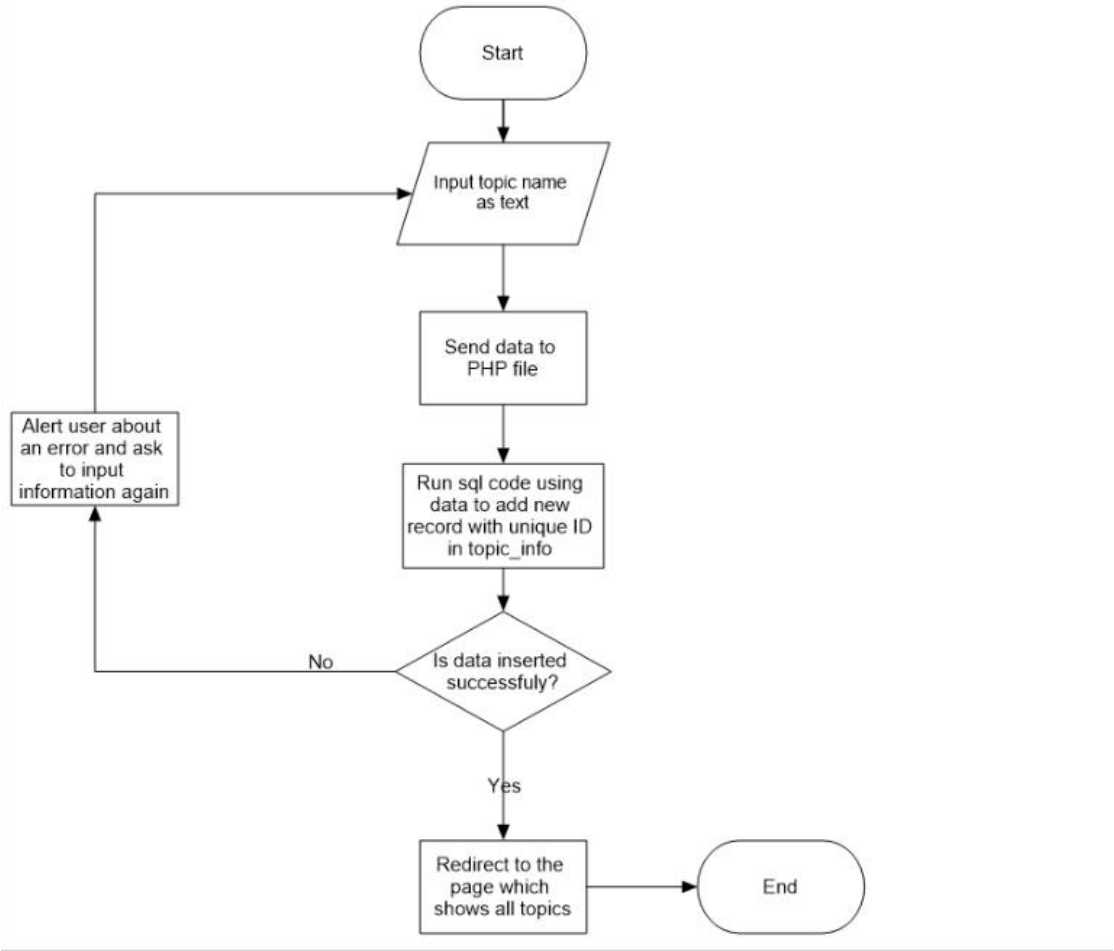


Figure 2 Add topic

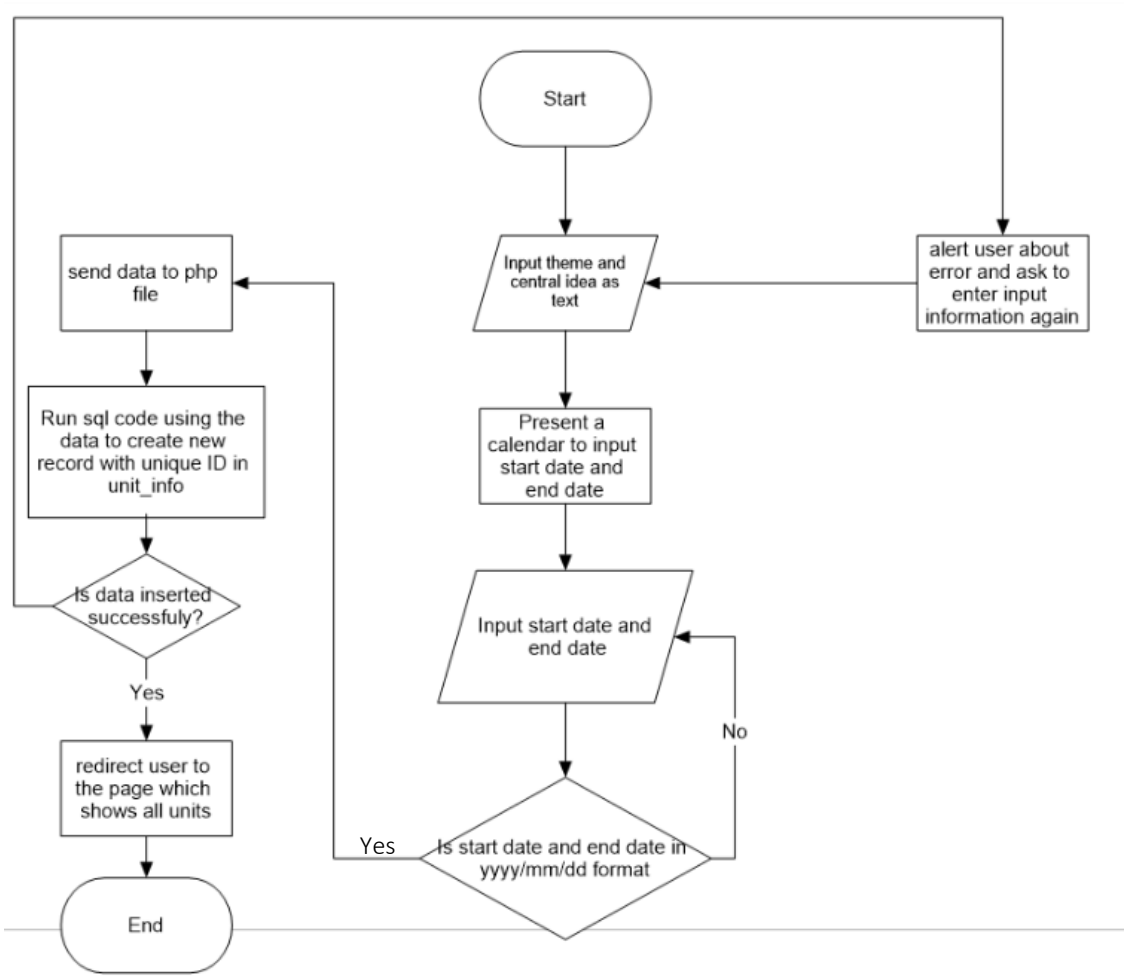


Figure 3 Add unit

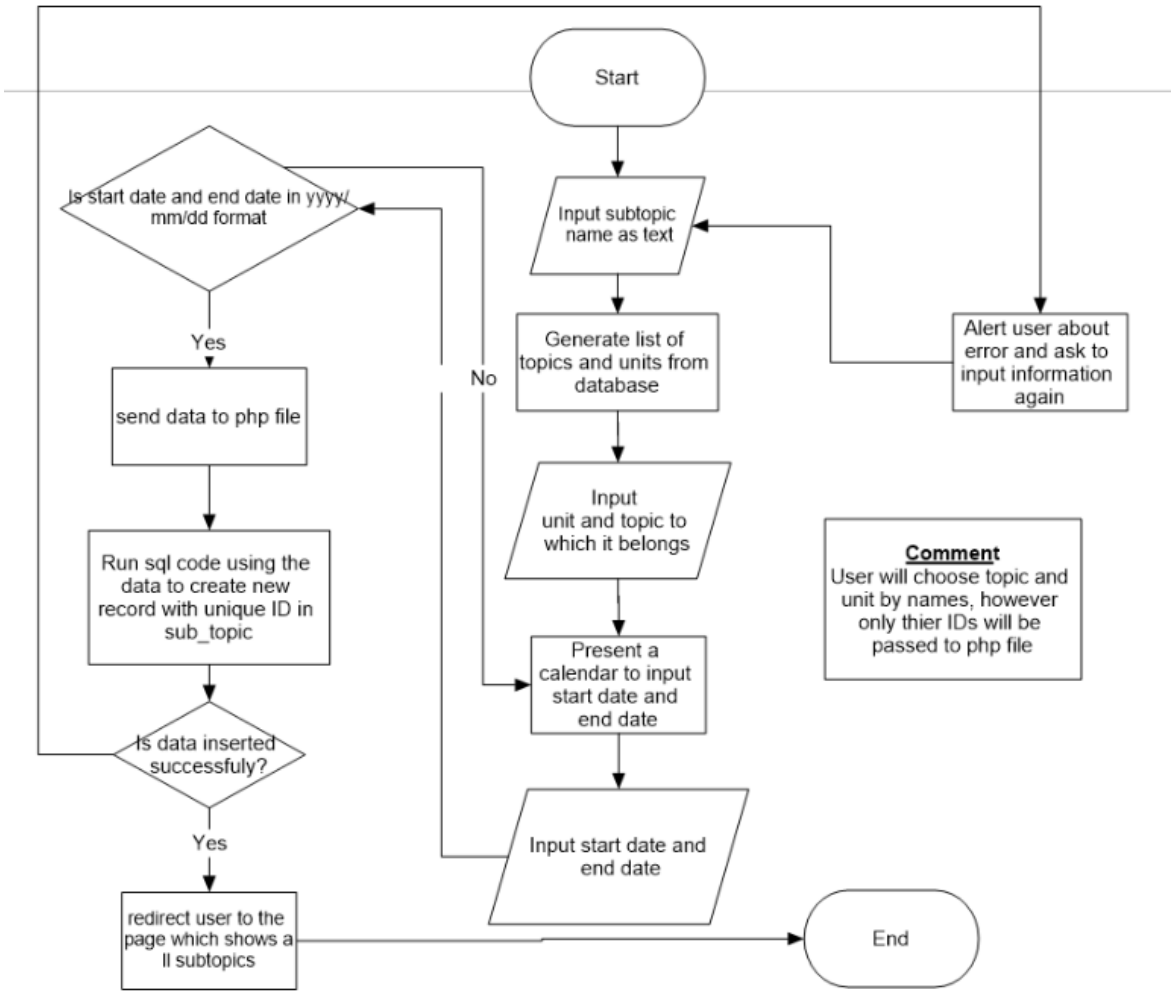


Figure 4 Add subtopic

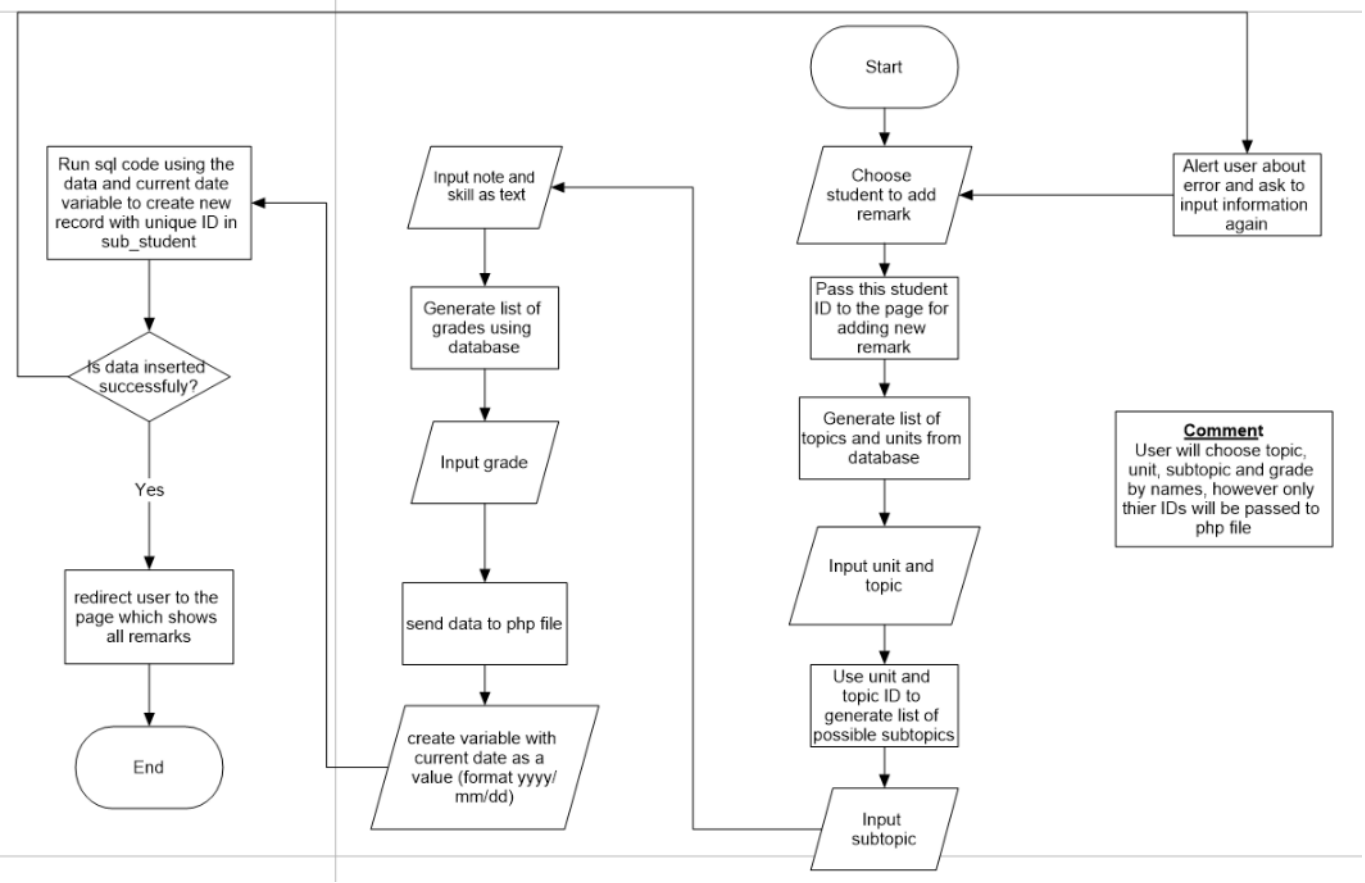


Figure 5 Add remark

Overview of flowcharts

As you can see from flowcharts, to add information user first has to input/type information in input boxes. This information, once user hits submit button, is sent to PHP file, where connection with database is established and SQL code runs to add information to the database.

It should be, however, noted that there are different ways information can be inputted. Generally, to input information, user will have to type in an input boxes, but there are other ways too. For example, when user wants to add new subtopic, the user will choose unit and topic, to which subtopic belongs, by choosing the names from a dropdown menu (see *Design of Input boxes* for more information). The dropdown menu is generated by retrieving data from database. Once topic and unit are chosen, their according IDs will be sent to the PHP file. This is done due to the fact that foreign keys sometimes used to refer to information from other places.

Flowcharts representing actions to view information from database

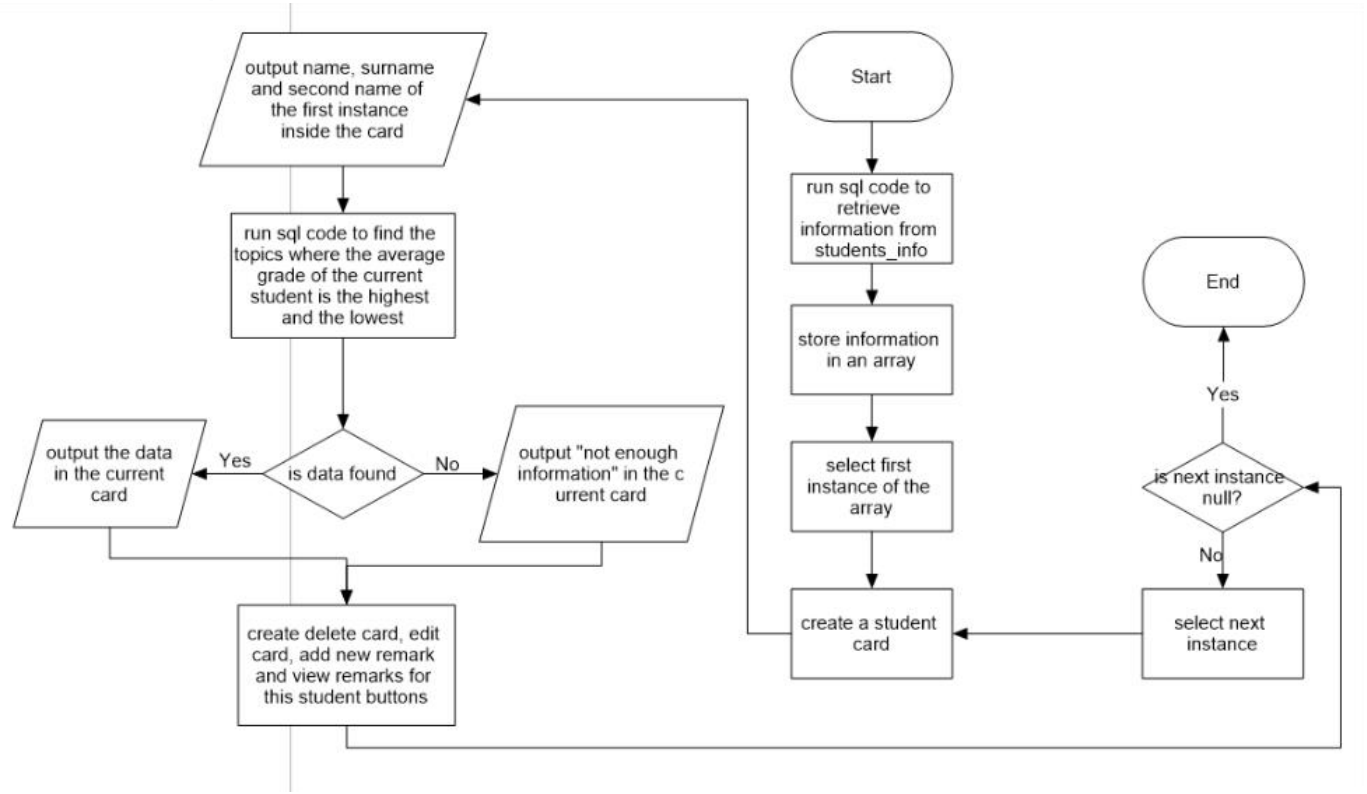


Figure 6 Viewing all students

Since the main purpose of this solution was creating remarks for different students, the page which shows information about all students became the home page. Using this page, edit student information and delete student information; and also go to the pages for creating new remark and viewing all remarks relating the student. *Figure 6* presents how data for the home page is generated.

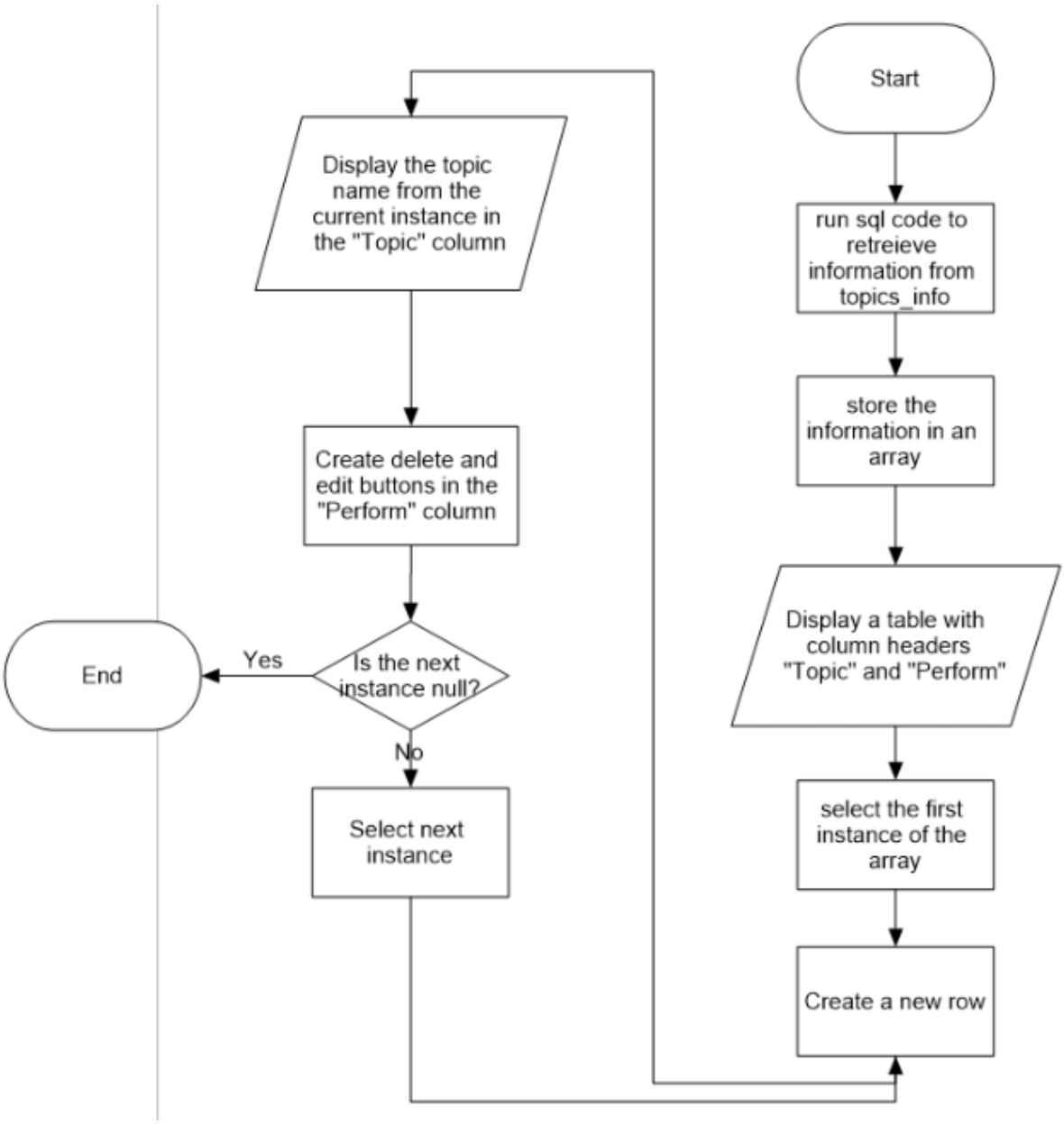


Figure 7 View topics

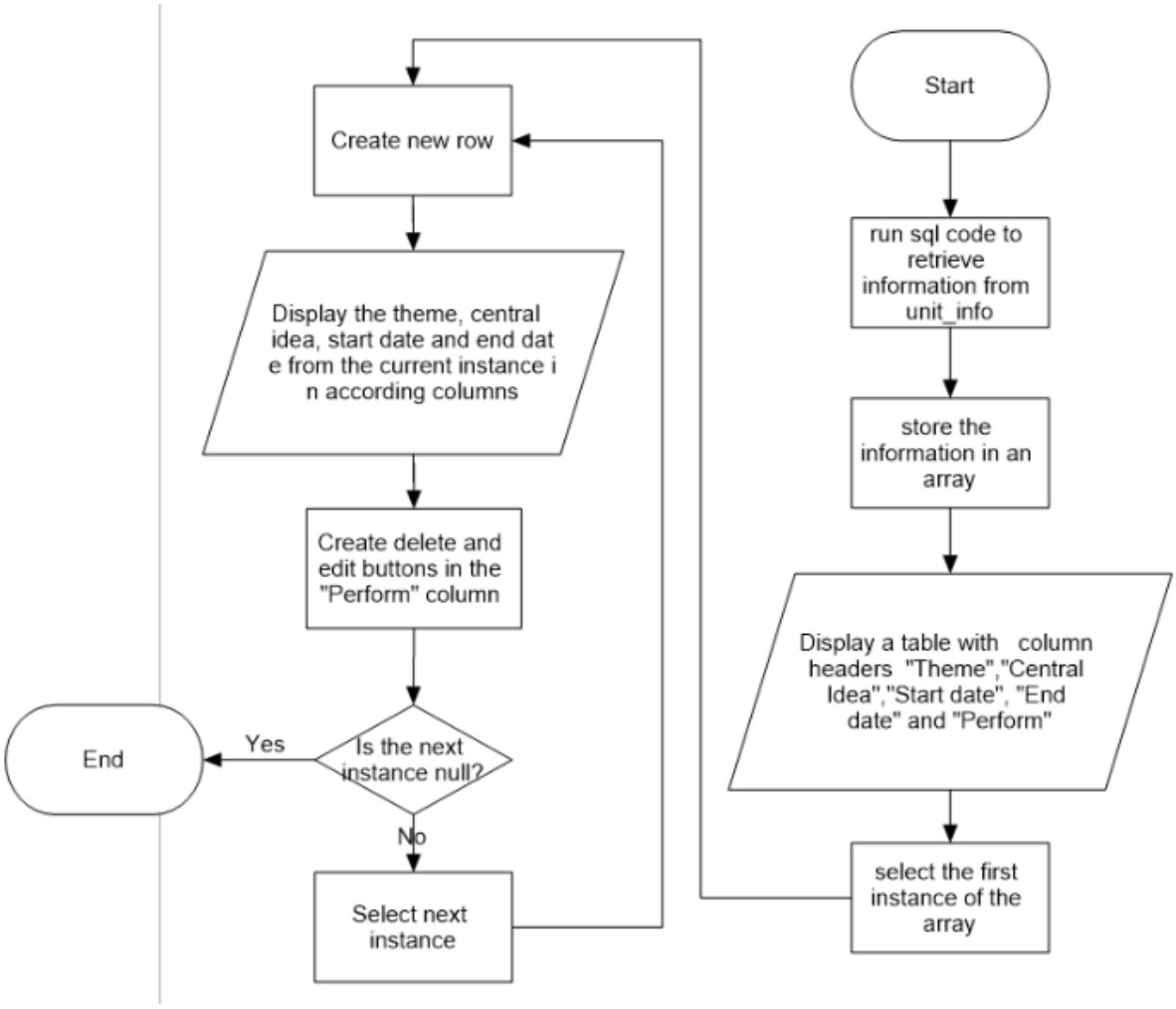


Figure 8 View units

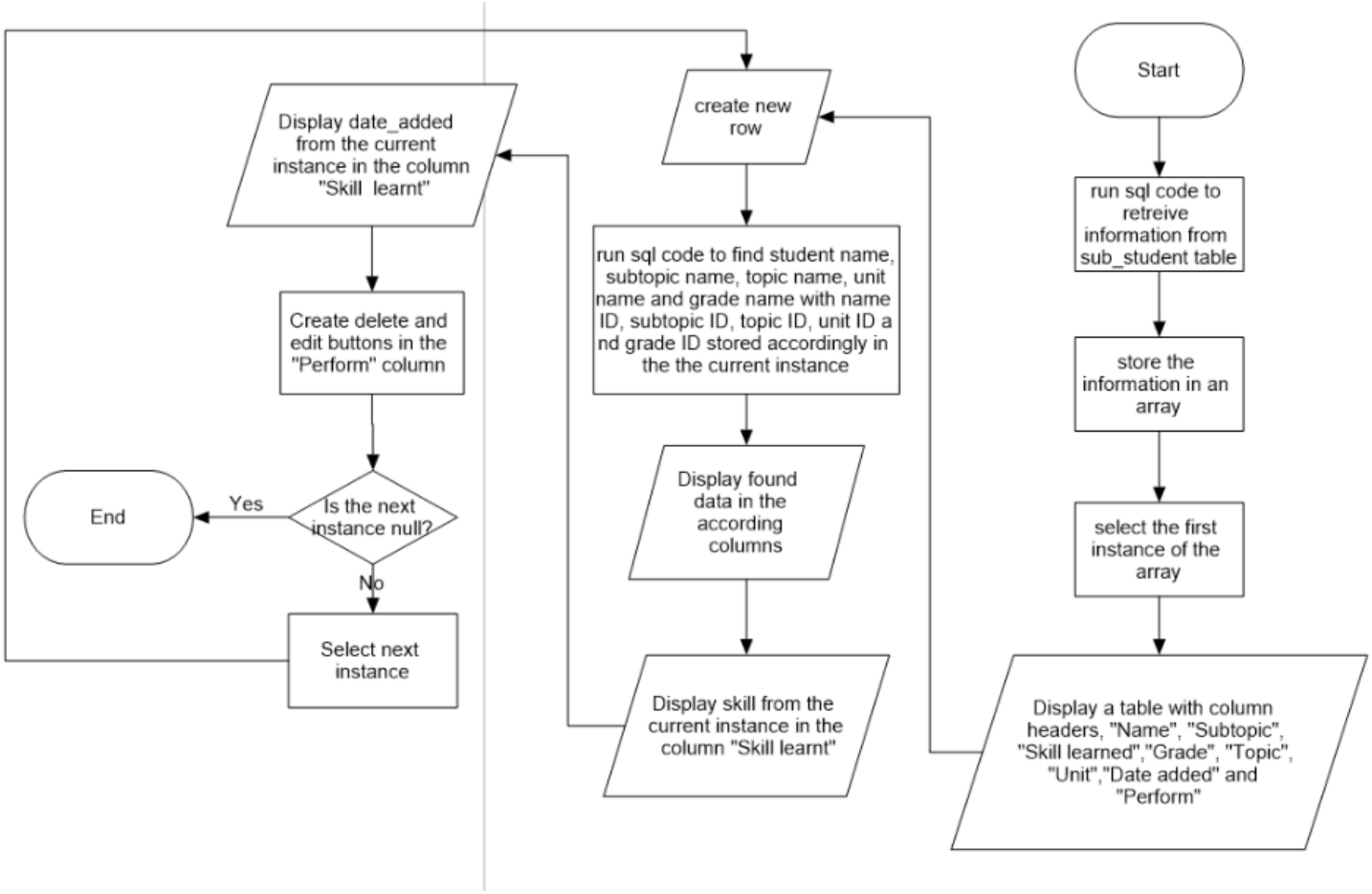


Figure 9 View remarks

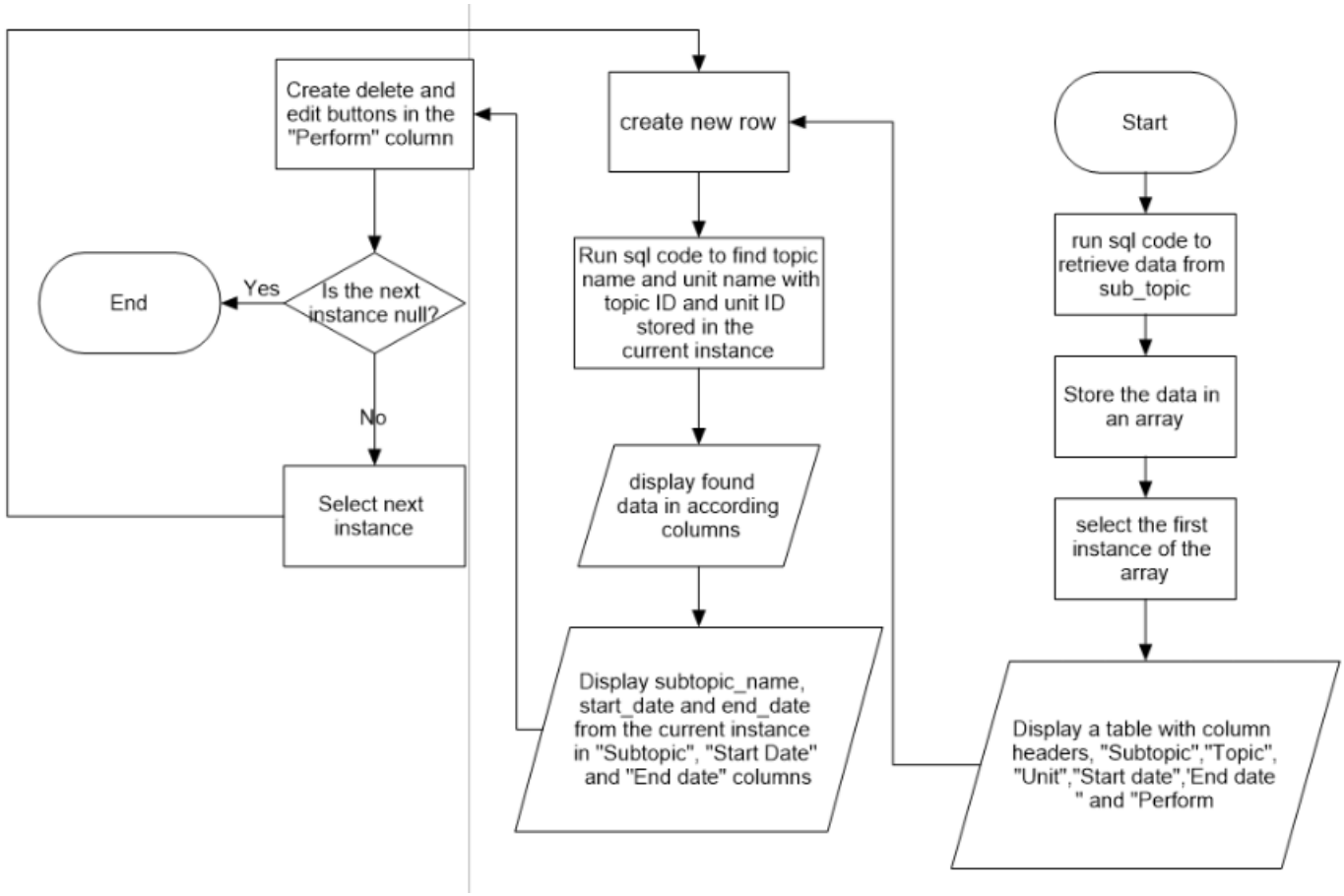


Figure 10 View subtopics

Overview of flowcharts

As you can see from flowcharts, in order to display data, first, SQL code is needed to retrieve data from database. Then, the data retrieved is stored in an array. After the storing is done, a table is generated; and a loop starts to create a new row for each element in array, and display the element in table.

It should also be noted that complex SQL search is needed in some cases. This is due to the fact that foreign keys are used to refer to the data from different tables. For example, when retrieving data from “sub_topic” table, ID of units and topics will be part of the data retrieved. These IDs will not be displayed for user. Instead, they will be used to refer to tables “unit_info” and “topic_info” in order to retrieve names of topics and units.

Flowcharts representing actions to edit data in the database

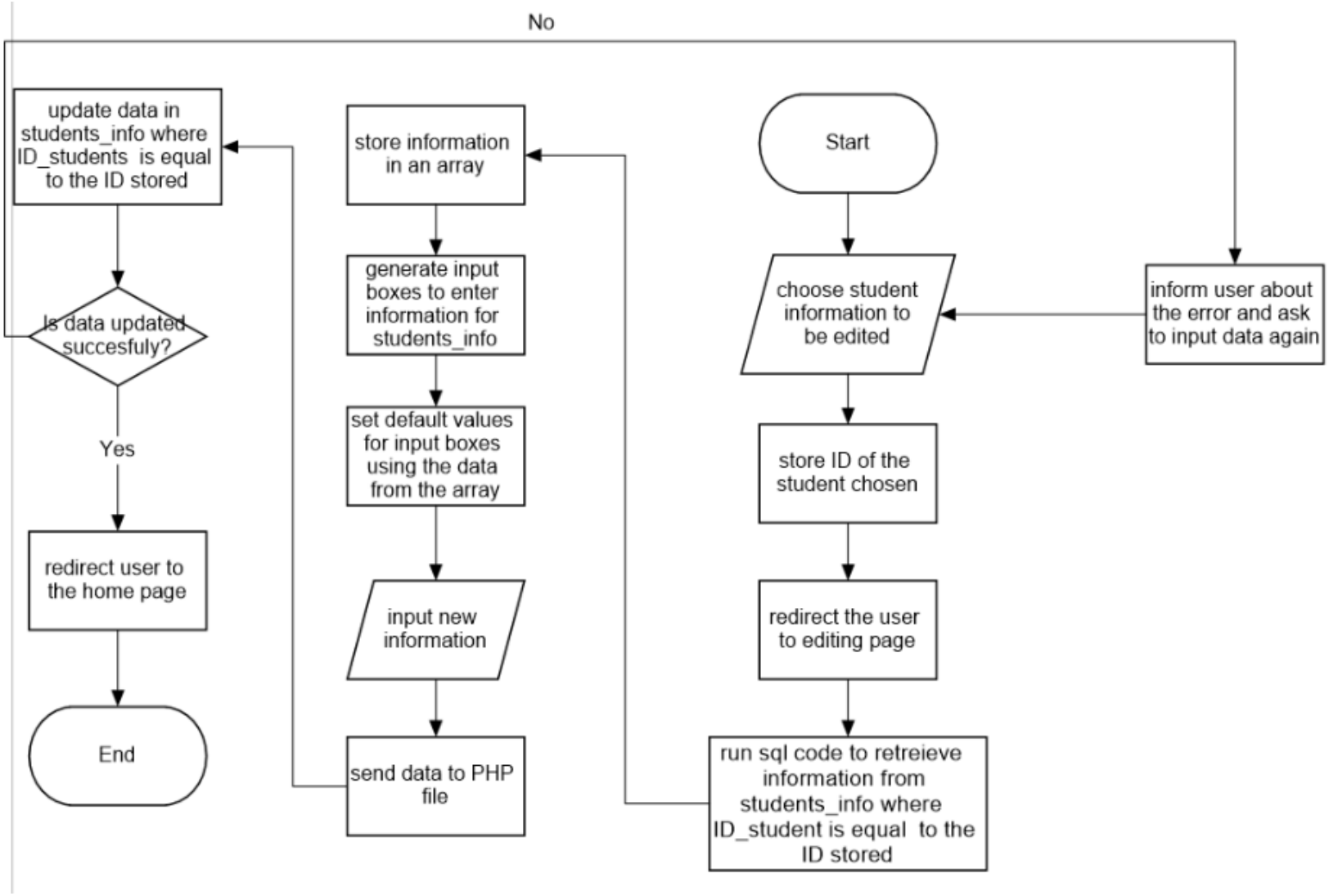


Figure 11 Edit student information

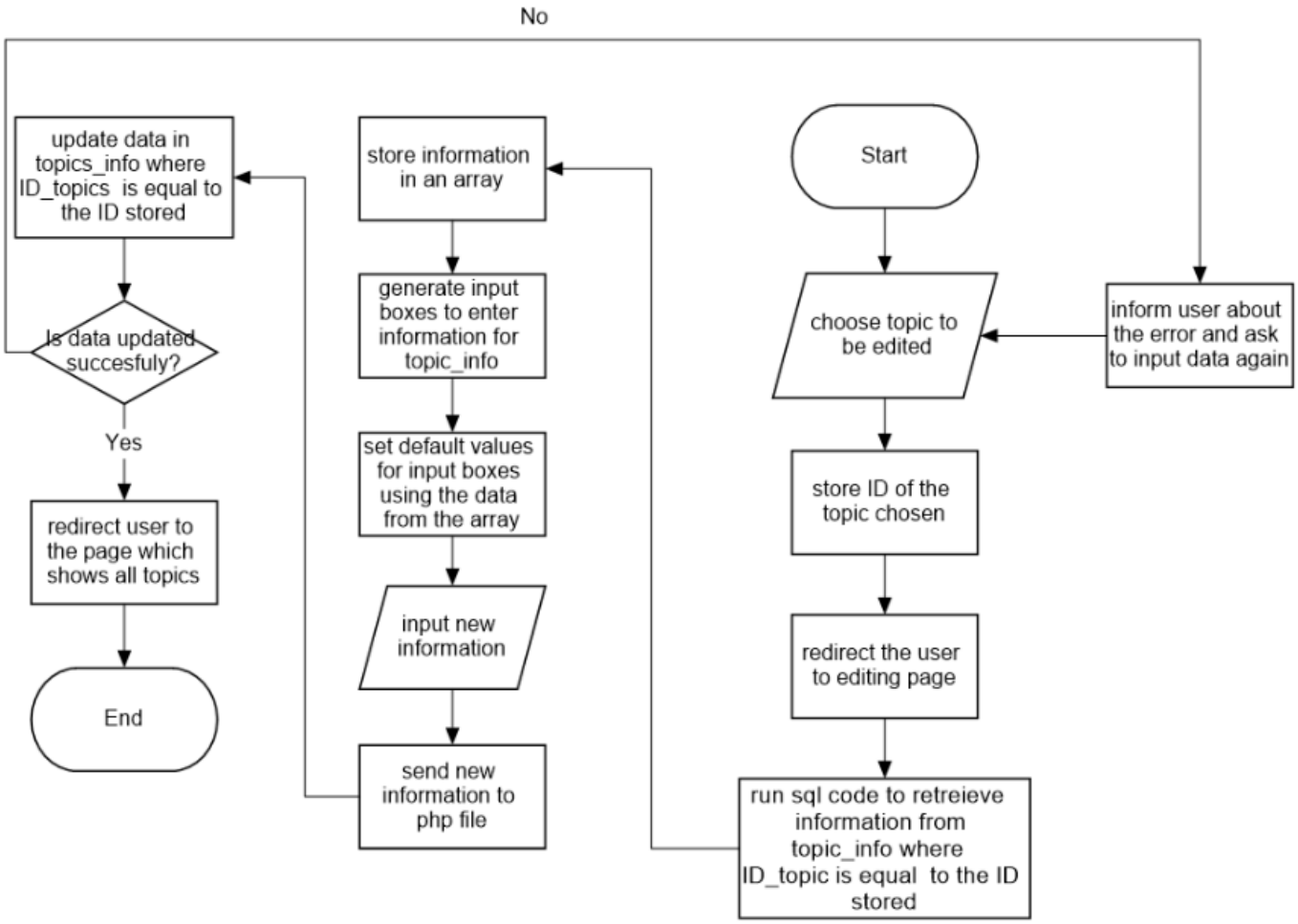


Figure 12 Edit topic information

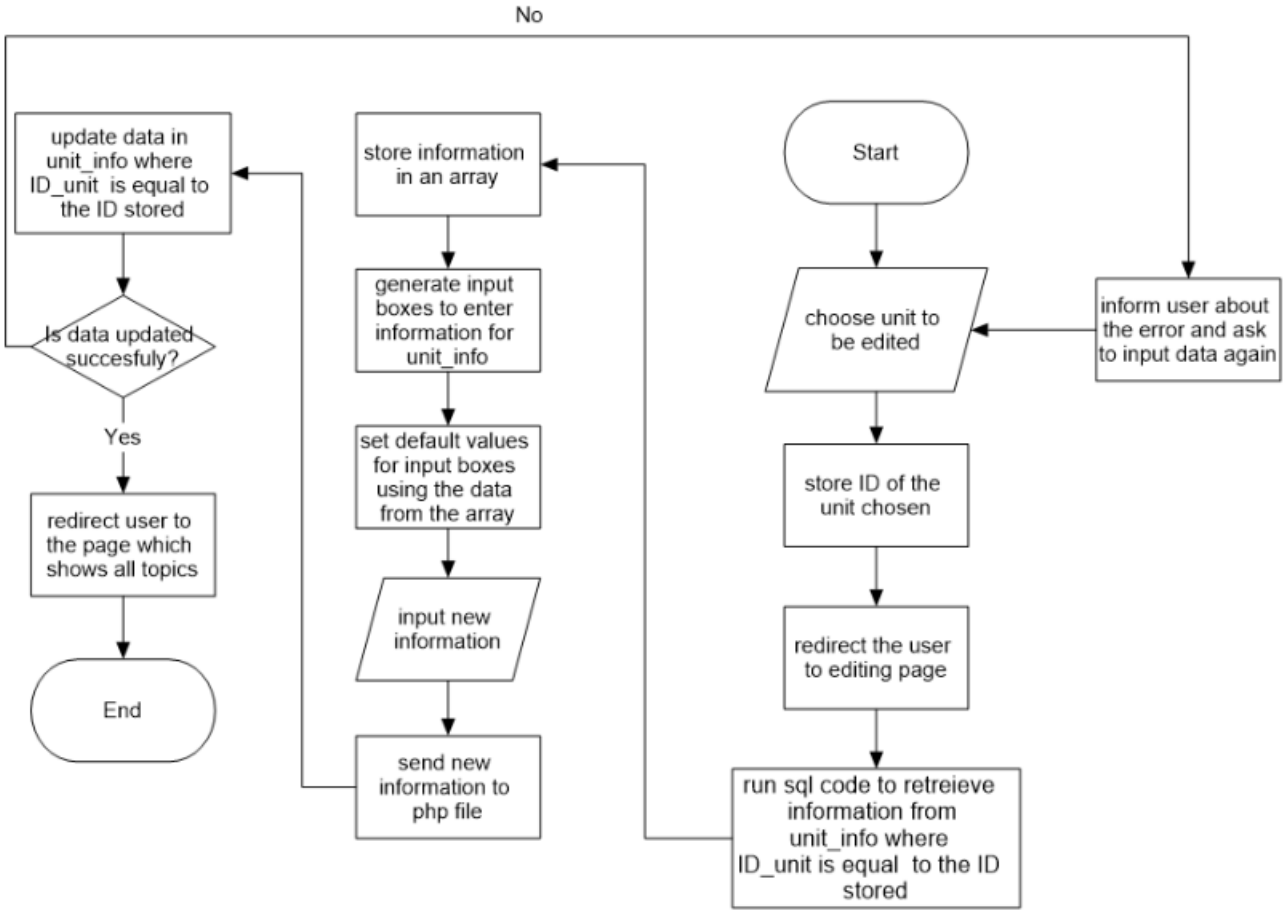


Figure 13 Edit unit information

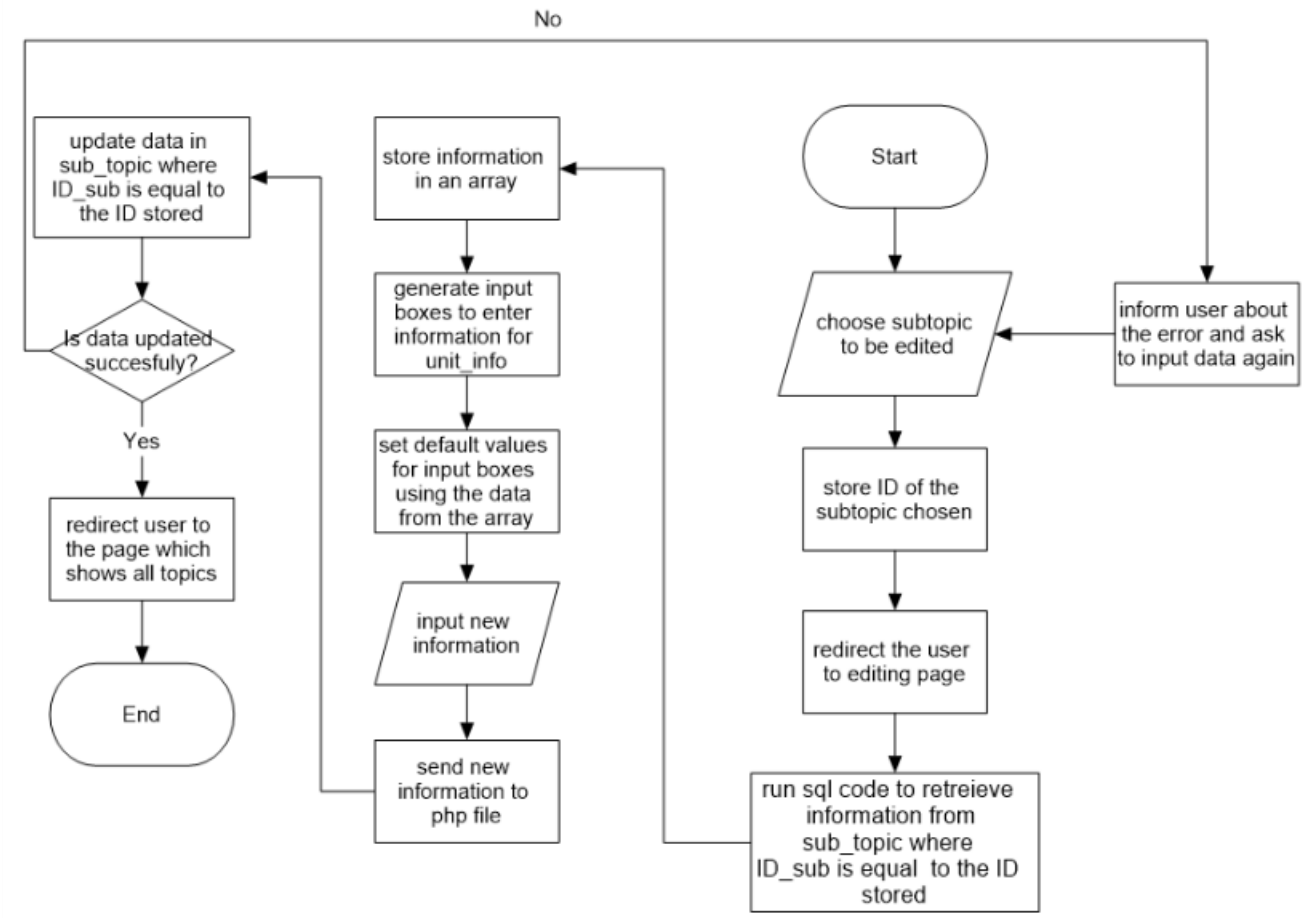


Figure 14 Edit subtopic information (special edit case)

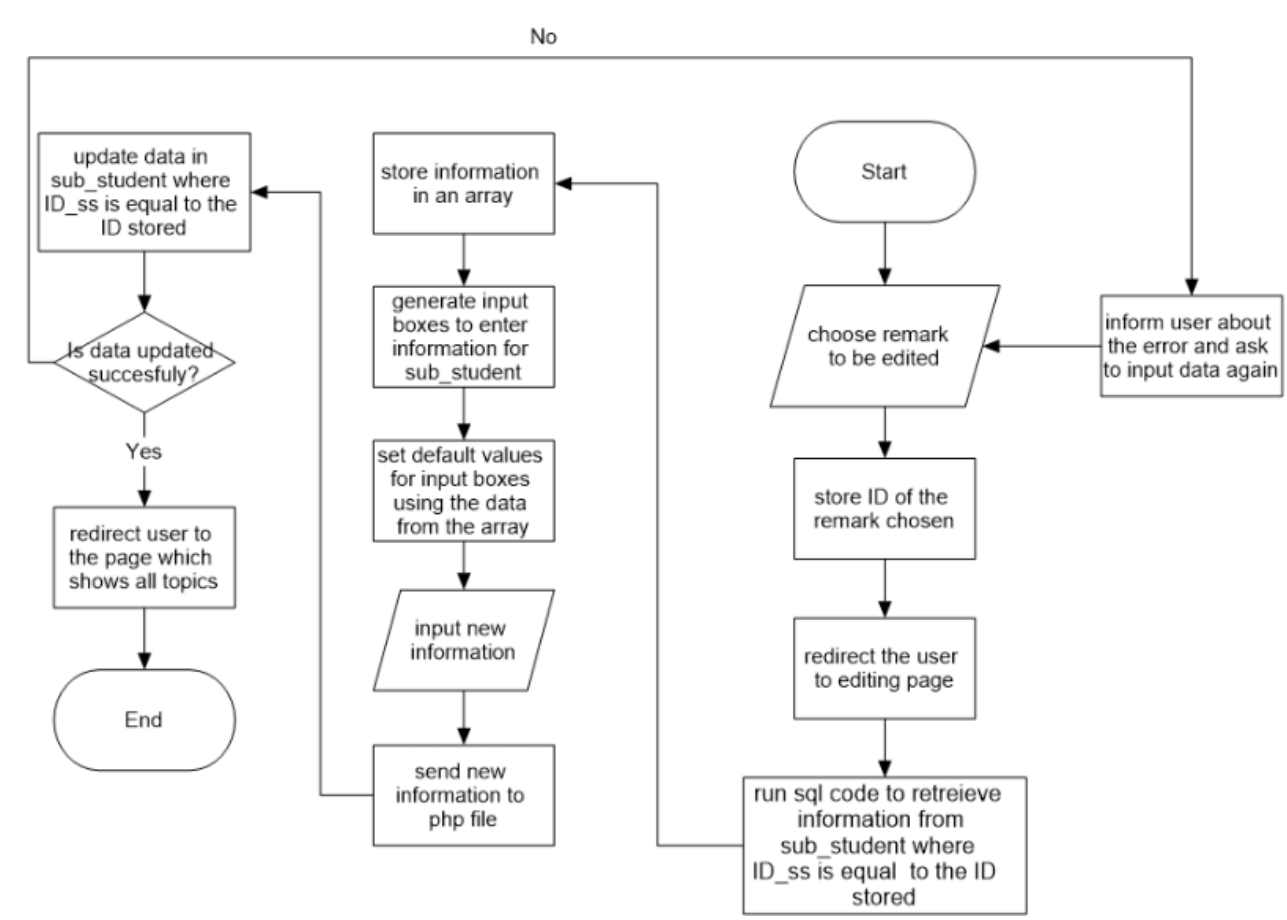


Figure 15 Edit remark (special edit case)

User is able to edit remarks from the page which shows all remarks and all remarks related to a certain student. In order to start editing, the user has to be redirected to the editing page. Although in both cases redirecting happens from 2 different pages, the flow of processes is the same in both cases.

Overview of flowcharts

As you can see from flowcharts, first data is retrieved from database. This data is used for default values in input boxes. User can change this data and later submit the data. This data is then sent to PHP file, where SQL code runs to update information in the database using the data sent.

It should be noted that are some *special edit cases*, due to different ways user input information. Since, the usual way to input data is by typing in input box, the default data will be displayed as a string in the input box. However, in the case of drop down menu, the default option will be displayed.

Flowcharts representing actions to delete information in the database

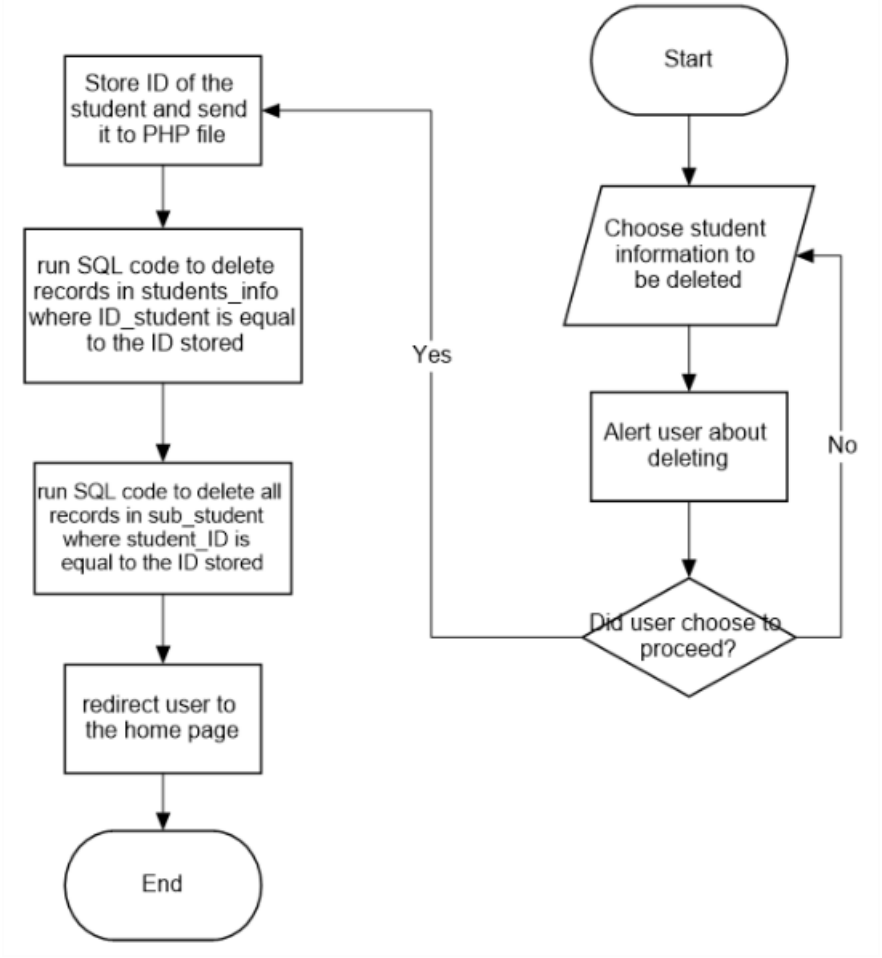


Figure 16 Delete student information

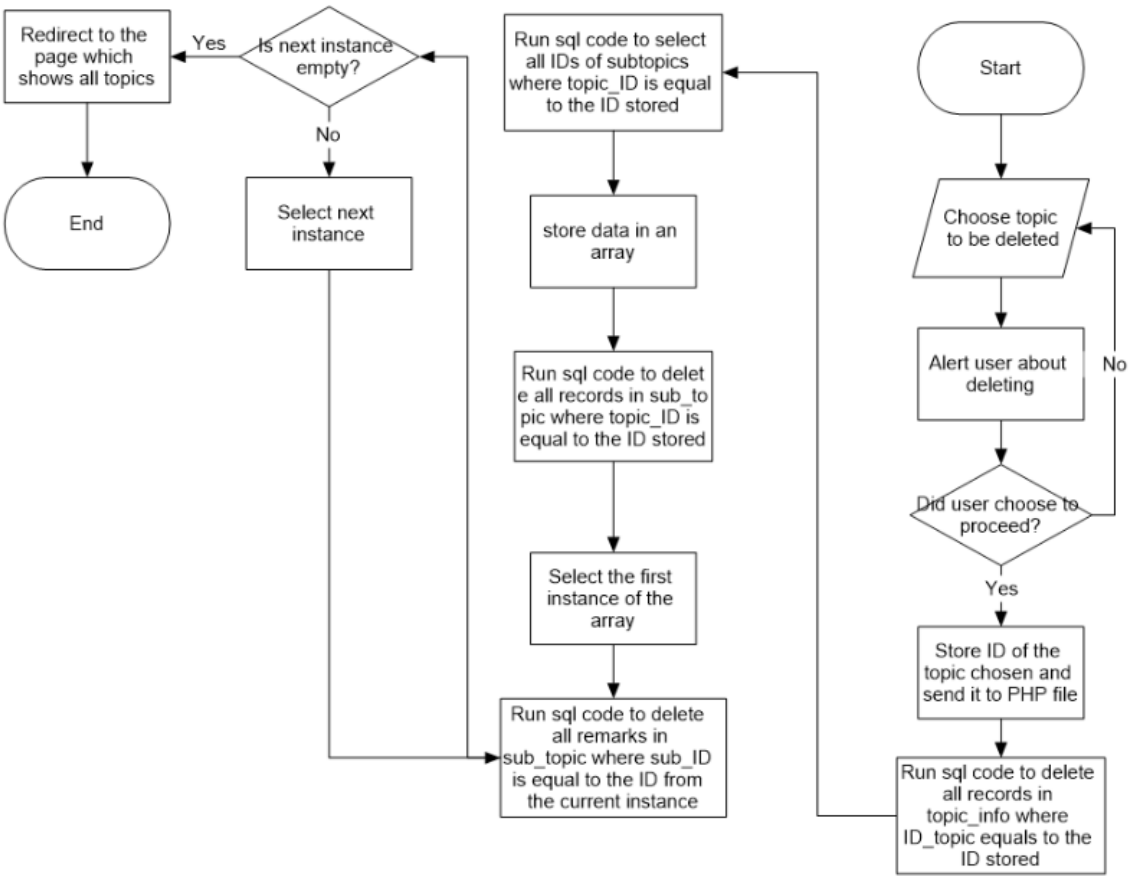


Figure 17 Delete topic information

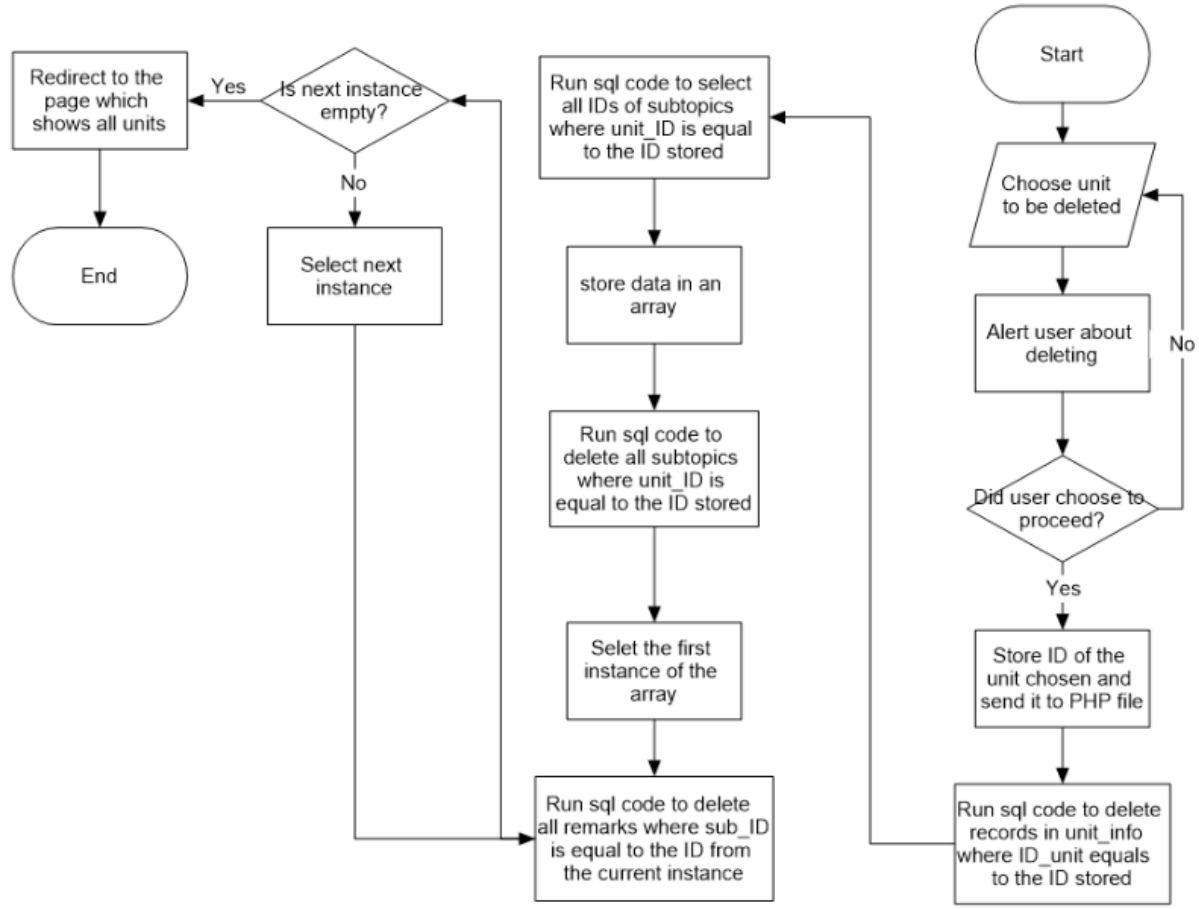


Figure 18 Delete unit information

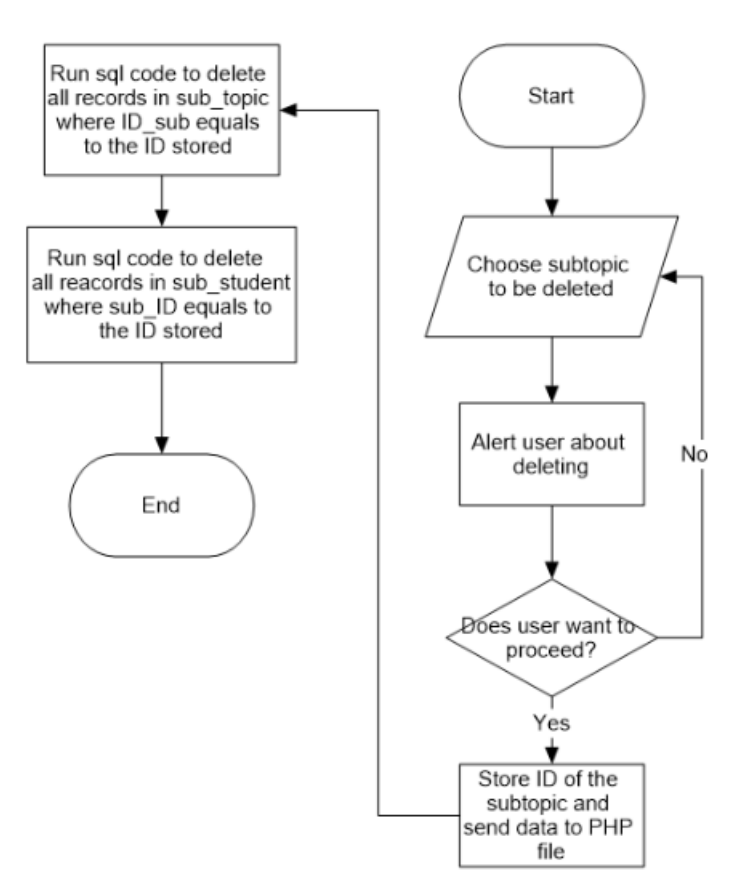


Figure 19 Delete subtopic information

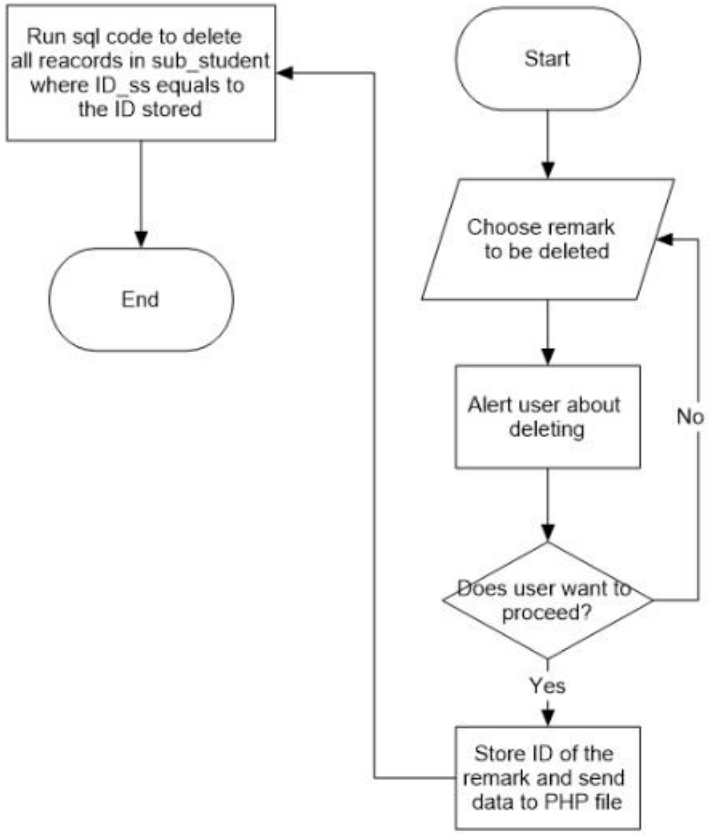


Figure 20 Delete remark information

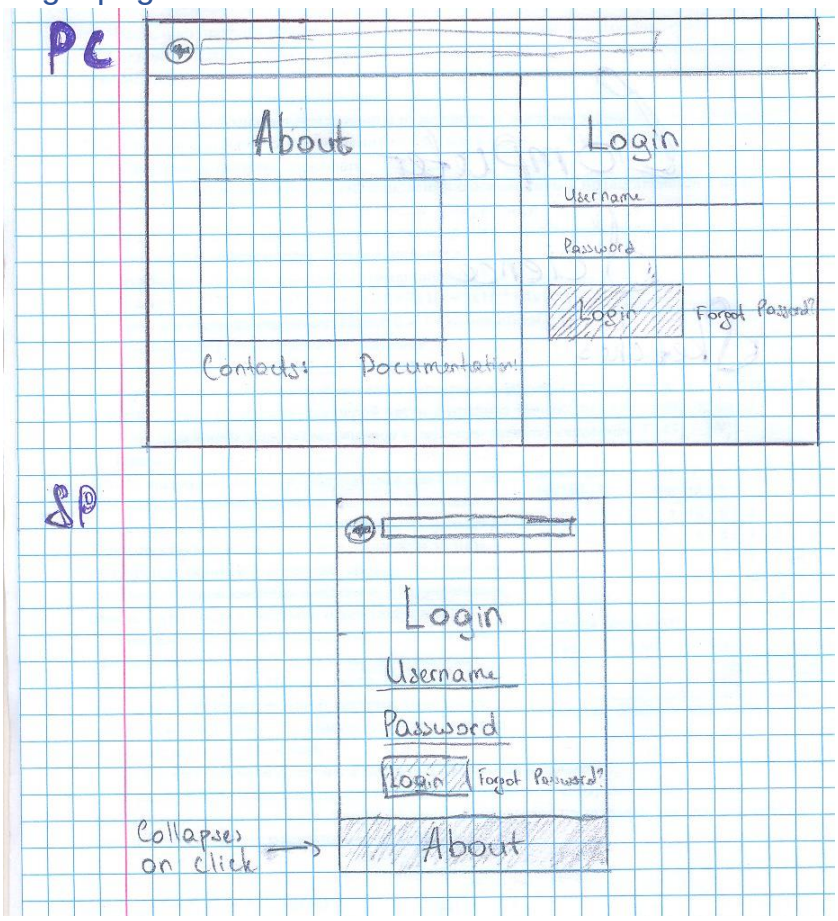
Overview of flowcharts

As you can see from the flowchart, in order to delete information, the ID of the information chosen is sent to PHP file. Inside of the PHP file, SQL code runs to delete records with ID that are equal to the ID sent.

It should be also remembered that records in related tables must also be deleted. For example, if user wants to delete a topic, related subtopics have to be also deleted. Furthermore, related remarks have to be deleted. This is done to avoid any data redundancy, e.g. have a remark for a non-existent subtopic or for a non-existent student.

Web pages sketches

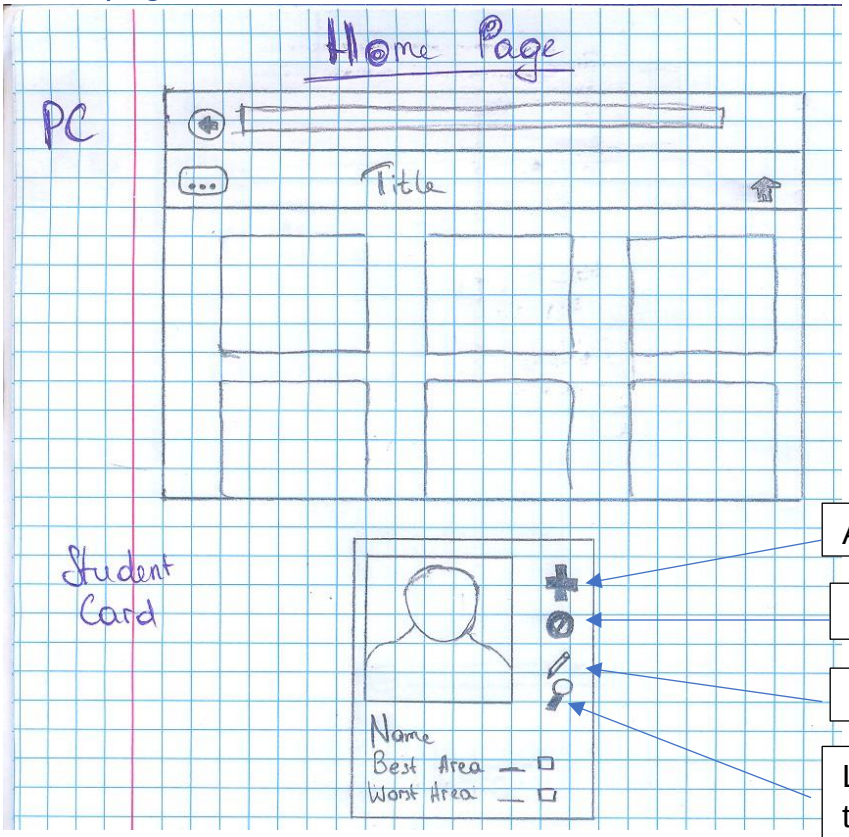
Login page



Version of a log in page for big screens

Version of log in page for small screens

Home page and student cards



Version for big screens should have 3 cards per row, while version for small screens should have 1 card

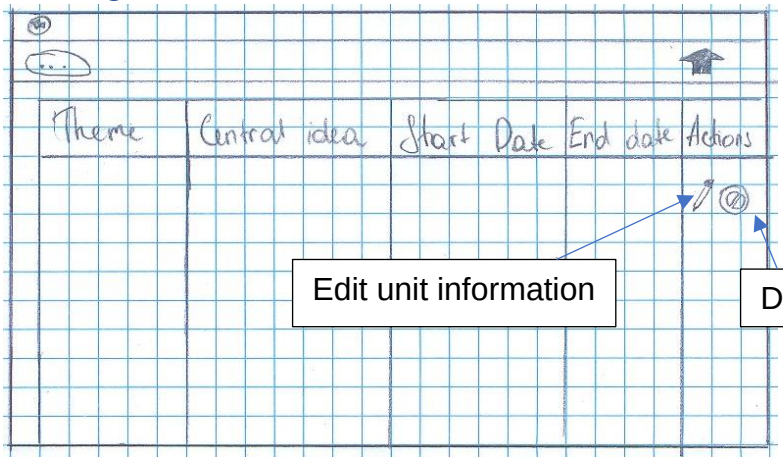
Add new

Delete student

Edit student information

Look at all remarks for this student

Viewing all units



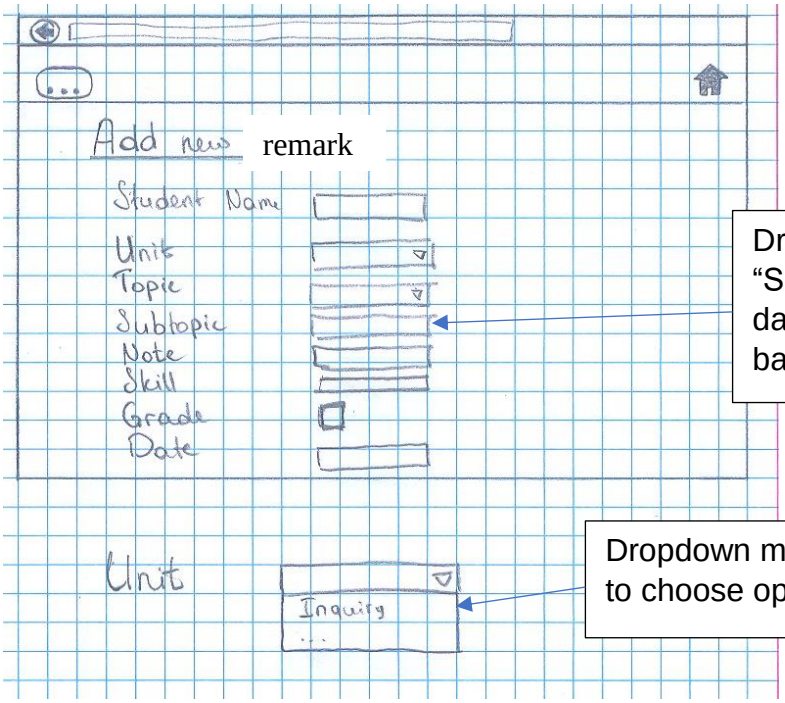
Look at *Figure 8* for processes needed to happen for this page to be generated

Edit unit information

Delete

All view pages should have similar design – simple table with 2 button per row

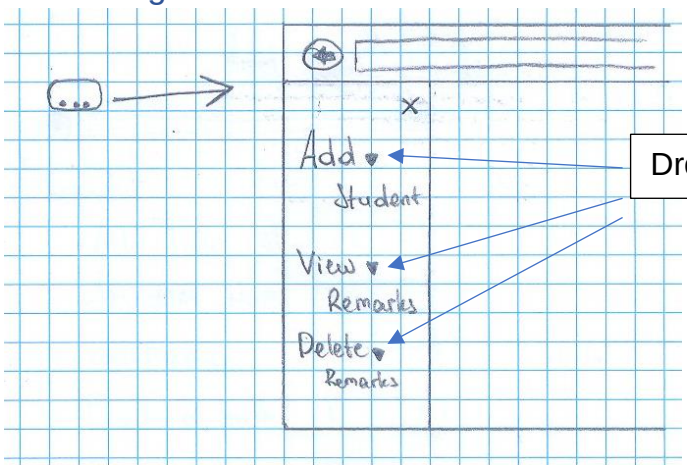
Add new remark



Dropdown menu will appear for “Subtopic” input box once any data is typed. Option will be based on what user typed

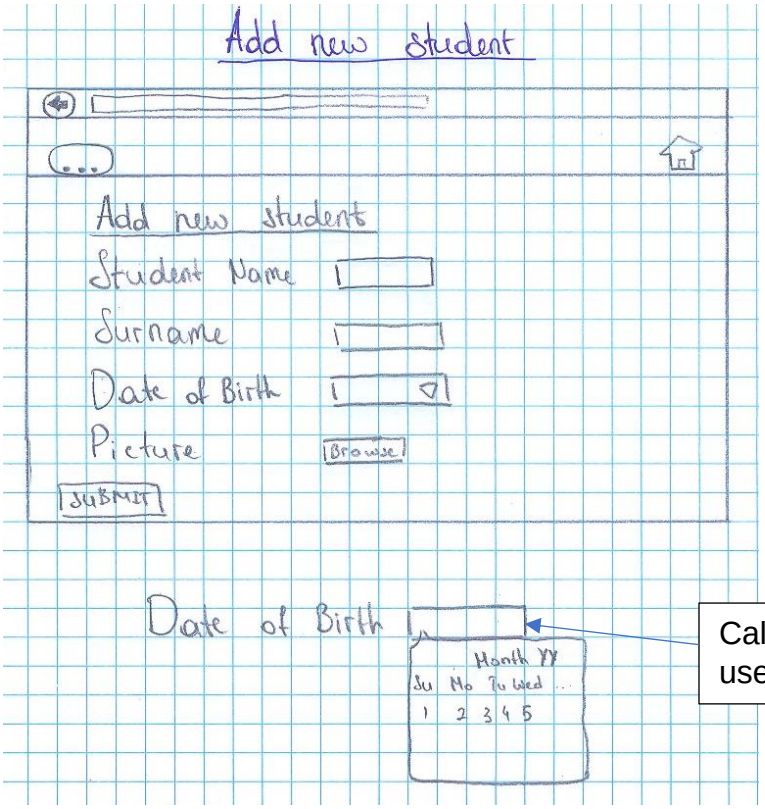
Dropdown menu to choose option

Side navigation bar



Dropdown menus

Add new student



See *Figure 1* for data processes needed for this page to load

Other data entry pages should have similar design.

Editing pages should have the same design, but predefined value must be present in input boxes

Calendar pops out when user clicks on the box

Tests required

Test type	Nature of test	Example
Ability to add information	After inputting information using data entry page, this information must be stored in the database	Add new student information using data entry page (**Note:** When a value for foreign key must be inputted, user inputs the values using reference names). Then, check using phpMyAdmin if new record with data submitted and unique ID was added to students_info table
Ability to view information	Table is generated on the view pages correctly shows information from the database	Go to the page which shows all units and see if the data displayed is valid by comparing it to the data in unit_info table in phpMyAdmin (**Note:** Foreign keys must not be displayed. Instead they should be used to retrieve data from other tables, and this data should be displayed)
Ability to delete information	User is able to delete information from the database using “delete buttons”	Delete a topic using the “delete button” on a view page. Check using phpMyAdmin if record of the chosen topic was deleted in topic_info table. Also check if records with topic ID (as a foreign key) equal to the ID of the topic deleted are also deleted.
Ability to edit information	Using “edit button”, user is able to go to editing page, which is similar to data entry page. On the editing page, user can edit the data of chosen information; and the data changed will update the database data	Go to the page which shows all remarks and choose one remark to be edited. Once chosen, press “edit button” and start editing information on the edit page. After finished, submit data and check if data was changed in sub_student table. (**Note:** The data must be

		updated. Consequently, no records should be deleted and no new records should be created)
Redirection to home page using top navigation bar	User can go to home page by clicking buttons on top navigation bar. This ability must be available on every page except a log in page	Go to all data entry pages and try clicking on buttons on top navigation bar. In all cases, the user must be redirected to home page
Side navigation bar	User can use side navigation bar to access data entry pages and viewing pages. This ability must be available on every page except a log in page.	Select 5 random pages and click on “View all topics” inside of the side navigation bar. In all cases, user must be redirected to the page which shows all topics.
Security of data access	In order to access any pages, user must first go through login page. On the login page, user must enter username and password that matches data from the database	Create a user in the “users” table. Go to the login page and enter random username and random password 10 times. Then enter correct username and password. In the first 10 cases user must be asked to enter username and password again, and on the 11th case, user must be redirected to the home page.