TURO: A Web-Mobile based Application used as a platform for Alternative Learning System (ALS) students to conveniently access learning materials in the different specified courses

---

A Research Project Presented to the Faculty of

School of Computer Studies

University of San Jose – Recoletos

Cebu, Philippines

In partial fulfillment

Of the Requirements for the Degree of

Bachelor of Science in Information Technology

---

by

Padigos, Krystel Nina P.

Pinote, Sean Milbert H.

Mr. Roderick Bandalan

Faculty Adviser

May 2023

ENDORSEMENT

PASSED

ACKNOWLEDGEMENT

ABSTRACT

Numerous people in the Philippines experience the lack of opportunity to be educated. However, not everyone loses the hope to finish their studies regardless of what factors there might be that would stop their education. The Department of Education created a program that would help the Out of School youth and Adults (OSYA) to finish their studies and give them more opportunities through the knowledge that they have already acquired. This program is called the Alternative Learning System program. The researchers in this study implemented an application that aims to help the ALS students and teachers to have a better way of learning through an application that serves as a Learning Management System specifically made for the ALS Program which is called TURO.

TURO is a web and mobile based system that aims to help the ALS Program in the aspect of their learning materials. The researchers used PHP Laravel, Javascript, and HTML & CSS for the web development which would be accessed by the General Admin which serves as the main Admin of DepED in the whole Philippines and the Regional Admin which serves as the admins per Region in the Philippines. Aside from the General and Regional Admins, there are also other users who can access the web development of the application and these are the Teachers and the students. The researchers also used Dart Flutter for the mobile development of the application and it is only accessed by the students. For the User Interface database of the TURO application, the researchers used MYSQL Workbench.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I: INTRODUCTION

Rationale of the Study

The Philippines encounters numerous social issues and crises wherein over the years and up to this point hasn't been fully solved yet and the effects of these social issues is like a domino since all of them are mostly connected. One example of this is the Philippine's Poverty and Education crisis. According to the Philippine Statistics Authority (PSA), about nine percent of Filipinos aged 6 to 24 were out-of-school children and youth (OSCY) based on the 2017 Annual Poverty Indicators Survey (APIS). OSCY are Filipinos who are not attending formal school, who are currently out of school, not gainfully employed, and have not finished college or postsecondary courses [1].

Education is the development of a person towards the goal of understanding and learning how our society and its system works by means of hard work, determination, skills and human empowerment. Education doesn't force anybody to take the opportunity, but it only requires the willingness of a person no matter what their social status and capability is. That's why it comes with a great responsibility and a right for every citizen to be educated. However, due to several social factors assessing the quality education such as Lack of Learning Materials, Poverty, Child Labor, People in Conflict area and Lack of Learning facility, students are no longer prioritizing the benefit of their bright future ahead but instead they only think of making a living as a form of survival in the midst of these social issues.

Moreover, Education comes with a great challenge especially for out-of-school youth and adults (OSYA) [1]. These people are usually in need of basic literacy skills particularly in reading, writing and simple computation but since environmental factors including the effects of emotional abuse and physical bullying in the schools, which is not equitable for them, they are forced to avoid classes and proceed with their everyday life. Furthermore, a government agency in the Philippines

introduced a parallel learning system which is a viable alternative to the existing formal education system. This system serves as a second chance education program where OSYA learners will have a chance to continue learning in a place and time suitable to their circumstances. This would enable them to surpass what they're going through with life and also prevent them from being bullied by the current society.

Theoretical Background

This study is anchored in four theories, namely: Connectivism Theory by George Siemens and Stephen Downes. E-Learning Theory and Multiple Intelligences by Howard Gardner Theory

Connectivism Theory is a relatively new learning theory that is strongly influenced by technology. This encourages students to incorporate thoughts, theories, and general information in a useful manner and too often find sources of information that are accurate and updated. The rise of technology gives a big impact to this theory thus it is accepted to have a major part of the learning process and that people's constant connectedness brings opportunities to successful learning choices [2]. According to connectivism, learning is not just merely our own internal knowledge but it is a part of the decision making on which medium a student should take as a source of information and connect previous information with current information to create new understandings [3].

Connectivism Theory also promotes learning through social media, online networks, blogs, or information databases as long as these sources are thoroughly studied and are accurate since knowing how and where to find the best information is as important as the information itself.
According to the founders of Connectivism Theory, George Aiemens George Siemens and Stephen Downes, connectivism is when a person seeks a solution for a problem from digital technology or to the internet [4].

Meanwhile, Mayer, Sweller and Moreno's E-learning theory is a new system of learning that emphasizes how the use of technology for education can develop an effective learning. The theory was developed from eleven design principles which were created based on Cognitive Load Theory which states that the amount of mental effort involved in working memory has limited capacity. If learners are gained with numerous information and knowledge, the brain will suffer from overload causing inefficient learning that's why it is relevant to balance the amount of information and the functionalities of the brain [5]. With the statement of cognitive load theory, these eleven principles were created to balance the way of learning at an appropriate level especially for learners using technology.

In the elaboration of the Connectivism Theory, E-learning theory belongs to the grand theory of Connectivism since it points out how the rise of technologies can be utilized to create new learning opportunities and to promote effective learning. In addition, the usage of digital devices or multimedia in a way of learning is one specific principle of E-learning theory, and it insists that learning can be promoted through audio, visual and text formats since children today are mostly visual learners [5].

Moreover, the theory of multiple intelligence proposes that humans achieve understanding through a wide range of abilities. Students have different capabilities and ways of understanding a certain concept however it doesn't necessarily mean that one is not smart while another one possesses more intelligence. In this theory, students may learn from various methods and excel in areas which are in accordance with their capabilities and understand the concept at a deeper level [6].

Review of Related Works

According to an article from Philippine News Agency Under the law, the Alternative Learning System (ALS) program started in 2004 by the Department of Education in the Philippines and is expanded and strengthened over the years wherein their goal is to provide increased opportunities for out-of-school youth and adult (OSYA), to acquire their right for education and develop basic literacy and life skills [7]. The scope of ALS does not limit learners from all over the country since the law guarantees access for all learners who are willing to take ALS including those who reside in the unreached communities. Students enrolled in ALS take the ALS Accreditation and Evaluation (A&E) Exam, offered by the government and these learners are classified into three levels which are the ALS Elementary and ALS Secondary. ALS also promotes lifelong learning opportunities now that with the K to 12 Basic Education Curriculum (BEC) students are able to see employment once they have graduated Senior High School. Moreover, based on a study conducted by researchers from Manila, ALS is not just a good way to learn basic academic education but it also shapes people who have violent pasts into better individuals. Taking proper education, May it be formal or non-formal through ALS, can make bystanders into employed people and successful individuals therefore we can conclude that education plays a very important role in the development and progress of an individual as well as for a certain country.

In addition, the materials used for the participation of ALS has always been in a traditional way. However, with the advancement of technology followed by the extension of the Enhanced Community Quarantine (ECQ) due to COVID-19, the need for more online learning opportunities is more apparent. In response, UNICEF Philippines established the ICT4ALS website to offer ALS students, instructors, and implementers access to the usage of internet tools and applications and also provide links to webinars, courses, and tutorials that ALS teachers can use to learn more about ICT-based distant teaching and learning.

Project Objectives

The main objective of this research is to develop an application that will provide a convenient way of learning and accessing modules for Alternative Learning System (ALS) students. This system, TURO, is a web and mobile application where the researchers specifically aims to:

1. Develop a mobile and web application that is accessible by the ALS teachers and ALS students.
2. Implement an Enrollment feature as a gateway for students to access the contents of the application.
3. Develop functionalities on managing the Learning Materials to be created by the teachers and viewed by the students that would use the application.
4. Integrate a Quiz and Assignment System where the students can assess their learning through the quiz and assignments posted by the teachers.

Project Scope and Limitation

This study develops a mobile and web application that specifically uses flutter and Laravel as a means of implementing the project application. The system includes the creation of courses and modules for the ALS teacher's end while for the ALS student's end, the system allows them to access the different courses provided by the teachers. This study focuses on ALS takers both young and adult people who have stopped their studies but are willing to finish their education and specifically limited to areas in the Philippines.

Significance of the Study

Alternative Learning System (ALS) was developed in 2016 to provide opportunities to Out of School Young and Adults (OSYA) provided with face to face classes and printed learning materials. However, as ECQ was implemented, social interactions were limited. That's why

technology takes part as a new learning material. With that, this study enables ALS students to have a convenient way of learning that can be less time consuming. This study can be relevant not just to students and educators but to the whole Department of Education in the Philippines to ensure that materials for Alternative Learning Systems are provided through technology since we are now in the modern and advanced world. This study accommodates ALS students through providing them with paperless modules/reviewers and a self-evaluation as a way to assess their learnings and knowledge with the different courses provided.

Research Methodology

In the study, the expected features to have for the Web application, the teacher's side is the creation of Classes, Courses, Modules, Topics, Topic Content (Text, File, Quiz, and Assignment), and Announcement as well as viewing and checking of student's Quiz attempts and Assignment submissions. For the student's side is the enrollment and viewing of all learning materials posted by the teachers. Enrollment of the ALS students requires them to provide their LRN, student's personal details, family background, educational background, and documents that are required by the ALS [8]. After enrollment, the student must wait for the teacher to approve the student's application. After the student is done with taking the Final exam physically at the ALS Learning Centers. The teacher can indicate in the app if the student has passed their certain program or not.

CHAPTER II: SOFTWARE REQUIREMENTS AND DESIGN SPECIFICATION

This chapter specifies the user and system requirements that are expected to be accomplished as well as the structure and process of achieving these. It contains sections for the Use Case Diagram, Use Case Narrative, Activity Diagram, ERD and User Interface Design.

Architectural Diagram

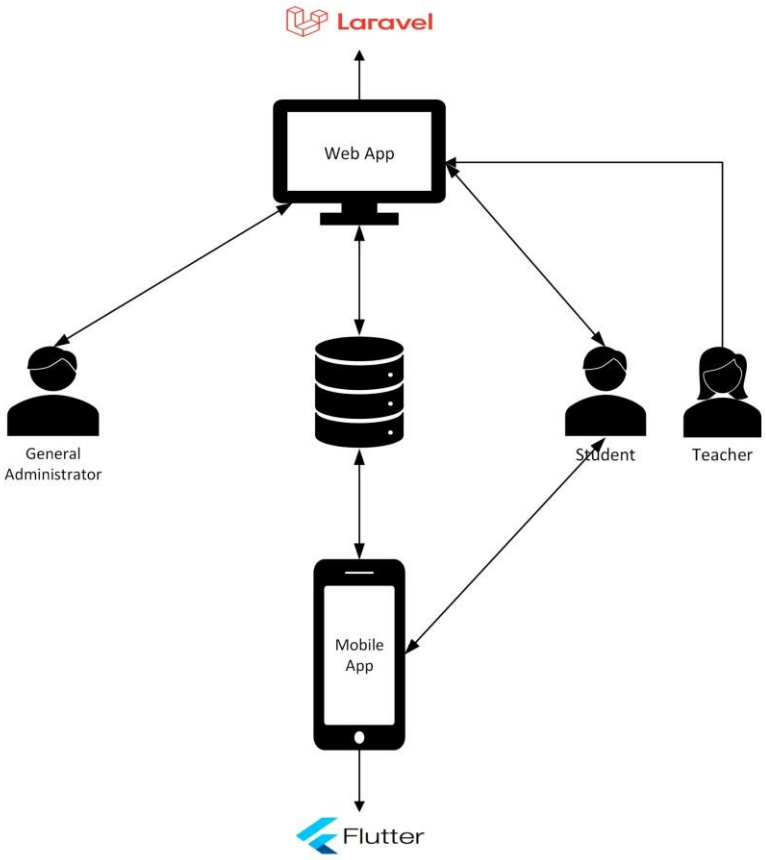

Figure 1: Architectural Diagram

The figure above (Figure 1) shows how the teachers and students interact with the mobile and web application platforms of TURO. The data can be viewed real time in the database. The administrator manages the users in the application.

Use Case Diagram

The use case diagram shows the relationship between actors and different use cases of the systems in which an actor is involved. It also shows an overview of the system in a high-level design. Each use is further detailed in its narrative.

Figure 2: Use Case Diagram

The figure above (Figure 2) defines the different use cases in which the TURO system performs when users, the teacher and student, create learning materials in the web app and accessed by students through the mobile app as well as in the web app.

Use Case Narrative

The use case narrative of TURO shows the main success scenario and alternative flows of a user case in web and mobile to achieve a specific objective. It provides more details about the use case.

Use Case: *Manage Users*

Actor(s): General Admin	
Description: The General Admin can add a Regional Admin and view its details.	
Precondition(s): The General Admin must login.	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. General Admin clicks the “Regional Admin” tab. 2. General Admin clicks the “Add Regional Admin” button. 3. General Admin inputs the details of the Regional Admin	1. The system displays the Regional Admin table. 2. The system displays a modal to enable/disable the user. 3. The system saves the details in the database
ALTERNATIVE FLOW OF EVENTS	
1. Add an existing user.	Prompts error message “username already exists” / “email already exists”

Table 1: Manage Users

Use Case: *Manage Admin Announcement*
Actor(s): Regional Admin
Description: The Regional Admin can create, edit and delete an announcement designated for the whole classes of ALS Program.
Precondition(s): The Admins are logged in.
Post condition(s): None

FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Regional Admin clicks “Announcement” tab 2. Regional Admin clicks “Add Announcement”, “Edit” & “Delete” button 3. Regional Admin input the Announcement details	1. The system displays the Announcement Page. 2. The system displays a modal to add, edit & delete an announcement. 3. The system saves the announcement details in the database.
ALTERNATIVE FLOW OF EVENTS: None	

Table 2: Manage Admin Announcement

Use Case: *Manage Programs*	
Actor(s): General Admin	
Description: The General admin can add & delete the ALS program that is currently offered by ALS in a specific Region.	
Precondition(s): The Admins are logged in.	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Admins click “Programs” button 2. Admins click “Add Program” & “Delete” button 3. Admins input the Program details	1. The system displays the table containing the programs. 2. The system displays a modal to add & delete a program. 3. The system saves the program details in the database.

ALTERNATIVE FLOW OF EVENTS: None

Table 3: Manage Program

Use Case: *Manage Learning Centers*	
Actor(s): General Admin & Regional Admin	
Description: The Admins can add & delete ALS Learning Centers that are currently available in a specific Region in the Philippines.	
Precondition(s): The Admins are logged in.	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Admins click “Learning Centers” button 2. Admins click “Add Learning Centers” & “Delete” button 3. Admins input the Learning Centers details	1. The system displays the table containing the Learning Centers. 2. The system displays a modal to add & delete a Learning Center. 3. The system saves the Learning Center details in the database and adds the Learning Center in the table.
ALTERNATIVE FLOW OF EVENTS: None	

Table 4: Manage Learning Centers

Use Case: *Sign Up*
Actor(s): Teacher & Student

Description: Teacher & Student registers to TURO App	
Precondition(s): None	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Actors input the registration details. 2. Actors click “Sign Up” button	Data is saved to the database.
ALTERNATIVE FLOW OF EVENTS	
1. Submits existing username and email 2. Submits blank field	1. Error “username already exists” / “email already exists” 2. Error “the field is required”

Table 5: Sign Up

Use Case: *Student Enrollment*	
Actor(s): Student	
Description: Student enrolls to a program in ALS through TURO app	
Precondition(s): Student must be logged in	
Postcondition(s): Student must be approved by the teacher	
FLOW OF EVENTS	
Actor Action:	**System Response:**

1. Student clicks the “Enrollment” tab. 2. Student clicks “Enroll now!” button 3. Students input their personal information including family background, educational background and required documents to be passed. 4. Student clicks “Submit” button	1. System displays the Enrollment Steps Page. 2. System displays Enrollment Form Page. 3. Data is save to the database 4. System displays the pending/waiting for approval page.
ALTERNATIVE FLOW OF EVENTS	
1. Submits empty field	1. Error “the field is required”

Table 6: Student Enrollment

Use Case: *Manage Class*	
Actor(s): Teacher	
Description: Teacher can add, edit, duplicate and hide a Class	
Precondition(s): Teacher must be logged in	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks the “Create Class” & “Edit” button. 2. Teacher clicks the “Hide” & “Duplicate” icons.	1. The system displays a modal to create & edit a class. 2. The system displays a modal to confirm Hide and Duplicate a class.
ALTERNATIVE FLOW OF EVENTS	
1. Submits empty field	1. Error “the field is required”

Table 7: Class Creation

Use Case: *Assign Student to a Class*	
Actor(s): Teacher	
Description: Teacher assigns the enrolled student to a certain Class	
Precondition(s): Teacher must create a Class, Student must be enrolled in ALS	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks “Manage enrollees” 2. Teacher clicks “Student settings”	1. System displays manage enrollees page
button	2. System displays a modal to assign the student to a class.
ALTERNATIVE FLOW OF EVENTS: None	

Table 8: Assign Student to a Class

Use Case: *Manage Course*
Actor(s): Teacher
Description: Teacher can add, edit and delete a Course
Precondition(s): Teacher must create a Class

Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks the “Add Course”, “Edit”, & “Delete” button. 2. Teacher clicks “Create” button	1. System displays a modal to create, edit, and delete a course. 2. System saves course to the database
ALTERNATIVE FLOW OF EVENTS	
1. Submits empty field	1. Error “the field is required”

Table 9: Manage Course

Use Case: *Manage Module*	
Actor(s): Teacher	
Description: Teacher can add, edit, and delete a Module	
Precondition(s): Teacher must create a Course	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks “Create Module”, “Edit, & “Delete” button 2. Teacher clicks “Create” button	1. System displays a modal to create, edit and delete a module. 2. System reloads page and displays the created module card.

ALTERNATIVE FLOW OF EVENTS: None

Table 10: Manage Module

Use Case: *Manage Topic*	
Actor(s): Teacher	
Description: Teacher can create, edit, & delete a Topic	
Precondition(s): Teacher must create a Module	
Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks “Add Topic”, “Edit Topic” & “Delete” button	1. System displays a modal to create, edit and delete a topic.
ALTERNATIVE FLOW OF EVENTS: None	

Table 11: Manage Topic

Use Case: *Manage Topic Content*
Actor(s): Teacher
Description: Teacher can add, edit & delete a resource inside a content. The teacher can choose to add an HTML Format, Upload a File, add a Quiz and add an Assignment.
Precondition(s): Teacher must create a Topic

Postcondition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Student clicks a Topic 2. Student clicks “Add Resource” button 3. Select a resource type	1. System displays the Topic Contents 2. System displays a dropdown of resources.
ALTERNATIVE FLOW OF EVENTS: None	

Table 12: Manage Topic Content

Use Case: *Manage Quiz*	
Actor(s): Teacher	
Description: Teacher can create a Quiz and add it as a resource in a certain topic. The teacher can also create Quiz Questions, edit Quiz settings, View Student’s Attempts and Mark student’s points.	
Precondition(s): Teacher must create a Course	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks “Quizzes” tab. 2. Teacher clicks the “Create Quiz” button. 3. Teacher clicks the “Add Question” & “Edit Question” button 4. Teacher clicks the “Student’s Attempts” & “Quiz settings” button	1. System displays the Quiz List Page 2. System displays a modal to create a Quiz. 3. System displays the Questionnaire page 4. System displays Quiz Attempts Page and the Quiz Settings page

ALTERNATIVE FLOW OF EVENTS: None

Table 13: Manage Quiz

Use Case: *Take Quiz*	
Actor(s): Student	
Description: Student can take the quiz given by the teacher	
Precondition(s): Teacher must create a Quiz	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Students click the “Quizzes” tab 2. Student clicks the “Quiz Title” 3. Student inputs quiz password 4. Student clicks “Take Quiz” button	1. System displays the Quiz List Page 2. System displays the instruction page 3. System stores the input to database 4. System displays the final Quiz page
ALTERNATIVE FLOW OF EVENTS	
1. Student exits page while quiz in progress	1. Error “quiz in progress”

Table 14: Take Quiz

Use Case: *Manage Assignment*
Actor(s): Teacher

Description: Teacher can create, edit and delete an Assignment. The teacher can also View Submission Student's attempts, edit Assignment Settings, and Mark Student's assignment.	
Precondition(s): Teacher must create a Course	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks "Assignments" tab. 2. Teacher clicks the "Create Assignments" button. 3. Teacher clicks the "Student's Submissions" & "Assignment settings" button	1. System displays the Assignments List Page 2. System displays a modal to create Assignments. 3. System displays Assignment Submissions Page and the Assignment Settings page
ALTERNATIVE FLOW OF EVENTS: None	

Table 15: Manage Assignments

Use Case: *Submit Assignments*	
Actor(s): Student	
Description: Student submits assignments in a specified course topic	
Precondition(s): Teacher must create an Assignment	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**

1. Students click the “Assignments” tab 2. Student clicks the “Assignment Title” 3. Student clicks “Submit” button	1. System displays the Assignment List Page 2. System displays the submit assignment page 3. System stores the submission to database
ALTERNATIVE FLOW OF EVENTS: None	

Table 16: Submit Assignments

Use Case: *Manage Course Announcement*	
Actor(s): Teacher	
Description: The Teacher can create, edit and delete an announcement in each Course	
Precondition(s): The Teacher must login.	
Post condition(s): None	
FLOW OF EVENTS	
Actor Action:	**System Response:**
1. Teacher clicks “Announcement” tab 2. Teacher clicks “Add Announcement”, “Edit” & “Delete” button 3. Teacher input the Announcement details	1. The system displays the Announcement Page. 2. The system displays a modal to add, edit & delete an announcement. 3. The system saves the announcement details in the database.
ALTERNATIVE FLOW OF EVENTS: None	

Table 17: Manage Course Announcement

Activity Diagram

The activity diagram here presents the series of actions and flows of the most prominent features, describing how they interact step-by-step.

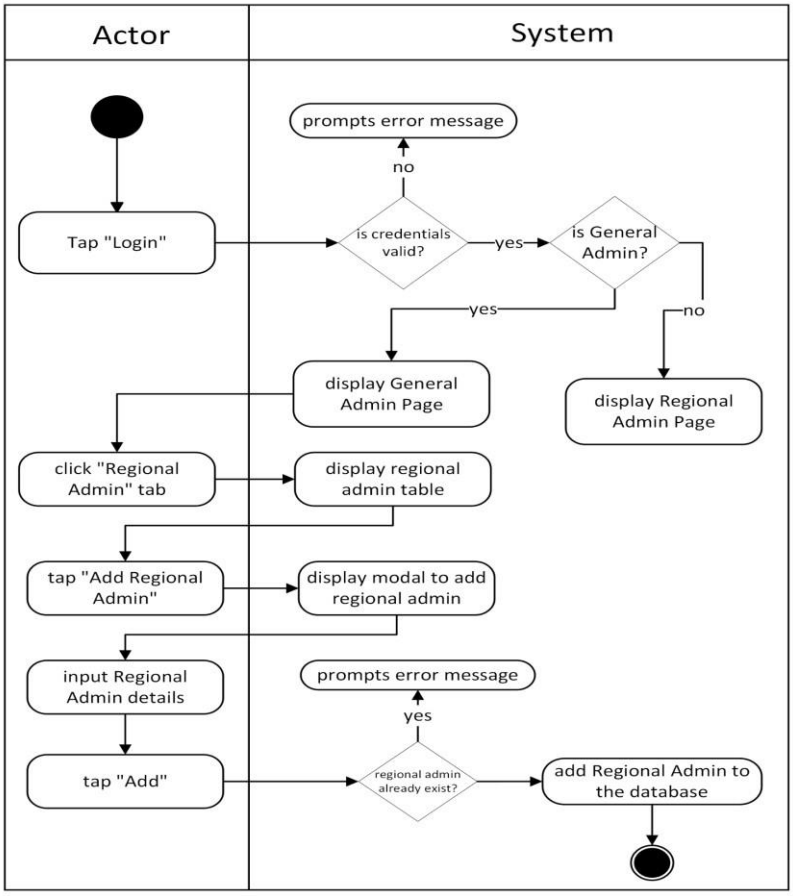

Figure 3: Manage Users

The figure above depicts the process of how the General Admin can add a Regional Admin. After Login, the General Admin can click the “Add Regional Admin” button and the modal will display. Right then the admin must input the Regional Admin details and press the “Add” button next. The system will then validate if the Regional Admin already exists in the database or not. If it does, an error message will display but if not the user will then be created and added to the Regional Admin table.

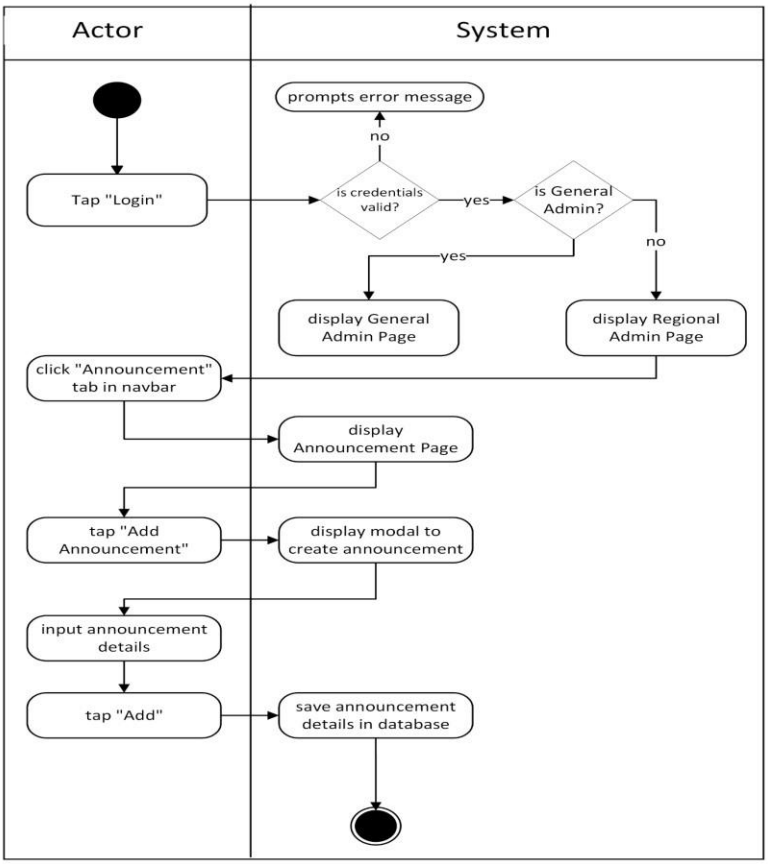

Figure 4: Manage Announcement

The figure above shows the process of how the admin can add an announcement. After login, the regional admin can click the “Announcement” tab in the navigation bar and the announcement page will display. The admin clicks the “Add announcement” and the modal will display and the admin must input the announcement details. Right then the admin must input the user details and press the “Add” button. The system will then add the announcement in the database.

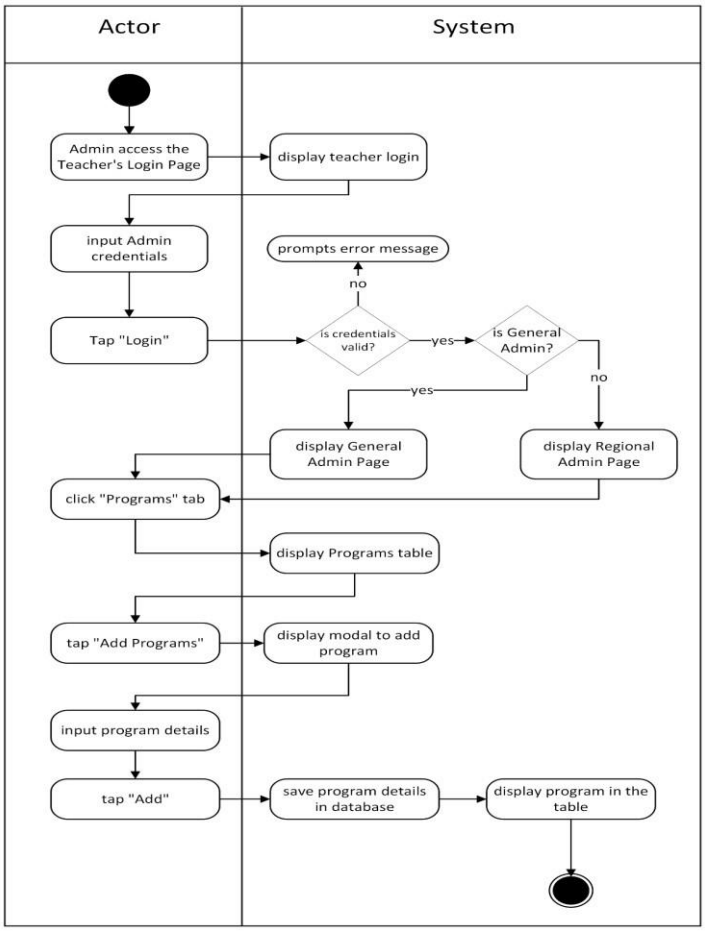

Figure 5: Manage Program

The figure above, shows how the Admins can add an ALS Program. After they are logged in, the admins click the “Programs” tab and the program table page will be displayed. After that the admin clicks the “Add Program” button and the modal to add the programs will display. The admin then inputs the Program Details and clicks “Add”. Then the system will program in the table.

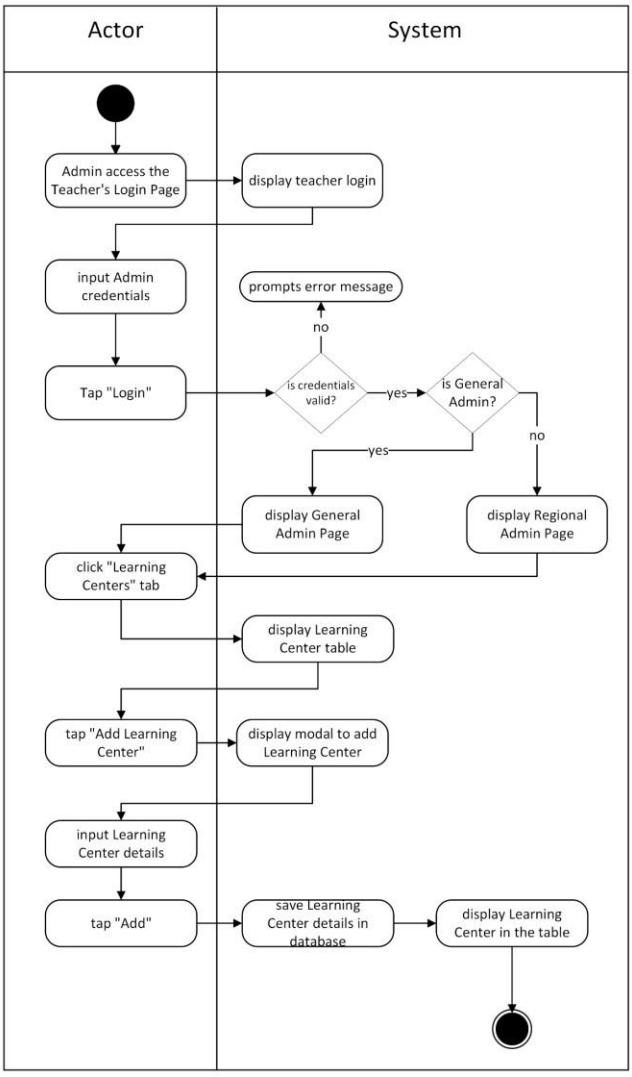

Figure 6: Manage Learning Centers

The figure above, illustrates how the Admins can add an ALS Learning Center. After they are logged in, the admins click the “Learning Center” tab and the program table page will be displayed. After that the admin clicks the “Add Learning Center” button and the modal to add the Learning Center will display. The admin then inputs the Learning Center details and clicks “Add”. Then the system will add the Learning Center in the table.

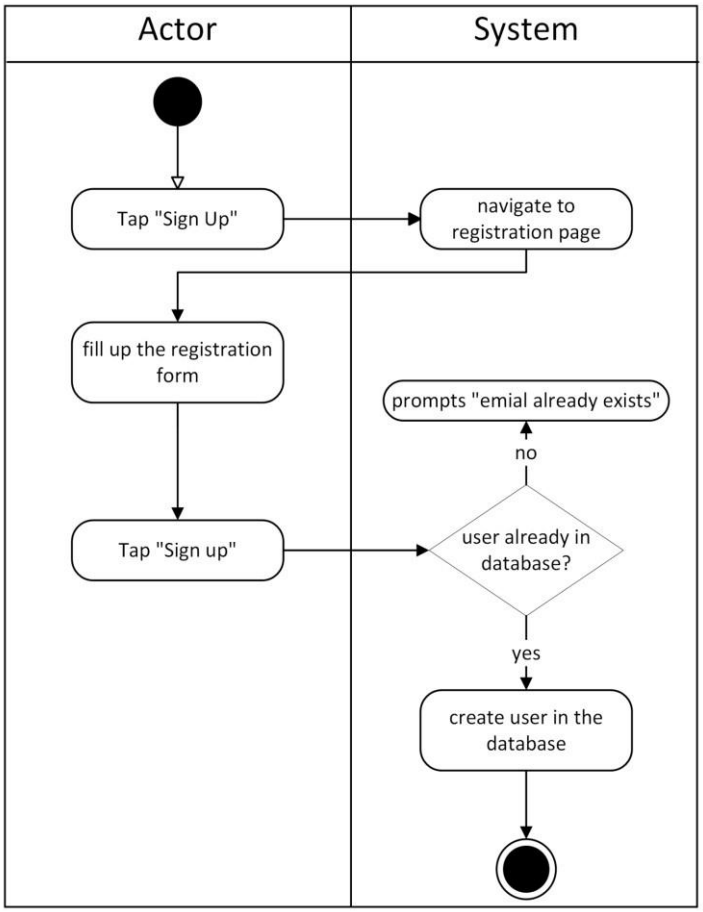

Figure 7: Teacher and Student Sign Up

The figure above demonstrates the flow of teacher and student registration. When the teacher and student clicks the “Sign Up” button, the system will redirect to the Registration page where the teacher and student will input their credentials such as First Name, Middle Name, Last Name, email, username, password and password confirmation. After filling out the registration, the teacher/student will then press the “Sign Up” button and will be registered to the system.

However if the user inputs an existing account, the system will prompt an error message.

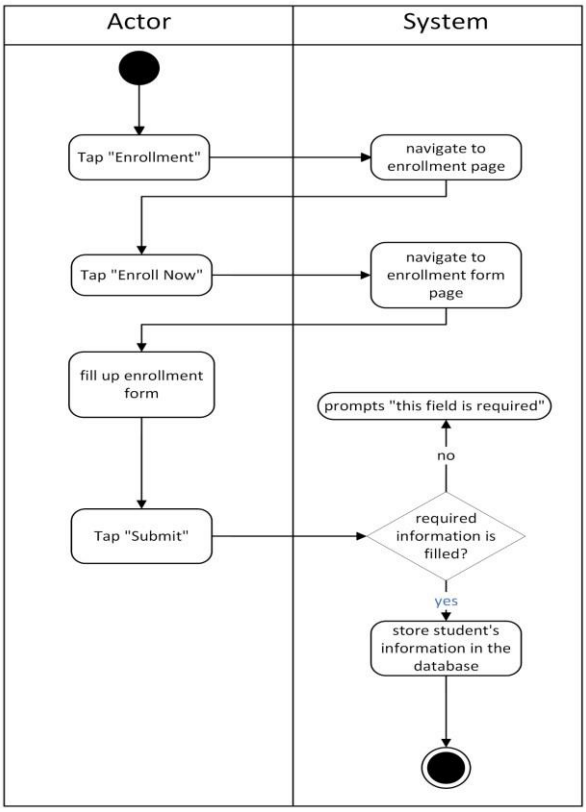

Figure 8: Student Enrollment

The figure above shows the process of how the students can enroll in the TURO ALS System. After registration, the student must enroll first to access the courses in ALS. Students click the “Enrollment” tab and will be redirected to the Enrollment Page where the available ALS teachers can be displayed and the status of the student will be displayed also in this page. The student clicks the “Enroll Now” button and will be redirected to the Enrollment Form Page. The student must fill out the form and click “Submit”. After that, if the required details are all filled out, the student’s details will then be saved to the database and the student must wait for approval from the teacher.

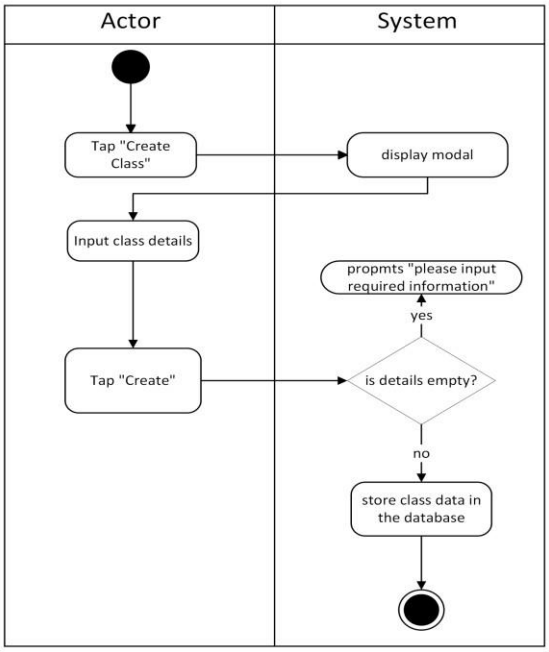

Figure 9: Manage Class

The figure on the left depicts the flow of Class Creation. The teacher presses the Create Class button and after that, a modal will display and the teacher must input a Title and Description of the Class then presses "Create". If the fields are empty, it will prompt an error message otherwise the Class details will be saved in the database.

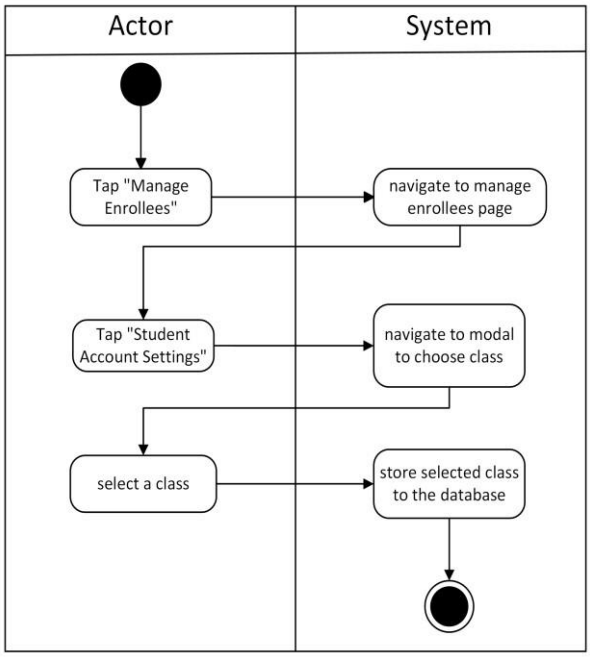

Figure 10: Assign Student to a Class

The figure on the left shows how the teachers assign the student into a specific class. The teacher clicks the "Manage Enrollees" tab. Clicking the "student account settings" will display a modal to choose for the classes. After that, the teacher must select the correct class that the student should attend.

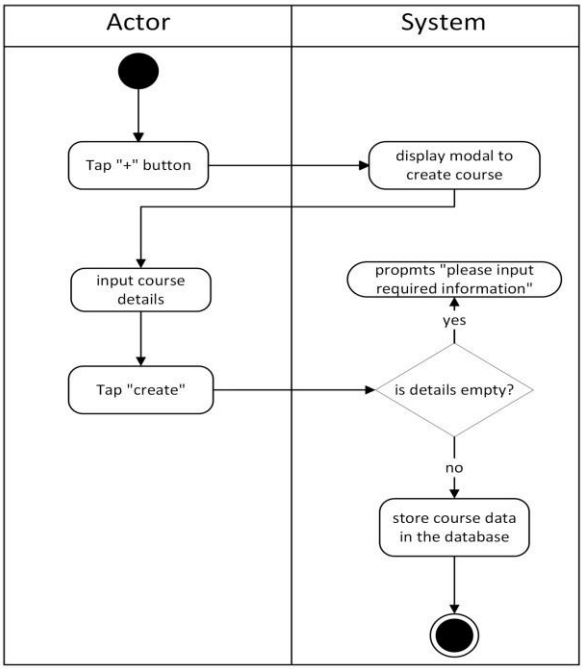

Figure 11: Manage Course

The figure on the left, depicts how the teacher creates a course. The teacher must tap the "+" button then a modal will display containing an input for course title, and description. After that the teacher would click "Create" then the course will be added to the database.

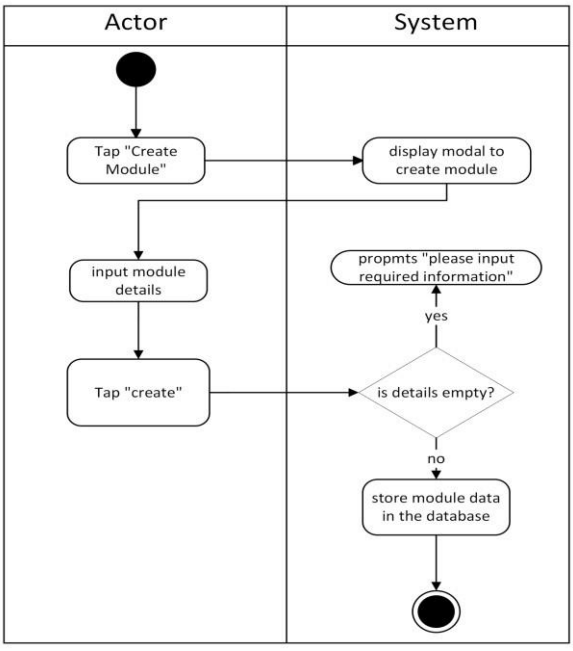

Figure 12: Manage Module

The figure on the left, shows the Module Creation. The teacher must click first the "Create Module" button to display the create module modal. The teacher then inputs the Module title, and description. After that if the teacher fails to enter a title it prompts an error otherwise the data will be stored in the

database.

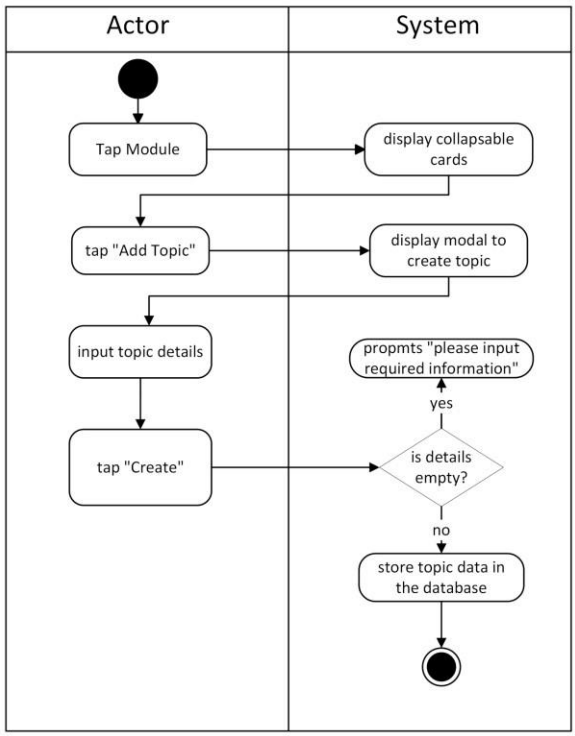

Figure 13: Manage Topic

The figure on the left shows how to create a Topic inside a course. The teacher presses the Module and the collapsible button will display. After that, the teacher clicks “Add Topic” then a modal will display to create a topic. Once the teacher is done with the input of Topic details, teacher clicks “Create” but if there are empty fields the system will prompt an error message, otherwise the data will be stored to database

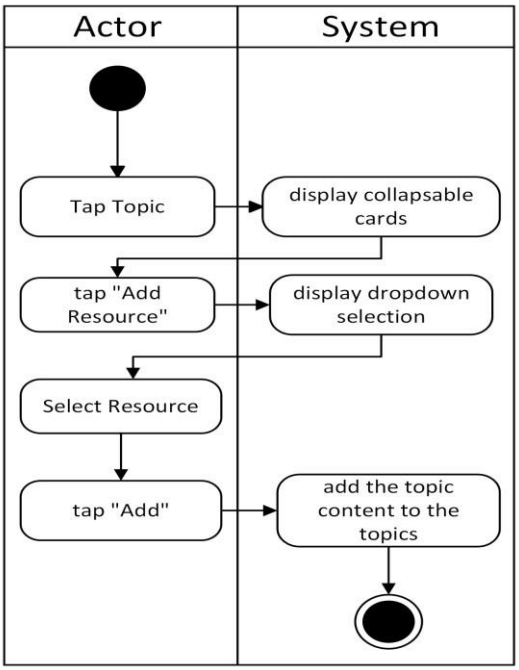

Figure 14: Manage Topic Content The figure on the left, shows how the teacher can add a Resource/Topic content

inside a Topic. First, After creating a Topic, the teacher must click a topic and click the “Add Resource” button. After that a dropdown will display and the teacher will just choose which type he/she will add.

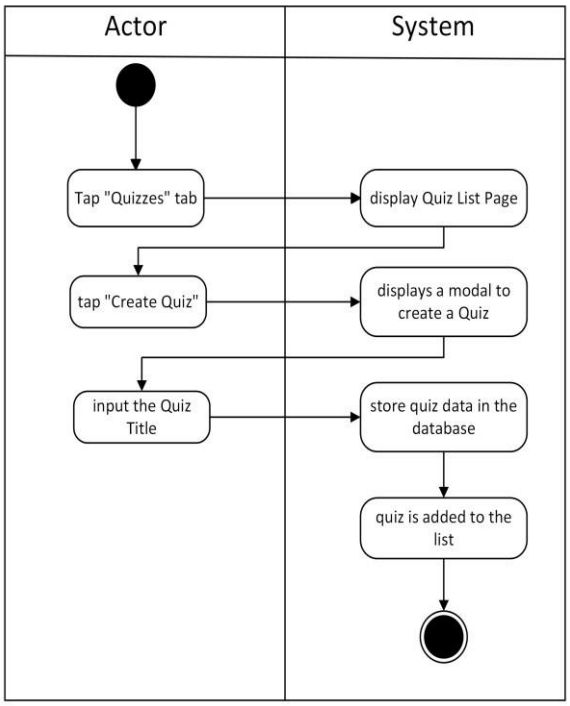

Figure 15: Create Quiz

The figure on the left depicts the steps on how to create a Quiz. Teacher clicks the “Quizzes” tab first. Then the Quiz List Page will be displayed. Later on, the teacher will tap “Create Quiz” then a modal would display to create Quiz, then the teacher will input the Quiz Title. Quiz Data will then be stored in the database and will be added to the Quiz List.

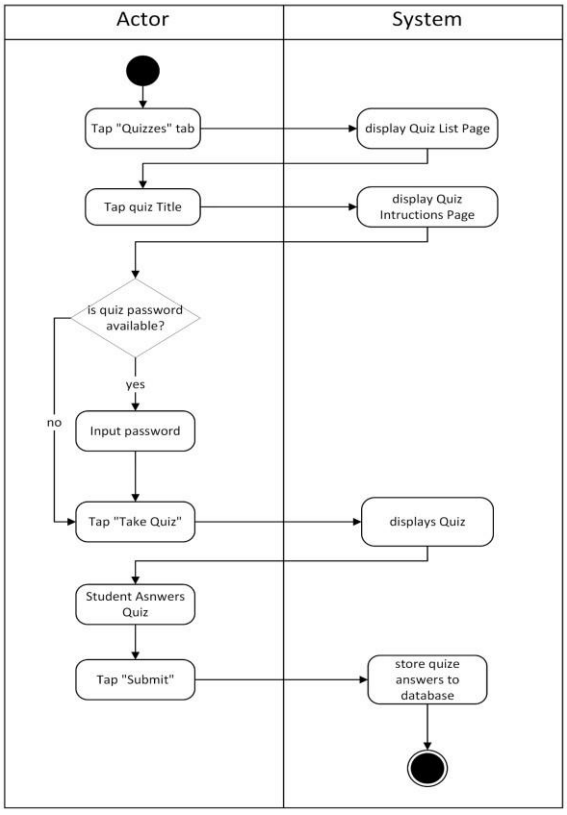

Figure 16: Take Quiz

The figure on the left illustrates the steps of how the student would take a quiz.

First the student clicks the “Quizzes” tab and the Quiz List Page will display. The student will then click the Quiz Title and the Quiz instructions page will display. If there is no quiz password needed, the student will be directed to the Quiz page but if there is a password required, the student must input the correct password and will then be redirected to the Quiz page.

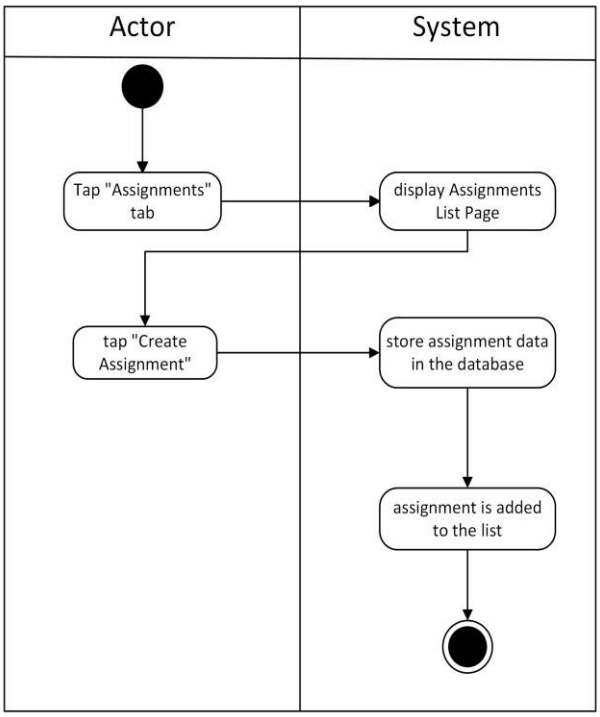

Figure 17: Manage Assignment

The figure on the left shows the steps on how to create an Assignment. First, the Teacher clicks the “Assignments” tab first. Then the Assignment List Page will be displayed. Next, the teacher will tap “Create Assignment” and input the Assignment Title. After that, the Assignment Data will be stored in the database and will be added to the Assignment List.

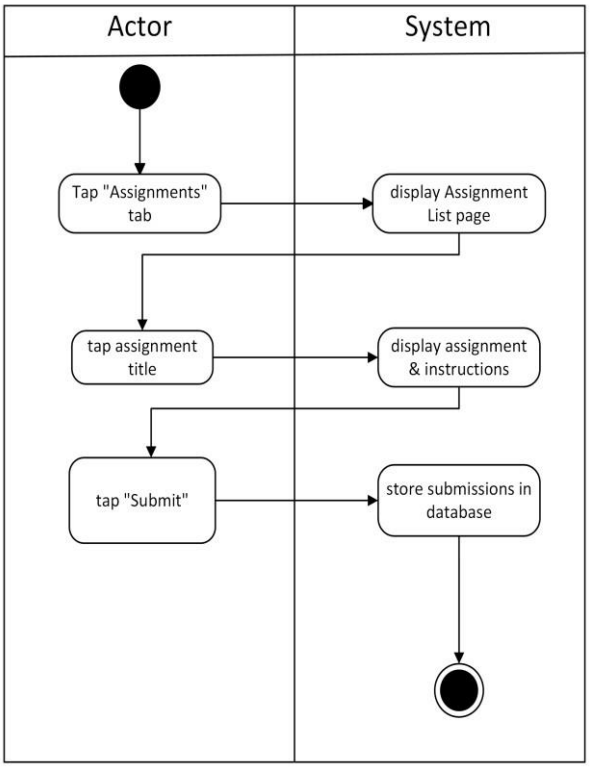

Figure 18: Submit Assignments

The figure on the left depicts how the student can submit an assignment. First the student clicks the “Assignment” tab then the Assignment List Page will be displayed. After that the student will click the assignment title and the assignment and instructions page will be displayed. Once the student is done, he/she can click submit and the system would store the submission in the database

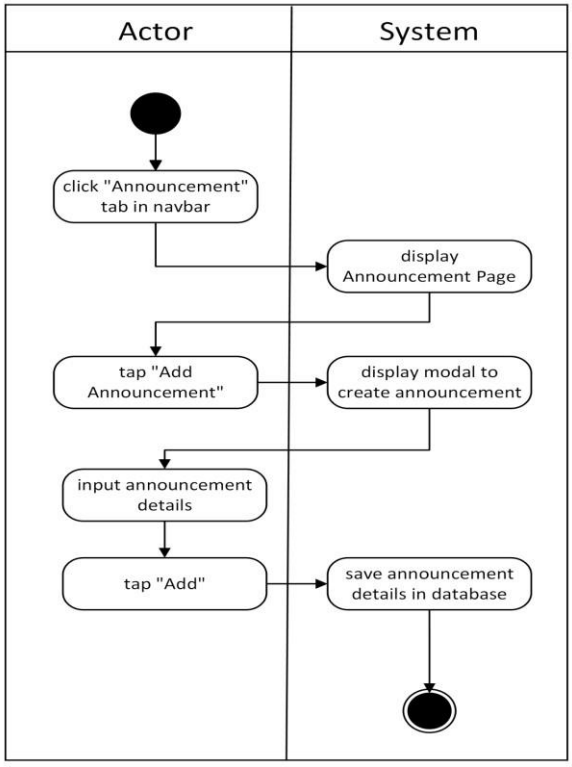

Figure 19: Manage Announcement

The figure on the left shows how the teacher can create an announcement on each course.

After logging in, the teacher clicks the “Announcement” tab. After that the teacher clicks the “Add Announcement” button and the modal will then display. The teacher will then input the announcement details and click the “Add” button and the system will save the announcement details in the database.

Database Design

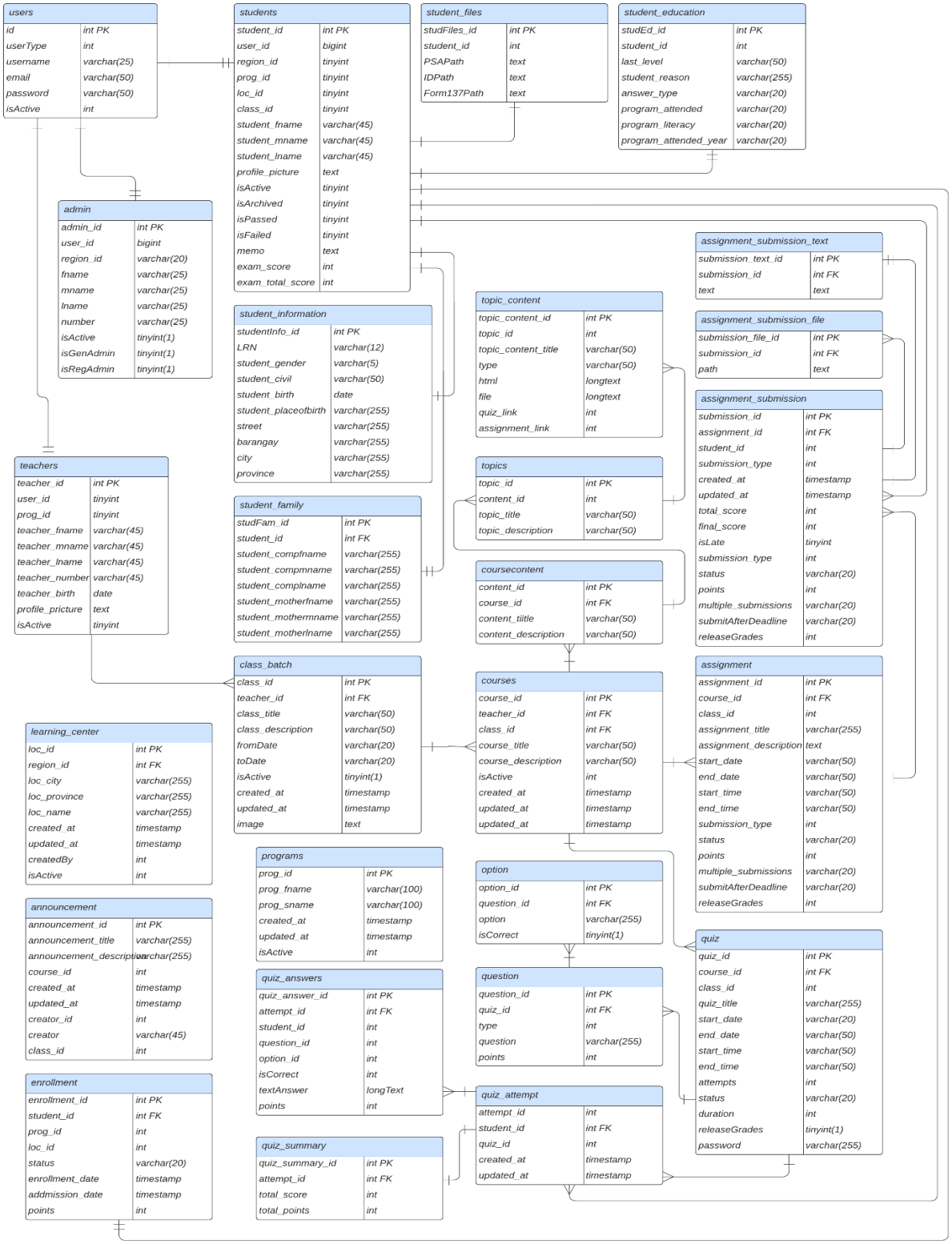

Figure 20: TURO Database Design

The figure above shows the Entity-Relationship Design of the application. These tables are responsible for the Data Manipulation of the application in the database. An ERD (EntityRelationship Design) shows the relationships of entity sets which are stored in a database. These

diagrams illustrate the logical structure of databases.

User Interface for Web Application

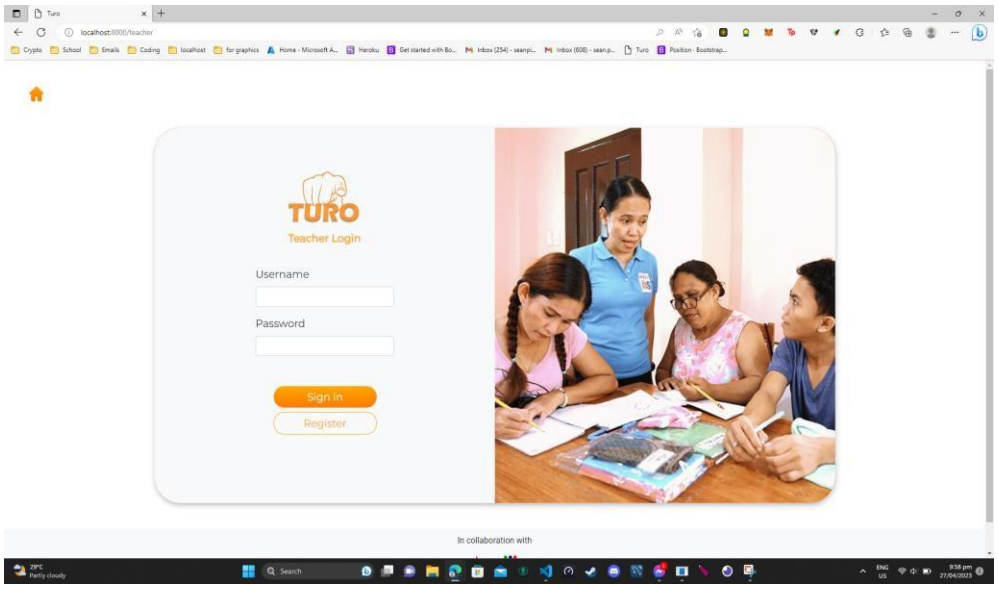

Figure 21: Teacher Login

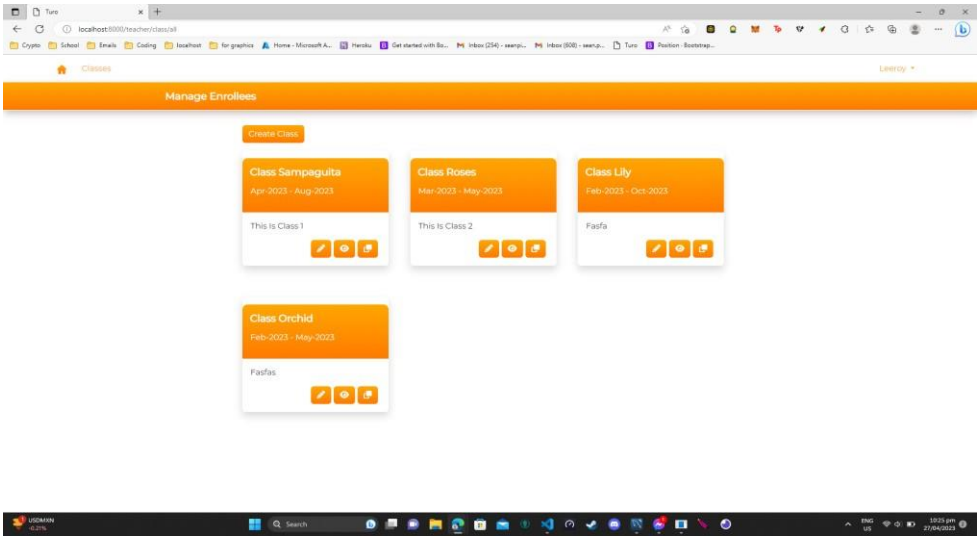

Figure 22: Teacher List of Classes

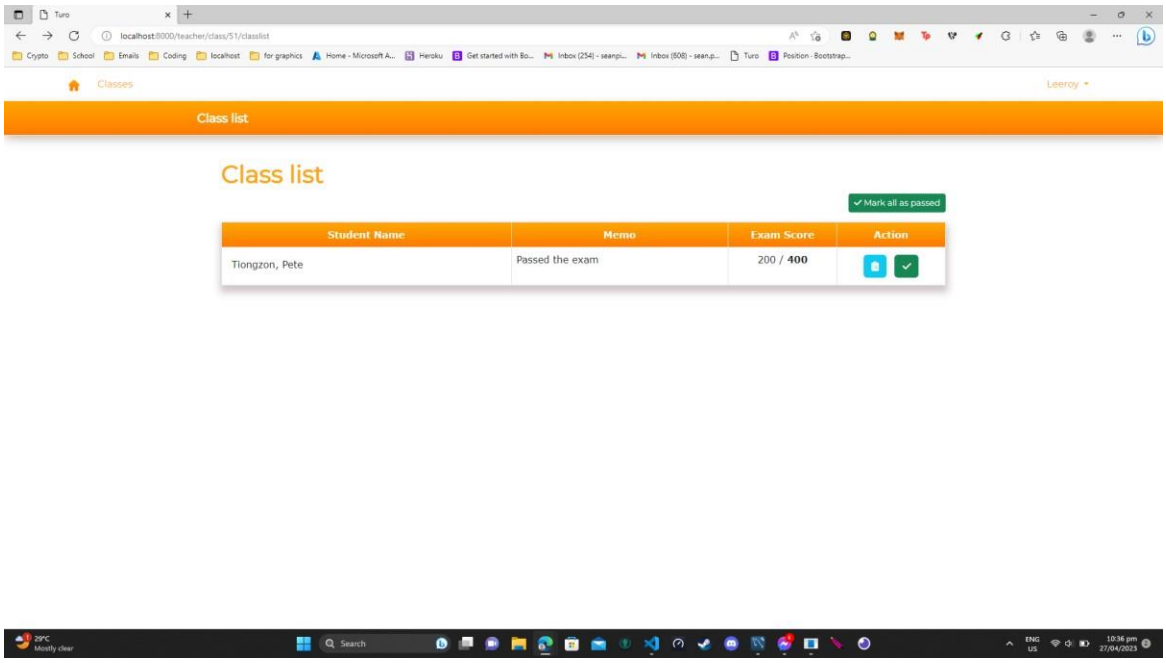

Figure 23: Teacher Class List

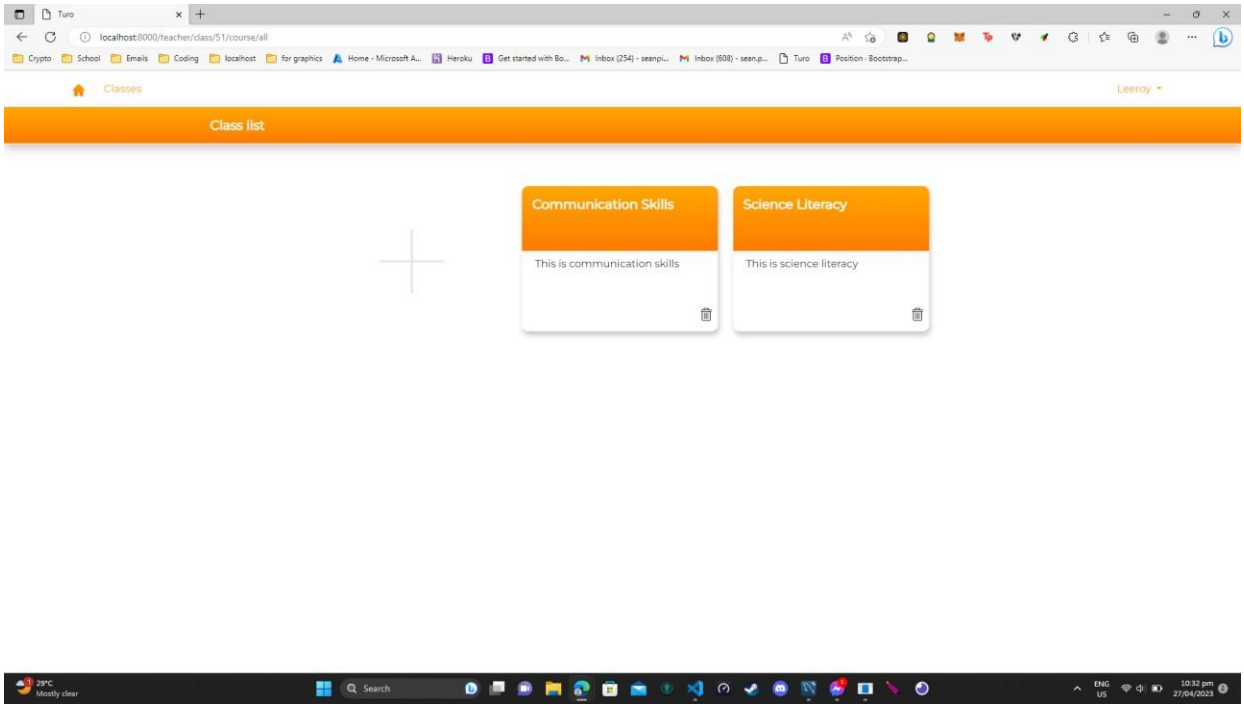

Figure 24: Teacher List of Courses

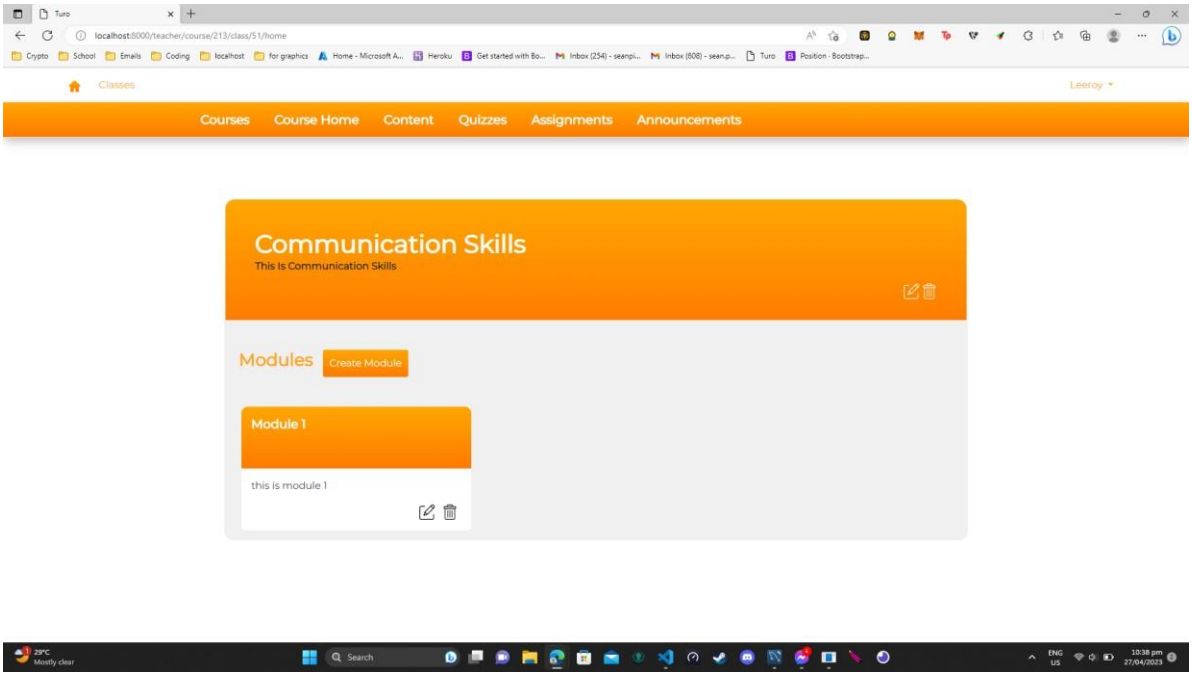

Figure 25: Teacher List of Modules

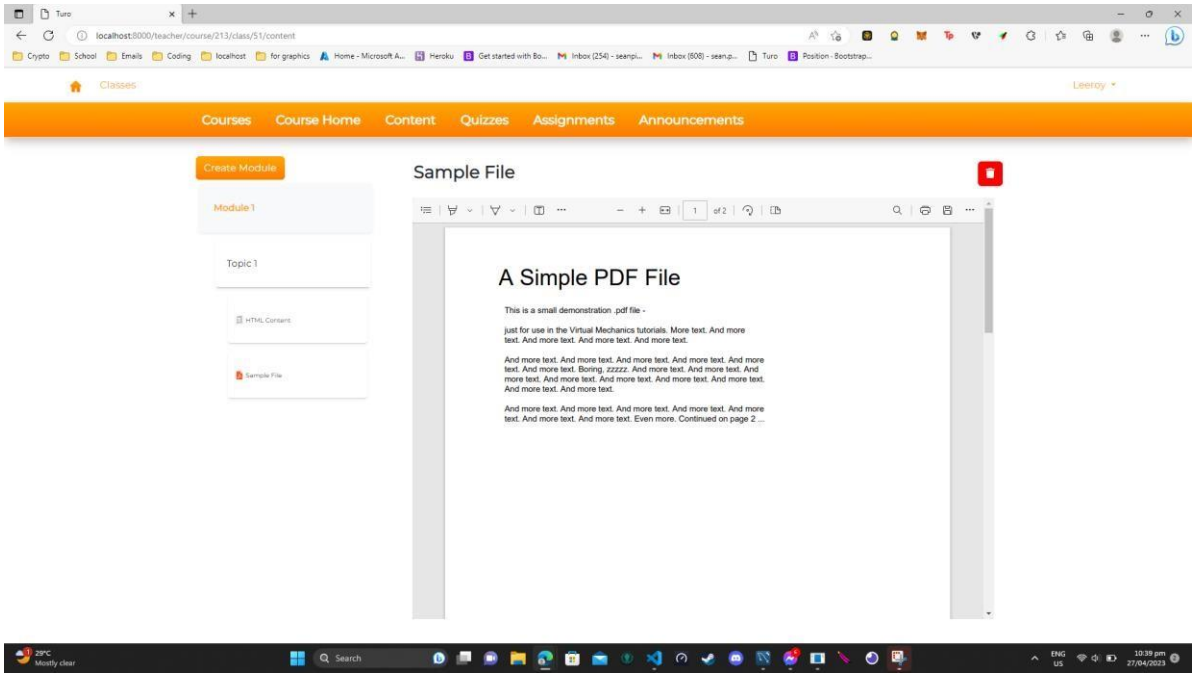

Figure 26: Teacher Topic and Topic Content

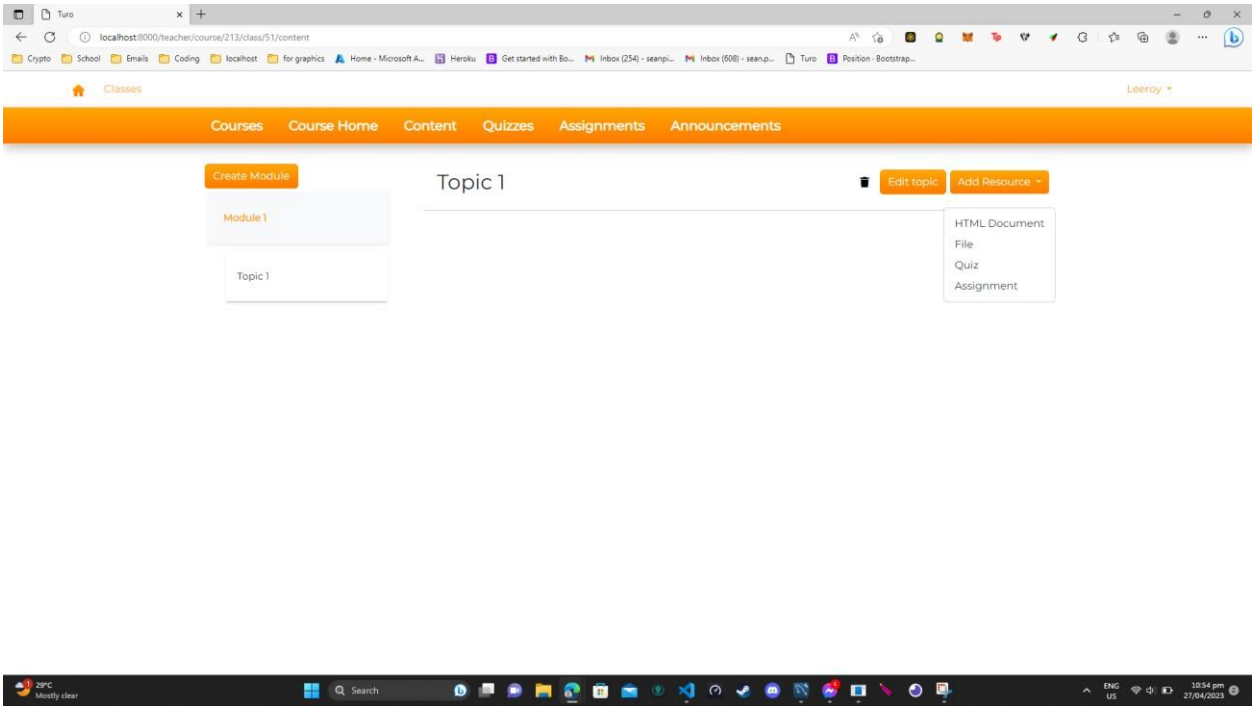

Figure 27: Teacher Add Topic Content/Resource

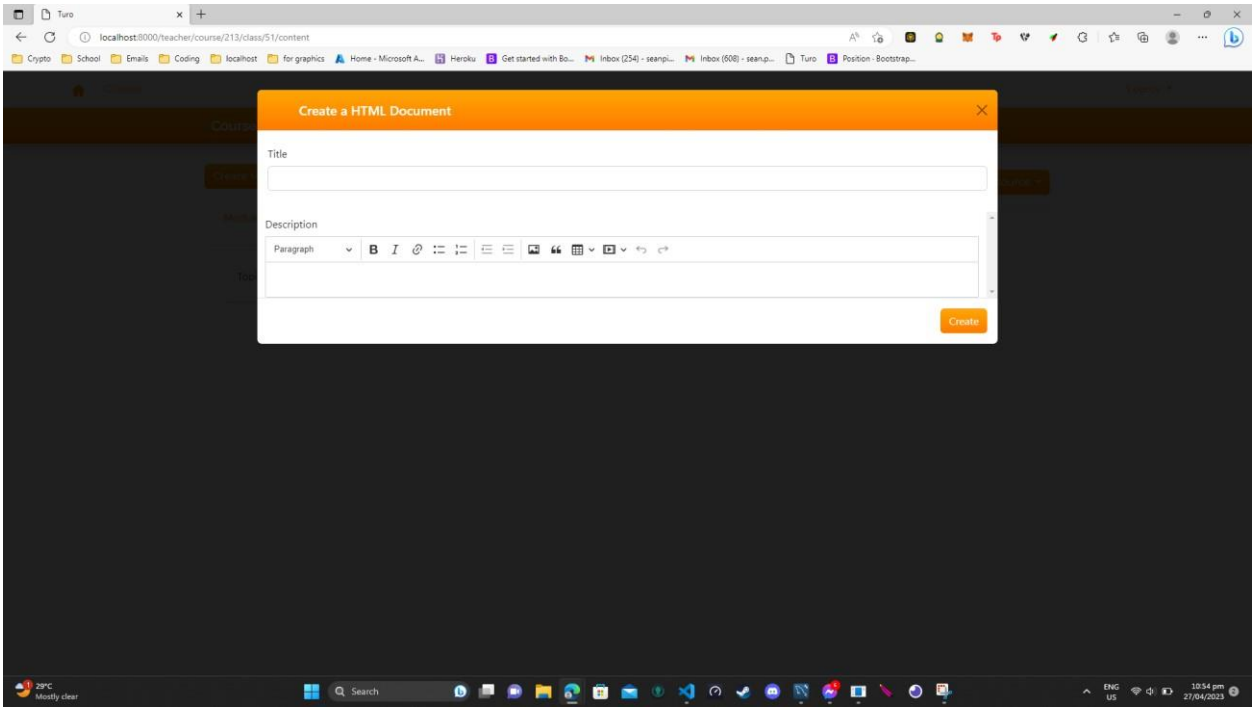

Figure 28: Teacher Create HTML Document

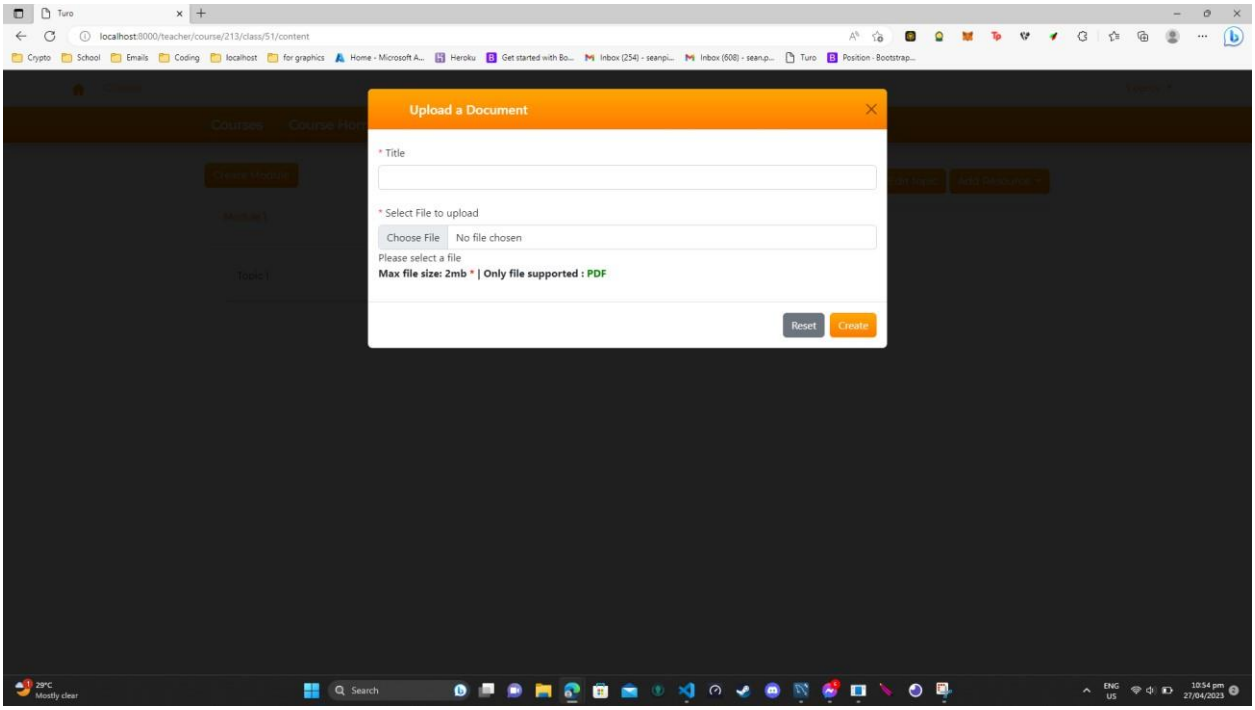

Figure 29: Teacher Upload Document

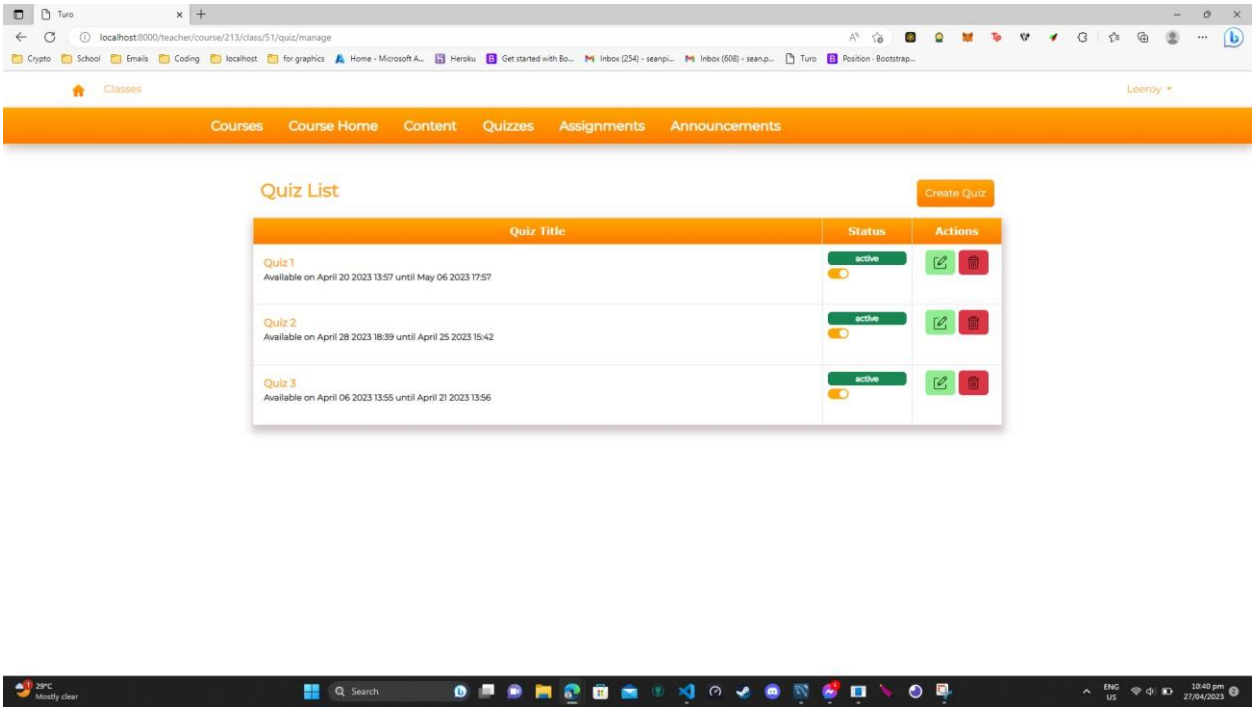

Figure 30: Teacher Quiz List

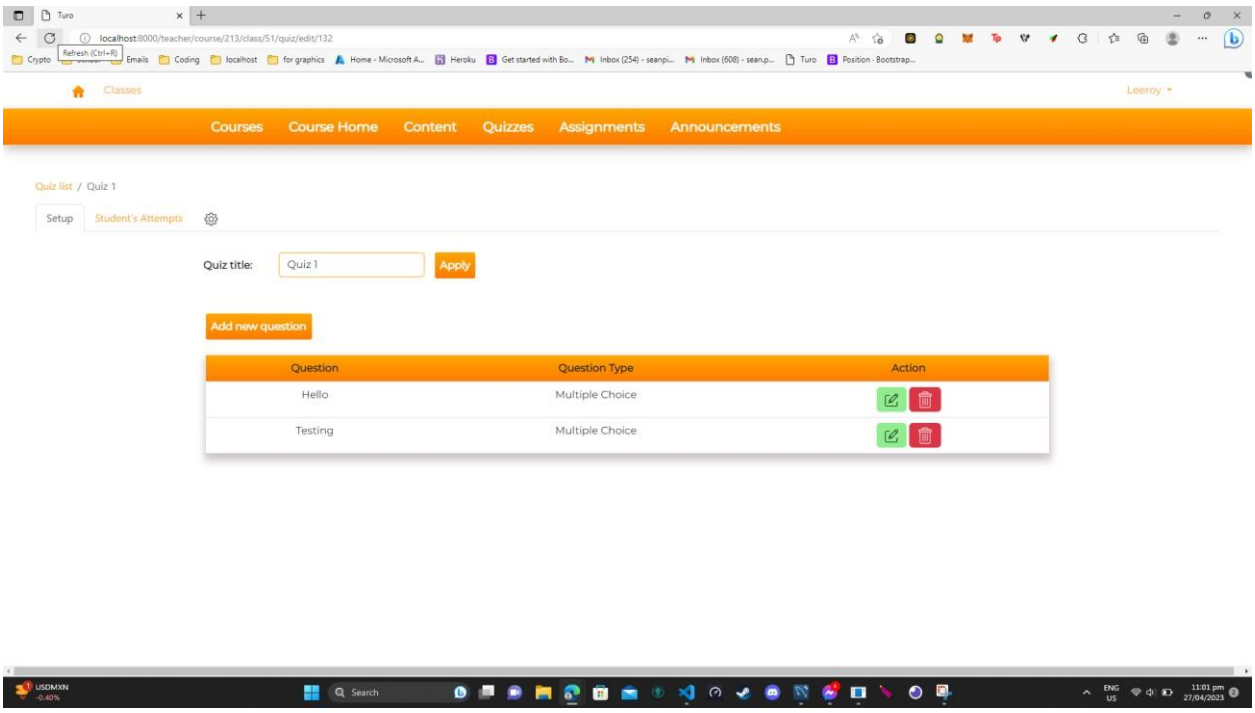

Figure 31: Teacher Quiz Setup

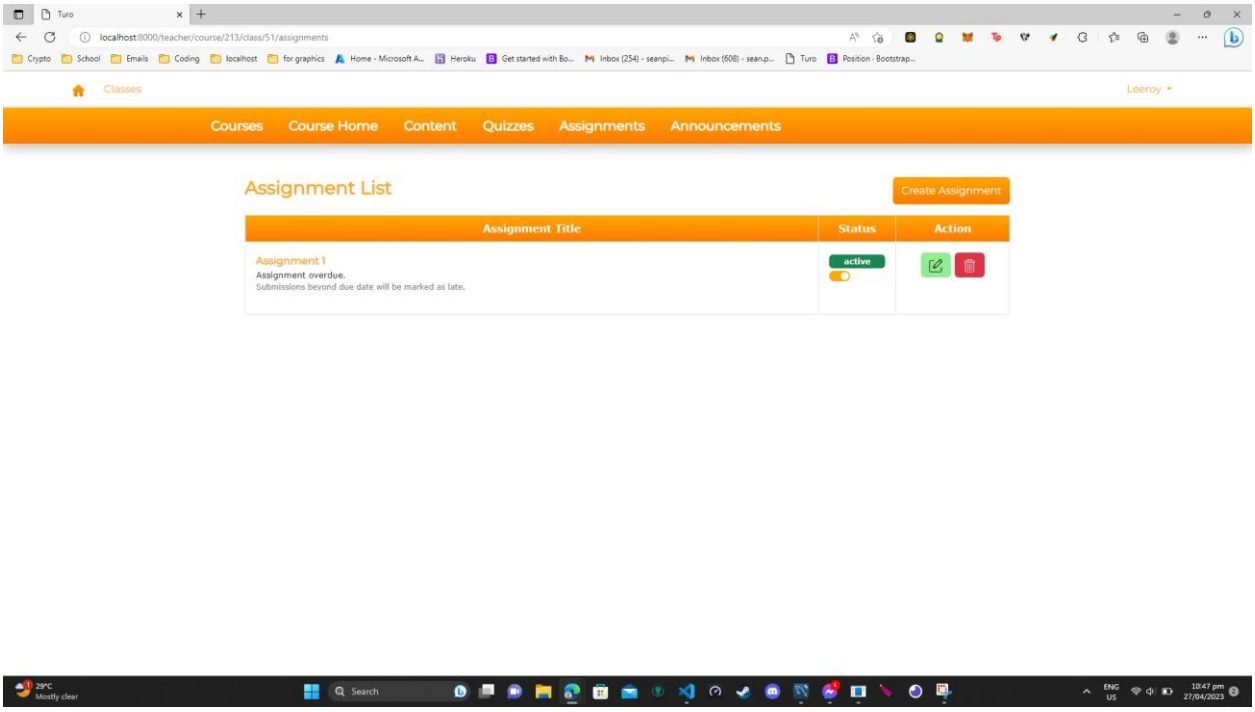

Figure 32: Teacher Assignment List

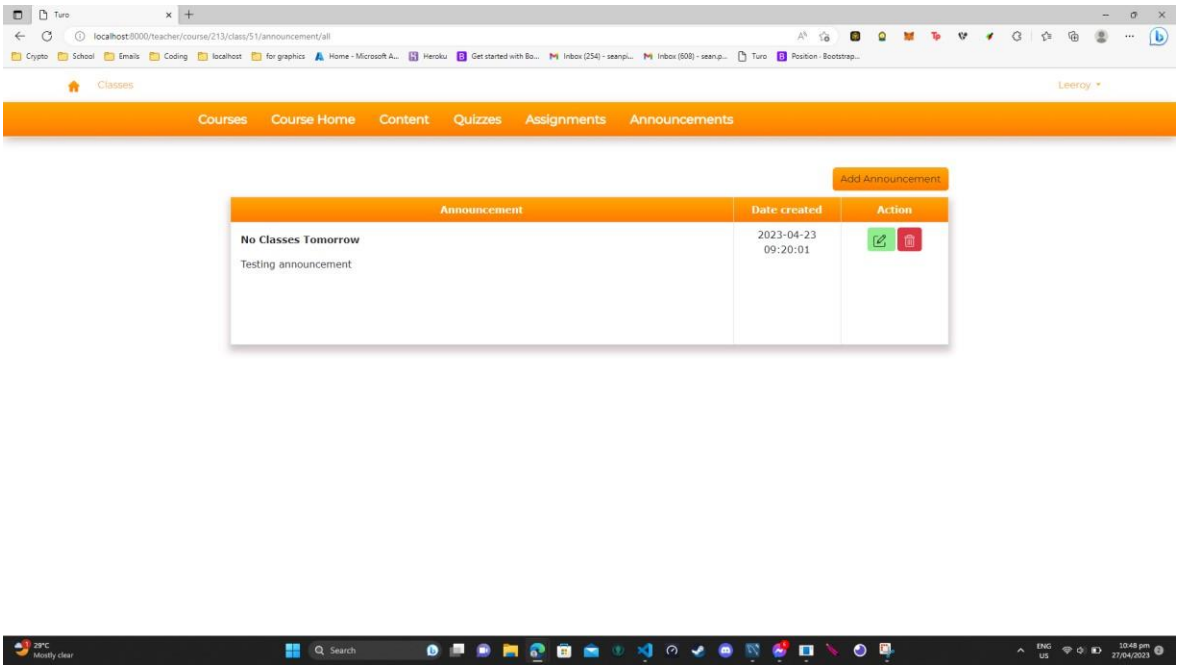

Figure 33: Teacher Announcement List

User Interface for Mobile Application

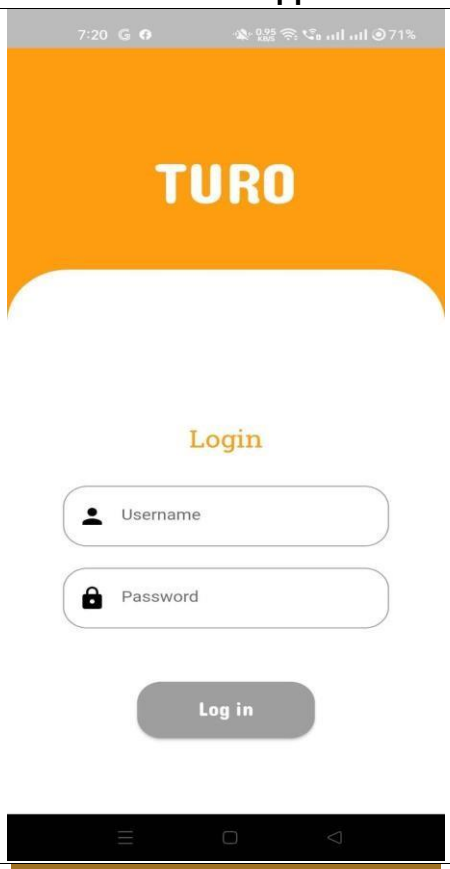

Figure 34: Student Login Screen

The figure on the left is the view where the students login to view the learning materials posted by the teacher.

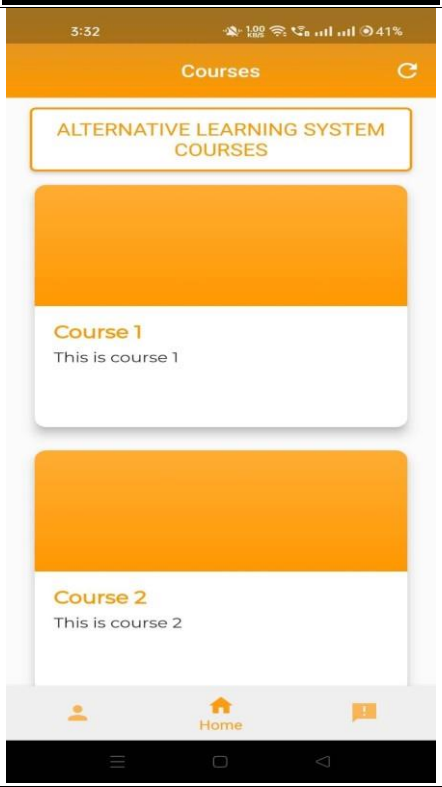

Figure 35: Course Home Screen

The figure on the left shows all the courses of the logged in student.

Figure 36: Profile Screen

The figure on the left shows the student profile. It shows the student's personal information and information of the class which the student belongs to.

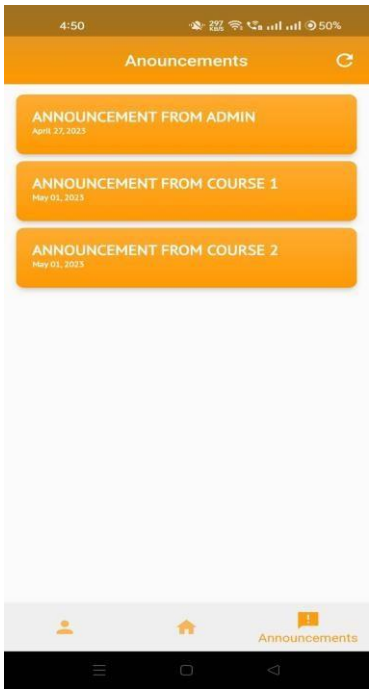

Figure 37: Announcement Screen

The figure on the left shows all the announcements. These announcements are from the student's courses and admin in that specific region.

Figure 38: Course Content Screen

The figure on the left shows the course content. These modules shown are the contents from a clicked course.

Figure 39: Topic and Topic Content Screen

The figure on the left shows the topic Screen. When a student clicks the topic, the topic content will be displayed in an expansion tile containing the different resources (HTML, File, Assignments, and Quiz).

Figure 40: Quizzes Screen

The figure on the left shows the list of quizzes. These quizzes belong to a specific course created by the teacher.

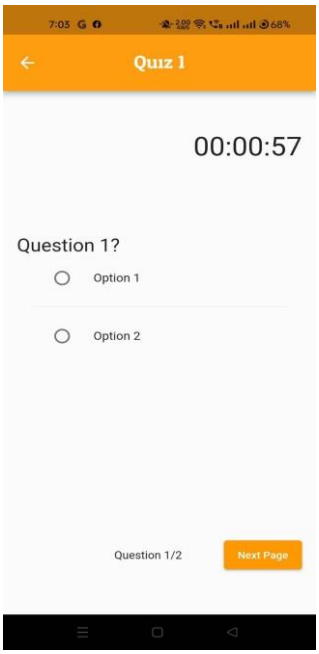

Figure 41: Take Quiz Screen

The figure on the left shows that the student takes and answers the quiz and must submit after the timer ends. The quiz result may be visible after taking the quiz.

Figure 42: Assignments Screen

The figure on the left shows the list of assignments. These assignments belong to a specific course created by the teacher.

Figure 43: View Assignment Screen

The figure on the left shows that the student

can view the assignment details and may answer the assignment directly based on the instruction.

CHAPTER III: SOFTWARE DEVELOPMENT AND TESTING

This chapter describes the implementation of the project in development as well as the various tools used to create and run the application. It contains sections for the Development and Testing Process.

DEVELOPMENT SOFTWARE PLATFORMS, DEVELOPMENT ENVIRONMENT AND PROJECT MANAGEMENT TOOLS

‘TURO’ is a web and mobile based application developed in Windows operating system and Visual studio Code as the tool for both web and mobile software development. Visual Studio Code is a powerful source code editor developed by Microsoft. The programming languages used in software development are PHP with Laravel Framework for the web while Flutter Dart for the mobile. There are also other tools and languages that have been utilized in software development such as Bootstrap, GitHub, GitKraken, JavaScript, and MySQL [9] [10] [11] [12] [13] [14].

The web version of the application intended for the Administrator, Teacher and Student is developed using PHP as the primary language with the Laravel Framework. Laravel is a free open-source PHP web framework created by Taylor Otwell and intended to develop web applications following the model-view-controller (MVC) architectural pattern. The mobile version of the application intended only for the Student uses Flutter framework and Dart as development language [9]. Flutter is Google’s user interface toolkit to develop natively compiled applications for mobile, web, and desktop in a single code-base [9].

APIs were developed and consumed to retrieve data from the data source. An API (Application Program Interface) is a set of routines, protocols, and tools for building software applications.

MySQL was used as the database. It is a full-featured relational database management system (RDBMS) for the application. MySQL is an open source relational database management system (RDBMS) which relies on the Structured Query Language (SQL) for processing the data in the database [11].

Visual Studio Code was used as the text editor of the project for coding. This is a code editor redefined and optimized for building and debugging modern web and cloud applications.

Canva was used as image manipulation tools and also for the creation of assets of the project such as backgrounds and logos for the application [15].

Development Process

In this section, the implementation of the core functionalities of the system was thoroughly discussed.

I. Creation of Learning Materials

A less hassle and paperless distribution of ALS learning materials is made possible through creating courses, modules and topics using the 'TURO' application. This system enables the teacher to create learning materials which include the quiz and assignment insertion that can be accessed by the students to serve as their ALS reviewer.

```php
public function create(Request $request)
{
    $rules = [
        'course_title' => 'required',
        'course_description' => 'required',
    ];

    $messages = [
        'course_title.required' => 'Please input a course title',
        'course_description.required' => 'Please input a course description',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {
        $currentDateTime = Carbon::now();
        $course = new Course();
        $teacherId = Teacher::where('user_id', Auth::id())->get()->first();
        $course->teacher_id = $teacherId->teacher_id;
        $course->course_title = $request->course_title;
        $course->course_description = $request->course_description;
        $course->created_at = $currentDateTime;
        $course->updated_at = $currentDateTime;
        $course->class_id = $request->class_id;
        $course->save();

        return response()->json([
            'status' => true,
            'course' => $course
        ], 200);
    }
}
```

Figure 44: Create Course

The figure above shows the create course code snippet. The ‘create’ function takes a request as a parameter. The request includes basic course information such as the course_title, ‘course_description’ and class_id. The method has validation rules for the required parameter. If the validation fails, the saving of course content data in the database will fail and it will return an error indicating the missing parameter. The course material will then be created allowing teachers to create sub materials.

```php
public function create(Request $request)
{
    $rules = [
        'content_title' => 'required',
        'content_description' => 'required',
    ];

    $messages = [
        'content_title.required' => 'Please input a title',
        'content_description.required' => 'Please input a description',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {

        $courseContent = new CourseContent();

        $courseContent->course_id = $request->input('course_id');
        $courseContent->content_title = $request->input('content_title');
        $courseContent->content_description = $request->input('content_description');
        $courseContent->save();

        return response()->json([
            'status' => true,
            'courseContent' => $courseContent
        ], 200);
    }
}
```

Figure 45: Create Modules

The figure above shows the create module code snippet. The ‘create’ function takes a request as a parameter. The request includes course content / module information such as the ‘content_title’, ‘content_description’ and ‘course_id’. Course Object is required as a prior condition before the teacher can create the course content object. The method has validation rules for the required parameter. If the validation fails, the saving of course content data in the database will fail and it will return an error indicating the missing parameter. After successfully creating the modules, teachers will be able to add multiple topics under its modules.

```php
public function create(Request $request)
{
    $rules = [
        'topic_title' => 'required',
    ];

    $messages = [
        'topic_title.required' => 'Please input a topic title',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {
        $topic = Topic::create([
            "content_id" => $request->content_id,
            "topic_title" => $request->topic_title,
            "topic_description" => $request->topic_description,
        ]);

        return response()->json([
            'status' => true,
            'topic' => $topic
        ], 200);
    }
}
```

Figure 46: Create Topic

The figure above shows the create topic code snippet. The ‘create’ function takes a request as a parameter. The request includes course content / module information such as the ‘topic_title’, ‘topic_description’ and ‘content_id’. Course Content Object is required as a prior condition before the teacher can create the Topic object. The method has validation rules for the required parameter. If the validation fails, the saving of topic data in the database will fail and it will return an error indicating the missing parameter. Otherwise, the Topic will be saved to the database. After creating the topic, the teacher will be able to create different resources such as the file, html document, insert assignments and quizzes.

```php
if ($request->type == 'html') {
    $rules = [
        'topic_content_title' => 'required',
    ];

    $messages = [
        'topic_content_title.required' => 'Please input a title',
    ];
    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {
        $topicContent = TopicContent::insertGetId([
            'topic_id' => $request->topic_id,
            'topic_content_title' => $request->topic_content_title,
            'type' => $request->type,
            'html' => $request->html,
        ]);

        return response()->json([
            'status' => true,
            'topicContent' => $topicContent
        ], 200);
    }
}
```

Figure 47: Create HTML Topic Content

```php
if ($request->type == 'file') {
    $rules = [
        'topic_content_title' => 'required',
    ];

    $messages = [
        'topic_content_title.required' => 'Please input a title',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {

        $file = $request->file('file');
        $originalFileName = $file->getClientOriginalName();
        $extension = $file->getClientOriginalExtension();

        if ($extension == 'pdf') {
            Storage::putFileAs('public/files/topic-' . $request->topic_id . '/', $file, $originalFileName);
            $file = 'storage/files/topic-' . $request->topic_id . '/' . $originalFileName;
            $topicContent = TopicContent::insertGetId([
                'topic_id' => $request['topic_id'],
                'topic_content_title' => $request['topic_content_title'],
                'type' => $request['type'],
                'file' => $file,
            ]);
        } else {
            return redirect()->back()->withErrors(['error' => 'Format "' . $extension . '" is not supported.']);
        }

        return response()->json([
            'status' => true,
            'topicContent' => $topicContent
        ], 200);
    }
}
```

Figure 48: Upload PDF File

```
if ($request->type == 'quiz') {
    $rules = [
        'quiz_link' => 'required',
    ];

    $messages = [
        'quiz_link.required' => 'Please choose a quiz.',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {
        $result = TopicContent::where('topic_id', $request['topic_id'])->where('quiz_link', $request['quiz_link'])->exists();
        if ($result) {
            return response()->json([
                'error' => 'Quiz already exist in this Topic.'
            ], 400);
        } else {
            $topicContent = TopicContent::insertGetId([
                'topic_id' => $request['topic_id'],
                'type' => $request->type,
                'topic_content_title' => $request['topic_content_title'],
                'quiz_link' => $request['quiz_link'],
            ]);
            return response()->json([
                'status' => true,
                'topicContent' => $topicContent
            ], 200);
        }
    }
}
```

Figure 49: Insert Quiz

```
if ($request->type == 'assignment') {

    $rules = [
        'assignment_link' => 'required',
    ];

    $messages = [
        'assignment_link.required' => 'Please choose an assignment.',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return response()->json(['errors' => $validation->errors()], 422);
    } else {
        $result = TopicContent::where('topic_id', $request->topic_id)->where('assignment_link', $request['assignment_link'])->exists();
        if ($result) {
            return response()->json([
                'error' => 'Assignment already exist in this Topic.'
            ], 400);
        } else {
            $topic = Topic::where('topic_id', $request->topic_id)->get()->first();
            $topicContent = TopicContent::insertGetId([
                'topic_id' => $request['topic_id'],
                'type' => $request->type,
                'topic_content_title' => $request['topic_content_title'],
                'assignment_link' => $request['assignment_link'],
            ]);
            return response()->json([
                'status' => true,
                'topicContent' => $topicContent
            ], 200);
        }
    }
}
```

Figure 50: Insert Assignment

The figures above (Figure 47 - Figure 50) shows the create topic content code snippet. The ‘create’ function takes a request as a parameter. The request includes topic content information such as the ‘topic_content_title’, ‘type’ and ‘topic_id’. Topic Object is required as a prior condition before the teacher can create the Topic Content object. Topic Content has different types which are the ‘**HTML**’ (figure 47), ‘**FILE**’ (figure 48), ‘**QUIZ**’ (figure 49), and ‘**ASSIGNMENT**’ (figure 50). Depending on what type is being chosen, it will be included in the parameter and its required resources. The method has validation rules for the required parameter. If the validation fails, the saving of topic data in the database will fail and it will return an error indicating the missing parameter. Otherwise, the topic content will be saved to the database. For the quiz and assignment resources, the teacher will be able to add these to the topic content by creating first the assignment and quizzes. Without creating these assessments, it will display an empty list upon insertion.

II. Student Assessment

There are two forms of student assessment which is the quiz and assignment. This system enables the teachers to create quizzes and assignments. Teachers are able to create quizzes and assignments once courses are already created on their end. These forms of assessment may be visible on the student’s end when the teacher is done setting up, putting the necessary information and enabling them. Teachers may not insert them as content since students can see the assessments in the quiz and assignment tabs.

The figures below (Figure 51 - Figure 52) shows the snippet to create and set up a quiz.

```php
public function create(Request $request)
{
    $rules = [
        'quiz_title' => 'required',
    ];

    $messages = [
        'quiz_title.required' => 'Please input a quiz title',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return redirect()->back()->withInput()->withErrors($validation);
    } else {

        $quiz = new Quiz();

        $quiz->teacher_id = Auth::user()->teacher->teacher_id;
        $quiz->quiz_title = $request->quiz_title;
        $quiz->course_id = $request->course_id;
        $quiz->class_id = $request->class_id;
        $quiz->save();

        return back();
    }
}
```

Figure 51: Create Quiz

```php
public function quizSetup($courseid, $quizid, Request $request)
{
    $setupQuiz = Quiz::where('quiz_id', $quizid)->get()->first();

    $hours = $request->duration[0] * 60; //hours in minutes
    $duration = $hours + $request->duration[1]; //total duration in minutes

    $setupQuiz->update([
        'start_date' => $request->start_date,
        'end_date' => $request->end_date,
        'start_time' => $request->start_time,
        'end_time' => $request->end_time,
        'attempts' => $request->attempts,
        'password' => $request->password,
        'releaseGrades' => $request->releaseGrades,
        'duration' => $duration,
    ]);

    return response()->json([
        'status' => true,
        'quiz' =>$setupQuiz
    ], 200);
}
```

Figure 52: Setup Quiz

The figures below (Figure 53 - Figure 54) shows the snippet to create and set up an assignment.

```php
public function create(Request $request)
{
    $rules = [
        'assignment_title' => 'required',
    ];

    $messages = [
        'assignment_title.required' => 'Please input an assignment title',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if ($validation->fails()) {
        return redirect()->back()->withInput()->withErrors($validation);
    } else {
        $assignmentId = Assignment::insertGetId([
            'course_id' => $request->course_id,
            'assignment_title' => $request->assignment_title,
            'assignment_description' => $request->assignment_description,
            'start_date' => $request->start_date,
            'end_date' => $request->end_date,
            'start_time' => $request->start_time,
            'end_time' => $request->end_time,
            'submission_type' => $request->submission_type,
            'class_id' => $request->class_id
        ]);

        return redirect()->to(route('assignment.view', [$request->course_id, $request->class_id, $assignmentId]));
    }
}
```

Figure 53: Create Assignment

```php
public function update($assignmentid, Request $request)
{
    $chosenAssignment = Assignment::where('assignment_id', $assignmentid)->get()->first();

    $chosenAssignment->update([
        'assignment_title' => $request->assignment_title,
        'assignment_description' => $request->assignment_description,
        'start_date' => $request->start_date,
        'end_date' => $request->end_date,
        'start_time' => $request->start_time,
        'end_time' => $request->end_time,
        'points' => (int)$request->points,
        'submission_type' => $request->submission_type,
        'multiple_submissions' => $request->multiple_submissions,
        'submitAfterDeadline' => $request->submitAfterDeadline,
        'releaseGrades' => $request->releaseGrades

    ]);

    return $this->successResponse("Assignment updated successfully", $chosenAssignment);
}
```

Figure 54: Setup Assignment

III. Student Enrollment

The student must enroll in a program to access the learning materials. Enrollment of the student will be based on the availability of the program they want to enroll in.

```
if ($validation->fails()) {
    return response()->json(['errors' => $validation->errors()], 422);
} else {

    $student->update([
        'prog_id' => $request->prog_id
    ]);
    Enrollment::insertGetId([
        'student_id' => $studentId,
        'prog_id' => $request->prog_id,
        'loc_id' => $locId,
        'enrollment_date' => Carbon::now(),
    ]);

    StudentInformation::create([
        'student_id' => $studentId,
        'LRN' => $request->LRN,
        'student_gender' => $request->student_gender,
        'student_civil' => $request->student_civil,
        'student_birth' => $request->student_birth,
        'student_placeofbirth' => $request->student_placeofbirth,
        'street' => $request->street,
        'barangay' => $request->barangay,
        'city' => $request->city,
        'province' => $request->province,
    ]);

    StudentFamily::create([
        'student_id' => $studentId,
        'student_compfname' => $request->student_compfname,
        'student_compmname' => $request->student_compmname,
        'student_complname' => $request->student_complname,
        'student_motherfname' => $request->student_motherfname,
        'student_mothermname' => $request->student_mothermname,
        'student_motherlname' => $request->student_motherlname,
    ]);

    StudentEducation::create([
        'student_id' => $studentId,
        'last_level' => $request->last_level,
        'student_reason' => $request->student_reason,
        'answer_type' => $request->answer_type,
        'program_attended' => $request->program_attended,
        'program_literacy' => $request->program_literacy,
        'program_attended_year' => $request->program_attended_year,
    ]);
```

Figure 55: Student Enrollment

```
$allowedfileExtension = ['jpg', 'png'];
$psa = $request->file('psa');
$picture = $request->file('picture');
$form137 = $request->file('form137');
$filesUploaded = [
    $psa, $picture, $form137
];
$filesKey = [
    'PSAPath', 'IDPath', 'Form137Path'
];

$studentFiles = StudentFiles::where('student_id', $studentId);
StudentFiles::create([
    'student_id' => $studentId,
]);
for ($i = 0; $i < $request->files->count(); $i++) {
    $originalFileName = $filesUploaded[$i]->getClientOriginalName();
    $extension = $filesUploaded[$i]->getClientOriginalExtension();
    $check = in_array($extension, $allowedfileExtension);
    if ($check) {
        Storage::putFileAs('public/files/enrollment/files/' . 'student_' . $studentId . '/', $filesUploaded[$i], $originalFileName);
        $path = 'storage/files/enrollment/files/' . 'student_' . $studentId . '/' . $originalFileName;

        $studentFiles->update([
            $filesKey[$i] => $path,
        ]);
    } else {
        return redirect()->back()->withErrors(['error' => 'Format "' . $extension . '" is not supported.']);
    }
}
return redirect()->to('/student/home');
```

Figure 56: Student Enrollment

The figures above (Figure 55 - Figure 56) shows the Student Enrollment code snippet.

The ‘enroll’ function takes a request as a parameter. The request parameter includes all the necessary information of the students shown in Figure 56 and the file content of the required documents being uploaded. If student enrollment is done, the teacher will need to validate the information of the student before approving its enrollment. After approving the enrollment, the teacher must assign the student to a class where they belong based on the student details provided.

IV. Viewing of Learning Materials (Student: Web and Mobile)

This system allows users to view the learning materials posted by the teacher. Students can view the learning materials in both web and mobile.

- **Web Application**

In the Web View, the researchers use the built-in functions in the Laravel controller that returns a view which is a blade component and at the same time with the use of *with* method, it also returns an instance of the defined object.

The figures below (Figure 57 - Figure 58) shows the snippet to retrieve the topic and the topic content.

```
public function studentDisplayContent($courseid)
{
    $courseCollection = Course::where('course_id', $courseid)->get();
    $access = new AccessContent();
    $accessCollection = $access->getAllAccessByStudent();


    return view('student.content.display')->with(compact('courseCollection', 'accessCollection'));
}
```

Figure 57: Retrieve Topic

```
@foreach ($courseCollection as $course)
    @foreach ($course->coursecontent as $module)
        <button type="button" class="collapsible" content-title="{{ $module->content_title }}"
            value="{{ $module->content_id }}">
            <b style="text-transform: capitalize">{{ $module->content_title }}</b>
        </button>

        @php
            $sessionCourseContent = Session('courseContent');
            $sessionTopicContentList = Session('topicContentList');
        @endphp
        <div class="content" @if ($sessionCourseContent == $module->content_id) style ="display: block;" @endif>
            @foreach ($module->topic as $topic)
                <div class="topic" topic-id="{{ $topic->topic_id }}"
                    topic-title="{{ $topic->topic_title }}" value="{{ $topic->topic_id }}"
                    onclick="getTopicId({{ $topic->topic_id }})">
                    <label for="" style="text-transform: capitalize">
                        {{ $topic->topic_title }}
                    </label>
                </div>
                <div class="topic-content-list" {{ $sessionTopicContentList }}
                    @if ($sessionTopicContentList == $topic->topic_id) style ="display: block;" @endif>
                    @foreach ($topic->topiccontent as $topiccontent)
                        <div class="topic-content" id="topic-content"
                            topic-content-id="{{ $topiccontent->topic_content_id }}">
                            @if ($topiccontent->type == 'html')
                                <img src="{{ asset('images/text.jpg') }}" alt="">
                            @elseif($topiccontent->type == 'quiz')
                                <img src="{{ asset('images/link.png') }}" alt="">
                            @elseif($topiccontent->type == 'assignment')
                                <img src="{{ asset('images/link.png') }}" alt="">
                            @else
                                <img src="{{ asset('images/pdf.png') }}" alt="">
                            @endif
                            {{ $topiccontent->topic_content_title }}
                            @foreach ($accessCollection as $key => $access)
                                @if ($access->topic_content_id == $topiccontent->topic_content_id)
                                    <div class="float-end"><img src="{{ asset('images/correct.png') }}"
                                            alt="" style="width: 15px; height:15px;"></div><br>
                                @endif
                            @endforeach
                        </div>
                    @endforeach
                </div>
            @endforeach
        </div>
    @endforeach
@endforeach
```

Figure 58: Retrieve Topic Content

The figure above shows how learning contents are displayed on the student's screen. Figure 58 shows how the topic and topic content are displayed by accessing the list of objects returned in Figure 57. Topic content consists of assignments, quizzes, PDF files and html documents. These topic content are displayed using a HTTP client called axios. The Get method request is used in the fetching of data to load the topic content asynchronously whereas when a

student clicks the topic content, it will be loaded and displayed in the main screen showing the necessary learning content details.

- **Mobile Application**

In the Mobile View, the course, course content, topics and topic content has its corresponding request. The getCourses request is called using the FutureBuilder while other requests are processed in a GetX Controller.

```
Future<List<Course>> getCourses(classID) async {
  final courses = <Course>[];
  final response =
      await http.get(Uri.parse('$baseUrl/class/$classID/course'));
  if (response.statusCode == 200) {
    final jsonData = convert.jsonDecode(response.body);

    for (var item in jsonData['courses']) {
      courses.add(Course.fromJson(item));
    }
    return courses;
  }

  return courses;
}
```

Figure 59: Retrieve Courses

The code snippets above shows how FutureBuilder is utilized in retrieving the Courses of the student. In the figure 59, the researchers implemented a function for retrieving the courses that sends a request to the Laravel backend using the HTTP package. After the request has been sent, the researchers initialized a variable for the storing of the list of courses. The response is then converted into a Course object and stores each of the objects on the initialized variable.

The researchers used the FutureBuilder to call the getCourses function asynchronously. When the loading of data is done, it will return a list of courses. Otherwise, it will return a loading icon which indicates that the fetching of data is in process or if the list is empty, a custom message will be displayed.

```
Future<void> getCourseContent(courseID) async {
  contentData.clear();

  final response = await service.getCourseContent(courseID);
  for (var element in response) {
    contentData.add(element);
  }
  isLoading.value = false;
}
```

Figure 60: Retrieve Modules

```
Future<void> getQuizCollection(courseID) async {
  quizData.clear();
  final response = await service.getQuiz(courseID);
  for (var element in response) {
    quizData.add(element);
  }
  isLoading.value = false;
}
```

Figure 61: Retrieve Quiz

```
Future<void> getAssignments(courseID) async {
  assignmentData.clear();
  final response = await service.getAssignments(courseID);
  for (var element in response) {
    assignmentData.add(element);
  }
  isLoading.value = false;
}
```

Figure 62: Retrieve Assignment

The code snippet above shows how GetX is used as a Reactive State Manager. The researchers created a void function that will add all objects in the RxList variable. After sending a request to the backend, the contentData which is the RxList variable is used as storage for the list

of course content. Same will apply in with figure 65 and figure 66 which uses the quizData and assignmentData for storage to their corresponding Object.

```
public function create(Request $request){

    $rules = [
        'announcement_title' => 'required',
    ];

    $messages = [
        'announcement_title.required' => 'Please input an announcement title',
    ];

    $validation = Validator::make($request->input(), $rules, $messages);

    if($validation->fails()){
        return redirect()->back()->withInput()->withErrors($validation);
    }
    else{
        $currentDateTime = Carbon::now();

        $announcement = new Announcement();

        $announcement->created_at = $currentDateTime;
        $announcement->announcement_title = $request->announcement_title;
        $announcement->announcement_description = $request->announcement_description;
        $announcement->creator = $request->creator;
        $announcement->creator_id = $request->creator_id;
        $announcement->course_id = $request->course_id;
        $announcement->class_id = $request->class_id;
        $announcement->save();

        return back();
    }
}
```

Figure 63: Create Announcement

The figure above shows the create announcement code snippet. The ‘create’ function takes a request as a parameter. The request includes topic content information such as the ‘announcement_title’, ‘announcement_title’, ‘creator’, ‘creator_id’ and ‘course_id’. The creator refers to who is the creator of the announcement either if it's a teacher or an admin. The course_id is nullable since it could be created by the admin in which admin does not own any courses. The method has validation rules for the required parameter. If the validation fails, the saving of announcement data in the database will fail and it will return an error indicating the missing parameter. All Announcements created are then shown in the student’s announcement screen.

Testing Process

The developers testing segment depicts that all the features of the application are working correctly and how efficient the process is.

WEB DEVELOPMENT

Test Case Scenario ID: 01	
Test Case Title	**TCS01 - Manage Users**
Test Scenario Description	Add new Regional Admin
Test Data	Region Assignment: Region VII Email: regionaladmin1@gmail.com Username: regionaladmin1 First name: regional Middle Initial: (none) Last name: admin Password: regionaladmin1 Phone number: 09123456789 Date of birth: 2000-01-10
Test Steps	1. Input Regional Admin details 2. Click “Add” button
Expected Result	Regional Admin Account successfully created.
Actual Result	Regional Admin Account successfully created.
Status	Passed

Table 18: TCS01 - Manage Users

Test Case Scenario ID: 02	
Test Case Title	**TCS02 - Manage Users**

Test Scenario Description	Input existing username/email
Test Data	Email: regionaladmin1@gmail.com Username: regionaladmin1
Test Steps	1. Input Regional Admin details 2. Click “Add” button
Expected Result	An error message "username already exists" will be displayed. An error message “email already exists” will be displayed.
Actual Result	An error message "username already exists" & “email already exists” will be displayed.
Status	Passed

Table 19: TCS02 - Manage Users

Test Case Scenario ID: 03	
Test Case Title	**TCS03 - Manage Admin Announcement**
Test Scenario Description	Create Announcement
Test Data	Title: Announcement 1 Description: ALS program batch 1 will start on June 2023
Test Steps	1. Input Announcement Title 2. Input Announcement Description 3. Click the “Create” button
Expected Result	Admin Announcement successfully created.
Actual Result	Admin Announcement successfully created.
Status	Passed

Table 20: TCS03 - Manage Admin Announcement

Test Case Scenario ID: 04	
Test Case Title	**TCS04 - Manage Program**
Test Scenario Description	Admins add ALS Program in TURO
Test Data	Program Name: Basic Literacy Program Program Code Name: BLP
Test Steps	1. Click the “Program” tab
	2. Click the “Add Program” button 3. Click the “Add” button
Expected Result	ALS Program successfully added.
Actual Result	ALS Program successfully added.
Status	Passed

Table 21: TCS04 - Manage Program

Test Case Scenario ID: 05	
Test Case Title	**TCS05 - Manage Learning Center**
Test Scenario Description	Admins add ALS Learning Center in TURO
Test Data	Learning Center City: Cebu City Learning Center Province: Cebu Learning Center Name: Banilad Elementary School
Test Steps	1. Click the “Learning Center” tab 2. Click the “Add Learning Center” button 3. Click the “Add” button
Expected Result	ALS Learning Center successfully added.
Actual Result	ALS Learning Center successfully added.
Status	Passed

Table 22: TCS05 - Manage Learning Center

Test Case Scenario ID: 06	
Test Case Title	**TCS06 - Student Enrollment**
Test Scenario Description	Input student personal information
Test Data	Student details
Test Steps	1. Click the “Enrollment” Tab 2. Click the “Step 1: Submit an Application”
	3. Click the “Enroll Online” button 4. Click the “Submit” button
Expected Result	Student Enrollment is successfully submitted to ALS Teacher
Actual Result	Student Enrollment is successfully submitted to ALS Teacher
Status	Passed

Table 23: TCS06 - Student Enrollment

Test Case Scenario ID: 07	
Test Case Title	Student Enrollment
Test Scenario Description	Input an Invalid Format information
Test Data	Mp3 file format
Test Steps	1. Click the “Enrollment” Tab 2. Click the “Step 1: Submit an Application” 3. Click the “Enroll Online” button 4. Click the “Submit” button
Expected Result	“please select a JPEG, JPG or PG File” will be displayed

Actual Result	“please select a JPEG, JPG or PG File” will be displayed
Status	Passed

Table 24: TCS07 - Student Enrollment

Test Case Scenario ID: 08	
Test Case Title	**TCS08 - Manage Class**
Test Scenario Description	Teacher creates a class in ALS
Test Data	Class Title: Sunflower Class Class Description: This class is the sunflower class Start Duration: May 2023 End Duration: July 2023
Test Steps	1. Click the “Create Class” button 2. Teacher inputs Class Details 3. Click the “Create” button
Expected Result	Class successfully created
Actual Result	Class successfully created
Status	Passed

Table 25: TCS08 - Manage Class

Test Case Scenario ID: 09	
Test Case Title	**TCS09 - Manage Course**
Test Scenario Description	Teacher creates a course

Test Data	Course Title: Reading Course Description: This is reading
Test Steps	1. Click the “Add Course” or “+” button 2. input Course Title 3. Input Course Description 4. Click the “Create” button
Expected Result	Course successfully created
Actual Result	Course successfully created
Status	Passed

Table 26: TCS09 - Manage Course

Test Case Scenario ID: 10	
Test Case Title	**TCS10 - Manage Course Announcement**
Test Scenario Description	Create Announcement
Test Data	Title: Announcement 1 Description: We will have an exam next week
Test Steps	1. Click a Course
	2. Click “Announcement” Tab 3. Click “Add Announcement” button 4. Input Announcement Title 5. Input Announcement Description 6. Click the “Create” button
Expected Result	Course Announcement successfully created.
Actual Result	Course Announcement successfully created.
Status	Passed

Table 27: TCS10 - Manage Course Announcement

Test Case Scenario ID: 11

Test Case Title	**TCS11 - Manage Module**
Test Scenario Description	Teacher creates module
Test Data	Title: Nouns Description: This topic is all about nouns
Test Steps	1. Click a Course 2. Click the “Add Module” button 3. Click the “Create” button
Expected Result	Module successfully created.
Actual Result	Module successfully created.
Status	Passed

Table 28: TCS11 - Manage Module

Test Case Scenario ID: 12	
Test Case Title	**TCS12 - Manage Topic**
Test Scenario Description	Teacher creates topic in a module
Test Data	Title: Fast Reading Description: Let’s talk about Fast Reading
Test Steps	1. Click a Module
	2. Click “Add a Topic” button 3. Input Module details 4. Click the “Create” button
Expected Result	Module successfully created.
Actual Result	Module successfully created.

Status	Passed

Table 29: TCS15 - Manage Topic

Test Case Scenario ID: 13	
Test Case Title	**TCS13 - Manage Quiz**
Test Scenario Description	Create Quiz
Test Data	Quiz Title: Quiz 1-Reading
Test Steps	1. Click “Quizzes” tab in the navigation bar 2. Click “Create Quiz” 3. Input Quiz Title 4. Click the “Create” button
Expected Result	Quiz successfully created.
Actual Result	Quiz successfully created.
Status	Passed

Table 30: TCS18 - Manage Quiz

Test Case Scenario ID: 14	
Test Case Title	**TCS14 - Take Quiz**
Test Scenario Description	Student inputs incorrect quiz password Student answers all questions correctly
Test Data	Quiz Password Quiz Answers
Test Steps	In the Quiz List Page, 1. Click Quiz Title 2. Input incorrect Quiz Password
Expected Result	An error message “password is incorrect” will be displayed

Actual Result	An error message “password is incorrect” will be displayed
Status	Passed

Table 31: TCS14 - Take Quiz

Test Case Scenario ID: 15	
Test Case Title	**TC15 - Manage Assignment**
Test Scenario Description	Create Assignment
Test Data	Assignment Title: Assignment 1
Test Steps	In the Course Page 1. Click the “Assignments” tab 2. Click the “Create Assignment” button 3. Input Assignment Title 4. Click the “Create” button
Expected Result	Assignment successfully created.
Actual Result	Assignment successfully created.
Status	Passed

Table 32: TCS15 - Manage Assignment

Test Case Scenario ID: 16	
Test Case Title	**TCS16 - Submit Assignment**
Test Scenario Description	Student submits an Assignment
Test Data	Student Assignment submission
Test Steps	In the Course Page 1. Click the “Assignments” tab 2. Click the Assignment Title 3. Submit an assignment
Expected Result	Assignment successfully submitted.
Actual Result	Assignment successfully submitted.

Status	Passed

Table 33: TCS16 - Submit Assignment

CHAPTER IV: SUMMARY, CONCLUSION AND RECOMMENDATIONS

This chapter talks about the challenges undergone by the developer. The things that they have undergone and learned during the development stages.

Summary of Findings

The researchers intended to create an application that has a goal to help ALS Program be more convenient in their ways of distributing reviewers. The project would involve creating the learning materials, retrieving the data of the learning materials, answering quizzes and submitting assignments as well.

Despite the success in making this research project, the developers also encountered difficulties in implementing some of the functionalities because of various conditions. Developing the web app was a challenging task for the researchers as they had to implement an API for the mobile app and also provide the basin functions intended for the web app. It was also an obstacle that there was less documentation on how these kinds of systems are implemented properly.

In the development process, there were also difficulties that the researchers have encountered. This includes server connectivity to the host, and how the mobile application retrieves data using the API from the web server. There were times that delays were frequently occurring upon retrieving data to the mobile application due to error mishandling and since the researchers are using an emulator, it takes time to reload and wait for the application to respond.

Conclusion

The project objectives of this study are successfully achieved. The system objectives are to implement an Enrollment system for ALS students, develop functionalities on managing the Learning Materials, and integrate a Quiz and Assignment System where the students can assess their learning through the quiz and assignments posted by the teachers. This study assures that the application, TURO, developed by the researchers will make a great impact to the ALS Program. The application will also help the actors to have a convenient way of learning since the system provides an easy viewing and a seamless access of learning materials for both teachers and students. Based on the process of the development, the developers were able to implement and deliver the features of the study despite the constraints of learning the framework and language to be used in the implementation.

Recommendation

Further research for this study would ensure the availability and access to the application so that response time and server access would be fast and always online. Based on this, the researchers have looked upon the uses of cloud storage for faster data access that would allow for access to different learning materials of the application, both mobile and web application.

Another feature recommended is that it would be better to include audio and video clips in the learning materials as this would enhance the learning capability and interest of ALS students. Having these as a future enhancement for the application, this will enable students to repeatedly access and practice to master their skills and life techniques.

Future work for this study is that recommendations be reviewed and validated and that the researchers will procure a detailed development plan to produce a reliable application and be easily accessed for future revisions.

BIBLIOGRAPHY

[1] "Department of Education Davao Region," [Online]. Available: https://depedroxi.ph/ufaqs/who-are-the-target-learners-in-the-als/. [Accessed june 2022].

[2] "Connectivism Learning Theory," 17 May 2021. [Online]. Available: https://www.wgu.edu/blog/connectivism-learning-theory2105.html#close. [Accessed 22 july 2022].

[3] Z. Underwood, "Connectivism: A Learning Theory for Today's Academic Advising," onacada, 22 august 2016. [Online]. Available: https://nacada.ksu.edu/Resources/Academic-Advising-Today/View-Articles/ConnectivismA-Learning-Theory-for-Todays-Academic-Advising.aspx. [Accessed 23 july 2022].

[4] "What Is Connectivism Learning Theory and How Can You Apply It in Learning and Development?," 360 Learning, [Online]. Available: https://360learning.com/guide/learningtheories/connectivism-learning-theory/. [Accessed 23 july 2022].

[5] S. M. Mayer, "learning theories," [Online]. Available: https://learning-theories.com/elearning-theory-mayer-sweller-moreno.html. [Accessed 23 july 2022].

[6] H. Gardner, "The Learning Classroom: Theory Into Practice," Detroit Public Television and Mort Crim Communications, 2003. [Online]. Available: https://www.learner.org/series/thelearning-classroom-theory-into-practice/different-kinds-of-smart-multiple-intelligences/.
[Accessed 23 july 2022].

[7] "Department of Education," [Online]. Available: https://www.deped.gov.ph/als-programs/. [Accessed 24 july 2022].

[8] M. A. Llego, "TEACHERPH," [Online]. Available: https://www.teacherph.com/depedlearner-reference-number-lrn/. [Accessed 27 july 2022].

[9] "Flutter Dev," [Online]. Available: https://flutter.dev. [Accessed june 2022].

[10] "Laravel," [Online]. Available: https://laravel.com. [Accessed june 2022].

[11] MySQL. [Online]. Available: https://www.mysql.com. [Accessed June 2022].

[12] "Bootstrap," [Online]. Available: https://getbootstrap.com. [Accessed July 2022].

[13] "GitHub," [Online]. Available: https://github.com. [Accessed june 2022].

[14] "GitKraken," [Online]. Available: https://www.gitkraken.com.

[15] "Canva," [Online]. Available: https://www.canva.com. [Accessed july 2022].

GRAMMARLY AND PLAGIARISM TEST CERTIFICATE

CURRICULUM VITAE