Intelligent Tutoring System

Final Report

Yip Kau Hei

Hong Kong Institute of Vocational Education

Final Year Project (ITE4116M)

Supervisor: Yan Yu Kan

Date: 02 April, 2023

Workload

Student	**Workload Percentage**
Yip Kau Hei (210392993)	100%

1 Abstract

From a student's point of view, the way most educators approach teaching is far from effective, in which teachers prepare one set of teaching materials, made as general as possible, and lecture them to students with the hope that most students can understand them. This is, in fact, a major problem that often gets overlooked by many. More talented students quickly master all of the materials, but are hindered by the slow course sequence that everyone must follow. On the contrary, less talented students have difficulty grasping some materials, while the class moves on to study other topics in order to catch up with the course sequence. Without choices, these troubled students either endeavor to study or are simply forced to leave behind uncomprehended topics. For many years, little effort has been put into solving this problem.

Hence, this project was initiated with the purpose of building a software tool to help students on their journey of learning.

2 Contents

3 Introduction

Traditionally, education has commonly been approached in a “one size fits all” way, namely that students in a classroom receive lectures from a teacher. This mode of learning is concerning, since students simply cannot all learn in the same way [1]–[3]. As a result, students better suited to such a learning method excel at the subjects, while others underperform. On top of that, the pandemic has transformed many classroom learning into distance learning. This reduced both student-student and student-teacher communication, which are both indispensable in learning.

In light of this, this project is initiated as an attempt to remediate the problem of “one size fits all” education by developing a digital tool for personalizing learning and promoting communication. Leveraging the fact that many classes are currently held online, the digital tool also aims to convert inquiries from students into a library of knowledge accessible by other students and to evaluate the quality of course materials by monitoring their usage.

One important thing to note is that, although this project is titled “Intelligent Tutoring System”, it is not the kind of system in the conventional sense that attempts to tutor students without human teacher’s intervention, or in other words to replace teachers [4]. Rather, it tries to enhance traditional education, whether online or offline, by providing tools that are easy to learn, easy to use, and more importantly, tools that can improve the learning experience and outcome of every single student. In short, the system intends to act as a facilitator of learning.

The aim of this report is to introduce the project, its system design, and evaluation. The remainder of this report is organized as follows: TODO

section 2 answers the question “How can Software Engineering techniques be used to develop software systems for supporting human activities?”, section 3 describes the problem this project attempts to solve, section 4 proposes solutions to the problems in section 3, section 5 lists the requirements of the system, section 6 presents the methodology adopted in developing the system, section 7 outlines the project plan.

4 Problem Analysis

4.1 Background

In the context of education, inefficient teaching and learning have always been a problem. In addition, the problem is exacerbated by the Coronavirus pandemic. This section provides a detailed analysis of problems this project attempts to rectify.

4.2 Problems

4.2.1 One size fits all teaching

The traditional “one size fits all” classroom teaching cannot accommodate the needs of every student since everyone has a different way of learning [3]. Under this mode of education, the syllabi are forced to be as general as possible in order to fit more students. This is obviously bad for the development of both well and poorly performing students, as they are all required to follow the pace of the curriculums. Consequently, teaching and learning are inefficient and ill optimized. Students who excel in STEM classes struggle with languages and history classes, while the others excel in non stem classes but struggle in math and science classes.

4.2.2 Difficulty in Assisting Students in Distance Learning

Education faced many challenges brought by covid [5]. One of which is providing assistance to students in an online learning environment.

4.2.3 Reduced Peer Learning

One problem of distance learning is that peer learning, which plays a crucial role in education [6], is reduced since students usually do not take initiative to discuss academic matters, which is even more true when they are separated by long distances.

4.2.4 Difficulty in Evaluating Quality of Courses Materials

Course materials evaluation, whether online or offline, are most conceivably done through surveys instead of from an objective point of view. Subjective evaluation could be useful in finding out the general opinion on the course materials. However, students likely do not take surveys seriously, especially when they consist of too many questions or they contain open ended questions. In addition to that surveys are often perceived as junk mails [7], this could in the end

produce inaccurate survey results. On the other hand, if they only ask a few multiple choice questions, the survey itself may not give much information.

5 Solutions

To address the above mentioned problems, a system with the following goals shall be developed.

5.1 Optimize Learning by Personalizing It

The system should help personalize learning so that every student can learn as much as possible.

In general, the system should automatically recommend learning resources to students. These resources may include reading materials, tutorial videos, and exercises. The resources are mainly prepared by teachers beforehand, while students are welcomed to suggest extra online resources to teachers. The system should recommend introductory resources before lessons, and recommend compulsory reading materials and exercises after each lesson.

To personalize learning for every student, the system should collect and analyze the performance of each student and identify his/her weakness and strength. After that, the system should suggest readings/exercises to reinforce or further expand students' knowledge on corresponding topics.

Further, to help particularly weak students, the system should allow teachers to easily identify students with poor performance and then intervene if it is necessary.

This approach of learning is founded on the concept of the zone of proximal development (see figure 1) introduced by psychologist Lev Vygotsky [8], [9]. By recommending suitable (more proximate to student's ability) learning materials to students, they are less likely to risk using materials that are beyond their capability of comprehension. This method emphasizes bridging student's knowledge with new knowledge and therefore should work better than using only standard learning materials since these could be too difficult for some students.

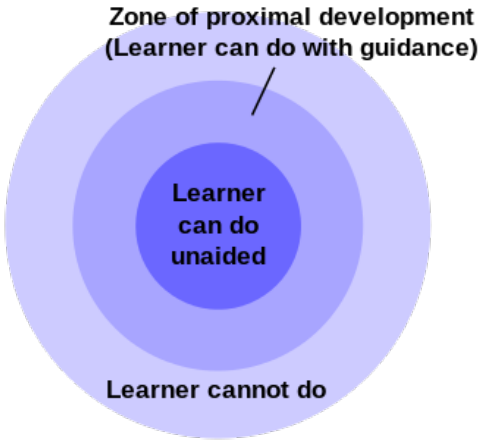


Figure 1: Zone of proximal development. Adapted from [7], CC0 1.0.

Nevertheless, there is no guarantee that the suggested learning resources are suitable for students, since they could still be too easy or hard for weak students. A remedy for this issue relies on providing extra learning materials (which teachers need to prepare) and student's ability to self-study.

While many believe that learning styles based instruction is more effective [10], studies have shown that this yielded no benefits [11], [12]. On the contrary, different modes of learning, namely concrete experience, reflective observation, abstract conceptualization, and active experimentation, should be engaged for one to have the most meaningful learning experience [13], [14]. Hence, this strategy does not lie in the scope of this solution.

5.2 Collecting Questions Asked by Students — QALib

To alleviate the problem that teachers have difficulties assisting students in a distance learning environment, the following solution is devised.

The system should provide a section for storing all questions asked by students and answered by teachers. All questions should be tagged for them to be categorized and searched more easily. The types of tags may include difficulties, subjects, and particular topics of study. The questions are first asked by students and viewed by responsible teachers (non responsible teachers will not see them). If the answer to the question already exists in the database, teachers can refer the student to the old answers instead of answering the question again. Otherwise, the teacher can answer the question. After that, the student or teacher can mark the question as resolved.

Teachers can then put tags on the question and submit it with the answer to the system. The system will then display these question-answer pairs in a public section so that students can search for them and view them. This collection of questions should greatly enhance students' ability to self-study, since all questions will be asked by students of previous years studying the same courses.

Although it may initially produce more work for the teachers, in the long run, students should gradually ask fewer questions as the database on asked questions expands. This can not only reduce teachers' workload, but also allow students to find solutions more quickly without having to wait for teachers' response every time they ask a question.

Regarding concerns on whether students can cheat on homeworks by using answers from the database, such an issue should not exist since every question is reviewed by teachers before they are posted. Teachers may also remove them if they wish to.

Moreover, to facilitate maintenance of the question-answer library, teachers should be able to view and modify them to improve the quality, integrity, and conciseness of the answers, as well as to mark answers as outdated and archive them if necessary.

The only drawback is that the goal to continuously improve the quality of the question-answer library inevitably requires extra work from teachers.

To improve brevity, the library of question-answer pairs is hereinafter called "QALib".

5.3 Facilitate Peer Communication

The system should provide a platform for academic discussion in which students can collaborate on tackling difficult problems in exercises.

One drawback is that students may prefer using other softwares for communication. Still, the goal of this solution is to encourage conversation between students. If that can be achieved, it does not matter what software they use afterward.

5.4 Evaluate Quality of Course Materials by Assessing their Usage

The system should record and display usages of course materials in the form of plots for teachers to evaluate their quality.

The usage data may include students' overall performance (average score), average time spent on using the materials, and percentage of students who used the materials. Teachers may then gain insights regarding their quality and then improve them. For example, if some course materials have short view time, it could indicate that they are too easy for most students. Teachers may then consider making them more concise or removing them entirely.

With the usage data, teachers may also identify students with special needs and provide them appropriate guidance. Since the system can display course material usage of every student, teachers can see whether students with poor grades have used the tutorials. For those who do not utilize learning materials and end up receiving poor grades, teachers may consider encouraging them to more actively participate in learning. For those who study most of the learning materials but still get poor grades, teachers may give them special help since they could be using wrong study methods.

In sum, this solution gives an objective evaluation of the course materials while not having any significant drawbacks.

6 System Requirements

6.1 Scope of the System

In general, to act as a lightweight tool for assisting the process of learning. Specifically, to provide functions to manage and display information in a way that guide and encourage students to continue learning with less effort, but greater outcome.

6.2 Functional Requirements

In this section, “FR” and “UC” will be used as abbreviations of “functional requirement” and “use case” respectively. The “FR ID” and “UC ID” will be used for cross reference. See data dictionary for the definition of QALib.

6.2.1 Administrator

Table: Functional requirements of administrators

FR ID	Description	UC ID
FR-A1	The system shall allow administrators to manage (create/modify/remove) student’s/teacher’s accounts.	UC-A1
FR-A2	The system shall allow administrators to manage (create/modify/remove/mark as outdated/archive/tag) entries in QALib.	UC-T2, UC-T3
FR-A3	The system shall allow administrators to log in their accounts.	UC-G1

6.2.2 Teacher

Table: Functional requirements of teachers

FR ID	Description	UC ID
FR-T1	The system shall allow teachers to manage (post/modify/remove/tag) learning resources (readings, videos, exercises).	UC-T1
FR-T2	The system shall allow teachers to see students’ scores on exercises.	UC-T5
FR-T3	The system shall allow teachers to see students’ learning resource usages.	UC-T6
FR-T4	The system shall allow teachers to manage (add/modify/remove) prerequisites to learning resources.	UC-T4
FR-T5	The system shall allow teachers to manage (create/modify/remove/mark as outdated/archive/tag) entries in QALib.	UC-T2, UC-T3

FR-T6	The system shall allow teachers to log in their accounts.	UC-G1

6.2.3 Student

Table: Functional requirements of students

FR ID	Description	UC ID
FR-S1	The system shall allow students to use any learning resources.	UC-S1
FR-S2	The system shall allow students to give feedback on learning resources.	UC-S3
FR-S3	The system shall allow students to receive recommendations of learning resources.	UC-S2
FR-S4	The system shall allow students to suggest learning resources to teachers.	UC-S6
FR-S5	The system shall allow students to view all (non-archived) entries in QALib.	UC-S4
FR-S6	The system shall allow students to search for entries in QALib.	UC-S7
FR-S7	The system shall allow students to log in their accounts.	UC-G1

6.3 Nonfunctional Requirements

6.3.1 Reliability Requirements

- The system shall allow staff members to access the application 98% of the time without failure.
- The system shall have a probability of failure of less than 0.0001% (1 out of 1,000,000) when the system undergoes a backend operation.
- The system update process shall roll back all related previous states when any update fails.

6.3.2 Performance Requirements

- At least 20 percent of the processor capacity and storage space available to the system shall be unused at peak load seasonal periods.
- 95% of all pages respond in 8 seconds or less

6.3.3 Client System Requirement

The following is the system requirements for clients' computers.

Table : System requirements for clients' computers.

Operating System	Windows XP
CPU	Core 2 Duo or Athlon X2 at 2.4 GHz
RAM	2 GB RAM
Storage	8 GB of free space

7 System Design

7.1 Architecture

The system will adopt the three tiered architecture.

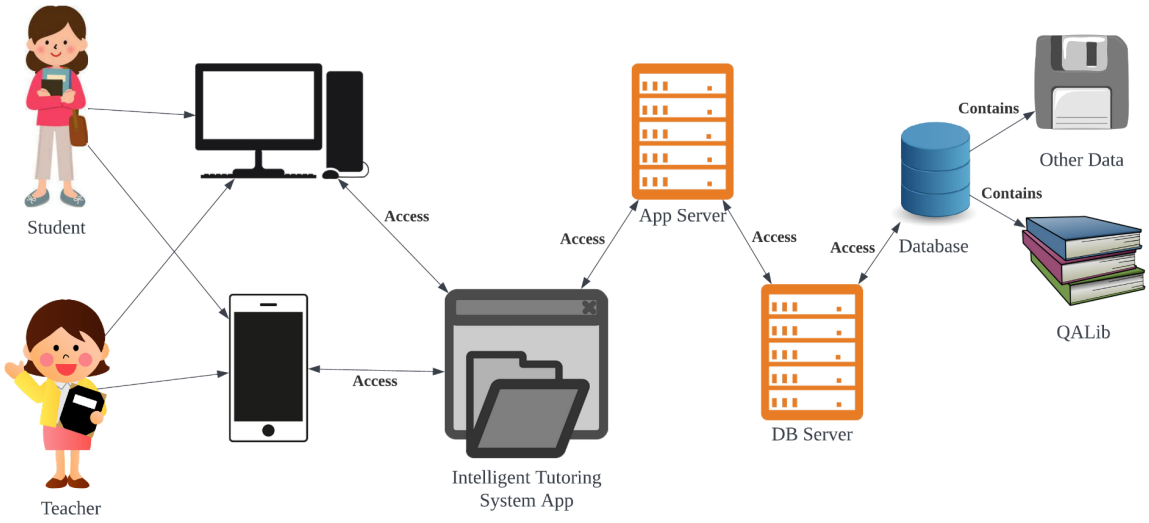


Figure 1: System architecture. Made in LucidChart. All PNGs have CC0 1.0.

7.2 Use Case Analysis

7.2.1 Actor Description

The following is a description of the three actors in the system.

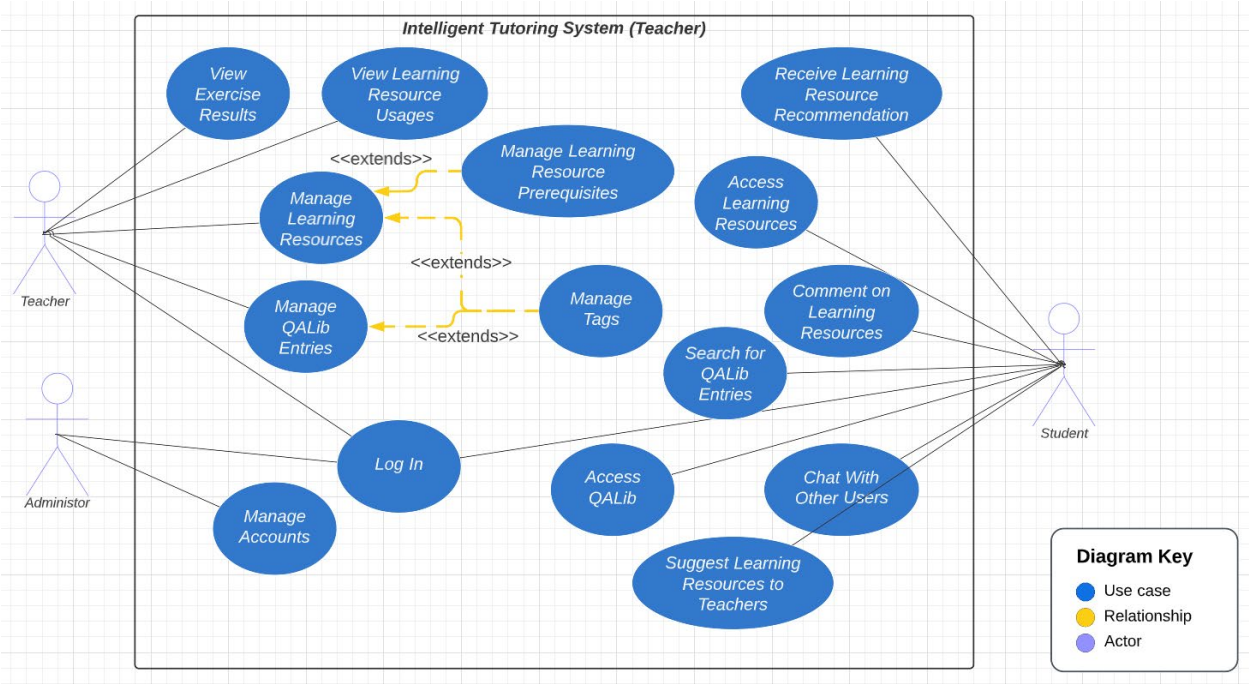
Table: Actor Description

Actor	**Description**
Administrator	Primarily responsible for managing accounts.
Teacher	One that teaches students. Responsible for preparing teaching materials, monitoring students' learning progresses, etc.
Student	One that is taught by teachers. Responsible for using teaching materials for learning.

7.2.2 Use Case Diagram & Description

In the this section:

- All use cases assume the users have logged in. This is done to reduce complexity and to improve readability in the diagrams.
- "Administrator" is abbreviated as "admin" for brevity.



Teacher Use Cases

Table: Teacher Use Case "Manage Learning Resources"

Use Case	Manage Learning Resources
Use Case ID	UC-T1
Description	The teacher manages (post/modify/remove/tag/manage prerequisites) a learning resource (reading, video, exercise).
Flow (post)	1. The teacher opens the learning resource control panel and clicks the Create Post button. 2. The system shows the page for entering information for the learning

	resource, including the title, description, media file, etc. 3. The teacher provides all required information and clicks Post. 4. The system creates the post.
Flow (modify)	1. The teacher opens the learning resource control panel. 2. The system shows a list of all learning resources. 3. The teacher searches for a learning resource using keywords of the title, sort by date created, and filter by tags and media type. 4. The system displays the search results. 5. The teacher clicks a learning resource. 6. The system shows the details of that learning resource. 7. The teacher clicks the Modify button. 8. The system shows a page for entering new information for that learning resource. 9. The teacher fills the new information and clicks Confirm. 10. The system updates the learning resource and returns to the learning resource list.
Flow (remove)	1-6. Same as above. 7. The teacher clicks the Remove button. 8. The system asks for confirmation. 9. The teacher confirms. 10. The system removes the learning resource and returns to the learning resource list.
Flow (tag)	See UC-T3
Flow (manage prerequisites)	See UC-T4
Postconditions	The entry is created/modified/removed/archive/marked as outdated/tagged.

Table: Teacher Use Case “Manage QALib Entries”

Use Case	Manage QALib Entries
Use Case ID	UC-T2
Description	The teacher manages (create/modify/remove/mark as outdated/archive/tag) an entry in QALib.
Flow (create)	1. The teacher opens the QALib control panel and clicks New Entry. 2. The system shows a page for entering information of the new entry. 3. The teacher fills the required information including the question, the corresponding answers, and optionally adds tags to the entry. Finally, he clicks Create.

	4. The system shows a completion message and then shows the newly created entry.
Flow (modify)	1. The teacher opens the QALib control panel. 2. The system shows a list of all entries. 3. The teacher searches for an entry using keywords in the question, sort by date created, and filters by tags. 4. The system displays the search results. 5. The teacher clicks an entry. 6. The system shows the details of that entry. 7. The teacher clicks the Modify Entry button. 8. The system shows a page for entering new information for that entry. 9. The teacher enters the new information and clicks Confirm. 10. The system updates the entry and returns to the QALib control panel.
Flow (remove)	1-6. Same as above. 7. The teacher clicks Remove Entry button. 8. The system asks for confirmation. 9. The teacher confirms. 10. The system removes the entry and returns to the QALib control panel.
Flow (mark as outdated)	1-6. Same as above. 7. The teacher clicks the Mark As Outdated button. 8. The system asks for confirmation. 9. The teacher confirms. 10. The system marks the entry as outdated whereafter the entry is sorted in less priority in search results (the non-outdated are shown first) and students can see the words “Outdated Entry”. Afterward the system returns to the QALib control panel.
Flow (archive)	1-6. Same as above. 7. The teacher clicks the Archive Entry button. 8. The system asks for confirmation. 9. The teacher confirms. 10. The system archives the entry whereafter the entry will not be shown in ordinary searches. It will only be shown when the user selects “show archived entries” in the advanced search box. Afterward the system returns to the QALib control panel.
Flow (tag)	See UC-T3
Postconditions	The entry is created/modified/removed/archive/marked as outdated/tagged.

Table: Teacher Use Case “Manage Tags”

Use Case	Manage Tags

Use Case ID	UC-T3
Description	The teacher manages (add/modify/remove) a tag of a learning resource/QALib entry.
Flow (add)	1. The teacher navigates to a learning resource/QALib entry. 2. The teacher clicks the tag section. 3. The teacher types in the name of a tag. 4. If such a tag already exists, the system shows the tag for the teacher to select. If not, a new tag with the name the user types will be created upon pressing Enter or Create Tag. 5. The teacher selects/creates the tag. He continues the process of creating tags (8-9) until he is done. 6. The teacher clicks Save. 7. The system updates the learning resource/entry.
Flow (modify)	1-2. Same as above. 3. The teacher selects an existing tag. He selects the display color of the tag and changes the name of the tag. He then clicks Save or presses Enter. 4. The system saves the changes.
Flow (remove)	1-2. Same as above. 3. The teacher selects an existing tag and clicks the Remove button or presses Delete. 4. The system asks for confirmation. 5. The teacher confirms. 6. The system removes the tag.
Postconditions	The learning resource/entry is tagged or the tag is modified or the tag is removed. The learning resource/entry can be searched with the added tags.

Table: Teacher Use Case “Manage Learning Resources Prerequisites”

Use Case	Manage Learning Resources Prerequisites
Use Case ID	UC-T4
Description	The teacher manages (add/remove) the prerequisites of a learning resource. The prerequisite can be the code of a course, name of the tag of a learning resource, or an arbitrary subject.
Flow (add)	1. The teacher navigates to a learning resource. 2. The teacher navigates to the prerequisite section 3. The teacher types the name of the prerequisite subject/course. 4. The system searches for the subject/course using the typed keywords. If

	it exists, it is shown for the teacher to select. Otherwise, the system will create a subject with the typed name upon pressing Enter. 5. The teacher selects the subject/course or creates the subject. 6. The teacher clicks Save. 7. The system saves the changes.
Flow (remove)	1-2. Same as above. 3. The teacher selects an existing prerequisite and clicks the Remove button or presses Delete. 4. The system asks for confirmation. 5. The teacher confirms. 6. The system removes the prerequisite.
Postconditions	Connections between learning resources are established using prerequisite relationships. Students can navigate from subject to subject or from course to course with these connections. Students are allowed to see the course materials even if they do not meet the prerequisite. They are however advised by the system to first learn the prerequisite materials.

Table: Teacher Use Case "View Exercise Results"

Use Case	View Exercise Results
Use Case ID	UC-T5
Description	The teacher views the result of an exercise, displaying a student's scores on every question.
Flow	1. The teacher navigates to the details page of an exercise and clicks the Attempts button. 2. The system shows all students who have attempted and submitted the exercise. 3. The teacher clicks on one of the students. 4. The system shows all attempts of that student. 5. The teacher clicks on an attempt record. 6. The system shows the questions along with the answers in that attempt record, the scores of each question, the total score, and whether the student passes the exercise (if a passing mark is set).

Table: Teacher Use Case "View Learning Resources Usages"

Use Case	View Learning Resources Usages
Use Case ID	UC-T6

Description	The teacher views the usage of a learning resource, which is shown in minutes viewed, which is the interval from when the student began using the learning resource and when the student finished using it.
Flow (add)	1. The teacher navigates to the details page of a learning resource. 2. The system shows the time spent on using the learning resource in minutes.

Student Use Cases

Table: Student Use Case "Access Learning Resources"

Use Case	Assess Learning Resources
Use Case ID	UC-S1
Description	The student accesses a learning resource.
Flow	1. Navigates to the learning resource center. 2. The system shows a list of all available learning resources. 3. The student clicks one learning resource. 4. The system opens the learning resource.
Postconditions	The clicked learning resource is opened.
Alternative Flows & Exceptions	After step 3, the student can optionally search for the desired learning resource by filtering tags, sorting, and searching by the search bar..

Table: Student Use Case "Receive Learning Resource Recommendation"

Use Case	Receive Learning Resource Recommendation
Use Case ID	UC-S2
Description	The student receives a recommendation of a learning resource the student might find useful after finishing accessing a learning resource.
Flow	1. The student finishes using a learning resource. 2. The system shows a page displaying the result of using the learning resource along with a recommendation of another learning resource.
Alternative Flows & Exceptions	Recommendations may not show in occasions when the learning resource the student has done is already comprehensive on its own and is not required by any other subjects as a prerequisite.

Table: Student Use Case “Comment on Learning Resources”

Use Case	Comment on Learning Resources
Use Case ID	UC-S3
Description	The student comments on a learning resource regarding the experience of using that resource.
Flow	1. The student finishes using a learning resource. 2. The system shows a page displaying the result of using the learning resource along with a field for writing feedback of the learning resource. 3. The student writes the feedback and clicks Submit. 4. The system saves the feedback, which is visible to everyone who has access to the learning resource.

Table: Student Use Case “Access QALib”

Use Case	Access QALib
Use Case ID	UC-S4
Description	The student accesses the QALib entries.
Flow	1. The student navigates to the QALib panel. 2. The system shows a list of all available QALib entries. 3. The student clicks on one entry. 4. The system shows the detailed page of that entry.
Alternative Flows & Exceptions	After step 2, the student can optionally search for the desired entry (see UC-S7).

Table: Student Use Case “Suggest Learning Resources to Teachers”

Use Case	Suggest Learning Resources to Teachers
Use Case ID	UC-S6
Description	The student suggests a learning resource to teachers.
Flow	1. The student navigates to the learning resource center. 2. The system shows a button for suggesting learning resources. 3. The student clicks the button. 4. The system shows a form for filling information regarding that learning resource.

	5. The student fills all necessary information and uploads necessary documents and clicks Submit. 6. The system receives the form which is visible to all teachers.
Postconditions	After submitting the form, teachers can view the form and decide whether they accept it or not. If the teacher accepts it, he/she may create a new learning resource (see UC-T1) based on the information given by the form. Otherwise, the teacher can click the Reject button, whereafter the student will receive notification regarding the rejection.

Table: Student Use Case "Search for QALib Entries"

Use Case	Search for QALib Entries
Use Case ID	UC-S7
Description	The student searches for a desired entry by filtering tags, sorting, and searching by the search bar.
Flow	1. The student navigates to the QALib panel. 2. The system shows a list of all available QALib entries. 3. The student clicks on some tag(s) and enters some keywords into the search bar to filter the search result. 4. The student uses the sort function to sort the search result. 5. The system shows the search results according to the search parameters.

Administrator Use Cases

Table: Administrator Use Case "Manage Accounts"

Use Case	Manage Accounts
Use Case ID	UC-A1
Description	The admin creates/modifies/removes an account of a student/teacher/admin.
Flow (create account)	1. The admin opens the account management panel and clicks Create Account. 2. The system shows a page for entering information for the new account. 3. The admin fills all required information and clicks the submit button. 4. The system shows a completion message and returns to the account management panel.
Flow (modify account)	1. The admin opens the account management panel. 2. The system shows a list of all accounts.

	3. The admin searches for an account using the account user's name/login username. 4. The system shows the search results. 5. The admin clicks an account. 6. The system shows the details of the account. 7. The admin clicks Modify Account. 8. The system shows a panel for entering now account info. 9. The admin enters new info and clicks Confirm. 10. The system updates the account info and returns to the account list.
Flow (remove account)	1-6. Same as Modify Account. 7. The admin clicks remove account. 8. The system asks for confirmation. 9. The admin confirms. 10. The system removes the account and returns to the account list.
Postconditions	The account is created/modified/removed.
Alternative Flows & Exceptions	When the admin enters the account information during account creation/modification, the system does input checking.

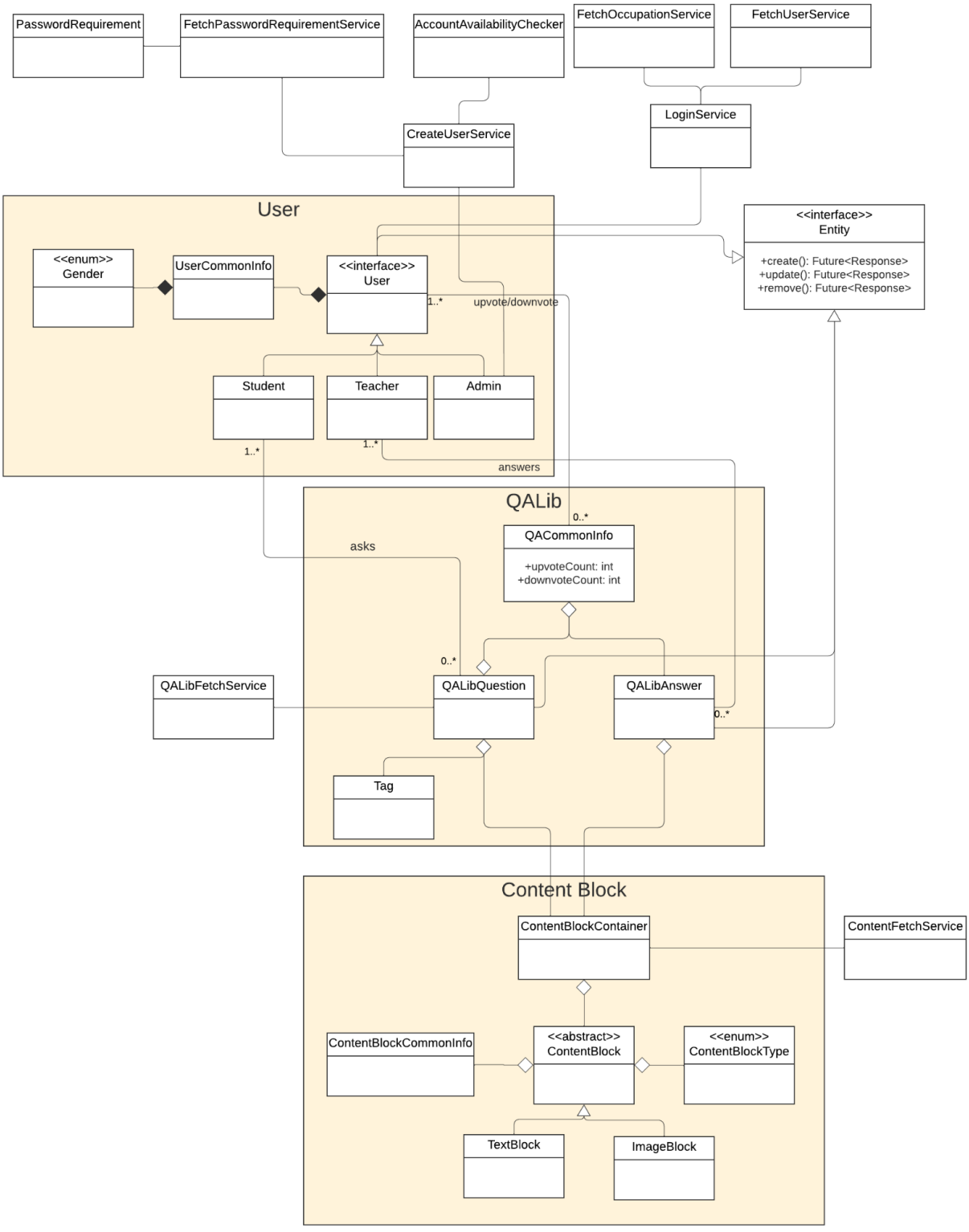
Generic Use Cases

Table: Generic Use Case "Log In"

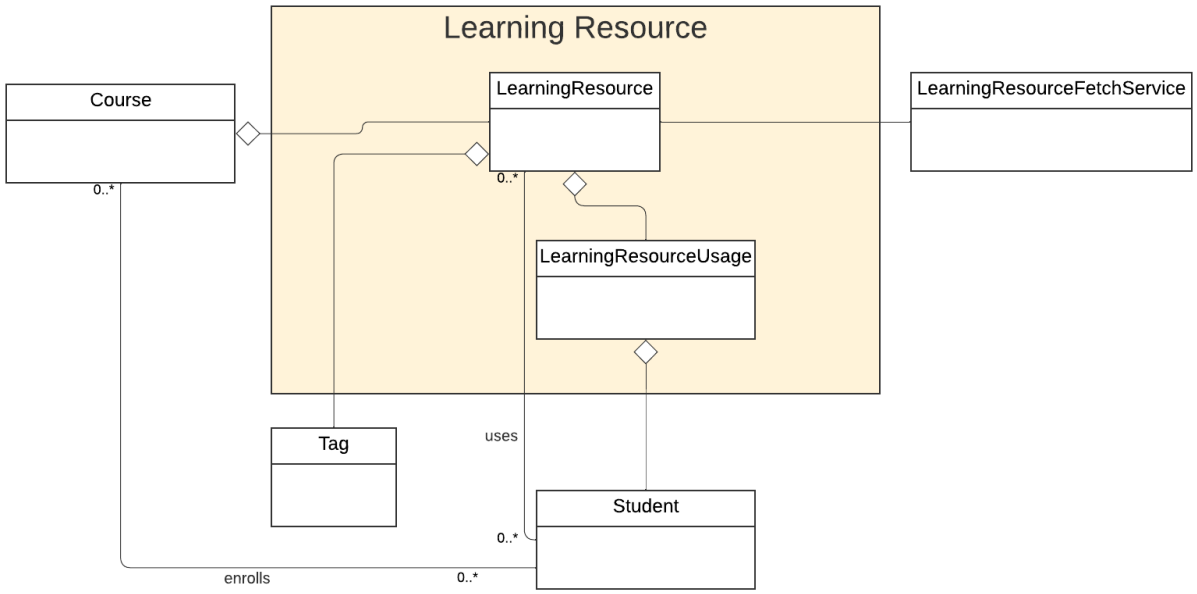
Use Case	Log In
Use Case ID	UC-G1
Description	The user logs into the system using his/her login ID/username and password.
Flow	1. The user enters his/her username and password into the input fields and then hits Enter or clicks the Login button. 2. The system shows the post-login page if the login credentials were correct. Otherwise, the system asks him/her to re-enter the login credentials.
Postconditions	The user has successfully logged in and is able to use all system functionalities he/she has access to.

7.3 Class Diagram

7.3.1 Diagram 1

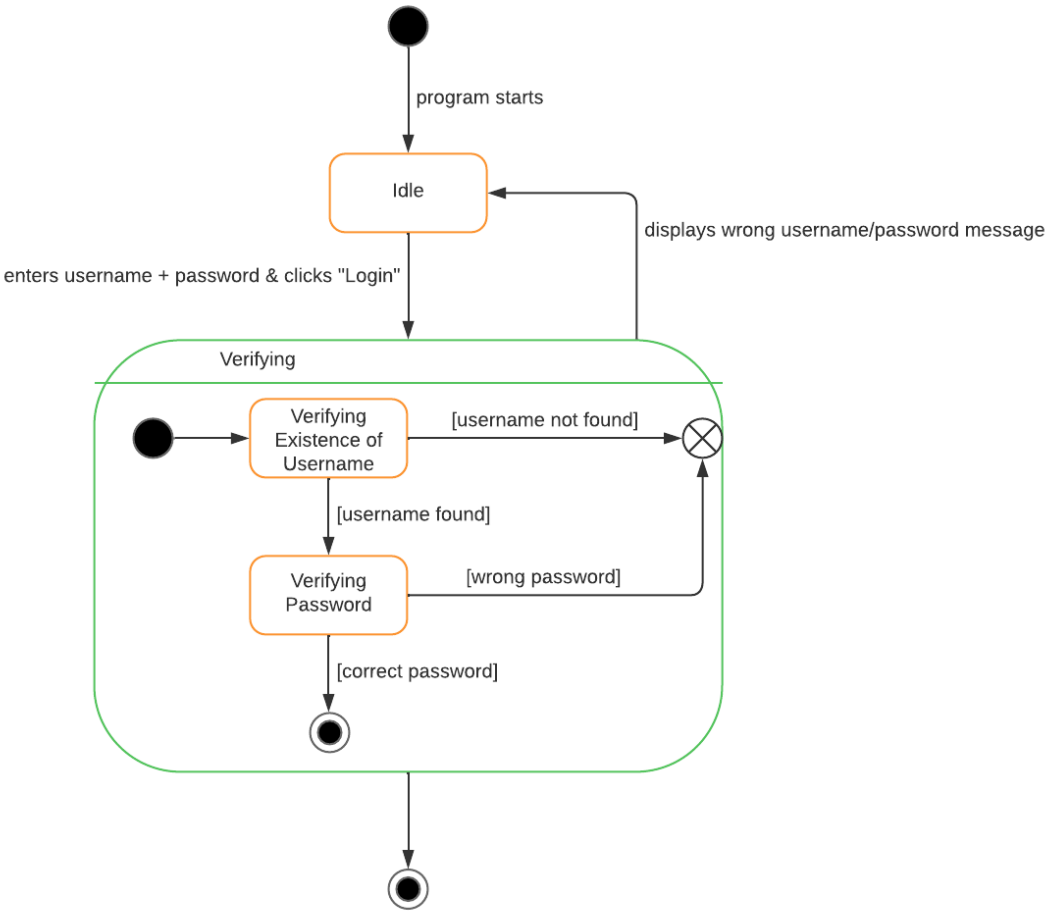


7.3.2 Diagram 2

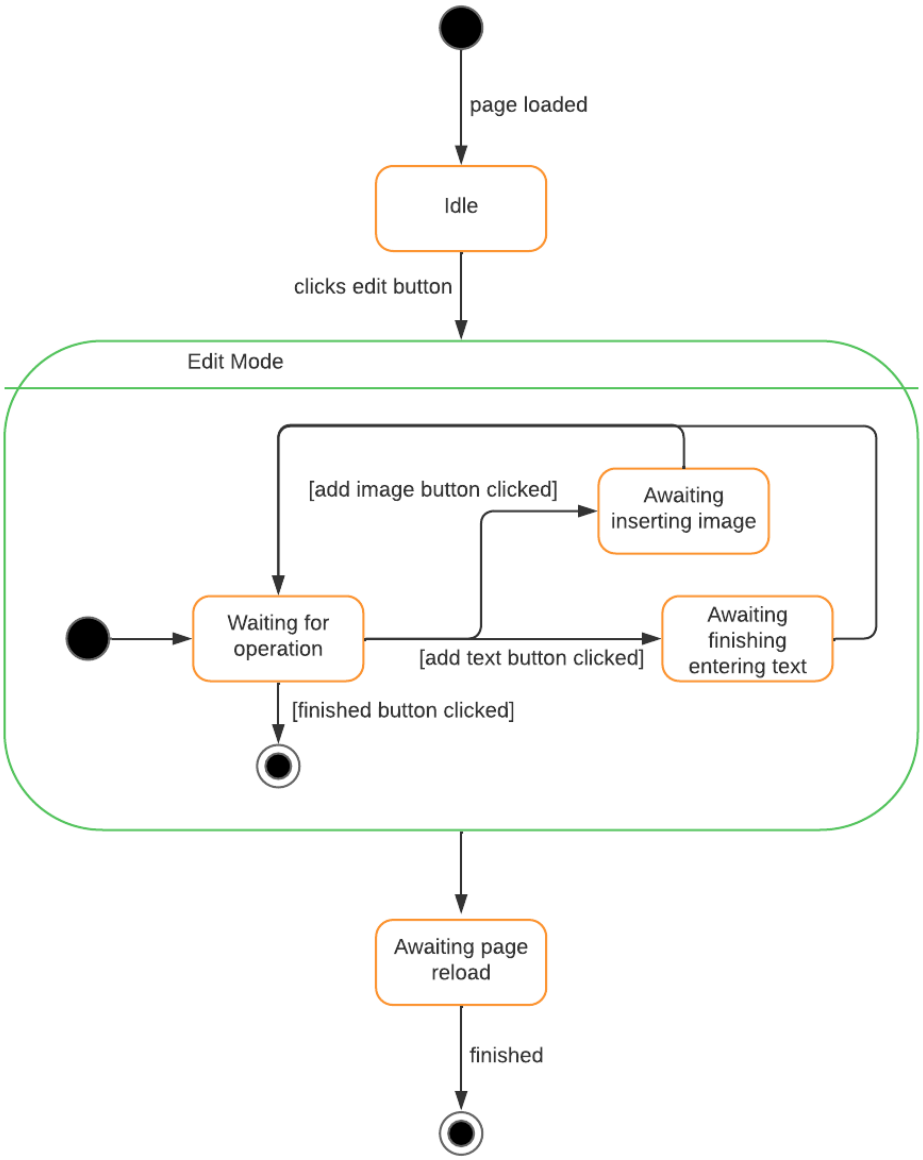


7.4 State Diagrams

7.4.1 Login Controller

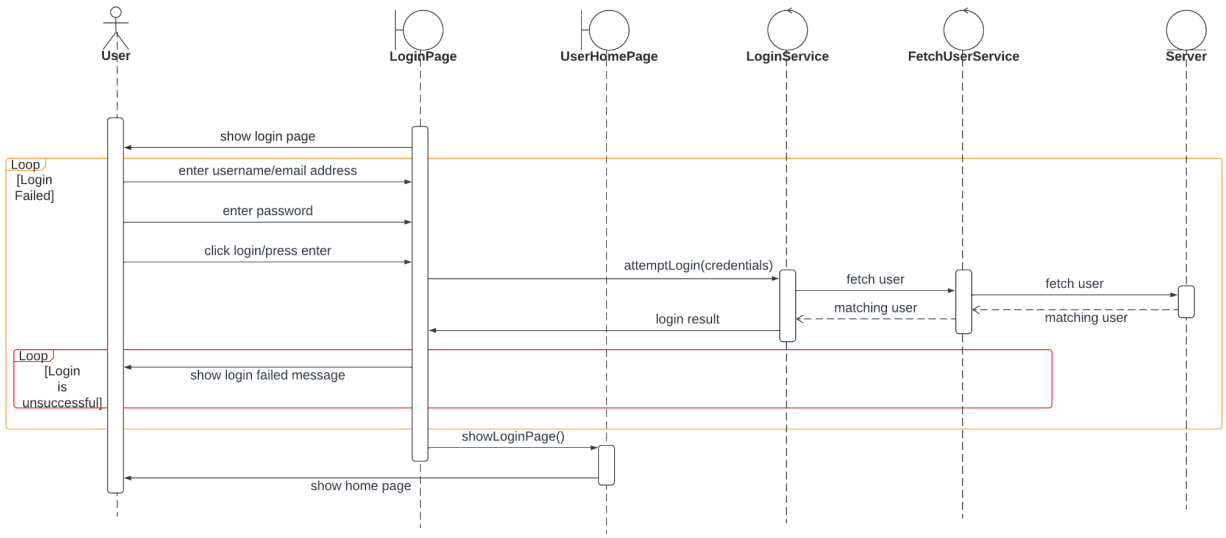


7.4.2 Qalib Entry Editing Controller

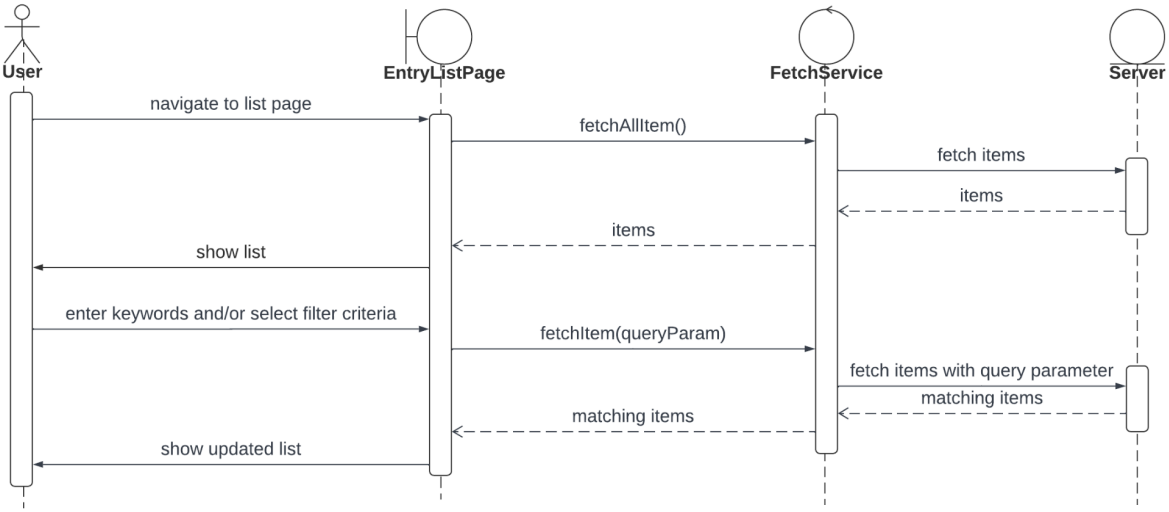


7.5 Sequence Diagrams

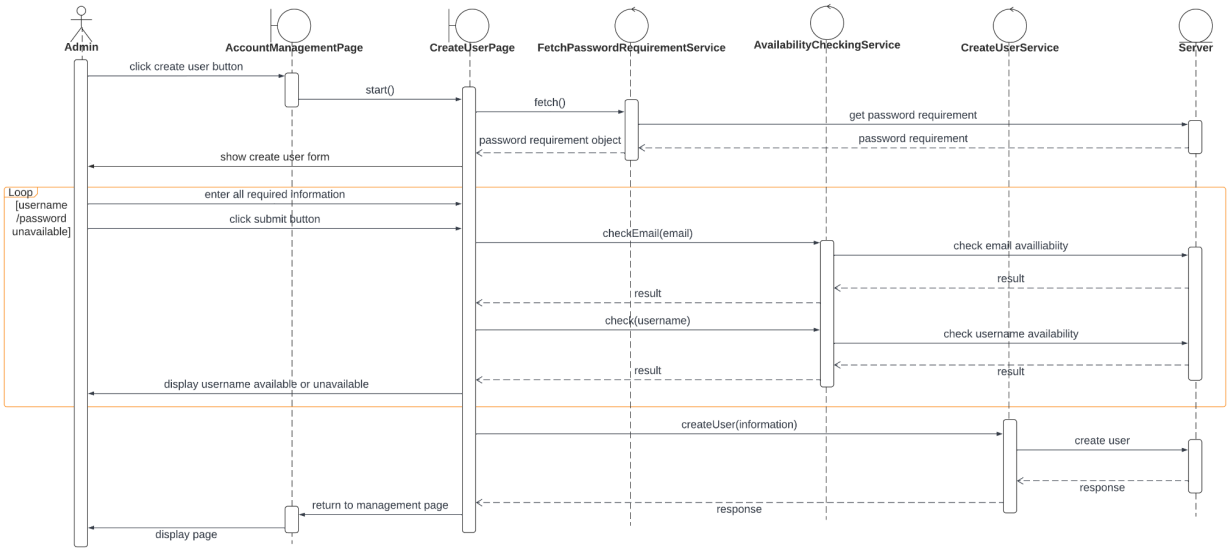
7.5.1 Login



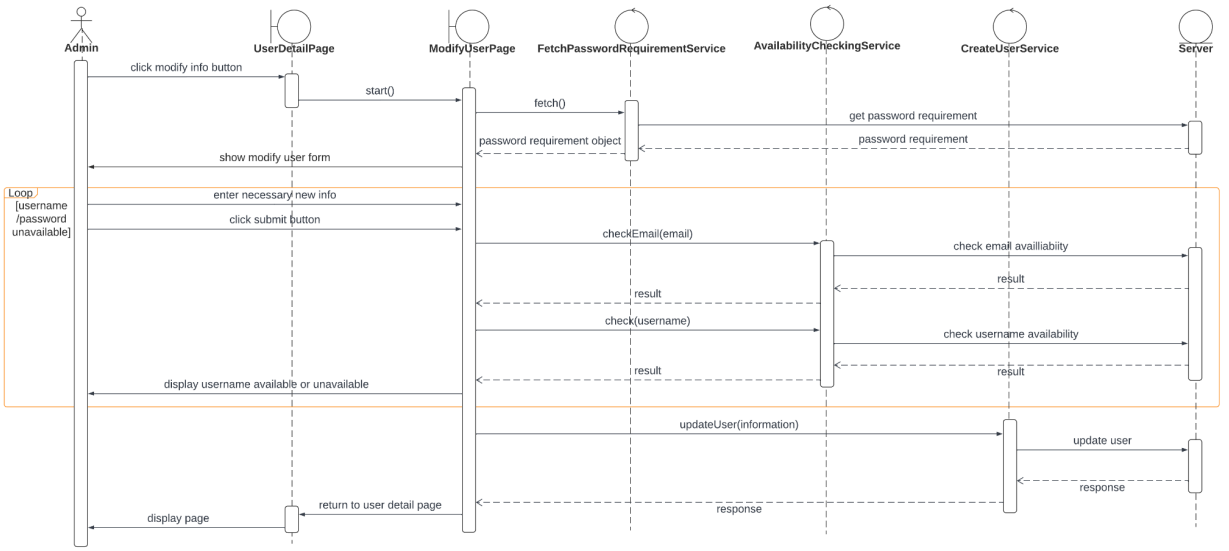
7.5.2 Search/Filter for Any Entries



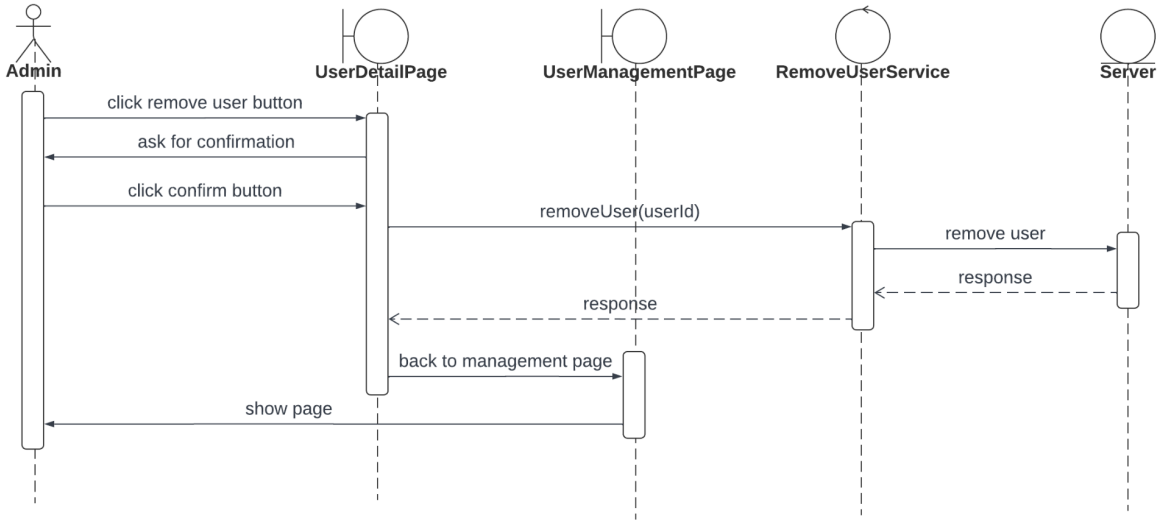
7.5.3 Create User (Admin)



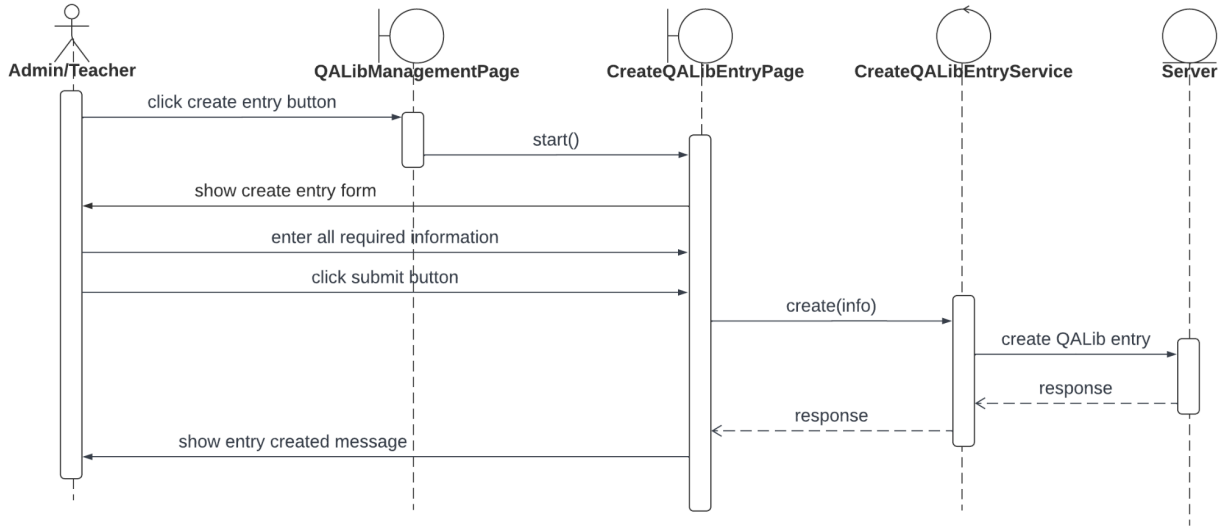
7.5.4 Modify User Account (Admin)



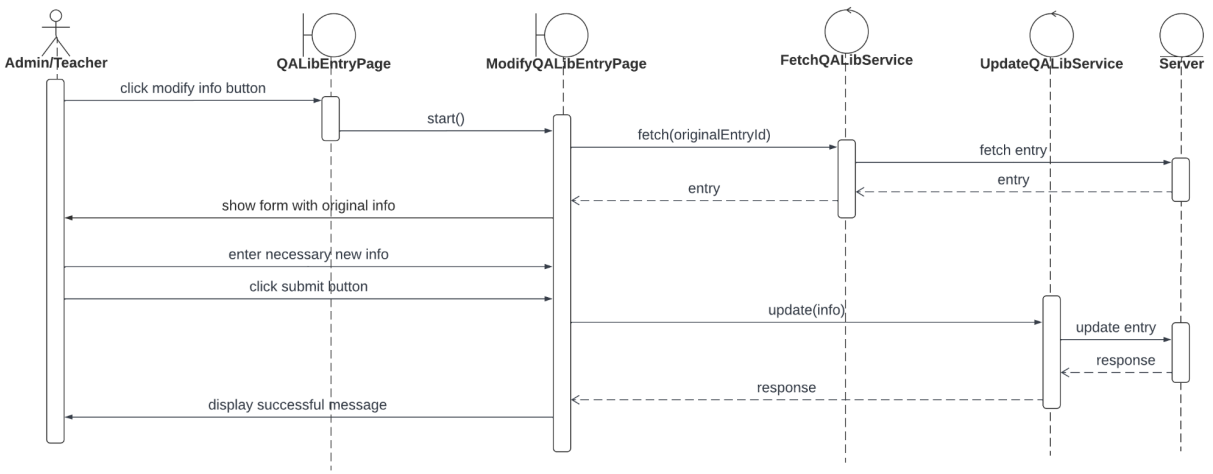
7.5.5 Remove User Account (Admin)



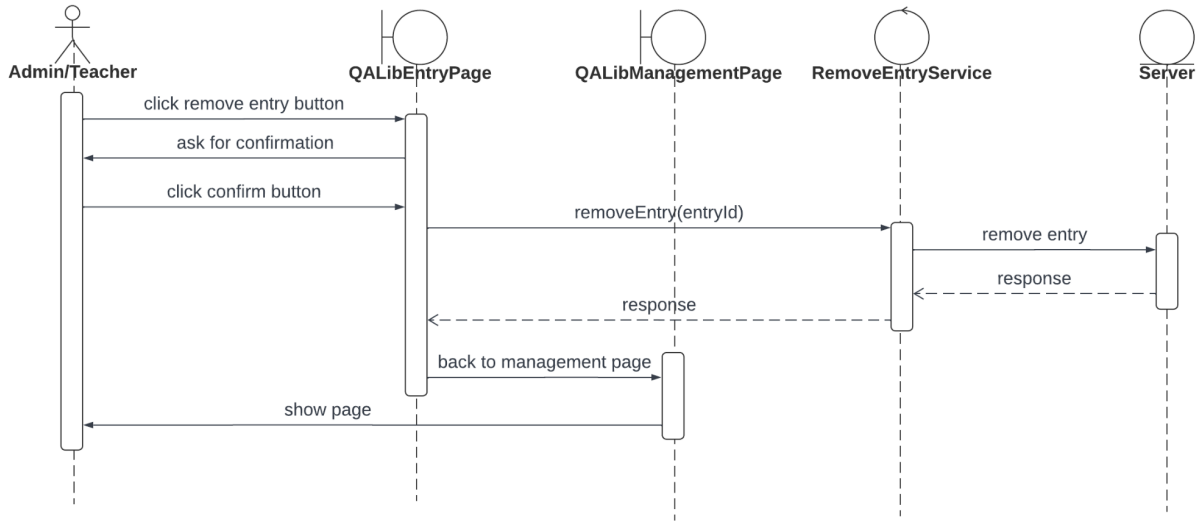
7.5.6 Create QALib Entry (Admin/Teacher)



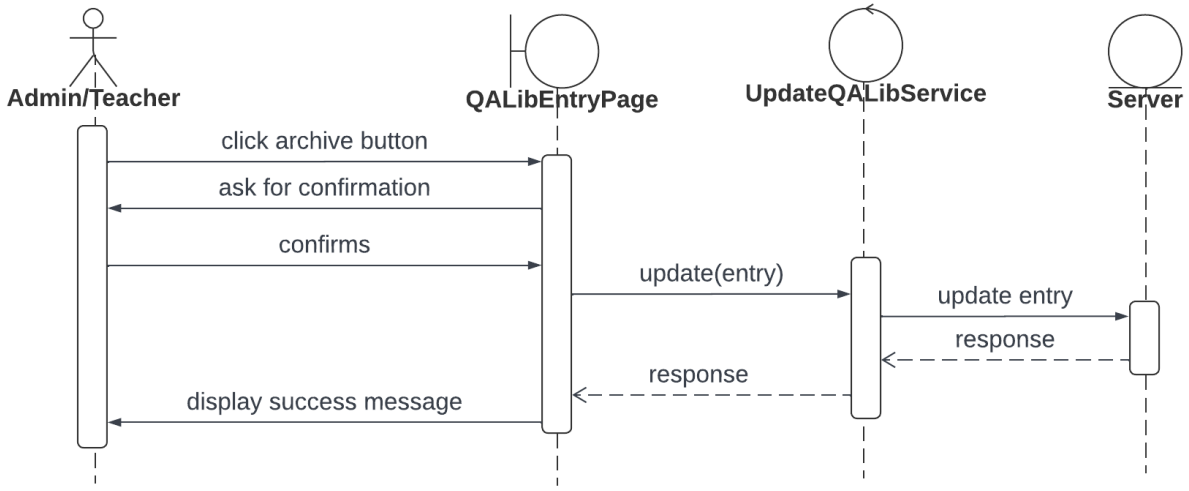
7.5.7 Modify QALib Entry (Admin/Teacher)



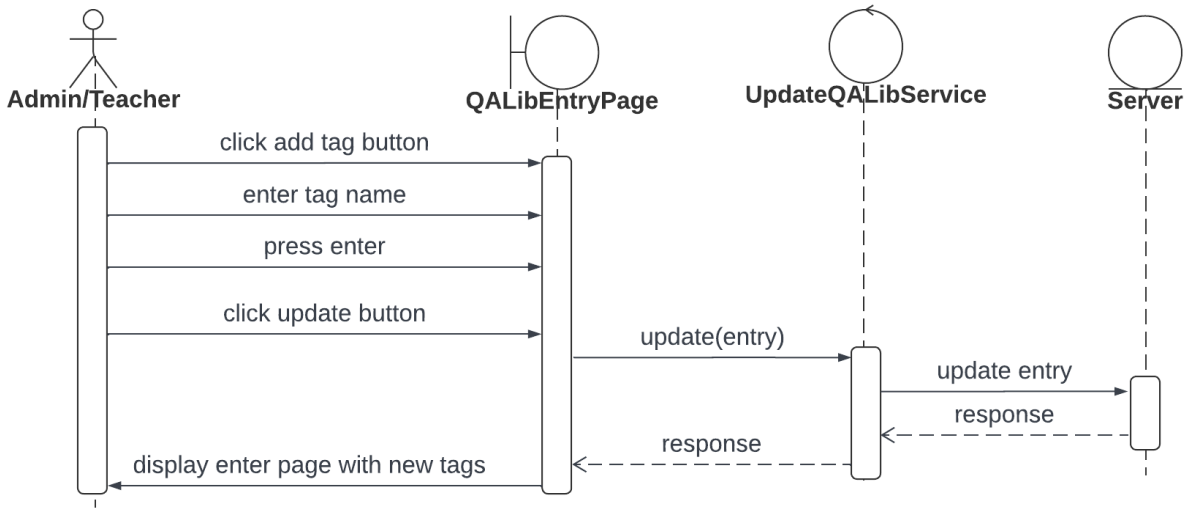
7.5.8 Remove QALib Entry (Admin/Teacher)



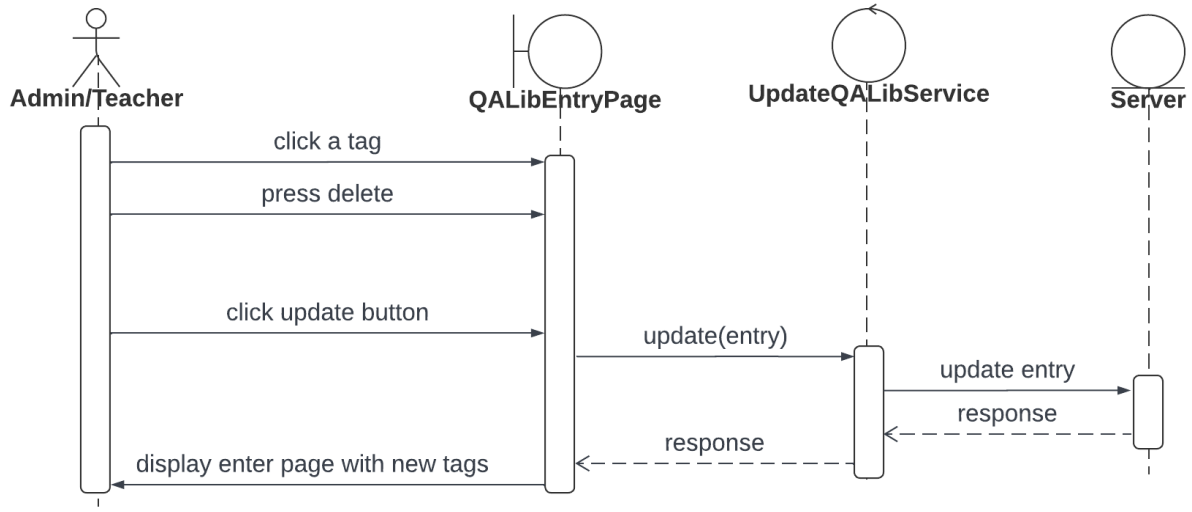
7.5.9 Archive QALib Entry (Admin/Teacher)



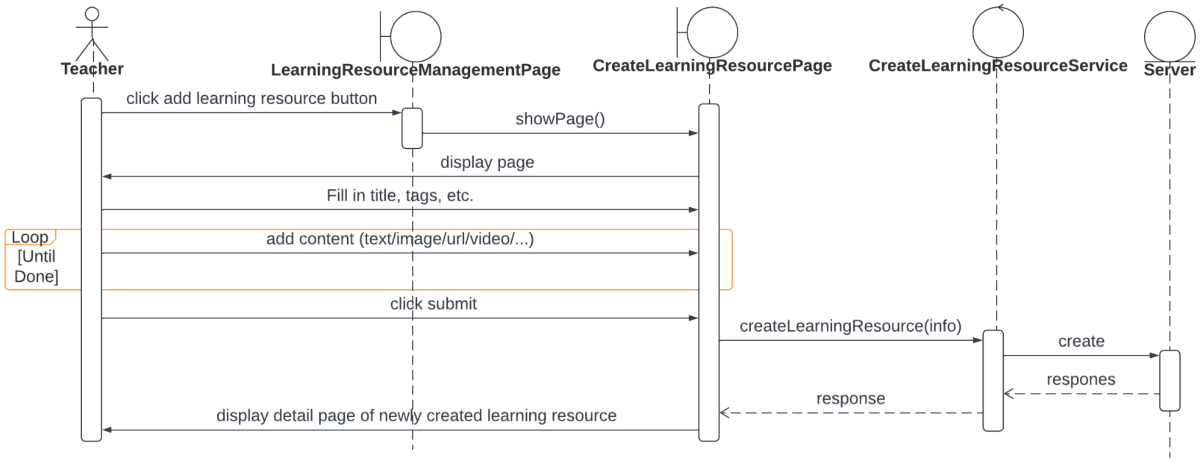
7.5.10 Add Tags for QALib Entry (Admin/Teacher)



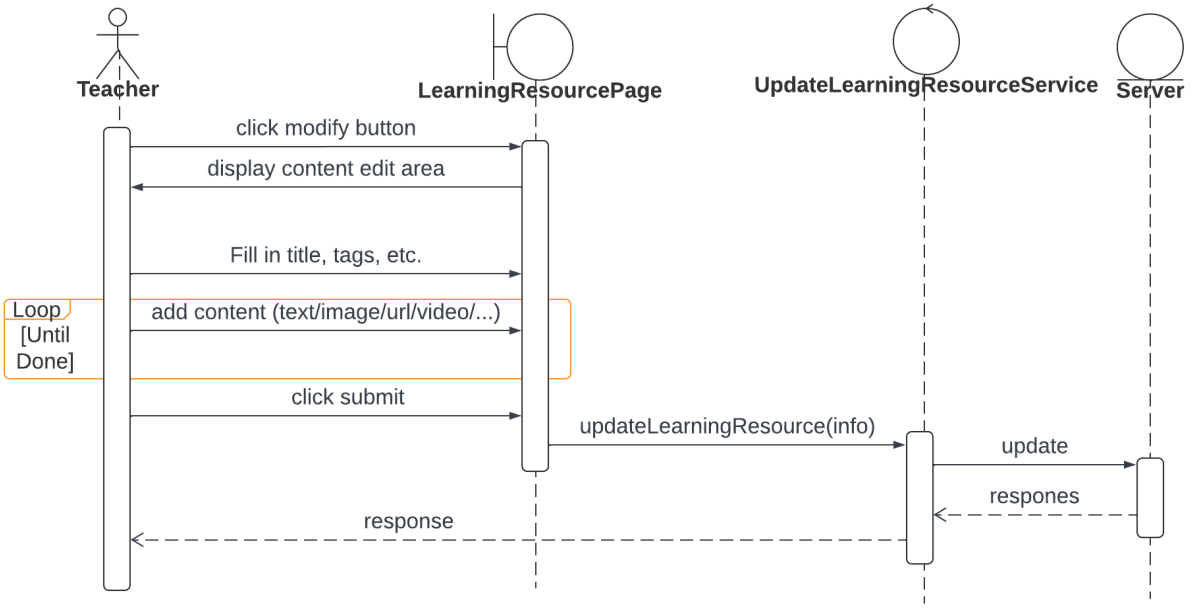
7.5.11 RemoveTags for QALib Entry (Admin/Teacher)



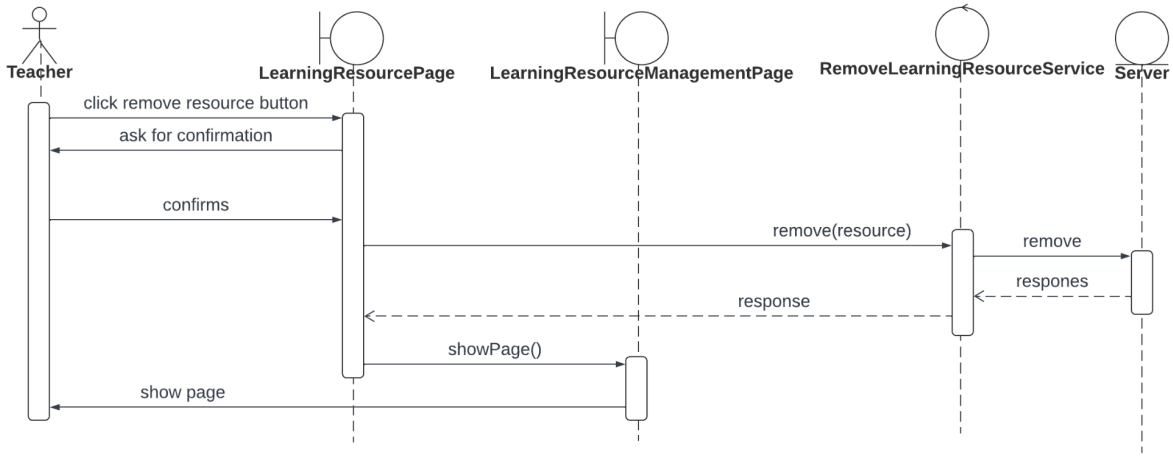
7.5.12 Post New Learning Resource (Teacher)



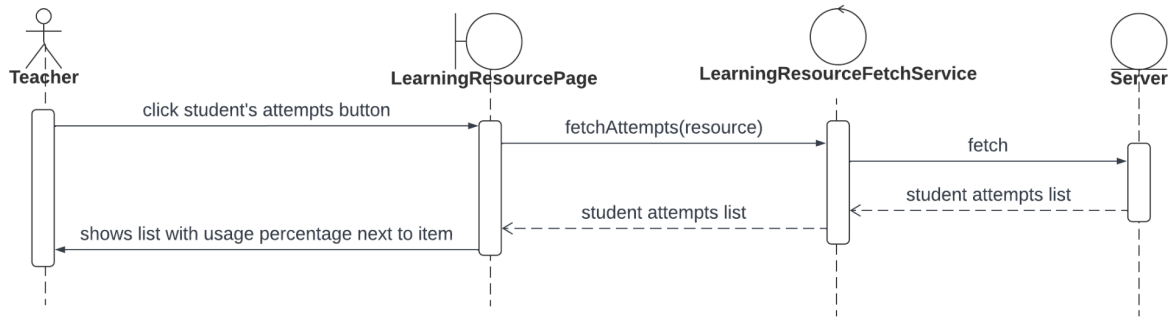
7.5.13 Modify Learning Resource (Teacher)



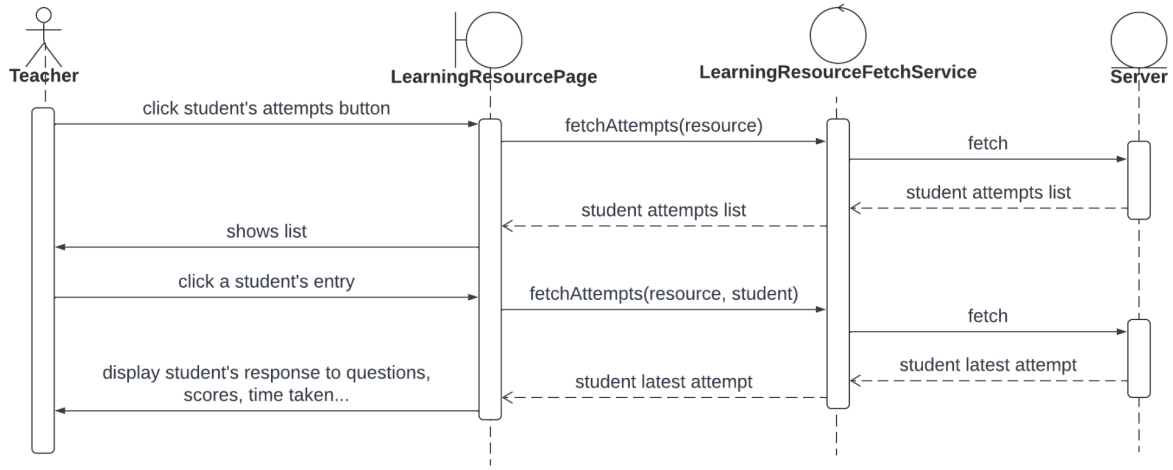
7.5.14 Remove Learning Resource (Teacher)



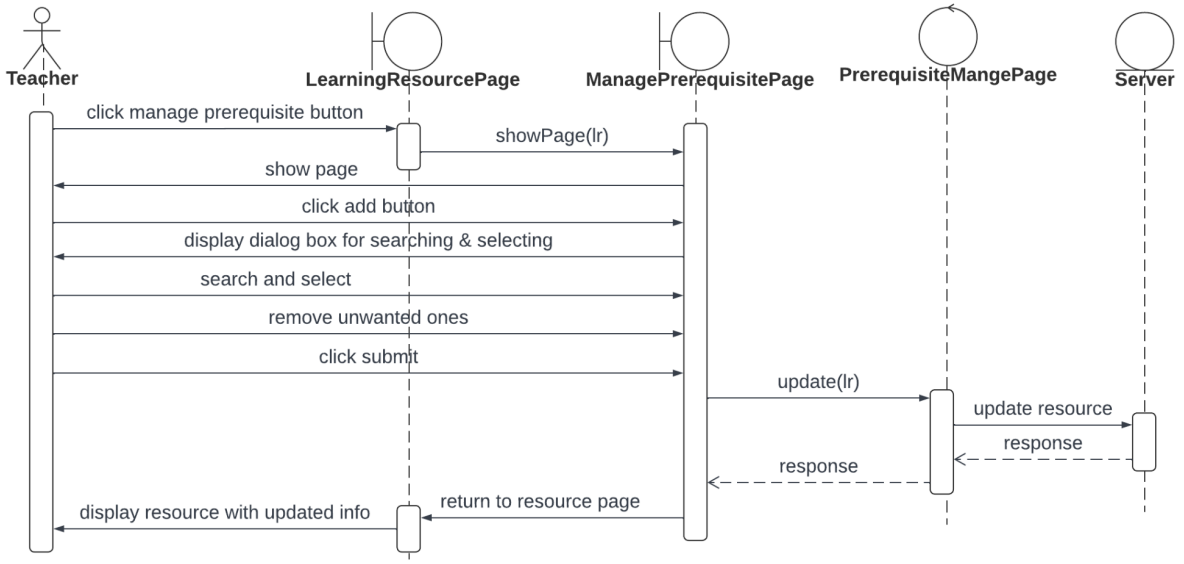
View Student's Usage in Learning Resource (Teacher)



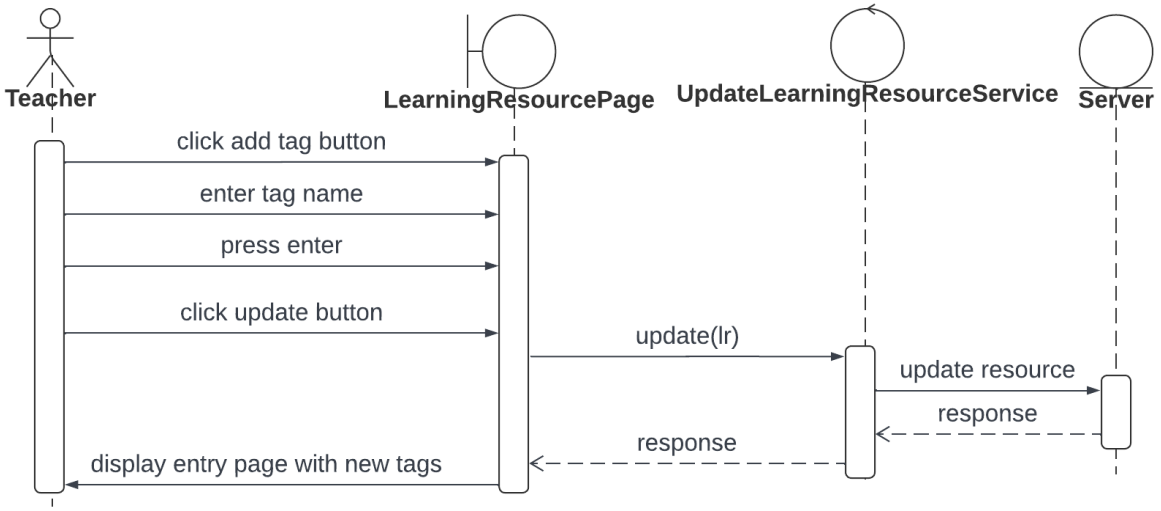
View Student's Performance on Exercises (Teacher)



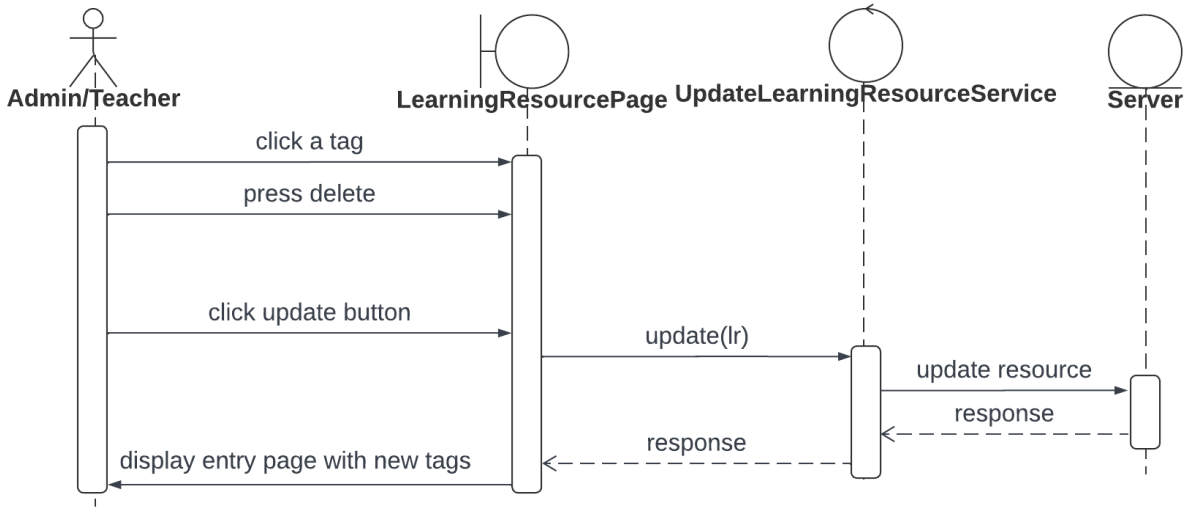
Manage Prerequisite for Learning Resource (Teacher)



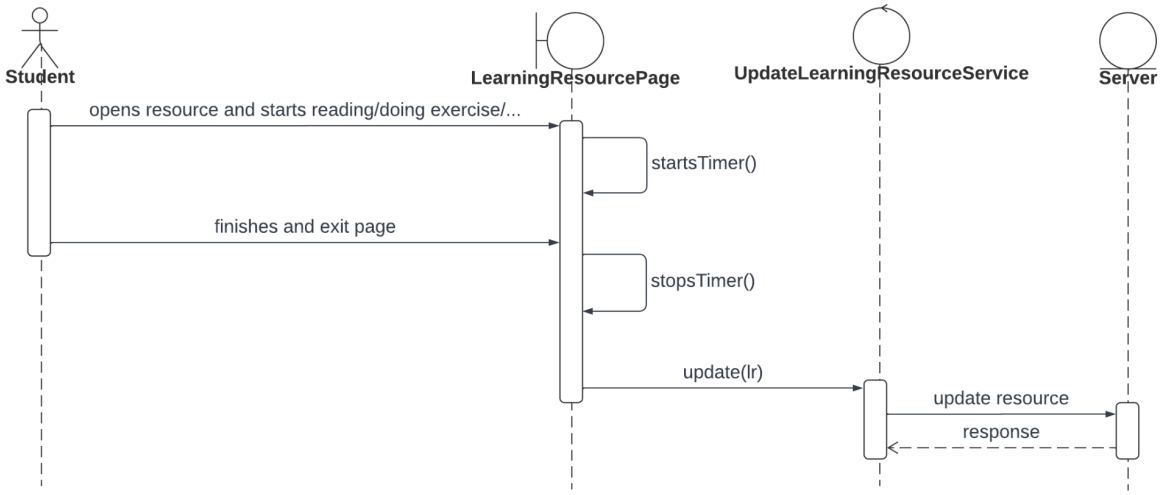
Add Tags for Learning Resource (Teacher)



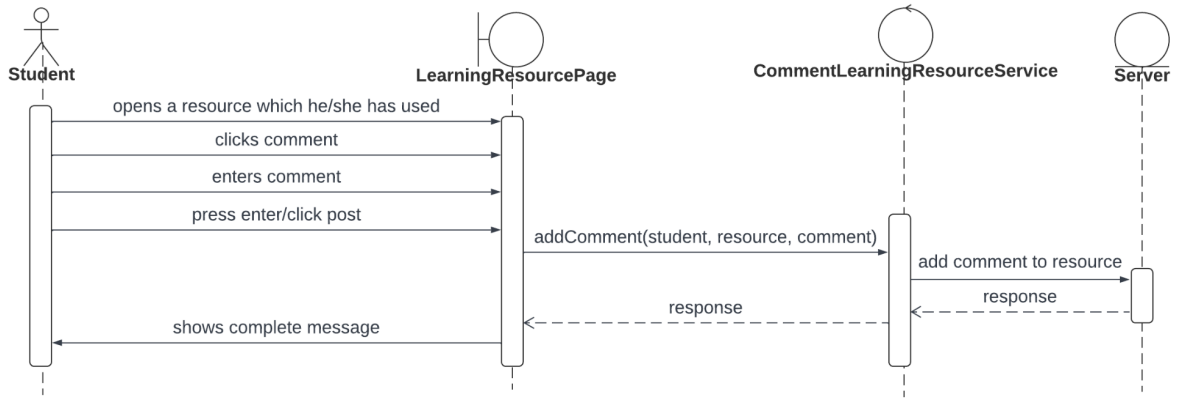
Remove Tags for Learning Resource (Teacher)



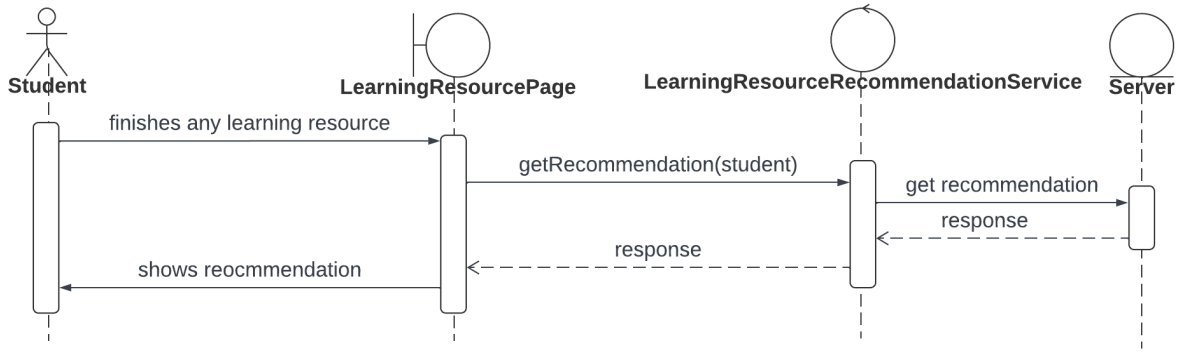
Use Learning Resource (Student)



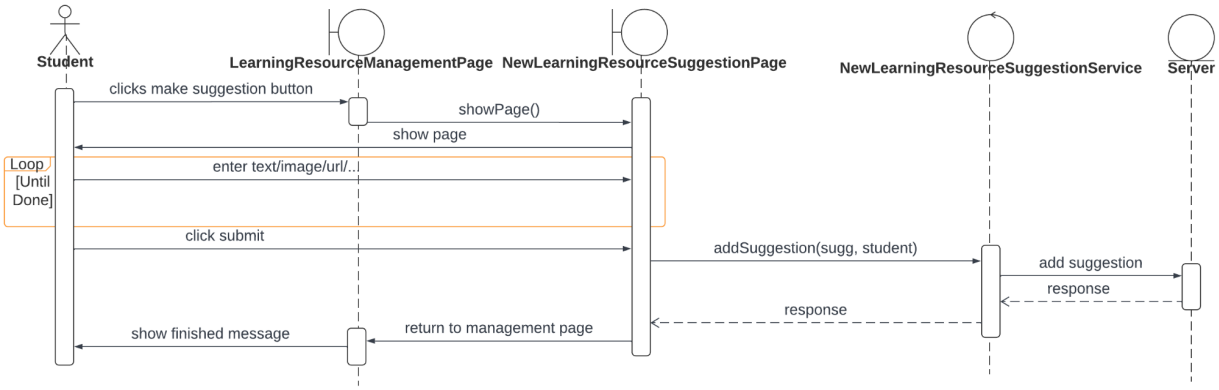
Give Feedback on Learning Resource (Student)



Receive Recommendation of Learning Resource (Student)



Suggest New Learning Resource to Teachers (Student)

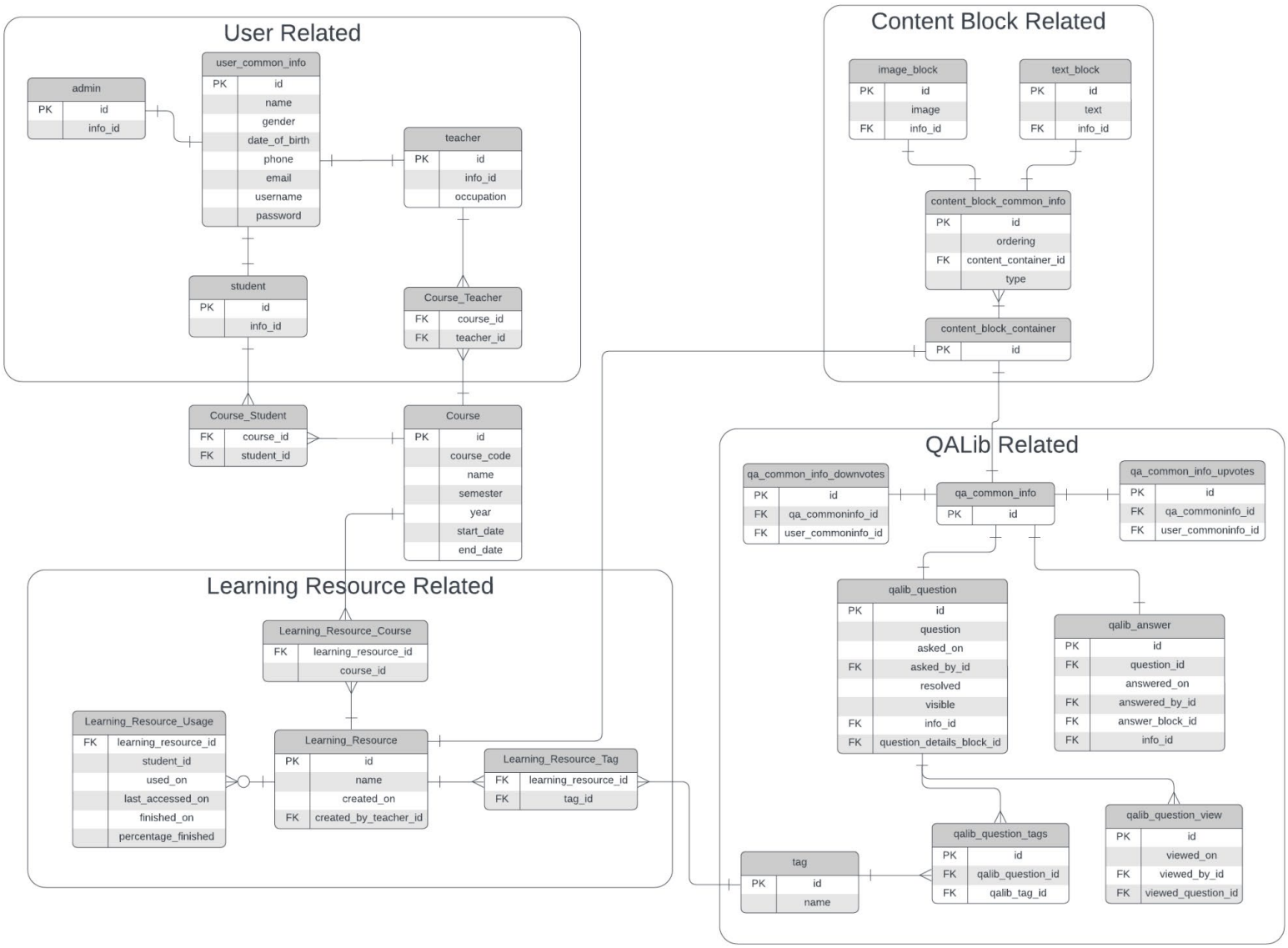


7.6 Data Design

7.6.1 Data Dictionary

Name	Definition
QA Entry	A QA entry is a (n+1)-tuple (Q, A_1, …, A_n) consisting of one question and n corresponding answers, where n is a non-negative integer, i.e. there can be no answer or any number of answers for one question.
QALib	QALib is the collection of all QA entries. A question is stored in the "QALib question" table while an answer is stored in the "QALib answer" table.
Content Block	A content block is an abstract representation of arbitrary types of content used in question entries, answer entries, learning resource entries, etc. A type of content can be image, text, video, etc. For example, a text block (defined below) is a content block of text type, an image block is a content block of image type.
Text Block	A text block is just a section of text.
Image Block	An image block conceptually is just an image. On implementation level, it is a reference (e.g. url) to the image file stored in the file system.
Content Block Container	A content block is a collection of content blocks. It is mainly used to contain the details, as content blocks, of question entries, answer entries, and learning resource entries.

7.6.2 Entity Relationship Diagram (ERD)



7.6.3 Table Descriptions

User Common Info Table

Description: This table is used for storing common information of a user (student/teacher/admin).

Name	Type	Key	Description

id	int	pk	-
name	varchar	-	User’s full name
gender	varchar	-	-
date_of_birth	date	-	-
phone	varchar	-	-
email	varchar	-	-
username	varchar	-	User’s username used for login and identification
password	varchar	-	-

Student Table

Name	**Type**	**Key**	**Description**
id	int	pk	-
info_id	int	fk	Student’s information stored in User Common Info Table

Teacher Table

Name	**Type**	**Key**	**Description**
id	int	pk	-
info_id	int	fk	Teacher’s information stored in User Common Info Table
occupation	varchar	-	Job title of the teacher

Admin Table

id	int	pk	-
info_id	int	fk	Admin’s information stored in User Common Info Table

Course Table

Name	**Type**	**Key**	**Description**
id	varchar	PK	-

course_code	varchar	-	Course code of a course
name	varchar	-	Subject name
semester	varchar	-	Semester of a course in an academic year.
year	date	-	Academic year of a course.
start_date	date	-	Date on which a course begins.
end_date	date	-	Date on which a course ends.

QA Common Info Table

Description: This table is for storing the common information shared by a question entry and an answer entry. Common information includes upvotes and downvotes, which are represented by a row in 'QA Common Info Upvotes' and 'QA Common Info Downvotes' tables. It is designed this way since both a question entry and an answer entry can be upvoted or downvoted and that the vote counts should be kept track of.

Name	**Type**	**Key**	**Description**
id	int	PK	-

QA Common Info Upvotes Table

Name	**Type**	**Key**	**Description**
id	int	PK	-
qa_commoninfo_id	int	fk	Foreign key for a QA Common Info entry.
user_commoninfo_id	int	fk	Foreign key for a User Common Info entry.

QA Common Info Downvotes Table

Name	**Type**	**Key**	**Description**
id	int	PK	-
qa_commoninfo_id	int	fk	Foreign key for a QA Common Info entry.
user_commoninfo_id	int	fk	Foreign key for a User Common Info entry.

QALib Question Table

Name	**Type**	**Key**	**Description**

id	int	PK	-
question	varchar	-	Question asked by a student
asked_on	datetime	-	When a question was asked
asked_by_id	int	FK	Which student asked the question
resolved	boolean	-	Whether the question is resolved
visible	boolean	-	Whether the question and its answers are visible to students.
info_id	int	fk	Foreign key for a ‘QA Common Info entry’
quesiton_details_block_id	int	fk	Foreign key for a ‘Content Block Container’ entry.

QALib Answer Table

Name	**Type**	**Key**	**Description**
id	int	PK	-
question_id	int	fk	Foreign key for a question entry.
answered_on	datetime	-	-
answered_by_id	int	fk	Foreign key for the teacher who answered the question.
answer_block_id	int	fk	Foreign key for a ‘Content Block Container’ entry.
info_id	int	fk	Foreign key for a ‘QA Common Info entry’

Content Block Common Info Table

Name	**Type**	**Key**	**Description**
id	int	PK	-
ordering	int	-	Display order for a content block.
content_container_id	int	fk	Foreign key for the content block container which contains this content block.
type	varchar	-	Type of the content block

Content Block Container Table

Description: A content block container represents the detailed information a question or answer entry contains, such as images and text blocks.

Name	Type	Key	Description
id	int	PK	-

Image Block Table

Name	Type	Key	Description
id	int	PK	-
image	varchar	-	-
info_id	int	fk	Foreign key for the Content Block Common Info entry.

Text Block Table

Name	Type	Key	Description
id	int	PK	-
text	varchar	-	-
info_id	int	fk	Foreign key for the Content Block Common Info entry.

QALib Question View Table

Name	Type	Key	Description
id	int	PK	-
viewed_on	datetime	-	-
viewed_by_id	int	fk	ID of the user who viewed it. References an entry of 'User Common Info' table.
viewed_question_id	int	fk	Foreign key for a question entry.

QALib Question Tags Table

Name	Type	Key	Description
id	int	PK	-
qalib_question_id	int	fk	Foreign key of a question entry.
qalib_tag_id	int	fk	Foreign key for a QALib Tag entry.

QALib Tag Table

Name	Type	Key	Description
id	int	PK	-
name	varchar	-	-

Learning Resource Table

Name	Type	Key	Description
id	varchar	PK	-
name	varchar	-	Name of the media file
extension	varchar	-	File extension of the media file
uploaded_at	datetime	-	When the media file is uploaded
media_type_id	int	FK	ID of the type of media file

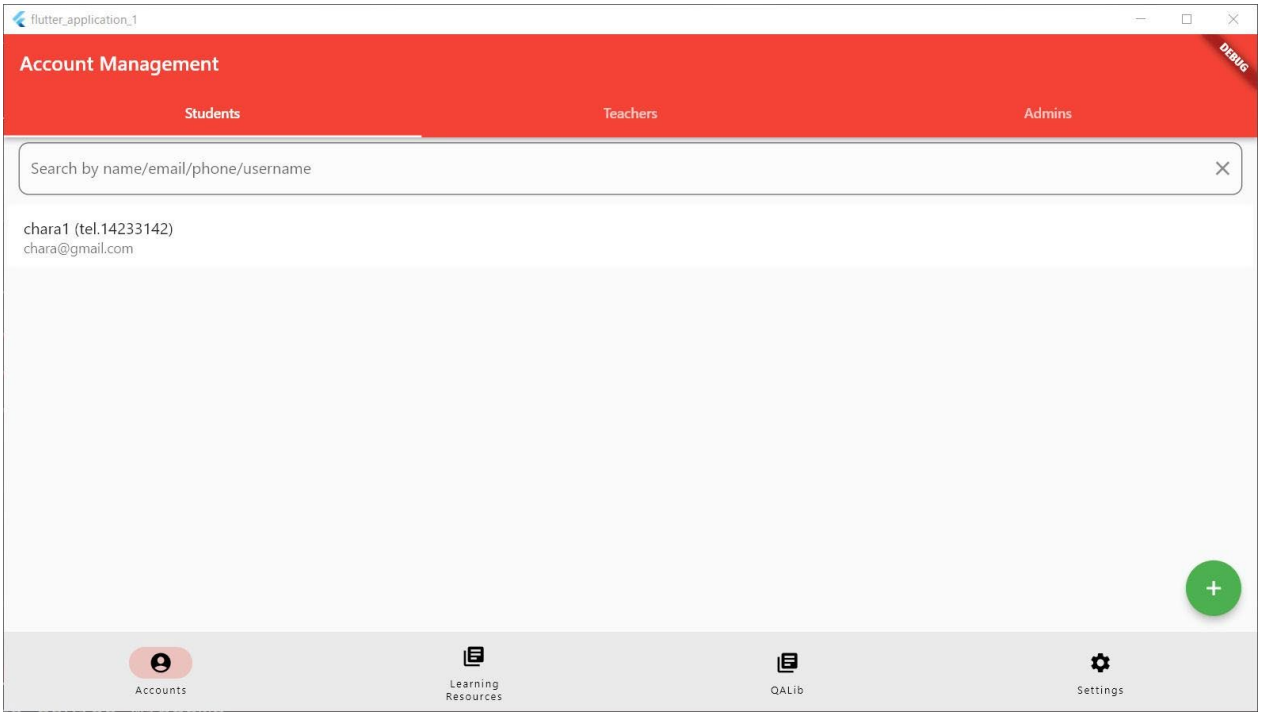
Media Type Table

Name	Type	Key	Description
id	varchar	PK	-
name	varchar	-	Name of the media type

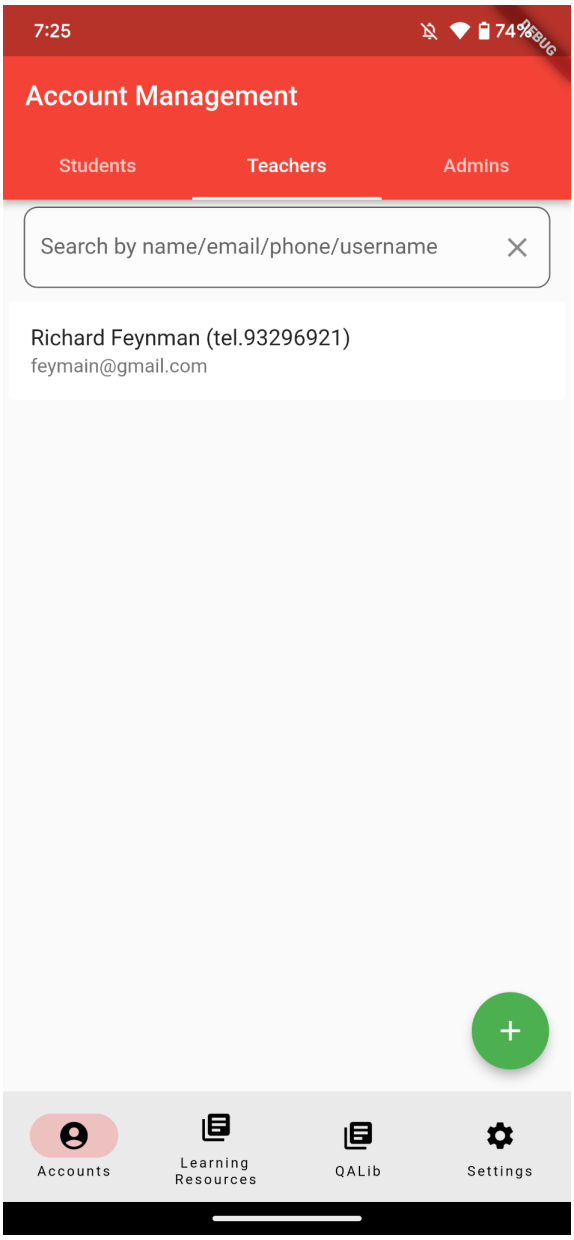
7.7 User Interface Design

7.7.1 Account Management

Desktop:

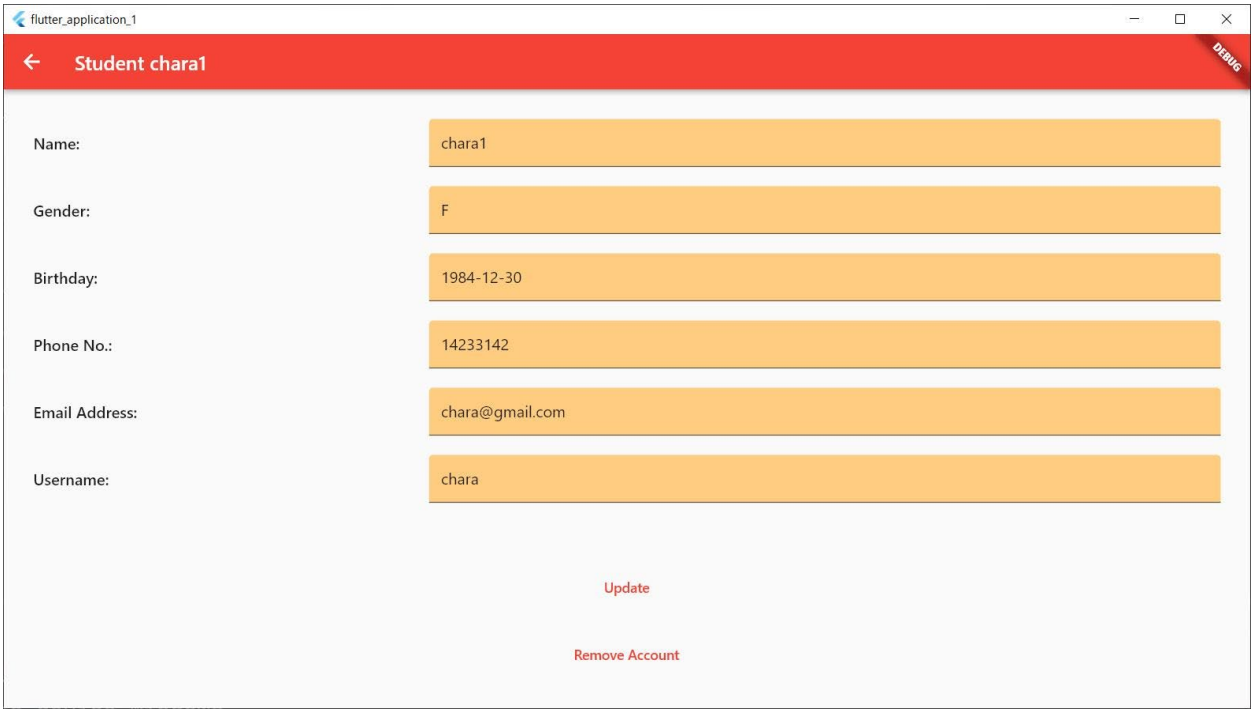


Mobile:

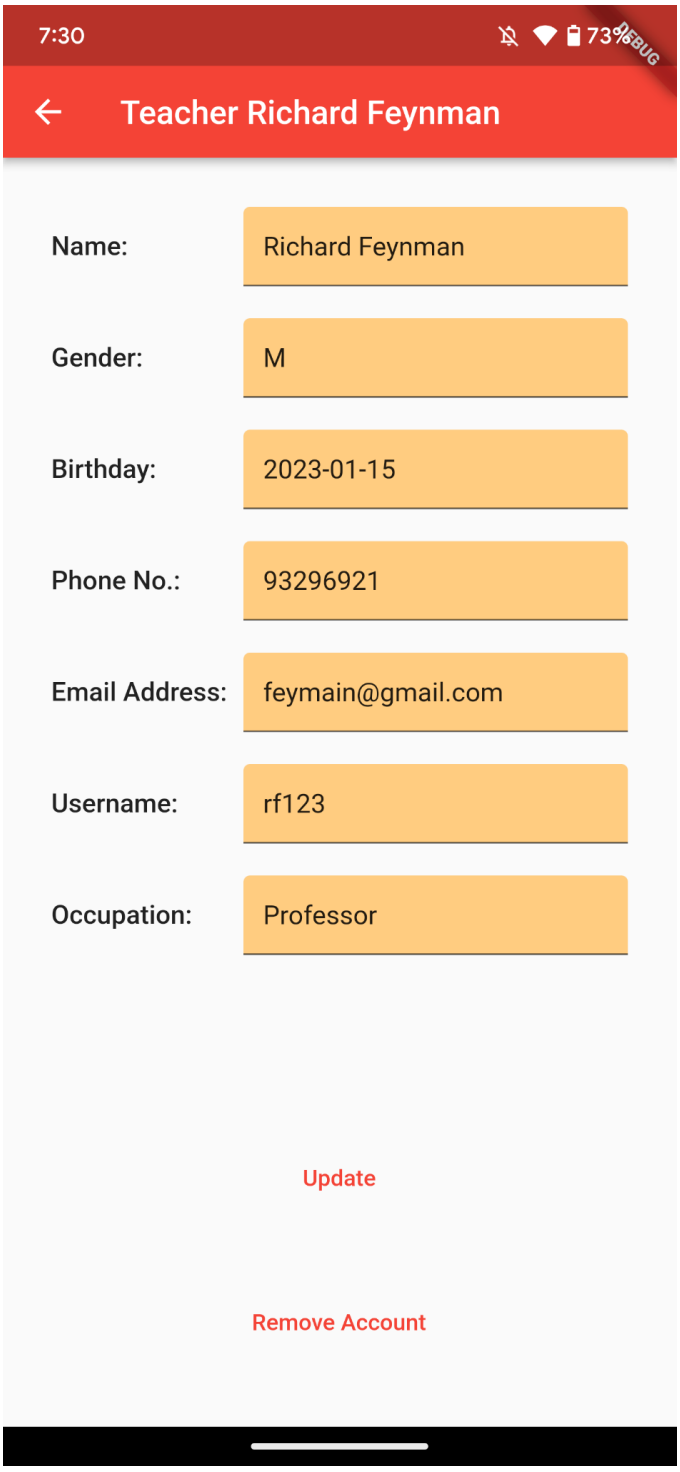

7:25
74%
DEBUG
Account Management
Students
Teachers
Admins
Search by name/email/phone/username
Richard Feynman (tel.93296921)
feymain@gmail.com
Accounts
Learning Resources
QALib
Settings

7.7.2 User Details Page

Desktop:

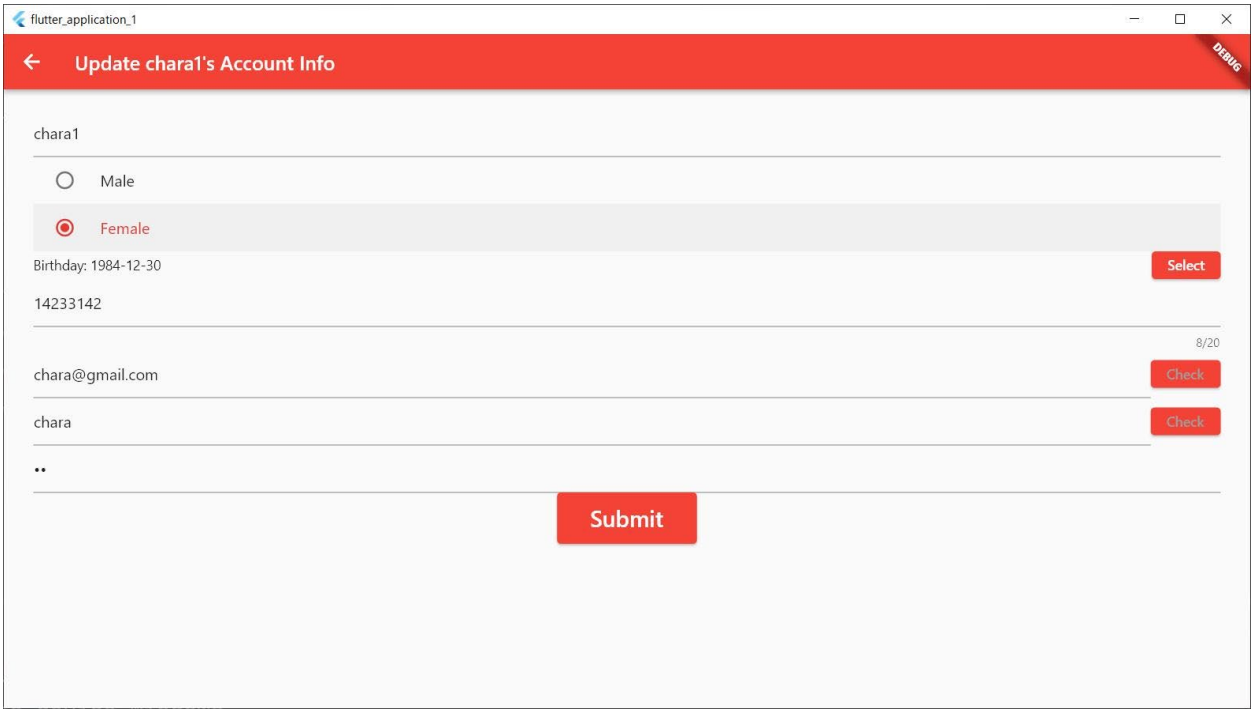


Mobile:

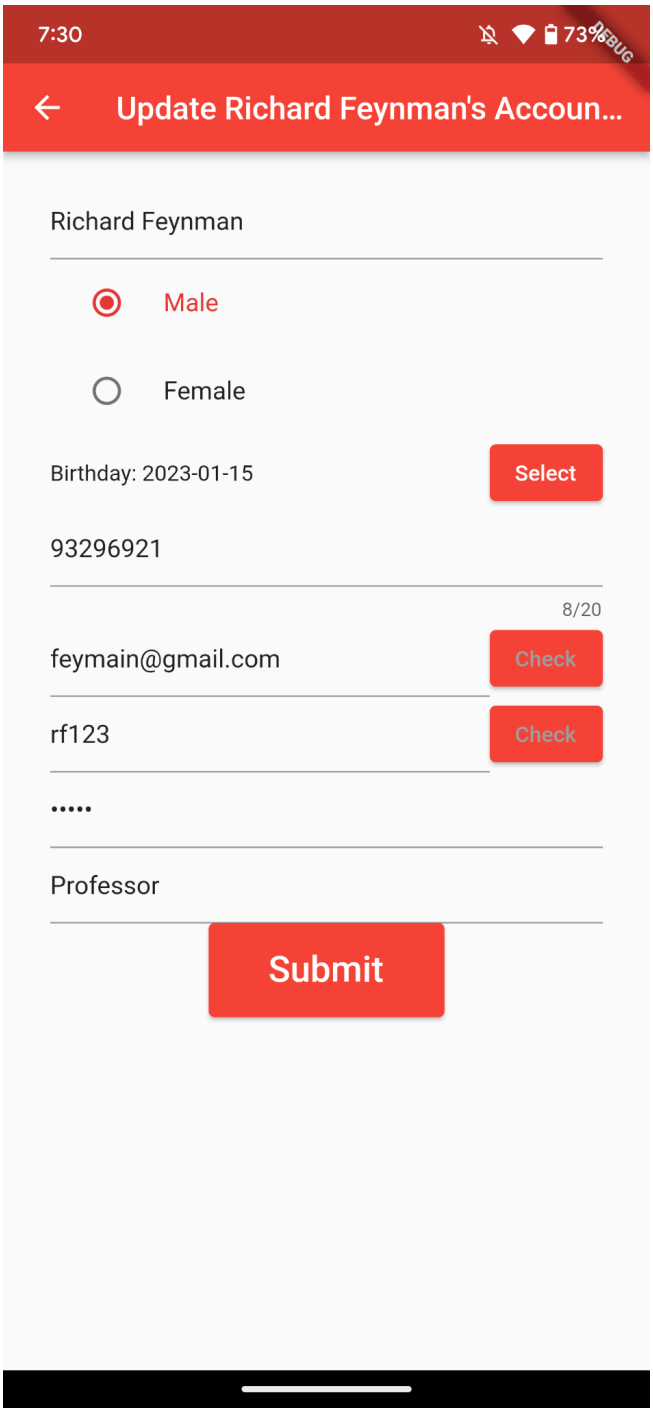

7:30
73%
DEBUG
Teacher Richard Feynman
Name:
Richard Feynman
Gender:
M
Birthday:
2023-01-15
Phone No.:
93296921
Email Address:
feymain@gmail.com
Username:
rf123
Occupation:
Professor
Update
Remove Account

7.7.3 Update User Page

Desktop:

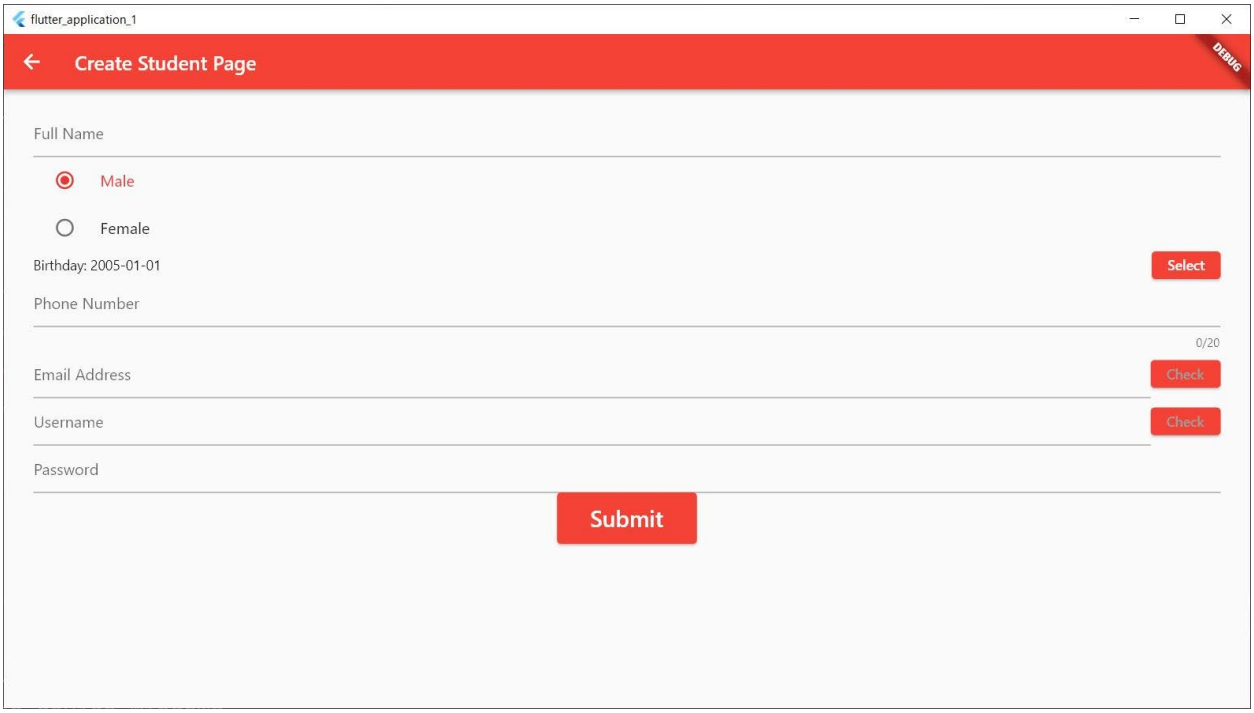


Mobile:

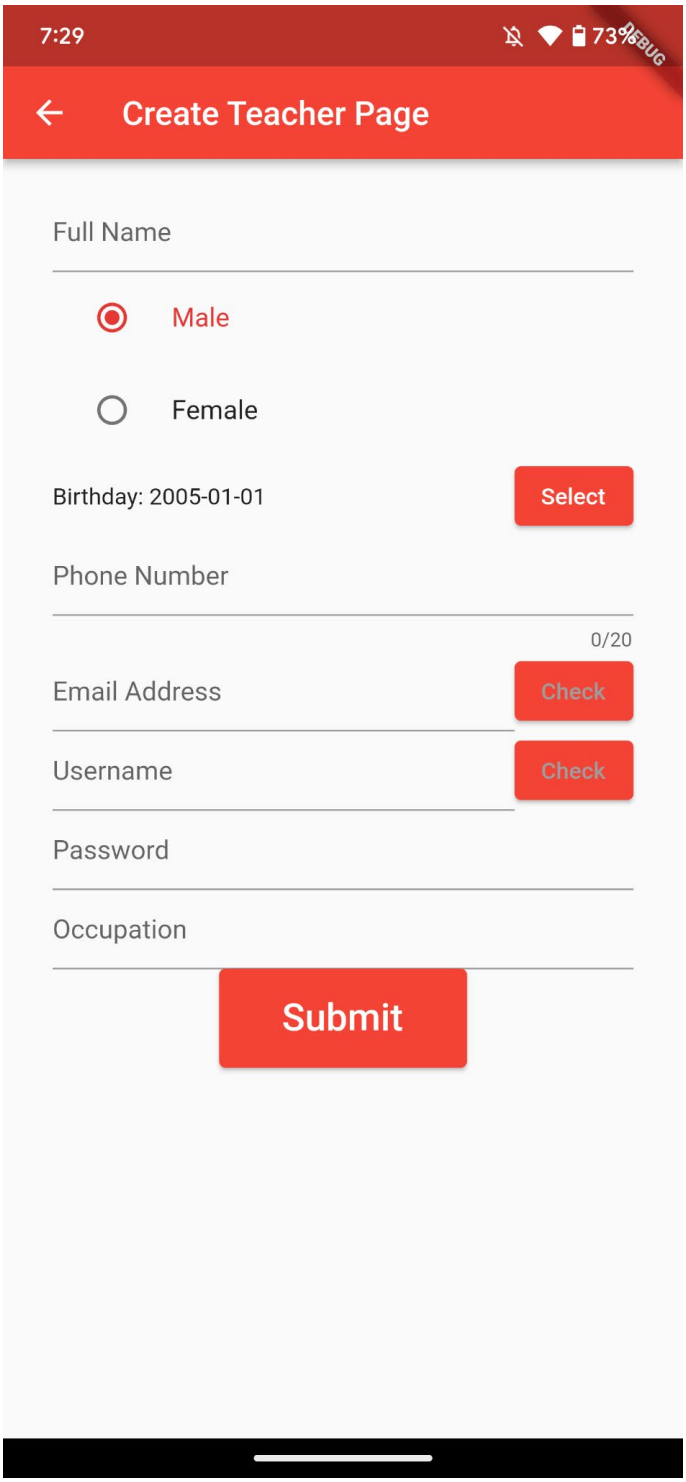

7:30
73%
DEBUG
Update Richard Feynman's Accoun...
Richard Feynman
Male
Female
Birthday: 2023-01-15
Select
93296921
8/20
feymain@gmail.com
Check
rf123
Check
•••••
Professor
Submit

7.7.4 Create User Page

Desktop:

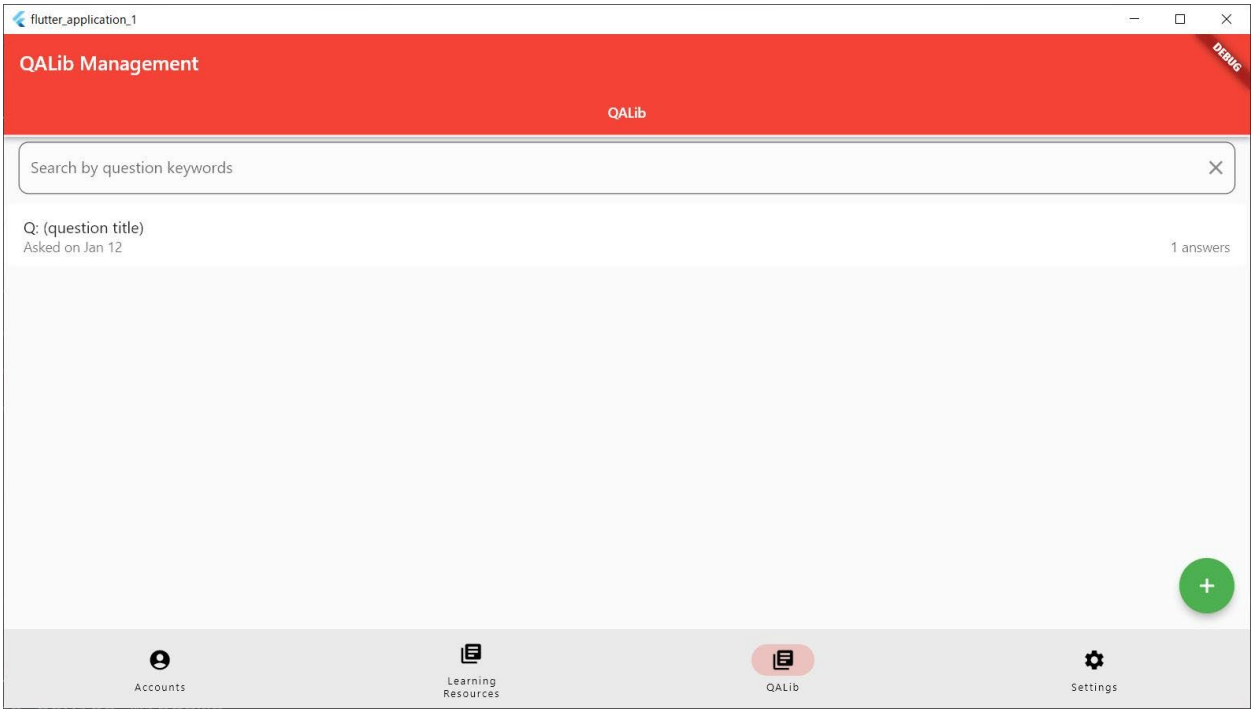


Mobile:

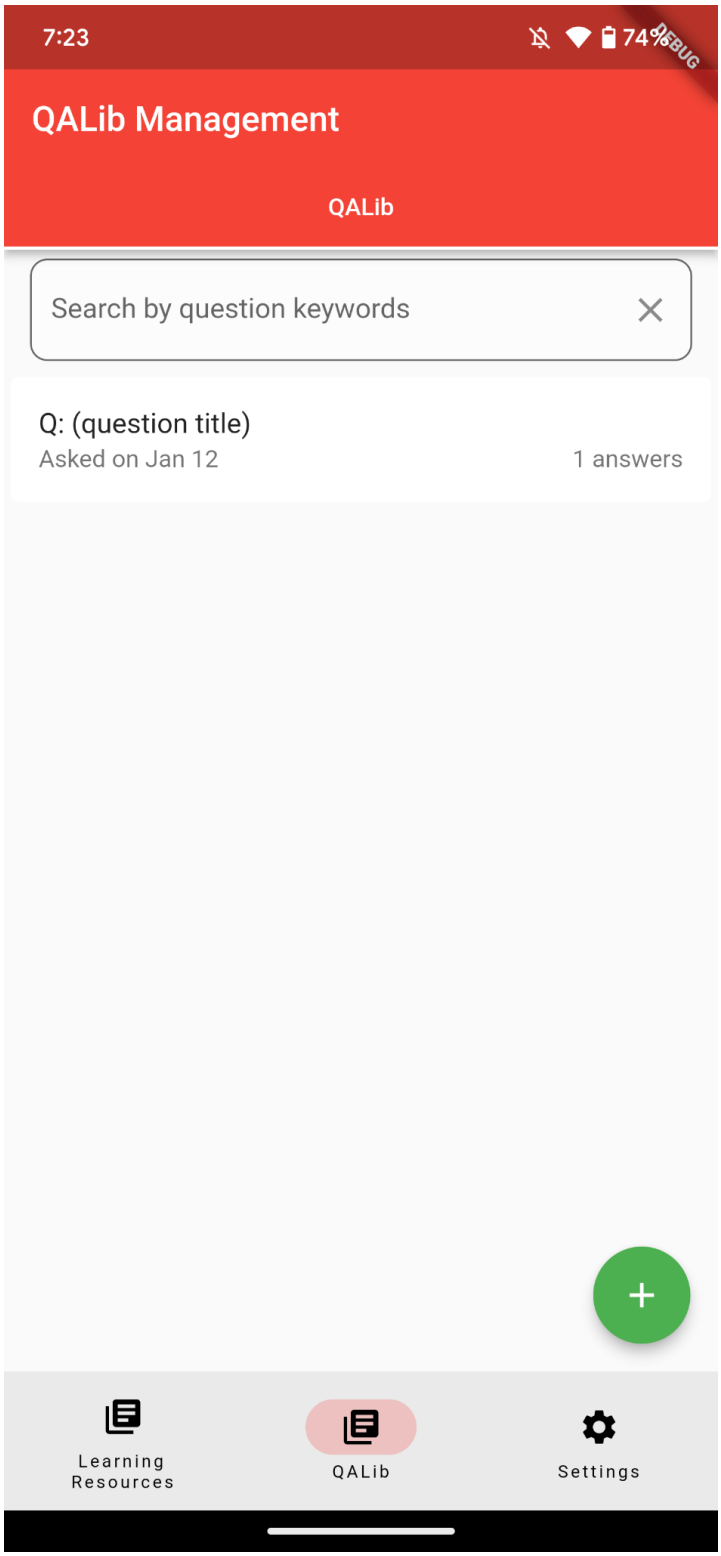

7:29
73%
DEBUG
Create Teacher Page
Full Name
Male
Female
Birthday: 2005-01-01
Select
Phone Number
0/20
Email Address
Check
Username
Check
Password
Occupation
Submit

7.7.5 QALib Page

Desktop:

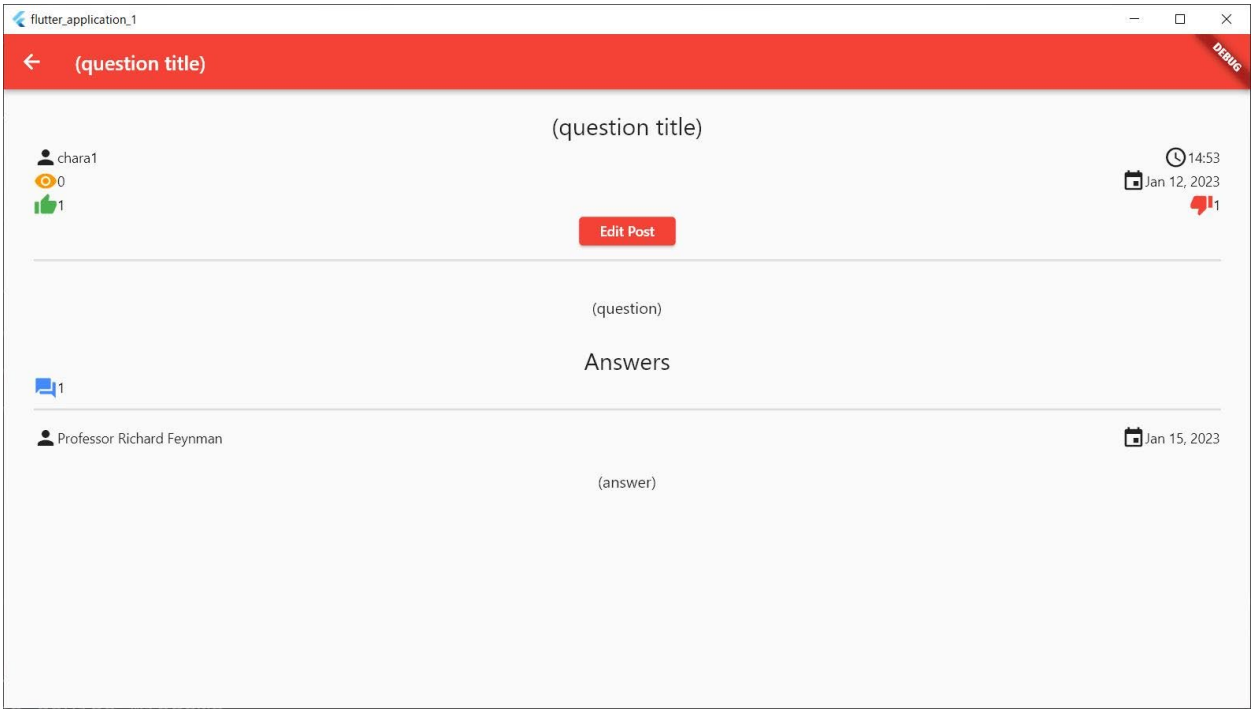


Mobile:

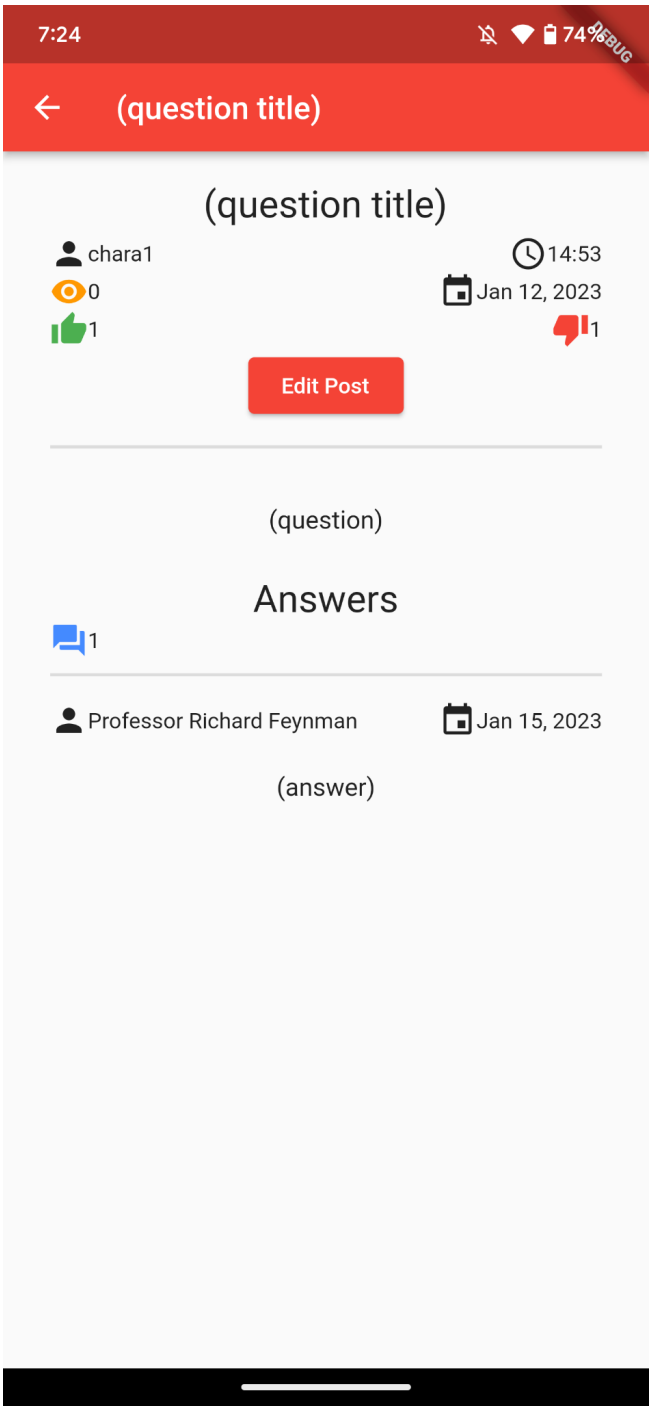

7:23
74%
DEBUG
QALib Management
QALib
Search by question keywords
Q: (question title)
Asked on Jan 12
1 answers
Learning Resources
QALib
Settings

7.7.6 QALib Entry Page

Desktop:



Mobile:


7:24
74%
DEBUG
(question title)
(question title)
chara1
14:53
0
Jan 12, 2023
1
1
Edit Post
(question)
Answers
1
Professor Richard Feynman
Jan 15, 2023
(answer)

8 Project Plan

8.1 Software Development Model

Due to its simplicity, the waterfall development method will be used in developing the software.

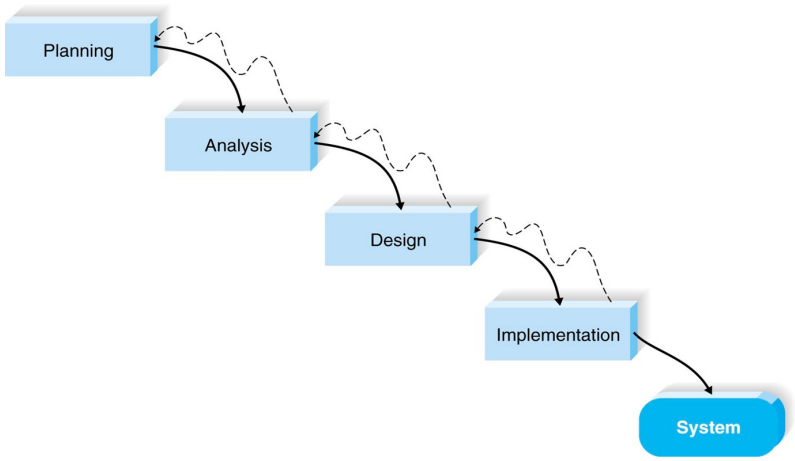


Figure 3: Waterfall model.

8.2 Project Schedule

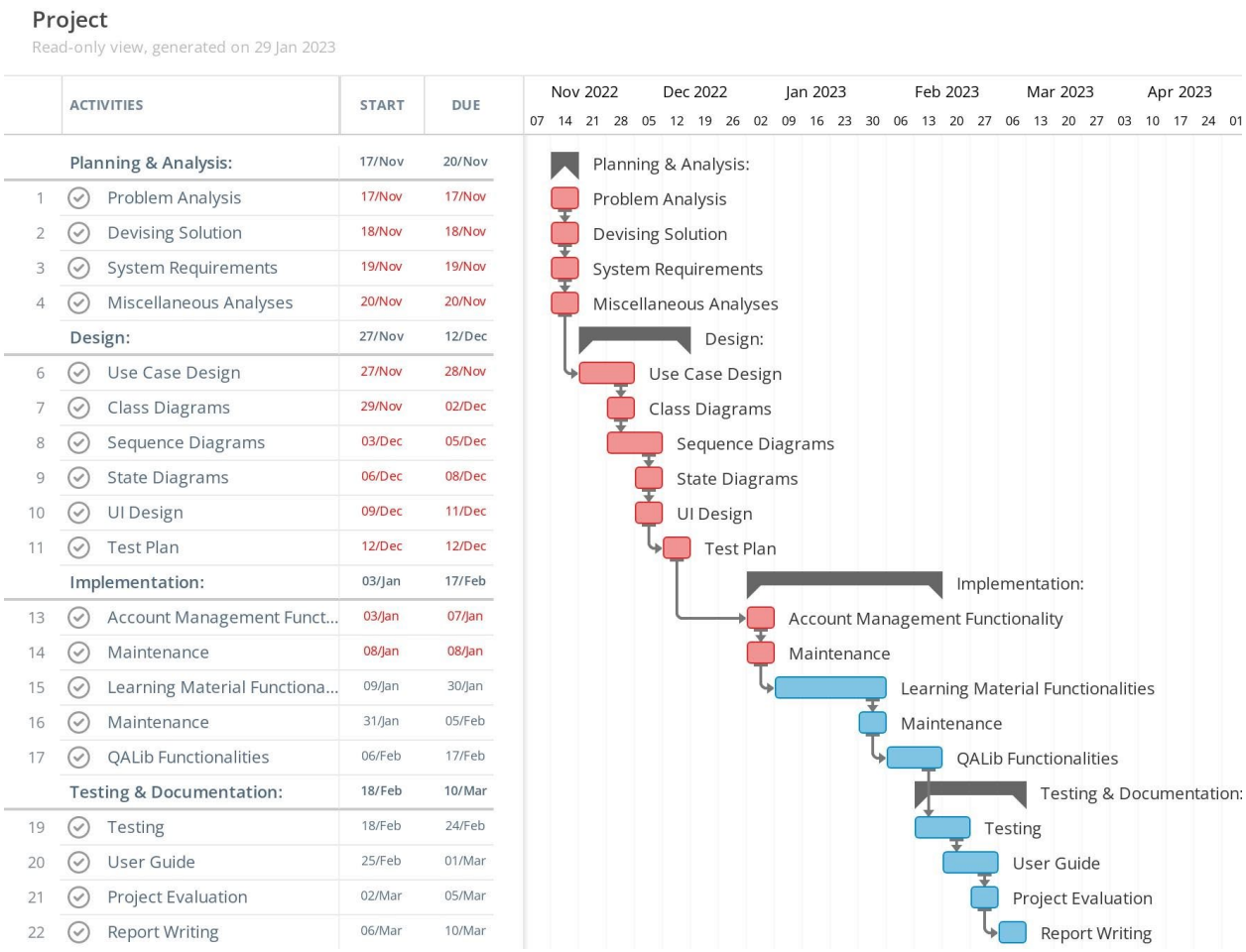


Figure 4: Project schedule. Made in Instagantt.

8.3 Deliverables

1. Initial report, interim report, final report
2. Windows desktop app
3. Android app
4. User guide

8.4 Software Tools Needed

- IntelliJ IDEA
- Lucid Chart

- Instagantt

8.5 Implementation Language

- Dart
- Python

8.6 Development Framework

- Flutter
- Django

9 Implementation

9.1 Test Plan

Scope

- The testable part of the codebase
- Some UI elements

Tools

- flutter_test library provided by the Flutter package

9.2 Test Cases & Test Data

Test Case	Log in with correct username and password
Premise	Staff has an account with username “saul” and password “goodman”
Steps	1. Enter username + password 2. Click login
Test Data	username: saul password: goodman
Expected Result	Login successful
Pass/Fail	Pass

Test Case	Log in with wrong credentials
Premise	Staff has an account with username “saul” and password “goodman”

Steps	1. Enter username + password 2. Click login
Test Data	username: abc password: abc
Expected Result	Login failed and a message is shown
Pass/Fail	Pass

Test Case	Login attempt with no credentials
Premise	Staff has an account with username “saul” and password “goodman”
Steps	1. Click login without entering any credentials
Test Data	username: abc password: abc
Expected Result	An message will be shown to remind the user to enter the login credentials
Pass/Fail	Pass

Test Case	Search for user record by name
Context	An admin wants to search for a user record in the account management page.
Premise	A user with the following information exists: Name: Josh Username: josh12 Email: josh@gmail.com
Steps	1. Admin enters “josh” in the search bar and clicks search button.
Test Data	Search phrase “josh”
Expected Result	System shows all users whose name contains josh.
Pass/Fail	Pass

Test Case	Search for user record by email address
Context	An admin wants to search for a user record in the account management page.
Premise	A user with the following information exists: Name: Josh Username: josh12 Email: josh@gmail.com
Steps	1. Admin enters “josh@” in the search bar and clicks search button.
Test Data	Search phrase “josh@”
Expected Result	System shows all users whose email contains “josh@”.
Pass/Fail	Pass

Test Case	Search for user record by username
Context	An admin wants to search for a user record in the account management page.
Premise	A user with the following information exists: Name: Josh Username: josh12 Email: josh@gmail.com
Steps	1. Admin enters “josh12” in the search bar and clicks search button.
Test Data	Search phrase “josh12”
Expected Result	System shows all users whose name contains josh12.
Pass/Fail	Pass

Test Case	Open the detail page of a user record
Context	An admin wants to view the details of user Josh in the account management page.

Premise	-
Steps	1. Admin clicks the record of user Josh.
Test Data	User Josh and all his info.
Expected Result	System opens the detail page of user Josh and shows all the details of Josh.
Pass/Fail	Pass

Test Case	Update user information
Context	An admin wants to update a user's information.
Premise	A user with the following information exists: Name: Josh Username: josh12 Email: josh@gmail.com The new username and email address doesn't already exist.
Steps	1. Admin enters the detail page of user Josh. 2. Admin clicks the update button. 3. System opens a page for entering new information for the user. 4. Admin enters new information. 5. Admin clicks submit button.
Test Data	A user with the following information exists: Name: Josh Wolfgang Username: josh123 Email: josh1@gmail.com
Expected Result	System updates the information accordingly, and returns back to the detail page.
Pass/Fail	Pass

Test Case	Remove user record
Context	An admin wants to remove a user's record.
Premise	-

Steps	1. Admin enters the detail page of user Josh. 2. Admin clicks remove account button. 3. System prompts a confirmation dialog. 4. Admin clicks confirm.
Test Data	-
Expected Result	System removes the user record, and returns back to the account management front page.
Pass/Fail	Pass

Test Case	Create new user account
Context	An admin wants to create a new user account for a user.
Premise	An account with the username “kel456” and email address “kel@gmail.com” doesn’t already exist.
Steps	1. Admin enters the account management page. 2. Admin clicks the create user button. 3. System opens a page for entering the user’s information. 4. Admin enters the user information. 5. Admin clicks submit button.
Test Data	New user’s info: Name: Kelly Username: kel456 Email: kel@gmail.com
Expected Result	System creates the user and navigates to the user detail page.
Pass/Fail	Pass

Test Case	Attempt to create a new user account with existing username.
Context	An admin wants to create a new user account with an existing username.
Premise	An account with the username “kel456” already exists.
Steps	1. Admin enters the account management page.

	2. Admin clicks the create user button. 3. System opens a page for entering the user's information. 4. Admin enters the user information. 5. Admin clicks submit button.
Test Data	New user's info: Name: Kelly Username: kel456 Email: kel@gmail.com
Expected Result	System shows a message saying that the username is being used by an existing account.
Pass/Fail	Pass

Test Case	Attempt to create a new user account with existing email.
Context	An admin wants to create a new user account with an existing email.
Premise	An account with the email address "kel@gmail.com" already exists.
Steps	1. Admin enters the account management page. 2. Admin clicks the create user button. 3. System opens a page for entering the user's information. 4. Admin enters the user information. 5. Admin clicks submit button.
Test Data	New user's info: Name: Kelly Username: kel456 Email: kel@gmail.com
Expected Result	System shows a message saying that the email address is being used by an existing account.
Pass/Fail	Pass

Test Case	Logout
Context	-

Premise	-
Steps	1. The user clicks the logout icon at the upper right corner. 2. System prompts a confirmation dialog. 3. The user clicks yes button.
Test Data	-
Expected Result	The system logs out the user and returns the page to login page.
Pass/Fail	Pass

Test Case	Create a question
Context	A student wants to create a question
Premise	-
Steps	1. Student clicks the add new question button. 2. System shows a page for entering information for the question. 3. Student enters question title, select tags, and writes the content for the question, then clicks create button.
Test Data	Question title: “How is the pullback of a tensor defined?” Tag: Differential Geometry Content: “Hi. I’m a beginner learning DG. I don’t understand the definition of a tensor pullback. Please help. Here’s the definition presented in my book:” + some LaTex code.
Expected Result	System creates the question and enters the detail page of the question.
Pass/Fail	Pass

Test Case	Edit a question
Context	A student wants to edit a question he/she created previously.
Premise	-
Steps	1. Student enters the detail page of the question. 2. Student clicks edit question button.

	3. System shows a page for editing the question. 4. Student enters new info and content and clicks apply.
Test Data	Question title: “How is the pullback and pushforward of a tensor defined?” Tag: Differential Geometry, Maths Content: “Hi. I’m a beginner learning DG. I don’t understand the definition of a tensor pullback and pushforward. Please help.”
Expected Result	System saves the changes and returns back to the detail page of the question.
Pass/Fail	Pass

Test Case	Delete a question
Context	A student wants to remove a question he/she created previously.
Premise	-
Steps	1. Student enters the detail page of the question. 2. Student clicks remove button. 3. System prompts a confirmation dialog. 4. Student clicks confirm.
Test Data	-
Expected Result	System removes the question and returns to the front page.
Pass/Fail	Pass

Test Case	Search for a question by question title
Context	A user wants to search for a question.
Premise	A question with the following info exists. Question title: “How is the pullback and pushforward of a tensor defined?” The question is not set as archived.
Steps	1. User enters “tensor” into the search bar and clicks search button.
Test Data	“tensor” as the search text

Expected Result	The system shows all questions with a title containing “tensor”.
Pass/Fail	Pass

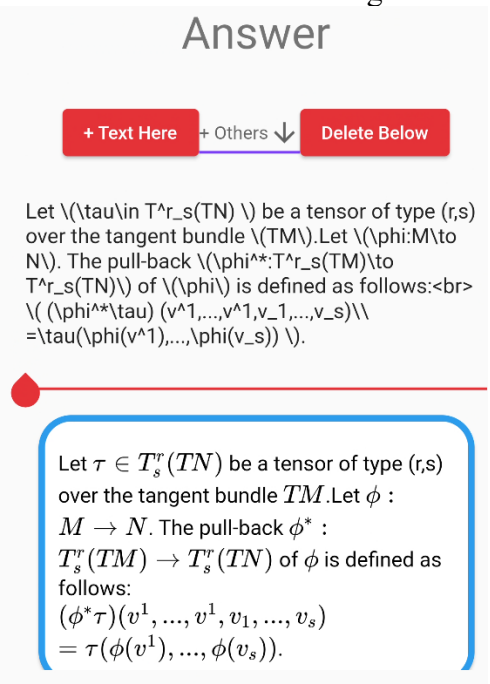
Test Case	Filter question by tags
Context	A user wants to search for a question.
Premise	The question is not set as archived. The question is tagged “Maths”
Steps	1. User clicks the advanced search settings button. 2. System shows the advanced search settings dialog. 3. User chooses the “Maths” for filtering and clicks apply.
Test Data	“Maths” as the filtering tag
Expected Result	The system shows all questions tag “Maths”
Pass/Fail	Pass

Test Case	Filter/Unfilter archived questions
Context	A user wants to search for a question.
Premise	The question *is* set as archived. The question is not created by the user.
Steps	1. User clicks the advanced search settings button. 2. System shows the advanced search settings dialog. 3. User checks/unchecks the checkbox “show archived” and clicks apply.
Test Data	-
Expected Result	The system shows (archived and) not archived questions.
Pass/Fail	Pass

Test Case	Filter questions by creator
Context	A user wants to search for questions he/she created.
Premise	-
Steps	1. User clicks the advanced search settings button. 2. System shows the advanced search settings dialog. 3. User checks the checkbox “show only mine” and clicks apply.
Test Data	-
Expected Result	The system shows questions created by the user, regardless of whether they are archived.
Pass/Fail	Pass

Test Case	Filter/Unfilter resolved questions
Context	A user wants to search for resolved/unresolved questions
Premise	-
Steps	1. User clicks the advanced search settings button. 2. System shows the advanced search settings dialog. 3. User checks/unchecks the checkbox “show resolved” and clicks apply.
Test Data	-
Expected Result	The system shows only unresolved/all questions
Pass/Fail	Pass

Test Case	Respond to a question
Context	A teacher wants to write an answer in response to a question asked by a student.
Premise	The question is not archived and is not resolved.
Steps	1. Teacher enters the question details page.

	2. Teacher clicks the add answers button. 3. System shows page for adding an answer. 4. Teacher enters the info and clicks submit.
Test Data	An answer with the following latex block: 
Expected Result	The system creates the answer for the question, which shows in the detail page of the question.
Pass/Fail	Pass

Test Case	Edit a response to a question
Context	A teacher wants to edit an answer he/she created earlier.
Premise	The question is not archived.
Steps	1. Teacher enters the question details page. 2. Teacher clicks the edit answers button. 3. System shows page for editing the answer. 4. Teacher enters the new info and clicks submit.
Test Data	-
Expected Result	The system updates the answer.

Pass/Fail	Pass

9.3 Design Changes

Minor parts of the system were redesigned due to some issues in the project problem formulation (see section 10.1), which are as follows:

- Removed function “The system shall allow students to communicate between each other and with teachers (including asking questions)” (originally called FR-S5)

Also, due to time limitations, the following two proposed minor functions were discarded, while the main functions of the system remain unaffected:

- “The system shall allow students to suggest learning resources to teachers” (FR-S2)
- “The system shall allow students to give feedback on learning resources.” (FR-S4)

10 Results & Conclusions

10.1 Critical Evaluation

The original formulation of the problems was problematic. During problem analysis, the team asserted that distance learning caused by covid would reduce peer learning and cause difficulty in helping students. However, this proposition was based on the premise that covid would still be a problem after the software has been developed. In hindsight, this is simply not true. The effects that covid brings are almost negligible at the time of writing this report. Most schools have long returned to their ordinary operation. Thus, the two stated problems this project attempts to solve do not exist anymore. The other two proposed problems, namely one size fits all teaching and difficulty in evaluating quality of course materials, on the other hand, remained as real and well posed problems since they are well known problems in modern education and have been repeatedly addressed by people in different fields.

The contrived solutions addressing the other problems remained plausible, although their efficacy is to be evaluated.

A lapse in judgment was present during the planning of the project. The software development framework should be carefully evaluated before the beginning of development to make sure problems occurring during development do not come from the framework itself. Also, caution should have been exercised when picking an unfamiliar programming language for the development.

Requirement specification was largely reasonable and most proposed functions align with the purpose of the system.

The QALib (question-answer library) should work satisfactorily given that students would ask questions themselves. The existing entries, however, may not be adequately accessible due to the poor overly primitive search function. The management tools for the library should be complete and fully functional.

10.2 Problems Encountered

Initially, the plan was to develop the system using .net maui, which is a multi-platform framework for building applications for both desktop and mobile using a shared codebase. However, the team was faced with tremendous unforeseen hardship, causing early development to be extremely inefficient and frustrating.

The primary issue was that aside from the unfamiliarity with the framework, the framework is relatively new. Its stable version was released only 6 months prior to the beginning of development, which was an overlooked problem due to it being a stable release. Even though the development had only begun for not long, there were already countless errors. On top of that, most error messages were hardly of any use. Sometimes there were even errors with no error messages. Most development time was wasted in debugging. In addition, development was prohibitively slow due to the fact that the hot reloading feature was very error prone and that the project build time was very slow.

The team decided to abandon the project and move to an alternative framework Flutter after struggling for around 10 days.

The next problem is that development in Flutter with the programming language Dart is significantly more difficult for writing readable code in Dart takes tremendous effort. Since all UI codes in Dart are nested together, code refactoring is often difficult and brittle. Besides, it is typical that every little widget takes so much code to write, and since UI codes are susceptible to changes, they are not suitable for refactoring, which means the DRY programming principle cannot be complied with. Also, code duplication further worsen the readability, which then slows down development. For example, it took about one week to code what can be done in 1-2 days in a C# windows form application.

10.3 Project Schedule Changes

In the design phase, the project finishing date was delayed to early March due to change in implementation framework, namely from .net maui to flutter.

In the implementation phase, the new expected completion date is delayed to early April due to the unexpected difficulty in programming under Flutter framework.

10.4 Limitations of the System

The major concern regarding the system is that the usability of QALib and the learning resource library is harmed by the lack of a good search engine. The current search engine relies merely on simple keyword matching. The problem appears when users search by keywords with different variations than the one the entry contains. For example, if the title of a question is “how to prove the determinant of a matrix is equal to the product of its eigenvalues?” but a user searches “matrix determinant eigenvalue”, this question entry will not show up in the search result due to the overly simple search engine. Preliminary analysis shows that implementing a satisfactory search engine requires much more time than available due to the lack of knowledge in related domains. Thus, this issue remains, and the team resorted to tags matching as an attempt to improve searchability of database entries.

10.5 Summary

Several obstacles emerge when one tries to solve the problems in education, such as educater’s or student’s reluctance to learn or to adopt new technologies, whether these new approaches actually perform better than traditional approaches, difficulties in popularizing them, etc. Nevertheless, this project made an attempt in tackling such problems.

In hindsight, the team thought that the problem the project tries to solve could have been better analyzed, which could have led to better designed system.

11 References

[1] “Education: One size no longer has to fit all,” *Technology and Operations Management*. https://d3.harvard.edu/platform-rctom/submission/education-one-size-no-longer-has-to-fit-all/ (accessed Nov. 18, 2022).

[2] M. W. Kreuter, V. J. Strecher, and B. Glassman, “One size does not fit all: The case for tailoring print materials,” *Ann. Behav. Med.*, vol. 21, no. 4, pp. 276–283, Dec. 1999, doi: 10.1007/BF02895958.

[3] D. Kolb and A. Kolb, *The Kolb Learning Style Inventory 4.0: Guide to Theory, Psychometrics, Research & Applications*. 2013.

[4] “Intelligent tutoring system,” *Wikipedia*. Nov. 09, 2022. Accessed: Nov. 21, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Intelligent_tutoring_system&oldid=1120995469#Modern_ITS

[5] E. Onyema *et al.*, “Impact of Coronavirus Pandemic on Education,” *J. Educ. Pract.*, vol. 11, pp. 108–121, May 2020, doi: 10.7176/JEP/11-13-12.

[6] J. G. Tullis and R. L. Goldstone, “Why does peer instruction benefit student learning?,” *Cogn. Res. Princ. Implic.*, vol. 5, no. 1, p. 15, Apr. 2020, doi: 10.1186/s41235-020-00218-5.

[7] J. R. Evans and A. Mathur, “The value of online surveys,” *Internet Res.*, vol. 15, no. 2, pp. 195–219, Jan. 2005, doi: 10.1108/10662240510590360.

[8] K. Shabani, M. Khatib, and S. Ebadi, “Vygotsky’s Zone of Proximal Development: Instructional Implications and Teachers’ Professional Development,” *Engl. Lang. Teach.*, vol. 3, no. 4, p. p237, Nov. 2010, doi: 10.5539/elt.v3n4p237.

[9] “Zone of proximal development,” *Wikipedia*. Oct. 29, 2022. Accessed: Nov. 19, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Zone_of_proximal_development&oldid=1118875301#cite_ref-ReferenceA_1-0

[10] P. M. Newton and A. Salvi, “How Common Is Belief in the Learning Styles Neuromyth, and Does It Matter? A Pragmatic Systematic Review,” *Front. Educ.*, vol. 5, 2020, Accessed: Nov. 19, 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/feduc.2020.602451

[11] J. Cuevas, “Is learning styles-based instruction effective? A comprehensive analysis of recent research on learning styles,” *Theory Res. Educ.*, vol. 13, no. 3, pp. 308–333, Nov. 2015, doi: 10.1177/1477878515606621.

[12] B. A. Rogowsky, B. M. Calhoun, and P. Tallal, “Providing Instruction Based on Students’ Learning Style Preferences Does Not Improve Learning,” *Front. Psychol.*, vol. 11, p. 164, 2020, doi: 10.3389/fpsyg.2020.00164.

[13] D. Kolb, *Experiential Learning: Experience As The Source Of Learning And Development*, vol. 1. 1984.

[14] “How Do I Address Learning Styles in My Course? | TLPDC Teaching Resources | Teaching Resources | TLPDC Home | TTU.” https://www.depts.ttu.edu/tlpdc/Resources/Teaching_resources/TLPDC_teaching_resources/LearningStyles.php (accessed Nov. 19, 2022).

[15] A. K. Goel and L. Polepeddi, “Jill Watson: A Virtual Teaching Assistant for Online Education,” Georgia Institute of Technology, Technical Report, 2016. Accessed: Sep. 19, 2022. [Online]. Available: https://smartech.gatech.edu/handle/1853/59104

[16] A. S. Hashim, W. A. Awadh, and A. K. Hamoud, “Student Performance Prediction Model based on Supervised Machine Learning Algorithms,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 928, no. 3, p. 032019, Nov. 2020, doi: 10.1088/1757-899X/928/3/032019.
[17] A. K. Hamoud, A. S. Hashim, and W. A. Awadh, “Predicting Student Performance in Higher Education Institutions Using Decision Tree Analysis,” *Int. J. Interact. Multimed. Artif. Intell.*, vol. 5, no. Special Issue on Big Data and Open Education, 2018, Accessed: Sep. 25, 2022. [Online]. Available: https://www.ijimai.org/journal/bibcite/reference/2654
[18] R. E. Ogunsakin, S. Moyo, Oludayo, O. Olugbara, and C. Israel, “Relating Student Engagement Indicators to Academic Performance Using Multiple Correspondence Analysis,” *Cybern. Inf. Technol.*, vol. 21, no. 1, pp. 87–102, Mar. 2021, doi: 10.2478/cait-2021-0007.