

VIETNAM NATIONAL UNIVERSITY OF HO CHI MINH CITY  
THE INTERNATIONAL UNIVERSITY  
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING



# **A BLENDED LEARNING MANAGEMENT SYSTEM WEB APP**

By

Nguyen Do Cuong

A thesis submitted to the school of Computer Science and Engineering in  
partial fulfillment of requirements for the degree of Bachelor of Computer  
Science

Ho Chi Minh City, Viet Nam

Year 2024

APPROVED BY:

\_\_\_\_\_ ,

Ly Tu Nga, Dr

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

THESIS COMMITTEE

(Whichever applies)



## ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my supervisor, Dr. Ly Tu Nga for her unwavering support and guidance throughout my thesis journey. Her expertise, patience, and encouragement have been instrumental in shaping the direction of this project. I am truly grateful for her valuable feedback and insightful suggestions that have greatly improved the quality of my work.

Secondly, I would like to extend my heartfelt gratitude to Assoc. Prof. Nguyen Van Sinh for unwavering commitment, insightful feedback, constructive criticism, and willingness to share his expertise have greatly enriched my work and helped me navigate through challenges with confidence.

Thirdly, I would like to take this opportunity to express my gratitude to Nguyen Duc Dang Khoi K18 for helping me come up with thesis ideas, support, and generously volunteered his time and shared his experiences for this project from the beginning.

Lastly, I am grateful to the faculty of the School of Computer Science and Engineering of the International University. Their unwavering commitment to education and their passion for nurturing students' growth have had a profound impact on my academic journey. I would also like to acknowledge the guidance and encouragement provided by Assoc. Prof. Vo Thi Luu Phuong and Dr. Ly Tu Nga throughout my time at International University. Their insights, constructive feedback, and dedication to fostering a positive learning environment have contributed significantly to my personal and intellectual development.

## TABLE OF CONTENTS

ABSTRACT .....	1
Chapter 1.....	2
INTRODUCTION .....	2
1.1. Background .....	2
1.2. Problem statement .....	3
<b>1.3. Objective</b> .....	4
1.4. Assumption and solution .....	5
Chapter 2.....	8
LITERATURE REVIEW .....	8
2.1. Asynchronous learning.....	8
2.2. Synchronous learning.....	9
2.2.1. Traditional synchronous learning.....	9
2.2.2. Synchronous online learning.....	9
2.3. Blended (Combination) .....	10
2.4. Similar system .....	12
2.5. Microservice Architecture.....	13
2.5.1. Introduction .....	13
2.5.2. Domain-Driven Design .....	15
<b>2.6. Techniques</b> .....	16
<b>2.6.1. Technology</b> .....	16
<b>2.6.2. Chunking</b> .....	17
<b>2.6.3. Multimedia resources</b> .....	18
Chapter 3.....	19
METHODOLOGY .....	19
3.1. Overview .....	19
<b>3.2. Diagrams</b> .....	20
3.2.1. Use case.....	20
3.2.2. Activity diagram.....	29
3.2.3. ERD Diagram.....	30
3.2.4. Class Diagram .....	33
3.3. Sequence diagram.....	34
3.3.1. Authorization and Authentication .....	34
<b>3.3.2. Request course materials</b> .....	36
<b>3.3.3. Submission</b> .....	36
Chapter 4.....	37

IMPLEMENT AND RESULT .....	37
<b>4.1. Frontend.....</b>	<b>37</b>
<b>4.1.1. User Interface .....</b>	<b>37</b>
4.1.1.1. Login page.....	37
4.1.1.2. Homepage .....	38
4.1.1.3. Navigation bars .....	39
4.1.1.4. Announcement page.....	39
4.1.1.5. Grading page .....	40
4.1.1.6. Assessment and submission page.....	41
4.1.1.7. Content page.....	42
<b>4.1.4. Chat implementation .....</b>	<b>45</b>
<b>4.2. Backend.....</b>	<b>46</b>
4.2.1. Overview .....	46
4.2.2. Services .....	46
4.2.2.1. User service .....	47
4.2.2.2. User management.....	47
4.2.2.3. Authentication and authorization .....	48
4.2.2.4. Announcement service .....	51
4.2.2.5. Assessment service.....	52
4.2.2.6. Course service .....	53
<b>4.2.2.7. Communication service .....</b>	<b>54</b>
<b>4.3. Database .....</b>	<b>55</b>
4.3.1. User service database .....	56
4.3.2. Course service database .....	56
4.3.3. Assessment service database.....	57
4.3.4. Announcement service database .....	58
4.3.5. Communication service database.....	59
<b>4.4. API Gateway .....</b>	<b>60</b>
<b>4.4.1. Implementation .....</b>	<b>60</b>
<b>4.4.2. Result.....</b>	<b>62</b>
Chapter 5.....	65
DISCUSSION AND EVALUATION .....	65
<b>5.1. Discussion.....</b>	<b>65</b>
<b>5.2. Comparison.....</b>	<b>65</b>
<b>5.3. Evaluation .....</b>	<b>66</b>
Chapter 6.....	67
CONCLUSION .....	67

<b>6.1</b>	<b>Conclusion.....</b>	<b>67</b>
<b>6.2</b>	<b>Future work .....</b>	<b>67</b>
	<b>REFERENCES .....</b>	<b>69</b>

## LIST OF FIGURES

Figure 1.2-1: Strengths and Challenges of Asynchronous Online Learning (Cahyani and Ni Made Wahyu Suganti, 2021).....	3
Figure 2-2.3-1: The blended learning model (Gururaj Bhadri, 2022).....	11
Figure 2-2.3-2: Synchronous, Asynchronous, and Blended Online Learning (Cahyani and Ni Made Wahyu Suganti, 2020).....	11
Figure 2.4-1: Emodo platform interface (Edmodo, 2023) .....	13
Figure 2.5.2-1:Tree diagram to organize chunks (Gururaj Bhadri, and L. P , 2022) .....	18
Figure 3.1-1: System Diagram .....	19
Figure 3.1-2: View course content use case.....	20
Figure 3.1-3: View Announcement use case.....	22
Figure 3.1-4:Assessment usecase.....	24
Figure 3.1-5:Communication usecase.....	27
Figure 3.1-6:Grading activity.....	29
Figure 3.1-7:Announcement service activity .....	29
Figure 3.1-8:Course service activity .....	30
Figure 3.1-9: ERD diagram.....	31
Figure 3.1-10: Class diagram .....	33
Figure 3.3-1: Authorization and authentication sequence diagram.....	34
Figure 3.3-2:Request material sequence diagram .....	36
Figure 3.3-3:Submission sequence diagram .....	36
Figure 4-1Next Js folder structure.....	37
Figure 4-2: Login UI .....	38
Figure 4-3:Login error UI .....	38
Figure 4-4:Home page UI .....	39
Figure 4-5: Announcement UI .....	40
Figure 4-6: Announcement notify.....	40
Figure 4-7: Grading center UI.....	41
Figure 4-8:Grading status.....	41
Figure 4-9: Student submission.....	41
Figure 4-10:Submission review .....	42
Figure 4-11:Presession content UI .....	42
Figure 4-12:Content UI detail .....	43
Figure 4-13:API fetching custom hook.....	43
Figure 4-14: Middleware implementation .....	44
Figure 4-15:Websocket server implementation .....	45
Figure 4-16:Serverconnection implementation.....	46
Figure 4.2.2-1:User Controller .....	47
Figure 4.2.2-2:User service structure .....	48
Figure 4.2.2-3:Authentication process .....	48
Figure 4.2.2-4: Authentication registration in Program.cs.....	49
Figure 4.2.2-5: Appsetting.json configuration .....	49
Figure 4.2.2-6: Login method implementation .....	50
Figure 4.2.2-7: Create token method.....	50
Figure 4.2.2-8:Announcement service folder structure.....	52
Figure 4.2.2-9:Assessment service folder structure .....	53
Figure 4.2.2-10: Course service folder structure.....	54
Figure 4.2.2-11: Communication service folder structure .....	55



Figure 4.3-1: User service database schema .....	56
Figure 4.3-2: Course service database schema.....	57
Figure 4.3-3:Assessment service database schema .....	58
Figure 4.3-4: Announcement service database schema .....	59
Figure 4.3-5: Communication service database schema .....	60
Figure 4.4-1: Ocelot package installment .....	60
Figure 4.4-2:Ocelot.json configuration .....	61
Figure 4.4-3:Program.cs configuration .....	62
Figure 4.4-4: Test result with api gateway .....	63
Figure 4.4-5: Test result with service api.....	63

## LIST OF TABLE

Table 3.1-1 Login use case description.....	21
Table 3.1-2:View content usecase.....	22
Table 3.1-3: View Announcement use case description. ....	22
Table 3.1-4: Create announcement use case description.....	23
Table 3.1-5: Modify announcement use case description .....	24
Table 3.1-6: Create assignments use case description .....	25
Table 3.1-7: Create submission use case description.....	25
Table 3.1-8: Grading use case description .....	26
Table 3.1-9: Create post usecase .....	27
Table 3.1-10: View Post usecase .....	28
Table 3.1-11:Post interaction usecase .....	28
Table 0-1: Comparison with available system .....	66

## ABSTRACT

In today's digital age, educational institutions and organizations are increasingly turning to Learning Management Systems (LMS) to streamline and enhance their learning and training processes. An LMS is a web-based software application designed to facilitate the creation, delivery, management, and assessment of learning content for students, employees, or other learners. The LMS web application offers a comprehensive platform for educators and learners to interact, collaborate, and manage course content. It combines the advantages of traditional classroom-based instruction with online learning components, providing flexibility and personalized learning experiences.

This project aims to provide a robust platform for educators and learners to access and interact with pre-existing learning materials. The blended LMS project leverages a range of features to facilitate content delivery and learner engagement. The LMS platform serves as a centralized repository for pre-existing courses and learning materials, which can be easily uploaded, organized, and accessed by learners. By collaborating with instructors or administrators who have already created these materials, the LMS project ensures that learners can access high-quality resources and engage with them effectively. The content delivery aspect of the blended LMS project focuses on providing learners with a user-friendly interface to navigate through various learning materials. The platform may include features such as multimedia integration, interactive modules, and intuitive navigation, enabling learners to engage with the content in a dynamic and immersive manner. These features aim to enhance learner comprehension, retention, and overall learning outcomes. Furthermore, the blended LMS project emphasizes learner engagement through interactive tools and communication features. Learners can actively participate in discussions, collaborate with peers, and seek clarification from instructors through integrated communication channels. These interactive elements promote learner-to-learner and learner-to-instructor interactions, fostering a sense of community and facilitating knowledge sharing, fostering a dynamic and personalized learning experience.

Keywords: LMS, Blended learning, asynchronous learning, synchronous learning, microservice

## Chapter 1

### INTRODUCTION

#### 1.1. Background

The rapid advancements in technology, particularly the rise of the internet, digital media, and learning management systems which provide new possibilities for delivering and accessing educational content. The arise in technology created opportunities to blend online and offline learning experiences as a response to the evolving needs and opportunities presented by advancements in technology and the changing educational landscape.

Blended learning (BL) is known as a combination of classic in person courses with digital learning. The purpose is to serve learners with well-structured learning concepts where analog and digital components optimally complement each other. The process of BL is switching between traditional and e-learning with online materials alternately and occasionally.

Blended Learning is the association of complex and diverse learning environments to increase the motivation to learn (Yagci, 2017). Learning management system (LMS) platforms serve as central hubs for managing course materials, facilitating communication, and enhancing flexibility within BL environments (Kilag, 2023).

The design is based on a Blended Learning approach which consists of 2 stages and they are Asynchronous learning and Synchronous learning. Blended Online Learning is then offered as an option that combines the characteristics of the two online learning modes. The strengths from each mode can be combined so the occurring problems can be reduced (Cahyani and Ni Made Wahyu Suganti, 2021). While Asynchronous learning refers to online learning with online materials for specific courses on the LMS platform (see Section 2.1), synchronous learning, on the other hand, refers to real-time learning online or face to face meeting (see Section 2.2).

The National Education Association (n.d.) discusses that online courses should be asynchronous and scheduled, since there should be flexibility for students. However, it was stated that there should be a time frame where the students should complete the assignments, and it is an advantage for students to have the activities “24/7” and access them whenever they want to (Amiti and Flora, 2020). This paper will focus more on Asynchronous learning since AL have some strength

and weaknesses can be developed such as authentic learning activities, flexible learning, accessible learning, development of critical thinking (Cahyani and Ni Made Wahyu Suganti, 2021) and student-centered learning and synchronization can be done face to face or by other online meeting platforms.

## 1.2.Problem statement

The common problem that LMS web app nowadays usually faces is how to encourage students in online learning environment, keeping them away from disorientation when learning in online platform since the app lack of interactive and engaging features which can lead to decreased student motivation and participation, students might find it challenging to stay in engaged to the course materials. Lack of understanding materials can lead to decrease of motivation of learning the course because since student doesn't adapt enough knowledge of one lecture it is difficult for them to catch up with the next one, even study in class, student may not give full attention into the lecture as they find it maybe boring or the difficulty is too high for them. Some students may ask for an explanation from the lecturer after class, but some may not. Therefore, students might fail the course. This is a challenging problem for both students and lecturers. If the students can't meet with the teacher, unlike in the traditional classroom, motivation should be given to them in other ways, by making their classes interesting. Assessment is a way of measuring how much the collaboration between the teacher and the students has come into effect. (Amiti and Flora, 2020).

NO	Aspects of Online Learning	Strengths	Challenges
1	Authentic learning activities	√	
2	A flexible learning	√	
3	An accessible learning	√	
4	A live interactive and engaging session		√
5	Development of Critical Thinking	√	
6	Comprehension and Topic Mastery		√
7	Enjoyable Class		√
8	Student-centered mode	√	
9	Connection issues		√
10	Network issues		√

*Figure 1.2-1: Strengths and Challenges of Asynchronous Online Learning (Cahyani and Ni Made Wahyu Suganti, 2021)*

According to Figure 1, we can see that while asynchronous online learning offers flexibility and convenience, it also presents some challenges related to enjoyable classes and comprehension topic mastery.

The second common problem that also decrease the motivation of student is the lack of communication, interaction and feed back between students and lecturers. As in (Gazan, 2020) states that both student-student and student-teacher interactions were significant predictors of student satisfaction and it has an impact on their achievement, we can see that the importance of communication in learning. Some LMS platforms may lack robust communication features, such as real-time chat, discussion forums, or video conferencing capabilities. Without these tools, it can be challenging for students and teachers to engage in meaningful discussions and collaborative activities. While asynchronous communication (e.g., discussion boards, email) can offer flexibility, it may also result in delays in response times and hinder real-time interaction and feedback exchange between students and teachers.

Additionally, other problem related to technology and connectivity such as poor connection, device compability makes a big impact to learning experience. Inadequate internet connectivity, especially in rural or underserved areas, can disrupt access to the LMS and online resources. Unstable connections can lead to dropped sessions, difficulty uploading or accessing course materials, and frustration for students and instructors alike. Limited bandwidth may affect the quality of multimedia content, such as videos or live streams, impacting the effectiveness of online instruction. Lack of computing skill among students and lecturer is also a problem of experiencing LMS as students come from diverse backgrounds with varying levels of technological proficiency. Some may struggle with basic computer skills such as file management, navigating online interfaces, or troubleshooting technical issues. Lack of familiarity with the LMS interface and features may hinder students' ability to fully engage with course materials, participate in discussions, or submit assignments. Students with lower computing skills may require additional support and resources to effectively navigate the digital learning environment, potentially widening the digital divide.

### **1.3.Objective**

The objective of the thesis is to research and implement an blended web app which is the combination of asynchronous learning with traditional synchronous

and guides students learning more efficiently, improving performance of online teaching and learning.

The first problem that need solving are content delivery and content management in a course. The method that will be implemented is chunking (see Section 2.6.1). In the context of content management in an LMS, chunking refers to the practice of organizing course materials into coherent, bite-sized segments that are easier for learners to process and comprehend.

The second objective of this thesis is to solve the communication, interaction problem by offering various communication channels within the LMS, such as discussion forums, messaging features, video conferencing tools, and announcement boards. This allows students and instructors to choose the most suitable channel for their needs. Also, the BL app provides timely and constructive feedback on assignments, discussions, and assessments to help students monitor their progress and improve their performance. Utilize rubrics and grading criteria to ensure consistency and transparency in feedback.

#### **1.4.Assumption and solution**

Assuming that the system all courses in the system have been registered to each student and lecturer. The goal of the web app is to focus on implementing an asynchronous online environment, modify and restructure the learning system for more efficient learning and teaching.

##### **1.4.1. Assumption**

Considering that the student and lecturer accounts have already been created and courses assigned through OAA, certain parts of the BL web app can be ignored or simplified.

- User Registration: Since the accounts are already created, the user registration process can be skipped. Instead, focus on providing a login functionality for students and lecturers to access their respective accounts.
- User Account Management: Since the accounts are pre-existing, creating accounts may not be included. However, users can still change their passwords or update their profile information if necessary.
- Course Enrollment: If courses are already assigned to students and lecturers through another system, the course enrollment process can be skipped. Instead, ensure that the assigned courses are readily available in the respective user accounts without requiring any additional enrollment steps.

- **Course Creation:** Since the courses and assignments are already assigned through externally, lecturers may not need the ability to create new courses within the LMS. However, lecturers can still obtain a simplified interface for to manage existing course materials and assignments if there are any minor adjustments or modifications required for the courses
- **Live Lecture and Virtual Classroom:** Since synchronous activities are face to face, the functionality for conducting live lectures or virtual classrooms within the LMS can be omitted. Instead, focus on providing pre-recorded lecture videos, downloadable materials, and other asynchronous learning resources.
- **Real-Time Communication:** Provide communication tools such as discussion forums, email integration, or messaging systems for asynchronous interaction between students and lecturers instead of live chat, video conferencing, or instant messaging.
- **Synchronous Assessment:** Focus on providing asynchronous assessment options, such as online quizzes, assignments, or essay submissions.

### **1.4.2. Solution**

#### **1.4.2.1.Front end**

**User Interface:** Develop a frontend application that provides an intuitive and user-friendly interface for students and lecturers to interact with the LMS. This frontend can be implemented as a single-page application (SPA) using frameworks like React, Angular, or Vue.js.

**User Authentication:** Implement user authentication and authorization features in the frontend to ensure secure access to the LMS. This can include login and registration forms, password reset functionality, and user profile management.

**Course Catalog and Navigation:** Create a user interface for students to browse and search for available courses. Implement navigation features that allow users to view course details, access course materials, and track their progress.

**Assignment Submission:** Design an interface for students to submit their assignments. This can include file upload functionality, submission forms, and progress tracking for submitted assignments.

#### **1.4.2.2.Backend**



User Management Service: Develop a microservice responsible for user authentication, registration, and profile management. This service will handle user-related operations and store user information securely.

Course Service: Build a microservice that manages course-related functionalities such as course creation, retrieval, and updates. This service will handle operations related to courses, including metadata, materials, and assignments.

Assignment Service: Create a microservice dedicated to managing assignments. It will handle operations like creating new assignments, receiving student submissions, storing files, and providing assignment-related information to the frontend.

Security and Authorization: Implement appropriate security measures, including authentication, authorization, and data encryption. by using JSON Web Tokens (JWT) for secure user authentication and authorization.

## Chapter 2

### LITERATURE REVIEW

Synchronous learning, asynchronous learning, and blended learning are different approaches to delivering educational content and facilitating learning. All three approaches often rely on an LMS or similar online platforms to deliver content, manage course materials, and facilitate communication between instructors and learners. An LMS provides a centralized hub for accessing resources, submitting assignments, participating in discussions, and tracking progress. In all three approaches, digital resources play a vital role in delivering course materials, including lecture videos, e-books, multimedia presentations, online quizzes, and interactive content. Learners can access these resources at their own pace in asynchronous learning, during scheduled sessions in synchronous learning, or as part of the blended learning experience.

#### 2.1. Asynchronous learning

Asynchronous online learning allows students to view instructional materials such as slides, instruction videos,...each week at any time which means students can review or learn the same lecture at different times. Asynchronous classes offer learners the flexibility to study in a self-paced manner. While most asynchronous classes still have submission deadlines, students can connect with materials, peers, and instructors on their own schedules, often over an extended period of time (Kilag, 2023).

Students can study the materials wherever and whenever they like without meeting other students and teachers. The teacher has provided the students with the learning materials the students need to learn and some online activities in an e-learning platform or other online tools. However, the teacher usually gives the students specific timelines (Sari, 2021).

Apart from advantages of asynchronous learning, there are a few critical pain point. There's a certain percentage of student have fear of online assessment especially test or examination due to unstable connection or incompatible devices (Dung, 2020). There's little of communication channels in asynchronous learning. Chat room are usually the only synchronous communication (Papadima-Sophocleous, 2016).

## **2.2.Synchronous learning**

Synchronous learning on the other hand requires the participation of both learners and instructor simultaneously. There are 2 types of Synchronous learning, traditional and online.

### **2.2.1. Traditional synchronous learning**

Synchronous learning required students to participate in a traditional class each week with an instructor and classmate. Alternatively, under some circumstances, such as disease quarantine, students will attend virtual class sessions through an online platform. This class is fixed and cannot be rescheduled.

The benefit of this type of learning is mostly the real time interaction between students and lecturers. It allows student to ask and teacher to answer question instantly, and student to interact with fellow students and instructor (Papadima-Sophocleous, 2016)

### **2.2.2. Synchronous online learning**

The differences between online learning and traditional learning is instructors and students are accessed from different locations and meet through a virtual environment with the help of technologies, it involves the help of a video conferencing platform. Synchronous e-Learning will reduce the imbalances and create a learning environment that is more equitable. The power dynamic of face-to-face learning environment can be avoided, where extroverts can dominate, and where gender and other personal characteristics can affect group activities (Rika Riwayatiningsih, 2020). Synchronous e-learning takes place through lecturers, discussions, online tutorials, etc. (Papadima-Sophocleous, 2016). Synchronous online courses require the instructor and student to interact online simultaneously. Students receive instruction from teacher and interact with their teacher and course mates through texts, audio chats, and video chats in a virtual classroom (Dung, 2020).

The advantages synchronous learning are students can receive feedback directly from fellow students or teachers (Fadhilah, 2021). Hence, some students have difficulties in the traditional classroom, they are introvert by nature, so the environment they are in, in their homes, makes them feel more comfortable, less stressed (Amiti & Flora, 2020). While it can offer much of the engagement available in a classroom, its participants can also benefit from studying in the comfort of their homes and no commute time. For fast learners, this form of

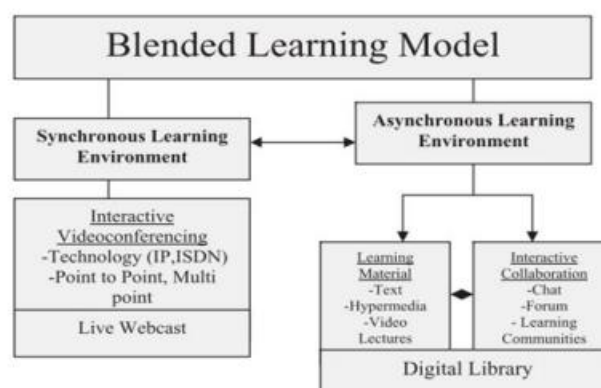
learning can be an improvement over classroom learning as it allows for a more dynamic exploration of concepts, topics and ideas (Rigo, 2021)

Base on the survey of (Dung, 2020) with 256 English-majored students from the first year to the fourth year cohort of the Department of English Language. 205 online questionnaires were collected and used for the analysis of data. In addition, the researcher interviewed 79 students and 22 lecturers on their online learning and teaching experiences. The survey shows the critical disadvantages from both teacher and student viewpoint. From the teacher's viewpoints, Students are often late for class and leave class in the middle of the lesson. Moreover, the majority of the students do not want to turn on camera in virtual classrooms and make the communication between the teacher and the students even more difficult. Students complains about long-hour classes online, which caused tiredness, boredom and concentration loss. Another research from (Gazan, 2020) with English Preparatory School students at a state university in Ankara. The total number of the participants was 24 and they were all at the age of 18 who studied via video-conferencing platform named Zoom (Zoom.us). Ten of the participants confront with technical problems, this included internet connection, computer, earphones, etc. Apart from these problems, as also mentioned by the participants, the tools that were used itself created problems such as poorquality sound, image, delays in emails

### **2.3.Blended (Combination)**

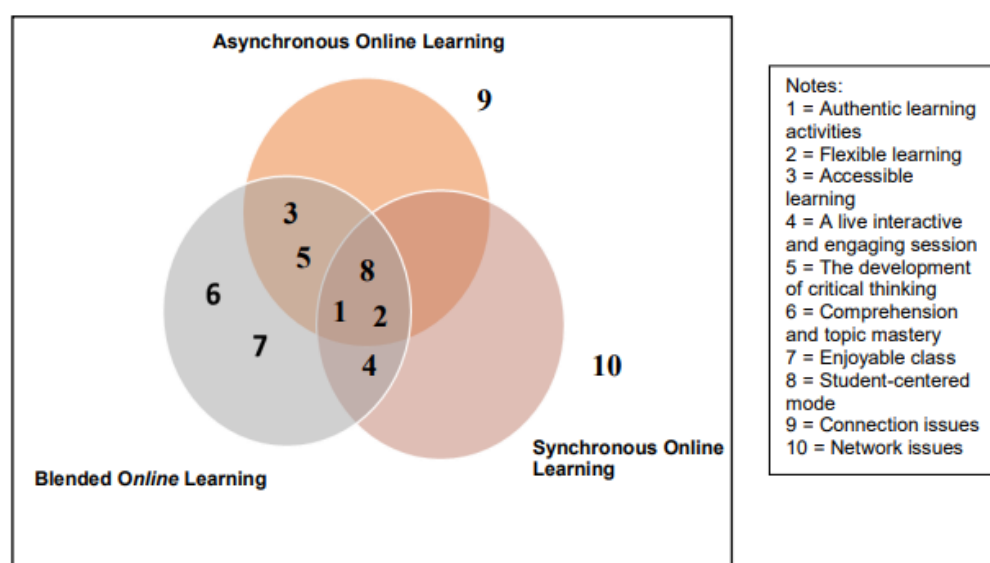
Blended learning (BL), also known as hybrid learning, combines elements of both synchronous and asynchronous learning approaches. It integrates face-to-face or real-time interactions with online, self-paced components. In a blended learning model, learners may attend in-person classes or synchronous sessions with the instructor while also accessing online resources, recorded lectures, or completing self-paced activities asynchronously. BL provides a balance between real-time interaction and flexibility, combining the benefits of both synchronous and asynchronous approaches. It allows for personalized learning experiences, promotes active engagement, and leverages technology to enhance the learning process. BL also means that students may be required to prepare for the lesson before class for more productive face-to-face class activities. Synchronous helps students and lecturers to collaborate their intellectual efforts. Hybrid online courses, alternatively blended courses, facilitate both in-person and online

interaction. Hybrid courses require meeting in-person during a semester and provide for computer-based communication in between those face-to-face sessions (Dung, 2020)



*Figure 2-2.3-1: The blended learning model (Gururaj Bhadri, 2022)*

Based on Figure 2, one related to the BL model proposed above, the process of online learning runs with two modes simultaneously: synchronous learning environment and asynchronous learning environment (Kholis, 2022). Online learning should be a mix of synchronous and asynchronous learning. online learning is accomplished only by one kind, while synchronous learning can be done, for example, when teachers provide new basic competence, their performance techniques, asynchronous learning can be applied to learners' reading and understanding material independently as well as caring for assigned tasks (Fadhilah, 2021).



*Figure 2-2.3-2: Synchronous, Asynchronous, and Blended Online Learning (Cahyani and Ni Made Wahyu Suganti, 2020)*

Figure 3 illustrates four areas of overlap in strengths among the three online learning modes. The initial overlap encompasses Synchronous, Asynchronous, and Blended Online Learning, represented by points 1, 2, and 8, respectively. This indicates that these modes share common attributes such as authentic learning activities, flexible learning approaches, and a student-centered learning environment. The second overlap, involving Asynchronous Online Learning and Blended Online Learning (points 3 and 5), highlights their shared characteristics of accessibility and fostering critical thinking skills among students. The third overlap, represented by the combination of Synchronous and Asynchronous Learning (point 4), showcases the aspect of live, interactive, and engaging sessions, particularly in the EFL (English as a Foreign Language) context. Lastly, the fourth overlap denotes the additional benefits unique to Blended Online Learning itself (points 6 and 7), which include improved comprehension, mastery of topics, and an enjoyable learning experience provided by the blended approach.

## **2.4. Similar system**

### **2.4.1. Emodo**

Emodo is an educational technology platform that aims to enhance learning outcomes through personalized learning experiences and data-driven insights. It provides tools for educators to create, deliver, and analyze instructional content, assessments, and interventions tailored to individual student needs (Edmodo, 2023), shown in Figure 2.4-1.

Emodo provides a variety of formative assessment tools, including quizzes, surveys, and interactive activities, to gauge student understanding and track learning progress in real time. Educators can create customized assessments, monitor student responses, and provide timely feedback to support ongoing learning. Edmodo provides its users with the feature of instant messaging, which is, however, limited only to the exchange of messages between the teacher and their students. Edmodo notifies its users, both teachers and students, about each relevant activity that takes place within a study group. A notification can be sent either as an e-mail notification, a mobile application notification or a text message (Rigo, 2021).



*Figure 2.4-1: Edmodo platform interface (Edmodo, 2023)*

#### **2.4.2. Blackboard Learn**

Blackboard Learn is one of the most popular LMS platforms used by educational institutions worldwide. It provides tools for course creation, content management, gradebook management, discussion forums, assessments, and collaboration. Blackboard Learn supports both synchronous and asynchronous learning activities, allowing instructors to deliver a variety of instructional materials and engage students in online learning experiences.

Blackboard has received consistently high ratings on various review websites such as SoftwareAdvice, G2, and TrustRadius, often achieving over 4 stars in categories like Ease of Use, Assignment Management, Course Variety, and Assessment Tools. Despite this, it remains a popular platform used by many schools and universities.

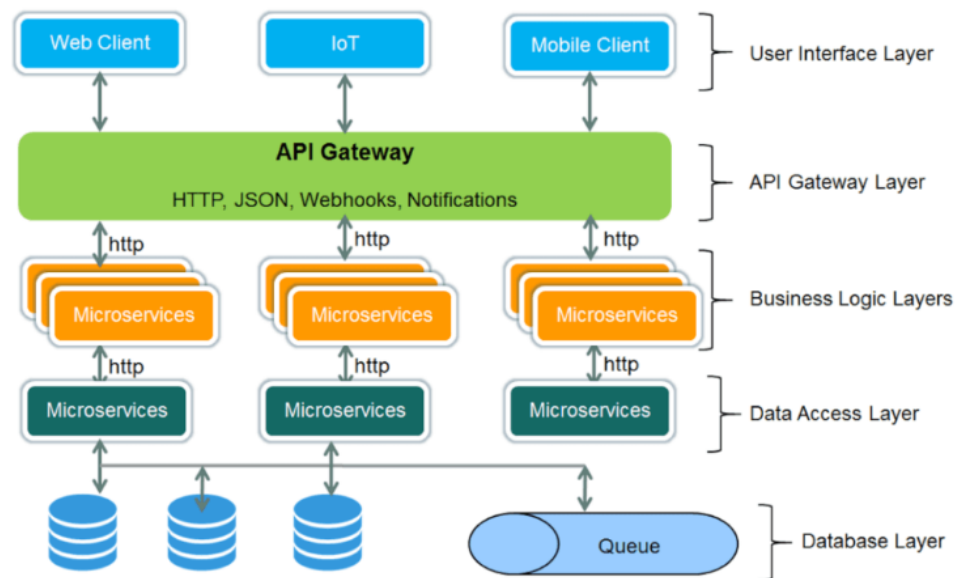
### **2.5. Microservice Architecture**

#### **2.5.1. Introduction**

Microservice is known as an architecture used to build multi platform applications, it soon evolved into a well-known method to build software. This architecture divides the structure of an application into collections of independent deployable, loosely coupled services that eliminates most of the disadvantages of the old-school architecture such as monolithic modules whose modules are aggregated into a single project.

Microservice has become a tendency in software development and Domain Driven Design (DDD) is the initiative concept. it is about focusing on the domain including its concepts, their relationships and business logic. Microservice architecture is about arranging and dividing distributed software building blocks (Steinegger, 2017). Also,DDD is a philosophy that illustrates the understanding and modeling of the business domain, developers, domain experts and users communicate a shared language across aspects of the software development process called Ubiquitous Language. This shared language will make sure everyone has a deep understanding, facilitating smoother collaboration.

One big advantage of microservices is that they are not tied to a programming language. They also overcome the cumbersomeness of dealing with databases (Isak Shabani, 2021)



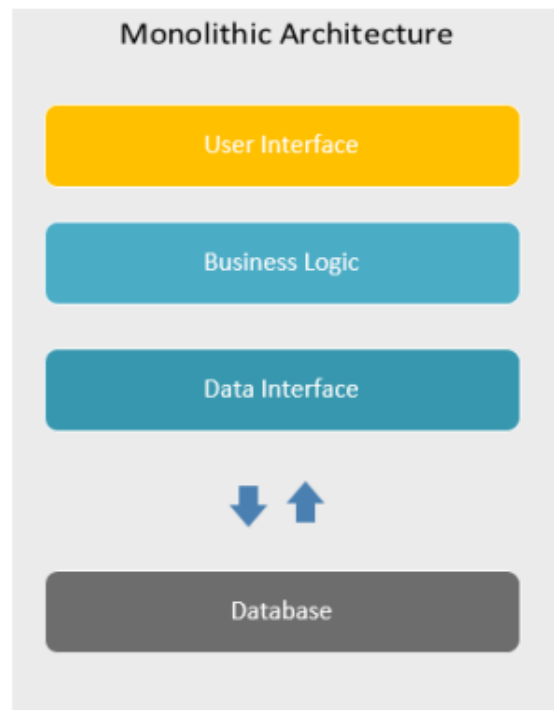
*Figure 2-4:Microservice architecture model*

The above Figure 5 is microservice architecture there are many variations of microservice models, each one depending on the qualification of the application. According to the above model, each microservice is actually an independent, deployable service, it has separate databases and owns the right to access it. Because of the independent and loosely coupled properties, all services communicate not directly but through an api gateway.

With issues regarding maintenance, response time and scaling, monolithic architecture should be avoided when designing large and complex applications which may be used in different environments with different configurations or in



applications which may change and need to be frequently updated (Isak Shabani, 2021), that's when microservices reveal its true advantage.



*Figure 2-6: Monolithic architecture model*

Scalability should also be considered. In the monolithic approach scaling can be done only in one dimension. Multiple copies of the application can be placed behind the load balancer, and this causes increase in the number of client requests that can be simultaneously served (Milovanović and Ana, 2021).

### **2.5.2. Domain-Driven Design**

Domain-Driven Design (DDD) is an approach from use case application to real business logic. It helps developers to design and construct models that satisfy the business requirement of a domain. It also allows developers to plan and scale the larger system into services. Main concepts of DDD that need to be considered are: Bounded Context, Aggregate, Ubiquitous Language. These concepts are mentioned in Section 2.5.1, now we will discuss each in detail.

#### **2.5.2.1. Bounded Context**

Bounded Context in the context of DDD means split, decomponent big system into self-contained components. Each Domain has its own boundary, explicit responsibilities need to be carried out. Bounded contexts hide implementation detail, this internal concern should be hidden to the outside world, which doesn't need to know, nor should it care (Newman, Building

Microservices: Designing Fine-Grained Systems, 2021). By this context, DDD becomes appropriate design for microservices.

#### **2.5.2.2. Aggregate**

An aggregate is a fundamental concept used to group together related objects and define transactional consistency boundaries within a domain model. Aggregates represent a cluster of objects that are treated as a single unit, ensuring that their internal consistency is maintained. An aggregate consists of an aggregate root and its associated entities and value objects. The aggregate root is the primary entity within the aggregate, and it is responsible for maintaining the consistency and integrity of the aggregate. The aggregate root is the only object that can be directly referenced from outside the aggregate. Aggregates typically refer to real-world concepts. The key thing to understand here is that if an outside party requests a state transition in an aggregate, the aggregate can say no (Newman, Building Microservices: Designing Fine-grained Systems, 2021).

#### **2.5.2.3. Ubiquitous Language**

Ubiquitous Language is a common vocabulary that is used by both domain experts and developers to describe and discuss the domain concepts, processes, and rules. It helps bridge the gap between the domain experts' understanding of the problem domain and the implementation in software. The idea is that having a common language between the delivery team and the actual people will make it easier to model real-world domains and also should improve communication (Newman, Building Microservices: Designing Fine-grained Systems, 2021). By using the Ubiquitous Language, the development team can create a more effective and accurate representation of the domain within the software code and design. It ensures that the software model aligns closely with the domain model, leading to better communication, reduced ambiguity, and improved collaboration between domain experts and developers.

### **2.6. Techniques**

#### **2.6.1. Technology**

##### **2.6.1.1. ASP.NET**

ASP.NET is a web application framework developed by Microsoft that allows developers to build dynamic websites, web applications, and web services. It is part of the larger .NET framework and provides a programming

model, a comprehensive software infrastructure, and various tools for building and deploying web applications. The main functionality of asp.net in this project is to implement microservice architecture. Each microservice is typically implemented as a separate ASP.NET Web API application. This allows for independent development, deployment, and scaling of individual services.

#### **2.6.1.2.Nextjs**

Next.js is a popular open-source React framework that enables server-side rendering (SSR), static site generation (SSG), and client-side rendering (CSR) for React applications. Developed by Vercel, Next.js provides a powerful toolset for building modern web applications with React while addressing performance, SEO, and developer experience concerns.

Main feature of Nextjs use in this project is API Route which create API routes within your application to interact with your microservices backend. You can define these routes in the pages/api directory and use them to fetch data, handle user authentication, and perform other backend operations. Routing defines routing rules in your Next.js application to navigate between different views and components. You can use dynamic routes to handle URLs that correspond to specific resources or endpoints in your microservices backend.

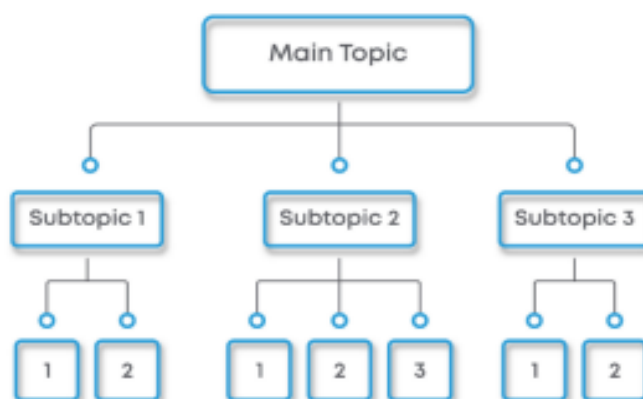
#### **2.6.1.3.Postgresql**

PostgreSQL is a powerful open-source relational database management system (RDBMS) known for its robustness, reliability, and advanced features. It offers a wide range of capabilities for storing, managing, and querying structured data, making it a popular choice for various applications ranging from small projects to large-scale enterprise systems.

### **2.6.2. Chunking**

The project will break down course learning materials into smaller tasks. Large chunks of information can be overwhelming in asynchronous learning. Breaking down into smaller, manageable tasks to set specific goals for each study session and celebrate achievements along the way. This approach will help students stay motivated and maintain a sense of progress. This technique is called chunking. Chunking is an act of breaking a component into smaller units called “chunks” of related information (shown in Figure 2.6.2-1). It involves breaking a

large topic into smaller, more digestible “chunks” of information and presenting these chunks of content one at a time to learners (Gururaj Bhadri, and L. P, 2022)



*Figure 2.5.2-1:Tree diagram to organize chunks (Gururaj Bhadri, and L. P , 2022)*

### **2.6.3. Multimedia resources**

By taking advantage of the diverse range of multimedia resources available in asynchronous learning, watch instructional videos, listen to podcasts, or explore interactive simulations and activities related to the course. Engaging with different formats can make students experience more enjoyable and enhance comprehension. Pre-recorded instruction video pre-lecture slides, documents will be provided before face to face class so students have a good preparation and enough knowledge before going to class. This way reduced the chance of students being “lost” from in-class lectures. This method solved problem stated in Chapter 1 by providing materials before class and maybe small self-quizzes, practice tests, or review questions provided by the instructor to gauge student’s comprehension. Identify areas where they need to focus more and review accordingly. Self-assessment will help students identify their strengths and weaknesses, allowing them to allocate their study time effectively

## Chapter 3

### METHODOLOGY

Blended learning (BL) web application revolutionizes the education landscape by seamlessly integrating asynchronous and synchronous learning methodologies through the innovative use of microservice architecture. Leveraging the power of chunking and multiple media channels, our platform offers a dynamic and immersive learning experience tailored to meet the diverse needs of modern learners.

#### 3.1.Overview

The BL web app aims for connecting lecturers and students by delivering online resources. The web app follows microservice architecture for more flexible development, deployment, and maintenance of the application.

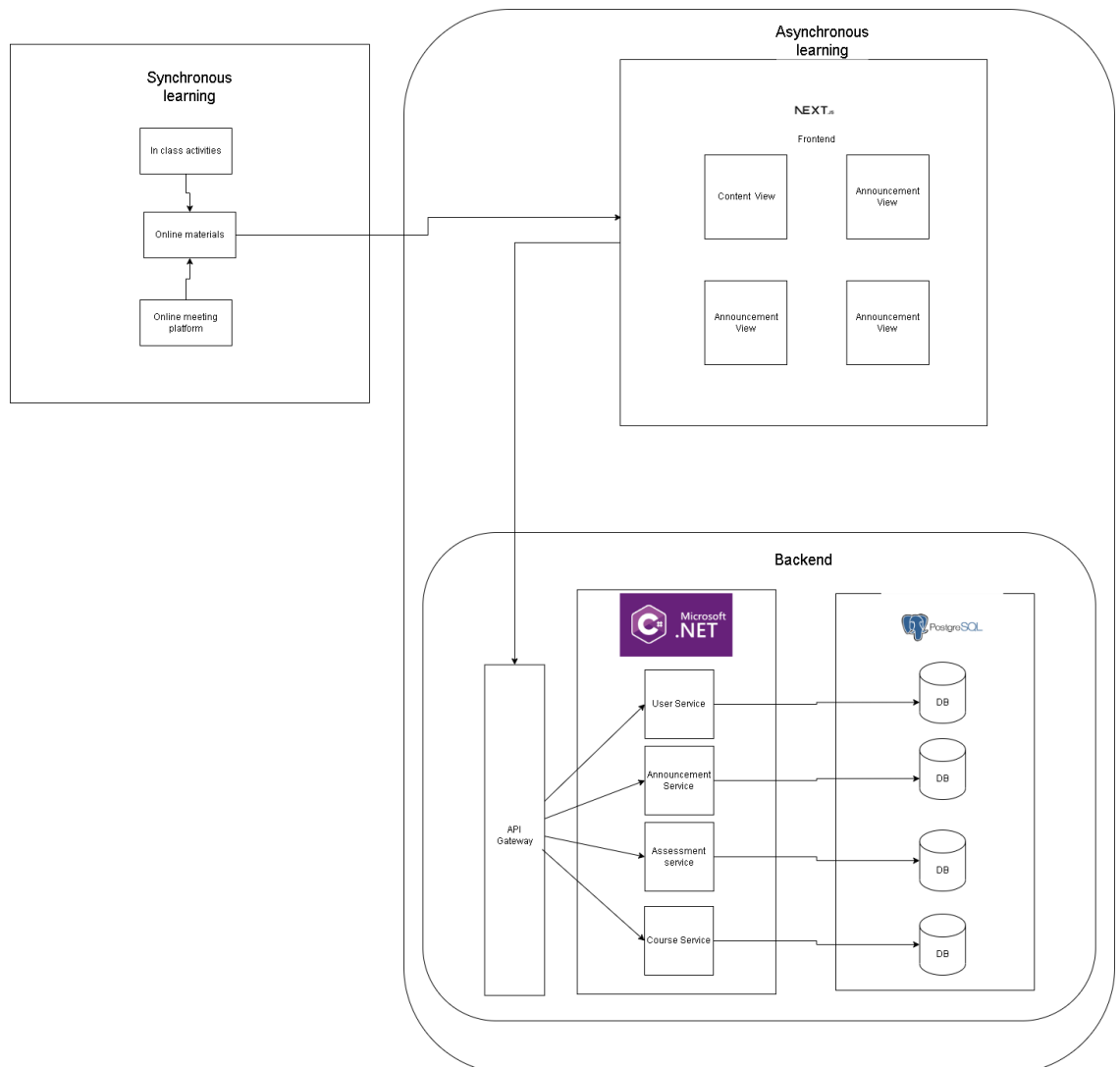


Figure 3.1-1: System Diagram

Blended learning is a combination of synchronous learning and asynchronous learning as shown in Figure 8.

Synchronous learning is activity that will be hold in traditional face to face class or in any special case like quarantine, it also be hold by online meeting platform such as Google Meet, Microsoft team, etc the common thing of traditional and online platform is it use online materials which can be received through web app.

The BL web app will focus on the asynchronous part which involves pre-recorded lectures, reading materials, discussion forums, and assignments that can be accessed and completed at any time. Learners have the freedom to choose when and where they engage with the learning materials based on their personal schedules and preferences. The front end of the app will be implemented by Nextjs framework, which will provide an interface to provide material for both synchronous and asynchronous. After being selected in the client side, the request will be sent to the backend through the api gateway, which will choose the appropriate service to call according to the request (see more in Chapter 4).

## 3.2. Diagrams

### 3.2.1. Use case

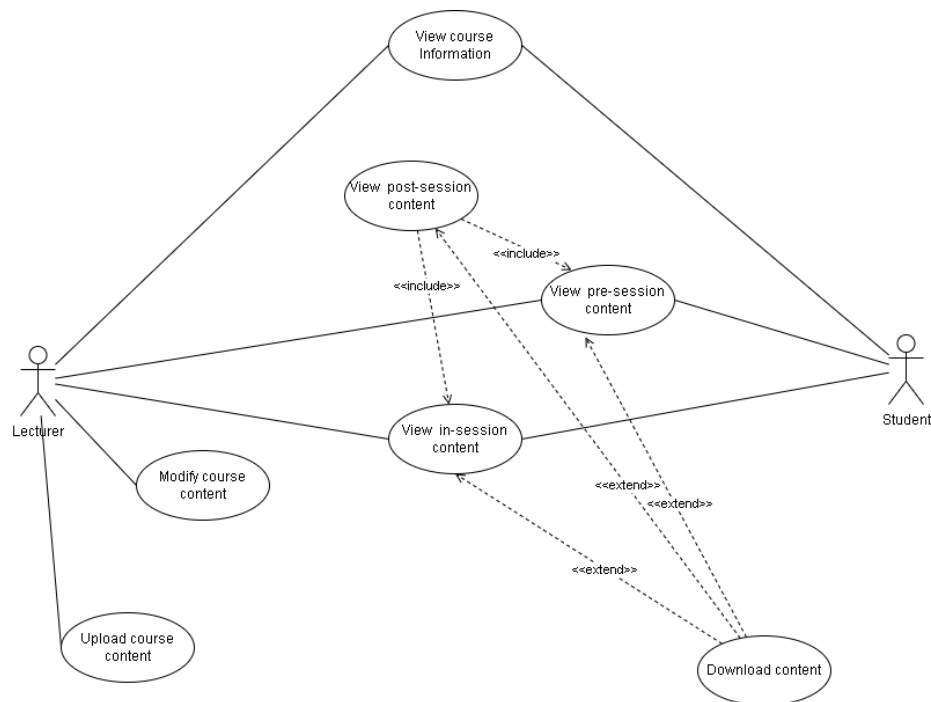


Figure 3.1-2: View course content use case

Figure 3.1-2 shows the viewing the course materials use case, there are 2 actors in this use case which are lecturer and student. Lecturer can uploads, modifies and views course information and its materials. Student can also view the course material but in an order pre-session material, in-session materials, after doing all the work in 2

session, they can view post-session materials, all the content can be downloaded by lecturer and student

Use case ID	UC-1
Use case Name	Login
Description	Students, Lecturers gain access to web app to view content
Actor(s)	Student, Lecturer
Priority	Must have
Precondition	Student, Lecturer is a member of university and have an account to login
Post-condition	User log in to the web site and study, teach
Basic Flow	<ul style="list-style-type: none"> <li>- User input their provided username and password to the form.</li> <li>- User press login and successfully access to the web app</li> </ul>
Alternative Flow	None
Exception Flow	User input invalid information, system refuse to login

*Table 3.1-1 Login use case description*

Use case ID	UC-1.1
Use case Name	View content
Description	Students, Lecturers views content of a course
Actor(s)	Student, Lecturer
Priority	Normal
Precondition	Student, Lecturer has logged in
Post-condition	None
Basic Flow	<ul style="list-style-type: none"> <li>- Users choose specific course</li> <li>- User navigateto content tab</li> </ul>
Alternative Flow	None

Table 3.1-2: View content usecase

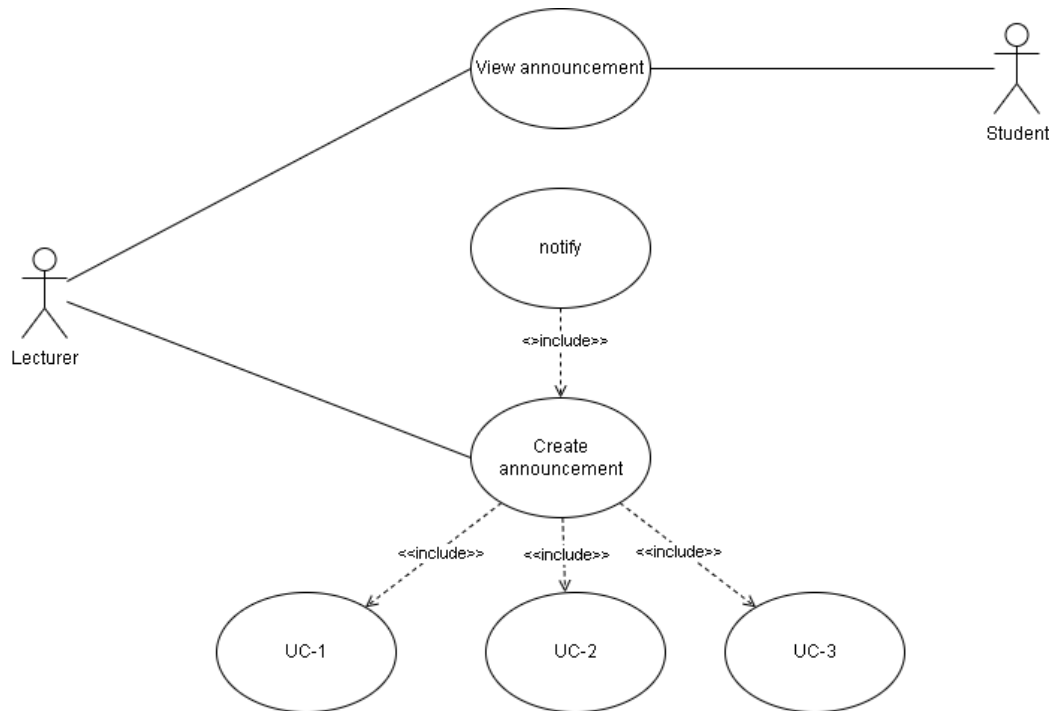


Figure 3.1-3: View Announcement use case

Use case ID	UC-2
Use case Name	View announcement
Description	Students, Lecturers want to view announcement of specific course or all announcements
Actor(s)	Student, Lecturer
Priority	Normal
Precondition	UC-1
Post-condition	User can see announcements of the course or all announcements
Basic Flow	User access list of announcements via navigation bar or through a course
Alternative Flow	New announcement will be notified when user login account
Exception Flow	None

Table 3.1-3: View Announcement use case description.



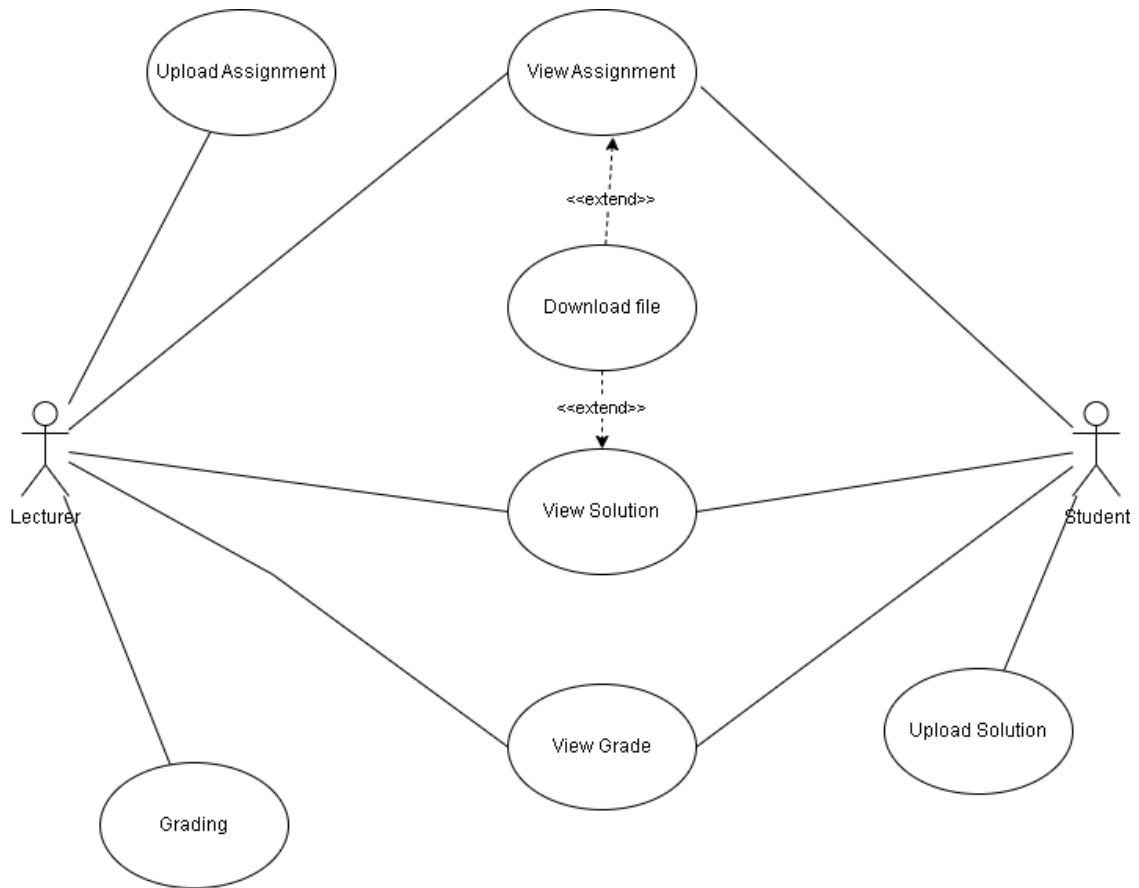
Use case ID	UC-2.1
Use case Name	Create announcement
Description	Lecturers wants to create a new announcement in a specific course
Actor(s)	Lecturer
Priority	Normal
Precondition	UC-1
Post-condition	Announcement(s) is created and notify to all student of that course
Basic Flow	<ul style="list-style-type: none"> <li>- Lecturer access course</li> <li>- Lecturer navigates to announcement panel.</li> <li>- Lecturer press create new announcement button.</li> <li>- Fill in the form.</li> <li>- Announcement is successfully created</li> </ul>
Alternative Flow	Lecturer post new assignment, content or quiz will also create new announcement
Exception Flow	Invalid form information, announcement and not be created

*Table 3.1-4: Create announcement use case description*

Use case ID	UC-2.2
Use case Name	Modify announcement
Description	Lecturers wants to modify an announcement in a specific course
Actor(s)	Lecturer
Priority	Normal
Precondition	UC-1,UC-2.1
Post-condition	Announcement(s) is modified
Basic Flow	<ul style="list-style-type: none"> <li>- Lecturer access course</li> <li>- Lecturer navigates to announcement panel</li> </ul>

	<ul style="list-style-type: none"> <li>- Lecturer view an announcement</li> <li>- Fill in the form.</li> <li>- Announcement is successfully modified</li> </ul>
Alternative Flow	None
Exception Flow	Invalid form information, announcement and not be modified

*Table 3.1-5: Modify announcement use case description*



*Figure 3.1-4:Assessment usecase*

Use case ID	UC-3
Use case Name	Create assignment
Description	Lecturers want to create new assignment for the specific course
Actor(s)	Lecturer
Priority	Normal
Precondition	UC-1

Post-condition	Assignment is created
Basic Flow	<ul style="list-style-type: none"> <li>- Lecturer navigates course</li> <li>- Lecturer navigates to assignment panel</li> <li>- Lecturer navigates create assignment button</li> <li>- Fill in the form</li> <li>- Assignment is successfully modified</li> </ul>
Alternative Flow	None
Exception Flow	Invalid form information, assignment can not be created

*Table 3.1-6: Create assignments use case description*

Use case ID	UC-4
Use case Name	Assignment submission
Description	Students want to submit solution to available assignment
Actor(s)	Student
Priority	Normal
Precondition	UC-1,UC3
Post-condition	Submission is created
Basic Flow	<ul style="list-style-type: none"> <li>- Students access a course</li> <li>- Student navigates to assignment panel</li> <li>- Student access a assignment</li> <li>- Fill in the form</li> <li>- Submission is successfully created</li> </ul>
Alternative Flow	None
Exception Flow	Invalid form information, submission can not be created

*Table 3.1-7: Create submission use case description*

Use case ID	UC-5
Use case Name	Grading submission
Description	Lecturer want to submit solution to available assignment
Actor(s)	Lecturer
Priority	Normal
Precondition	UC-1, UC4
Post-condition	Grading submission of a student
Basic Flow	<ul style="list-style-type: none"> <li>- Lecturers access a course.</li> <li>- Lecturer navigates to grading center.</li> <li>- Lecturer views list of submission</li> <li>- Choose a submission.</li> <li>- Download submission</li> <li>- View submission and grading</li> </ul>
Alternative Flow	None
Exception Flow	There's no submission in a specific assignment, the score will be 0

*Table 3.1-8: Grading use case description*

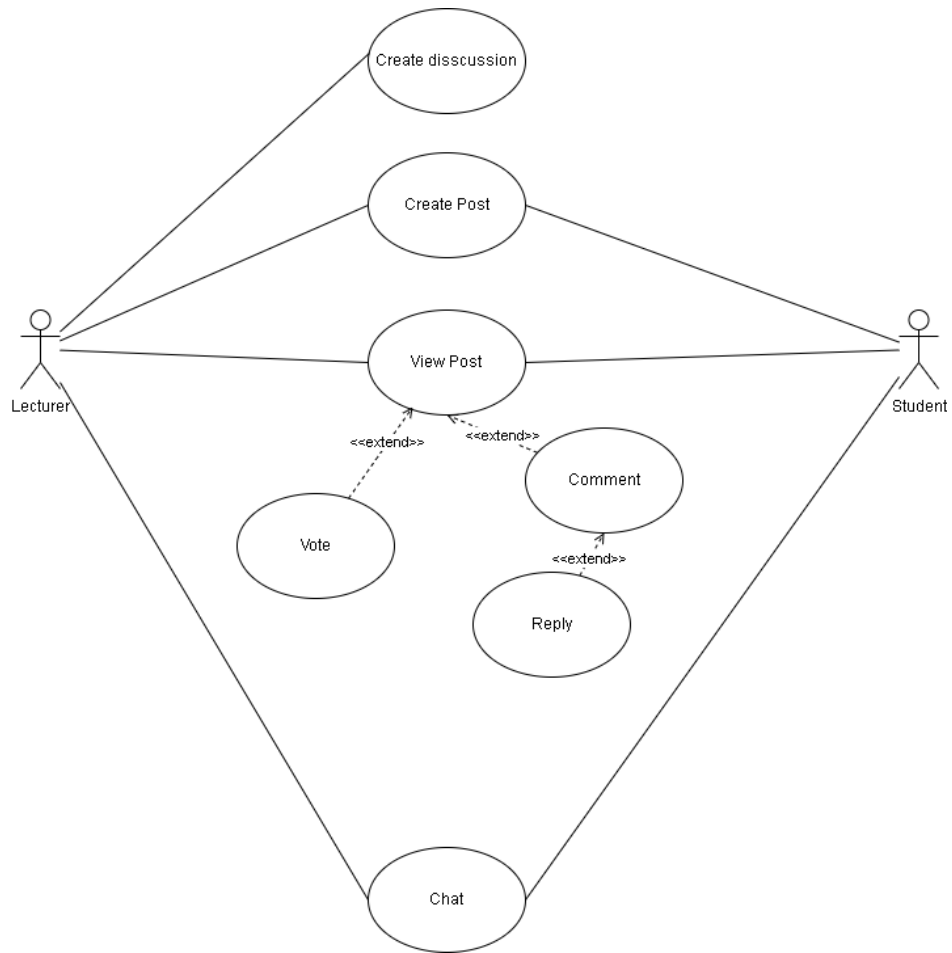


Figure 3.1-5: Communication usecase

Use case ID	UC-6
Use case Name	Create Post
Description	User wants to create new discussion post
Actor(s)	Lecturers, Students
Priority	Normal
Precondition	UC-1
Post-condition	None
Basic Flow	<ul style="list-style-type: none"> <li>- Users navigate to create post button to go to create post form</li> <li>- Users fill out the form and submit</li> <li>- New post is successfully posted</li> </ul>
Alternative Flow	None
Exception Flow	None

Table 3.1-9: Create post usecase

Use case ID	UC-7
Use case Name	View Post
Description	User wants to view discussion post
Actor(s)	Lecturers, Students
Priority	Normal
Precondition	UC-1,UC-6
Post-condition	None
Basic Flow	- Users views Post directly in homepage
Alternative Flow	- User view Post through navigate by courses
Exception Flow	None

*Table 3.1-10: View Post usecase*

Use case ID	UC-8
Use case Name	Post interaction
Description	User interact with post by voting and comment.
Actor(s)	Lecturers, Students
Priority	Normal
Precondition	UC-1
Post-condition	None
Basic Flow	- User views Post - User presses upvote/downvote to raise/decrease votes - User comments/replies to a post
Alternative Flow	None
Exception Flow	None

*Table 3.1-11:Post interaction usecase*

### 3.2.2. Activity diagram

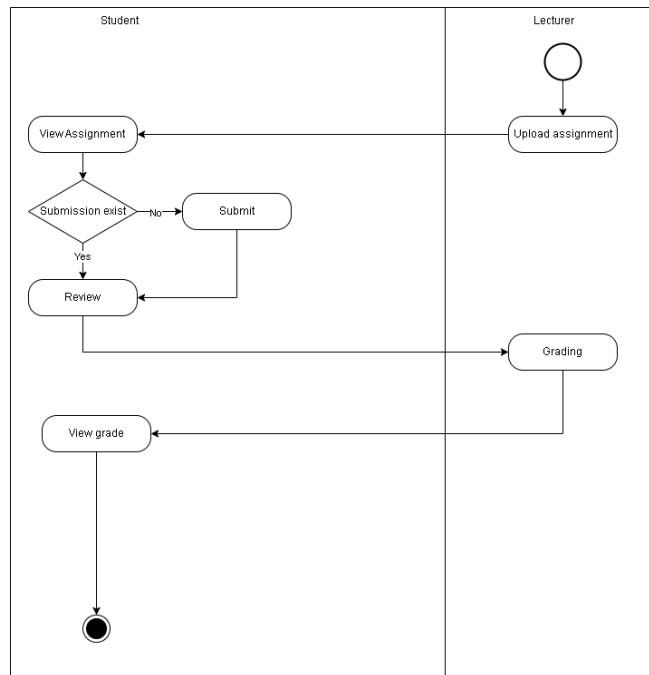


Figure 3.1-6: Grading activity

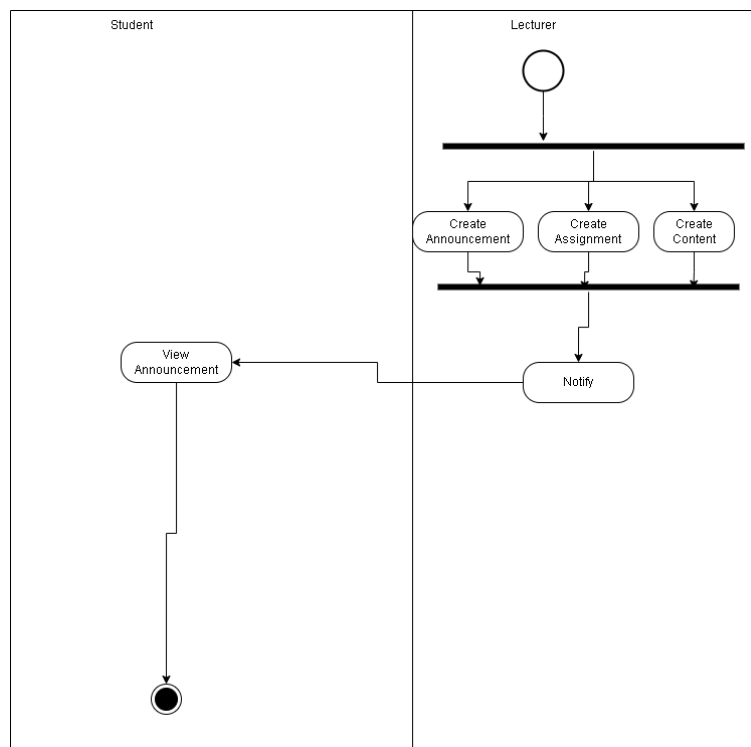


Figure 3.1-7: Announcement service activity

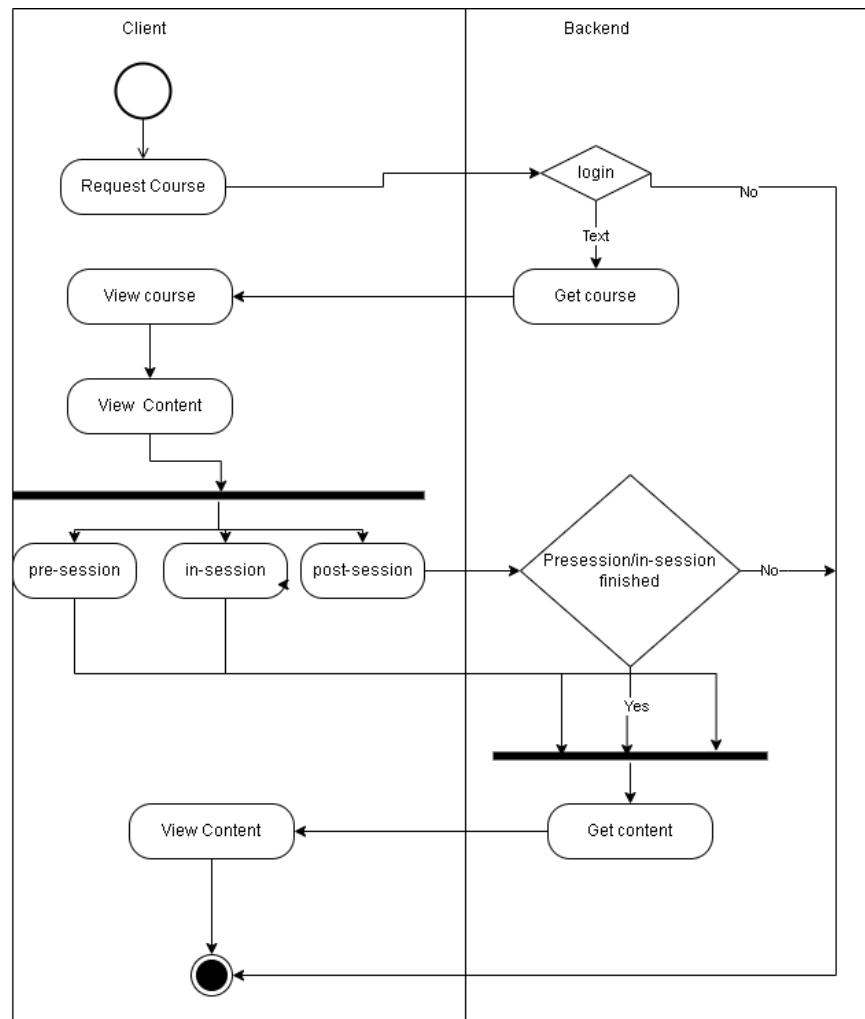
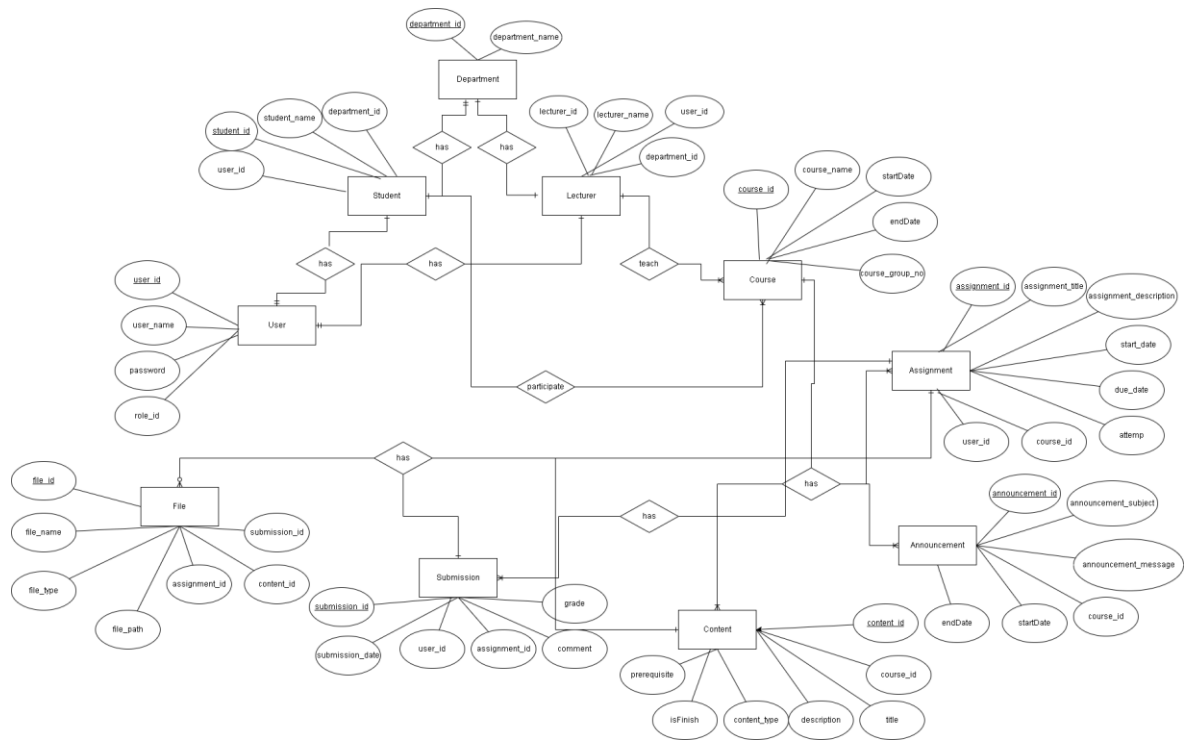


Figure 3.1-8: Course service activity

### 3.2.3. ERD Diagram





*Figure 3.1-9: ERD diagram*

There are 9 entities in this project, those are user, announcement, student, lecturer, course, assignment, submission, content and file entity.

User entity has 4 attributes and those are user\_id to identify each individual user, it is also the primary key (PK) of user entity. Username and password attribute used to login into the account. Role\_id is used to distinguish between student and lecturer. It establishes a relationship between the user entity and the role entity, allowing users to be assigned specific roles or access levels in the system. The role\_id serves as a foreign key, linking the user entity to the corresponding role entity.

The student entity can facilitate student management and allow for relationships between students and their respective departments. The department\_id foreign key establishes a link between the student entity and the department entity, enabling queries and operations that involve department-specific information for each student. The department\_id serves as a foreign key, linking the student entity to the corresponding department entity. Student\_id (PK) serves as the unique identifier for each student in the system. It ensures that each student has a distinct identification within the database. The student\_name attribute stores the name of the student. It can be a string value that represents the full name or a combination of first name and last name.

The lecturer entity can facilitate lecturer management and allow for relationships between lecturers, departments, and user accounts. Similar to student entity, the department\_id and user\_id FK serves the same purpose is to establishes a relationship between the lecturer entity and the department entity, allowing each lecturer to be associated with a specific department and establishes a relationship between the lecturer entity and the user entity, allowing the lecturer to have a user account within the system.

Course entity describes information of a single course with 5 attributes: course\_id (PK) is a unique identifier for each course in the system. It ensures that each course has a distinct identification within the database. Course\_name stores the name or title of the course. It represents the name by which the course is commonly known. startDate: The startDate attribute represents the date when the course is scheduled to start. It can be a date or timestamp value that indicates the beginning of the course. endDate: The endDate attribute represents the date when the course is scheduled to end. It is a timestamp value that indicates the completion date of the course. course\_group\_no attribute represents a grouping or categorization of courses. It can be a numerical or alphanumeric value that classifies courses into specific groups or categories within the system. This attribute can be useful for organizing and managing courses based on similar characteristics or requirements.

Announcement entity contains information of announcement within a course with announcement\_id (PK) serves as the unique identifier for each announcement in the system. It ensures that each announcement has a distinct identification within the database. Announcement\_subject stores the subject or title of the announcement. It represents a concise description of the announcement's content. announcement\_message: The announcement\_message attribute stores the actual content or message of the announcement. It can be a text field that includes the details, instructions, or any other relevant information to be communicated to the intended recipients. course\_id attribute represents the course to which the announcement is associated. It serves as a foreign key, linking the announcement entity to the corresponding course entity. This attribute establishes a relationship between the announcement and the specific course it pertains to. startDate represents the date when the announcement becomes effective or valid. It can be a date or timestamp value indicating the start date of the announcement's relevance. The endDate attribute represents the date when the announcement

expires or is no longer applicable. It can be a date or timestamp value indicating the end date of the announcement's relevance.

Content entity manages content information in a course such as slides, lecture videos, .. Content\_id (PK): is the unique identifier for each content item in the system. It ensures the file and the content item it pertains to, and the submission\_id attribute associates the file with the submission it belongs to. The file\_path attribute provides the location of the file for easy retrieval and access. These attributes collectively facilitate the management and retrieval of files within an educational or document management system. The file\_type attribute indicates the type or format of the file, such as "PDF," "Word document," "image," "video," etc. It specifies the nature of the file content. file\_path: The file\_path attribute stores the location or path where the file is stored in the system, either on the local storage or a remote server. It allows for easy retrieval and access to the file as needed.

### 3.2.4. Class Diagram

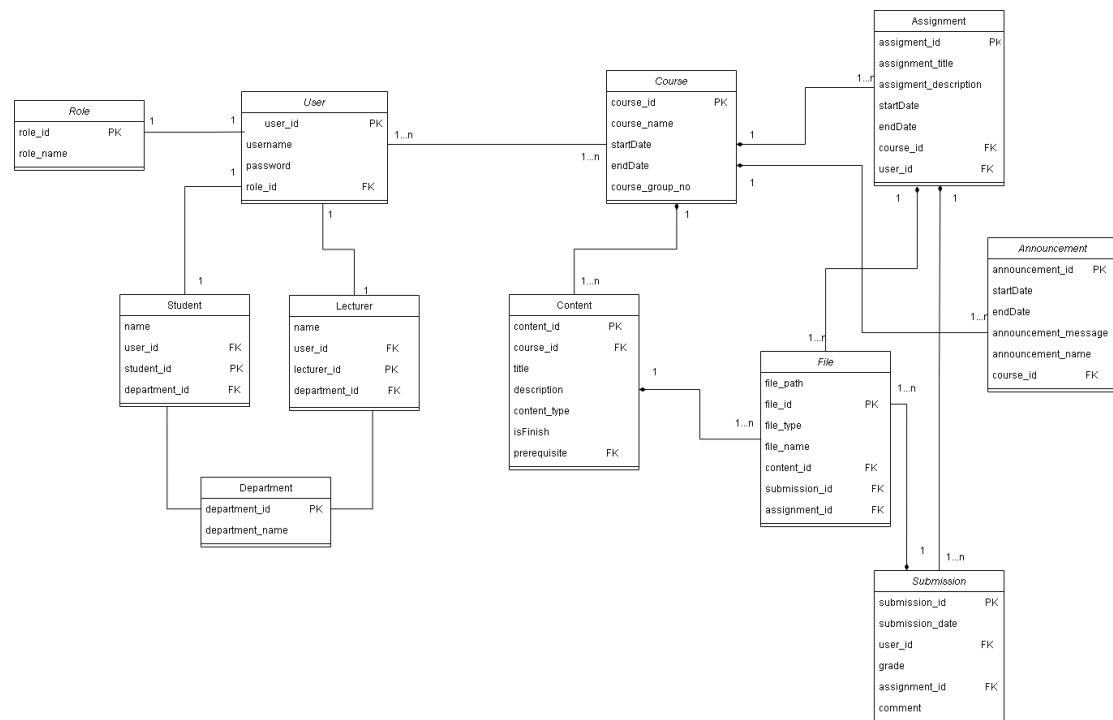


Figure 3.1-10: Class diagram

Users have a composition relationship with Role, as each user has a specific role. Students and Lecturers each have a user account. Therefore, it forms a relationship with User entities, representing the specialization of users. Lecturers and students also have an aggregation relationship with the Department class,

indicating that a lecturer and student belong to a specific department. Assignment has an aggregation relationship with User (creator), indicating that a user creates an assignment. It also has a composition relationship with Submission, as multiple submissions are associated with a single assignment. Announcements have an aggregation relationship with User (creator), indicating that a user creates an announcement. File has an aggregation relationship with User (uploader), indicating that a user uploads a file. It also has an aggregation relationship with Assignment, Content and Submission entity as each one must have 1 or multiple files. Submission has an aggregation relationship with Student and Assignment, indicating the association with a specific student and assignment. Content has an aggregation relationship with User (creator), indicating that a user creates the content. Content also has an aggregation relationship with Course entity indicating that a course may have 1 or more content materials.

### 3.3.Sequence diagram

#### 3.3.1. Authorization and Authentication

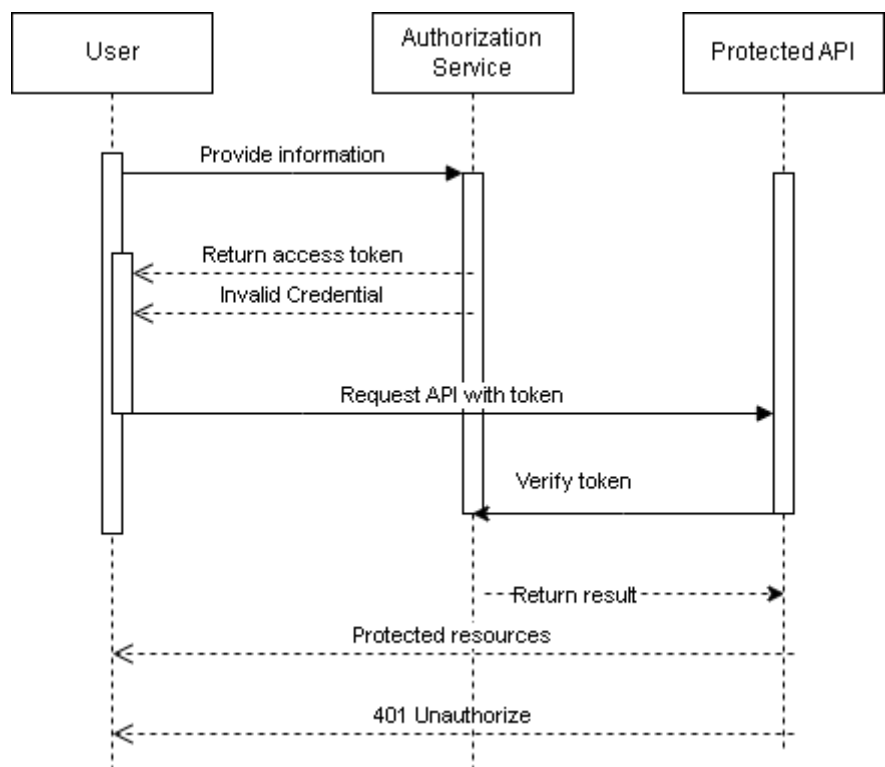


Figure 3.3-1: Authorization and authentication sequence diagram

This project uses JSON Web Token (JWT) to implement authorization and authentication, JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information

between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA. (“JSON Web Token Introduction - jwt.io”)

Here are the key components of a JWT:

**Header:** The header defines the type of token (JWT) and the signing algorithm used. It is base64url-encoded and typically consists of two parts: the token type and the signing algorithm.

**Payload:** The payload contains the claims or statements about the user or other data being transmitted, such as the user ID, roles, expiration time, or custom data. The claims can be categorized into registered claims, public claims, and private claims. The payload is also base64url-encoded.

**Signature:** The signature is created by combining the base64url-encoded header, payload, and a secret key (in the case of HMAC-based algorithms) or using a private key (in the case of RSA or ECDSA-based algorithms). The signature ensures the integrity of the JWT and can be used to verify its authenticity.

The process of JWT (JSON Web Token) authentication typically involves the following steps:

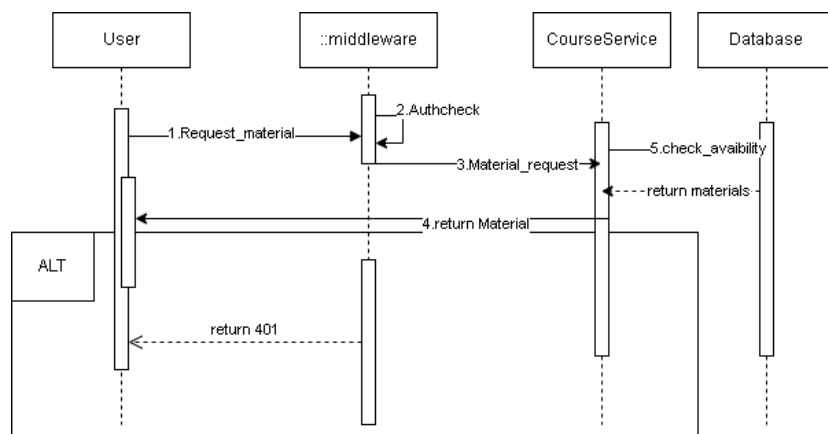
**User Authentication:** The user provides their credentials, such as a username and password, to the authentication server or system. The authentication server verifies the user's credentials, typically by checking against a user database or an external authentication provider. If the credentials are valid, the authentication server generates a JWT.

**JWT Generation:** The authentication server creates a JWT, which consists of three parts: header, payload, and signature. The header typically contains the algorithm used for signing the token (e.g., HMAC, RSA), and the payload contains the claims (e.g., user ID, expiration time, roles). The server signs the JWT using a secret key or private key, creating the signature.

**JWT Issuance:** The authentication server sends the JWT back to the client as a response to the successful authentication request. It can be included in the response body or as an HTTP header (e.g., Authorization header). **Client Storage:** The client (e.g., web browser, mobile app) receives the JWT and stores it securely. In web applications, it is commonly stored in a client-side storage mechanism, such as local storage or cookies.

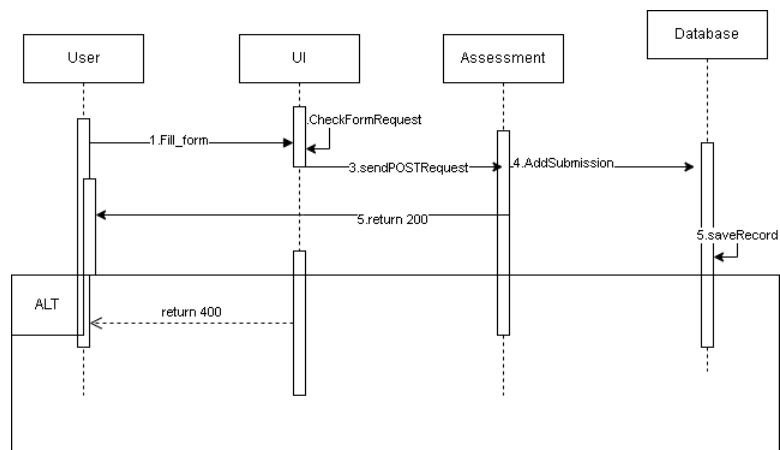
**JWT Usage:** For subsequent requests, the client includes the JWT as part of the request, typically in the Authorization header. The server receives the JWT and validates it to ensure its integrity and authenticity. The server verifies the signature using the secret or public key associated with the issuer of the JWT. If the signature is valid, the server decodes the JWT to extract the user information and performs any necessary authorization checks. **Authentication and Authorization:** The server uses the extracted user information to authenticate the user and authorize the requested resources or actions. The server may also check the token's expiration time and other claims to ensure the token is still valid (see section Chapter 4, Section 2.2.3).

### 3.3.2. Request course materials



*Figure 3.3-2:Request material sequence diagram*

### 3.3.3. Submission



*Figure 3.3-3:Submission sequence diagram*

## Chapter 4

### IMPLEMENT AND RESULT

#### 4.1.Frontend

The front end follows monolithic architecture, which is a big project. Therefore, a handy framework that works extremely well with implementing server side and client side is Nextjs, an open-source framework used by some of the largest companies, based on the React platform. With many powerful and handy features like automatic routing, component, dynamic routing, it makes the front-end implementation a lot easier and faster.

##### 4.1.1. User Interface

The UI is built with Next.js and Bootstrap, each one has its own powerful aspect. Bootstrap is used for fast UI building with its responsive grid system, pre-built UI components. Next.js is used for client side rendering and fetching data from the backend.

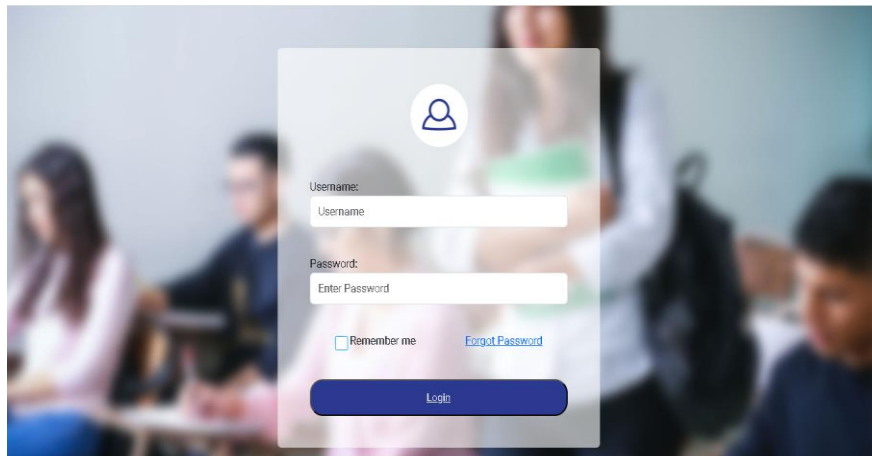


*Figure 4-1Next Js folder structure*

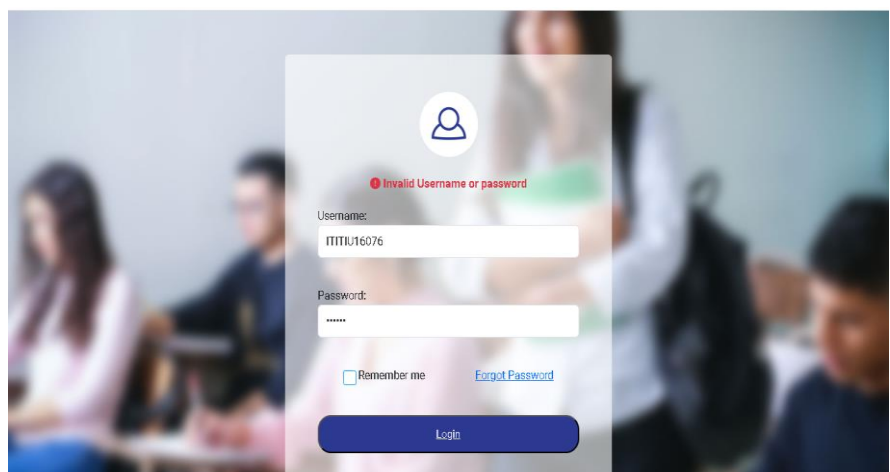
##### 4.1.1.1.Login page

The login page is responsible for handling authentication before going further into the web app. The login page contains a login form that takes username and password and then sends it to the back end for authorization. After the authorizing process is completed. Users will receive access tokens to view protected

resources, the token will be saved in a cookie for future use. The expiration time of the cookie will last for 30 minutes. After the expiration time, if there's no refresh token found, user will be logged out.



*Figure 4-2: Login UI*

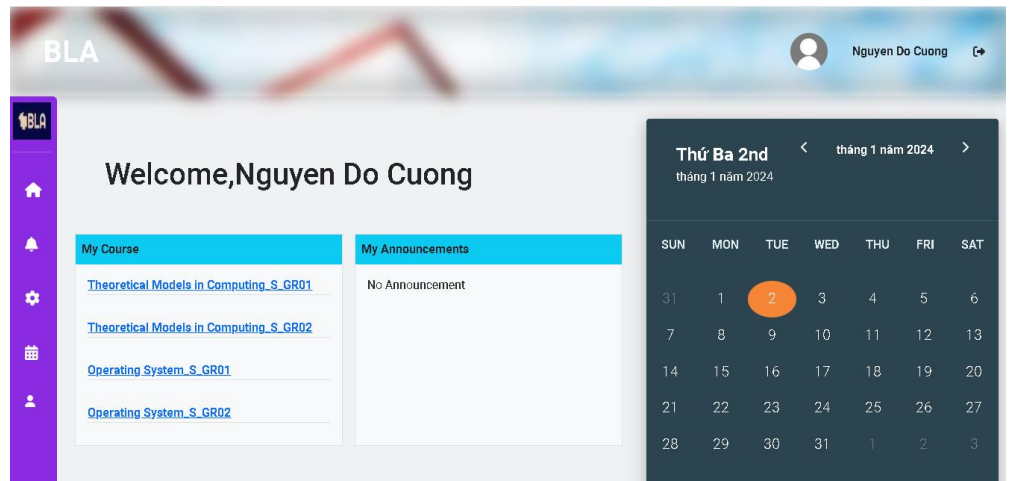


*Figure 4-3:Login error UI*

#### **4.1.1.2.Homepage**

Home page contains dashboard along with all information of the course, announcement of each user





*Figure 4-4:Home page UI*

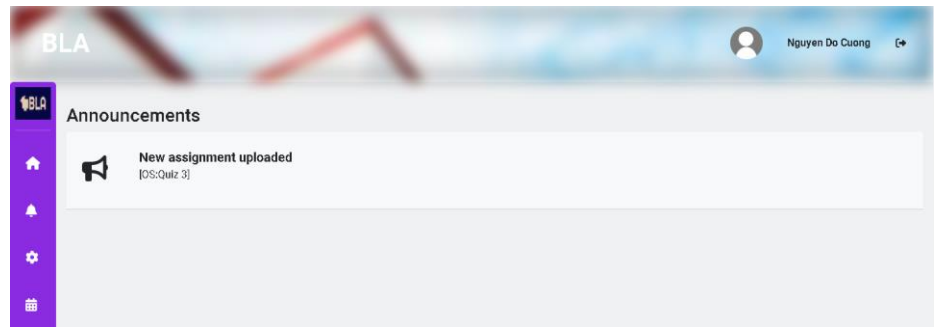
#### 4.1.1.3.Navigation bars

Navigation bar is also a shortcut to navigate to each page. It contains the following elements:

- Logo/Brand: Include logo or brand name at the top-left corner of the navigation bar. Clicking on the logo can redirect users to the home page.
- Create a vertical menu that includes the main sections. Common menu items can include home links to the home page and course links to the course listing page where users can explore and search for courses, announcements links to the announcement page where users can find all important updates and notifications.

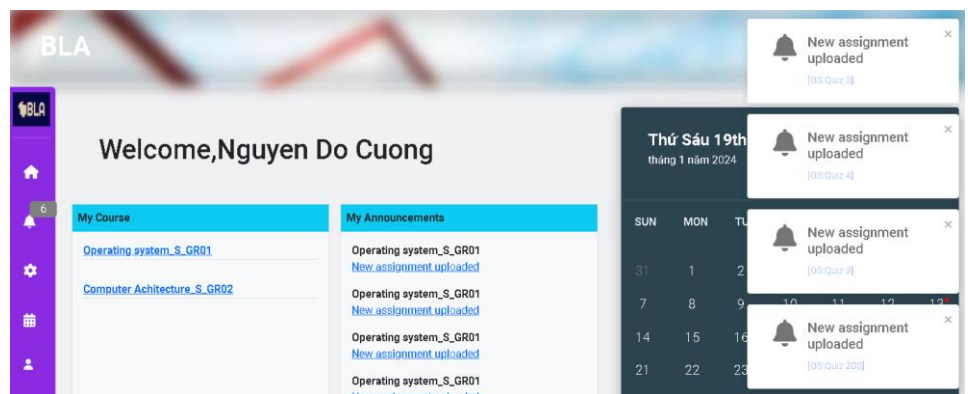
#### 4.1.1.4.Announcement page

Announcement page will display all announcements of the user. Announcements can only be created by instructor or admin of the page. After an announcement has been created, there will be notifications to all student within the courses every time they login to their account. The main section of the page is dedicated to displaying the list of announcements. Each announcement is presented as a separate card or item within the list. The card typically includes the announcement title, date, author, and a summary or preview of the content. Depending on the design, additional information like the number of comments or attachments may be included. The list of announcements may be paginated or scrollable to accommodate a large number of announcements.



*Figure 4-5: Announcement UI*

If the user is a lecturer, the page may provide a separate section or button to create new announcements or edit existing ones. Clicking on the creation/editing feature may open a form or modal where lecturers can enter the announcement title, content, attachments, and other relevant details. The form may include validation to ensure the required fields are filled and appropriate data formats are used. The page should differentiate between student and lecturer roles to provide appropriate functionality and access levels. Students typically have read-only access to announcements, whereas lecturers can view, create, and edit announcements. User roles and permissions should be enforced to ensure that only authorized users can perform specific actions. Furthermore, when new announcement is posted, user will be notified about it and the announcement can be mark as read.



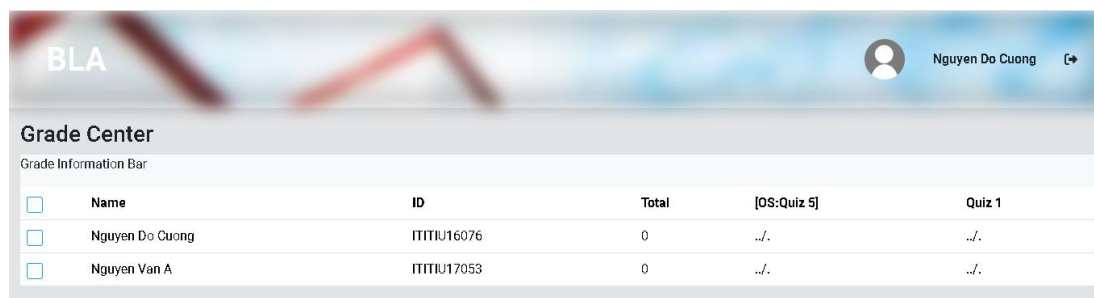
*Figure 4-6: Announcement notify*

#### 4.1.1.5. Grading page

In grading pages, only the Lecturer role can access this page. Lecturers can see a list of available assignments in a specific course along with the submission and its status of students in that course.

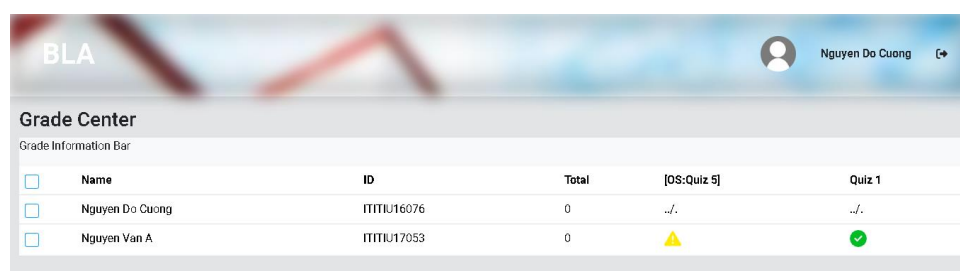
After an assignment is uploaded, it will also be added to the list in grade center page along with a list of student of that course, the submission status also display so that Lecturer can be aware whether student has already done the

assignment, the submission has 2 status: Need grading which tells lecturer that student has already submitted the solution to that assignment and awaiting for grading. Second status is Completed which means lecturer is done with grading.



	Name	ID	Total	[OS:Quiz 5]	Quiz 1
<input type="checkbox"/>	Nguyen Do Cuong	ITTTIU16076	0	...	...
<input type="checkbox"/>	Nguyen Van A	ITTTIU17053	0	...	...

*Figure 4-7: Grading center UI*

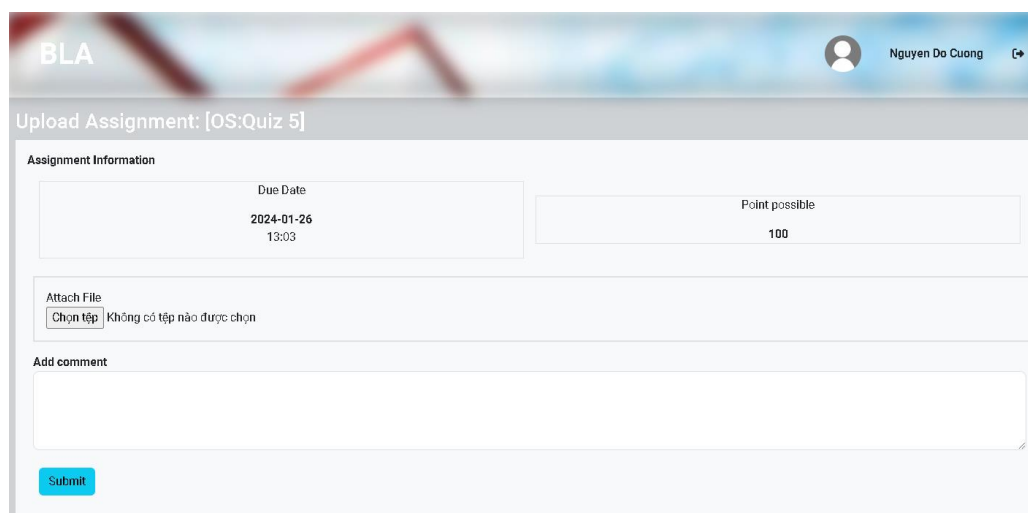


	Name	ID	Total	[OS:Quiz 5]	Quiz 1
<input type="checkbox"/>	Nguyen Do Cuong	ITTTIU16076	0	...	...
<input type="checkbox"/>	Nguyen Van A	ITTTIU17053	0	⚠	✓

*Figure 4-8: Grading status*

#### 4.1.1.6. Assessment and submission page

The assignment and submission page enables learners to submit their solutions for assignments. It can include a form where users can upload their files, enter additional information, and submit the assignment. Instruction is provided, deadline reminders, and feedback on previously submitted assignments. Also, Lecturer is the only one who can view all the submissions of all students and only them can upload new assignments.



**Upload Assignment: [OS:Quiz 5]**

**Assignment Information**

Due Date

2024-01-26 13:03

Point possible

100

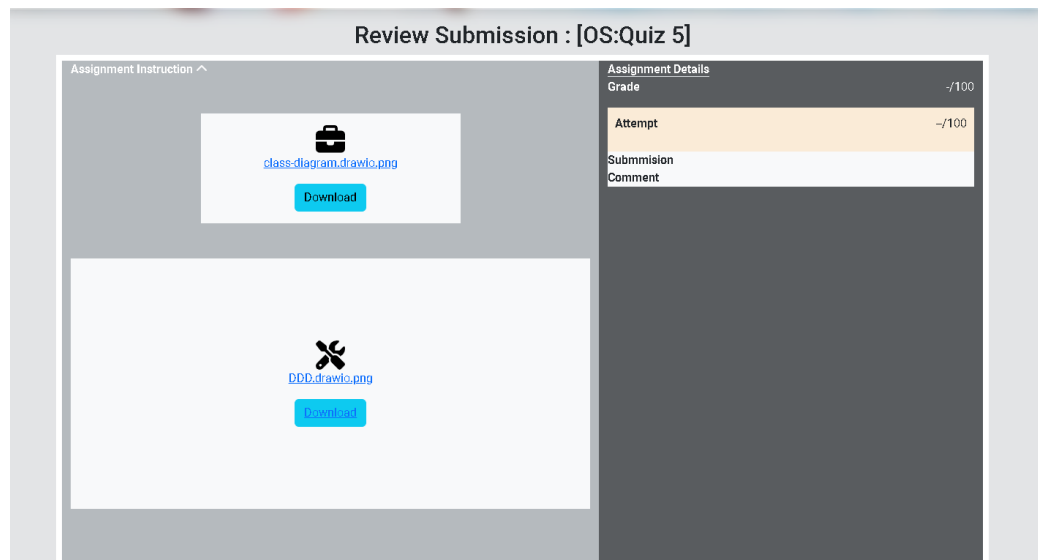
**Attach File**

Chon tệp Không có tệp nào được chọn

**Add comment**

**Submit**

*Figure 4-9: Student submission*

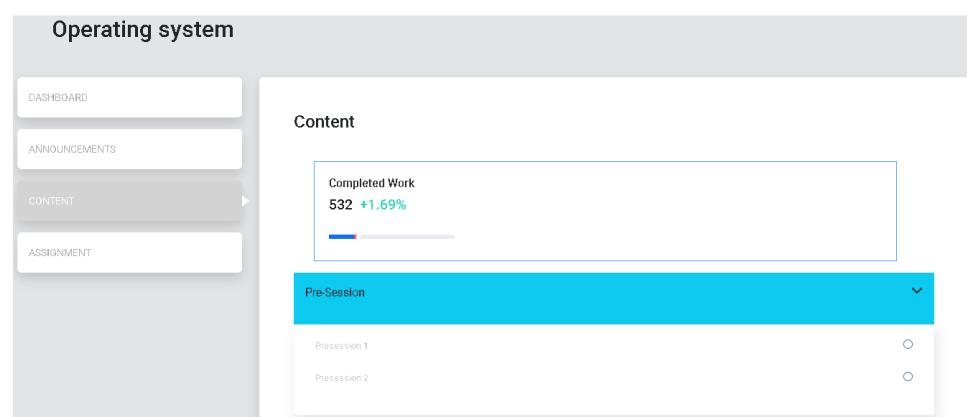


*Figure 4-10:Submission review*

#### 4.1.1.7.Content page

The content viewer page enables users to access and interact with different types of course materials, such as lecture slides, videos, documents, or quizzes. Depending on the content type to embed media players, document viewers, or interactive components. Users can navigate through the content, view progress indicators, and access related resources.

The pre-class session unit focuses on introducing learners to the upcoming topics or concepts before they attend the physical or virtual class session. This unit aims to provide foundational knowledge or pre-learning that prepares students for the in-class activities and discussions.



*Figure 4-11:Presession content UI*

The in-class session unit is dedicated to interactive and collaborative activities that happen during the scheduled class time. This unit focuses on applying and reinforcing the concepts learned in the pre-class session through active engagement and participation.

The post-class session unit involves activities and resources that learners can access after the class session to deepen their understanding and reinforce the learning. This unit encourages students to review and consolidate what they learned, extend their knowledge, and apply it to real-world scenarios.

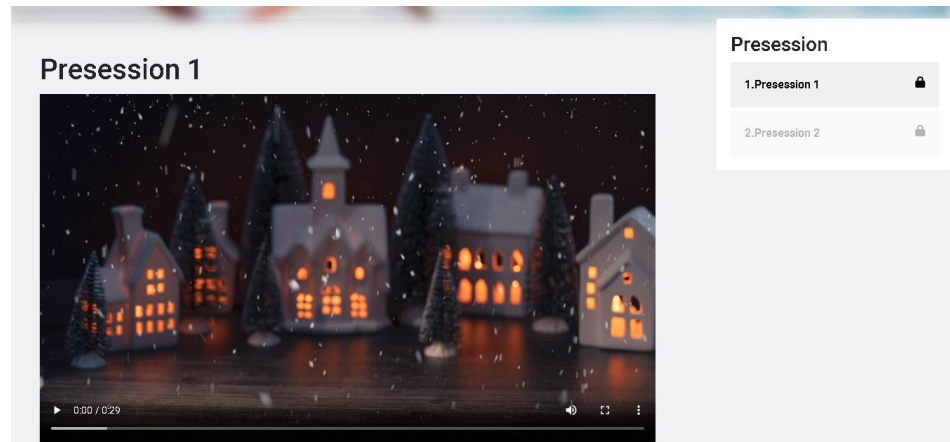


Figure 4-12:Content UI detail

#### 4.1.2. Backend connection

```
const fetchWithToken = async (url, token) => {
  try {
    const response = await fetch(url, {
      headers: {
        Authorization: `Bearer ${token}`,
      },
    });

    if (!response.ok) {
      if (response.status === 401) {
        removeToken();
      }
    }

    const data = await response.json();
    return data;
  } catch (error) {
    //
  }
};

const useFetch = (url) => {
  const token = getToken();
  const { data, error } = useSWR([url, token], ([url, token]) => fetchWithToken(url, token), { revalidateOnMount: true });

  return data;
};

export default useFetch;
```

Figure 4-13:API fetching custom hook

Figure 4-13 shows api fetching by making HTTP requests from the frontend to the backend API. The connection to the backend is established through the fetch API, which is a built-in JavaScript method for making HTTP requests. In the fetchWithToken function, the fetch function is used to send a request to the specified url with the authorization token included in the request headers. If the response from the server is not successful (response.ok is false), the code checks if the response status is 401 (Unauthorized). If it is, it calls a function called removeToken(), which presumably removes the token from the application's state. Inside the useFetch function, the useSWR hook is used to perform data fetching.

The useSWR hook takes three arguments: the url and token as the key for caching purposes and a function that returns the result of fetchWithToken(url, token) to fetch data. The useSWR hook returns an object with data and error properties. The data property contains the fetched data, and the error property contains any error that occurred during fetching.

### 4.1.3. Middleware

Middleware is a concept commonly used in web development frameworks to handle requests and responses in a modular and reusable way. It provides a layer of functionality that sits between the server and the application logic and allows interception and modification requests and responses as they pass through. In Next.js middleware-like functionality is created by using the built-in middleware pattern provided by the API routes feature. API routes allow server-side framework to define serverless functions that can handle requests similar to middleware in traditional server-side frameworks.

```
JS middleware.js > middleware
1  import { NextResponse } from 'next/server';
2  import * as jose from 'jose';
3  const middleware = (request) => {
4    let token = request.cookies.get('jwt');
5    if(!token){
6      const loginUrl = new URL('/login', request.url);
7      return NextResponse.redirect(loginUrl);
8    }
9
10
11  }
12
13  export default middleware
14  export const config = {
15    matcher: [
16      /*
17       * Match all request paths except for the ones starting with:
18       * - api (API routes)
19       * - _next/static (static files)
20       * - _next/image (image optimization files)
21       * - favicon.ico (favicon file)
22       */
23      '/(?!(?!(?!_next/static|_next/image|favicon.ico|login).)*.)',
24    ],
25  }
```

Figure 4-14: Middleware implementation

Middleware is implemented by creating middleware.js in the root of project directory. The purpose above middleware implementation is to check for authentication when logging into the web app, before the request is completed. Inside the middleware function, it retrieves the JWT token from the request's cookies using request.cookies.get('jwt'). If the token is not found or is falsy, it will return a NextResponse.redirect() response, redirecting the user to the loginUrl which is the '/login' path relative to the current request's URL. The config object is

exported, defining the matcher using a regular expression. The regular expression `'/(?!_next/static|_next/image|favicon.ico|login).*)'` matches all request paths except those starting with `'_next/static'`, `'_next/image'`, `'favicon.ico'`, or `'login'`.

#### 4.1.4. Chat implementation

When implementing chat functionality, I need to set up the server side of web socket server using Socket.IO in Node.js with Express and CORS enabled.

```
const io = new Server(server, {
  cors: {
    origin: 'http://localhost:3000', // Allow requests from all origins, you can specify specific origins here
    methods: ["GET", "POST"]
  }
});

io.on('connection', (socket) => {
  console.log('A user connected ' + socket.id);
  socket.on('join_with', id => {
    socket.join(id);
    console.log('Joining ' + id);
  });
  socket.on('send_message', (message) => {
    console.log(message);
    socket.to(message.room).emit('receive_message', message); // Broadcast the message to all connected clients but
  });
  socket.on('disconnect', () => {
    console.log('A user disconnected');
  });
});

server.listen(3001, () => {
  console.log('WebSocket server listening on port 3001');
});
```

Figure 4-15: Websocket server implementation

Figure 4-15 shows the code that creates a WebSocket server that allows clients to connect, join rooms, send messages, and receive messages in real-time. Initialize a new instance of the Socket.IO server, passing the server and a configuration object. In the configuration object, cors is set to allow requests from `http://localhost:3000` and only allow GET and POST methods. When a client connects (`io.on('connection', (socket) => { ... })`), it logs a message and assigns a unique socket ID. The `join_with` event allows a client to join a specific room by calling `socket.join(id)`. The `send_message` event receives a message from a client and broadcasts it to all clients in the same room except the sender using `socket.to(message.room).emit('receive_message', message)`. When a client disconnects (`socket.on('disconnect', () => { ... })`), it logs a message.

```
const ChatRoom = () => {
  const router = useRouter();
  const room = router.query.chatId;
  const socket = io('http://localhost:3001'); // Replace with your server URL
  const chat = useFetch(`https://localhost:7053/api/chats/${router.query.chatId}`)
  useEffect(() => {
    socket.emit('join_with', room)
    console.log(chat)
  }, [])
  if(!chat) return <Loader />
  return (
    <ChatBox socket={socket} />
  )
}

export default ChatRoom
```

*Figure 4-16:Serverconnection implementation*

First, Fetch chat data from the API endpoint `https://localhost:7053/api/Chats/${router.query.chatId}` using a custom hook called `useFetch`. The result is stored in the `chat` variable. If the chat is unavailable in the database, after user sending a new message, it will create a new room for 2 users with the information of the chat. After that, use the `useEffect` hook to perform side effects when the component mounts. Emit a `join_with` event to the server, passing the `room` variable to join the specific chat room: `socket.emit('join_with', room)`. Log the `chat` variable to the console. The empty dependency array (`[]`) ensures that the effect runs only once when the component mounts. If the chat data is not yet available, render a loader component (e.g., `<Loader />`). Finally, Render the `ChatBox` component, passing the `socket` instance as a prop.

## 4.2.Backend

### 4.2.1. Overview

The backend side is implemented by C# programming language with the support of .NET framework. Folder structure of the each microservice will be follows.

- Controller: handle various actions and endpoints required. Use attribute-based routing and HTTP verbs (GET, POST, PUT, DELETE) to define the API routes and actions. Validate incoming data using model validation attributes and handle error responses.
- Service layer: Implement service classes within the `LMS.Services` project to encapsulate the business logic. These services interact with the Data layer to retrieve and manipulate data, perform business operations, and enforce business rules. Services can include user management, course management, assignment management, announcement handling,
- Data Access Layer: Use Entity Framework (EF) within the `LMS.Data` project to interact with the database. Define entity classes that map to database tables and create repositories to perform CRUD (Create, Read, Update, Delete) operations. Use LINQ (Language-Integrated Query) to query and manipulate data from the database.

### 4.2.2. Services

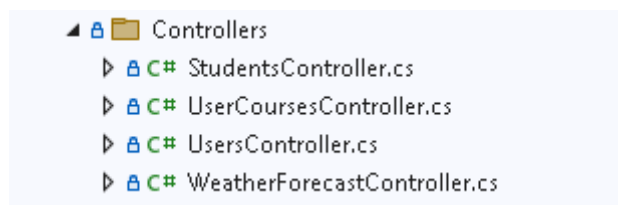
There are a total of 4 main services, those are User service, Announcement service, Assessment service, Course service. User Service responsible for user



authentication, profile management, and access control. Course service handles management, and organization of courses, including course content, syllabus, and related information. Assessment Service: Manages and delivers learning content such as videos, documents, and interactive materials. Grading Service: Deals with the grading and assessment of student assignments, exams, quizzes, and other assessments. Announcement Service: Facilitates the creation, distribution, and management of announcements to inform students and instructors about important updates, deadlines, and course-related information.

#### 4.2.2.1. User service

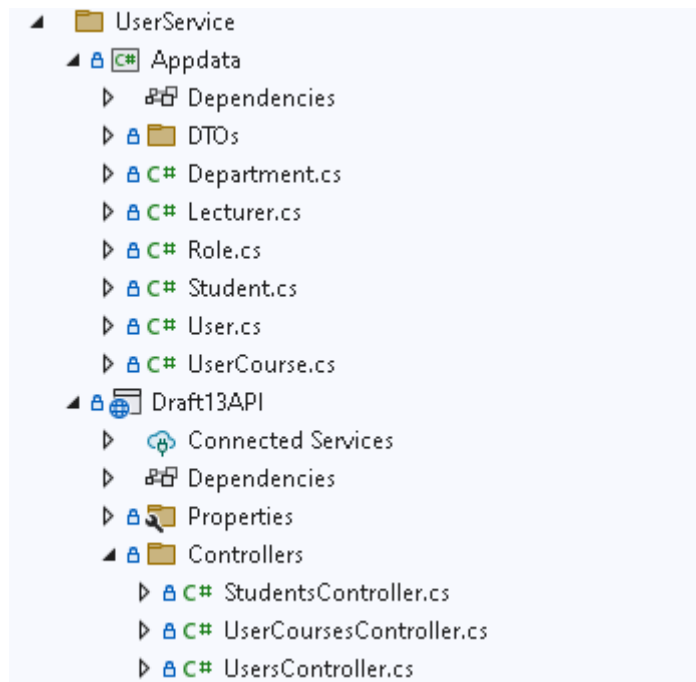
This service is responsible for managing user data such as user information and user account management. The controller of this service handles authentication, authorization of users, process user login, logout, handle access token, refresh token.



*Figure 4.2.2-1: User Controller*

#### 4.2.2.2. User management

There are 2 main user roles, that is student, lecturer, each role has its own access to a group of endpoints. This service also stores user information which can be retrieved by their id

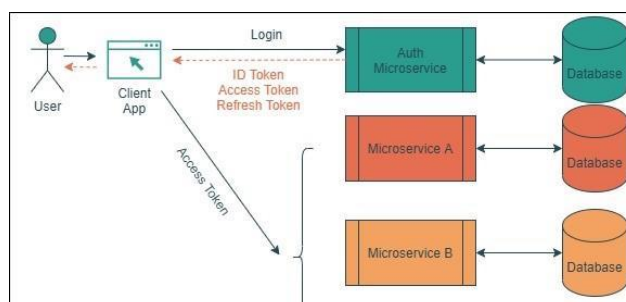


*Figure 4.2.2-2:User service structure*

- We have Department entities responsible for handling user department, Lecturer and student entities store information about lecturers and students.

#### 4.2.2.3.Authentication and authorization

After user logging in the service will send back a token to the client app and the token will be saved in client cookie. So that, when user login to protected resources, client app will send a bearer token along with the request, the backend then will verify the token and return resource for the user, the process is explained in the figure below



*Figure 4.2.2-3:Authentication process*

Here is the step by step configuration of authentication in this project. Firstly, authentication scheme is specified by registering services in Program.cs:

```

builder.Services.AddAuthentication(x =>
{
    x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
    x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
}).AddJwtBearer(o =>
{
    var configuration = new ConfigurationBuilder()
        .SetBasePath(builder.Environment.ContentRootPath)
        .AddJsonFile("appsettings.json", optional: false, reloadOnChange: true)
        .AddEnvironmentVariables()
        .Build();
    var Key = Encoding.UTF8.GetBytes(configuration["JWT:Key"]);
    o.SaveToken = true;
    o.TokenValidationParameters = new TokenValidationParameters
    {
        ValidateIssuer = false,
        ValidateAudience = false,
        ValidateLifetime = true,
        ValidateIssuerSigningKey = true,
        ValidIssuer = configuration["JWT:Issuer"],
        ValidAudience = configuration["JWT:Audience"],
        IssuerSigningKey = new SymmetricSecurityKey(Key)
    };
});

```

*Figure 4.2.2-4: Authentication registration in Program.cs*

After invoking `AddAuthentication`, it will utilize scheme-specific extension methods `AddJwtBearer`. These extensions, in turn, utilize `AuthenticationBuilder.AddScheme` to register the schemes along with their respective settings. In `Program.cs`, the Authentication middleware is incorporated by invoking `UseAuthentication`. This registration enables the middleware to utilize the authentication schemes that have been previously registered. It is essential to call `UseAuthentication` before any middleware that relies on authenticated users. The configuration setting is saved in `appsetting.json` file:

```

{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "Draft13APIContext": "server=localhost;username=postgres;password=cuong123;database=Draft13APIContext-ccf4e91f-87b"
  },
  "JWT": {
    "Key": "a37efe82ce33f42ed935d3782a9d4cbe4f954067f787606239aaf106968512f",
    "Issuer": "https://localhost:44300",
    "Audience": "https://localhost:44300"
  }
}

```

*Figure 4.2.2-5: Appsetting.json configuration*

After setting up `Program.cs` and `appsetting` files, I created a Login method which takes the username and password as the request body. The password is verified by BCrypt hashing algorithm, BCrypt uses a combination of the password, a randomly generated salt, and a work factor (cost factor) to generate a secure hash. The salt is a random value that is different for each user and is stored alongside the hashed password. The work factor determines the computational cost of hashing and can be adjusted to make the hashing process slower over time as hardware becomes faster. When verifying a password, BCrypt extracts the salt

from the stored hashed password and uses it to hash the provided password. It then compares the resulting hash with the stored hash. Since the salt is unique for each user, even if two users have the same password, their stored hashed passwords will be different.

```
[HttpPost("check-user")]
0 references
public ActionResult<User> CheckUser([FromBody]UserDto userDto)
{
    var user = _context.User.Where(u => u.username == userDto.username).FirstOrDefault();
    if (user != null && BCrypt.Net.BCrypt.Verify(userDto.password, user.password))
    {
        string token = CreateToken(user);
        return Ok(token);
    }

    return BadRequest("Wrong username or password");
}
```

*Figure 4.2.2-6: Login method implementation*

User is retrieved from the `_context` object based on the provided username value in the `userDto` object. It uses LINQ to query the database and filter the users based on the matching username. The `FirstOrDefault()` method returns the first user that matches the query or null if no matching user is found. `BCrypt.Net.BCrypt.Verify()` method to compare the provided `userDto.password` with the stored hashed password in the `user.password` field. If the password verification is successful, the condition evaluates to true.

```
private string CreateToken(User user)
{
    var role = _context.Role.Where(r=>r.role_id==user.role_id).FirstOrDefault();
    List<Claim> claims = new List<Claim> {
        new Claim(ClaimTypes.Role,role.role_name),
        new Claim("user_id",user.user_id.ToString()),
        new Claim("role_name",role.role_name)
    };
    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(configuration.GetSection("JWT:Key").Value!));
    var credential = new SigningCredentials(key, SecurityAlgorithms.HmacSha256Signature);
    var token = new JwtSecurityToken(
        claims: claims,
        expires: DateTime.Now.AddMinutes(10),
        signingCredentials: credential
    );
    var jwt = new JwtSecurityTokenHandler().WriteToken(token);

    return jwt;
}
```

*Figure 4.2.2-7: Create token method*

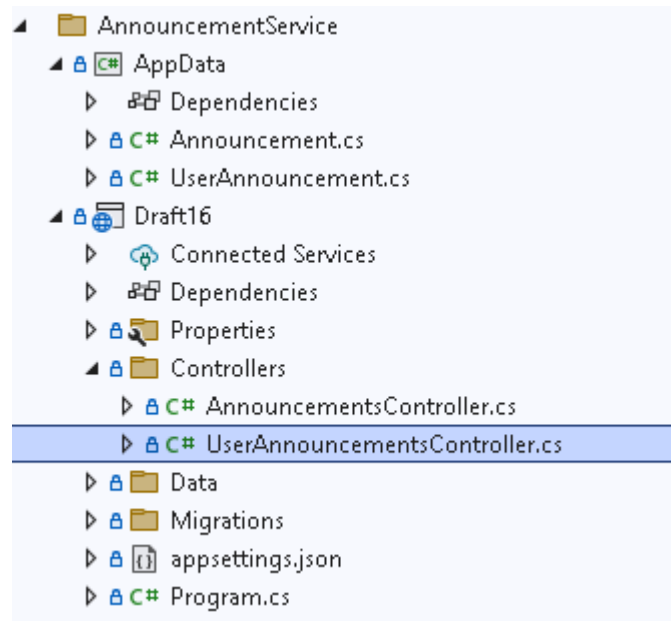
After verifying the username and password process completed, a new list of Claim objects. Claims represent the statements about the user that are encoded within the JWT. In this case, three claims are added to the list: A claim representing the user's role, stored as `ClaimTypes.Role`. A claim representing the user's `user_id`, stored as a custom claim named "user\_id". A claim representing

the user's role name, stored as a custom claim named "role\_name". The CreateToken method retrieves the user's role, generates a list of claims, creates a JWT token with the specified claims, expiration time, and signing credentials, and returns the generated token as a string. New instance of JwtSecurityToken with the specified parameters: The list of claims associated with the token. expires which is the expiration date and time for the token. In this case, it is set to 30 minutes from the current time and signingCredentials is used to sign the token.

#### **4.2.2.4. Announcement service**

An announcement service microservice is a component within a microservices architecture that focuses on managing and distributing announcements or notifications to users within a system or application. Only the Lecturer or admin has the right to create new announcements, new announcements will be notified to all students within the course. Also, when new assignment or content is uploaded, new assignment also created and notified to all students.

Instructors can use the announcement service to create and compose announcements related to course updates, important deadlines, or general information. They can specify the title, content, formatting options, and any relevant attachments. The announcement service allows for course-specific announcements, ensuring that important information is shared with the learners enrolled in a particular course. Instructors can create announcements that are visible only to the learners of a specific course. The service also enables the creation of system-wide announcements that reach all learners and instructors across the LMS. System-wide announcements can be used for general updates, platform maintenance notifications, or policy changes. Delivery and Notification: The announcement service handles the delivery and notification of announcements to the intended recipients. Learners receive notifications through various channels, such as email, in-app notifications, or mobile push notifications, depending on their preferences and the LMS's capabilities.



*Figure 4.2.2-8:Announcement service folder structure*

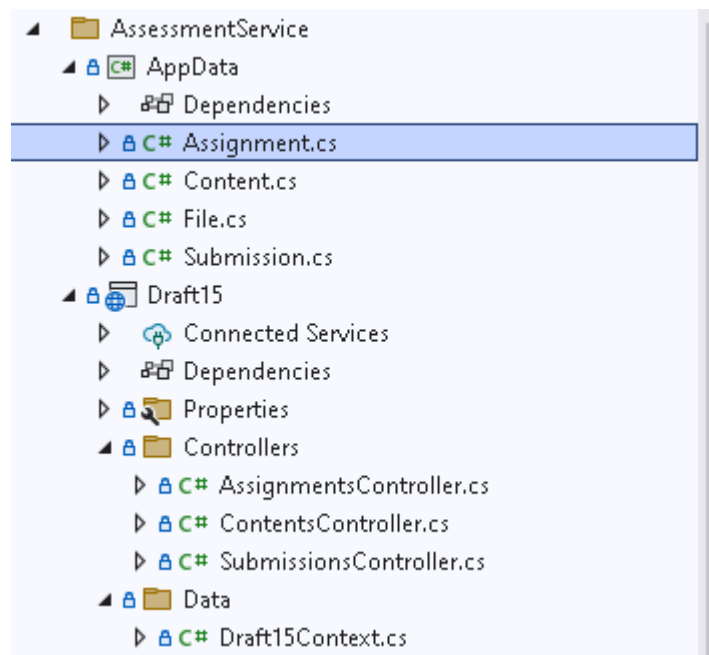
#### 4.2.2.5.Assessment service

Assessment service handling quizzes, assignments management and content of the course which is slides or pre-recorded videos. It serves as an independent and scalable service that focuses on managing assessments, evaluations, and grading processes within a system or application.

Some of the features of this service are:

- Assignments, quizzes, content creation: Lecturer is allowed to upload new assessments for the courses. This includes defining questions, specifying answer options, setting time limits, and configuring grading criteria. The assessment was then delivered to students through a web interface. It ensures that the assessments are presented correctly and securely to the intended audience.
- Submissions: After completing required assessment within the due date, students can submit their completed assessments through the microservice, which handles the storage and management of submitted answers or responses. It ensures that submissions are accurately recorded and associated with the appropriate assessment.
- Grading: The service incorporates the necessary logic to evaluate and grade assessments based on predefined criteria. It automates scoring processes, calculates grades, and provides feedback to users regarding their performance.

- Content manages the storage and retrieval of course content, such as lecture materials, videos, and documents. It provides endpoints for uploading, downloading, and managing content files associated with courses.
- File is responsible for managing file storage and retrieval within the LMS. It handles file uploads, downloads, and storage management. Other services, such as Content Service or Assignment Service, may rely on the File Service for storing and retrieving files associated with their functionalities.



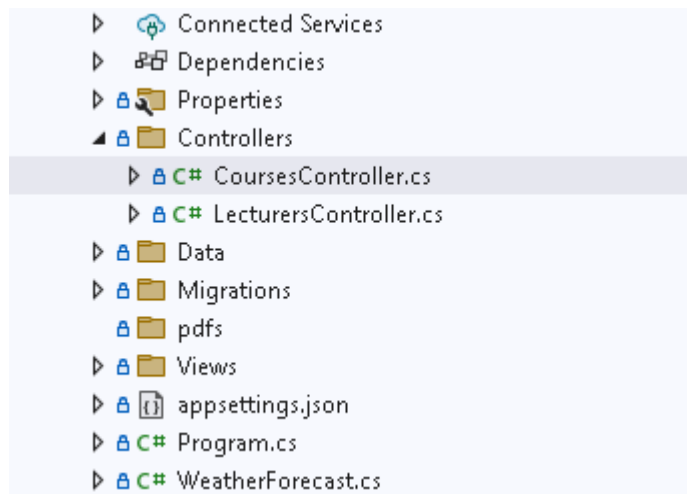
*Figure 4.2.2-9:Assessment service folder structure*

Students can access the assessment such as slides, pre-record lecture's videos, assignments via a specific course. After choosing the course they want, a list of lecture will be display week by week according to Lecturer's preference along with available assignments,quiz, etc.. Especially the lecture sildes,videos are broke down into smaller units in each week. These smaller units can be more easily understood, processed, and retained by students. Contentst in a course are splitted in to 3 small unis: pre-session,post-session,in class session.

#### **4.2.2.6.Course service**

This service is responsible for handling courses that students and lecturer participate in a semester, course service manage information of the courses and handle which user belong to which course by user id. It handles operations such

as creating, retrieving, updating, and deleting courses. It also provides endpoints for course enrollment, unenrollment, and retrieving course information.



*Figure 4.2.2-10: Course service folder structure*

#### **4.2.2.7. Communication service**

The Communication Service microservice in an LMS plays a crucial role in facilitating communication and collaboration between learners, instructors. The Communication Service provides a messaging feature that allows learners, instructors, and administrators to send direct messages to each other. Learners can communicate with their peers, ask questions to instructors, or seek assistance from administrators. Instructors can provide feedback, answer queries, and communicate important updates.



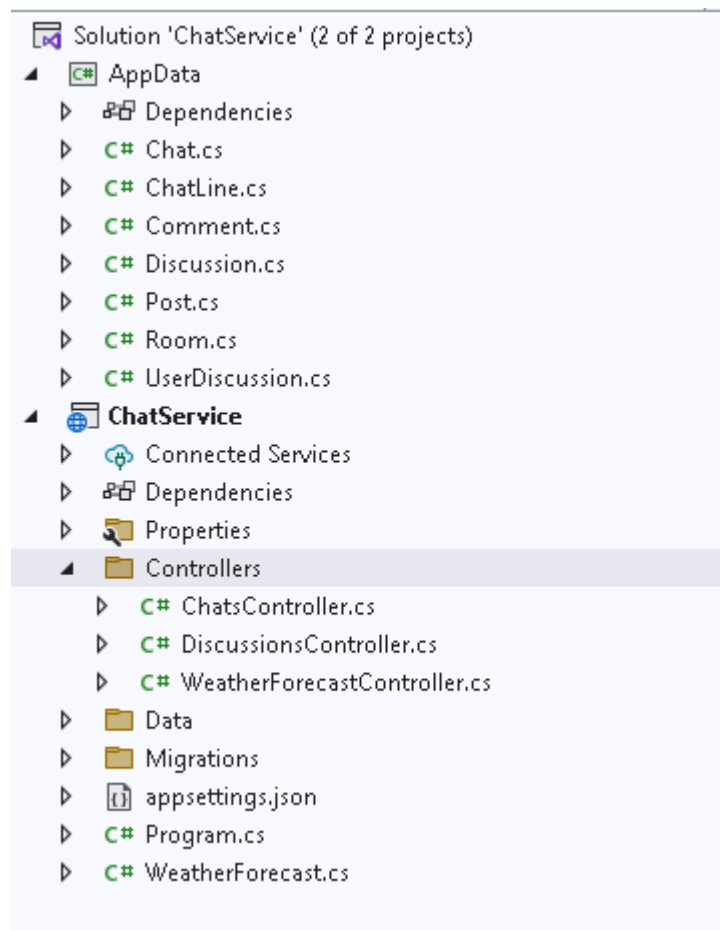


Figure 4.2.2-11: Communication service folder structure

This service includes discussion forums where learners can engage in asynchronous discussions on course-related topics. Learners can post questions, share ideas, and participate in discussions with their peers. Instructors can moderate the forums, provide guidance, and encourage active participation.

### 4.3.Database

The database used in this web app is Postgresql. There are a few different ways to keep a service's persistent data private.

- Private-tables-per-service – each service owns a set of tables that must only be accessed by that service.
- Schema-per-service – each service has a database schema that's private to that service.
- Database-server-per-service – each service has its own database server.

This project uses PostgreSQL, often referred to as Postgres, is a powerful open-source relational database management system (RDBMS) known for its robustness, scalability, and extensibility. In the context of microservice Database-server-per-service will be implemented to ensure data hidden and loosely coupling

property of the architecture. Each microservice will have its own database table(s) to manage its specific domain. The tables can be designed based on the specific data requirements and business logic of each microservice.

#### 4.3.1. User service database

There are 6 tables in this database schema user table which contains account information of a single user, student and lecturer and their department can be identified through role table and department table using foreign key (FK). The UserCourse table is used to reference which course is attempted by which student by creating a copy record of a course id in course service and user\_id FK to reference the user table, when requested to a specific course. Instead of duplicating the entire record, I reference identifier to establish a link between entities in different services. The reference is a unique identifier (such as an ID or key) that identifies and retrieves the related information when needed.

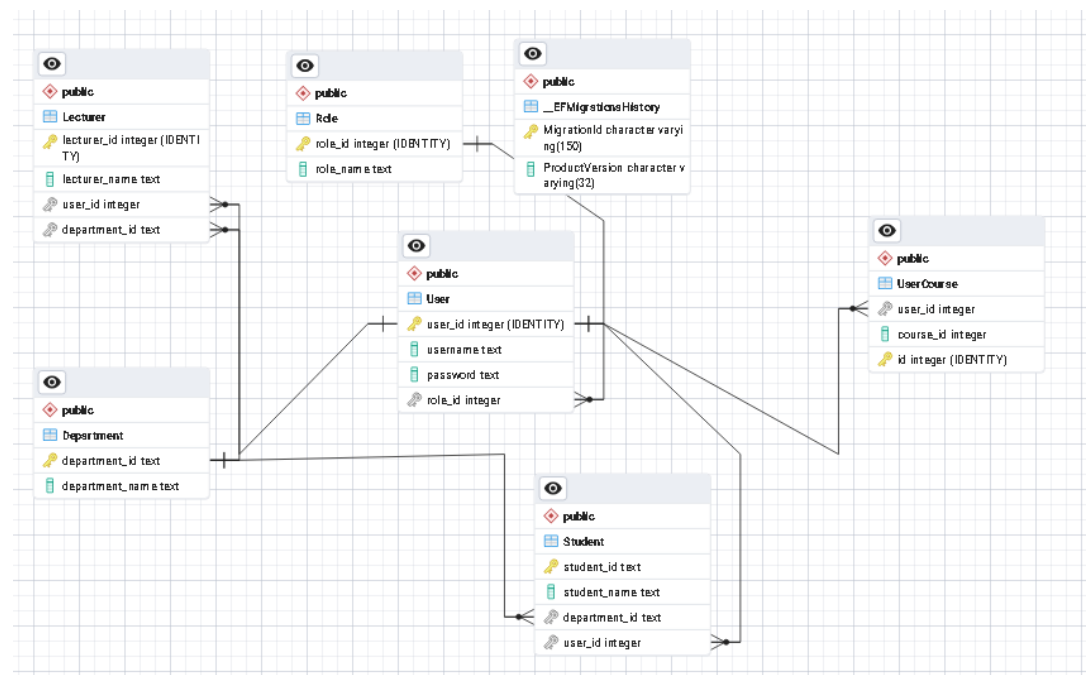
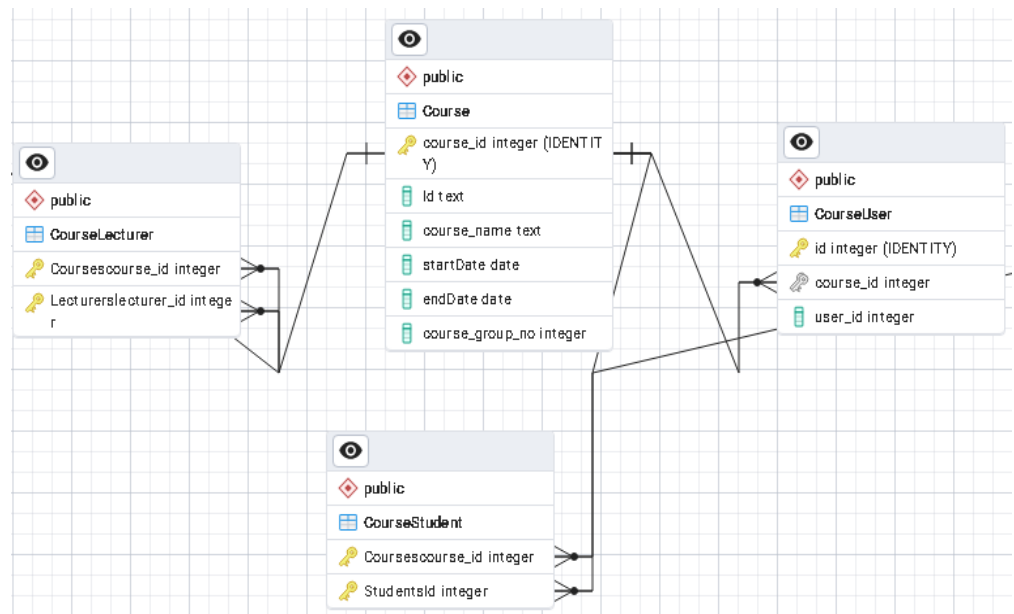


Figure 4.3-1: User service database schema

#### 4.3.2. Course service database

As the name of the service states, the course service contains information related to the course such as course table has course\_id (Primary Key): Unique identifier for each course, course\_name of the course. d start\_date: Start date of the course. end\_date: End date of the course. and course\_group\_no defines the

group number of a course. CourseUser is used to define which user belongs to which course since the user is in another service, by introducing user\_id which is the copy identifier. This column would establish a link between the user and the course they belong to. When user information retrieval needed, it would make a request to the User Service using the user\_id to fetch the relevant user details.



*Figure 4.3-2: Course service database schema*

### 4.3.3. Assessment service database

The database stores information about various assessments, such as tests, quizzes, assignments, and exams. This includes details such as the assessment title, description, duration, scoring criteria, and associated course or module. Instructors can provide grades, feedback, and comments on student assessments. This information is typically stored in the database, associating it with the respective student and assessment.

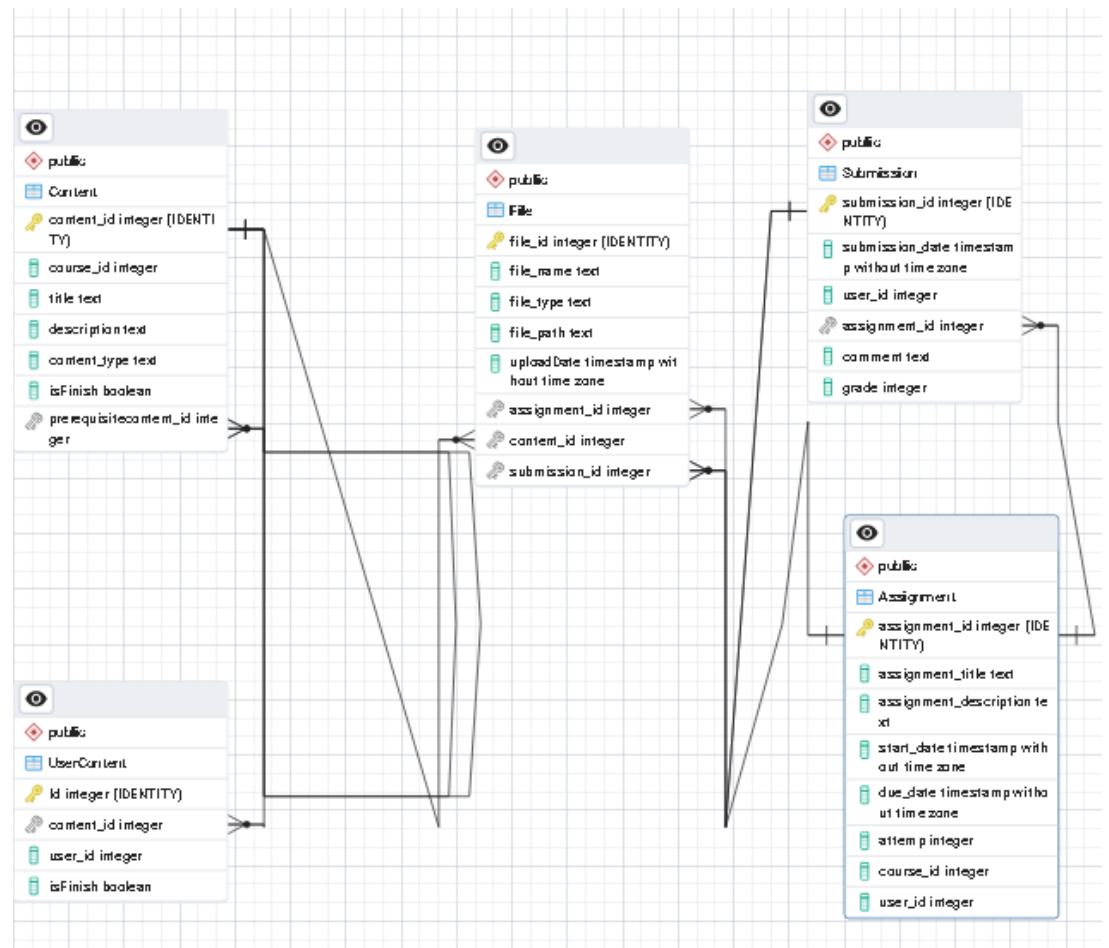


Figure 4.3-3:Assessment service database schema

#### 4.3.4. Announcement service database

The announcement table stores information about each announcement, including the title, content, creation timestamp. The UserAnnouncement table stores information about users who can create announcements which can be retrieve through user service by appropriate user\_id in the table below.

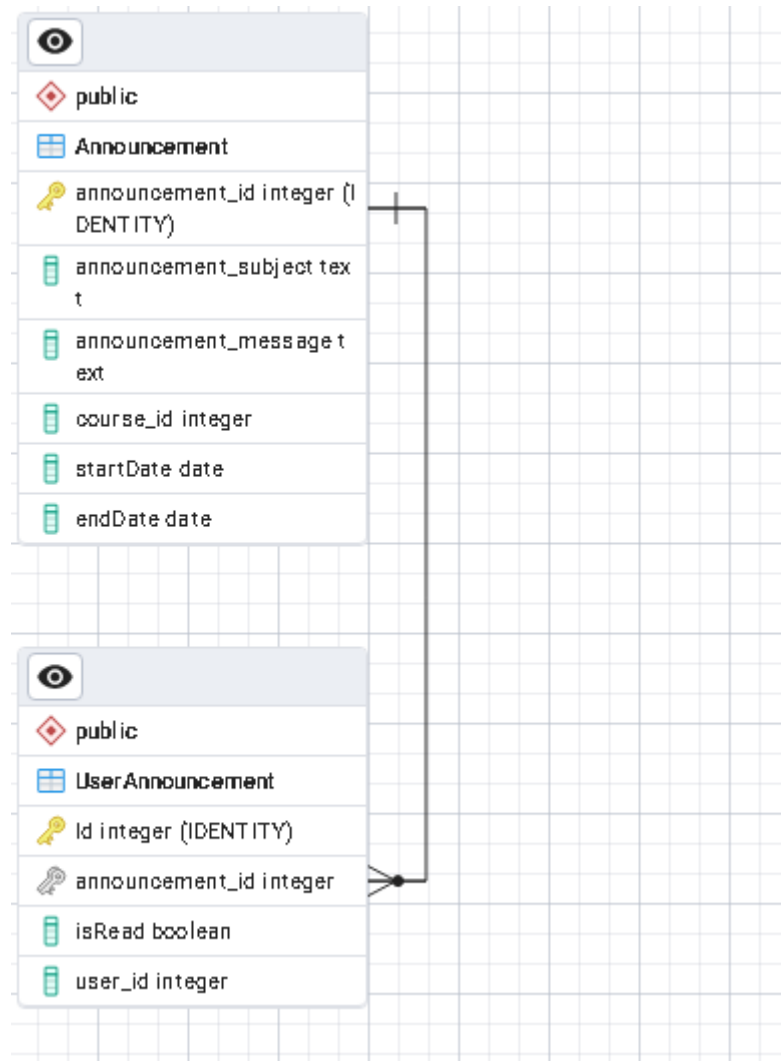


Figure 4.3-4: Announcement service database schema

#### 4.3.5. Communication service database

In this schema, the Discussion table stores information about each discussion forum, including its name and description. The Post table contains details about each discussion thread within a forum, such as the thread title, content, creator, and creation timestamp. The Post table stores individual posts within each thread, including the post content, creator, and creation timestamp.

The Chat table stores information about each conversation, including the creation timestamp and the timestamp of the last message in the conversation. The Participant table establishes a many-to-many relationship between users and conversations, allowing multiple users to participate in a conversation. The ChatLine table contains details about individual messages within a conversation, including the content, sender, and the timestamp when the message was sent.

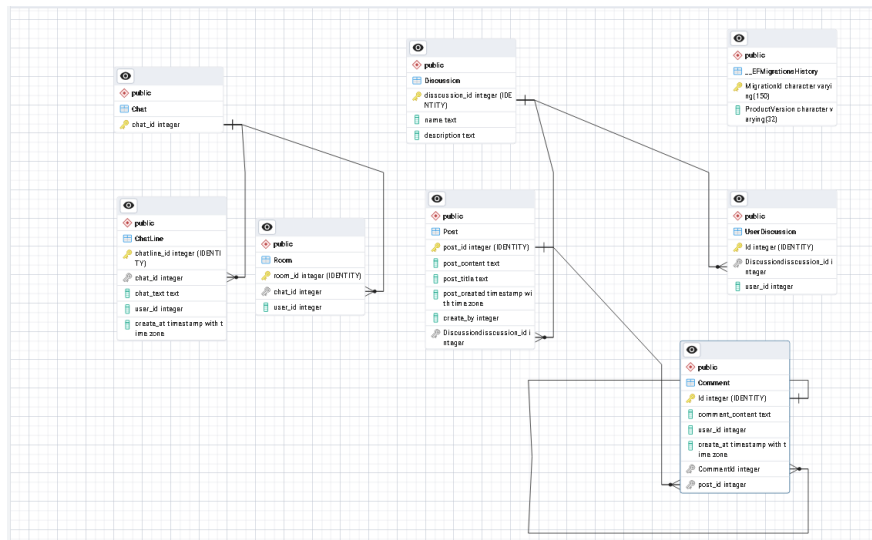


Figure 4.3-5: Communication service database schema

## 4.4. API Gateway

Ocelot is an open-source API gateway that can be used to manage the communication between clients and microservices in this project. Ocelot acts as a centralized entry point for client requests in the LMS. It exposes a single API endpoint that clients interact with, abstracting away the individual microservices. This simplifies the client-side integration and provides a unified interface for accessing functionalities

### 4.4.1. Implementation

First, i create a separate project for ocelot api gate way,inside the project, for the gateway be able to run properly, an nuget package must be installed, it is called Ocelot.

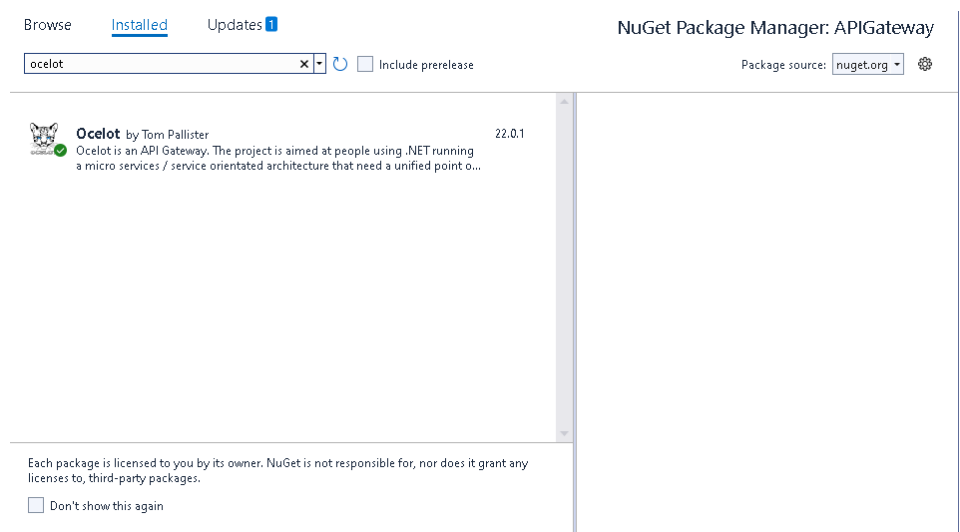


Figure 4.4-1: Ocelot package installment

Next step is to create and config configuration file ocelot.json to define the routing and configuration. This file will hold the Ocelot configuration. Configure

the routes, which specify how incoming requests should be routed to the respective microservices. Each route should include the upstream path, downstream path, and other relevant settings.

```
{
  "Routes": [
    {
      "DownstreamPathTemplate": "/api/Users/{everything}",
      "DownstreamScheme": "https",
      "DownstreamHostAndPorts": [
        {
          "Host": "localhost",
          "Port": 7277
        }
      ],
      "UpstreamPathTemplate": "/users/{everything}",
      "UpstreamHttpMethod": [ "POST", "GET" ]
    },
    {
      "DownstreamPathTemplate": "/api/Courses/{everything}",
      "DownstreamScheme": "https",
      "DownstreamHostAndPorts": [
        {
          "Host": "localhost",
          "Port": 7051
        }
      ],
      "UpstreamPathTemplate": "/course/{everything}",
      "UpstreamHttpMethod": [ "GET" ]
    },
    {
      "DownstreamPathTemplate": "/api/Announcements/{everything}",
```

*Figure 4.4-2: Ocelot.json configuration*

The Routes section is an array that holds individual route configurations. Each route configuration defines how a specific request should be handled and forwarded to the appropriate microservice. Multiple routes can be defined to handle different types of requests or direct them to different microservices. Each route configuration includes DownstreamPathTemplate, DownStreamHostAndPorts, DownStreamScheme, UpStreamPathTemplate. The downstream path template specifies the path that the request should be forwarded to in the downstream microservice. It can include placeholders (e.g., {everything}) to capture dynamic segments of the path. The scheme (e.g., http, https) of the downstream microservice. An array of objects containing the host and port details of the downstream microservice. Multiple hosts and ports can be specified for load balancing or failover scenarios. UpstreamPathTemplate: The upstream path template defines the path that clients will use to send requests to the API gateway. It can include placeholders (e.g., {everything}) to capture dynamic segments of the path. The route configuration can include additional options, such as: UpstreamHttpMethod: Specifies the HTTP method (e.g., GET, POST, PUT, etc.) for the upstream request. This allows user to define specific methods for each route.

```

using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.IdentityModel.Tokens;
using Ocelot.DependencyInjection;
using Ocelot.Middleware;
using System.Text;

var builder = WebApplication.CreateBuilder(args);
builder.Configuration.AddJsonFile($"*{Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT")}.json",
builder.Services.AddOcelot(builder.Configuration);

builder.Services.AddCors(options =>
{
    options.AddDefaultPolicy(builder =>
    {
        builder.WithOrigins("http://localhost:3000")
            .AllowAnyHeader()
            .AllowAnyMethod();
    });
});
var app = builder.Build();
app.UseAuthentication();
//app.MapGet("/", () => "Hello World!");
app.MapControllers();
await app.UseOcelot();
app.Run();

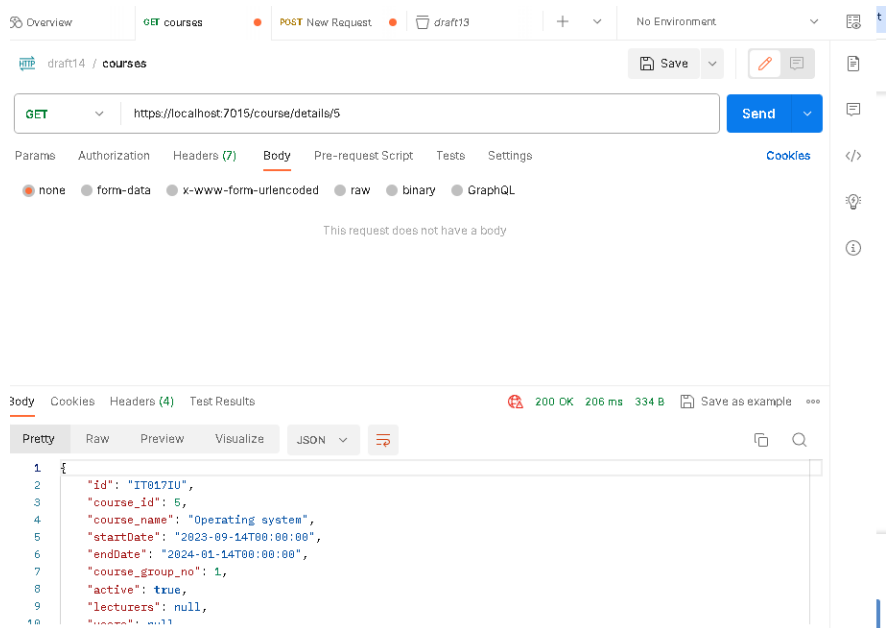
```

*Figure 4.4-3: Program.cs configuration*

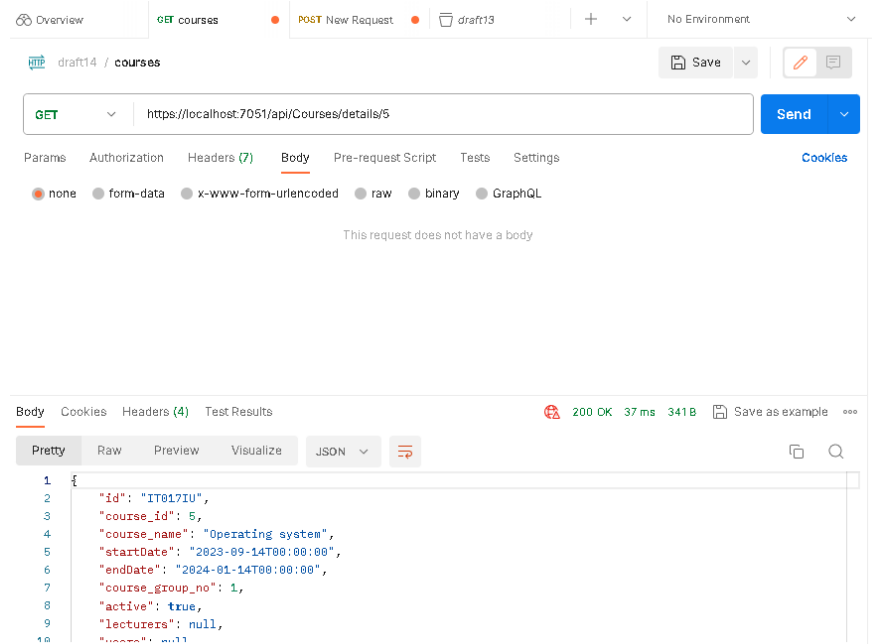
Figure 4.4-3 shows the setup Ocelot in the project. `builder.Configuration.AddJsonFile` adds an Ocelot-specific JSON configuration file based on the current environment. It loads the configuration file named `ocelot.{Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT")}.json`. The configuration file should contain the routing and configuration details for Ocelot. `builder.Services.AddOcelot(builder.Configuration)` adds Ocelot services to the dependency injection container. It registers the necessary Ocelot services to enable routing and other Ocelot features. `builder.Services.AddCors` adds Cross-Origin Resource Sharing (CORS) configuration to allow requests from client (`http://localhost:3000` in this case). CORS allows web applications to make requests to a different domain or port. `app.UseAuthentication()` adds authentication middleware to the application pipeline. It enables authentication features, such as validating access tokens or performing user authentication. `app.MapControllers()` maps controllers to the application pipeline. It ensures that the controllers defined in the application are available to handle incoming HTTP requests. `await app.UseOcelot()` registers and uses the Ocelot middleware. It captures incoming requests and routes them based on the Ocelot configuration, forwarding them to the appropriate microservices

#### 4.4.2. Result





*Figure 4.4-4: Test result with api gateway*



*Figure 4.4-5: Test result with service api*

As a result from Figure 4.4-4 and Figure 4.4-5, the return JSON response from ocelot api gateway match with the api directly called from the course service which indicates that the API gateway is successfully forwarding and returning the responses from the underlying microservices. This ensures that the clients interact with the microservices through a unified entry point, the API gateway, without being aware of the underlying service architecture.



## Chapter 5

### DISCUSSION AND EVALUATION

#### 5.1. Discussion

Blended learning, which combines traditional face-to-face instruction with online learning components, has gained significant popularity in educational settings. A blended learning web application microservice can serve as a valuable tool to support and enhance the blended learning experience. One of the key advantages of this web application microservice is the increased accessibility and convenience it offers to learners. By providing a web-based platform, learners can access learning materials, complete assignments, and participate in discussions at their own pace and convenience. This flexibility allows learners to engage with the content anytime and from anywhere, making learning more accessible to a diverse range of individuals. Additionally, a blended learning web application microservice enables personalized and individualized learning experiences. Learners can benefit from a more customized learning path, addressing their specific needs and learning preferences. This personalization fosters a learner-centric approach, promoting engagement and motivation. Collaboration and interaction are crucial components of blended learning, and a web application microservice can facilitate these aspects.

#### 5.2. Comparison

		My web app	BlackBoard	
Instructor	Preview course items, assignments, tests	Yes	Yes	
	Create and upload course content	Yes	Yes	
	Participate in discussions	Yes	Yes	
	Send announcement	Yes	Yes	
	Grade assignment and update assesment settings	Yes	Yes	
	Change item settings for assessments and other content	Yes	Yes	
	Send Email	No	Yes	
	Real-time chat	Yes	No	
Student	Access both Original and Ultra courses	No	Yes	
	Take assignments and tests	No	Yes	
	Participate in discussions	Yes	Yes	
	Send Email	No	Yes	

	Real-time Chat	Yes	No	
	View course items and course announcements	Yes	Yes	
Mobile app		No	Yes	

*Table 0-1: Comparison with available system*

Table 8 provides a comparison between the BlackBoard web app and my blended learning web app in terms of user experience from instructor and student perspectives. While there are a few similarities between my web app and BlackBoard there are some features that Blackboard is more capable of archiving such as accessing Original and Ultra courses, taking assignments, sending emails and tests, or having a separate mobile app. Apart from that, my web app has the ability to make a real-time chat with the constructor, and the structure of the content material is different from Blackboard.

### **5.3. Evaluation**

In general, the project has successfully implemented the basic components of the typical LMS. Although there is some absence of functionality such as taking online tests directly from the web app due to security risks such as technical issues or security vulnerabilities which can affect the learning experiment for both synchronous and asynchronous learning, so important exams can be taken directly class or in form of assignment submission alternatively. Also, the chat functionality can be a replacement solution for sending email in BlackBoard, the purpose of this is to reduce the complexity of sending email for an occasion like a simple and straightforward question that requires a brief response, informal conversations, urgent matters, or time-sensitive issues. Chat can be a convenient alternative, but there are cases where email is still preferred. Lengthy or detailed discussions, formal communication, and situations that require a documented record are better suited for email but considering using an available application like Gmail, Outlook,.. is more reliable scaleable, and user-familiar.

## Chapter 6

### CONCLUSION

#### 6.1 Conclusion

The main purpose of this Blended Learning web app has successfully achieved its goal of providing a robust platform for educators and learners to access and interact with pre-existing learning materials. The project leverages a range of features to facilitate content delivery, learner engagement, and personalized learning experiences. Also, the project collaborating with instructors who have already created these materials guarantees the availability of high-quality resources to enhance the learning experience by chunking method.

Moreover, the project emphasizes learner engagement through interactive tools and communication features. Integrated communication channels enable learners to actively participate in discussions, collaborate with peers, and seek clarification from instructors. These interactive elements foster a sense of community, facilitate knowledge sharing, and create a dynamic and personalized learning experience.

#### 6.2 Future work

##### a) Adaptive Learning

Incorporate adaptive learning capabilities which use artificial intelligence and machine learning algorithms to personalize the learning experience for each individual learner. The system can analyze learner performance, identify strengths and weaknesses, and dynamically adjust the content and activities to meet each learner's specific needs. The LMS can assess learner's prior knowledge and skills through pre-tests or diagnostic assessments. Based on the results, the system can dynamically generate personalized learning paths that address learner's specific learning gaps and challenges. Learners can progress at their own pace and focus on areas where they need the most support.

##### b) Gamification

Gamification is a technique that introduces game-like elements into the learning process to increase learner engagement and motivation. Gamification can include rewards, such as unlocking additional learning content, bonus materials, or exclusive resources, as learners achieve specific goals or milestones. This creates a sense of anticipation and provides incentives for learners to actively engage with the learning materials.

##### c) Mobile learning

Mobile learning allows learners to access course materials, engage in activities, and communicate with instructors and peers using their smartphones or tablets. The web app

should have a responsive user interface that automatically adjusts to different screen sizes and resolutions or develop a dedicated mobile app, available on iOS and Android platforms. The app should provide a user-friendly interface, push notifications for important updates, and allow learners to easily access their courses, engage in discussions, submit assignments, and track their progress on the go. This ensures that learners can access and navigate the platform seamlessly on various mobile devices. Provide offline access to selected learning materials. Learners can download content to their mobile devices and continue learning even when they don't have an internet connection. Once they regain connectivity, their progress and any completed activities can be synced.

## REFERENCES

1. Gururaj Bhadri, L. P. (2022). Blended Learning: An effective approach for Online Teaching and Learning. *Journal of Engineering Education Transformations* .
2. Isak Shabani, E. M. (2021). Design of Modern Distributed Systems based on. *International Journal of Advanced Computer Science and Applications*,, 153-159.
3. Kholis, A. (2022). The Implementation of Blended Synchronous and Asynchronous Online Language Learning during the Covid-19 Pandemic. *Lingua Didaktika*, 83-93.
4. Kilag, O. K. (2023). Optimizing Education: Building Blended Learning Curricula with LMS. *International Multi-disciplinary Journal of Education*, 238-250.
5. Rika Riwayatiningsih, S. (2020). THE IMPLEMENTATION OF SYNCHRONOUS AND ASYNCHRONOUS E- LANGUAGE LEARNING IN EFL SETTING: A CASE STUDY. *Jurnal Basis*, 309-318.
6. Sari, M. I. (2021). Blending Asynchronous and Synchronous Online Learning to Teach Pre-Service Teachers to Utilize Technology in English Language Teaching And Learning. *Best Practices and Research in ELT Classrooms*, 159.
7. Steinegger, R. H. (2017). Overview of a domain-driven design approach to build microservice-based applications. *The Third Int. Conf. on Advances and Trends in Software Engineering*.
8. Wintemute, D. (2023, 8 22). *Synchronous vs. Asynchronous Classes: What's the Difference?* Retrieved from The Best School: <https://thebestschools.org/resources/synchronous-vs-asynchronous-programs-courses/>
9. Yagci, M. (2017, 2). *Journal of Education and Training Studies*. Retrieved from <https://redfame.com/journal/index.php/jets/article/view/2118>
10. Amity, and Flora. "Synchronous and asynchronous E-learning." *European Journal of Open Education and E-Learning Studies* 5.2, 2020.
11. Cahyani, and Ni Made Wahyu Suganti. "Blended online learning: Combining the strengths of Synchronous and Asynchronous Online learning in EFL context." *urnal Pendidikan Teknologi Dan Kejuruan*, 18 2 2021, pp. 174-184.
12. Clark, et al. "Preparation and synchronous participation improve student performance in a blended learning experience." *Australasian Journal of Educational Technology* 37.3, 2021, pp. 187-199.

13. Amati, and Flora. "Synchronous and asynchronous E-learning." *European Journal of Open Education and E-Learning Studies* 5.2, 2020.
14. "JSON Web Token Introduction - jwt.io." *JWT*, <https://jwt.io/introduction>. Accessed 16 January 2024.
15. Newman, Sam. *Building Microservices: Designing Fine-grained Systems*. O'Reilly, 2021.