UNSW
AUSTRALIA

School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

A Meta Learning Management System

Thesis submitted as a requirement for the degree of

Bachelor of Engineering (Honours)

Allen Wu	z5205003	Software Engineering
Daniel Ferraro	z5204902	Software Engineering
David Nguyen	z5166106	Bioinformatics Engineering
Edward Webb	z5207215	Software Engineering
Emily Ngo	z5164090	Software Engineering
Rason Chia	z5084566	Software Engineering
Rebekah Chow	z5160152	Software Engineering

Supervisor:	Maurice Pagnucco
Assessor:	Dr John Shepherd
Submitted:	November 2021

Abstract

More and more educational institutions depend on online learning management systems for distributing resources to students, which has been accelerated by the coronavirus pandemic, and so there are a huge number of learning management systems available. These systems have a wide variety of features such as quizzes, blogs, assessment management, integrations with third party platforms such as Zoom and TurnItIn, and more.

However, many of these systems do not have an efficient or useful way to reuse and organise content and resources used in other courses. Time is wasted organising and uploading content for students that potentially may be already curated and used in a different course. Instructors find it quite difficult to create and manage courses as it involves creating content such as lecture slides that may be already used, allowing for duplication of content in the learning management system. Many of these learning management systems are also difficult to use for both teachers and students and are missing key features such as forums and teachers may look to other platforms such as Piazza for these features in conjunction with the learning management system.

Therefore, we will implement a learning management system with an improved UI that is more accessible and easier to use for both teachers and students, and adopts a system where academics can upload content to a specific "topic" or "subject" that is part of a broader database of content. This topic or subject amounts to approximately three hours of content, and can have prerequisite topics which students must complete first. Topics can also be reused easily with a cloning feature available and the use of prerequisites. This avoids duplication of content, and helps build up a large database of content that is well organised and can be used to facilitate learning more effectively. Also, we have built a wide range of features of typical learning management systems such as forums, gamification, quizzes, assessments, and more around this idea of "topics".

Acknowledgements

The authors of this thesis would like to extend their deepest gratitude to Professor Maurice Pagnucco for supervision, support and assistance to this thesis, and Dr John Shepherd for his feedback and suggestions as well. The student authors would like to thank previous thesis students Apoorva Jayesh Bhagchandani, Ki Fung Kelvin Ting, Suphachabhar Nigrodhananda and Katrina Ding for providing examples and assistance with their experience in their thesis and copies of their thesis A and C documents. Finally, the authors would like to thank Tammy Zhong for her thesis documents and code, which helped immensely in the development of the topic tree.

Contributions

This thesis has been a group effort between Allen Wu, Daniel Ferraro, David Nguyen, Edward Webb, Emily Ngo, Rason Chia and Rebekah Chow. The work was separated evenly amongst each group member, as agreed upon during weekly meetings on Monday afternoons. Each group member was in charge of researching one learning management system and leading at least one feature. Additional sections were also divided amongst group members to make up the final thesis presentation and report. A more detailed breakdown of those responsible for the different sections of this report can be seen below.

0.1 Literature Review

Each group member did a survey of the current learning management systems available and produced an in depth review of one of these learning management systems.

- Google Classroom - Allen Wu
- OpenLearning - Edward Webb
- Moodle - Rebekah Chow
- Edmodo - Emily Ngo
- WebCMS - Rason Chia
- Canvas - David Nguyen
- D2L Brightspace - Daniel Ferraro

0.2 Project Execution

Each feature had a feature lead and potentially another student to assist. The feature lead, denoted by the bolded name below, was in charge of producing the report for their feature.

- Topic Tree - **Edward Webb** and Allen Wu
- Course Pages - **Allen Wu** and Rebekah Chow
- Forums - **Rebekah Chow**
- Assessments - **Emily Ngo**
- Accounts - **Daniel Ferraro**
- Backend - **David Nguyen** and Daniel Ferraro
- Lectures and Tutorials - **David Nguyen**
- Gamification - **Rason Chia**

0.3 Additional Sections

Group members also took on additional sections in order to produce a complete report.

- Contributions - Rebekah Chow
- Introduction - Edward Webb
- Aims - Allen Wu
- Project Execution - Rebekah Chow, Daniel Ferraro and Rason Chia
- Conclusion - Daniel Ferraro and David Nguyen

Abbreviations

BE Bachelor of Engineering

UNSW University of New South Wales

LMS Learning Management System

Contents

Chapter 1

Introduction

1.1 Motivation

A learning management system is a software system that administers, documents, facilitates and tracks the progress of educational courses, training programs, and more. They are mostly deployed by educational institutions such as universities and schools, but are also used at corporate institutions to facilitate training programs[Ell12]. A learning management system helps academics create and organise course materials, track students' progress, improves communication between academics and students and helps facilitate remote learning, which is especially important with the pandemic [Tiw]. Students will have easier access to course resources and can also track what assessments are due, as well as communicate with other students of the same course.

This thesis will focus on learning management systems (LMS) that are deployed at institutions. There are a wide range of learning management systems available, such as Moodle, WebCMS, Edmodo, OpenLearning, and more that will be looked at in this thesis, however most do not allow for easy reuse and management of content and do not provide good import or export functionality of educational material. Some provide simple functionality of reusing materials from the same course in a new offering of the course, but if a new course is created from scratch it can be difficult to import and reuse material from other courses.

Most LMS also do not provide key features such as searchable forums or a polished UI, and so some instructors will look at using other software platforms in conjunction with the learning management system, such as Ed or Piazza. This can degrade the learning experience for students, and make it more difficult to facilitate student learning.

1.2 Meta Learning Management System

This meta learning management system (Meta LMS) aims to provide easier reuse of content and an improved structure of content for instructors. Instructors will upload content to a "topic", a subject that students will learn about. For example, a COMP1511 (Introduction to Programming) instructor would upload content about pointers to a topic called "Pointers". Topics can have prerequisites, for example to learn the topic called pointers, students must learn about "Memory in C". This allows instructors to easily reuse content, as they can assign a topic as a prerequisite. Instructors can also easily clone topics, improving reusability. There is also more structure to content, making it faster to create and manage content.

This is as opposed to a traditional LMS, because a traditional LMS involves instructors adding content to a specific "course". Content is not added to a global repository of content and if another course uses similar content, instructors will curate new content for that course. The "meta" part of this LMS involves instructors being able to reuse content for their topic group or "course", by assigning topics as a prerequisite or cloning existing topics into their "course", and otherwise curating and uploading content to this global repository under a "subject" or "topic".

The Meta LMS will be built collaboratively, with students developing their own features of the LMS and collaborating to integrate it into the system. This is explained further in the contributions section of this thesis.
Features such as the topic tree, exam management and quizzes will help improve course management for instructors. Similarly, features such as gamification, assessment notifications and forums also help improve students' learning by facilitating student learning.

1.3 Aims

The aims for developing a meta LMS are to provide better flexibility, utility and usability for both course instructors and students. Therefore the aims are to develop a LMS that allows instructors to easily curate courses and reuse content and provide students with a high quality learning experience. This report has considered a comprehensive set of features that we plan on implementing to achieve these aims. These features were developed through analysing other LMS's which are compiled in our literature review. The literature review provides a basis in which to analyse other competitors and understand their design principles and implementation. Through the literature review a set of features for the designed meta LMS are planned in which further the aims for developing a meta LMS.

1.4 Contributions for the LMS

This Meta LMS was developed in sections, with the backend team working with everyone to integrate each feature into the overall system. Each feature has a lead but most features have multiple people contributing. These features include:

Accounts - Daniel Ferraro
Accounts involve user management, logging in and registering users. A secure form of authentication must be chosen, and users should be logged in automatically if they have previously logged in.

Topic Tree - Edward Webb and Allen Wu
A topic tree feature will allow teachers and academics to add educational material under a specific topic or subject. Each topic will have prequisites of other topics, for example the topic Graphs will be a prequisite for the topic Depth first Search. Allen developed the Topic Tree List View, and Edward developed the Graph View and the rest of the features related to the topic tree.

Course Pages - Allen Wu and Rebekah Chow
Course Pages involve the home page of the LMS and course management overall. This includes course outlines and page creation. It is the place where students and academics mainly interact with the LMS and acts as the centralised page in which other components are linked, for example the lectures, tutorials and quizzes.

Lectures and Tutorials - David Nguyen
Lectures and Tutorials includes how academics will post content to the LMS, access tutorials and lecture recordings/live streams, and how students will view lecture and tutorial content.

Assessments - Emily Ngo
Assessments involve quizzes, viewing grades, displaying user progress and assignment management. Academics will be able to control when to release grades and quizzes can open and close at certain times.

Forums - Rebekah Chow
Forums include how students and academics communicate through the LMS. Forums will be searchable and users can create posts and comment on other posts.

Gamification - Rason Chia
Gamification includes implementing game thinking and game mechanics into core features of the platform to provide a more interactive learning experience for students.

Backend - David Nguyen and Daniel Ferraro
Backend involves working with the rest of the team to integrate each feature with the whole LMS. An API will be set up using tools listed in this thesis which will be used as the main form of communication between the frontend and backend.

1.5 Thesis Structure

The overall thesis structure includes:

- Chapter 2 - Background (Literature Review)
  This chapter covers the current state-of-the-art surveying several popular learning management systems available such as Moodle and WebCMS. It compares each on their features, usability and performance with a comparison table;
- Chapter 3 - Project Approach
  This chapter discusses our approach to the thesis, including requirements and definitions;
- Chapter 4 - Project Execution
  This chapter discusses implementation details including designs, database details and more;
- Chapter 5 - Project Walkthrough
  This chapter walks through the actual implementation of the proof of concept system, including screenshots of the system as well;
- Chapter 6 - Analysis
  This chapter analyses how successful the proof of concept is in achieving the defined requirements, including usability and performance testing;
- Chapter 7 - Conclusion
  This chapter concludes the thesis and includes any future work that could be implemented after the conclusion of this thesis.

Chapter 2

Background

In order to get a good understanding of what is required of an LMS, popular learning management systems used by education institutions today were surveyed and reviewed based on their feature set and performance.

The learning management systems considered in this thesis include

- Google Classroom;
- OpenLearning;
- Moodle;
- Edmodo;
- WebCMS;
- Canvas; and,
- D2L Brightspace.

Previous research conducted by 2020 Thesis A students, Suphachabhar Nigrodhananda, Ki Fung Kelvin Ting [NT20], Apoorva Jayesh Bhagchandani [Bha20] and Tammy Zhong [Zho20] was also taken into consideration when surveying the current landscape for learning management systems.

2.1 Comparison table

Categories	Canvas	WebCMS	Moodle	D2L Brightspace	OpenLearning	Edmodo	Google Classroom
Topic Tree	No	No	No	No	No	No	Yes
Account	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Course Pages	Yes	Yes	Yes	Yes	Yes	No	Yes
Assignments	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Dashboard	Yes	Yes	Yes	Yes	No	Yes	Yes
Quizzes/Exams/Tests/Polls	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Forums	Yes	Yes	Yes	Yes	No (comments system only)	No	No
Multiplatform Access	Yes	No	Yes (Has separate mobile application)	Yes	Yes	Yes	Yes
Accessibility	Yes	No	Yes	Yes	Yes	No	Yes
Grading	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Attendance	Yes	No	Yes (Requires a plugin)	Yes	No	Yes	No
Calendar	Yes	No	Yes	Yes	Yes	Yes	Yes (Provided by Google Calendar)
Enrolment	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Blogs/Wikis/Discussions	No	Yes	Yes	Yes (discussions)	Yes	Yes (Discussion only)	No
Notifications	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Lectures and Tutorials	Yes	Yes	Yes	Yes	Yes	No	No
Third Party Integration	Yes	No	Yes	Yes	No (API is provided)	Yes	Yes
Inbox/Messaging	Yes	No	Yes	Yes	Yes	Yes	No
Gamification/Karma System	Yes	No	No	No	Yes	Yes	No
High Quality User Interface	Yes	Yes	Yes	Yes	No	Yes	Yes
Open Source	Yes	No	Yes	No	No	No	No
Data Migration	Yes	No	Yes	Yes (SCORM, Common Cartridge)	No	No	Yes
Performance (out of 10)	8	8	6	6	4	8	8

2.2 Google Classroom (https://classroom.google.com/)

Google Classroom is an online learning management tool provided by Google which offers a free and easy tool for helping educators manage and assess progress of students. It has become one of the most popular learning management platforms adopted in schools across Australia. The main selling points of this platform are:

- The ability to easily manage students learning:
  - Allowing students to join classes directly or by sharing a code or link;
  - Setup a class quickly and create class work that is displayed on students' calendars; and,
  - Providing communication with students' guardians and automatically sending them updates.
- The ability to easily measure student progress:
  - Storing frequently used feedback to be used for fast and personalised responses;
  - Allowing teachers to grade consistently and transparently with rubric integrated student work; and,
  - Providing students with a plagiarism checker to help them produce their own original work.
- The ability to provide collaboration between students and teachers:
  - Connecting students and teachers online in virtual classes;
  - Allowing communication of announcements on the Stream page; and,
  - Enabling face to face connections with students using Google Meet.
- Keeping data secure:

  - Authenticating users through a login feature;
  - Restricting classroom activities to members; and,
  - Assuring data is never used for advertising purposes.
- Allowing for third party apps to be integrated in learning for example:
  - Schoolytics (https://www.schoolytics.io/): Allows educators and teachers to generate insights on student learning; and,
  - Studyo (https://studyo.co/): Provides educators and students with more comprehensive planning tools.

2.2.1 Features

Google classroom provides all of the basic functionalities and some unique features of a learning management system.

- Allows teachers to create classes/groups which students can join through a link or code;
- Allows users to create announcements in which others can comment within the class;
- Allows teachers to create classwork by creating questions, assignments, quizzes or class material;
- Classwork can be scheduled and the due dates of the classwork will show up on students' calendars;
- Classwork, posts and announcements can be reused;
- Other Google applications (e.g. Google Drive, Google Docs, Google Sheets, Google Presentations and Google Meet) are integrated within Google Classroom, offering more utility for all users; and,
- Allows teachers to input grades which are viewable by students

2.2.2 Review

In terms of the aspect of ease of use, flexibility and usability, Google Classroom offers teachers to reuse questions or sets of questions for classwork. Figure 2.1 highlights the functionality that allows teachers to reuse classwork. While reusing classwork, teachers are able to then set separate parameters such as the due date, add more questions, change the topic and change mark allocation.

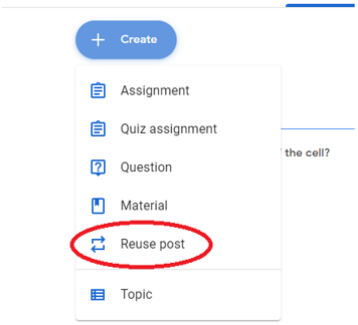

Figure 2.1: Teachers can reuse posts.

Topics can also be created in Google Classroom. Topics allow for categorisation of class work. An example of the topics is shown in Figure 2.2. The example displays 2 topics, each with 1 question. One functionality that is not implemented is the use of prerequisites for topics. For example, if topic1 was a prerequisite for topic2, then students who have not completed topic1 cannot access topic2. This design does not show how each topic is linked with each other which could be improved on.

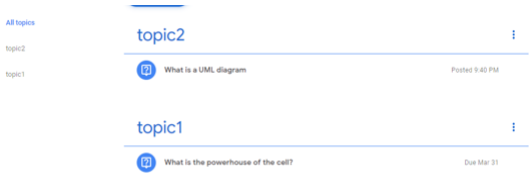

Figure 2.2: List of topics.

One interesting feature within Google Classroom is the integration of other first party applications such as Google Calendar and Google Drive. Google Calendar is a very popular task reminder and calendar application and would benefit greatly with the incorporation of displaying due dates for classwork from Google Classroom. Google Calendar allows for the easy visualisation of important dates and thus is suitable for Google classroom. Google Drive is also a very popular file hosting server where files can be easily shared with other people. For the case of Google Classroom, it can be utilised to share classwork with students.

Due to Google Classroom being affiliated with Google many of the other platforms

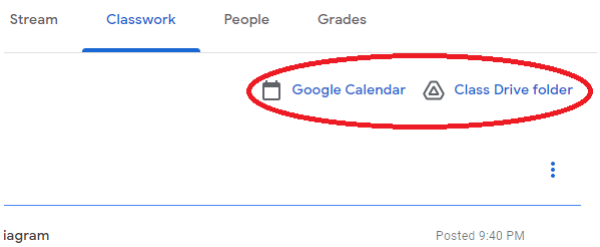

Figure 2.3: Google Drive and Google Calendars are integrated.

related to Google are integrated very well. Announcements can be created which have Google Drives and YouTube videos attached which further improve the usability of Google Classroom.

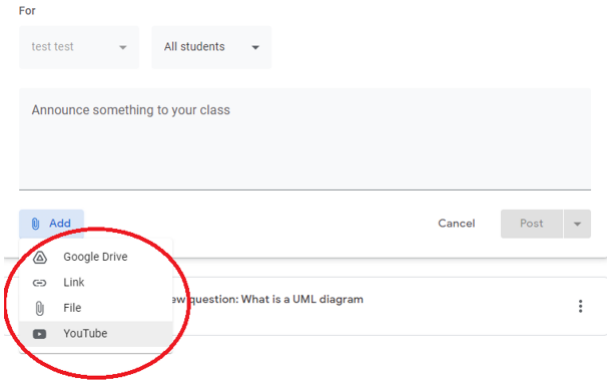

Figure 2.4: Google Drive and YouTube are integrated

Overall Google Classroom does not have many unique features that make it an outstanding learning management platform; however, it does provide the integration of other Google platforms such as Google Drive, Google Calendars, Google Docs, Google Sheets, Google Presentations and YouTube.

2.3 OpenLearning (https://solutions.openlearning.com/)

2.3.1 Overview

OpenLearning is an ASX listed company that provides an LMS for educational providers. It was founded by Adam Brino, Richard Buckland (CSE lecturer) and David Collien and is used in security courses at UNSW[Opea].
It works with universities such as UNSW and Taylor's University (based in Malaysia) to deliver MOOCs (massive open online courses). It also features an LMS that educational institutions can use. Some courses at UNSW currently use it such as COMP6441 (Security), and many universities such as UNSW, UTS, ACU, Charles Sturt University and more also use the platform.

Unlike other content management systems such as Moodle, OpenLearning is not free or open source and can be quite costly[Opeb].

2.3.2 Home

When logged in, the home page of OpenLearning is quite confusing.

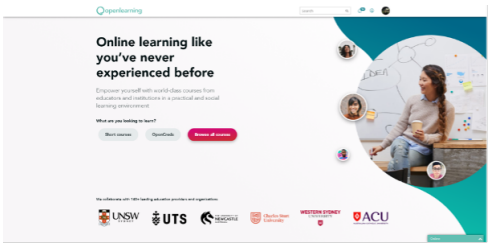

Figure 2.5: OpenLearning Homepage

For students, it can be quite confusing how to reach their courses. The home page clearly directs you to search for a new course offered by OpenLearning, not a course you are already enrolled in.

The balloon icon in the top right should be clicked to reach my courses.

Figure 2.6: This icon is confusing and it is not clear how to get to my educational resources.

2.3.3 Course Home Page

Figure 2.7: OpenLearning Course Home

The course home page UI is good, with clear links to information and important announcements being easily viewable.

2.3.4 Performance

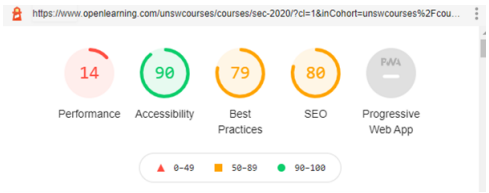

Figure 2.8: Google Lighthouse Result

The overall performance of OpenLearning is poor, with pages feeling slow and sluggish with Google Lighthouse, a performance testing tool [Gooa], giving 14/100 for performance. Google Lighthouse uses several metrics such as time until the website is interactive, how quickly contents of a page are populated, etc. to measure performance and produce a score [Goob]. It also measures other metrics such accessibility which seems otherwise quite good, with a 90/100 score.

2.3.5 Forum and Commenting

OpenLearning makes it quite difficult to communicate with other students, with a comment feature available but not searchable or sortable at all. OpenLearning does support liking comments, which will promote a comment to the top of the list. However, it can be difficult to find a specific comment unlike other learning management systems such as

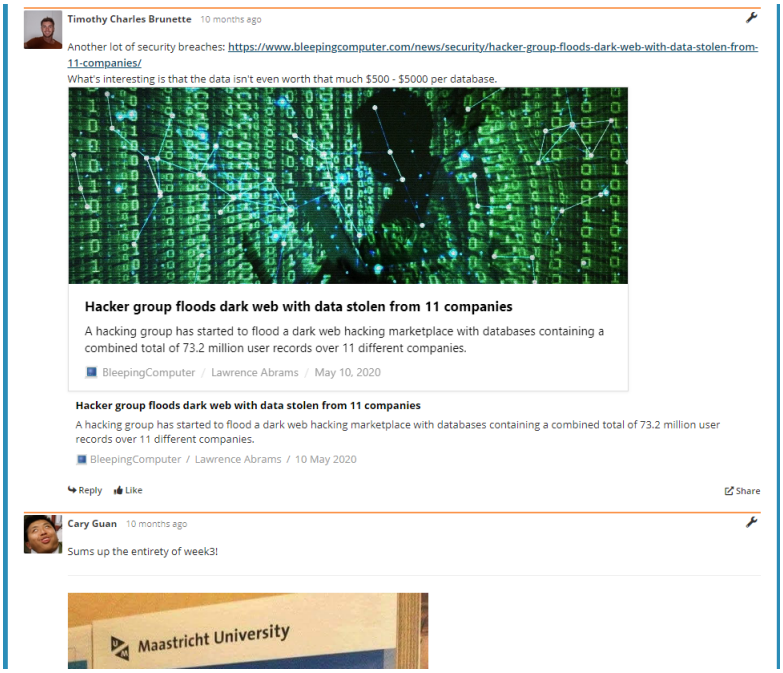

Figure 2.9: OpenLearning comments system

WebCMS, which allows students to search comments (further expanded below). There does not seem to be a forum feature either, and so in courses such as COMP6441, it can be very difficult to collaborate and communicate with classmates.

2.3.6 Data Migration

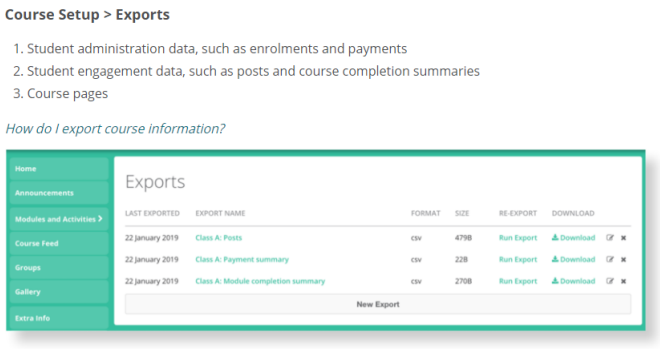

Figure 2.10: Exporting data in OpenLearning

Data migration seems quite poor in OpenLearning. There are no support articles available on how to import data from another course, and so it seems quite difficult to

create a new course. There is a basic system to export data into a csv file[Opec], but there does not seem to be a way to import data either.

2.3.7 Conclusion

OpenLearning is a good platform for instructors and students, with a huge feature set such as blogs, commenting, and a very flexible course setup. However, the UI, performance and stability can be significantly improved to be more competitive with other LMS platforms. A searchable forum can also be implemented, and little data migration features can hinder courses that use the platform, as time is wasted uploading and curating content for courses.

2.4 Moodle (https://moodle.com/)

2.4.1 Overview

Modular Object-Oriented Dynamic Learning Environment, more commonly known as Moodle, is a learning management system founded in 2002 by Martin Dougiamas. Supporting over 60% of higher education institutions around the world [Mood], including UNSW, Moodle provides an open-source, personalised learning platform aimed at supporting both students and educators. Moodle's modular design is centered around providing a personalised experience, giving users' the flexibility to build and access courses in a way that suits them. With a vast library of plugins, and the ability for developers to design their own, Moodle really provides users with unlimited functionality, [eTh].

2.4.2 Dashboard

When a logged-in user first accesses Moodle, they are directed to the Dashboard which consists of content blocks for each of the user's courses. User's are provided with numerous ways to customise the dashboard, including:

- Hiding and showing individual courses;
- Filtering courses by past, present and future;
- Sorting courses alphabetically or by date; and,
- Displaying courses as cards or in a list.

Users can use these customisation tools to organise the dashboard based on their needs and, as a result, easily find a desired course.

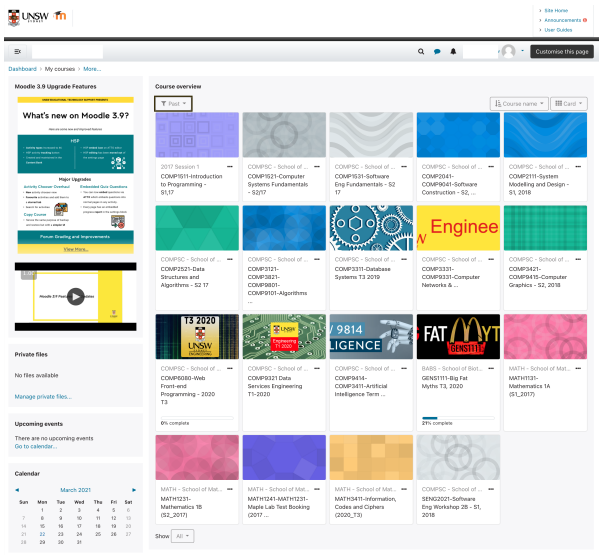

Figure 2.11: Moodle's Student Dashboard.

2.4.3 Course Pages

A course page consists of various links to resources. These links can be separated into collapsible subsections which help with the organisation of a course. Course administrators can customise the subsections based on how they would like to structure a course. Course administrators can also customise the way in which students work through course content by blocking access to resources until certain criteria have been met.

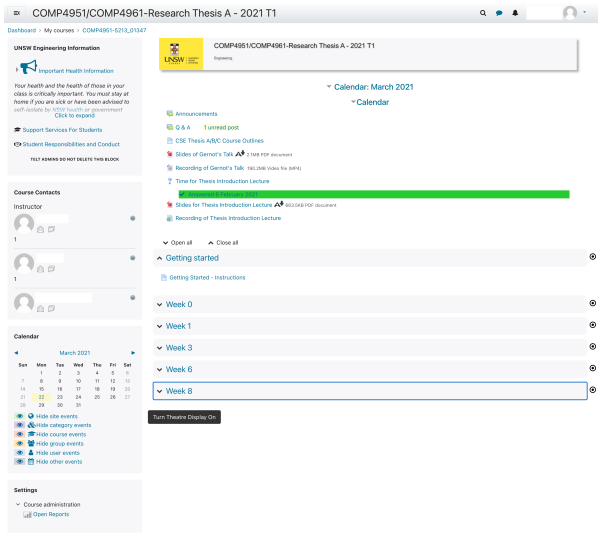

Figure 2.12: Moodle's Student Course Page.

The course page sidebar can also be edited to include different sections like course

contacts, calendars, announcements and useful links.

Within a course page, students can access announcements, discussions, forums, quizzes, polls and course resources like lecture notes and assessment notifications. Course administrators can open assignment submissions and provide students with assessment feedback and grades.

2.4.4 Forums

Moodle provides course administrators with the ability to create one or more forums attached to a course page. Students and administrators are able to post to the forum, reply to others, star posts and subscribe to conversations. There is no easy way to search through forum posts which can make it difficult to find information, especially when the post list is long. There is also no functionality for filtering or sorting posts. Overall, the forum functionality in Moodle is pretty basic and could use a lot of improvement when it comes to the organisation of information and the overall experience.

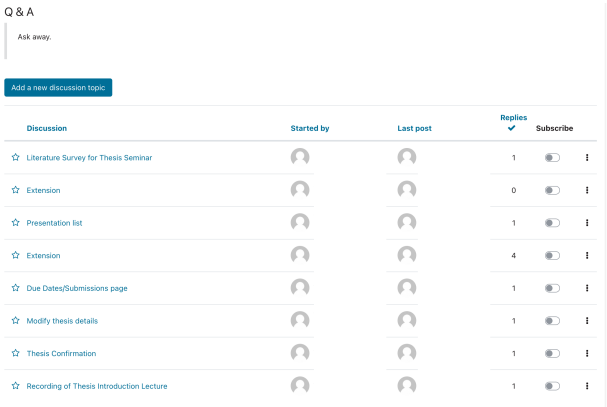

Figure 2.13: Moodle's Forums.

2.4.5 Performance and Accessibility

In order to assess the performance and accessibility of Moodle, Google's Lighthouse tool was used. When run in the Google Chrome browser, Lighthouse is able to scan a page and identify any issues that may be affecting the performance or accessibility of a site [Gooa]. The performance of Moodle is average, with Lighthouse giving it a score between 60% - 70%, depending on the amount of content on the page. A similar score was achieved for the accessibility of the website, with the biggest issues being lack of contrast between background and foreground colours, as well as HTML elements missing the appropriate attributes required for assistive technology.

2.4.6 Data Migration

Moodle provides users with the option to backup and restore data associated with a course [Mooa]. Automated backups for courses can be set up to reduce the risk of losing data. These backups can also be used to migrate data to a different platform [Moob]. Administrators are able to easily select which parts of the course they would like to backup in the backup settings. The backup files can then be easily saved and restored at any time [Mooc].

2.4.7 Conclusion

While the user interface, performance and accessibility could use some improvements, Moodle successfully delivers all the required functionality needed to support students and educators.

2.5 Edmodo (https://new.edmodo.com/)

2.5.1 Overview

Edmodo is a learning management system founded by Nick Borg, Jeff O'Hara and Crystal Hutter in 2008. Edmodo offers an "all-in-one LMS" which includes essential management tools for teachers and a platform for which teachers and their students can collaborate and communicate outside of class. The learning management system serves as a complementary tool catered towards primary and secondary education with a focus on classes rather than stand-alone courses (such as those in tertiary education)[Wik].

Edmodo offers the following features[Edma]:

- Assessment tracking: ability for teachers to see remaining submitted assessments to review, reviewed assessments, scheduled assessments to release
- Grading and recording attendance: ability for teachers to set grades for a student's assessment and fill in attendance for a class
- Collaboration and communication: ability to create a post, discussion, poll and direct messaging
- Assessment options and interactive activities: ability to create assignments, quizzes and gamified activities[Edmb]
- Unlimited storage space: ability to create files in a single click (Word Document, PowerPoint Presentation and Excel Worksheet) and also upload any files

- Social media platform: students and teachers can interact and view others' posts and shared resources, allowing them to learn more about specific topics or develop skills
- Calendar: shows due dates of upcoming assessments, quizzes and events created by the teacher

2.5.2 Dashboard

The dashboard is similar to dashboard of social media platforms, encouraging discussions amongst the students and their teachers as well as keeping students informed of homework and assessments assigned by the teacher.

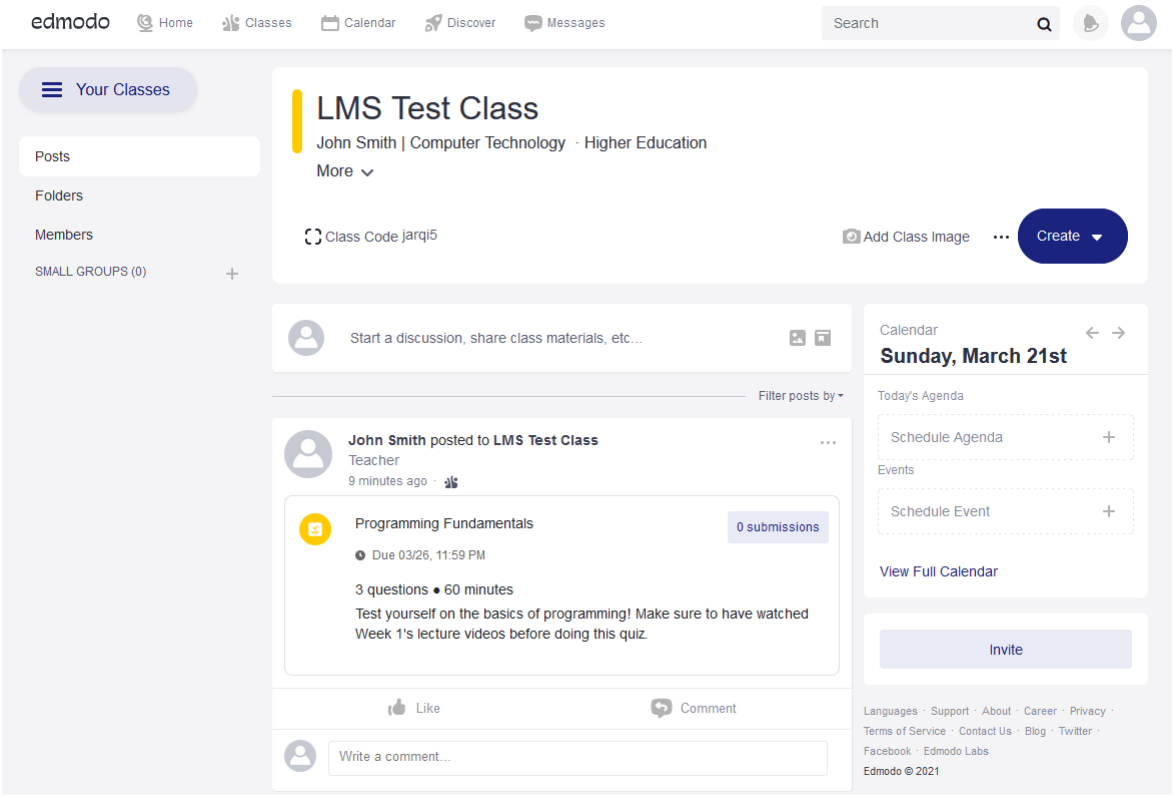

Figure 2.14: Teacher's view of the dashboard on Edmodo

2.5.3 Assignments and quizzes

The creation of assignments and quizzes in Edmodo is very simplistic, however provides limited options and features in which the teacher can use. There is no ability to customise the assessment and teachers are forced to provide extra details in the attachment section in the form of a file.

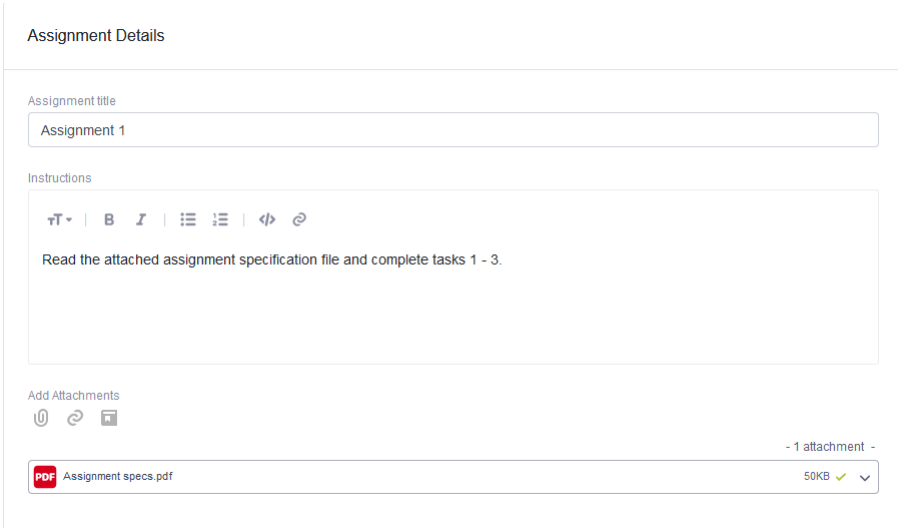

Figure 2.15: Creating an assignment on Edmodo

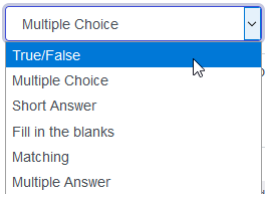

Figure 2.16: Options offered for a quiz question on Edmodo

2.5.4 Data migration

Edmodo does not provide any features to import packages or courses from other learning management systems and is not SCORM compliant[SCO]. This is due to its nature of class content and assessments being in a fixed structure. Teachers are only offered the ability to copy from existing assignments and quizzes made on the Edmodo platform. New teachers transitioning from another learning management system to Edmodo will have no option to reuse their past course content or assignments.

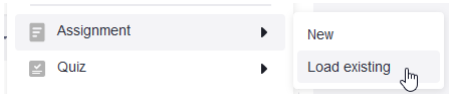

Figure 2.17: Loading an existing assignment on Edmodo

2.5.5 Conclusion

Edmodo offers a clean, social-media style user interface familiar with younger users and provides the essential management tools for teachers, however is one of the weakest

learning management systems in terms of re-usability and flexibility for its features.

2.6 WebCMS (https://webcms3.cse.unsw.edu.au/)

WebCMS is a LMS designed and used internally for UNSW CSE courses. WebCMS is a browser based LMS that allows unauthenticated users to only see course announcement page and other pages require an authenticated user with correct enrolment access to the course.

2.6.1 Enrolments

WebCMS has a robust enrolment system that is managed by lecturers and course admins. Course Staff creates a course for a specific term and enrols students from enrolment list. Students can be enrolled into multiple courses for each term.

- WebCMS has multiple roles that can be assigned to individuals for specific courses (Lecturers, Students, Tutors, Course Admin). Each page in the course can be customised to be accessible by users with specific roles.
- Users enrolled into multiple courses will be able to navigate through the courses via the nav bar at top of page. Users can have different roles for each course. (Student in course A and Tutor in course B)

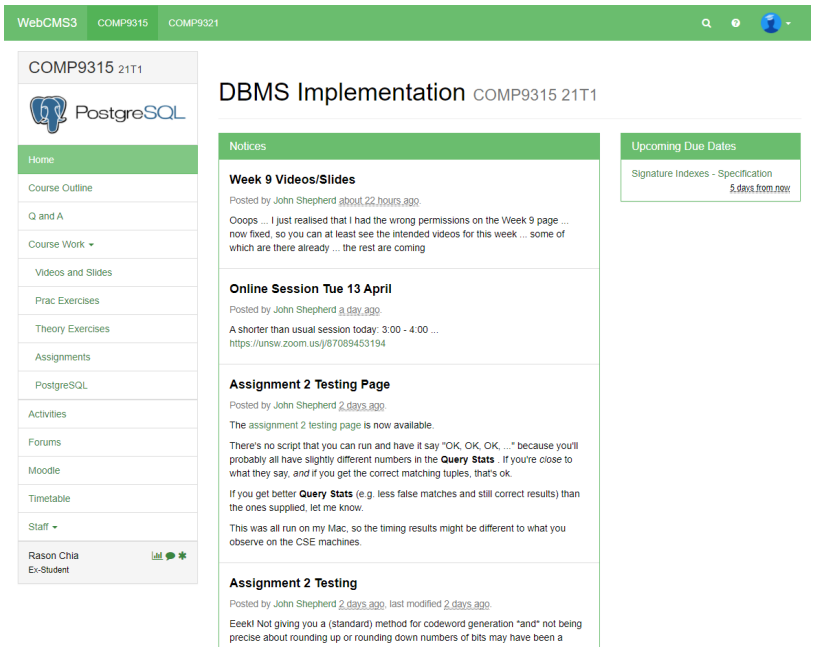

Figure 2.18: WebCMS's Student Course Page

2.6.2 Course Page Feed

Each Course has a homepage that users will land on when navigating to the course. This main page consists of 3 main components, a static navbar to navigate course content, Course Notices / Announcements and due dates for deliverables.

- Course Notices / Announcements can be posted by lecturers or staff member. These posts will automatically generate an email notification to all enrolled users of the course. Posts are visible to all users and provide course wide information like release of assignments and exam information.
- Due dates of deliverables are retrieved from quizes and assignments that have a due date set. The panel will show the number of days remaining as well as a link to the specific deliverable.
- The static sidebar provides an indexed view of course content. Content links can be grouped under a heading. This allows users to quickly navigate through to the content they are looking for. This sidebar also display the authenticated user's name, role in the course, shortcut links to user's gradebook, wiki and blog.
- Course Admins can set course theme color which is propagated through all the above elements as well as throughout all pages for the specific course. Each course can have a different theme color and this helps users differentiate which course they are looking at.

2.6.3 Email Notifications

WebCMS has a email notification feature which can be triggered automatically on various conditions. By default, any course wide notices always striggers email notifications.

- Users have the eability to subscribe to their posts (comments, replies, forum posts) which will automatically triggger notifications if there has been a new update to their posts. This can be toggled in the settings menu.

2.6.4 Quiz and Assignment System

WebCMS allows multiple quizes and assignments to be created for each course. These deliverables have a due date and can be fully customised, automarked and results displayed directly into the user's gradebook.

- Quizes can comprise of only multiple choice answers or have a mix with short & long response answers. Correct answers can be set for each quiz, once quiz is due, users are not allowed to make anymore submissions and quizes can be automatically marked by WebCMS.
- Assignments can be created with multiple tabs (Specification, Make Submission, Check Submission, Collect Submission). Specification can be formatted or displayed through embedded pdf. Make, Check and Collect Submission functionalities interacts with CSE give command line tools for specific assignment.

2.6.5 Display Content

WebCMS has a robust page formatting toolset which allows the user to format content with headers, create lists and embed content (videos, images).

- Course Setting can restrict that only lecturers or staff members can create or edit the page.
- Pages created can be pinned into the static navbar on the platform for easy access
- Pages can be access by direct link to its content resource number that is unique to each page. e.g. `https://webcms3.cse.unsw.edu.au/COMP9321/21T1/resources/59281`
- Pages can be embedded with pdf files which provides a basic pdf viewer component so users can view pdfs while remaining on the platform

2.6.6 Gradebook and Profile

WebCMS gradebook stores marks from deliverables like quizes, labs, assignments. User Profiles also has a blog and wiki feature where users can document notes as well as store a blurb about themselves which is visible to other users.

- Quizes which are set to automark can record their scores automatically into the gradebook.
- Marks for Quizes, Labs and Assignments can be manually inputted into the gradebook for each user. This feature can be restricted to be used by staff members only.
- Gradebook allows for total of specfic marks to be displayed on the gradebook. This is done to provide a summary view of a student's score over all the deliverables.

2.6.7 Admin Functionalities

WebCMS administrative functionalities can be restricted to certain user groups like staff members. Admin Functions allow staff members to create and manage course content & structures.

- Course Creation Tools: Creating Brand New Courses and Stetup, Cloning previous courses, Link Course to myunsw enrolment course student lists
- Course Management Tools: Setup deliverables for the course (Quizes, Assignments, Labs), Setup Course Content (Lecture Slides, Labs, Assignment Pages, Quiz Questions),
- Student Interaction Tools: Setup pages to have commenting functionality, Setup forums, Setup Notification Settings

2.6.8 Conclusion

WebCMS is a comprehensive and robust Learning Management System, however it does fall short of some other LMS in the market. WebCMS is not an open source platform that can easily recieve external data or export its own data, it is built as a platform to store / share content but it does not engage students directly with any self serve content like gamification features or self serve modules.

2.7 Canvas (https://www.instructure.com/)

2.7.1 Overview

The Canvas LMS is considered the best in North America and is used by all Ivy League Schools[Cana]. Instructure Canvas uses open-source and cloud-based technologies to enable easy integration of desirable learning content for teachers and students. Moreover, the LMS is can be deployed for primary, secondary, tertiary and business institutions. Some notable features of Canvas include a collaborative work space with video platform tools, a mobile integrated application and customizable content.

2.7.2 Course Management

The user interface for the course management page for teachers has a minimalist design which allows users to effectively navigate and use the page. This page offers the feature to import course materials locally and from the Canvas Commons Repository.

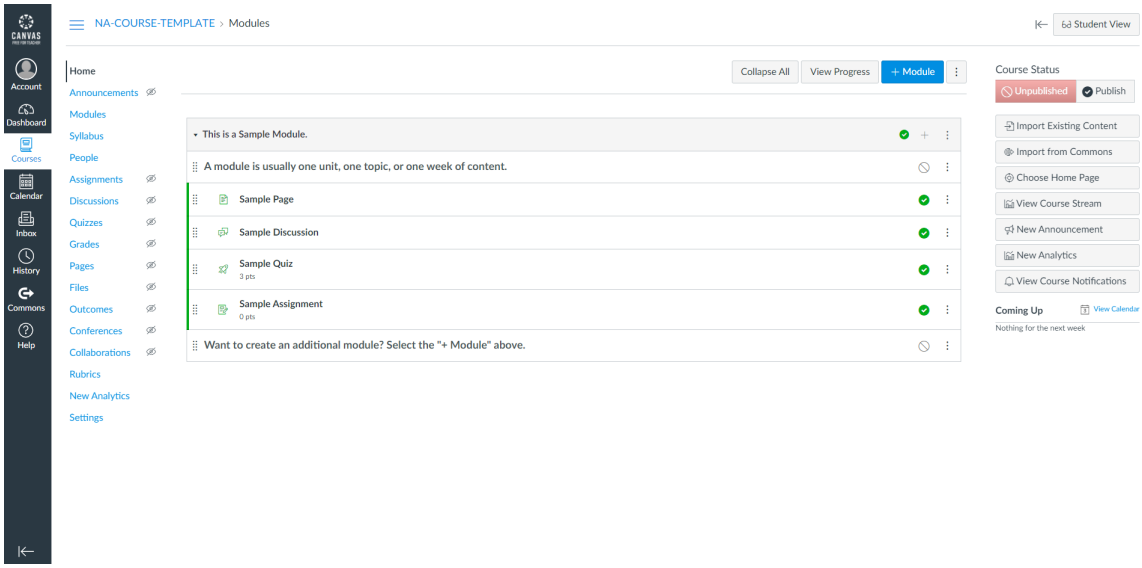

Figure 2.19: Course management page for teachers

From this page the teacher can create announcements, view course notifications, edit the layout of the course, toggle visibility of course sections, set important course dates, view course statistics.

Teachers can utilise the Speed Grader feature which supports fast grading of student assignments using a simple point scale or a complex rubric. Additionally, the teacher can set grading schemes for each course. The teacher may automatically set grades for missing or late submissions for assignments.

2.7.3 Discussions

The discussions section of Canvas contains similarities to a forums feature. In this section, any user enrolled in the relevant course can start a discussion. Moreover, the discussions component contains important features that allows users to pin important threads, file attachment to posts and close threads for comments.

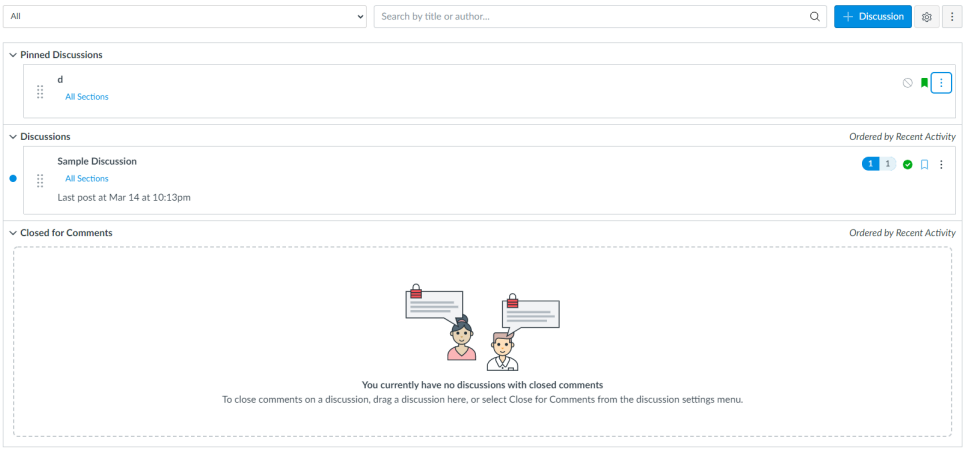

Figure 2.20: Discussions with a pinned thread

2.7.4 Collaborations

Canvas has integrated Google collaborative tools such as Docs and Slides to empower collective learning. This is a powerful feature of Canvas as it allows these collaborative documents to be accessed from the course dashboard and allows the teacher to specify which users can access the document.

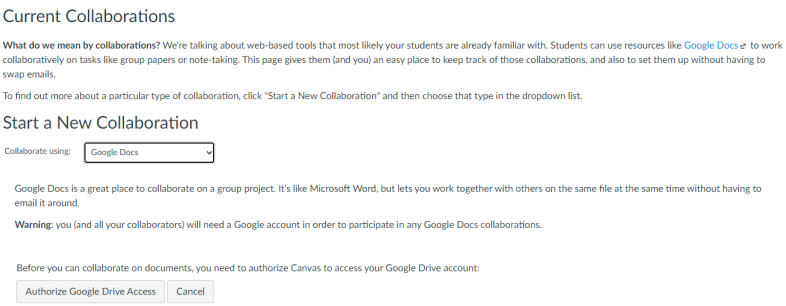

Figure 2.21: Course collaboration page

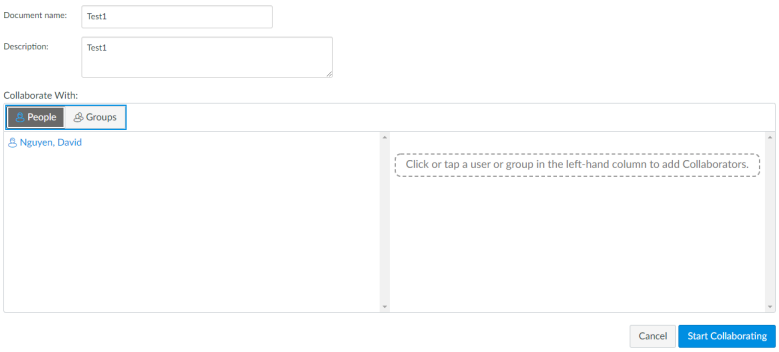

Figure 2.22: Specifying users for collaboration document

2.7.5 Commons

Canvas employs a system called Commons which optimizes re usability among teachers[Canb]. This re usability is possible through teachers uploading their course materials to the Canvas Commons repository.

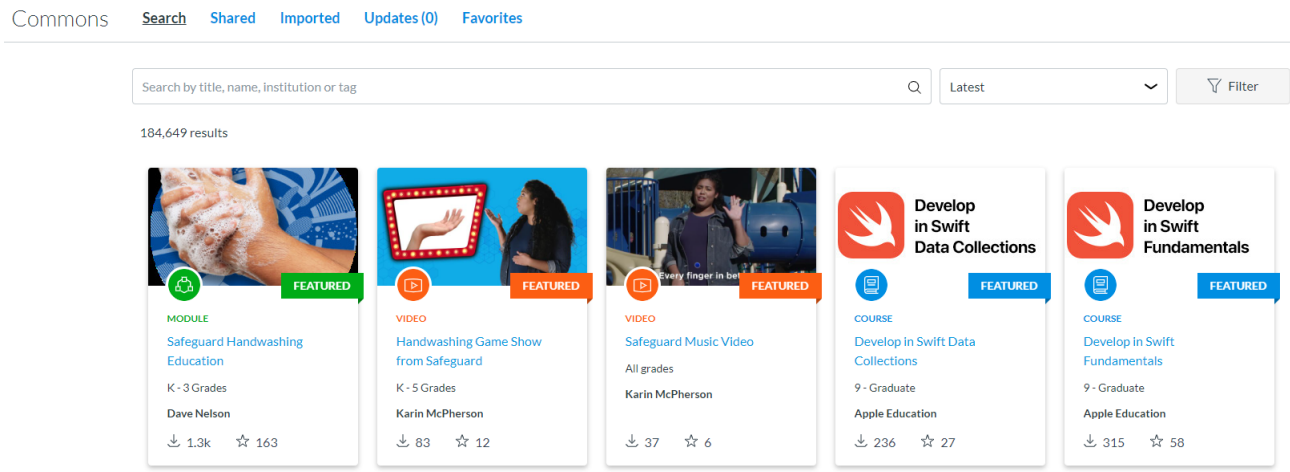

Figure 2.23: A selection of courses on the Commons repository

Additionally, other teachers may utilise the uploaded course materials and incorporate it into their own courses at their leisure. This is accomplished through directly importing the course or downloading the files.

The course creator also has the option to export their course onto the Commons repository for other users to import and use.

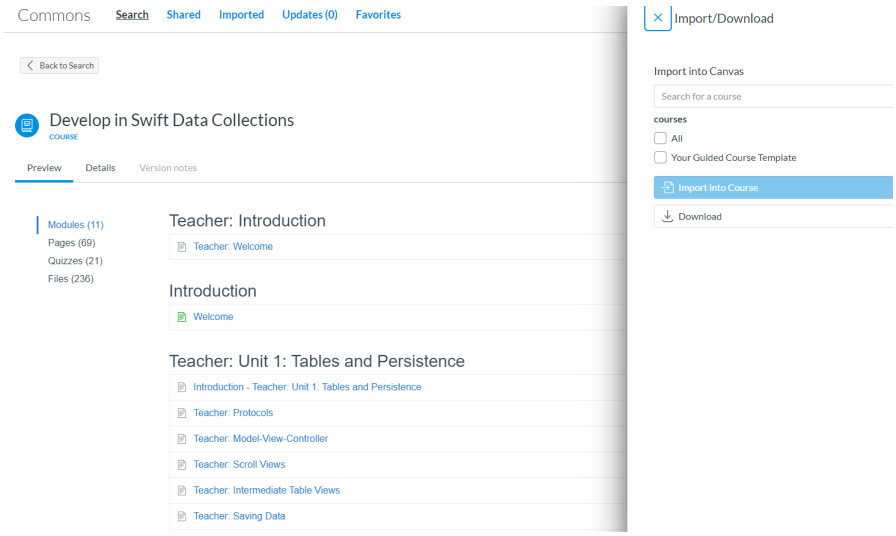

Figure 2.24: Importing into a course

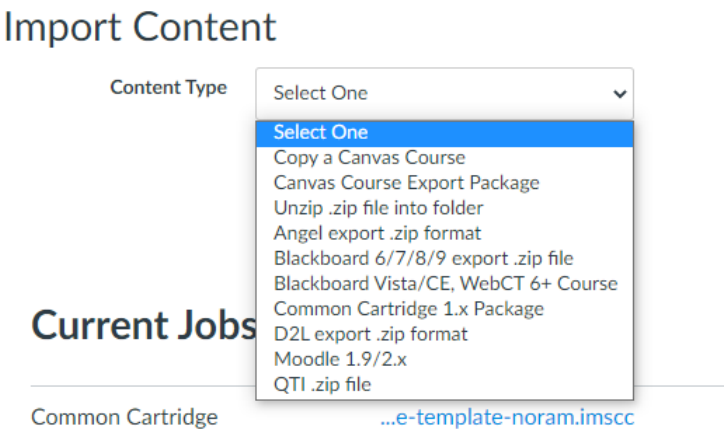

Figure 2.25: Allowable file types for course importing

Figure 2.26: Exporting a course to the Canvas Commons

2.7.6 Conferences

There is a support for a collaborative work space through video platform tools. This is demonstrated through Canvas' Conference feature. This feature allows teachers to stream tutorials and lectures with a variety of white boarding tools during this live stream (Figure). The user interface and usability of the conference page is also simple and intuitive to use.

In this figure there are drawing tools, shape tools and text box which allow the teacher to further emphasise on key features of their live stream. Additionally, the teacher may allow other users in the live stream to utilise the white boarding tools for purposes such as clarification.

Moreover, there is also the presence of a live chat Q and A during the live stream which allows teachers to answer queries during the stream. (Figure)

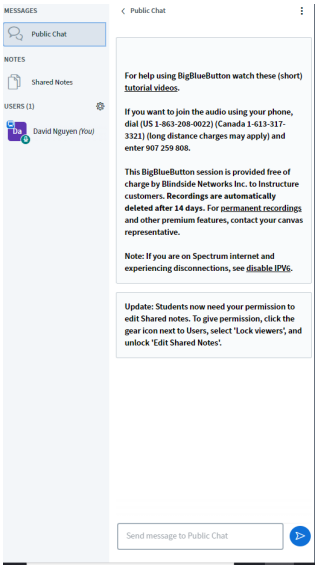

Figure 2.27: Live chat Q and A

During this live stream, the teacher can share their screens and allow other users to share their screens. This further supports the versatility of the conference feature in the education sector. (Figure)

In summation, the Canvas LMS possesses many useful features which allow it to be considered an effective LMS. This is supported through the practical features offered and the overall user experience and design of the LMS which is friendly and accessible for users.

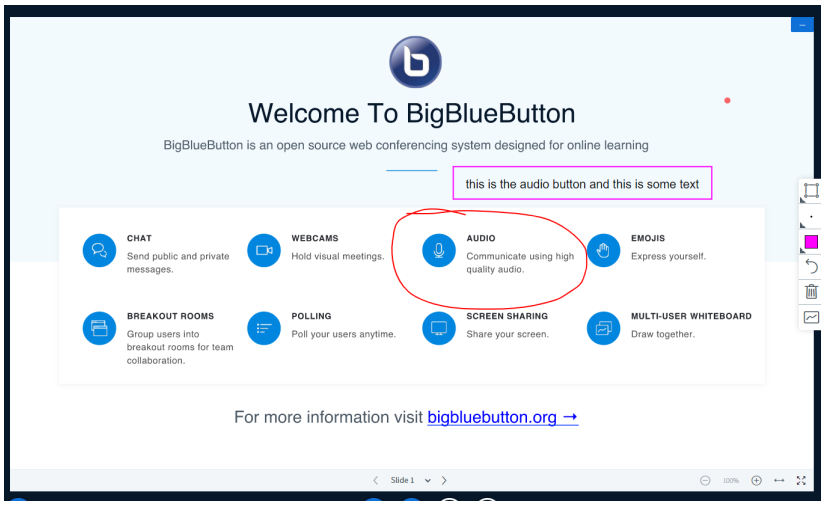

Figure 2.28: White boarding tools

2.8 D2L Brightspace

2.8.1 Overview

Brightspace is a cloud based LMS developed by the software company Desire to Learn (D2L). Its design philosophy is one of flexibility, with the primary goal being to allow every student to study in a way which most benefits them. Brightspace is targeted towards primary and high schools, universities, and corporate learning environments. It offers key features expected of an LMS such as:

- Importing and exporting course contents to and from other SCORM compliant LMS's.
- Easy management of course contents and resources from the courses page.
- Full suite of tools to create assignments, exams and quizzes.
- Comprehensive and fully features mobile services.
- Organisation features for students and teachers such as calendars and reminders.

[D2L]

2.8.2 Course Management and Contents

The UI provided by Brightspace for managing and updating courses and their contents is highly simplistic and intuitive, while still allowing administrators to have a high level

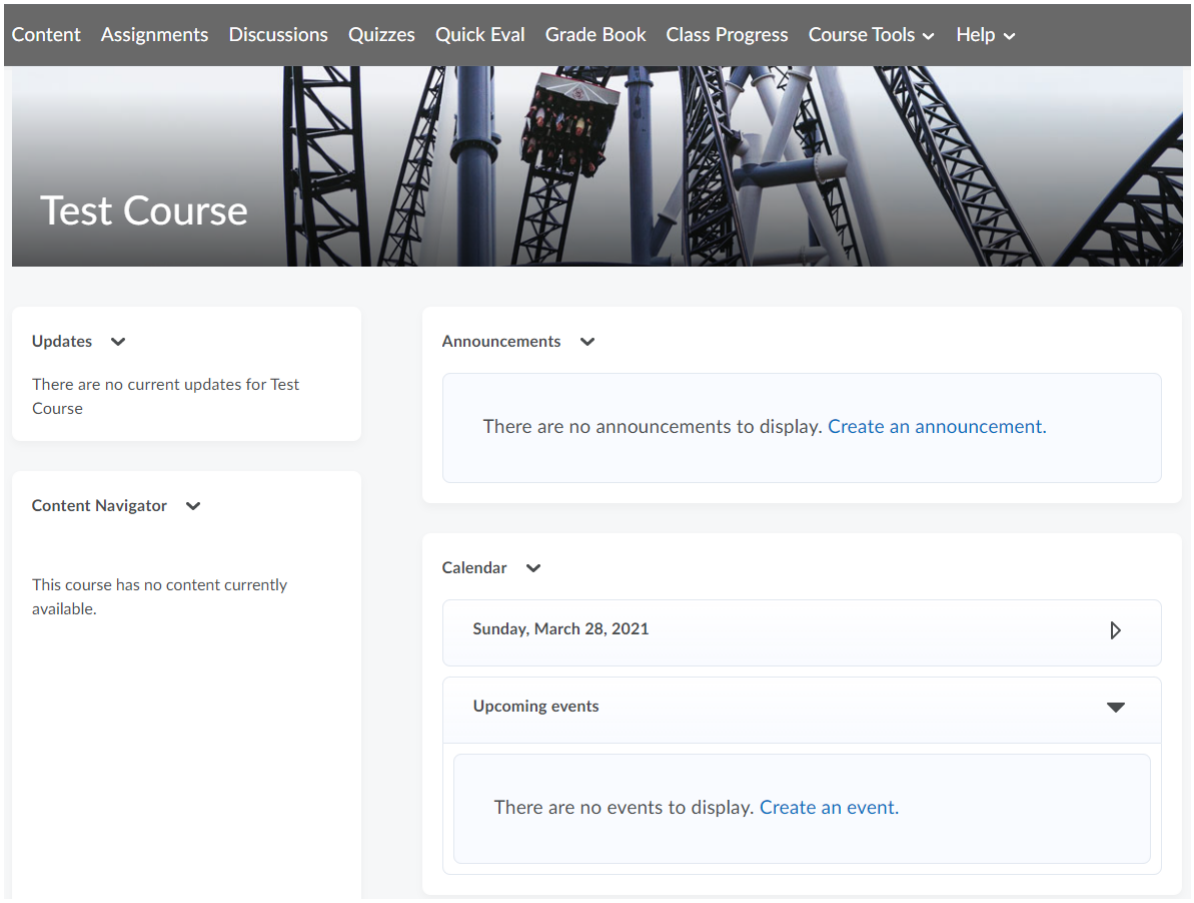

Figure 2.29: The course dashboard from an instructor's perspective.

of control over every aspect of a course. From within a course page, an instructor or administrators can manage contents such as lectures, assignments and quizzes in terms of modules and sub-modules, with the idea that each module is for a specific topic or idea. These modules can be linked to learning outcomes from a rubric, and can be made visible or invisible to specific groups of students within a course.
From within these course pages, instructors and administrators can also view enrollments, grade-books, and have the ability to quickly view any student submissions that require their attention, both on a course by course view or as a combination of submissions from all courses they manage. From the course page, instructors and administrators can also manage the calendar, forum, and attendance modules for that course.

2.8.3 Accounts and Enrollment

Brightspace gives the option for a course administrator to allow a course to either be built with automatic enrollment, where administrators and instructors enrol students

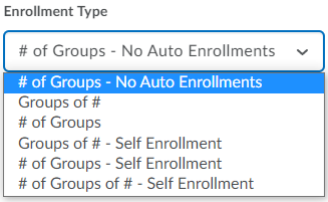

Figure 2.30: The ability to allow groups of students to either self enroll or be enrolled by an administrator.

into the course, self enrollment, where students can enroll themselves either by entering a course code or searching for a publicly viewable course, or as a combination of both, where a course can have a number of spaces assigned to each method of enrollment.

2.8.4 Data migration

D2L Brightspace provides very robust features to import and export course content, and is SCORM 1.2 and SCORM 2004 compliant. These tools allow course administrators to export course components as a Brightspace package, for use on other Brightspace sites, as well as Common Cartridge and Thin Common Cartridges for exporting modules to other LMS's who also follow the Common Cartridge standard such as Moodle and Canvas.
In terms of importing, Brightspace allows administrators to import archived course content from Brightspace itself, as well as other common LMS' such as Angel Learning, Blackboard Learn, Moodle and Sakai, as well as course content compiled according to industry standards including IMS Content Packaging, IMS Common Cartridge, IMS Thing Common Cartridge, and SCORM. It also allows importing of courses and course materials from an online Learning Object Repository (LOR) of hundreds of publicly available courses.

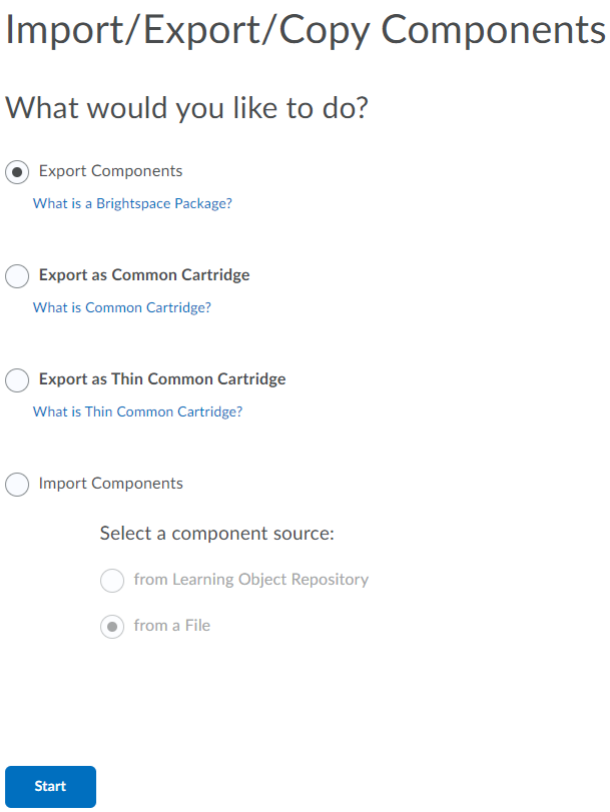

Figure 2.31: Options to import/export a course on Brightspace.

2.8.5 Conclusion

Brightspace delivers a responsive, well designed and fully fleshed out LMS experience with great ease-of-use and compatibility with other SCORM and IMS compliant LMS's, with the primary negative being its cost and lack of open source.

Chapter 3

Project Approach

3.1 Accounts and Enrolments

3.1.1 Overview

The ability to create and manage accounts and enrollments of students, teachers and administrators is a key feature of any LMS. Without being able to create and manage individual accounts for each type of user, the system will be impractical for use in educational environments, and without being able to enroll into courses and topics, many other features of our Meta LMS will not be useful.

There are 4 main pages relating to accounts and enrollment:

- Sign in and sign up;
- Viewing and editing account details;
- administrators enrolling students and creating course invite codes; and,
- Students self-enrolling.

Sign in and sign up

The sign in and sign up screens need to allow users to enter the necessary details to either sign in or sign up, while also being as accessible as possible. Many designs already exist for these components so it is a matter of utilizing the features of existing designs that support accessibility the most. These features include high contrast colors, large readable text, a clear page flow and sufficient text instructions.

Viewing and Editing Account Details

Figure 3.1: A user's view of their account.

User's need to be able to easily view their account information, and that of other users. Accessibility is a key concern for this page, so a clear page flow and easy to read text are most important, as well as a sleek and visually appealing user interface to ensure that information can be comprehended from the page as easily as possible.

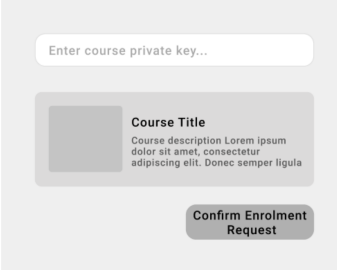

Figure 3.2: A student user's screen to enrol in a course via invite code.

Administrators Enrolling Students

administrators need to be able to individually add student's to a course, and generate an invite code that students can use to enroll in a course.

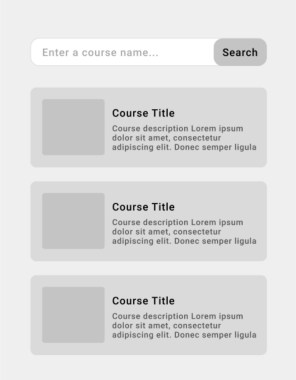

Figure 3.3: A student user's view to search for courses and topics.

Student's Self Enrolling

A student user needs to be able to either enter a course invite code generated by an administrator, or search for a course or topic via its name or course code, to enroll themselves into a course. It is important that when searching for courses and topics, as much information is as clearly displayed as possible.

The accounts system is also in charge of governing the data of the different users. Thus it is important to plan the design of the database schema for a user account. Below is an initial design of what the database table headings for a user may look like:

id	type	name	email	password	phone	address	topics

The ID field has dual purposes, it both stores a users student ID, and is also a unique identifier for every student. The type field stores the account type of a user, such as student or administrator. We also need to be able to store a users personal information, such as their name, phone number and address so there are fields for that. The user needs to be able to log in so an email and password will also be stored. Finally a list of all of the topics a user has completed must also be stored so that their progress can be reflected in any courses they are enrolled in.

3.1.2 Additional Features and Refinements

No additional features were added throughout developments to the accounts and enrollment feature, however many of the design decisions were altered and refined as development and testing caused new issues to be identified. For example the initial database schema above changed during development. It became apparent that this schema would not meet the needs of the LMS in terms of storing user data. This new schema was what was decided on:

id	type	name	email	password	studentID	staff	img_url	last_accessed_topic

The phone number and address fields were removed, as it became apparent that it was not necessary to store this information for the functionality of the LMS. The topic groups column was also removed as the notion of enrollment was moved to its own table to enhance functionality. The user's id was separated from their student ID, to prevent issues with storing data, and an img_url field was added to store the user's profile picture. The last accessed_topic_field was also added so that this information could be stored to be displayed on a users dashboard to create a more personal user experience.

3.1.3 Requirements

The initial requirements for the accounts and enrolment system are listed below. Each requirement has a priority that was measured by surveying the other thesis members. Higher priority items will be completed first, with medium and then low priority requirements to be implemented later in the project in their respective order.

The initial list of requirements were broken up into accounts and enrollments, and were as follows:

Accounts

1. Users can register and login High
2. Users can view and update their details High
3. administrators can manage users, roles and permissions Medium
4. administrators can create accounts on behalf of students Medium
5. administrators can view other users account details Medium

Enrollments

1. administrators can open and close enrolments for courses High
2. administrators can enrol and un-enroll students on their behalf High
3. administrators can generate course invite codes High
4. administrators can set courses to be open enrolment or invite/code only Medium
5. Students can enrol themselves in courses with a code High
6. Students can enrol themselves in an open course or topic High
7. Students can search for open courses Low

However, as the project progressed, it became clear that it did not make sense to keep these requirements divided like this, and that some of the priorities were incorrect compared to the importance of the requirement to the LMS. The requirements were instead adjusted in priority and refined to have acceptance criteria as follows:

Login and Sign Up

Users can register and login High

AS A	User
I WANT TO	Create a new account and log in to my existing account
SO THAT	I can use the LMS
Acceptance Criteria:	
- The user can successfully sign up to the LMS - The user can use an account previously made to log in	

Staff can create accounts on behalf of students Low

AS A	Staff Member
I WANT TO	Create accounts on behalf of students
SO THAT	I can streamline the formation of a course for new students
Acceptance Criteria:	
- The staff can create a batch of student accounts - Students can use these accounts to log in	

Account Management

Users can view and update their details Medium

AS A	User
I WANT TO	View and update my details
SO THAT	I can ensure my information is up to date
Acceptance Criteria:	
- Users can access a page that shows their account information - Users can edit their information from this page	

Staff can manage other users Low

AS A	Staff member
I WANT TO	Manage other users
SO THAT	I can ensure student's accounts contain the correct information
Acceptance Criteria:	
- Staff can access a page that shows a students account information - They can edit this information from this page	

Enrollment Dashboard - Staff

Staff can enrol and un-enroll students on their behalf High

AS A	Staff member
I WANT TO	Enrol and un-enroll students from a course
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that shows all students enrolled in a course - They can enroll new students or un-enroll existing students from the course	

Staff can generate course invite codes High

AS A	Staff member
I WANT TO	Generate course invite codes
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that allows them to generate and manage course invite codes - They can add restrictions to code like number of uses and time valid - They can manage existing codes - When a student uses the code they can enroll in a course	

Staff can set courses to be open enrolment or invite/code only High

AS A	Staff member
I WANT TO	Set courses to be open enrolment or invite/code only
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that allows them to manage whether a course is publically search-able - They can control whether a course is public	

Enrollment Dashboard - Student

Students can enrol themselves in courses with a code High

AS A	Student
I WANT TO	Enrol myself in courses with a code
SO THAT	I can enroll in a new course
Acceptance Criteria:	
- Students have access to an enrollment page where they can enroll in a course with an invite code - After submitting the code, they are enrolled in the corresponding course	

Students can search for open courses High

AS A	Student
I WANT TO	Search for open courses
SO THAT	I can enroll in a new course
Acceptance Criteria:	
- Students have access to an enrollment page where they can search for courses - After searching, a list of courses will be shown to the student - Students can interact with this list to enroll in a course	

The priority of the enrollment features for both staff and students were all increased to high priority due to the discovery of their importance to the entire LMS during development and internal testing. Without these features, the LMS becomes non-functional and thus they needed to be included. The admin management of student accounts was lowered in priority, as it became clear that these features were not essential to the functionality of the LMS, and if they did not manage to be completed it would not affect the functionality of the overall LMS too greatly.

In addition to assessing how well the accounts and enrolments features meet the requirements specified above, it will also be evaluated on how well it meets the following criteria:

1. Performance - Whether it is quick to create an account or sign-up, and the responsiveness of the enrolment features.
2. Accessibility - Can a wide array of users easily navigate account management areas and enrolment forms.

The software tool Google Lighthouse will be used to evaluate these metrics. Google lighthouse uses automated tests to asses the response time of the page, and then analyses the code executed to build the page to find what may be causing a page to take a long time to respond and render. It also uses this same analysis to determine whether best accessibility practices have been used such as using appropriate alt-tags and aria labels where possible, and ensuring that components have proper contrast.

3.2 Topic Tree

3.2.1 Overview

Most learning management systems do not have a simple and easy method to import course material and resources from other courses. Some LMS platforms such as OpenLearning have no method of importing any data at all, and other platforms such as Canvas allows you to import crowd sourced material into your own course, however still does not allow you to import topics of resources.

This new LMS has a new topic tree feature, which will allow teachers and academics to add course material under a specific topic instead.

Topics
A topic is a collection of educational material that is related to a particular subject. For example, in UNSW's Introduction to Programming course (COMP1511), one of the

topics in the course is "Pointers", which contains all course material related to pointers.

At Charles Sturt University, each topic is defined as a small subject that is three hours in length [Uni]. CSU's curriculum covers around 1000 topics with each topic being three hours in length for a degree. In this thesis, we will also define a topic being three hours of content, but instructors can choose what amount of content constitutes as a topic.

Topic Groups
Topic groups are the same as courses themselves. We found that it would be much easier to enrol students into a group of topics instead of each individual topic, and it makes searching the topic tree much easier as there are a smaller number of topic groups than topics themselves.

Topic groups themselves do not contain assessments, exams, or any other resources. Only topics themselves contain this information. Therefore, academics must store assessments in individual topics and set up the prerequisites so if they wish, they can store a final exam in the last topic that students must complete or in an empty "milestone" topic. We decided to name this grouping of topics as "topic groups" instead of courses due to how a traditional course contains assessments and other content for the entire course, and does not have prerequisites between the topics themselves. This naming helps differentiate the system from the traditional university model.

Prerequisites

Each topic can have its own prerequisites, as discussed briefly previously. Some topics must be completed in order to complete the current topic. Prerequisites can be across topic groups as well. For example, Linked Lists in the "Introduction to Programming" topic group must be completed in order to start Graphs in "Data Structures and Algorithms". This improves reusability, as topics in other topic groups can be set as prerequisites eg. Git can be set a prerequisite for "Web Front end Programming" instead.

In the above example, pointers, structs and discrete mathematics must be learned in order to learn graphs. Likewise, graphs and recursion is required to learn depth first search.

Reusability
The topic tree enables academics to easily reuse content - by setting specific topics, academics can reuse content instead of creating new content for the same topic in their own course. A good example of this is "Web Front end Programming" at UNSW -

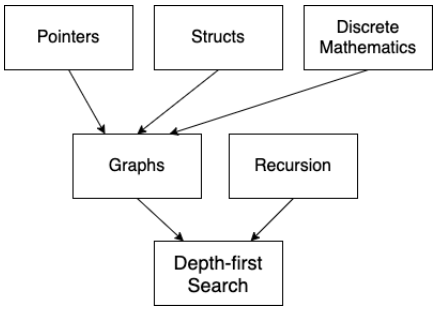

Figure 3.4: Example of a topic tree with Depth first search

instead of creating new content for Git, academics can set Git as a prerequisite. However, to encourage reusability, a cloning feature is also provided where academics can clone a topic and its resources from another topic group and put it in its own topic group. This further enables reusability, demonstrating that there is no need to spend time creating new content.

Course Materials
The Meta Learning Management System also organises content into four sections:

- Content;
- Practice;
- Preparation;
- and Assessments.

These various sections were chosen as they fit many learning models, including UNSW's learning models as listed on their website [UNS]. Students are introduced to it in the content section, and then get to know more about it in the practice section, and try it out in the preparation and assessments sections. After receiving feedback, this cycle repeats again.

3.2.2 Initial Designs

The above design is an initial design of the graph view of the topic tree. It was intended that there would only be topics when the topic tree was first designed, and the user could select a topic group to view the topic tree for that topic group. However, it was impossible to view prerequisites across topic groups with this design, and made it more difficult to view topics inside different topic groups. Therefore, a new design was chosen as follows.

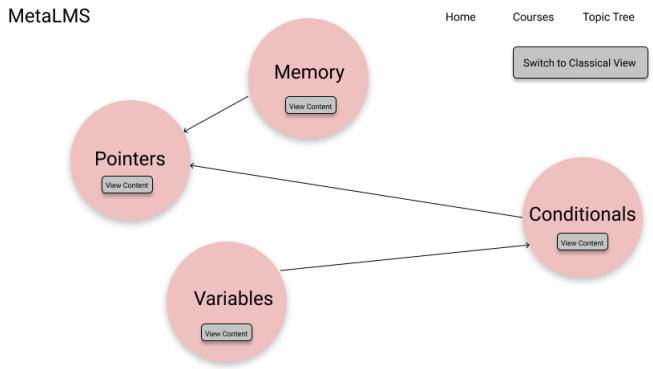

Figure 3.5: Topic Groups Graph UI

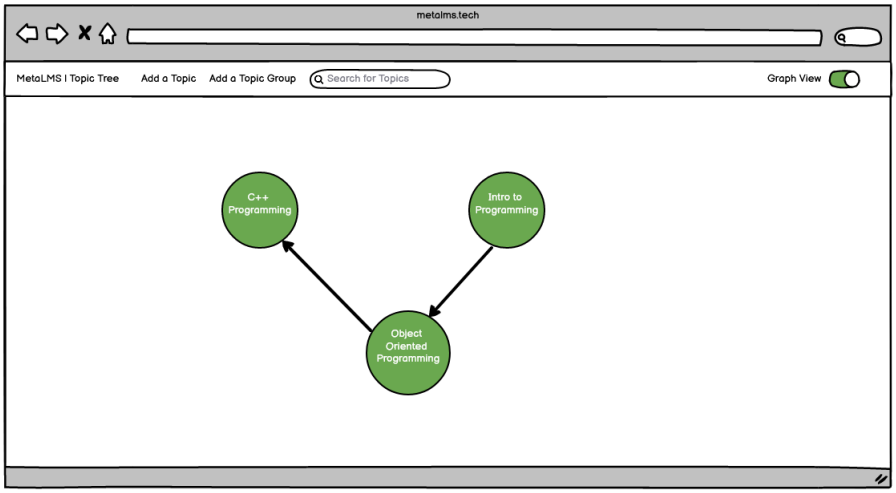

Figure 3.6: Topic Groups Graph UI Final Design

Instead, the topic groups are all shown on one graph, and a simple toggle to switch between the list view and the graph view is displayed in the top right. There are more options to add topic groups or topics at the top, and the user can click a topic group to show the individual topics as shown below.

As shown, when a topic group is clicked, a hull in the background surrounds the topics to indicate they are part of the same group. Topics are shown in blue, and the individual prerequisites are shown with prerequisites pointing to topics in other topic groups as well.

The final design of the list view shows both topics and topic groups on the same page to improve performance and usability, as there is less of a need to click through different pages to view different topic groups. Users can click on a specific topic group to expand and show the topics inside it, and click Add Topic to add a specific topic. Topics can be clicked to view their prerequisites.

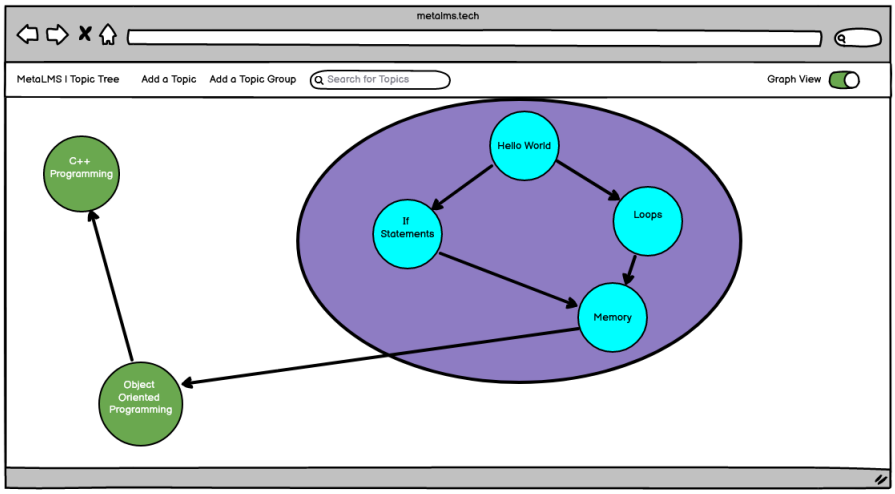

Figure 3.7: Topic Groups Graph UI Expanded Topics

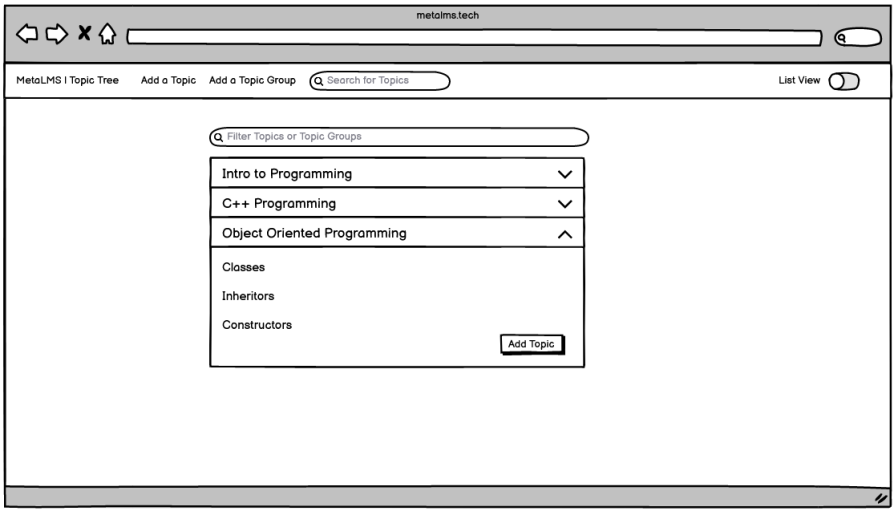

Figure 3.8: Topic Groups List UI Final Design

3.2.3 Requirements

Each requirement will have a priority: High, Medium or Low. High priority requirements will be completed first, and then medium and low priority requirements respectively.

3.2.4 Functional Requirements

Viewing Topics

As a	User
I want to	View groups of topics
So that	I can see how topic groups interact with each other
Priority	Medium
Status	Complete

As a	User
I want to	View topics within each topic group
So that	I can see how topic groups interact with each other
Priority	High
Status	Complete

As a	User
I want to	View topics within each topic group
So that	I can see how topic groups interact with each other
Priority	High
Status	Complete

As a	User
I want to	Search for specific resources
So that	I can find other resources easily in the many topics available
Priority	High
Status	Not Complete

As a	User
I want to	View topic prerequisites
So that	I can see which topics must be completed in order to complete the current topic
Priority	Medium
Status	Complete

As a	User
I want to	view a graph of topics and their prerequisites
So that	The topics and their prerequisites are easily viewable
Priority	High
Status	Complete

As a	User
I want to	view the topic tree with a traditional interface
So that	topics can be viewed more easily using a less confusing interface
Priority	Low
Status	Complete

Adding Topics and resources

As a	User
I want to	add a new topic
So that	I can edit the topic tree and add more content
Priority	High
Status	Complete

As a	Academic
I want to	upload course material
So that	I can add content to topics
Priority	High
Status	Complete

As a	Academic
I want to	add a new topic group
So that	topics are more organised and I can create new topics under the new group
Priority	Medium
Status	Complete

As a	Academic
I want to	Add tags to a topic
So that	Search for topics faster and more easily
Priority	Low
Status	Complete

As a	Academic
I want to	add a new topic group
So that	topics are more organised and I can create new topics under the new group
Priority	Medium
Status	Complete

Deleting Topics

As a	Academic
I want to	delete a topic that I've created
So that	I can remove content from the topic tree
Priority	High
Status	Complete

As a	Academic
I want to	delete a topic group that I've created
So that	I can remove content from the topic tree
Priority	Medium
Status	Not Complete

As a	Academic
I want to	remove course material from a topic
So that	I can remove content from the topic tree that is not needed
Priority	High
Status	Complete

Topic Prequisites

As a	Academic
I want to	add a topic prerequisite
So that	I can require other topics to be completed before students can learn the current topic
Priority	Medium
Status	Complete

As a	Academic
I want to	delete a topic prerequisite
So that	I can remove requirements for a topic
Priority	Medium
Status	Complete

Integration

As a	Academic
I want to	allow third party integration such as YouTube and The Box
So that	I can use external services with the topic tree
Priority	Low
Status	Not Complete

Exporting

As a	Academic
I want to	Export data from topics and course material
So that	I can use the topic tree data with other services
Priority	Low
Status	Not Complete

3.2.5 Non Functional requirements

The feature also has non functional requirements in order to better evaluate the completion of the feature.

Performance
Performance must be high and the feature must be responsive to provide a high quality

user experience for both academics and students. Performance will be assessed using tools such as Google Lighthouse [Gooa]. Google Lighthouse uses several metrics to assess performance, as explained previously.

Accessibility
The topic tree feature will also be assessed on how accessible the interface is. If features such as alt tags, high contrast colours and keyboard navigation are not implemented then not all academics and students will be able to use the topic tree feature. Therefore, tools such as Google Lighthouse will be used to access its accessibility as the tool also includes metrics for measuring accessibility of a website [web].

Usability
Finally, the feature will be assessed on its usability. This includes whether the topic tree feature is attractive and easy to use. Various students and academics will be surveyed on how easy they found the system to use. The number of errors and time taken to complete a task will also be assessed.

3.2.6 Changes from initial requirements

The initial requirements did not mention anything about third party integration such as integrating YouTube and The Box. Later, it was realised that third party services would be very useful in uploading files and managing content, and so the requirement was added.

Originally, the requirements also stated that there should be functionality to upload course material by selecting multiple topics. This did not make sense in the context of the topic tree, as course material is attached to specific topics, not topic groups, and so was removed from the final requirements.

Finally, tags were added to the final requirements. The searchability of the topic tree proved a difficult problem to solve, and by adding tags to topics, users would be able to search specific topics by their tags instead of by name. This was then added to the list of requirements as it greatly improved the searchability of the topic tree.

3.3 Course Pages

3.3.1 Overview

The course pages are a crucial component of any learning management system. It is the place where students and academics mainly interact with the LMS and acts as the centralised page in which other components are linked, for example the lectures, tutorials and quizzes. The course pages also includes the course dashboard, course content and course selection pages. The course dashboard page is the home page for a course, it displays important announcements to notify students of important information. The course content is the page where slides, tutorial questions and notes are found and the course selection page is where users have an overview of all of the courses they are enrolled in and provides them a way to navigate the MetaLMS. These pages offer other bonus functionality such as widgets which will be discussed in further sections.

3.3.2 Pages

The main 3 pages that the course pages will encompass are:

- The course dashboard
- The course content page
- The course selection page

Course Dashboard page

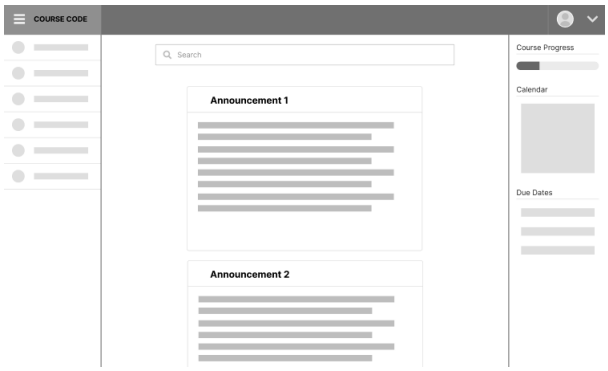

Figure 3.9: The main dashboard/page within the course.

The course dashboard is the centralised aspect of the course pages. This page provides users with somewhat of a default home page for a particular course. Within this page

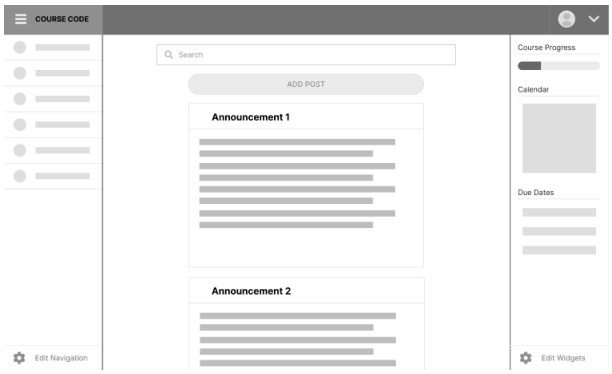

Figure 3.10: The administrator view of the main dashboard/page.

there will be a main dashboard/feed, a sidebar which links users to different aspects of the course and customisable widgets which provide further functionality.

The main dashboard acts as the display of all announcements by administrator and academics. Administrators and academics are able to create announcements in which all users in the corresponding course can view. Announcements are crucial to a course as they provide an efficient method of communication between administrators, academics and students/users. Users creating announcements will be able to attach files and links to better improve the usability of the planned LMS. Within each announcement, users will also be able to create comments which are linked to the forums.

The sidebar for the dashboard page is a component within the page that provides the user with the ability to navigate through other aspects of the LMS. Currently the components that will be linked in the sidebar are:

- Course Content;
- Course Forum;
- Lectures;
- Tutorials.

The final component of the main page is the widgets bar. These widgets offer more extensive usability and utility for users of the LMS. Some features that they offer are a course progress tracker and a calendar for users to view course due dates.

Course Content Page

The course content page is the area in which content is uploaded by administrators or academics and downloaded, viewed or completed by students. The course content page will contain content from the topic tree and will be catergorised accordingly. One

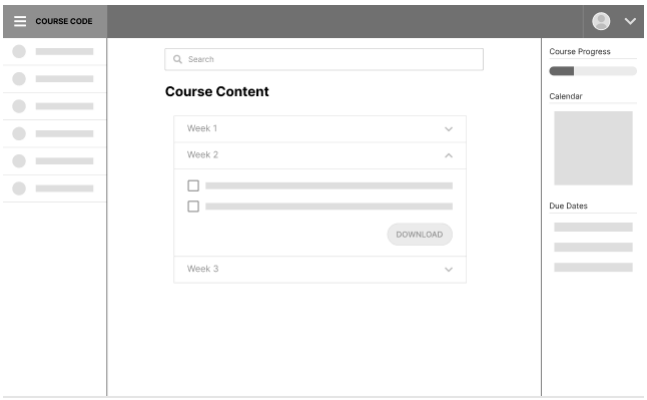

Figure 3.11: The course content page within the course.

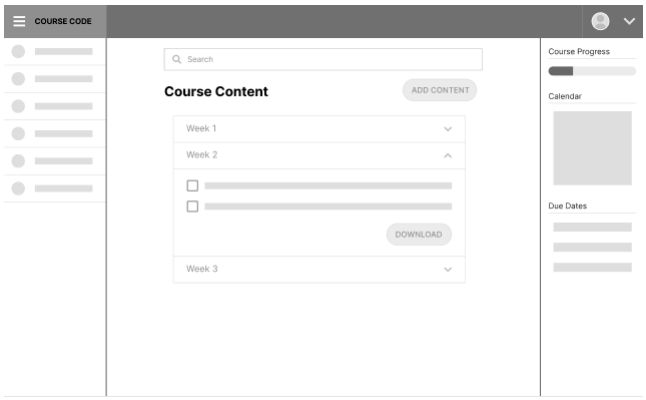

Figure 3.12: The administrator view of the course content page.

assumption made for the topic tree component is that the topic tree will be utilised as a database which stores content that can be utilised in courses. Course administrators would have the ability to select which content to import into the course content page. The topic content will be categorised into 4 distinct types (preparation, content, practice, assessments). Course content involves the topics that will be stored within the topic tree. This can range from lecture slides, tutorial questions, lecture recordings an even quizzes and assignments.

Course Selection page

The course selection page is utilised as the central hub for users to have an overview of all of their enrolled courses. It acts as the area of the MetaLMS to navigate to the topic tree, enrolments, courses and other features. It also contains other features to provide utility for users to better their learning.

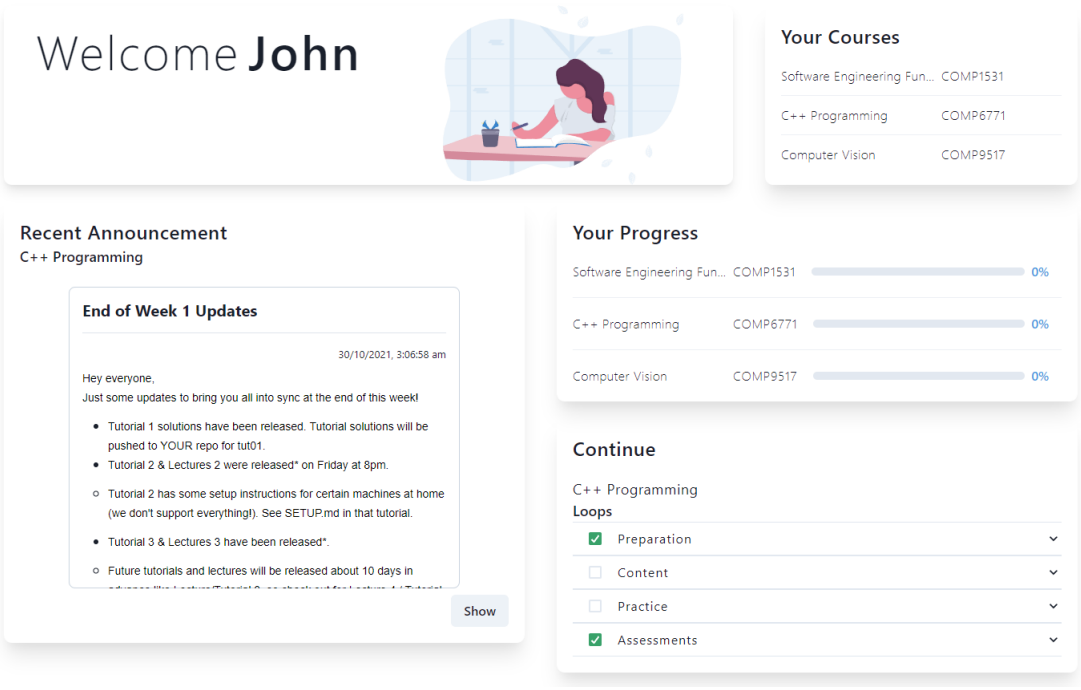

Figure 3.13: Student view of the course selection page

3.3.3 Additional Features and Refinements

The main changes that were implemented in Thesis B and Thesis C were the addition of the course selection page and the removal of the course outline page. The course outline page was deemed low priority as it would not encompass high priority use cases and the course selection page was more deemed more crucial. The previous plan was to develop only the course outline page, course content page and the course dashboard page. Without the course selection page, a user would not have the ability to transition to different courses. However with further development, the course outline page can be implemented.

3.3.4 Requirements

The requirements and use cases for the course pages are outlined below. These features are prioritiesed using the MoSCoW method which assists with identifying the order to implement requirements.

Functional Requirements

Course Selection Page

As a	User
I want to	View my enrolled courses
So that	I can see what classes I am enrolled in
Priority	High
Status	Complete

As a	User
I want to	Select a course I wish to navigate to
So that	I can navigate to the course pages of the corresponding course and begin studying
Priority	High
Status	Complete

As a	User
I want to	View my progress for an enrolled course
So that	I can determine any courses that I may need to complete
Priority	Low
Status	Complete

As a	User
I want to	View the most recent announcements from my enrolled courses
So that	I quickly see the most recent notification and be alert to any changes
Priority	Medium
Status	Complete

As a	User
I want to	Be able to access my most recently accessed topic
So that	I can quickly continue my studies without needing to navigate to the course page
Priority	Low
Status	Complete

As an	Administrator
I want to	Be able to access the forums and enrollements of a course
So that	I can quickly access these features without needing extra navigation
Priority	Low
Status	Complete

Course Dashboard Page

As a	User
I want to	View the announcements of a course
So that	I can see any important notifications for a particular course
Priority	High
Status	Complete

As an	Administrator
I want to	Be able to create announcements
So that	I can notify enrolled students of important information
Priority	High
Status	Complete

As an	Administrator
I want to	Be able to edit announcements
So that	I can change an announcement if any information is out of date or has changed
Priority	Medium
Status	Complete

As a	User
I want to	Be able comment on an announcement
So that	I can ask for clarification or ask a question about the announcement
Priority	Medium
Status	Complete

As a	User
I want to	Be able filter announcements
So that	I can search for a specific announcement for information
Priority	Medium
Status	Complete

Course Content Page

As a	User
I want to	Be able to view topics and their corresponding topic files
So that	I can view what content this topic group contains
Priority	High
Status	Complete

As a	User
I want to	Be able to view and download topic files
So that	I can save the content and complete it in my own time
Priority	High
Status	Complete

As a	User
I want to	Be able convey I have completed a topic file
So that	I can determine what content I have not completed
Priority	Medium
Status	Complete

As a	User
I want to	Be able to view the prerequisites of a topic
So that	I can determine what content I will need to complete first before continuing
Priority	Medium
Status	Complete

As a	User
I want to	Be able to filter topics
So that	I can search for a particular topic I may want to complete
Priority	Medium
Status	Complete

As a	User
I want to	Be able to filter topics
So that	I can search for a particular topic I may want to complete
Priority	Medium
Status	Complete

As an	Administrator
I want to	Be able to add topics or topic files to the content page
So that	I can easily modify the content of the course
Priority	Medium
Status	Not Complete

Sidebar

As a	User
I want to	Be able to click on a link and navigate to the corresponding component
So that	I can navigate within the MetaLMS
Priority	High
Status	Complete

As an	Administrator
I want to	Be able to customise the links on the sidebar
So that	I can create or delete certain components within the MetaLMS for a course
Priority	Low
Status	Not Complete

Widgets bar

As a	User
I want to	Be able to view reminders on a calendar
So that	I can easily determine the number of days before an important date
Priority	Medium
Status	Complete

As a	User
I want to	Be able to create reminders on the calendar
So that	I can not down important dates so that I will not forget
Priority	Medium
Status	Complete

As a	User
I want to	Be able to delete reminders on the calendar
So that	I can remove unnecessary reminders
Priority	Medium
Status	Complete

As a	User
I want to	Be able to edit reminders on the calendar
So that	I can update the information of a reminder in case the important event has changed
Priority	Low
Status	Not Complete

As a	User
I want to	Be able to view the progress of a particular course
So that	I can easily see the amount of content I still need to complete
Priority	Medium
Status	Not Complete

As a	User
I want to	Be able to customise what widgets are present on the widgets bar
So that	I can determine what widgets are on my widgets bar
Priority	Low
Status	Not Complete

Course Outline Page

As a	User
I want to	Be able to view the course outline of a particular course
So that	I can determine the learning outcomes of a course
Priority	Medium
Status	Not Complete

As an	Administrator
I want to	Be able to upload a course outline to a particular course
So that	I can provide learning outcomes for a course
Priority	Medium
Status	Not Complete

3.3.5 Evaluation

The evaluation of the course pages will depend on a set criteria. This criteria is proposed as followed:

- Performance - Whether the course pages are fast and responsive;
- Accessibility - Can a wide array of users use the course pages easily;
- UI/UX - Is the feature easy to use and attractive; and,
- Errors - Is the feature bug and error free.

3.4 Lectures and Tutorials

3.4.1 Overview

As referenced in the researched learning management systems, most LMS' utilise other software to implement lecture and tutorial features. For instance, UNSW Moodle uses echo360 for lectures and Zoom or Teams for tutorials. This feature will be lightly influenced by Canvas' integrated conference feature and therefore will attempt to eliminate the need for third party software to host tutorials and lectures.

In the proposed LMS, there will be an integrated tutorial and lecture system which will allow teachers and students to host and join their classes without the need to be redirected to another third-party software. Additionally, there will also be the added functionality of downloadable or watchable recorded lectures and tutorials that students will be able to utilise to revise their learning.

The tutorials and lectures content will be accessible through the relevant course homepage under their respective sections. LMS academics will be able to host lecture and tutorial sessions for the course and there will be notifications to enrolled users. Moreover, each student enrolled in the course will be enrolled in their chosen Tutorial and Lecture classes.

Overall, the successful integration of lecture and tutorial materials to the proposed LMS will eliminate the need for third party software's and subsequently will shorten the time taken to access lecture and tutorial content.

3.4.2 Design

The design of the lecture and tutorials will be split into two parts. Both designs will feature recording of the relevant sessions. The lecture design will be covered in the following subsection.

The design of the lecture feature will be a subsection added to the relevant course homepage. This subsection will be expandable and will display a list of lectures and their attached titles. Additionally, there will be an icon that allows users to download the recorded session onto their local devices.

There will be an icon or text that will allow users to determine if it is live or completed. Students will be able to click on the relevant lecture row to view the content live or recorded. Additionally, there will be a feature to allow the lecture to be moved on the LMS screen (Video Pop Out). This feature will support users multitasking while watching their lecture videos. Course academics can click this row to start the relevant lecture and stream their classes.

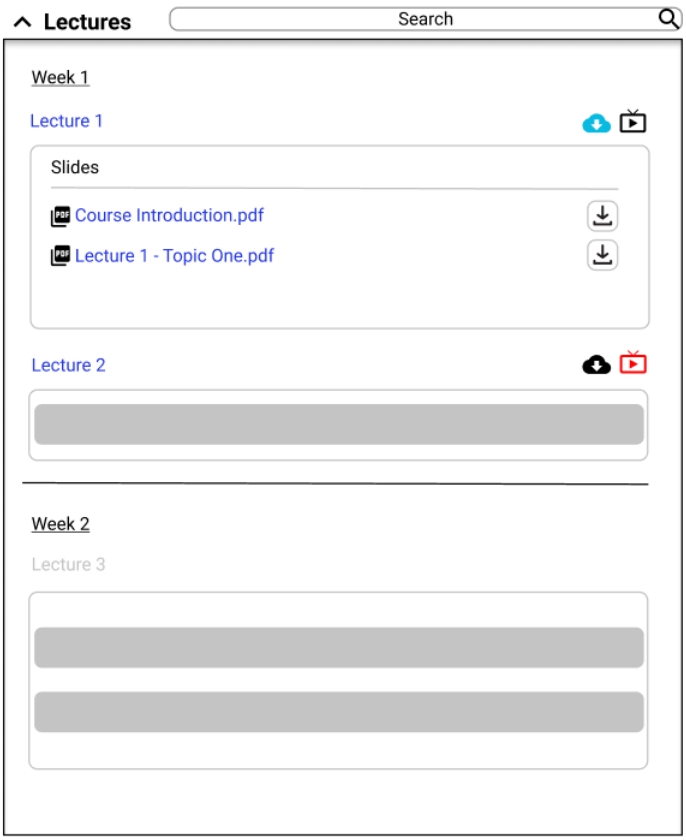

Figure 3.14: Lecture User Interface

The user interface for lecture video streaming will take inspiration from Canvas and will include white-boarding tools, a live chat, and a private messaging feature. There will be options to hide these tools and features to assist in user experience and accessibility. The lecturer will also be able to mirror their screen in the lecture. Also, the videos will also offer subtitles to assist users who have hearing impairments.

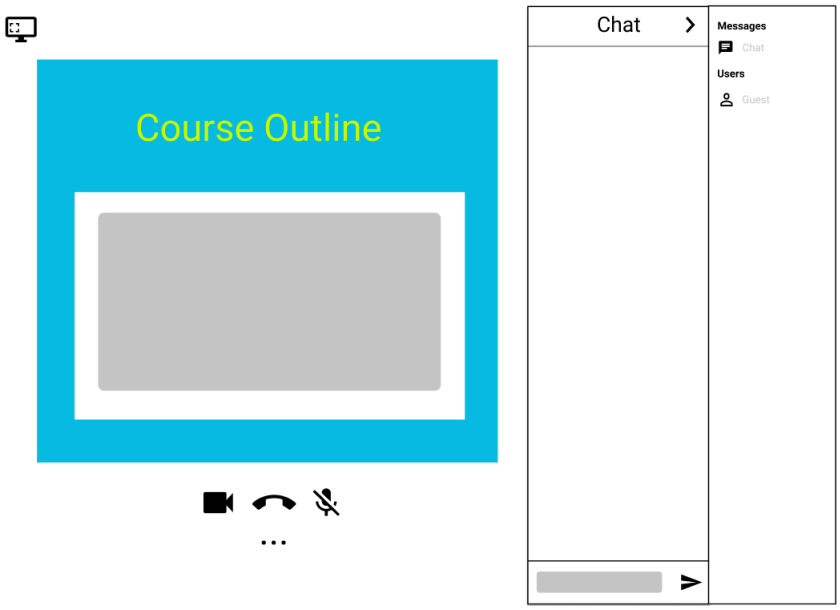

Figure 3.15: Video Streaming Features

The tutorial video streaming design will be similar to the proposed design for the lecture videos. There will be the enabled addition of screen sharing between users present. This will be added to a toolbar during the tutorial session. The tutor will be able to check off attendance with a feature that automatically marks students off at an adjustable timeframe. For instance, the tutor may wish to consider attendance as 15 minutes in the tutorial class.

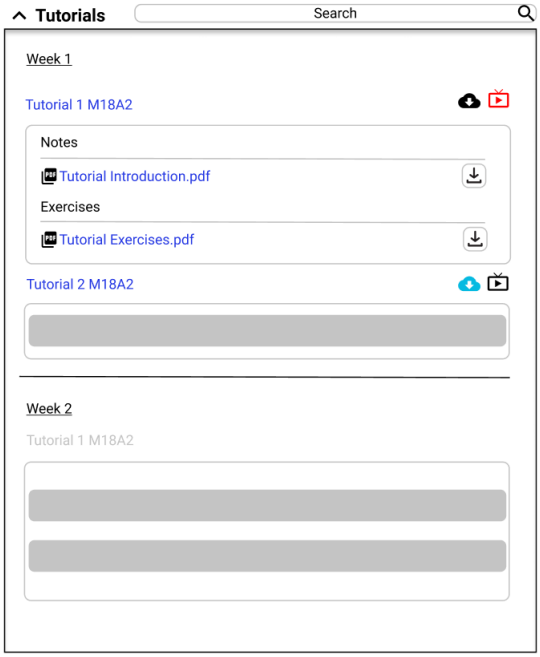

Figure 3.16: Tutorial User Interface

3.4.3 Functional Requirements

Lectures and Tutorials

1. Users can toggle lectures and tutorials subsection visibility

Video Streaming

1. Users can watch a live lecture
2. Users can watch a live tutorial
3. Instructors can stream a live lecture
4. Instructors can stream a live tutorial

File System

1. Users can search for a file
2. Instructors can upload files to tutorials subsection
3. Instructors can upload files to lectures subsection

4. Users can download a recorded lecture
5. Users can download a recorded tutorial
6. Users can download tutorial files
7. Users can download file files
8. Users can view tutorial files
9. Users can view lecture files

3.4.4 Non-Functional Requirements

1. Usability - The feature must be efficient and effective
2. Performance - The feature must operate within a reasonable timeframe
3. Aesthetics - The feature must be visually pleasing
4. Learnabilty - The feature must be easy to learn and use
5. Accessibility - The feature must accomodate varying users

3.4.5 Evaluating Results

There will be an evaluation criterion which will be used to determine the success of the lectures and tutorials features in satisfying requirements and purpose.

1. Aesthetics: Is the feature visually pleasing to the user
2. Accessibility: Does the feature accommodate users with varying backgrounds (disabilities, children, etc)
3. Performance: Does the feature run within a reasonable timeframe
4. Usability/Learnability: Are the feature easy to learn and use

3.5 Assessments

3.5.1 Overview

The Assessment feature consists of a quiz and poll system where the poll can be used as a supplementary tool for the quiz or to be used as a general poll system. Quizzes are a widely used form of assessing students' knowledge about a particular topic or

course. It is beneficial to students as it gives a good indication of their understanding of a topic and whether they need to revise more in a particular area. Likewise, it is beneficial to course lecturers as they'll be able to gauge the students' understanding and whether they should change their teaching methods or revise the content with students to resolve difficulties with concepts. Therefore it is an essential in any learning management system to provide quizzes, which most, if not all, learning management systems do. However, some existing quiz systems lack in providing an aesthetic user interface and useful feedback for both parties. Therefore, the Assessment feature aims at:

- allowing quiz submissions and grading to be performed on the same platform for convenience
- providing additional useful statistics as feedback to help improve the teachings of a course instructor and the learnings of a student enrolled in the course
- having an appealing, readable interface for both course instructors and students
- providing a poll system for general poll use or as a means for students to communicate what topic they'd like the course lecturer to revise

The core features for quizzes include being able to:

- create/remove/edit a question
- answer/edit an answer to a question
- choose between three types of questions to create:
  - multiple choice - select one answer only
  - short answer
  - checkboxes - select none, one or multiple answers
- auto-save and manual save answers during quiz attempt
- timer that runs during quiz attempt

The core features for polls include being able to:

- create/remove a poll
- create/remove/edit a poll option
- select poll options/s to vote and change your vote
- choose between two types of polls to create:
  - restricted - can only vote for 1 option
  - open - can vote for multiple options

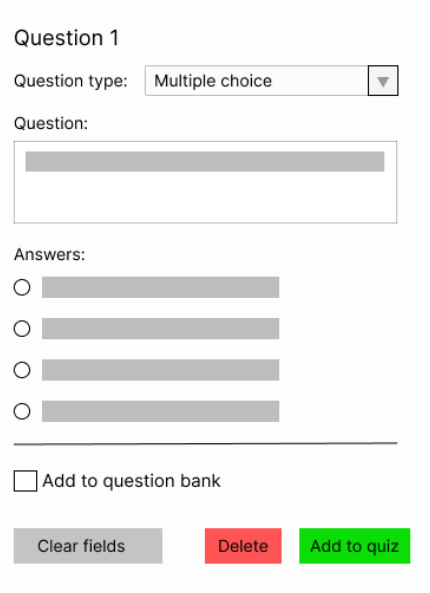

Figure 3.17: Quiz creation (Instructor's view)

Figure 3.18: Quiz usage (Student's view)

3.5.2 Stakeholders

- Course lecturers
- Admins
- Students

3.5.3 Functional Requirements

The list of functional requirements' priority level will be determined using the MoSCoW method to clearly outline what needs to be implemented throughout the thesis and what the minimum viable product should have.

The MoSCoW method splits requirements based on 4 categories:

- Must Have
- Should Have
- Could Have
- Won't Have

A requirements survey was conducted and completed by me and other Thesis A students to gauge what was the priorities for each requirement. The results for the Assessment feature from the survey and my own ideas were combined to form the following functional requirements:

Quiz creation

1. Course lecturers can create a quiz [Must Have]
2. Course lecturers can add questions to a quiz [Must Have]
3. Course lecturers can create different types of quiz questions (multiple-choice, short answer, checkboxes) [Must Have]
4. Course lecturers can set up a timer or due date for a quiz [Must Have]
5. Course lecturers can add "drag and drop" type questions [Could Have]
6. Course lecturers can add "connect the pairs" type questions [Could Have]
7. Course lecturers can add media (audio or video) into a question as the entire question or as a supplementary material to the question [Could Have]
8. Course lecturers can create a question bank [Should Have]
9. Course lecturers can add a question to a question bank [Should Have]
10. Course lecturers can import a question from a question bank [Should Have]

Quiz modification/removal

1. Course lecturers can edit/remove a quiz [Must Have]
2. Course lecturers can edit/remove a question [Must Have]
3. Course lecturers can remove a question bank [Should Have]
4. Course lecturers can remove a question from a question bank [Should Have]

Quiz usage

1. Students can answer a quiz question [Must Have]
2. Students can edit any of their answers during the quiz [Must Have]
3. Students can submit a quiz attempt [Must Have]
4. Students can manually save their progress at any time [Should Have]

Poll creation

1. Course lecturers can create different types of polls (can vote either 1 option only or multiple options) [Must Have]
2. Course lecturers can add an option to a poll [Must Have]

Poll modification/closing

1. Course lecturers can edit/remove options [Must Have]
2. Course lecturers can close polls [Must Have]
3. Course lecturers can set a poll closing date [Should Have]
4. Course lecturers can close a poll but make results viewable [Could Have]

Poll usage

1. Students can vote for one or multiple options (depending on poll type) [Must Have]
2. Students can add an option (if course lecturer enables the option for it) [Could Have]

Viewing quiz results and feedback

1. Students can view results against their answers [Should Have]
2. Students, course lecturers and admins can see how many students selected each answer for a question [Should Have]
3. Students can view topic or lecture the question derives from [Should Have]
4. Students can view an explanation of the correct answer if the question is answered incorrectly [Could Have]

Re-usability

1. Course lecturers can add a question to a general question bank [Should Have]
2. Course lecturers can import/export quizzes [Should Have]
3. Course lecturers can import/export questions [Should Have]

3.5.4 Non-functional Requirements

The non-functional requirements used is heavily inspired by the Jakob Nielsen's 10 usability heuristics that's used to evaluate the usability of user interfaces.

1. **Efficiency** - users are able to perform tasks without taking too many steps
2. **Learnability** - new and returning users are able to quickly learn how to use and interact with the feature on the go
3. **Performance** - operations within the feature are completed within a timely manner, resulting in a smooth process
4. **Consistency** - components, colour themes and formats are consistent across pages
5. **Aesthetic and minimalist design** - the user interface is appealing and simple while providing the expected features
6. **Re-usability** - users are able to re-use components and content they've previously made

Technologies Used

The technologies that will be used are:

- Back-end runtime environment: Node.js
- Web application framework: Express.js
- Database management system: PostgreSQL

Database

The use of a relational database will be ideal as it'll be more organised and querying for particular data related to each other will be easier during the implementation of the Assessment feature.

Potential tables involved would include a Question, Poll, Quiz and User table.

Technologies Used

The technologies that will be used are:

- Web application framework: React
- Component library for React: Material UI

Feature Breakdown

Since the React frameworks structures content into re-usable, isolated components, the Assessment feature will need to be broken down into such components. Below are the main components that will likely be implemented during Term 2:

- Question
- Quiz
- Poll Option
- Poll
- Timer
- Question Bank
- Quiz Results
- Answer Explanation

3.5.5 Evaluation

The system will be evaluated based on:

- **Functional requirements** - what was completed, with the satisfactory being all "Must Haves" and some "Should Haves"
- **Non-functional requirements** - whether it satisfies the goals of the feature and performs operations in a timely manner
- **Usability tests** - if users can navigate through the feature and be able to complete operations within the feature

3.6 Forums

3.6.1 Overview

Forums are a common feature in learning management systems as it provides a community environment where students can ask questions and discuss content with educators and other students. Many of these built-in forums, however, are very basic and lack sufficient search, sorting and filtering functionality. In many cases, educators are turning to external, third-party forums in order to take advantage of the more advanced features that they offer.

The aim of this feature is to develop a built-in forum that meets students' and educators' needs so that they no longer need to use an external application.

The forum interface consists of the overview page and the individual post pages.

Forum Overview Page

In general, the forum overview page consists of a list of posts, as well as search and filtering mechanisms. The collapsible filter menu allows users to filter the forum posts based on pre-defined tags. Forum posts are ordered such that the list of pinned posts are at the top, followed by the remaining posts in chronological order. Each row in the table includes the post title, date created, number of replies, number of comments and the associated tags.

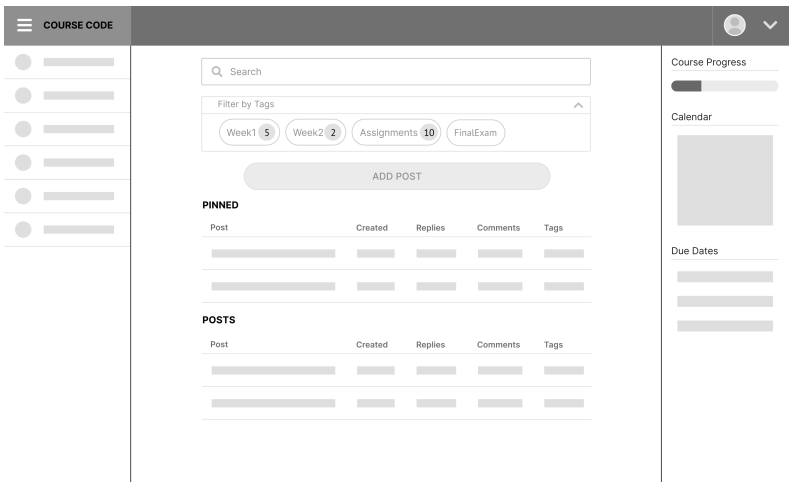

Figure 3.19: Forum overview page for a student.

Staff have an additional button that allows them to pin and unpin posts.

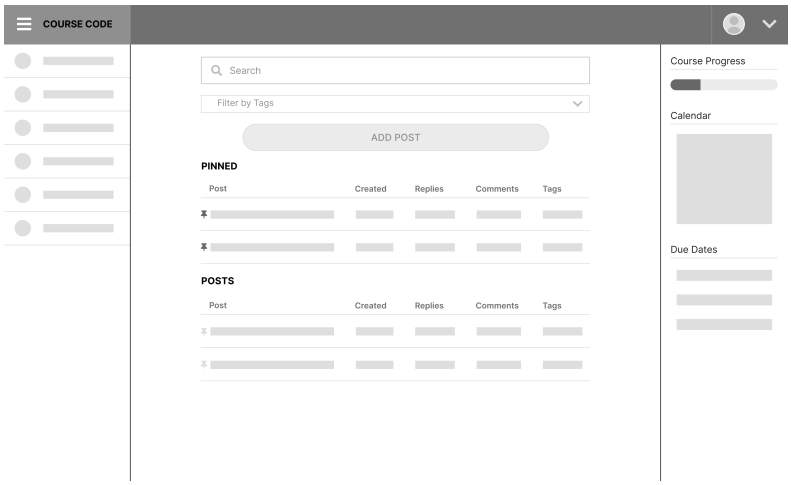

Figure 3.20: Forum overview page for an admin.

Post Page

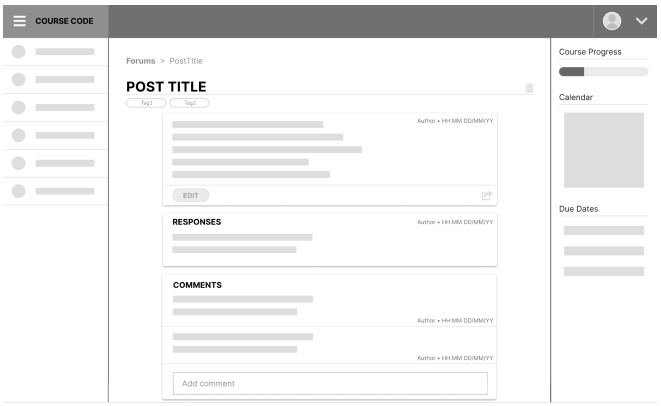

Figure 3.21: Forum post page for a student.

Each forum post has its own post page which contains the post details, responses and comments. Students are able to edit their posts from the post page if required. They can easily view the instructor's response, if any, in the responses section. The comments section allows the author to view and leave any additional comments. It also gives other students a place to write a response or ask questions based on that forum post.

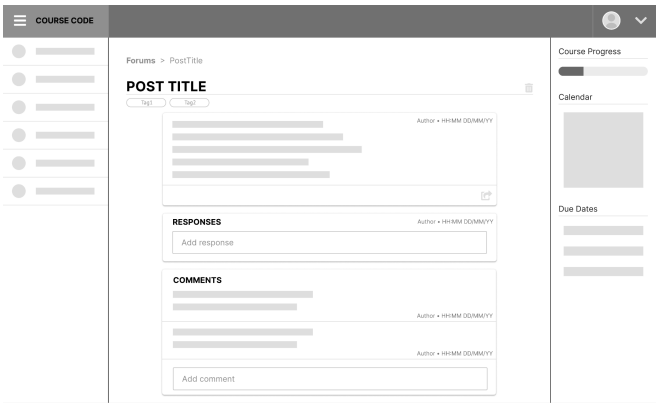

Figure 3.22: Forum post page for an admin.

If a forum post is unanswered, an admin is able to leave a response. The current idea is to restrict this so that only Staff can write responses to ensure that all responses are verified. It also ensures that forum posts aren't left unanswered if a student accidentally leaves a follow-up question in the responses area, instead of the comments section. This method of implementation may be reassessed if time allows.

3.6.2 Additional Features

As a result of reaching major milestones quicker than expected when building the forum component, additional features were able to be implemented to enhance the functionality and usefulness of the forums.

Sortable Table

To provide more methods to help users with finding specific posts, sorting functionality was added to the post tables on the forum overview page. Users can click on the headings of each column in the table to toggle the sort order based on that heading. For students, they could use this functionality to find the oldest or latest post in the forums. Staff could sort the posts by number of replies to find which posts require their attention. By default, the table is sorted in reverse chronological order based on the date created for each post.

Upvotes

An upvote button and count was added to each of the posts on the forum overview page, as well as each individual post page. Upvoting allows users to like a post to assist in bringing attention to it. Students could use this functionality to like a post that asks a question that they have also been thinking about. This could help in reducing the number of duplicate or similar posts asked to the forum. Staff could then sort the

posts on the forum overview page by number of upvotes to see which posts are popular amongst students at the moment and require their attention. Student could also use the upvote count to work out which posts are more important than others.

Endorsed Posts

In cases where students leave a comment on a forum post with the correct answer, it was decided that there should be a way for staff to be able to mark this answer as verified so that they don't have to also reply to the post. An endorse button was added to each comment on the post page for a staff so that they could use it to endorse a comment. This results in the comment being marked with a green check mark. The ability for staff to endorse a whole post was also added. This provides staff with another way to draw attention to important posts.

Enhanced Filter

While a filter was always going to be included in the implementation of these forums, it only allowed users to filter posts by the pre-defined tags for the course. Additional filter options were added to the overall filter to provide users with more ways to find specific posts. The options added allowed users to filter by posts linked to announcements, answered and unanswered posts, and endorsed posts.

3.6.3 Finalised Requirements

The following features are prioritised using the MoSCoW method which assists in identifying the order in which to implement the requirements [Pro]. It contains the following priorities

- **Must have** - vital features that are critical to the basic functionality of a project.
- **Should have** - important features that aren't critical but add to the basic functionality of a project.
- **Could have** - desired features that aren't necessary to the overall project but can provide a better user experience.
- **Won't have** - low-priority features that likely won't be able to be completed in the given time-frame.

Functional Requirements

1. Forum Overview Page

As a	User
I want to	View a list of forum posts
So that	I can see everything that has been posted to the forums.
Acceptance Criteria	
- Users can see all forum posts in a table (Must have) - Users can see all pinned posts in a separate table (Must have)	
Overall Priority	Must have

As a	User
I want to	Clearly see which posts have been read and actioned
So that	I can view posts that have an answer
Acceptance Criteria	
- Users can easily find answered posts (Should have) - Users can sort posts by number of replies or comments (Should have) - Users can see which posts have been endorsed (Could have)	
Overall Priority	Should have

2. Adding a post

As a	User
I want to	Add a post to the forum
So that	I can ask a question or make a comment related to the course
Acceptance Criteria	
- Users can add a post with a title and description (Must have) - Users can format their post description (Should have) - Users can optionally attach files (Should have) - Users can optionally attach tags (Must have) - Users can optionally attach links (Should have)	
Overall Priority	Must have

3. Search, filter and sort

As a	User
I want to	Be able to search for a forum post
So that	I can easily find the post I am looking for
Acceptance Criteria	
- Users can search a post's description or title (Must have)	
Overall Priority	Must have

As a	User
I want to	Be able to filter through forum posts
So that	I can easily find the post I am looking for
Acceptance Criteria	
- Users can filter forum posts using pre-defined tags (Must have) - Users can filter forum posts by those linked to announcements (Could have) - Users can filter forum posts by those that are answered (Should have) - Users can filter forum posts by those that are unanswered (Should have) - Users can filter forum posts by those that are endorsed (Should have)	
Overall Priority	Must have

As a	User
I want to	Be able to sort the forum posts
So that	I can easily find the post I am looking for
Acceptance Criteria	
- Users can sort forum posts by the different headings in the post table (Should have)	
Overall Priority	Should have

4. Forum Post Page

As a	User
I want to	View the individual post details for a chosen post
So that	I can see the post's description and all the comments and replies
Acceptance Criteria	
- Users can navigate to a post page for an individual post (Must have) - Users can easily see the post title, description, tags, author and date created (Must have) - Users can see all the responses to the post (Must have) - Users can see all the comments on the post (Must have)	
Overall Priority	Must have

As a	User
I want to	Share a forum posts with others
So that	Others don't miss out on important course information
Acceptance Criteria	
- Users can copy a link to a forum post (Must have) - Users can directly share a forum post to other platforms (Could have)	
Overall Priority	Must have

5. Upvotes

As a	User
I want to	Upvote a forum post
So that	I can bring attention to it
Acceptance Criteria	
- Users can upvote a post on the forum overview page (Should have) - Users can upvote a post on the forum post page (Should have) - Users can see the number of upvotes a post has (Should have)	
Overall Priority	Should have

6. Editing and Deleting Posts

As a	User
I want to	Edit a post I have added
So that	I can make any necessary changes
Acceptance Criteria	
- User can edit a post after it has been posted (Must have)	
Overall Priority	Must have

As a	User
I want to	Delete a post I have added
So that	I can remove it from the forums
Acceptance Criteria	
- User can delete a post after it has been posted (Must have)	
Overall Priority	Must have

7. Comments

As a	User
I want to	Comment on a post
So that	I can try to answer the question or leave a follow-up question
Acceptance Criteria	
- Users can leave a comment on a forum post (Must have) - Users can format their comment (Should have) - Users can edit their comment after it has been posted (Should have) - Users can delete their comment after it has been posted (Should have)	
Overall Priority	Must have

8. Staff - Manage Tags

As a	Staff User
I want to	Manage the tags for the topic group
So that	I can add new tags or remove any unnecessary tags
Acceptance Criteria	
- Staff can add new tags (Must have) - Staff can remove tags (Must have) - Staff are unable to create duplicate tags (Must have) - Students are unable to create tags (Must have)	
Overall Priority	Must have

9. Staff - Pin Posts

As a	Staff User
I want to	Pin posts to the top of the forums
So that	I can bring the post to the students' attention
Acceptance Criteria	
- Staff can pin posts to the top of the forums (Should have) - Staff can unpin posts to the top of the forums (Should have) - Students are unable to pin and unpin post (Should have)	
Overall Priority	Should have

10. Staff - Reply

As a	Staff User
I want to	Reply to a forum post
So that	I can answer the student's question
Acceptance Criteria	
- Staff can leave a reply on a forum post (Must have) - Staff can format their reply (Should have) - Staff can edit their reply after it has been posted (Should have) - Staff can delete their reply after it has been posted (Should have)	
Overall Priority	Must have

11. Staff - Endorse

As a	Staff User
I want to	Endorse a post or comment
So that	Students can see that I have verified the post or comment
Acceptance Criteria	
- Staff can endorse a post (Could have) - Staff can unendorse a post (Could have) - Staff can endorse a comment (Could have) - Staff can unendorse a comment (Could have)	
Overall Priority	Could have

12. Staff - Link Materials

As a	Staff User
I want to	Directly link or embed materials to the forum post
So that	I can reference materials when I reply to a question
Acceptance Criteria	
- Staff can add a direct link to a reply (Could have) - Staff can embed course materials in a reply (Could have)	
Overall Priority	Could have

13. Staff - Forum Collections

As a	Staff User
I want to	Curate forum questions into collections
So that	I can categorise questions and share them with students
Acceptance Criteria	
- Staff can create a collection of related forum posts (Won't have)	
Overall Priority	Won't have

Non-functional Requirements

1. **Usability** - the forums component is easy for staff and students to use. All the features that are implemented should be useful and enhance the functionality of the forums.

2. **Accessibility** - the forums should be accessible to all users, including those with disabilities.

3. **Performance** - the forums component should be able to complete requests and tasks in a reasonable period of time.

3.7 Gamification

3.7.1 Overview

Learning Management Systems (LMS) are often only used as tools to distribute course content in various forms and act as a communication tool between course staff and students. Many LMS do not provide features that actively engage, retain and influence students for a better academic outcome. Futhermore, many LMS do not have sufficient capability to ensure content is not lost, can be easily readapted or reused and provide academics a simple flow efficiently set up interactive content. This gamification feature will focus on addressing this lack of capability in the market to provide students and academics an immersive and engaging learning experience within a wider meta-lms where content can be easily reused, readapted and monitored.

Gamification of LMS can achieve this through an interactive and immersive approach by applying game-like mechanics, creating a fun learning experience for students within an integrated environment of the meta-lms where data is tightly connected to other parts of the lms (i.e. Course Page, Forums, Assessments). A Gamification feature must also link seamlessly into a wider meta-lms to ensure course content can be easily retrieved, reused and updated to provide a high-quality learning journey for students.

3.7.2 Introduction to LMS Gamification

Gamification of a Learning System is defined as implementing game thinking and game mechanics into core features of the platform to provide a more interactive learning experience for students. Gamification is built on three simple concepts: Objectives, Rewards and Competition. Objectives can be set by platform admins, aligning to specific course learning objectives. These objectives provide students with a purpose to remain on the platform and ultimately achieve course learning outcomes. A reward system is a positive reinforcement for students and can be a driving factor for students to stay on the platform. In a well designed gamification experience, competitive interest pushes students to perform better and to gain more knowledge to attain a higher position than others. Through these fairly simple concepts, the gamification feature in a learning management system can help students achieve course learning objectives in an immersive and rewarding manner.

3.7.3 Goal Of Gamification Feature

The goal of this feature is to form a crucial component in the meta-lms to provide a reusable, customisable and seamless experience for course admins as well as a fun, interactive and immersive learning experience for students using the platform. This feature aims to implement the 3 gamification concepts of Objectives, Rewards and Competition. The gamification feature will be structured into levels with Core Knowledge modules providing clear objectives for students, Challenge modules providing practical hands on practice that when combined with the reward system create a competitive environment for students. These modules will provide a level of customisation based on wider lms permissions to academics.

Academics can customise the questions and question types in each module, set the correct answers as well as set the achievement / rewards that students will earn on completion. The types of games in scope for development will be multiple-choice style (Timed and Untimed) games, True / False games and coding questions (Assessed on test suite). The feature will also enable academics to customise the levels and select the games they like to be included in their course run. The Gamification feature will be integrated into the larger meta-lms where the data is stored.

- High Levels of Engagement and Retention on Platform
- Present content with a fun and immersive approach
- Better retention and improved proficiency through practice
- Challenge and stimulate creativity
- Trigger change in thinking
- Influence positive behavioral change

3.7.4 Functional Requirements

Since the backend for the Meta LMS is being built out independently to the individual features, the following contains a description of the ideal backend design for the forum component.

Learning Structure

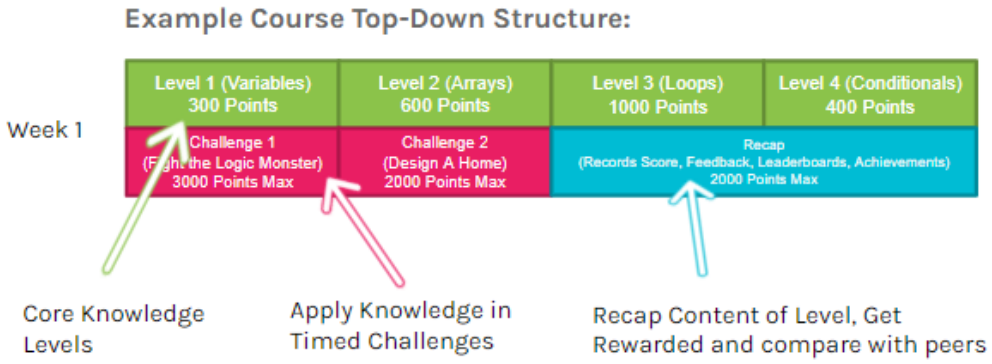

Figure 3.23: Learning Level Structure

The above outlines the learning structure for the gamification feature. This includes Core Knowledge modules that can be set as objectives for the week. Academics can customise the content and the number of these core knowledge modules which should be broken down by topics (E.g. Variables, Arrays, Loops). The Challenge module allows academics to create challenging practical exercises for students to apply their knowledge learnt in core knowledge modules. Academics can set the rewards and achievements that students will earn in each of the different modules. These rewards will allow the system to automatically produce leaderboards and in game prompts to drive a competitive environment.

Core Knowledge Modules

Core Knowledge modules aims to present the fundumental knownledge topics in video, slides or pdf form and followed up with some simple practice to reinforce the new learnings for students. This sub feature will have functionality to add video content, content from webpage as well as pdf content. After these content, several game types can be added (MCQ, True/False, Coding Questions) to test the knowledge students just learnt. Core Knowledge modules can be set as prerequisite for challenges and allow the students to earn reward points as well. Core Knowledge modules will be used to introduce specific topics through short videos followed by questions to test the newly gained knowledge.

Challenge Modules

Challenge modules aims to create interactive content that enables students to apply knowledge they learnt by providing an environment to develop their critical thinking skills. Challenges are more challenging timed games where students need to apply the core knowledge in various ways to earn points. This reinforces and simulates a review of the core knowledge. Challenges are timed levels with a higher difficulty level but provide students with the most rewards based on their performance (Time Taken and Accuracy Of Answers).

Reward System

The reward system aims to provide a mechanism for positive reinforcement, create a competitive environment and set some goals for students to achieve, ultimately increasing engagement and retention of students. In Core Knowledge and Challenge modules, students can earn points based on their accuracy in answering questions under time pressure. The number of points earned per activity can be customised by course admin. These points will be recorded for modules, weeks and all time to generate leaderboards. Based on points earned and certain criteria, the platform can issue an achievement that is unique and only be earned once per course. These achievements can be customised by the course admin. Students will be able to use the points earned to redeem items that will help them earn more points. Some examples: x2 points for 1 hour.

Game Types

Game Types facilitate the assessment of student's knowledege in an interactive manner. These games should be extremely reusable and customisable. In scope are Timed or Untimed multiple choice questions, True / False questions as well as coding questions (through integrated text editor). The game type structures pull data from the meta-lms database that are related to their specific data structure. Course admins will be able to create, customise and choose reward parameters for each game type. This feature will be implemented to be easily modified to include more game types in the future.

Student Frontend Requirements

1. As a Student, I need to be able to access the gamification feature to navigate and complete levels for my topic group. (Permissions)
2. As a Student, I need to be able to complete Core Knowledge Modules.
3. As a Student, I need to be able to complete Challenge Modules.
4. As a Student, I need to be able to see a recap of my progress in level.

5. As a Student, I need to be able to view all my enrolled topic groups and access weekly activities.
6. As a Student, I need to be able to monitor my progress compared to other students.
7. As a Student, I need to be able to see my achievements and reward points. (Rewards)
8. As a Student, I need to be able to see my leaderboard rankings compared to my peers. (Rewards)

Academic Frontend Requirements

1. As an Academic, I need to be able to access the gamification feature to navigate, customise levels and set permissions for students. (Permissions)
2. As an Academic, I need to be able to create a new topic group run either new or clone from previous topic groups.
3. As an Academic, I need to be able to customise content and number of questions for Core Knowledge Modules and Challenge Modules.
4. As an Academic, I need to be able to create new timed questions of types (MCQ, True / False, Coding).
5. As an Academic, I need to be able to monitor the leaderboards for the course and monitor course level statistics.
6. As an Academic, I need to be able to customise the rewards awarded to students on completion of questions in modules. (Rewards Management)

Backend Requirements

Gamification Feature will require some backend connections to the meta-lms to setup for integration of permissions, content and results. This feature will also need backend functions that will enable setup and customisation of courses, modules and student profiles.

1. Retrieve a list of students enrolled in course from metalms
2. Retrieve a list of all or topic groups in metalms
3. Retrieve permissions for user
4. Play through level
5. Store topic groups
6. Store level
7. Update level
8. Update user points

3.7.5 Software Development Approach

In this new gamification feature for the meta-lms, we will be using React as our frontend architecture. It is the most modern and widely used framework that will ensure the longevity of this feature. While React is used as a frontend framework, node js is used as a middle layer for data manipulation and api creation. Node js is a simple framework based on js that allows for quick development of REST APIs that are hosted on the web server. The data manipulation layer will interact with the PostgreSQL backend which will hold all the data for the meta-lms. Gamification feature will have its own suite of REST APIs developed that are specific to implementation of the 3 concepts.

1. Data Structure for Gamification will first be set up in the PostgreSQL. These include the SQL tables required and a UML diagram to outline how each table will interact with one another.
2. REST APIs created to facilitate interactions between frontend and backend.
3. Frontend Development
4. Testing of Gamification Feature
5. Integration to meta-lms

3.7.6 Evaluation

Before implementation, we need to understand our users (Academics and Students). In Term 2, we will provide several students and academics with a questionnaire surveying knowledge about gamification as well as popular features and use cases in a gamified LMS. Using the survey, we can test and refine our initial user stories to ensure we meet the end user's needs.

To evaluate the quality of the implementation, the gamification feature must be integrated into the wider meta-lms and pass various feature specific automated tests as well as meta-lms automated tests. These tests will ensure that the feature is performing to its specifications and that this remains through when operating it as an integrated feature in the meta-lms.

On completion of a working prototype of the meta-lms gamification feature, we will conduct user testing that will help to refine and iron out issues with the platform. The user testing will include hands-on interaction with the prototype running through various use cases on the gamified lms. Upon completion we will utilise the feedback from user testing to refine the platform. Success of the thesis will determine completion of the gamified LMS feature with positive user feedback for the platform.

3.8 Backend

3.8.1 Overview

The backend of the meta learning system is arguably one of the most important components, as it is responsible for storing and organising data. Additionally, this component will be appropriately implemented to optimise the LMS runtimes.

The backend component will have no user interface and will only include the database, data, endpoints and documentation. In addition, the backend component will list all the endpoints via documentation. This component will only be accessible to developers and admins through a combination of both backend files and a command terminal.

Overall, the successful implementation of the backend component of the LMS will constitute the foundation of the LMS and ensure effective runtimes for the entire LMS system.

3.8.2 Design

There will be many database schemas which will be implemented in the backend. These schemas will be dependant on the requirements for each feature. For example, the schema for the lectures and tutorials feature will include relations between lectures and lecturers.

Moreover, an example of a table structure for the lectures and tutorials will include: the lecture id, the lecture title, the lecture associated files, the lecturers in charge and the enrolled students in the lecture.

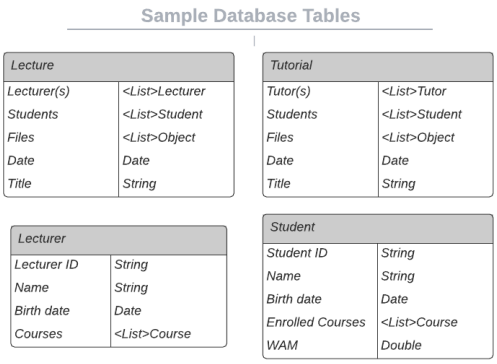

Figure 3.24: Sample Database Tables

The calls that can be made to the backend will include data retrieval, updates, creation and deleting. An example of such a creation call is creating a lecture entry. Moreover,

the calls samples will be further updated in the future to include a variety of calls for different schemas. For instance, a GET request for students enrolled in a course.

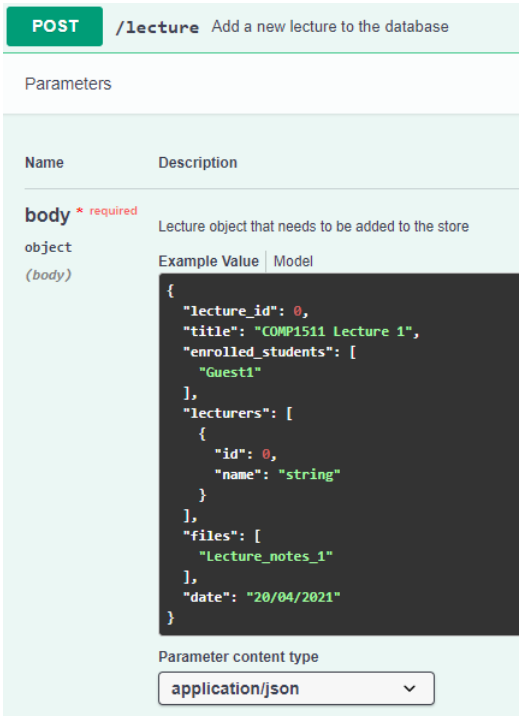

Figure 3.25: Lecture POST Request

3.8.3 Requirements

The requirements listed below will be in the general form as specific API calls for each component have not been completely finalised. Moreover, the requirements for the backend component will be adjusted and changed in accordance with the needs of the other components.

3.8.4 Functional Requirements

API

1. Users can retrieve data
2. Users can change data
3. Users can delete data
4. Users can create data

Database

1. Admins can view database entries
2. Admins can edit database entries
3. Admins can create database entries
4. Admins can delete database entries

3.8.5 Non-Functional Requirements

1. Usability - The feature must be efficient and effective
2. Performance - The feature must process queries within a reasonable timeframe
3. Learnabilty - The feature must be easy to learn and use

3.8.6 Evaluating Results

The evaluation criterion which will be used to determine the success of the backend components in satisfying its purpose and requirements.

1. Performance: Does the feature process queries within a reasonable timeframe
2. Usability/Learnabilty: Whether the Feature is easy to learn and use
3. Functional Requirements: Does the feature correctly satisify each requirement

3.9 System Architecture

This section will outline the technology stack we will use for the project. The system can be broken down into 3 layers. The presentation layer, the application layer and the data layer. Below we will highlight the different technologies used for each layer

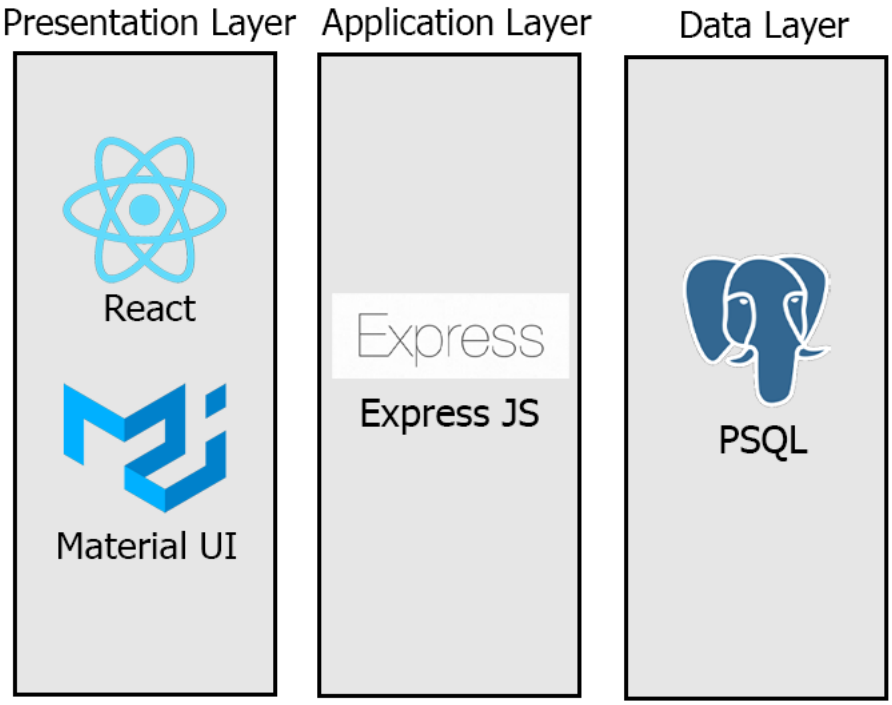

Figure 3.26: A visual breakdown of the different layers of our technology stack.

3.9.1 Presentation Layer

React will be used for the presentation layer, alongside the Material UI component library. React gas been chosen as its the current industry standard for front-end web design and can be easily used to develop high quality user experiences. Also many of our team members have had experience with using React in the past.
Material UI has been selected for a component library as its usage will make it very easy to standardise the visual style of our Meta LMS, as well as decreasing development time as it provides many high quality UI components that are ready to use.

3.9.2 Application Layer

Express JS has been chosen to create our REST-API on the back-end due primarily to its ease of use, and the fact that having the front-end and the back-end written in the

same language will make development easier for the entire team, as every member only has to be proficient in one programming language in order to work on all component of the project. Many of our team members already have experience with Express, and the PostgreSQL libraries for node also mean that using Express will simplify interactions between the API and the database.

3.9.3 Data Layer

A PostgreSQL has been selected for the Data layer due to its free and open source nature. It has also been selected based upon the experience of team members, as most of us have more experience with PostgreSQL over other database technologies.

Chapter 4

Project Execution

4.1 Project Management

This section details the methods and tools used to help manage this project and separate workload between the seven group members of this thesis.

4.1.1 Communication

Throughout the duration of this project, the two main platforms used for communication between group members and staff were Slack and Microsoft Teams. In order to ensure that all discussions about the implementation, presentations and reports were kept in one place, Slack was used as the main platform for text communication. Weekly meetings were held on Microsoft Teams and allowed each group member to explain what they had been working on the week prior and ask any questions. Occasionally, additional student-only meetings were held to prepare for presentations or plan reports.

Comunication Between Teams

While everyone lead the development of a frontend component, there were only a couple group members who were in charge of the backend. This meant that there needed to be a lot of discussion between group members during the backend development phase of the project. To relieve some of the load on the backend developers when it came to designing the API and database, each of the frontend leads devised the schema required for their feature. These schemas were comprised of backend requirements like database tables and API endpoints. This helped speed up the backend implementation process as the developers could focus mainly on building the API instead of designing it.

Once the initial API was completed, any changes or requests for new features were sent to the backend developers via Slack so that they could be implemented. When necessary, frontend developers could also make their own changes or add features to the backend.

4.1.2 Integration

The dashboard and course pages was the central hub when it came to integrating all of the LMS components. Links to the Topic Tree and Gamification features were added to the sidebar on the dashboard while the course page contained links to the Accounts, Lectures and Tutorials, Assessments and Forums components.

4.1.3 Code Management

A GitLab repository was used to house all the code for this project. Version control was essential in a group this size as it ensured that no one's work was being overwritten, accidentally deleted or changed without them knowing. Each feature had its own branch that would be merged into master once it reached a stage in development where it was production ready. This meant that the implemented features were complete and there were minimal bugs. This ensured that the code in master could be deployed and used without any issues.

A similar process was used for writing the thesis reports, where each group member created their own branch for writing their section of the report. These branches were all merged into master to produce a final copy for submission.

4.2 System Architecture

This section will outline the technology stack we used for the project. The system can be broken down into 3 layers. The presentation layer, the application layer and the data layer. Below we will highlight the different technologies used for each layer, and both the benefits we found and challenges we faced using these technologies.

4.2.1 Presentation Layer

The presentation layer was the area that had the most changes during development compared to the initial technology stack set out during the project plan. React was still chosen as the framework we would be using, but rather than using Material UI as our component library, we instead chose to use a framework called Chakra UI. The two main reasons for this choice were that some members of our team had created previous

projects using Chakra UI which would be very easily incorporated into our Meta LMS, reducing development time. Rather than using both Chakra UI for the re-used parts of the project and Material UI for the rest, we decided to use Chakra for the whole project to ensure a consistent visual language throughout the product. The second reason was that as development continued, the team decided that the visual style of Material UI did not lend itself to the welcoming environment we wished to create for our LMS, as the aesthetic was too harsh and corporate for our vision. Chakra UI instead offered a much more pleasant design aesthetic and was far more conducive to the project, leading us to adopt it instead.

4.2.2 Application Layer

The application layer consisted of NodeJS and Express as laid out in the project plan, but also expanded to two other key technologies during development. The first of these is JSON Web Token, a tool for managing user authentication states throughout the app. JSON Web Tokens were chosen due to their ease of implementation which was crucial due to the time restrictions of the project, and they were also highly compatible with the JavaScript back-end we had created. The second of these technologies is Open API and Swagger UI. While these are not user facing tools, using Open API and Swagger UI to create an easy to use, visual representation of our API during development was invaluable developer tool. Open API allows us to create metadata to describe the structure of our RESTful API and all of its required input and potential outputs, while Swagger UI is a visualisation tool that takes in the metadata from Open API to create an interactive visual representation of the API to assist developers with testing, and implementation of the front-end environment.

4.2.3 Data Layer

The data layer was largely unchanged throughout the development process from what was decided during planning. We continued to use PostgreSQL for our database, and it adequately served our needs for quickly serving all of the data we needed, and had enough capacity for the level of scale we reached during testing and development.

4.3 Deployment

4.3.1 Continuous Integration

Throughout the thesis, we have an instance of Teamcity that polls our gitlab repository master branch for any updates. Once an update is detected, teamcity will trigger a build that creates a docker image and sends it to our technology stack in portainer to update on our product server at metalms.tech. We had 2 triggers, one for the backend

and one for the frontend. Through this, any updates to the gitlab repository is on the live environment within 5 minutes. This proved to be useful especially for updating the backend apis which all the features are pointing towards. This continuous integration feature helped the team to iterate their features faster and allowed the team to work more efficiently.

4.3.2 Continuous Deployment

All of our frontend and backend services are bundled into a docker image which makes it modular and easily scalable.

Metalms Services

- Postgres Database at db.metalms.tech
- Backend API at api.metalms.tech
- Frontend interface at metalms.tech

The continuous integration helps us to iterate quickly and see changes live on the metalms environments. Our services are deployed on a linux machine in the sydney datacentre through OVH and is managed through portainer's user interface. If there is extra capacity required for the platform, we can simply multiply the number of services running the various docker images and have a load balancer manage the traffic coming into each service to ensure uptime.

Chapter 5

Project Walkthrough

This section provides a walkthrough of the Meta Learning Management System, with screenshots and features shown below. The section is separated into the several main features of the management system, with each author responsible for their various features.

The various academics and students who will use the system are referred to as Users.

5.1 Accounts and Enrollments

5.1.1 Login and Sign Up

Design

The login and sign up pages are the first pages the user will see when interacting with the Meta Learning Management System. As such, it needs to be easy to understand while also highly functional. To achieve this, a style that resembles what has become the standard login and sign up procedure across the entire web has been used. On the login screen, a user is prompted for their email address and password, and have a sign in button to submit these credentials for verification. As for the sign up page, the form is similar in format, except with more fields to take in the required information to create a new user account. In addition to requiring a user to enter their email and password, the sign up for also asks for a full name, their student ID, and a password confirmation to ensure that the password they are entering is correct.

Purpose

By allowing users to create personal accounts within the LMS, it allows all of the other functionality of the LMS to be tailored specifically to that user. The accounts feature

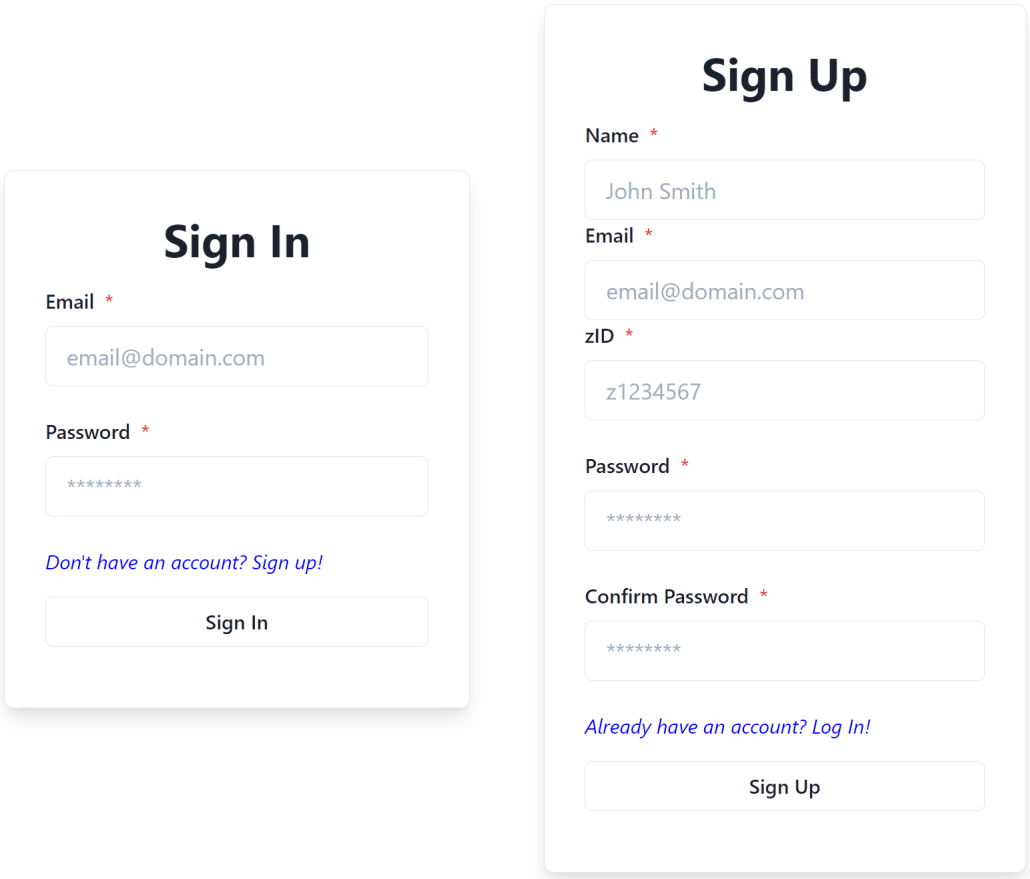

Figure 5.1: Login and Sign Up Forms

is what ties all of a user's progression within their courses to that users, and without it the rest of the LMS would not function.

What Was Implemented

Two forms have been created, a log in form that asks for the users email and password, and a sign up form that asks for a name, email, student ID, password and password confirmation. The inputs on these two forms are text fields that automatically sanitise inputs to ensure they meet the correct formatting, and there is a submission button at the bottom of the form. There is also a hyperlink that allows the user to switch between the sign up and log in context.

How It Was Implemented

The log in for and sign up forms both contain a number of text fields for user input. For log in, the user fills in their log in credentials, their email and password, and submits them. The site then passes this data to the backend, which first searches for that email to confirm it exists within the database, then compares the submitted password to that users password to see if they got the correct credentials. In the case of a failure, it

will return an error to the front-end, however on a successful login, it returns a token to the front-end which must then be used with all future requests to the backend to authorise said requests. This token is stored in the browsers storage alongside the users ID, which is also necessary for many of the requests.

The sign up form operates very similarly, except instead of confirming the email already exists in the database, it instead confirms that the submitted email and student ID do not currently exists in the database, then confirms that both the password and confirm password fields both contain the same data. If all of these checks pass, the data is entered into the database, and an authorisation token and user ID are returned to the front-end, the same as the log in form.

Considerations

A lot of detail was also put into creating intuitive error messages on the login form that are informative to the user, without compromising the security of the Meta LMS. For example, an error message will tell you when a field is missing/has been left blank, but if either the email or password is typed incorrectly, it will simply state "Incorrect email or password" so as to not reveal to a potential bad actor whether an email address they have entered is in our database or not.

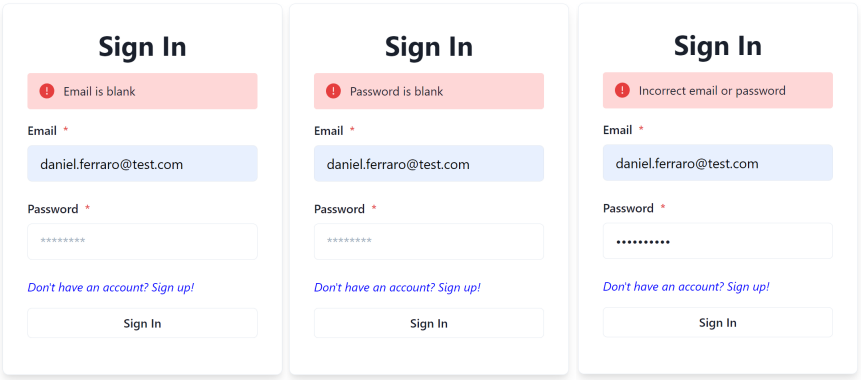

Figure 5.2: Login Form Errors

5.1.2 Account Management

Design

The account management screen allows users to view and edit their account information easily within the Meta LMS. This page is accessed from the drop-down menu attached to the profile indicator in the top right of the home page. This drop-down also houses the log out button, which when pressed logs out a user and returns them to the log in screen.

On the account management screen, a user is shown their full name, student ID and

Figure 5.3: Profile Drop-down

account type, either staff or student, in the page header. They are then show a list of fields that contain their other account information, such as email and profile image URL. Under these fields, is a table showing the courses they are currently enrolled in, in the form of a table showing the course code, course name and course outline. There is also an edit button attached to the account information fields that allows a user to edit their profile.

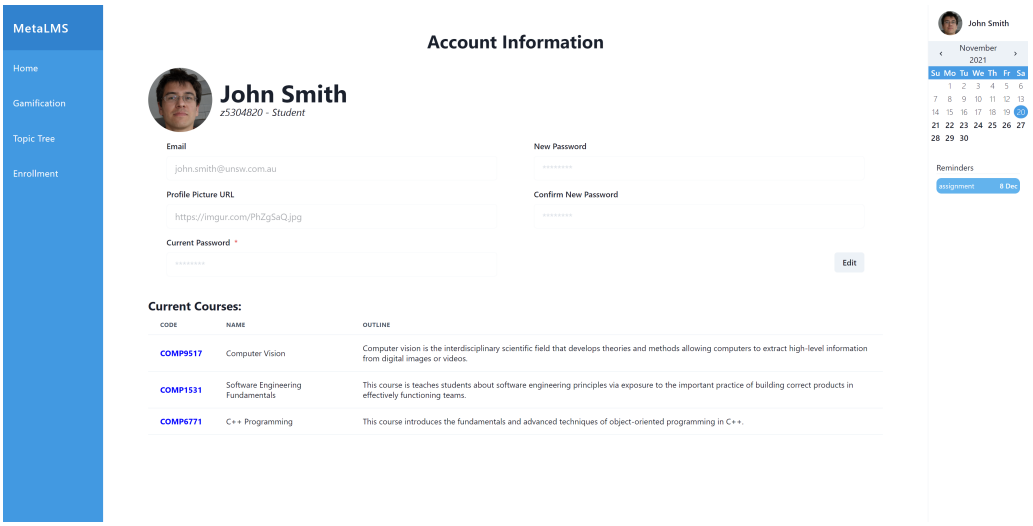

Figure 5.4: Account Management

Purpose

The intention of the account management screen is to have a place within the LMS where the user can view all of the information associated with their account, as well as edit and update this information. By the inclusion of this feature, users have a quick and easy way to ensure all of their personal information is correct and up to date, increasing the user satisfaction as this allows the Meta LMS to remain personalised to the user. It is also a useful tool for quickly viewing all current courses a user is enrolled in or is managing.

What Was Implemented

The accounts screen not only displays all of a users information, there is also an edit

button attached to these fields. When clicked, it changes these fields to mutable text boxes, and the edit button changes to a save and cancel button. To edit their account information, a user must mutate the field they wish to change, and also enter their current password to pass the security check required to make the changes. If a user wishes to change their password, they must enter their current password, their new password, then re-enter their new password to confirm it. To change their email, they must simply enter a correctly formatted email, and to change their profile image, they must enter a URL for an image. When these changes are saved, all of the information is quickly updated on the page to reflect these changes.

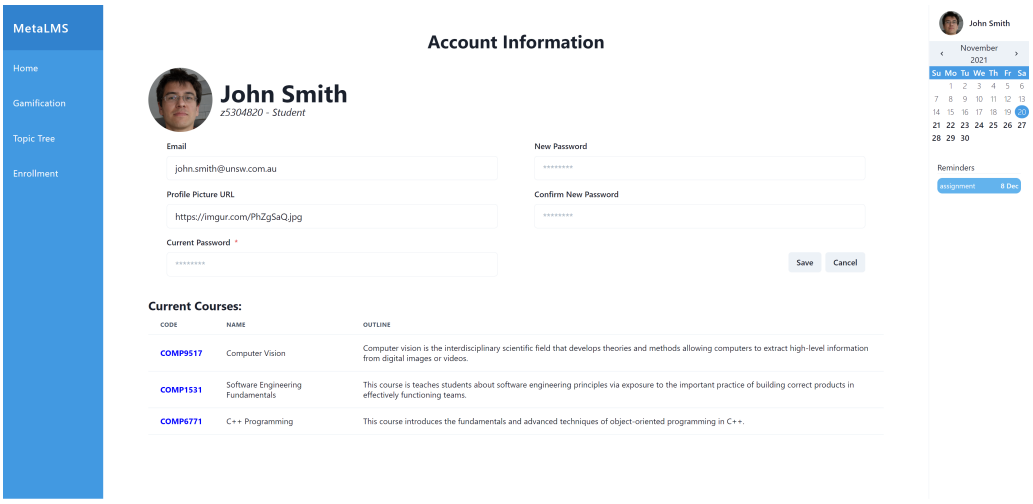

Figure 5.5: Account Editing

The current courses table allows users to at a glance view all important information regarding the courses they are currently enrolled in for student users, or managing for staff users. It shows the courses' course code, which is a hyperlink that when clicked, will take the user to that course's home page, the course's name, and the course outline, which is a brief description of what the course is about.

How It Was Implemented

When the page is loaded, the front-end sends a request to the back-end for all of the user's data. The back-end collects all of the personal information related to the user to send back. In the same response, it also looks at all of the courses the user is enrolled in, as the courses ID's are linked to the user's database entry. It then gathers the appropriate information for these courses and packages them into its response. This data is then reflected on the front-end.

When the user clicks the edit button, it triggers a state change on the front-end to change it to the editing state. This makes all of the text fields in the user information section editable. After the user makes their changes and clicks the save button, the front-end optimistically updates to reflect the new information while simultaneously sending a request to the backend. If the security checks pass and no other errors occur (such as the user attempting to change their email to an email in use by another user),

the backend will return success. Otherwise, the backend will return an error and revert its optimistic update to the previous state and display an error toast to the user.

Considerations

The primary consideration for this page was security. We do not want bad actors to maliciously change a users information, so major decisions were made to prevent this. First, any changes to a users information requires the user to resubmit their current password. Failing to get this password correct will cause any changes to not be made. When updating the users password, they must not only re-enter their old password, but also confirm their new password to ensure no input errors accidentally get saved to the database.

5.1.3 Enrollment Dashboard - Staff

Design

For staff users, each course will have a dashboard screen that allows them to manage enrollments for that course. There are quite a few elements within the enrollment dashboard to allow staff to easily manage all enrollment features of a course from a single easy to use screen. The top portion of the page is dedicated to generating course invite codes. An invite code is a code or URL a teacher can pass onto students to allow them to enroll themselves into a course. Below that is a table that displays all currently active invite codes, how many uses they have, and how much time until they expire. It also gives action buttons that allow staff to either copy these codes to the clipboard for sharing, or to delete these codes. Beneath the table, there is a list of all currently enrolled students within the course, and all current staff of the course. There is buttons to allow staff members to manually un-enroll students. To the left of this, there is a text input form that allows staff to manually enter a students student ID and enroll them into the course. The last UI element is a toggle-able switch which sets whether a course is publicly search-able or not.

Purpose

It is important to the quality of the Meta LMS that staff have a large number of tools to quickly and view and manage enrollments within a course. Due to the nature of the structure of courses as a collection of topics, and how topics are pre-requisites for other topics and as such, other topic groups, it is important for staff to be able to make changes in enrollments quickly so as to not disrupt students' learning and progress. As such, by having one central dashboard for enrollments, it gives staff a very powerful tool to quickly manage all facets of the enrollment feature, from invite codes, to manual enrollments, to search-ability.

What Was Implemented

This screen contains all of the important controls for managing the enrollments for

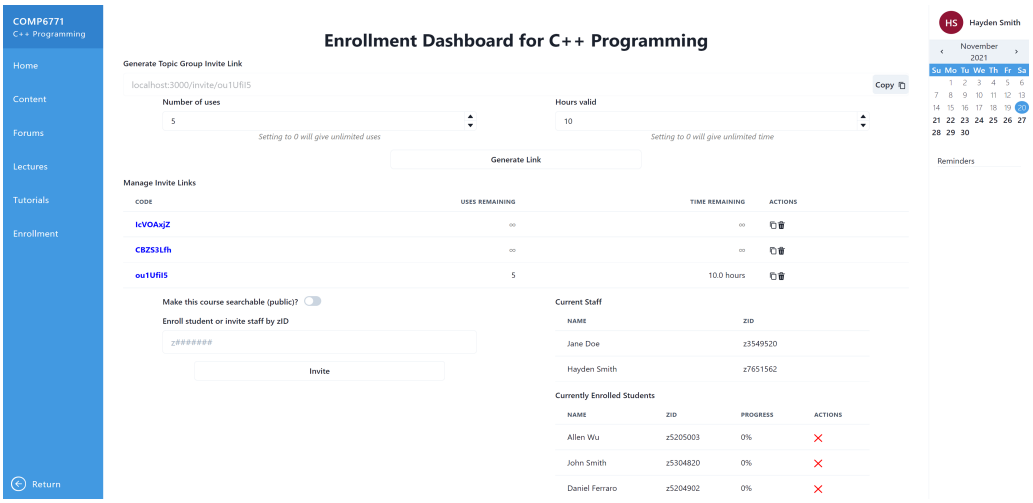

Figure 5.6: Staff Enrollment Dashboard

a course. It required the creation of three key enrollment features, the course invite code, course search-ability, and manual enrollment by student ID. The course invite codes requires a staff member to be able to create and manage course invite codes, and manage optional parameters for these codes such as the number of uses its valid for and the time its valid for. They are also able to easily share codes with the automatic copy functionality, and delete said codes from the code management table. The search feature is very easily managed through a simple toggle switch which toggles the course between public mode, or search-able mode, and private mode, or non-search-able mode. The manual enrollment feature simply requires the staff member managing enrollments to enter the student ID of a student they wish to enroll, and submitting it. This feature also encompasses the student list, where staff can view all currently enrolled students in the course, and manually un-enroll them if they so choose.

How It Was Implemented

There are two options a staff member has when generating an invite code. First, they can set the number of uses a code has, with a range from zero to infinite. They can also set the time an invite code is valid for, also ranging from zero to infinite. Under the inputs for these two fields is a “Generate Link” button which when pressed packages the values set for the optional parameters, and sends them in a request to the back-end. The back-end receives this request, and inserts a new code object into the database. If the parameter for time remaining or uses are set to 0 when sent to the back-end, the back-end does not add these parameters to the data entered into the database and thus the code has infinite uses or infinite time remaining. After entering the data for the code into the database, the backend responds to the front-end with the usable code. The front-end then automatically appends this code to the URL of the site and displays this in the text-box at the top of the screen. It also adds the code to the enrollment codes table below.

Below the code generation area is a table where all of the currently outstanding invite

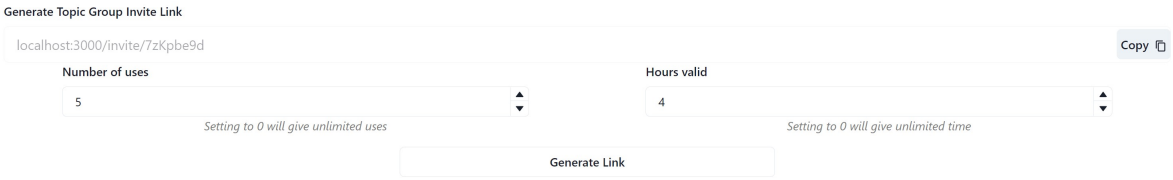

Figure 5.7: Generate Invite Link

links can be viewed. The data for these codes are fetched whenever the page is loaded. First, the front-end requests the list of all invite codes for the course. The back-end then gets a list of all codes for that course from the database. It then checks this list for any which have 0 uses remaining, or have 0 time remaining and deletes them from the database. It then sends this updated list to the front-end to be displayed. For these codes in the table, if the user clicks the copy button, it copies the invite link corresponding to that code directly to the clipboard. There is also a delete button, that when pressed, optimistically updates the front-end to reflect the delete, and sends a delete request to the backend, which then deletes it from the database.

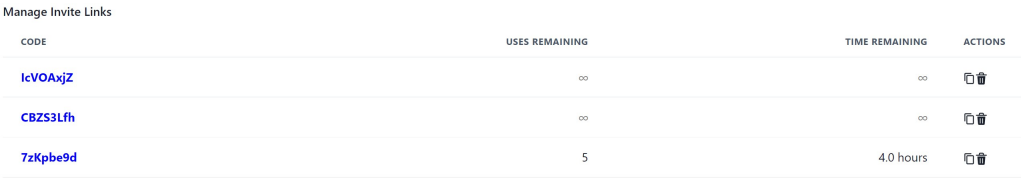

Figure 5.8: Invite Link Table

The next UI element is the search-ability toggle. On the page load, it reads the state of the search-ability of the course from the payload sent from the back-end that is requested when the page loads. It then sets the state of the search-ability toggle accordingly. When a user interacts with the toggle, the front-end optimistically updates, and sends a request to the back-end to change the state. The back-end then updates the state in the database, and sends a confirmation to the front end.

The final component is the enroll by student ID and student list section. The enroll by student ID feature allows a staff member to enter a student's ID into the text field and submit it to be enrolled into the course. When it is submitted, the front-end checks to see if the student ID is in a valid format, and if so, sends it to the back-end to enroll the student in the course. The back-end checks to see if the student exists, then if the student is already in the course. If both of those checks pass, the back-end enters the student into the course in the database, and sends an updated enrolled student list of the course to the front-end. Unlike many other components on this page, the manual enroll is not optimistically updated on the front-end due to the higher chance of failure due to a high reliance on user input. Thus, the front-end waits to receive the updated student list from the back-end to update the student list on the front-end. The student list is simply a list of all users enrolled in the course, with students separated

out from staff. A staff member will also see a red x next to all the students in the course allowing them to manually un-enroll a student. When clicked, the front-end optimistically updates the list to remove the student, then sends the delete request to the back-end, which then removes that students enrollment from the database and sends a confirmation to the front-end.

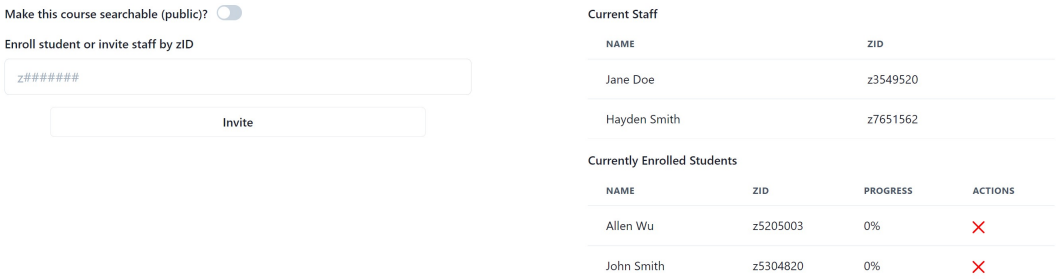

Figure 5.9: Manual Invites and Student List

For the entire page, if at any point an error occurs, any optimistic updates will be reverted, and an error toast will be displayed to the user explaining what went wrong. This is by design, so that the information on the page is always correct, and errors are in a consistent and easy to understand format.

Considerations

The primary consideration made when designing and developing this dashboard was ease of use. As a result, the initial plan to separate these features into multiple pages, one for invite links, one for manual invites and one for monitoring student lists and managing search-ability, was decided to not be followed in favour of the current implementation, having it all in one place. This allows users to more quickly manage enrollments, and not have to waste time switching contexts.

5.1.4 Enrollment Dashboard - Student

Design

The student enrollment dashboard is split into two main areas, the join with invite code area, and the search for a course area. Both sides consist of two very basic UI elements, a text input, and a submit button. For the invite code side, students input a code given to them by a teacher, and submit that to be enrolled into a course. For the search side, students can enter either the course code, course name, or key words related to something they wish to learn about and submit it. This will then return matching courses in the form of cards, which say the courses name, code and course outline, and has a button they can press to enroll.

Purpose

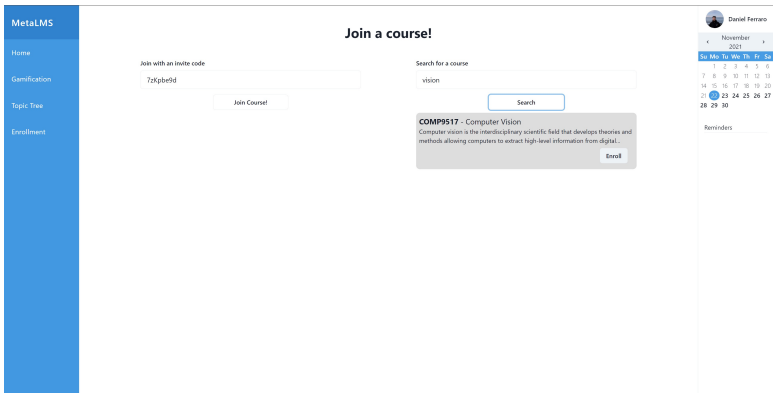

Figure 5.10: Student Enrollment Dashboard

Due to the fast-paced nature of learning that comes from using the topic group approach, it is important that students have a high level of agency over their learning. As such, it is essential the tools students have access to that allow them to manage their enrollments are powerful enough to afford them this agency. This adds a tremendous amount of value to the Meta LMS as it gives students access to the full benefits of the topic group system.

What Was Implemented

The two systems implemented are the student face of the invite code system, and the course searching system. The course invite system also encapsulates the login redirect feature, that allows students who are not currently logged in to the LMS to simply click an invite link, login or sign up, and then automatically be redirected to the student enrollment dashboard with the invite code pre-filled to quickly and easily begin using the LMS to learn.

How It Was Implemented

The course invite system works by allowing a user to input a course invite code, and submit it to enroll themselves into a course. This field will be blank on the page load if the user has navigated to the enrollment page manually, but will be pre-filled if the user has arrived to the page by using an invite link. This is done by reading the ID URL parameter of the invite link, and setting that as the default value of that text field. When the user submits an invite code, the front-end sends that code in a request to the back-end. The back-end first checks to see if the code exists in the database, then checks if the code is valid, that is, if it has time remaining, or uses remaining. If the code has become invalid, it deletes it from the database and sends an error to the front-end. If the code is valid, it then checks if the student is already enrolled in the course, and if they aren't, it enrolls them, and updates the remaining uses for that invite. If the student's enrollment causes this code to become invalid, it is then removes that code from the database. Once the enrollment is successful, its sends the confirmation payload to the front-end. This payload contains the information of the course the student was enrolled in, as well as the confirmation status. This information

is then displayed to the user on the front-end in the form of a toast that confirms the enrollment was successful and the course they were enrolled in.

The search for a course feature is much the same, the user inputs a string and sends it to the back-end. However, the string entered will correspond to either a course's code, a course title or details about a course. When the user submits this, the front-end sends a request to the back-end for results related to the user's query. The back-end then compares the query to the course codes, names and course outlines stored in the back-end, but only of courses that have been marked as search-able. It then assembles a list of results and sends that response back to the front-end. The front-end then displays this list to the user in the form of cards. These cards contain all the basic information about the course, and also have an enroll button. When the user clicks the enroll button, the front-end sends the enroll request to the back-end. The back-end then does checks to see if the user is already in the course, and if not, enrolls them in the course and sends confirmation back to the front-end. This successful enrollment is then displayed in the form of a toast.

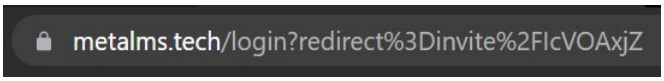

Figure 5.11: Invite Code Login Redirect

The feature that handles the redirect to the invite screen from the login page does so by storing the invite code from an invite link in the redirect URL parameter of the invite page. If this parameter is present, once a user has successfully logged in, it will redirect them to the invite page and pre-fill the code.

Considerations

The primary consideration for this feature was the redirect from the login screen, particularly for new users. To create as positive a user experience as possible, the sign up to enrollment process needs to be as seamless and friction-less as possible. Allowing users to simply input an invite code, create an account then automatically be redirected to the enrollment page with the code pre-filled makes this enrollment process as easy as possible for the user, allowing them to begin using the LMS to learn very quickly.

5.2 Topic Tree

5.2.1 Home Page

From the homepage, a user (both students and academics) can get to the topic tree from the sidebar.

This allows quick access to the topic tree so students can easily view which topics they

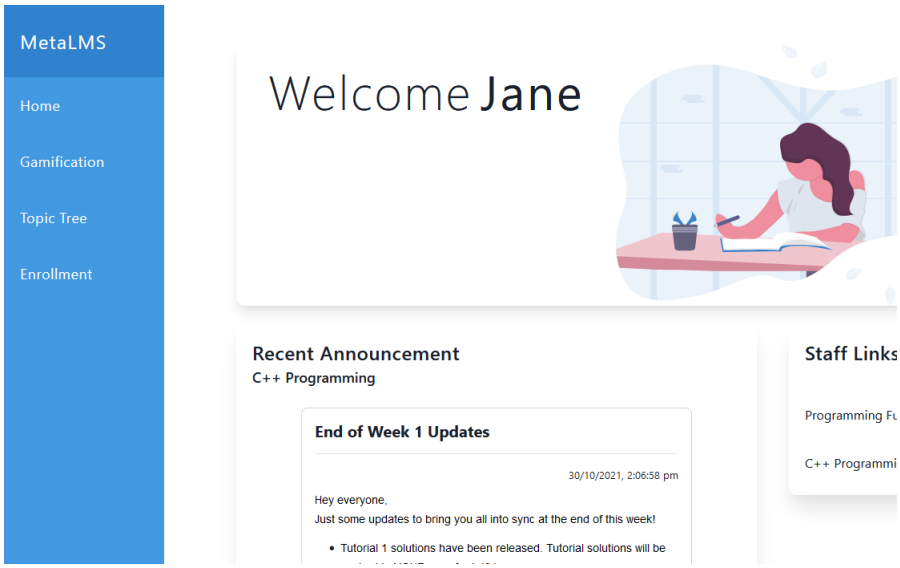

Figure 5.12: Topic Tree Link in Dashboard

must complete next and academics can also easily edit topics and upload content.

5.2.2 Graph View

Design

The main feature of the topic tree is the graph view. Here the user can view the topic groups, topics and how each topic is related to each other (through prerequisites). On first load, the topic groups are shown with arrows to indicate the prerequisites. In the below screenshot, there is a topic in Programming Fundamentals that is a prerequisite for a topic in C++ Programming i.e. a student must complete the topic in Programming Fundamentals before they can start topics in C++ Programming.

A topic group is marked as green, and when clicked it expands into the individual topics as shown below.

A hull around the topics is drawn to indicate that they are part of the same topic group, as in the above example the topics are part of C++ Programming.

The graph can be zoomed in, and you can drag topics out to make topics to be more visible.

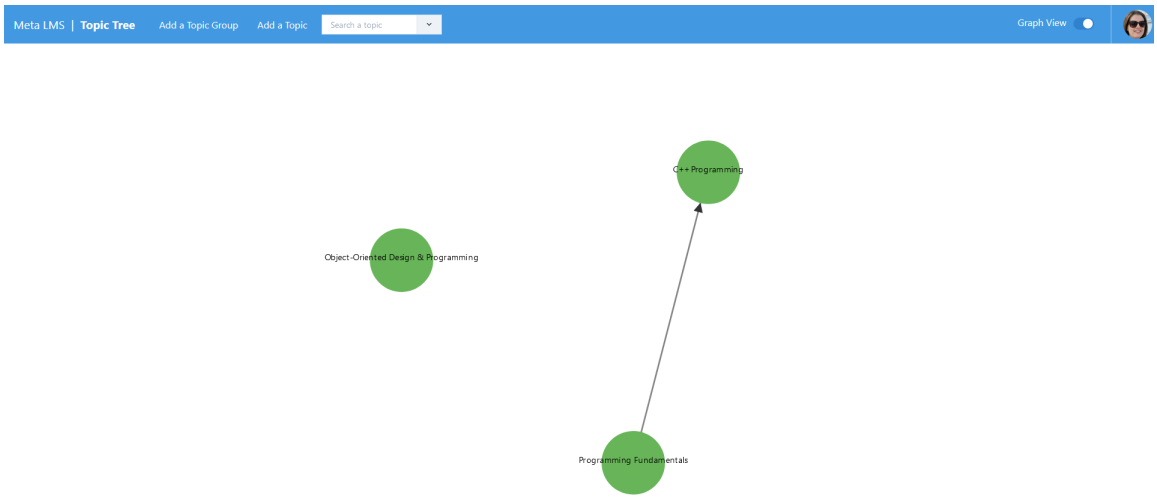

Figure 5.13: Topic Tree Graph View

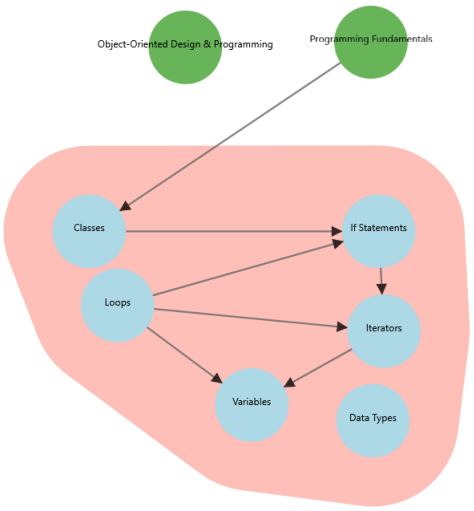

Figure 5.14: Topic Tree Expanded

Purpose

The topic tree graph view allows users to view the relationship between topics and topic groups. Users can easily see which topics are the prerequisites of which other topics, and view the topics inside topic groups. This simplifies reusability for academics, as they can choose topics to add as prerequisites to their own topics.

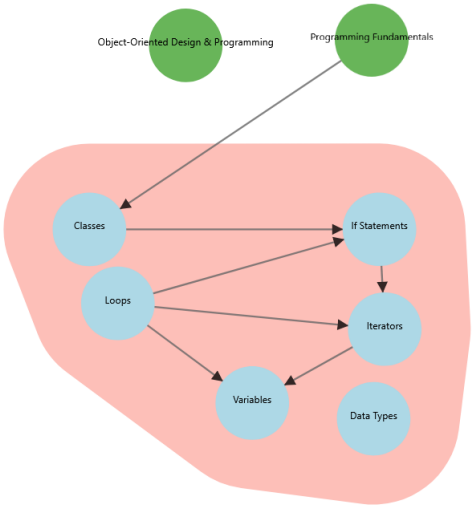

Figure 5.15: Topic Tree - Hull

What was implemented
The main graph view has been implemented as shown above. Features include clicking a topic group to expand into the topics inside it, showing the relationship between topics, and outlining the topics inside a topic group with a hull i.e. shape around the topics.

How it was implemented
D3 was used as a library to display the graph view. The list of topics with their topic groups is loaded from the database, and D3 algorithms are applied to create the hull, topic groups and topics on the graph.

5.2.3 List View

Some users may want to view the topics in a list instead of a graph.

Design

A list of the various topic groups is viewable, and when a topic group is clicked, the topics that are in the topic group expands as shown below.

Purpose
The purpose of this view is to allow academics and users to view the topics in a simpler

Figure 5.16: Topic Tree - List View

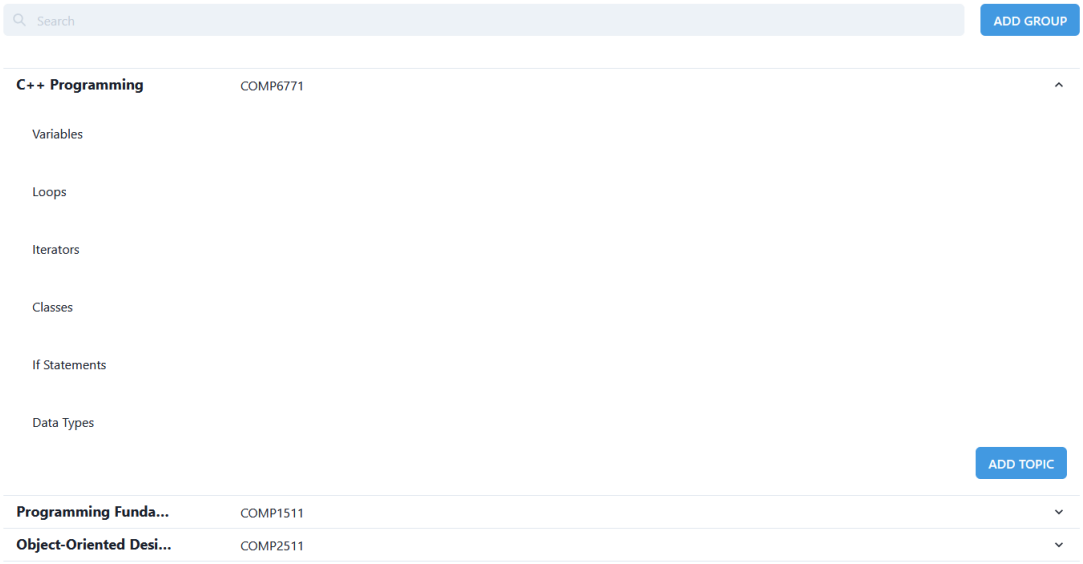

Figure 5.17: Topic Tree - List View Expanded

view if they wish. Some users may find the graph view too confusing to use or not as efficient when searching for topics, and so the list view provides an alternative to the graph view. As discussed in the analysis section, potential users reported that having the choice between the two views was a good feature of the topic tree.

What was implemented

Users can view the topic groups in list form, and click a topic group to view the relevant topics. A search bar is viewable at the top of the list view, to allow users to search for their topic or topic group.

How it was implemented

Chakra UI components were used to design the list view. The same endpoint as the graph view was used to load the topic groups and topics, and several React features

were used to implement searching.

5.2.4 Resources

When a topic is clicked on the graph view or the list view, the user can then view the prerequisites of that topic, tags for search optimisation, and any files associated with the topic.

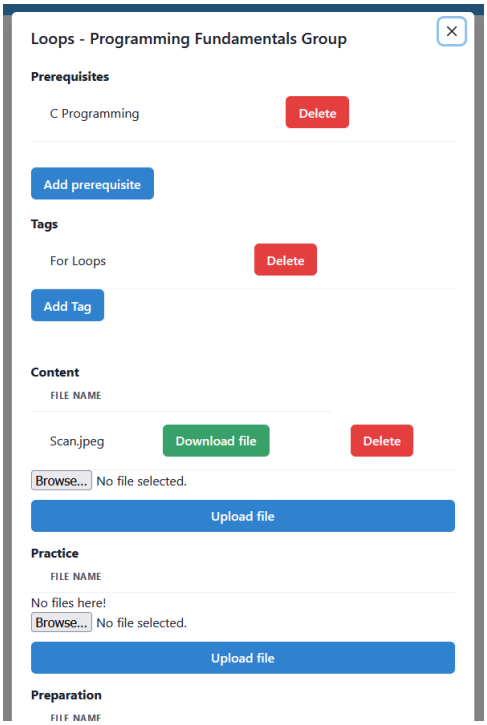

Figure 5.18: Topic Tree - View Resource

There are four file sections: Content, Practice, Preparation and Assessments. This is so it is adaptable to most learning models. Academics can upload any file they would like to a topic, and it will be viewable for all to view. In the future, permissions will be implemented to restrict access to certain resources. Third party integration with services such as YouTube, Echo360 and The Box will also be added.

Purpose
The purpose of the resources modal is to allow users to view files and edit prerequisites and tags. This makes it easier for users to add or remove files, and makes it possible to edit the topic tree itself.

What was implemented
Users can view, add and delete prerequisites, as well as tags and files. Students are

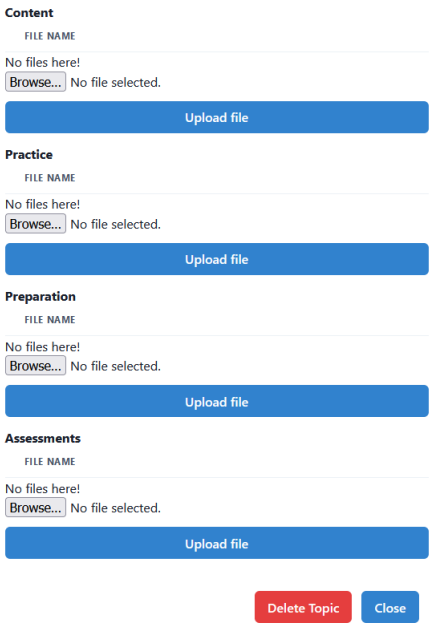

Figure 5.19: Topic Tree - File Sections

put in a view only state, where they cannot edit, delete or add prerequisites, tags or files.

How it was implemented
This modal was implemented using Chakra UI's modal library. Topic details such as files are taken from the original backend request when the topic tree is loaded, and POST and PUT requests are made when prerequisites, tags and files are edited.

5.2.5 Adding topics and topic groups

Users can also add a topic to a topic group, and add prerequisites to this topic in the same popup window.

Design

Users can also clone a topic i.e. if there is a topic in another topic group, academics can clone the materials in that topic and put it into another topic group.

Academics can also add topic groups, and search for a topic in the search bar in the header. Tags can be assigned to a topic which users can search for in the search bar to improve searchability of topics.

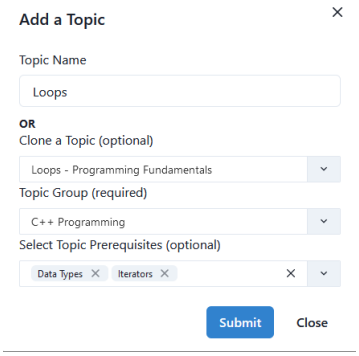

Figure 5.20: Topic Tree - Add Topic

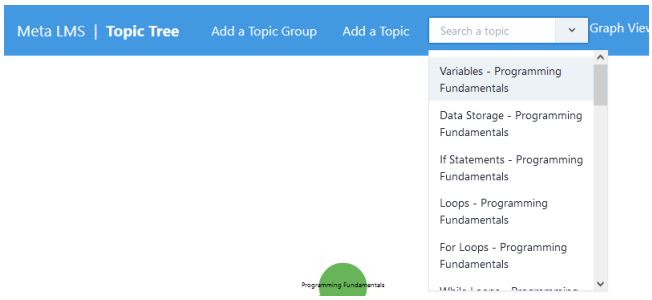

Figure 5.21: Topic Tree - Search Bar

Purpose
Academics (not students) can add topics and topic groups to the topic tree, to allow further editing capability to the topic tree. The cloning feature also furthers reusability, allowing academics to clone topics and their respective materials into their own topic group.

What was implemented
Users can type a new name of their topic group into the add topic group window to create a new topic group. Users can also select prerequisites when creating a new topic, or select an existing topic to clone.

How it was implemented
Similar to previous modals, Chakra UI was used to create the topics and topic groups. POST end points were set up so when a new topic or topic group is created, POST requests are sent to the server to add new entries to the database.

5.3 Course Pages

5.3.1 Course Selection Page

Design

The course selection page provides a home page for students to be able to have an overview of all of their courses. It also provides some other functionality such as showing the most recent announcement, course progress and the most recently accessed topic. This is the page that a student will see after signing in.

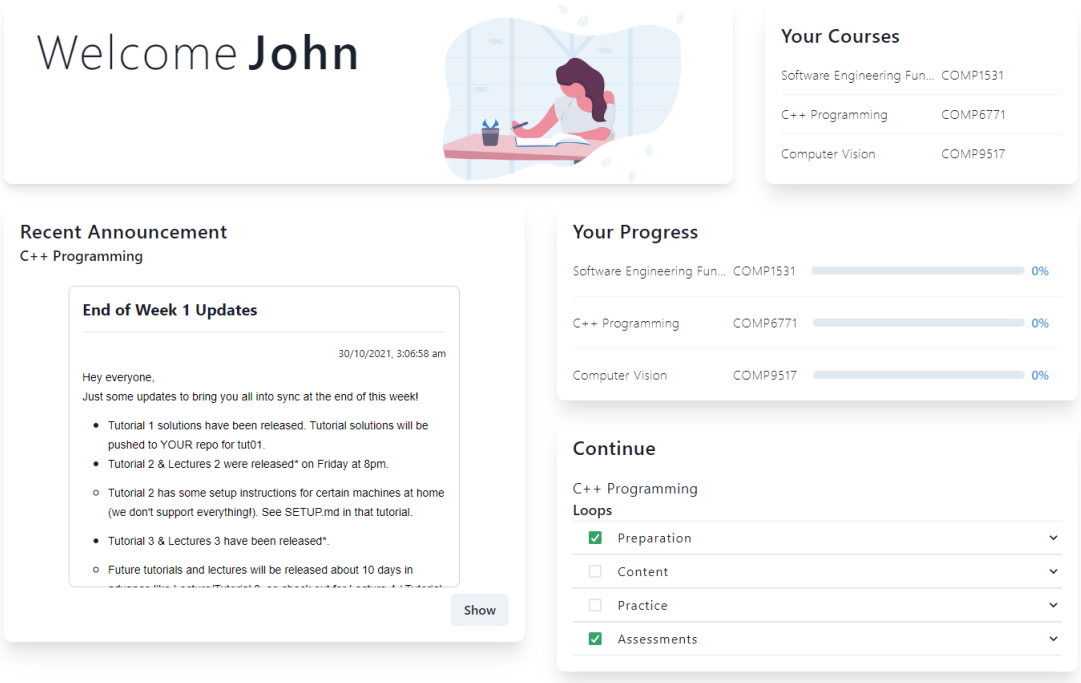

Figure 5.22: Student view of the course selection page

Purpose

The main purpose of this page is to provide users with an overview of their courses. It also is the central page in which users select and traverse into a particular course page.

What was implemented

The features within the course selection page that were developed are:

- Course selection;
- Viewing course progress;
- Viewing most recent announcement;
- Continuing most recently accessed course content.

How it was implemented

The course selection page was developed by drafting an overall dashboard UI. The current design utilises a grid based view to separate each feature in boxes. Each box represents a new feature. The course selection feature was developed by having each course link to the particular course dashboard. The course progress feature was developed by having the backend database storing each topic and topic content that a user had completed for a particular topic group. The frontend would then calculate the overall progress of the topic group and then display it on the dashboard. For the most recent announcement, all announcements from each enrolled course had to be extracted from the backend database. Then the most recent database can be identified and displayed. The most recently accessed topic is a field that when getting user data would be returned along with the other user information. This would then allow the dashboard to display the most recently accessed topic for a particular user. When a user clicks on a topic accordion in the course content page this value would change and be updated in the backend database.

Considerations

Many considerations had to assessed. The main consideration was identifying what possible features could be added in the dashboard. The features currently implemented are still in development and can be modified based on what could provide more utility to users. Other considerations include, what should the dashboard look like for a brand new user or admin? For a new user, the dashboard will not have any feature, but instead will direct the user to enrol into a course through the enrolment link on the sidebar. For an administrator, the course progress and most recently accessed topic features will not be displayed, but instead provide the functionality of having links to the forums or enrollment of a particular course. This feature would benefit an administrator as it would provide a quick and easy way to access those features which an administrator would utilise often.

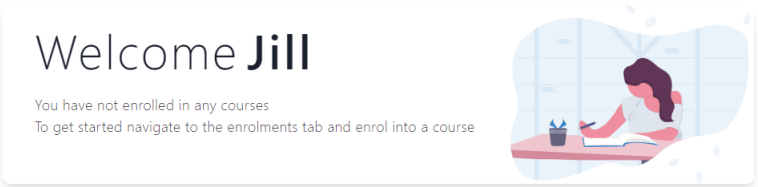

Figure 5.23: New user view of the course selection page

5.3.2 Course Dashboard Page

Design

The course dashboard allows users to view the announcements of the corresponding course and also ask forums questions linked to that particular announcement. For administrators, they can create and update announcemnts.

Purpose

The main purpose of this page was to provide an area for users to view important announcements of a course. It also allows administrators to create these announcements to notify students.

What was implemented

The features that were designed were the ability to view the information of an announcement (title, content, post/edit date and author), the ability to search and filter

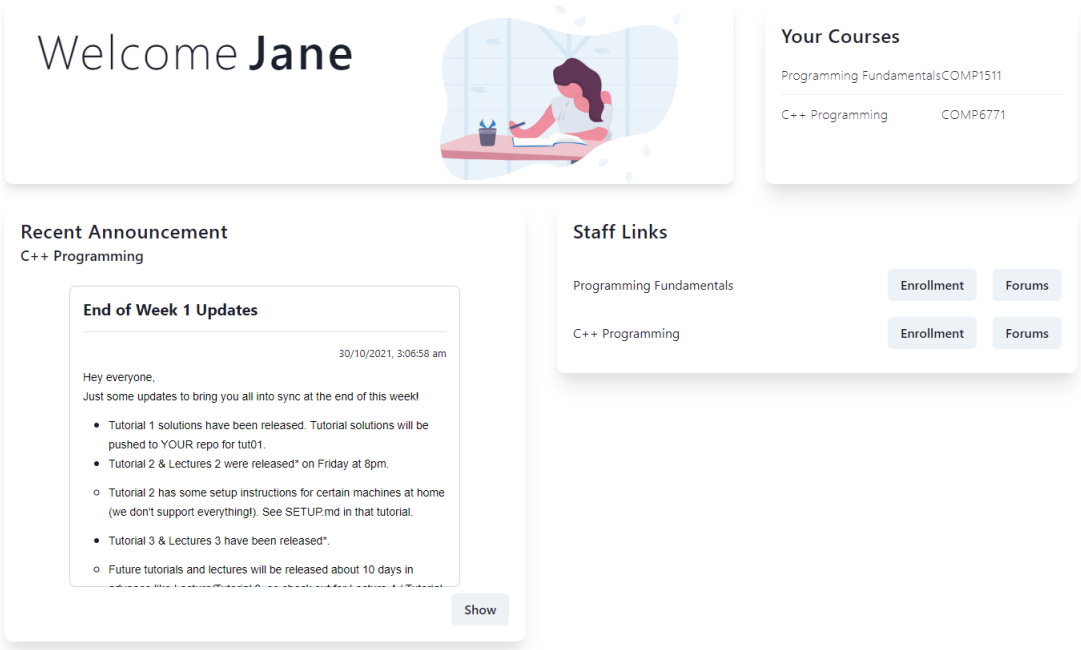

Figure 5.24: Administrator view of the course selection page

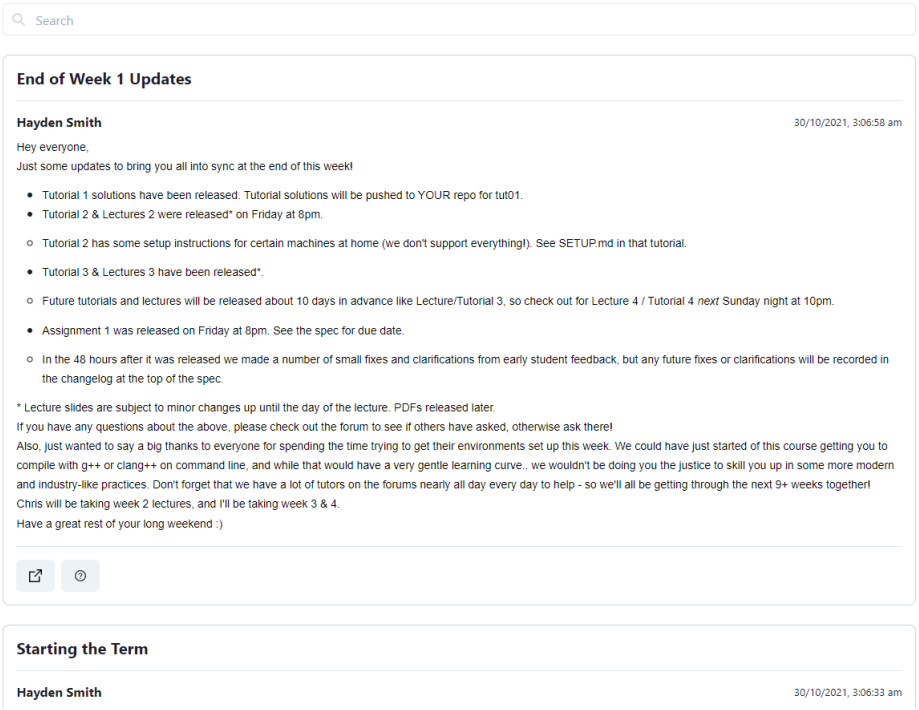

Figure 5.25: Student view of the course dashboard page

for announcements and the ability to create and edit announcements as an administrator.

How it was implemented

The design of the announcement was implemented in a way to ensure that it would be similiar to a forum post. This would allow users to identify that announcements act similiarly to forums posts. Thus many of the forum functionality of creating, editting and commenting are very similiar to how the announcement features were developed.

Considerations

The main consideration for the course dashboard was determining if a user should be able to directly comment on the course dashboard page or have it be done in the forums page. The final design has announcement comments linked in the forum so that the forum is the singular area within the course for discussion. This provides easier use and less confusion for users to swap between pages.

5.3.3 Course Content Page

Design

The course content page contains the accordion which displays the topics and corresponding topic content of a particular topic group/course.

Purpose

The purpose of the course content page is to provide the topic files of the particular topic group. It is also used to properly categorise the topic files within each topic for a user to learn from.

What was implemented

The features that were implemented are displaying the topics and their prerequisites and the ability to search and filter for specific topics.

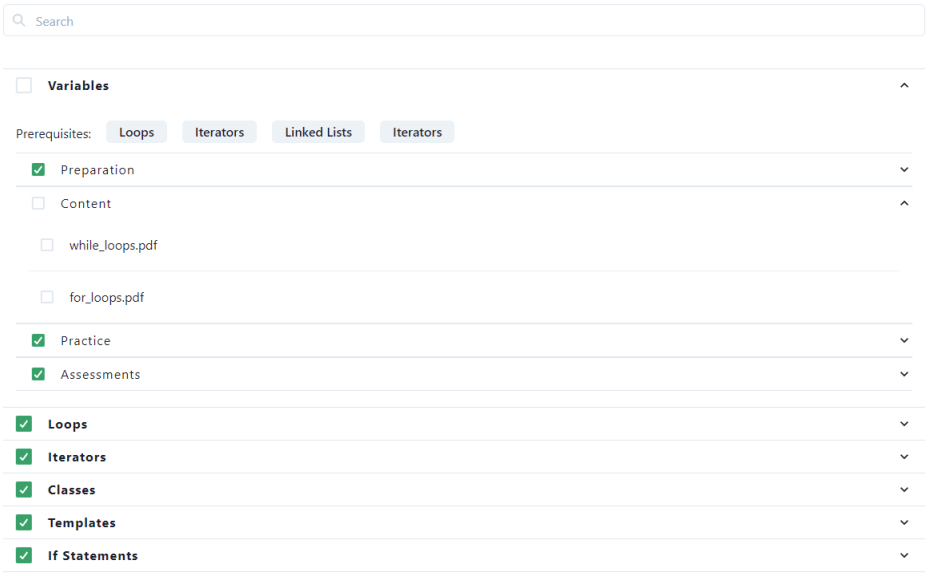

Figure 5.26: Student view of the course content page

How it was implemented

This page was implemented by first getting all of the topics within the corresponding topic group. Then for each topic categorise each topic file into 4 distinct types, preparation, content, practice and assessments. The search functionality was implemented by matching any topics or topic files names with the search term.

Considerations

A consideration made for the course content page was how to display the topic prerequisites. Since the topic tree displays these topics within a graph with edges to convey prerequisites, a list view needed a way to display the same information. The current design now uses a list within the topic to display the prerequisites of the corresponding topic.

5.3.4 Widgets Bar

Design

The widgets bar exists on the right hand side of both the main selection page and the course pages.

Figure 5.27: Widgets bar

Purpose

The purpose of the widgets bar is to provide extra utility for users.

What was implemented

The features of the widgets bar include the calendar feature, account section and the reminders list. The calendar feature allows users to create reminders on the calendar UI, these reminders will then be displayed on the reminder. The account section provides an area for users to log out and view their account.

How it was implemented

Since most pages within the metaLMS contain the sidebars, a universal template page was created with both side bars present and the main content of the page changing based on the route URL. The widgets bar utilise a calendar which allows users to click on a particular date and create a reminder which is stored in the database. These reminders would then be shown on the reminders list, with a maximum of 5 showing at a time. Users can also delete reminders.

Considerations

The main consideration for the widgets bar was to determine what features to add to the side bar. Due to the limited space, careful consideration was made, however possible additions can still be added.

5.4 Lectures and Tutorials

This section will depict a walk through for both the lectures and tutorials for each subsection.

5.4.1 Overview Page

The lectures and tutorials components are accessible from the side page of the course page. This will then render to users the overview page for either the lectures or tutorials component. Additionally, there will be a unique lectures and tutorials component for each different course.

Purpose

The purpose of the overview page is to display to users the existing weeks for either the lectures or tutorials component.

What Was Implemented

The overview page for the lectures and tutorials component is implemented via a list in chronological order of weeks to aid in readability of the component. Also, there is a panel for lecture/tutorial videos which display the external link to the videos.

How It Was Implemented

This component was implemented using the frontend tech stack. The component renders the lectures/tutorials and their corresponding files to the frontend using API calls to the backend.

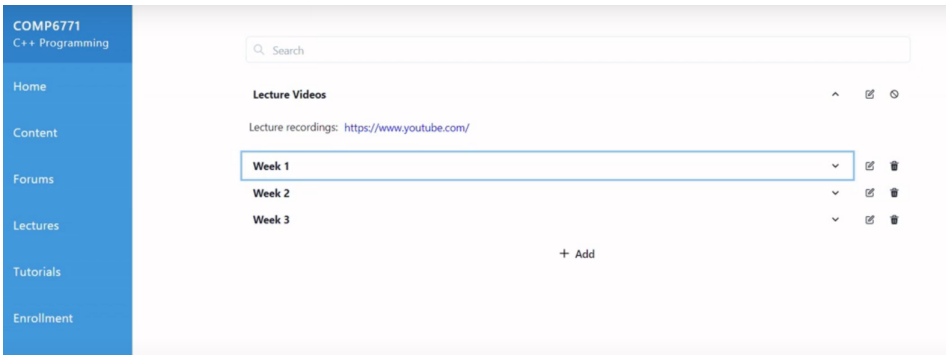

Figure 5.28: Lectures Page

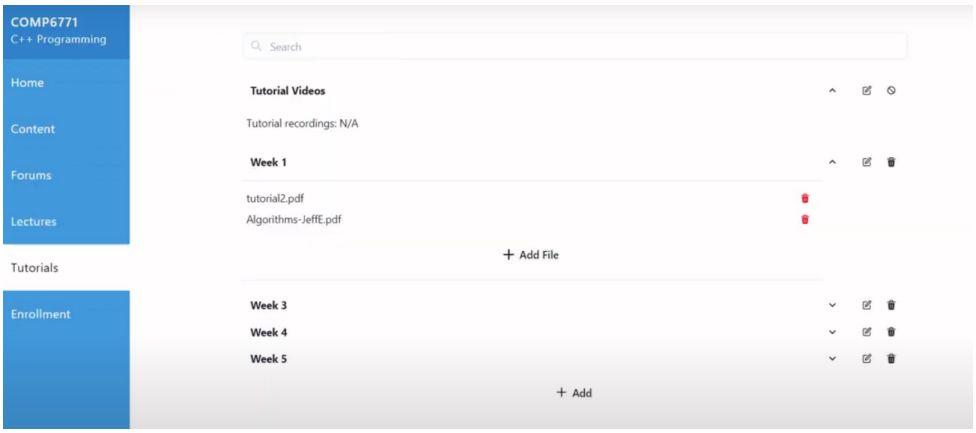

Figure 5.29: Tutorials Page

Considerations

As there is a possibility of admin users creating wrong weeks, there is a functionality to edit the weeks to the users discretion. Additionally, if an admin user were to create an existing week with new uploaded files, the files would be added to the relevant week instead of creating a duplicate week. Also, the inclusion of trash and edit icons help infer their functionalities to users.

5.4.2 Video Links

The user is able to clear an existing video link to recordings or streams by clicking the related clear button. Additionally, users can also edit and add another link to their external video platform.

Purpose

The purpose of the video links feature is to allow admins to update the video links and for students to access their lecture/tutorial recordings or attend the lectures/tutorials.

What Was Implemented

There is a panel in the overview page that depicts the relevant lecture/tutorial video links. In this panel, there is a hyperlink to the admins chosen video hosting site. For instance, the admin may choose to set the link to a YouTube link. Also, there is an edit button that allows admins to edit the link to their liking and a clear button which allows admins to clear the link and set it to N/A.

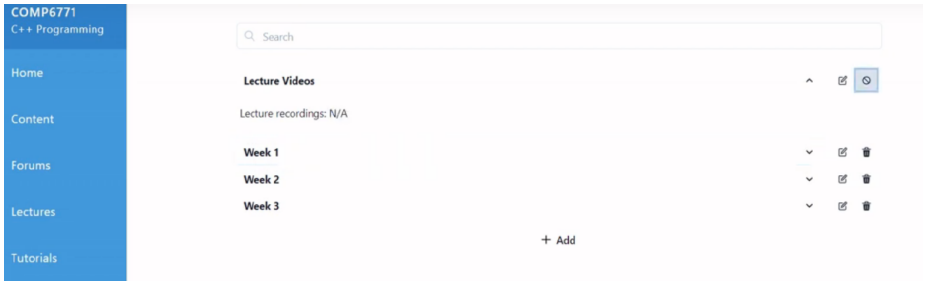

Figure 5.30: Clearing a video link

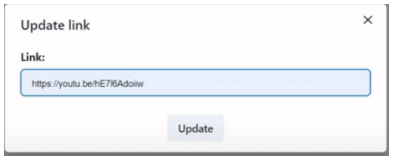

Figure 5.31: Adding a new video link

How It Was Implemented

This component was implemented using an API from the backend that retrieves the current video link and renders it into the panel. Also, this component uses APIs to edit and delete the link to the admins discretion.

Considerations

There is a potential for admin users to enter an invalid link, which the component would then notify the user that there was an invalid link inputted. As a result, this ensures that admin users correctly input links to their lecture/tutorial videos. Moreover, admins may also clear and set the link to N/A to ensure that student users are aware that there

are no lecture/tutorial videos available. Also, users whom are students are unable to edit the video links but are able to click on the links themselves.

5.4.3 Weeks

The weeks sections will include: adding weeks, deleting weeks and editing weeks for the lectures and tutorials feature.

Purpose

The purpose of the weeks feature is to allow users to distinguish between weeks in a tutorials/lectures component. This also allows users to determine the relevant course materials for their corresponding weeks of the course also.

What Was Implemented

In the overview page, there are panels present that indicate the weeks present in the relevant lecture/tutorials of the course. A user may click on the relevant weeks to reveal files corresponding to the week.

At the bottom, there is an addition button that infers to users that it adds a week to the table. This would open a modal which includes a week and file inputs.

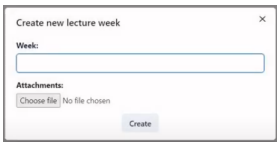

Figure 5.32: Adding a week

There exists bin icons to the right of each week to indicate deleting for each week.

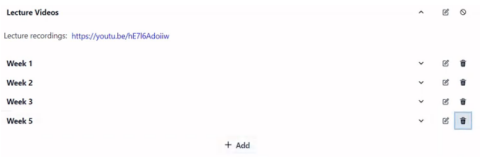

Figure 5.33: Deleting a week

Also, on the right side of each week would also include an editing icon which indicates modifying the week number. Once clicked, this reveals a modal which allows admin users to input a week number to update with.

Figure 5.34: Editing a week

How It Was Implemented

The component was implemented using APIs from the backend that involve retrieving the weeks of a lecture/tutorials for the course, deleting weeks, adding and editing weeks for the lectures/tutorials. Specifying either lectures or tutorials and inputting the course code in the backend would allow for a request that returns the weeks. Also, deleting and editing a week is possible through an input specifying the ID attribute and sending it to the API. Moreover, in order to add a week the admin user must provide the week number and if desired a file input.

Considerations

The adding, editing and deleting weeks functions are only available to admin users, whereas student users are only able to view the weeks. Also, admins may wish to edit weeks that are already created to suit their needs. Additionally, the weeks are sorted in chronological order to aid in readability for users. Also, when an admin user creates a new week where the week number is 1 then that week would appear near the top once rendered.

5.4.4 Files

The files section includes: adding files, deleting files and downloading files for the lectures and tutorials component.

Purpose

The purpose of the files functionality is to allow users to download relevant files, admin users to delete and upload new files to the lectures/tutorials component of the course.

What Was Implemented

In the lectures and tutorials overview page, there are expandable panels which then reveal to users files that relevant to each week in the lectures/tutorials component.

There is also an addition button with an add file description that infers users of its function.

Figure 5.35: Adding a file

There is a bin icon that is on the right side of the relevant file name or row in the panel which would infer to users that this would delete the file from the lectures/tutorials system for the relevant course.

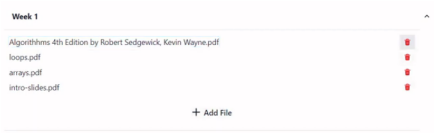

Figure 5.36: Deleting a file

The user is also able to download a file by clicking on the relevant file name or row in the panel.

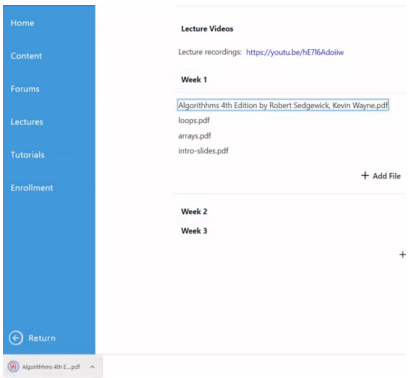

Figure 5.37: Downloading a file

How It Was Implemented

The component was implemented using APIs from the backend that involve: posting new files to the server and deleting files from the backend. In the adding a file, there will be a file attached which is then used for a POST api which saves the file locally and the address is kept into a table in the database. Then when the user wants to download a file, the user simply clicks the link to the file which is a reference to the files physical location on the server. Also, in order to delete a file the user clicks the relevant bin icon for the item which then sends to the backend the ID of the item to remove from the server.

Considerations

There is a consideration where a user may also want to create a week and upload items simultaneously. This would add to efficiency of the feature. Additionally, adding a week that already exists with a file would then upload the file to the relevant week. This allows for admin users that may make mistakes in creating a week with files that already exist. Also, only admin users are able to add and delete files in the lectures/tutorials feature of the course.

5.4.5 File Searching

All users are able to search for a file in the lectures/tutorials sections by typing keywords that match to available files in the system.

Purpose

The purpose of the file searching feature is to allow users to quickly look for any files related to their inputs. For instance, a user may wish to see which files are PDFs in the lectures section of their course.

What Was Implemented

In the lectures or tutorials overview page, there is a search bar at the top that allows for users to input queries which then match to files in the lectures/tutorials system. When the user inputs a query, the overview page will then render a results table depicting all the file matches in relation to the search query.

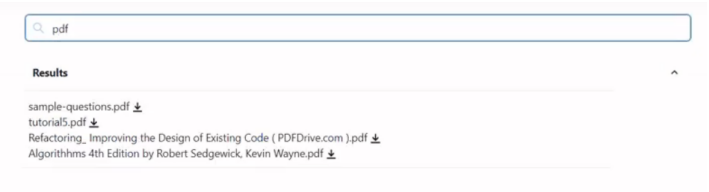

Figure 5.38: Searching for a file

How It Was Implemented

The component was implemented using APIs from the backend that involve searching for a file either in the lectures or tutorials tables in the database. Also, implementing the search bar and its results were possible using ChakraUI and ReactJS. When the

user enters an input search query, the query is sent to the backend which then searches for files that match that query and then output a list of files that have been matched to the frontend. Then the frontend takes this response and then renders it into the results table.

Considerations

There was a consideration for users who may wish to search via the type of file in the system such as PDF or doc. This allows for an extra parameter for searching available for all users. Additionally, in the future there will be an implementation where instead of submitting the search query, the files automatically filter based on keystrokes made by the user.

5.5 Assessments

5.5.1 Creating a quiz

Design

A quiz can be created as a course lecturer or staff member while on a topic group page. That is the intended workflow, however the current implementation is not integrated with the topic group page. A button for adding a new quiz will be visible and once clicked, a modal will appear to initiate the quiz creation:

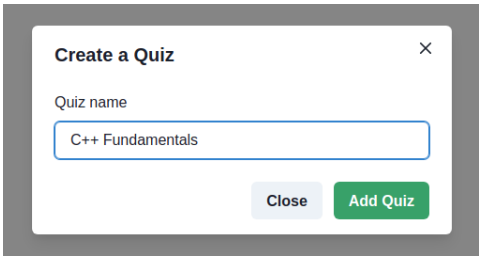

Figure 5.39: Quiz creation modal

After entering a quiz name and clicking the 'Add Quiz' button, the user will be redirected to the quiz creation page where they can start adding questions.

Purpose

Staff members are able to create quizzes quickly and get started on questions right away.

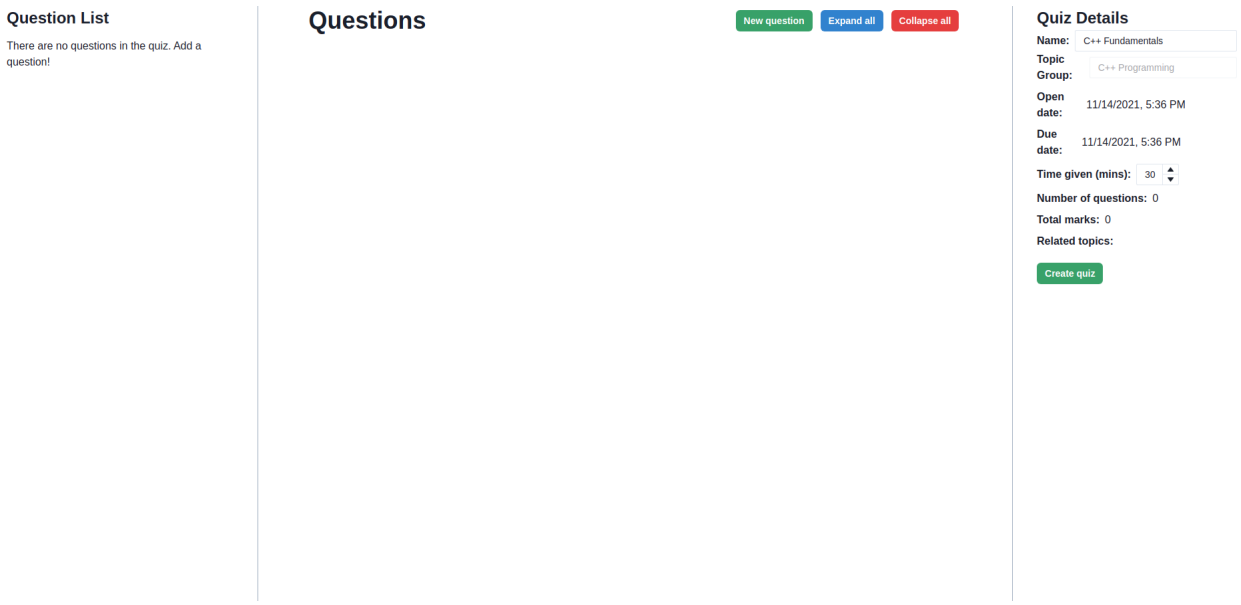

Figure 5.40: Quiz creation page

What was implemented

Staff members can set the important details and settings for a quiz.

How it was implemented

It was implemented using dividers and modals.

Considerations

No considerations were made.

5.5.2 Adding a question

Design

To add a question, the user can click on the 'New Question' button.

A modal will appear, asking the user whether they want to create a new question or import a question that was made and stored in the question bank in a previous quiz.

Figure 5.41: New Question button

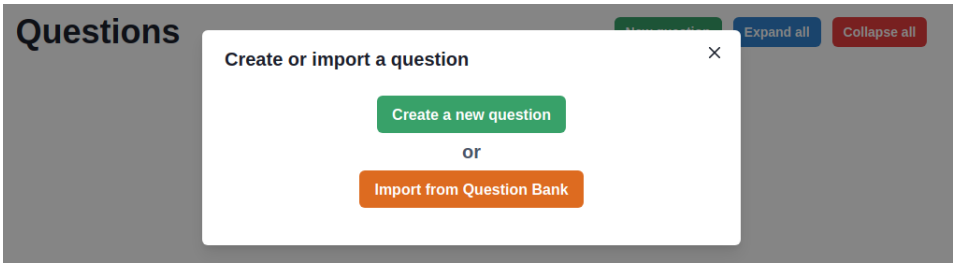

Figure 5.42: Create a question modal

Since the import option is incomplete, the user is only able to create a new question as of now. The modal will now update to show data fields for the new question that must be filled in before they're able to add the question to the questions list that was shown on the page. The user can select which topic the question is related to, the question type (multiple choice, short answer and checkboxes) and also add answers along with explanations if desired.

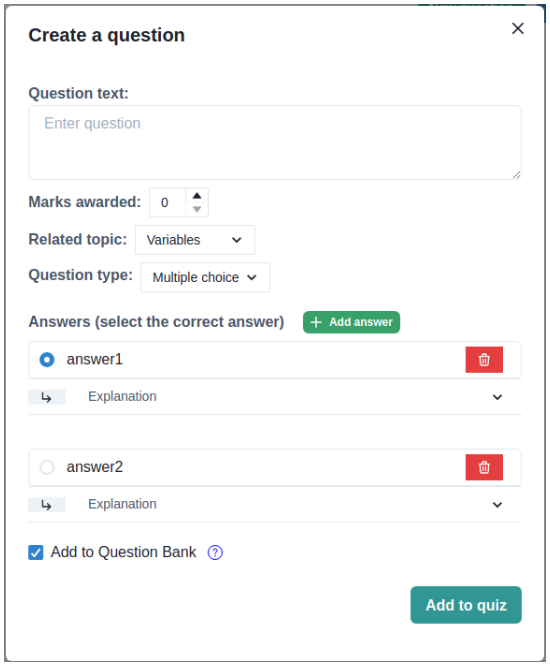

Figure 5.43: New Question data fields

The staff member can add an answer via the "Add Answer" button, as well as delete an answer by clicking on the respective red garbage bin icon on a particular answer. Explanations can be added to an answer if desired. An explanation can be added by

expanding the 'Explanation' accordion under an answer and entering the explanation there.

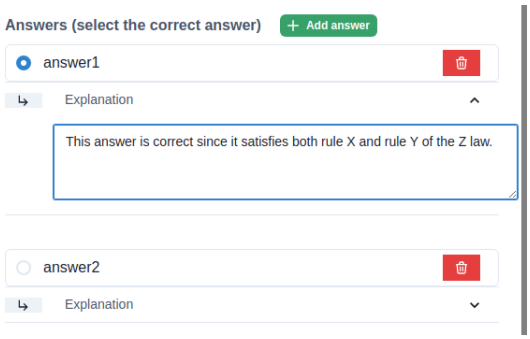

Figure 5.44: Adding an explanation to a possible answer

Once the fields have been filled in, the staff member clicks on the "Add to Quiz" button to add the question to the questions section. By default, "Add to Question Bank" is ticked so all questions will automatically be saved in the question bank for future re-use.

Figure 5.45: Add To Quiz button

The question will appear in the question list and questions section. In the quiz details, the number of questions for a topic is also shown. In the below image, there is 1 question for the topic "Variables".

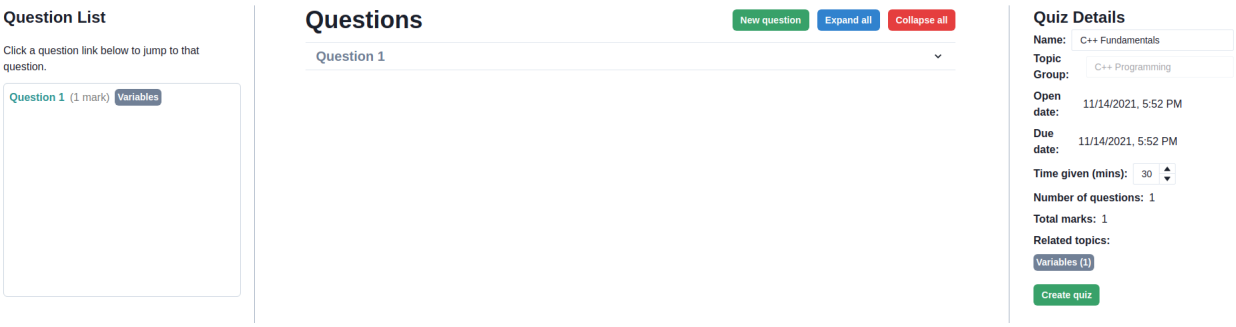

Figure 5.46: Question list and section

To show fields of a particular question, you can expand it by clicking on its header in the question section, or on its question list link. Similarly, you can collapse a question by clicking on a question's header/link while it's expanded. There is also a "Expand all" and "Collapse All" button to expand or collapse all questions.

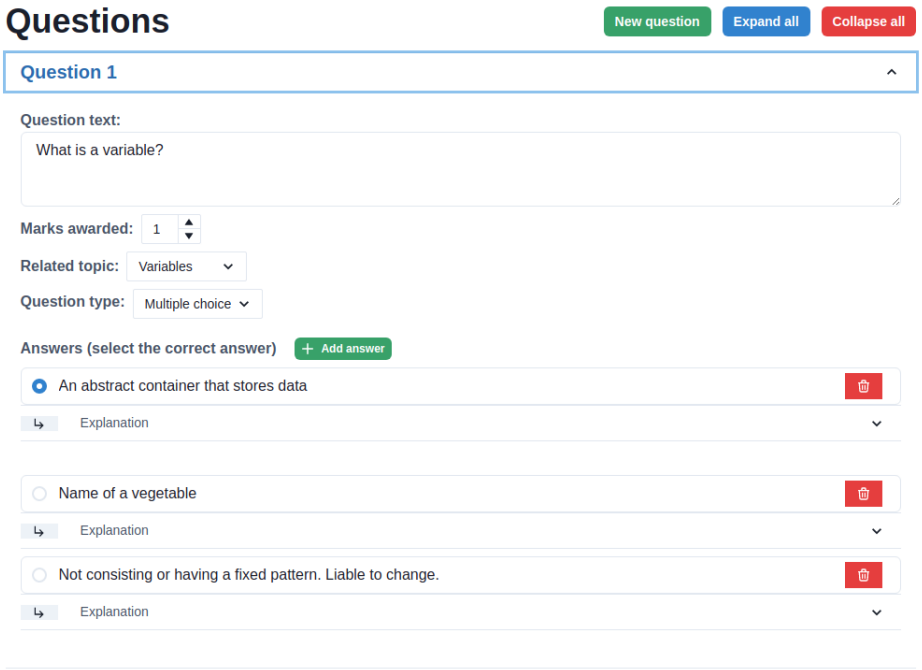

Figure 5.47: Question list and section

Purpose

This feature allows staff members to create questions of different question types and provide explanations for each answer to benefit the students' learnings.

What was implemented

Staff members can create questions of 3 question types - multiple choice, checkbox and short answers. Along with this, they can add a desired number of answers, along with optional explanations to each answer. These explanations describe why the specified answer is correct or incorrect, and will be viewable by students who attempted the quiz when they are on the submission review page for the quiz. These allow the staff members to help students understand the topic or concept better with a more detailed explanation than the lecture slides and can also be used as a 'sticky note' with advice by mentioning something like 'revise on Topic X' or 'refer to Lecture Slide X Page Y for a detailed explanation of this concept'. This saves time for the staff member and allows more immediate help for the students instead of them asking a question on the forums.

The question list displays the marks awarded and topic for each question and allows the staff member to quickly see whether they need to add more questions, keep track of marks for each question, and whether to add more or less questions for a particular

topic.

How it was implemented

The 'Create a Question' modal was created using the radio buttons, checkbox and textboxes provided in Chakra UI. The data structure for a question includes the fields shown in the "New Question data fields" figure, and a list of answers.

Considerations

Key decisions made was mainly around the variety of question types. More question types were planned to be added, such as a fill in page, dragging terms from the left side to their definitions on the right side by connecting a line between them and a coding editor that you can compile your code answer in. This was not done due to lack of time.

5.5.3 Editing quiz details

Design

The design is kept simple, aimed at being more easy to scan for particular fields.

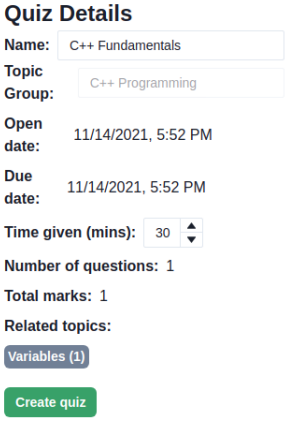

Figure 5.48: Quiz details

Purpose

This allows the staff member to quickly modify quiz details any time while they add or modify questions.

What was implemented

The quiz name, open date, due date and time given can be modified in the Quiz Details column of the Quiz Creation page. Other details such as the number of questions and total marks were also implemented.

How it was implemented

Variations of Chakra UI's box components were used to vertically and horizontally align the quiz details fields.

Considerations

Key decisions made was to keep the quiz details to only contain the most core fields to keep implementation more simple. There were considerations to add more settings.

5.5.4 Answering a question

Design

The Quiz submission page is where students enrolled in a topic group can attempt a quiz that was made for the topic group. From left to right, the columns are the quiz progression column, the question itself, and the timer showing the time remaining for the quiz attempt. The quiz progression column indicates to the student which questions they've attempted or not attempted yet. A blue icon with a question mark represents an unattempted question while a green icon with a tick represents an attempted question. This allows the student to be always aware of which questions they'll need to go back to later. The student can also click on any of the question number icons to jump to that particular question.

The student can answer a question by clicking on one of the options (multiple choice), one or more of the options (checkbox), or by entering text into the textbox (short answer).

Once they've selected an answer, that counts as an attempt of the question and the question icon (in the quiz progression column) for that question will turn green with a tick as shown below.

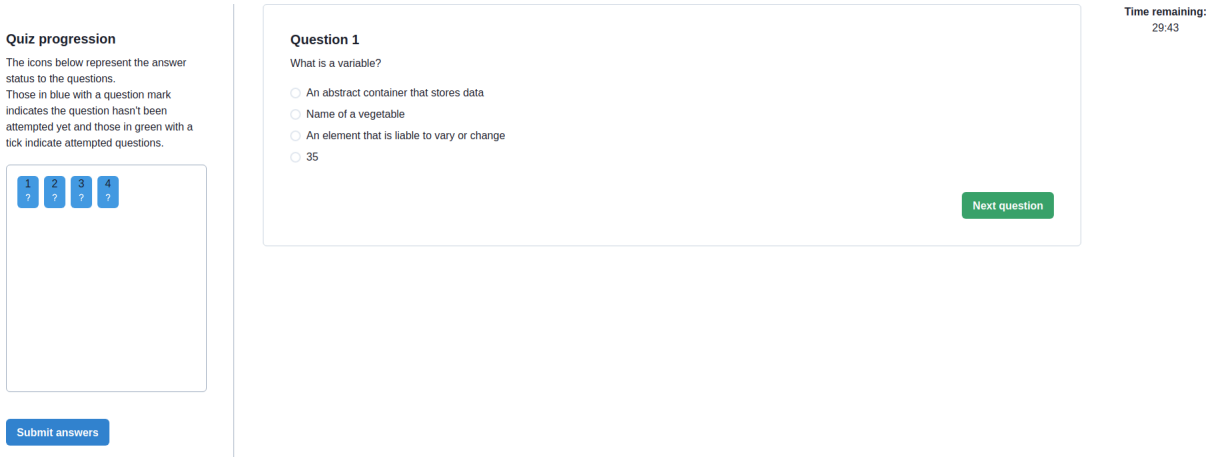

Figure 5.49: Quiz submission page

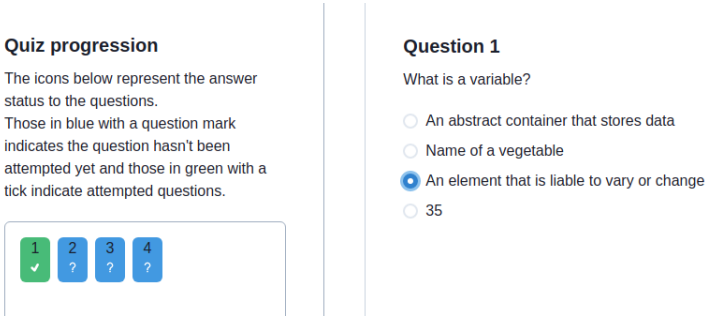

Figure 5.50: Successful question attempt

Purpose

This feature allows students to test their knowledge on a particular topic or topics, as well as allow staff members to know which topics may be good to revise on in future lectures.

What was implemented

The ability to answer questions, a quiz progression tool that allows students to see which questions have been attempted or not attempted as well as allow them to jump to a desired question. A timer is also visible for students to keep track of time.

How it was implemented

This feature was implemented using many box components and the respective components for the three question types.

Considerations

There were considerations to re-design the interface to be different to existing learning management systems' quiz features but decided to stick to the common interface due to lack of time.

5.5.5 Editing an answer

Design

To edit an answer, the student can select another possible answer and their new answer will be the updated answer to the question.

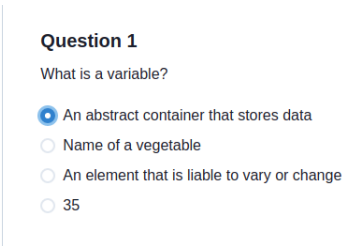

Figure 5.51: Editing an answer

Purpose

This feature allows students to change their mind on an answer for a question at any time.

What was implemented

The ability for the student to change their answer.

How it was implemented

It was implemented by updating the data structure of the answer in a student answer object in JavaScript.

Considerations

No considerations were made.

5.5.6 Submitting an attempt

Design

Once the student is satisfied with their submission, they can click on the "Submit answer" button below the Quiz Progression section to submit their quiz submission.

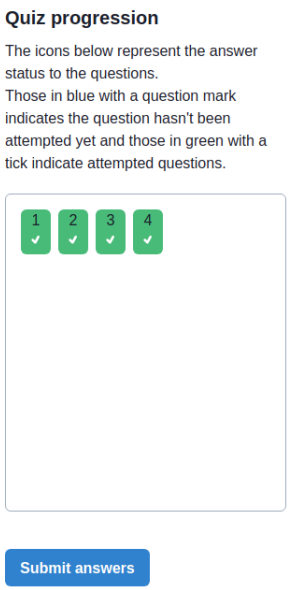

Figure 5.52: Submitting answers button

A confirmation modal will appear before the quiz submission is submitted and no changes can no longer be made to the attempt.

After pressing the "Submit" button, a submission confirmation message will be shown.

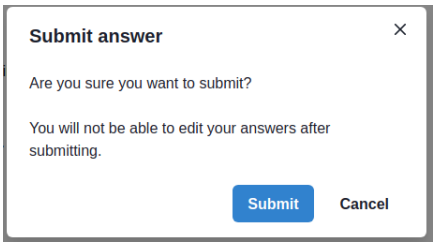

Figure 5.53: Submit quiz attempt confirmation modal

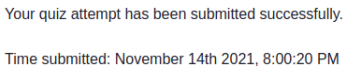

Figure 5.54: Successful quiz submission message

Purpose

This feature allows students to submit their quiz attempt and allow the attempt to be marked automatically (for multiple choice and checkbox type questions), and manually for any short answer type questions that exists in the quiz.

What was implemented

A button to open the 'Submit answer' modal and a confirmation message.

How it was implemented

It was implemented using Chakra's modal components.

Considerations

No considerations were made.

5.5.7 Viewing correct answers against their answers

Design

After a student has submitted their quiz attempt, their attempt has been marked and the due date has passed, the student can review their quiz submission on the Quiz

Review Submission page.

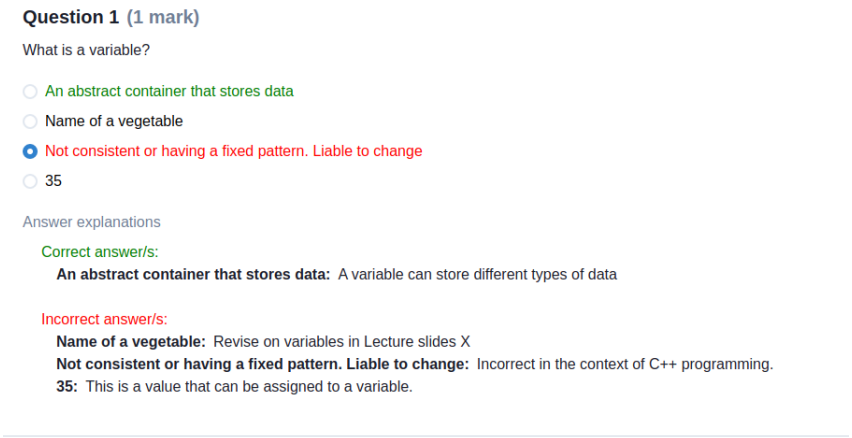

Figure 5.55: Review submission page

The student can compare their answers with the correct answers. If they answered a question incorrectly, their selected answer's text will turn red and the correct answer's text will turn green.

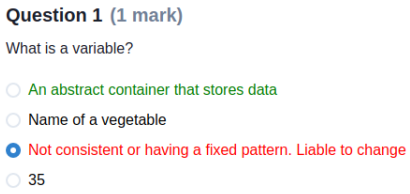

Figure 5.56: Compare correct answers against a student's answers

Purpose

This feature allows students to compare their answers to the correct answers.

What was implemented

Highlighting of the correct and incorrect answers texts (if the student selected an incorrect answer), as well as the answer explanations were implemented.

How it was implemented

It was implemented using the existing implementations of the question format displayed for the Quiz submission page, and the text colours were modified.

Considerations

Considerations were made to re-design how to highlight which was the correct answer against the student's selected answer and whether their answers were incorrect or correct.

5.5.8 Viewing an explanation of the correct answer and incorrect answers

Design

Below the possible answers will be the explanations, if the staff member has written any. Explanations for all the answers will be shown. This allows the student to understand why a particular answer is wrong or right.

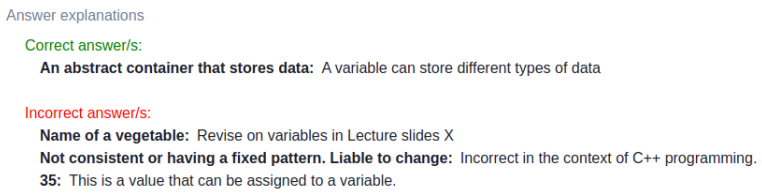

Figure 5.57: View explanations for each answer to a question (if provided)

Purpose

This feature helps students understand why their answer is right or wrong.

What was implemented

The feature allows the students to read any explanations if they want to revise on particular topics or want to understand why an answer is correct or incorrect. The staff member is free to mention a particular lecture slide that the students can go onto to revise on the particular topic the question targets.

How it was implemented

The answer explanations are grabbed from the explanation strings stored in the answer objects of a question made during the quiz creation.

Considerations

Considerations were made to re-design the layout of the answer explanations as it somewhat blends in with the question, making it hard to differentiate and less readable.

5.5.9 Overview of User Interface Designs

The user interface is what the user relies on to complete tasks and thus it is important that the design is usable and easy to use. A few changes were made from the initial designs to support the added content that was not planned in earlier stages of the Assessments feature.

Final Designs

The final design had more emphasis on possible tools that'd improve the user experience and was added to reduce the amount of time the user needs to spend on their tasks:

1. Quiz Creation page: contains the questions list and the expand/collapse functionality to allow the user to focus on particular questions
2. Quiz Usage page: contains the quiz progression column that allows them to keep track of which questions they've attempted or not attempted as well as the ability to navigate to any question with one click
3. Quiz Review Submission page: highlights incorrect answers vs. the correct answers and shows all explanations (that were provided by the quiz creator) for the possible answers

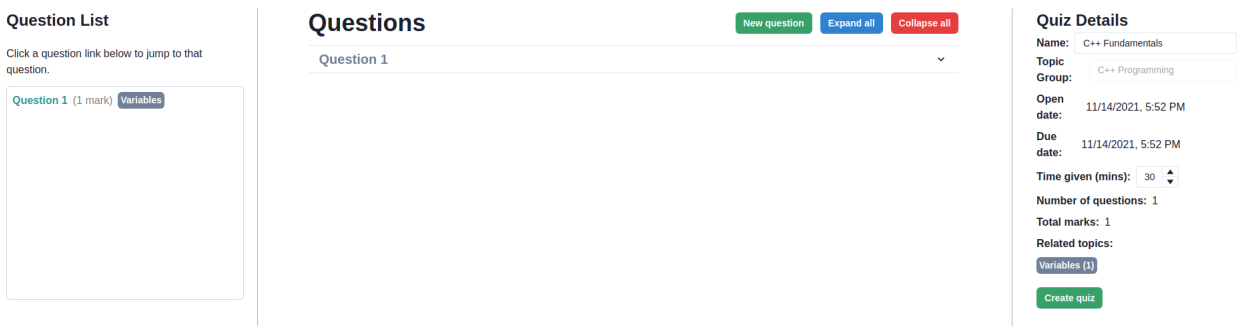

Figure 5.58: Quiz creation page

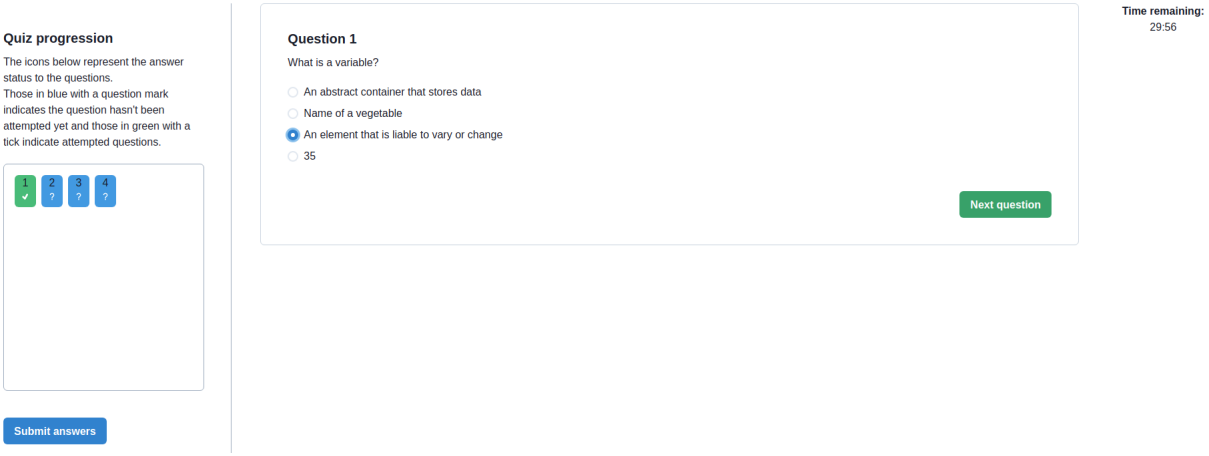

Figure 5.59: Quiz usage page

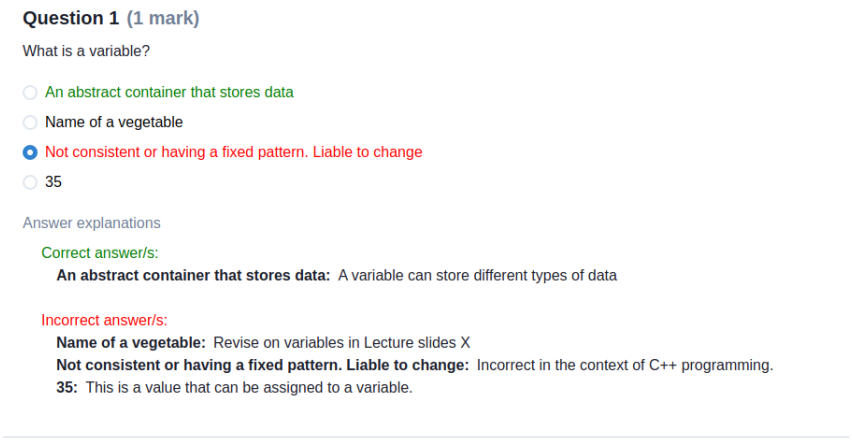

Figure 5.60: Quiz review submission page

Changes Made From Initial Designs

There were significant changes to the initial designs in terms of button layouts and colour schemes, however the core content displayed is the same. Additional data that was not foreseen or thought of during initial designing was added to the user interface design. Furthermore, extra tools were added for convenience and to improve the user experience.

Below, the initial and final designs next to each other to see the similarities and differences.

Question creation:

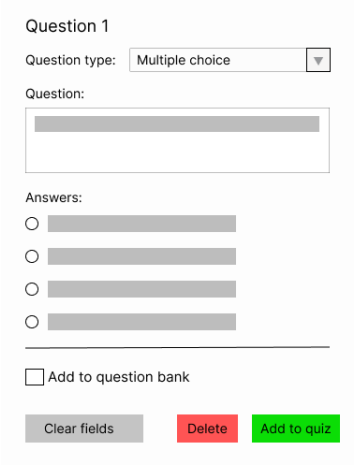

Figure 5.61: Initial design of question creation

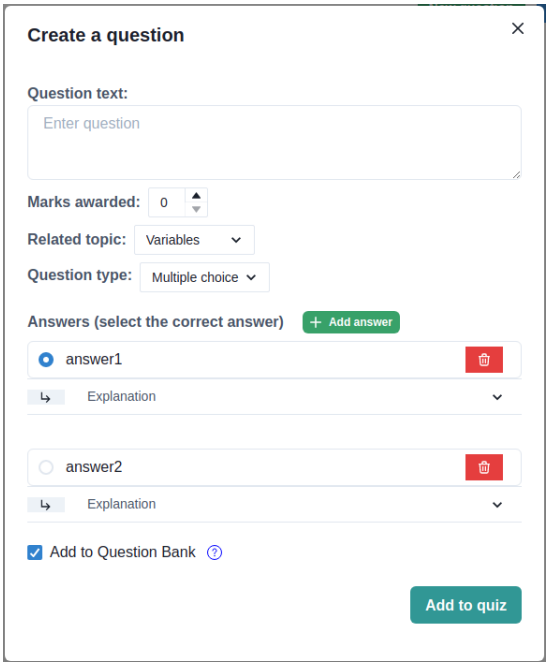

Figure 5.62: Final design of question creation

Quiz usage:

Figure 5.63: Initial design of quiz usage

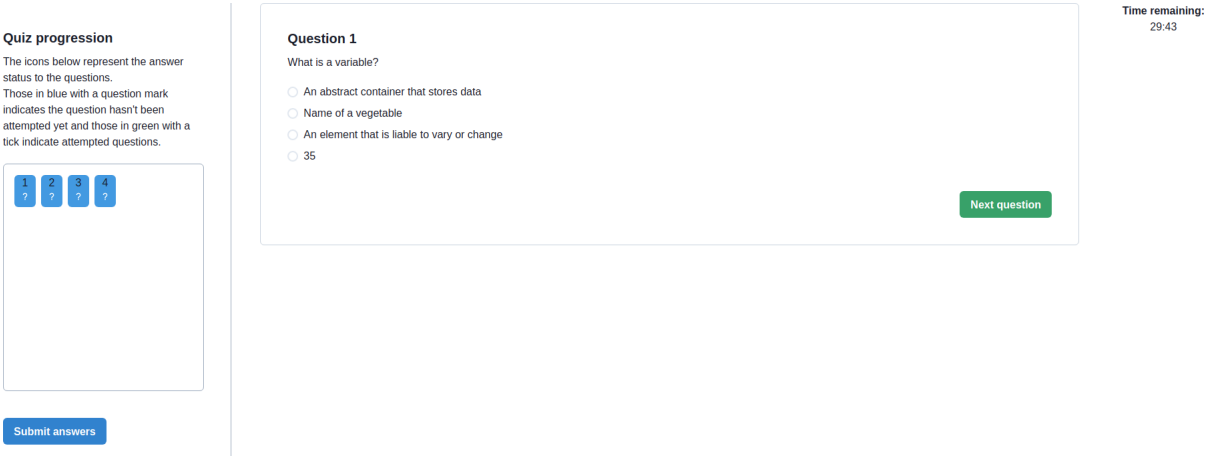

Figure 5.64: Final design of quiz usage

5.6 Forums

5.6.1 Overview Page

The forums are accessible from the sidebar of a course page. Each topic group has a dedicated forum.

When first opening the forums, the user would see the forum overview page.

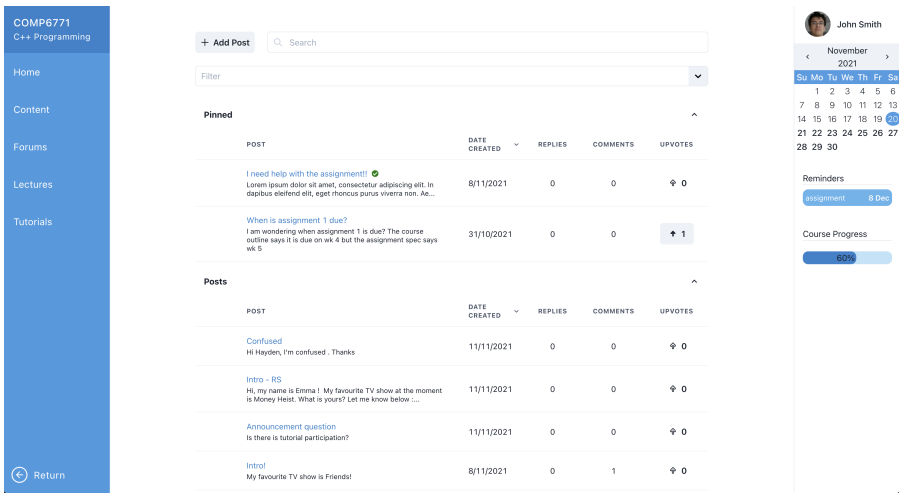

Figure 5.65: Forum overview page for a student.

Purpose

Shows users an overview of all the posts that have been posted on the forum page.

What Was Implemented

The forum overview page is split into two different tables, the pinned posts and all posts in reverse chronological order. Within each table, there are columns for the post title and description, date created, number of replies, number of comments, and upvotes. Posts that are endorsed are shown with a green check mark. A user can change the post order by clicking the heading of each table. This will toggle the column's order between decreasing and increasing. At the top of the forum overview page, there is a button to add posts, a search bar and a filter menu.

How It Was Implemented

The system uses the course code that is in the url of the site to get the correct forum data from the backend. When the page loads, two backend calls are made to populate the post tables - one to grab all the posts, and another to grab the pinned posts. The React library, `react-table` was used to add the sorting functionality. It takes a list of column objects that define the heading and contents for the column.

Considerations

Instead of having a single post list with the pinned posts marked with a pin icon to highlight that it was different, a separate pinned post table was built. This is to bring more attention to the pinned posts and make them easier to find.

The post tables were made collapsible to prevent users having to scroll past many pinned posts in order to see the rest.

5.6.2 Add Post

Posts can be added using the call-to-action button on the top left corner of the forum overview page. Clicking this button will open up the add post modal.

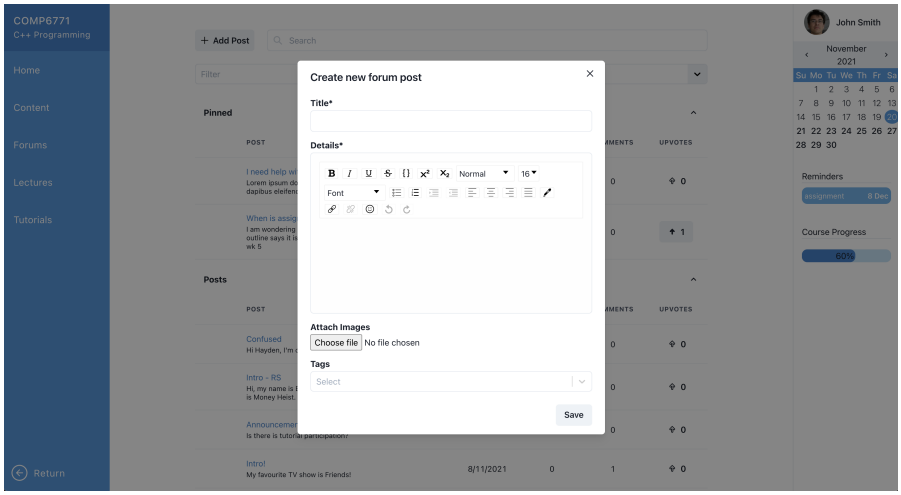

Figure 5.66: Add post modal.

Purpose

Allows any user to easily add their own forum post to ask a question or make a comment.

What Was Implemented

The modal that shows when a user clicks the "Add Post" button allows the user to input a title and description for their post. They can also attach an image, as well as select tags that apply to the post. The tags help with categorising and filtering the posts.

How It Was Implemented

The modal shown is part of the `chakra-ui` library. For the details section, the React package `react-draft-wysiwyg` was used so that the users had a rich text editor to interact with. The attachments use Javascript's built-in `input` element and file handling. When the modal opens, a backend call is made to pull the tag data for the current course page. The list of tags are passed into the `react-select` component as options and the `isMulti` attribute is set to allow users to select multiple tags for their post. As the fields for title, details, images and tags are populated, the inputs are stored and sent to the backend when the user clicks "Save". Saving the post takes the user directly to the post page.

Considerations

A modal is used, instead of navigating users to a new page, to keep the add post flow as simple as possible.

For the rich-text-editor, the `draft-js` package was originally used because of its various plugins, including allowing users to use markdown syntax when typing however, the switch to `react-draft-wysiwyg` was made since it was easier to work with and already had the required formatting options built in by default. In doing this, the markdown syntax option was lost however, the tradeoff was worth it since `react-draft-wysiwyg` provides the user with far more formatting options.

Due to time constraints, only images can be attached to a forum post. Since screenshots are typically used to assist when asking a question on forums, it was decided that it was more useful to at least have the option to attach an image than not having any attachment options at all.

5.6.3 Search and Filter

The search and filter options sit at the top of the forum overview page, making it easily accessible for users.

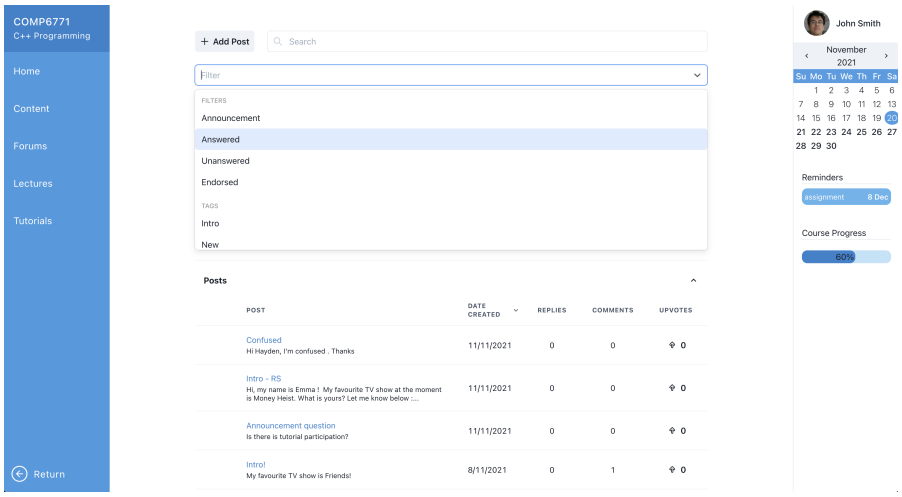

Figure 5.67: Search bar and filter options

Purpose

Allows users to easily search for a specific post or filter through posts.

What Was Implemented

The search bar is a simple input field that the users can use to search through posts. The search queries the title and descriptions of each post. The search results are shown in a table and the "Pinned" posts table is hidden.

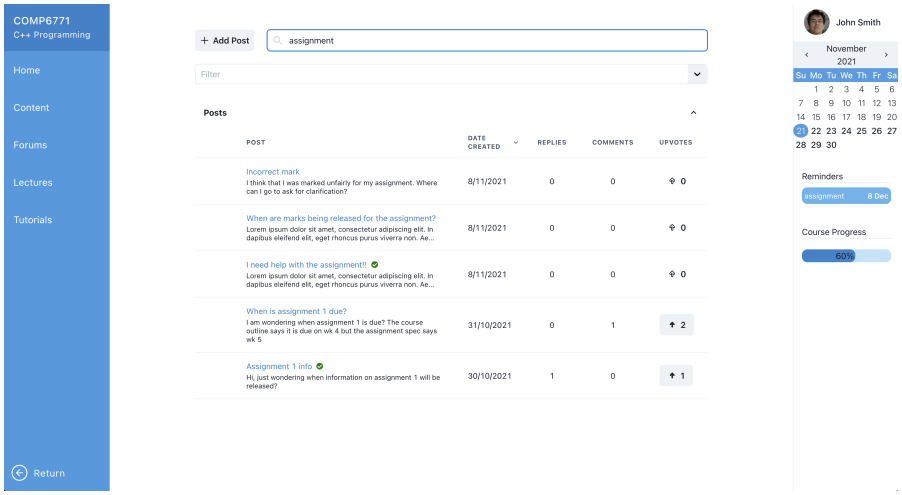

Figure 5.68: Search results when searching for "assignment"

Users have two options to filter the posts by, they can either use the tags that have been defined by course staff, or they can filter by the pre-defined filters. The pre-defined filters consist of posts linked to announcements, answered posts, unanswered posts and endorsed posts. A user can also use one or more tags when filtering the posts. A user can either select their filter in the menu or by typing the filter into the filter bar. Similar to search, the filter results are shown in a table and the "Pinned" posts table is hidden.

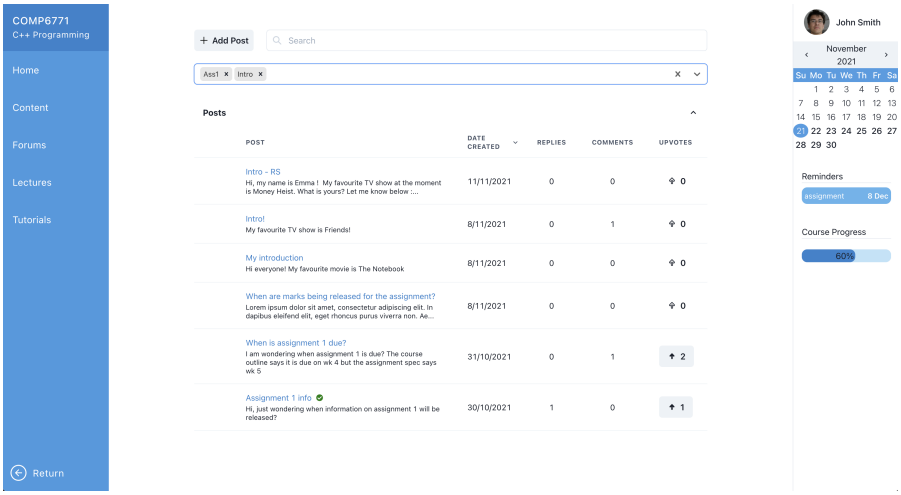

Figure 5.69: Filter results when filtering by tags "Ass1" and "Intro"

How It Was Implemented

When the user types a search term and hits "enter", the search term is passed to the backend where a query to the database is made. Since it directly queries the database, the search functionality is very basic. If a user inputs more than one word, only posts that have a title or description that contains the exact search term will be returned. There is also a small bug where the user has to clear the search bar and hit "enter" again in order to re-render the original posts.

The filter uses the same `react-select` library mentioned previously. If the user chooses to filter by tag, then the backend simply returns a list of all posts that have the corresponding tag. When filtering by multiple tags at once, the posts returned from the backend are those that have any of the tags chosen. For filtering by posts linked to announcements, the backend looks for posts with the "Announcement" tag. For the remainder of the pre-defined filters, the posts aren't explicitly tagged with them so there is logic built into the backend to handle these requests. When filtering by "answered" or "unanswered" posts, the backend queries the number of comments and replies each post has. For endorsed posts, the backend looks for posts with the `isEndorsed` flag.

Considerations

Instead of being hidden in a menu on the side or top of the forums, the search and filter options were placed at the top of the page to make it easy to find and interact with. Due to time constraints, the search bar bug was unable to be fixed however, the search functionality itself isn't impacted too much by this. A drop down menu was used for the filter to assist users who aren't quite sure what they're looking for however, the option to type directly into the bar is also available for users who know what they want.

5.6.4 Post Page

A user can click on a post title on the forum overview page to open the individual page for that post.

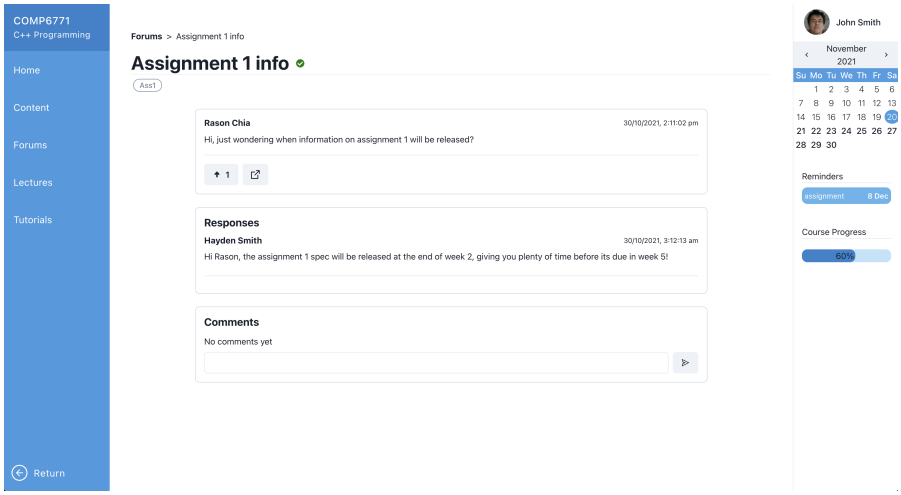

Figure 5.70: Forum post page.

Purpose

Allows users to see all the post details, replies and comments in one place. From this page, users can interact with the post by upvoting or sharing. Upvoting allow users to like posts to draw attention to them. The share button makes it easy for users to send a direct link to the post to others. Depending on whether or not a user is staff, they can also leave a response and/or comment.

What Was Implemented

The top of the post page has links to navigate back to the forum overview page. Underneath that is the post title and tags. If the post has been endorsed by staff, there

will be a green check mark next to the post title. If users are unsure of what the green check mark means, they can hover over it for an explanation.

Figure 5.71: Tooltip explaining endorsed icon.

The post page is then split into three sections. The first is the post details which contains the author, date and time created, and post description. There are also buttons to upvote and share the post. The second and third sections contain all the responses and comments for the post. Similar to the post details section, each response and comment also displays the author and timestamp.

How It Was Implemented

When navigating to the post page, the `id` of the post is used to display the correct data from the backend. This same call is also used to populate the responses and comments for the post.

When a user clicks the upvote button to like a post, a backend call is made to increment the value and add the user to the list of upvoters. This ensures that users can't upvote the same post more than once. It also allows the frontend to rerender the button with the updated number of upvoters, and change the icon to the filled in arrow. Similarly, the user can click the upvote button to remove their upvote which makes a backend call to decrement the value and remove them from the upvoters list.

Figure 5.72: Upvote buttons that have and haven't been liked.

If a user clicks the share button, the url of the page is directly copied to their clipboard. A toast element shows to confirm that the link has been clicked

Figure 5.73: Toast to confirm that post link has be copied.

Considerations

Initially, an endorsed post was marked by the messaging *"This post is endorsed by staff"* which showed under the post content.

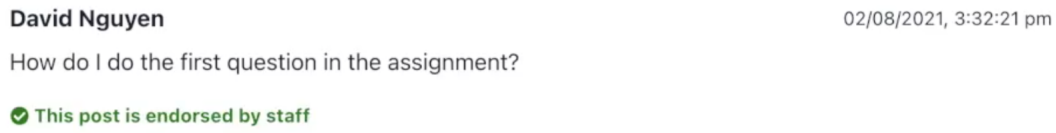

Figure 5.74: Old endorsed messaging on forum posts.

There was a concern with this taking up too much vertical space so now, the messaging has been removed and only the endorsed icon shows next to the post title. The tooltip for the endorsed icon was added however, to ensure that a user could easily find out what the symbol meant if they were unsure.

By splitting the page into three sections with obvious borders, users can easily scan the post page and find what they're looking for. It was an early design decision to create separate responses and comments section. The idea was to only allow staff members to post in the responses section. This is to make it easy for students to find exactly where the most reliable answer to a question may be. It also ensures that staff members can find what posts haven't been replied to and require their attention. The comments section gives students a place to answer a post themselves or leave a follow up question.

It was decided that the upvote button would only show the number of upvotes, and not the names of the upvoters, so that students can remain anonymous when liking a post. Hopefully, students will feel more comfortable using this feature since no one will know exactly who upvoted a post.

5.6.5 Edit and Delete Post

If the logged in user is the author of the post, they will see additional buttons on the post page to edit or delete their post.

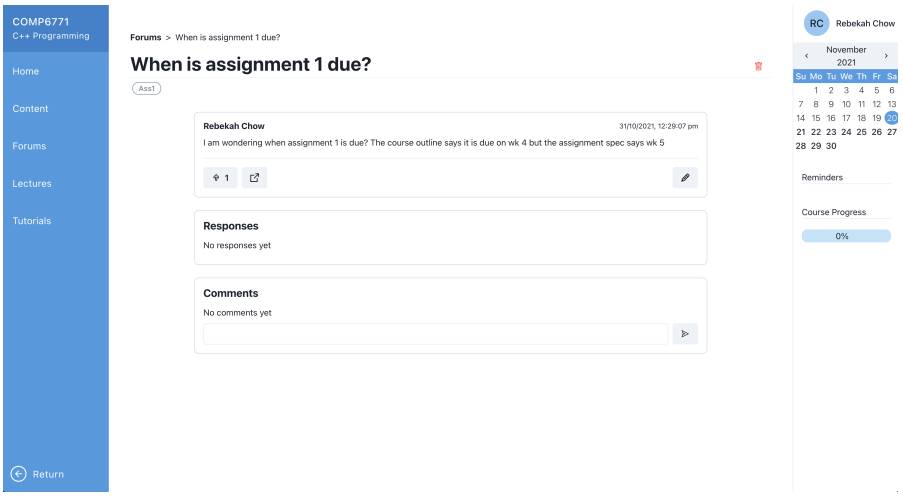

Figure 5.75: Forum post page when the logged in user is the author of the post.

Purpose

To allow users to easily make changes to their post or delete their post.

What Was Implemented

When viewing one of their own posts, a user would be able to see the delete icon in the top right corner. Clicking this button will show a confirmation message to ensure that the user definitely wants to delete their post. Once confirmed, the post will be deleted and the user will be navigated back to the forum overview page.

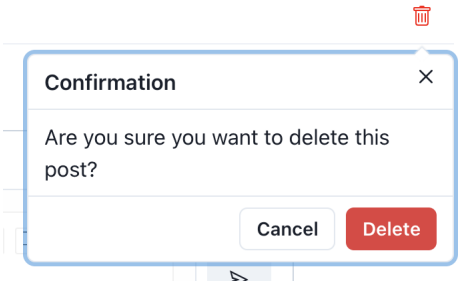

Figure 5.76: Confirmation messaging for deleting a post.

To edit a post, there is an edit button on the bottom right of the post details section of the page. Clicking this button will show the rich text editor and allow users to edit their post description. Saving the changes will automatically update the page.

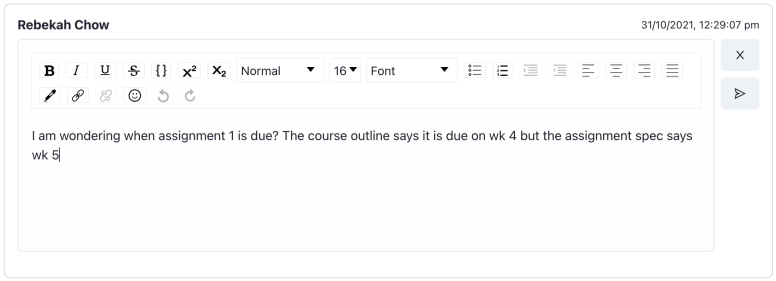

Figure 5.77: Interface to edit post.

How It Was Implemented

When deleting a post, a backend call is made to remove it from the database.

When editing a post, the draft editor component will be initialised with the existing post content. As the user types, the state of the editor will be updated so that when the user saves their change, the backend will be updated immediately. This ensures that the frontend is also updated instantly.

Considerations

The confirmation message for deleting was added to ensure that posts weren't accidentally deleted because of incorrect clicks.

It was decided that the editor would open in the post details section when editing a post so that users aren't navigated elsewhere to make changes, providing the simplest user experience.

5.6.6 Comment

A user can click the input box in the comments section to open the rich text editor and leave a comment.

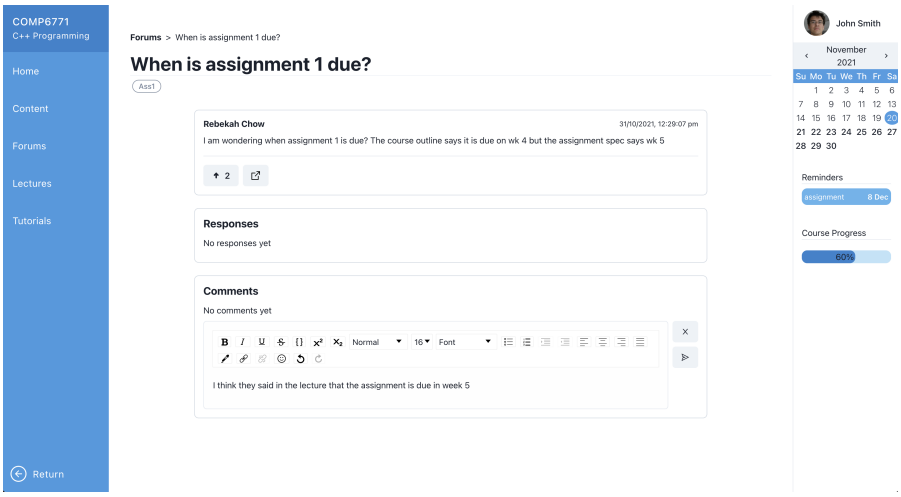

Figure 5.78: Leaving a comment on a forum post.

Purpose

Gives students a dedicated area to answer forum posts themselves or leave follow-up questions.

What Was Implemented

By default, the page shows a simple input field in the comments section. When a user clicks on it, it opens the rich-text editor so that they have formatting options for their comment. A user can either close the rich-text editor to cancel leaving a comment, in which case the rich-text editor will close and be replaced with the simple input field. Otherwise, the comments section will automatically be updated with the user's comment when they hit "Save".

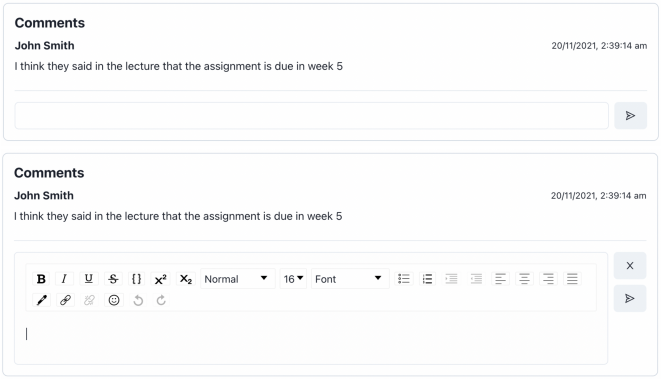

Figure 5.79: Clicking on the input field will open the rich-text editor.

If the logged in user is the author of a comment, they will see additional buttons that allow them to edit or delete a comment.

Figure 5.80: Comment with edit and delete options.

This works very similarly to editing and deleting a post. Editing a comment will open a rich-text editor with the existing comment so user's can make changes. Deleting a comment will ask for the user's confirmation before deleting the comment from the database.

How It Was Implemented

Similar to editing a post, the draft editor component is initialised when the simple input field is clicked. The user's input is stored as they type and sent to the backend once they hit "Send". This causes the frontend to re-render so that the comment instantly shows on the post page.

Additionally, the implementation for editing and deleting a comment is the exact same as editing and deleting a post.

Considerations

The flow for editing and deleting a comment was kept as similar to editing and deleting a post as possible to reduce the amount of learning a user has to do when first interacting with the system. By keeping the implementation relatively the same, and using icons that are typically associated with editing and deleting, users can easily work out how to complete these actions.

5.6.7 Manage Tags

Users who have a staff account will see a "Manage Tags" button on the top right corner that, when clicked, opens up the "Manage Tags" modal.

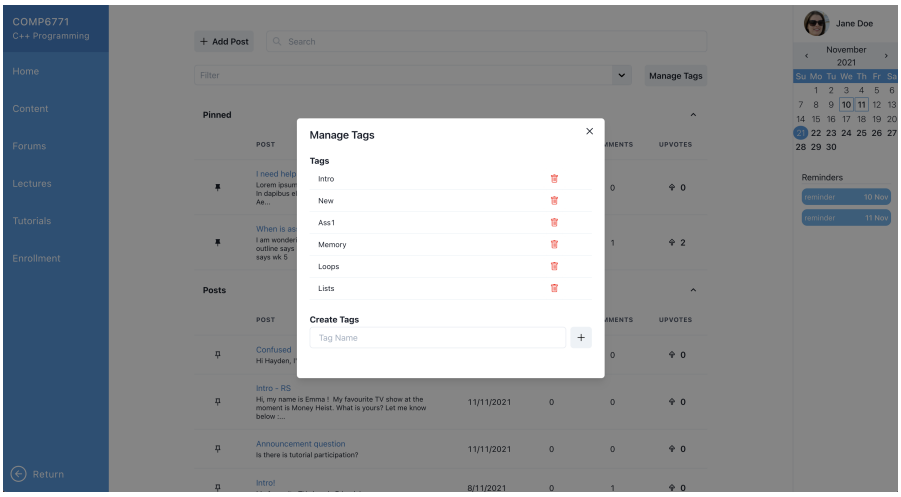

Figure 5.81: Manage tags modal.

Purpose

Allows staff users to see all the tags for a course, create new tags and delete tags. Tags are useful for categorising and filtering through posts.

What Was Implemented

The "Manage Tags" button sits at the top of the forum overview page, next to the filter. When clicked, it opens the "Manage Tags" modal which contains a list of all the tags that have been created for the topic group. Next to each tag is a delete button that allows staff to delete the tag. At the bottom of the modal, staff can type the name of a new tag and click the "+" to add it to the list.

How It Was Implemented

When the modal is opened, a backend call is made to get a list of all the tags. These are then displayed in a simple `table` component alongside a delete button for each tag.

Deleting a tag follows a similar flow to deleting a post or comment. When the button is clicked, a confirmation message is displayed to ensure the user doesn't accidentally delete a tag. Once confirmed, the tag is deleted from the database and removed from this list.

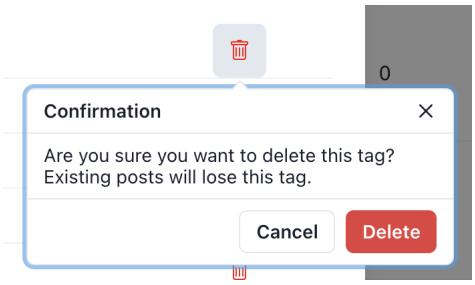

Figure 5.82: Deleting a tag.

A staff user can create a new tag by typing its name into the input field and clicking "+". This makes a backend call that adds the tag to the database and links it to the current topic group. If a staff user tries to create a duplicate tag, the backend will return an error and the user will be informed.

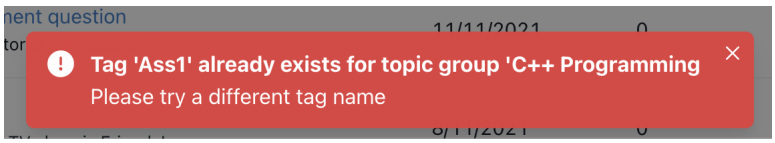

Figure 5.83: Error messaging when trying to create a duplicate tag.

Otherwise, a toast will show to confirm the tag has been created.

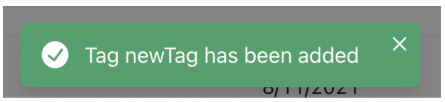

Figure 5.84: Confirmation messaging when creating a new tag.

Considerations

Similar to adding a post, a modal was used for managing tags to avoid users being navigated to a new page. This keeps the flow as simple as possible.

In order to prevent the tags from becoming a bloated system and ensure that the functionality is useful for users, it was decided that only staff are allowed to create new tags. This way, course staff don't have to worry about students creating unnecessary, inappropriate or redundant tags.

5.6.8 Pinned Posts

Staff users are able to pin posts to the top of the forum overview page by clicking the pin icon in the posts table.

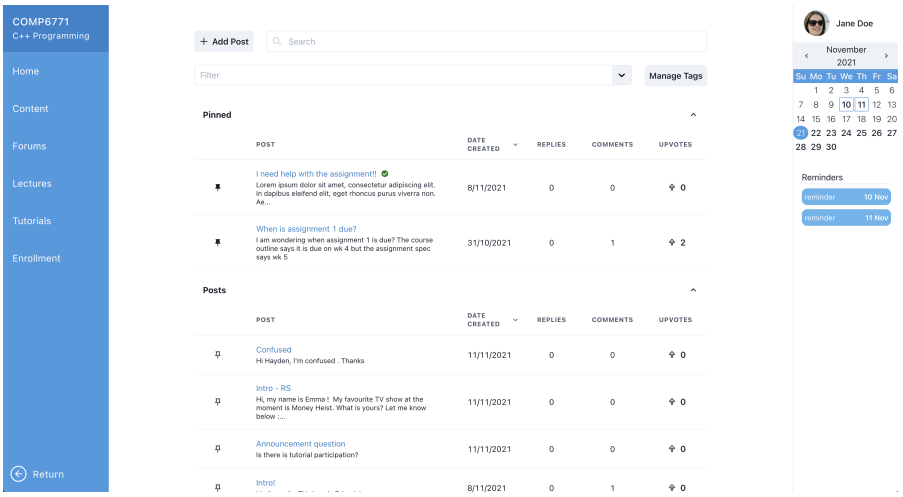

Figure 5.85: Forum overview page for a staff user.

Purpose

Allows staff to highlight certain forum posts as important. This assists in bringing these posts to the students' attention.

What Was Implemented

Each post in the table has a pin icon next to it. When a staff user clicks the pin icon for a post in the "Posts" table, that post will be added to the "Pinned" posts table. The pinned post will also continue to show in the regular "Posts" table but the pin icon will change to a filled pin icon.

Similarly, if a user clicks a filled pin icon, the post will be unpinned and removed from the "Pinned" post table.

How It Was Implemented

When a staff user clicks the button to pin a post, a backend call is made to set the `isPinned` flag for that post. This triggers the frontend to re-render so that the post immediately shows in the "Pinned" table.

Similarly, when a staff user clicks the button to unpin a post, the `isPinned` flag is set to false and the frontend is re-rendered to reflect the change.

Considerations

When a post is pinned, it is still listed in the "Posts" table, but with a filled pin icon. By keeping the pinned post in the "Posts" table, it maintains the chronological order of all the posts. This is mainly to ensure that no post is missed if a user is simply scrolling through the forum to catch up on new posts.

5.6.9 Reply

Staff users can reply to a post in the responses section of the post's individual page.

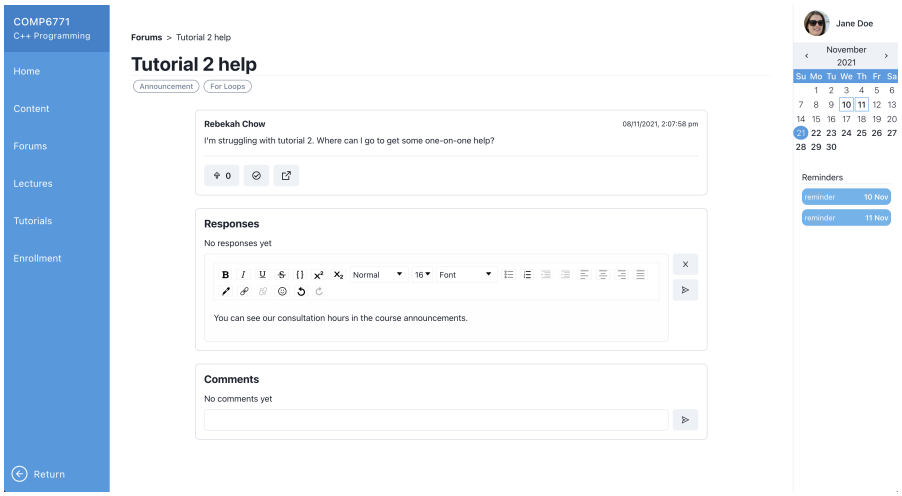

Figure 5.86: Leaving a reply on a forum post.

Purpose

Gives staff users a dedicated area to leave a response to a forum post.

What Was Implemented

Similar to comments, staff can click on the input field in the responses section to open the rich-text editor and leave their response. The rich-text editor allow staff to embed a link to specific resources or course material in their response.

Authors of a response will also be able to see buttons for editing and deleting their reply.

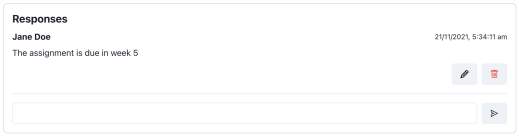

Figure 5.87: Reply with edit and delete options

For non-staff users, there is no input field visible for the responses section.

How It Was Implemented

Similar to posting a comment, clicking on the input field triggers the editor to be initialised. When the user clicks "Send", a backend call is made to store the reply in the database, and the frontend is re-rendered to immediately display the response.

Editing and deleting a reply works exactly the same way as editing and deleting posts and comments.

Considerations

For the reasons behind separating responses from comments, please see Section 5.6.6.

5.6.10 Endorsed Posts and Comments

From the post page, staff users are able to endorse a comment or the whole post by clicking the endorse button.

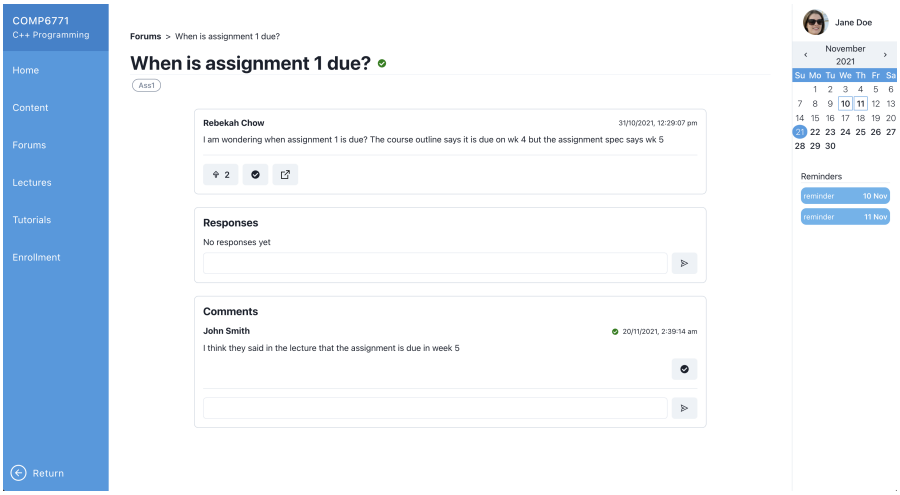

Figure 5.88: A post and comment that has been endorsed by staff.

Purpose

For endorsing a post, it gives staff another way to mark a forum post as important.

For endorsing a comment, it allows staff to verify a student's answer to a question so that other users know that the answer is correct.

What Was Implemented

In the post details section of the post page, staff users will see an additional button that has a check mark icon. Clicking this button will allow staff users to endorse a post, as denoted by the green check mark next to the post title.

Similarly, a staff user can endorse a comment on a post page by clicking the endorse button for that comment.

For more information on the endorse icon for posts and comments, please see Section 5.6.4.

How It Was Implemented

When a staff user clicks on the endorse button, it makes a backend call to set the `isEndorsed` flag to true for the current post. This triggers the frontend to re-render so that the endorsed icon immediately displays. The icon on the endorsed button is also filled in to signify that the post has already been endorsed.

When clicked again, the endorsed button will return to the unfilled icon and the endorsement will be removed from the backend. The endorsed icon on the post or comment will be hidden immediately.

Figure 5.89: Endorsed buttons that have and haven't been endorsed.

Considerations

For the reasons behind how the endorsed icon is displayed, please see Section 5.6.4.

It was decided that the ability to endorse responses was not required since they can only be left by staff in the first place and therefore, are already verified.

5.7 Gamification Walkthrough

5.7.1 Definition of Levels in Gamification

Purpose

Level is a playable set of questions with content from a single topic that students can play through and earn points. It also allows staff to customise questions, question type, points available and time allowed for each question.

What was Implemented

Levels are sets of questions that consist of content from a specific topic within a topic group. Levels can be categorized as Knowledge, Practice and Challenge with increasing difficulty and point earning potential respectively. Levels are built to be reusable and customisable by any staff on the platform.

How it was Implemented

Levels store general info like name of level, type of level, week it is visible as well as an array of questions in the backend. We can create a level through our backend api built that takes in all the information and creates a level entity stored in postgres database. In gamification, Level is directly related to a topic on a one to one basis, a level should only have content from a single topic and a topic group will encompass multiple levels which are content from multiple topics.

Considerations

- Levels should be customisable, editable after creation and performant as the platform could be retrieving many levels at once.
- Tradeoff between time allocated for entire level vs time allocated per question Time limit per question increases difficulty and will generate more competition compared to the time limit for the entire level. I have decided to implement the time allocated per question to increase customizability. As points are awarded based on correct answers, this decision also increases the value of points on the platform.

5.7.2 Definition of Topic Groups in Gamification

Purpose

Topic Groups hold information on topics created within the topic group. Topic Groups are read from the wider metalms to present appropriate content to users in the gamification platform.

What was Implemented

In gamification, topic groups are pulled from the metalms and enable the system to identify students assigned to topic groups and levels (topics) created within the topic group. With this relationship between topic groups and levels (which are a set of questions on a topic), admins can easily add or remove topics to topic groups and in doing so change which students have visibility of the levels.

How it was Implemented

Staff will first select a topic group from a list retrieved from metalms. The staff must then add start and end dates for the term that the topic group will run in. Once this is done, any levels assigned to this topic group will be assigned to all the students that are enrolled in this topic group. For students, the platform retrieves topic groups assigned to them, this is pulled through the backend api from the enrolment table in DB. Using the topic group id, the platform retrieves all levels associated with the topic groups. The frontend will then filter and only display the levels to users that are visible according to its visibility setting customised by admins.

Considerations

- Levels should be customisable, editable after creation and performant as the platform could be retrieving many levels at once.

5.7.3 Accessing Gamification from metalms

From the homepage of the metalms, students and admins can get to the gamification feature by clicking Gamification on the sidebar.

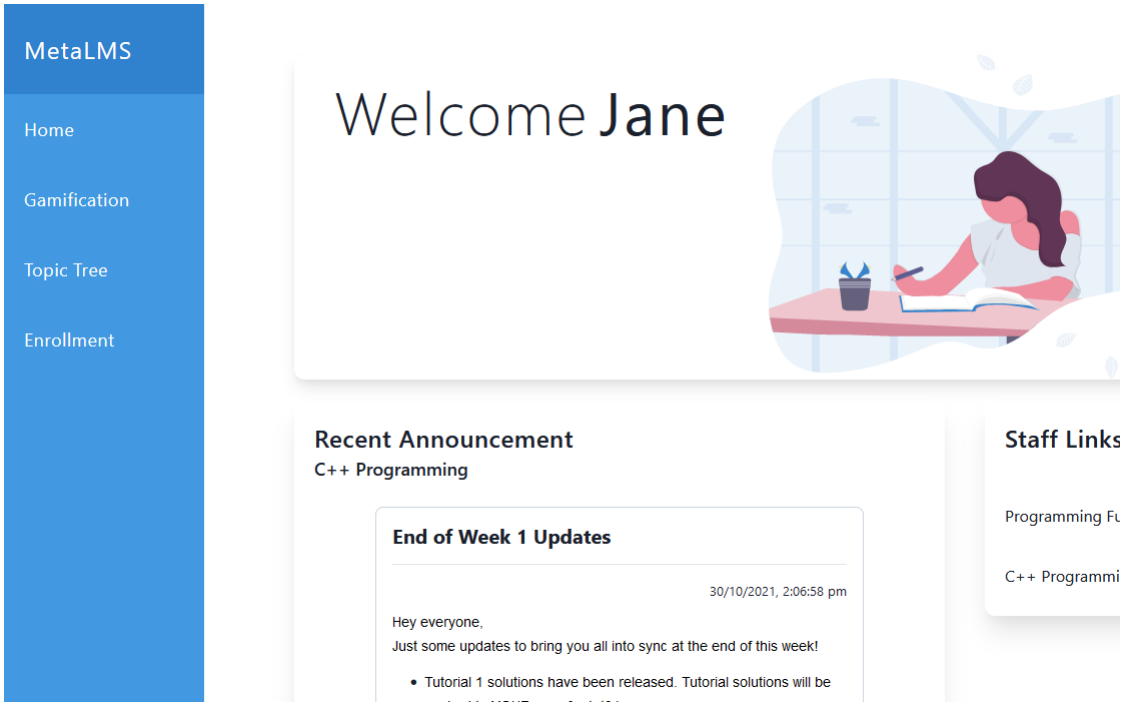

Figure 5.90: Accessing Gamification

Purpose

Serves a connection to the metalms and create a seamless move from metalms to gamification platform

What was Implemented

When a user clicks on the gamification link, the system will redirect the user to the gamification feature. Once this redirect occurs, the user's profile metadata is loaded up into the gamification feature as well for a seamless experience.

How it was Implemented

The metalms has a stored local variable for the authentication token. This token is passed through into the gamification feature which then authenticates the user with the backend database and at the same time retrieves metadata relating to the user. This profile includes user's full name, user's role (Student or Staff), Assigned Topic Groups, Levels Progression and Points Earned.

Considerations

- In the final design, I assumed that all users must access the gamification feature through the metalms and cannot seperately login into the gamification system. This was to simplify the routes required and create the integrated experience for users.
- A performance and security consideration was how to pass the user from metalms to gamification. Ultimately implemented it based on passing of authentication token which is used to query the backend when user arrived on gamification landing page. This query pulls the user's latest metadata. This has performance impact compared to simply passing the data along, however the security benefit of this outweighed the performance gains that would have been realised.

5.7.4 Student Dashboard

Design

Students will be redirected to the Dashboard page where they can see all the levels assigned to them.

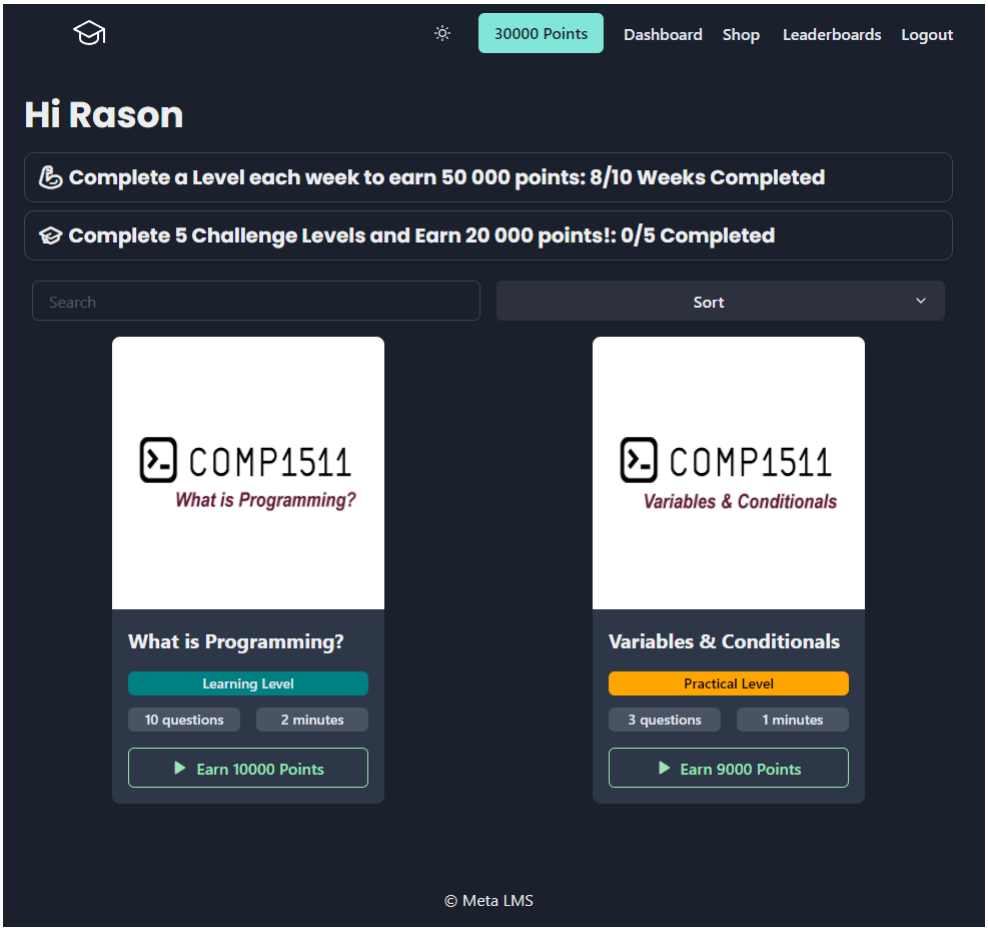

Figure 5.91: Student Dashboard

Purpose

Provide students an overview of levels available to be played. Provide staff to see the existing levels that they have edit permissions on as well as ability to create new levels.

What was Implemented

A grid list of level previews that display the levels available to user. The grid list is searchable to allow users to find specific levels. Level previews display level name, number of questions in level, time allowed and an option to start playing level.

How it was Implemented

The system will pull user's assigned topic groups and populate the levels. Metadata for each level is displayed onto a tile for each level including name of level, duration of level, number of questions and how many points available to be earned.

Considerations

- The tile presenting level information should be clear and concise, providing students only information they need.
- Decided to implement grid list search and sort functions although it took longer than expected as it allows students to navigate levels easily if there are multiple levels displayed.
- Decided to implement view of goals in the dashboard as a combined page as dashboard will ensure students see the goals first before interacting with other functionality.

5.7.5 Starting and Playing a Level

Design

Upon clicking the play button on the level tile on the dashboard page, students will see a get ready page below. Clicking Start Level will move the student to the first question.

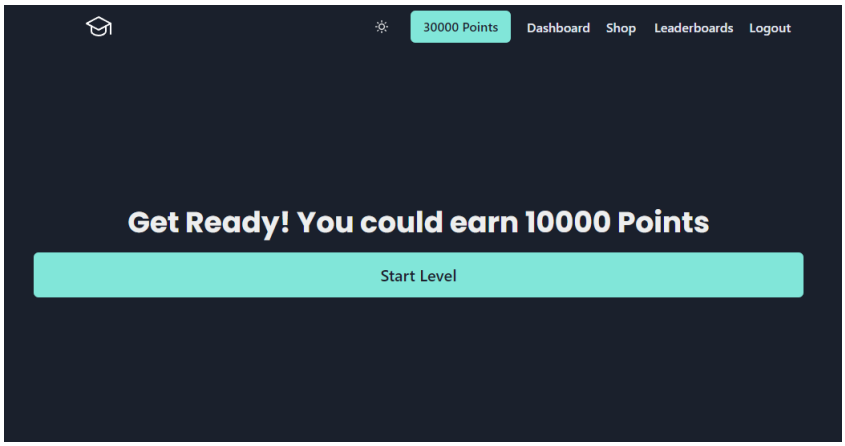

Figure 5.92: Starting Level

Students will be presented with the question and possible answers. They are tasked to select the most correct / appropriate answer. The timer starts the countdown for the time available for this question immediately. Upon reaching 0 time left, the answers will no longer be selectable.

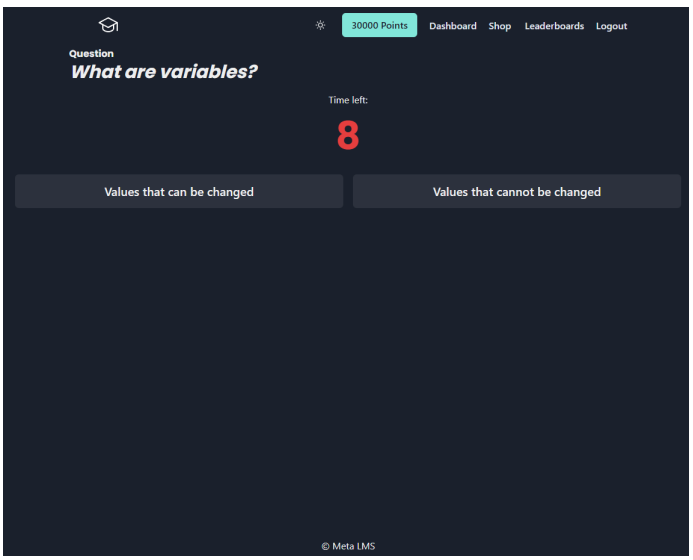

Figure 5.93: Playing Level

Upon the end of time available for the question, the correct answer is displayed in green and wrong answer displayed in red. The student is also provided with feedback on whether their answer was correct or wrong. If a student did not input an answer, by default their answer is wrong. They will be presented with a button to move onto the next question.

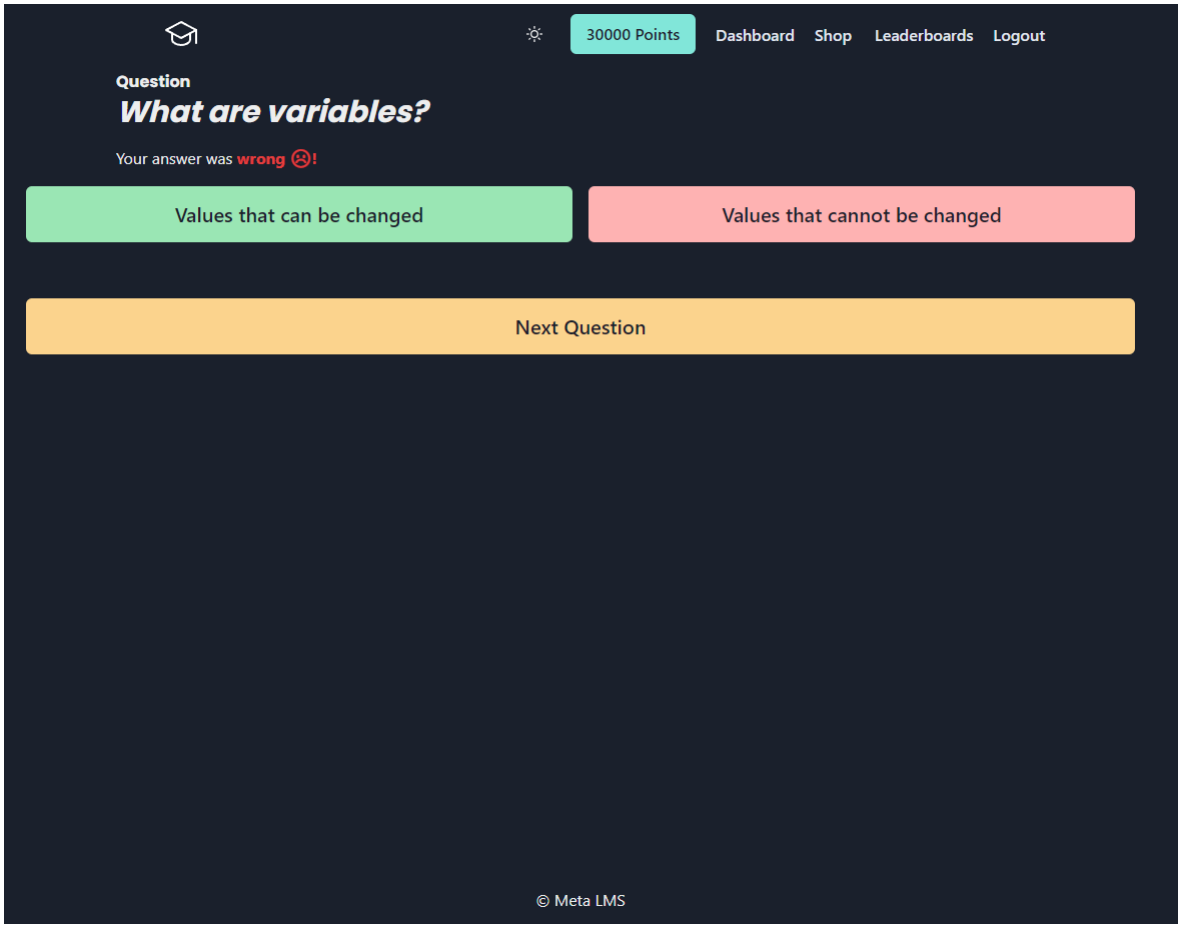

Figure 5.94: End Of Question

At the end of the level, students will be presented with an overview of how they performed in the level through correct / wrong outcomes for each question. Additionally, a feedback statement is provided based on the result. If the student has achieved more than 60% of answers correct, they will see a statement confirming their amount of points earned. If a student achieved between 41% - 60% of answers correct, they will see a statement confirming no points awarded.

Finally if a student achieved less than 40% or lower of answers correct, they will see a statement noting the amount of points penalty applied. The points changes are immediate and students are able to see the new points in the top navigation bar. Additionally, if a student had any boosters active from the shop, their points earned will be adjusted for the effect of the booster. After reading this page, students can return to the dashboard using the return home button.

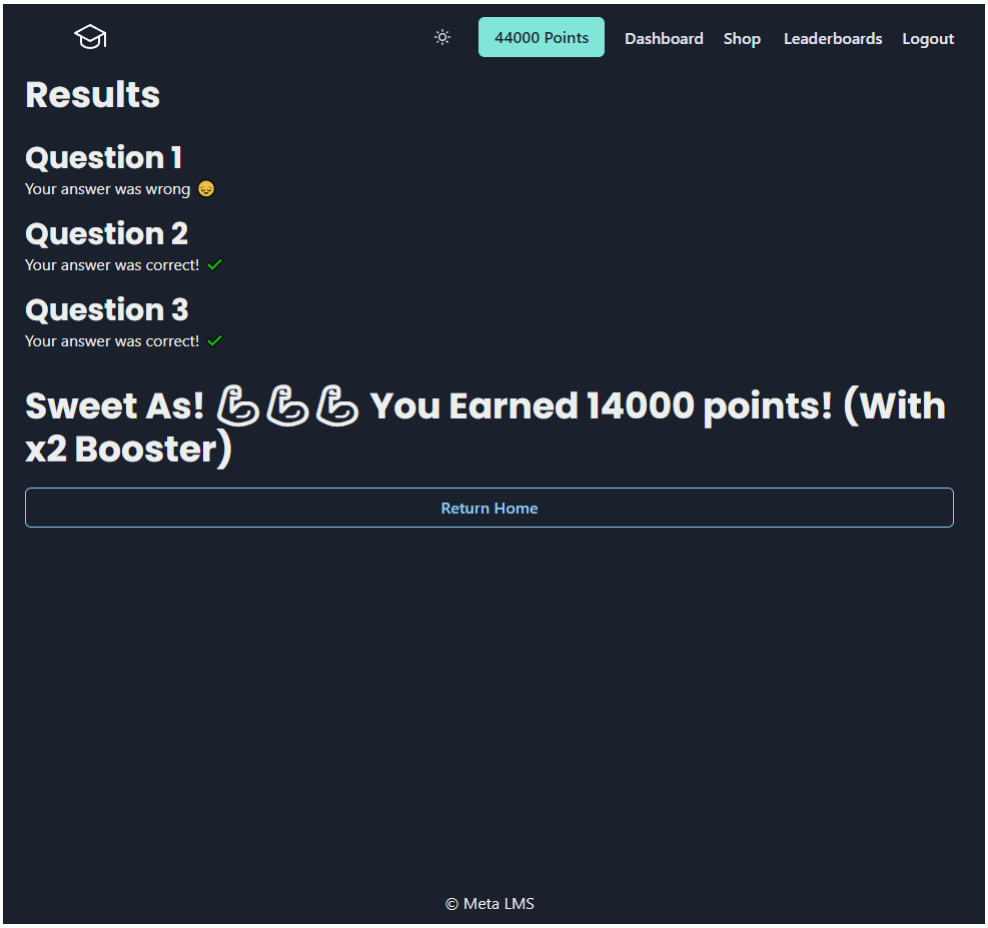

Figure 5.95: End Of Level

Purpose

Playing levels feature enables students to play through the level questions, display time limit per question, review correct answers and determine points earned at end of level.

What was Implemented

Students start the playing level feature by clicking play on the selected level. This will bring students to the loading page and subsequently the first question. A countdown timer starts immediately and the question & answers are displayed. Students can select / change answers until the timer reaches 0 where the UI changes to compare the student's answer to the correct answer and display if the answer was correct. Clicking the next button will allow students to move onto the next question.

After the last question, the feature calculates total points that the user has earned based on the number of questions students have answered correctly / incorrectly. If students answered less than 40% of questions correctly, they will receive negative points on the questions they answered wrongly. If a student answered between 41% - 60% of questions correctly, they do not earn or lose any points. If a student answers more than 60% of questions correctly, they will earn points based on the questions that they answered correctly.

How it was Implemented

Upon clicking the play button on a level, the platform uses the level id to retrieve the questions with respective time allocated, answers for each question and points available for each question. During this process, the user can observe a loading wheel followed by a screen that tells students the maximum amount of points they can earn for the specific level. Students will then be able to click start to begin the level.

Students will then see the first question. The platform will create a countdown timer using the time allocated for the question. During this time before countdown ends, if the user selects an answer, the platform updates selected answers state and sends it off to the backend to ensure that all changes in answers are captured immediately instead of at the end of the countdown. Once the timer ends, it updates the state and locks the selection of answers, it will then compare if selected answers are equal to the answers for the question. If incorrect, it will display "Your answer was wrong", if correct, it will display "Your answer was correct". In the background, the platform stores the selected answers back into the backend server and sets a completion timestamp for the question. This same pattern is reused until the last question has been answered.

At the results page, the platform retrieves the latest snapshot of a student's answers and calculates the points earned by the student. This is then validated to find out which category the result falls into (40%, 41%-60%, 61% - 100%) and determines a points result which could be negative, 0 or positive. In the background, the platform automatically updates student's points with the points result. This is how a user loses, earns or maintains their points when playing through a level.

Considerations

- Student's selected answers are recorded into the system immidiately when they select / change the answer. This deisgn inherently causes more requests to the backend api as students may change answers multiple times during the question compared to keeping the state locally and update backend at the end of the countdown. Ultimately decided with the frequent update due to increased reliability and ensure that changes just before the countdown ends are definitely captured correctly.
- The points penalty system took alot of deliberation to design and implement. If there was no penalty to simply guessing the answers for the level, students may not take the levels seriously and make random guesses. However, the system must also be fair to ensure that students who are trying but struggling are not negatively impacted but rather encouraged to do better. The final design was with the categories of less than or equal to 40% correct inurring penalty, between 41% and 60% answers correct will not earn or lose points and greater than 60% answers correct will earn points on the answers they got correct.

5.7.6 Buying and Using Items from Shop

Design

To access the shop where students can spend their points, they click on the Shop link in the top navigation bar. Once on the shop page, students can see the available items to be purchased using their points. These items are generally to help students earn more points within a set limited time period.

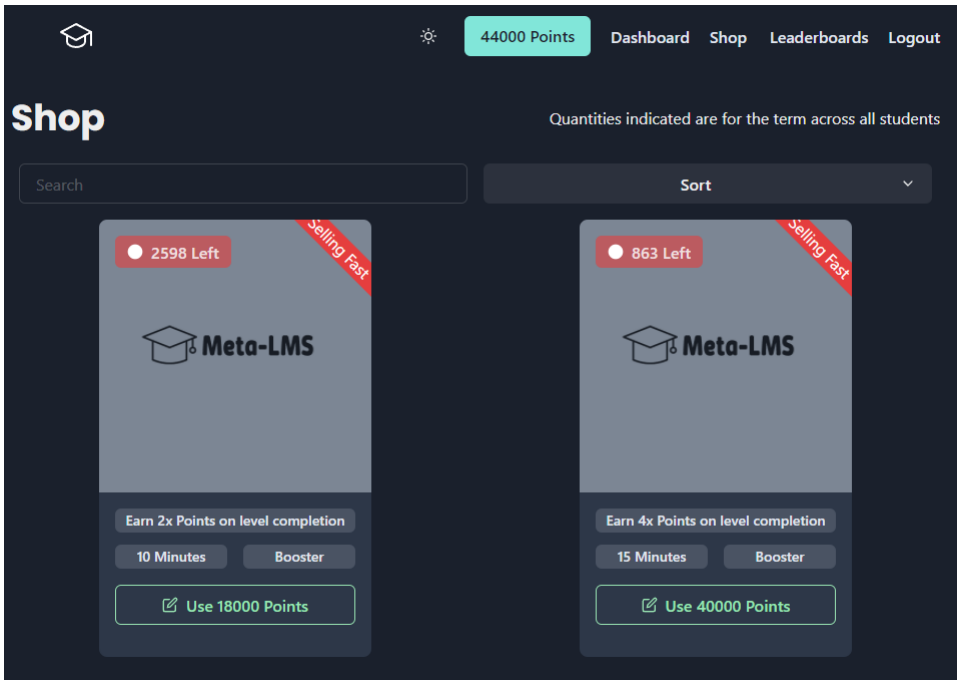

Figure 5.96: Browse Shop Page

Students can simply select the Use X Points on the specific item they are looking for. The details of the effect and duration of the item is shown on the same tile. Upon clicking Use X Points where X is the number of points it will cost, students will see a confirmation page which will confirm that the booster's effect will begin immediately. Upon clicking buy, the effect starts and the student will see a banner on the top of the platform to display that the booster is active.

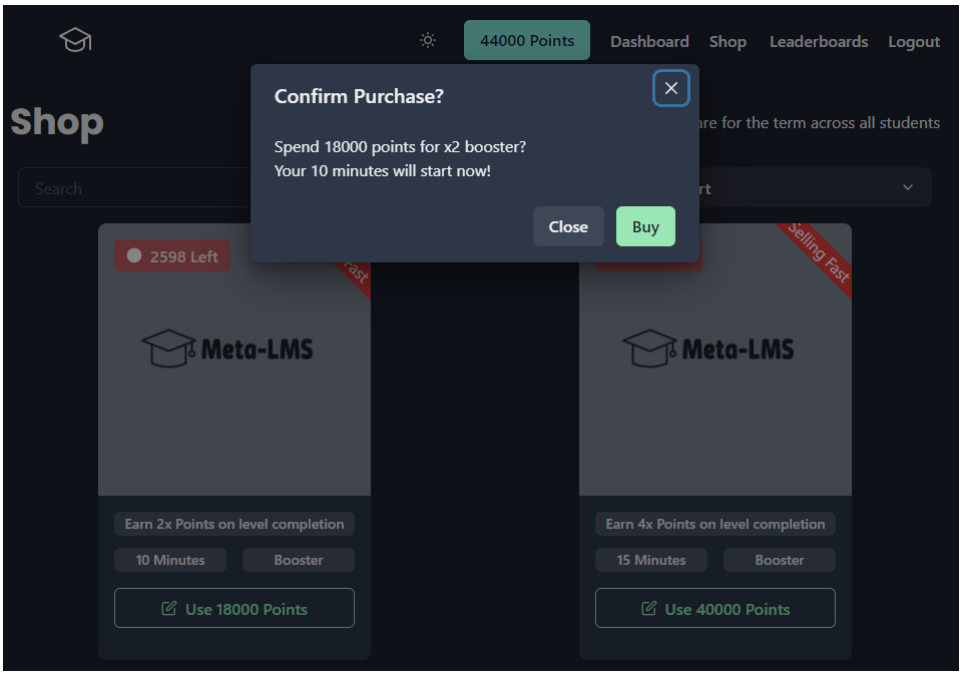

Figure 5.97: Buy an Item

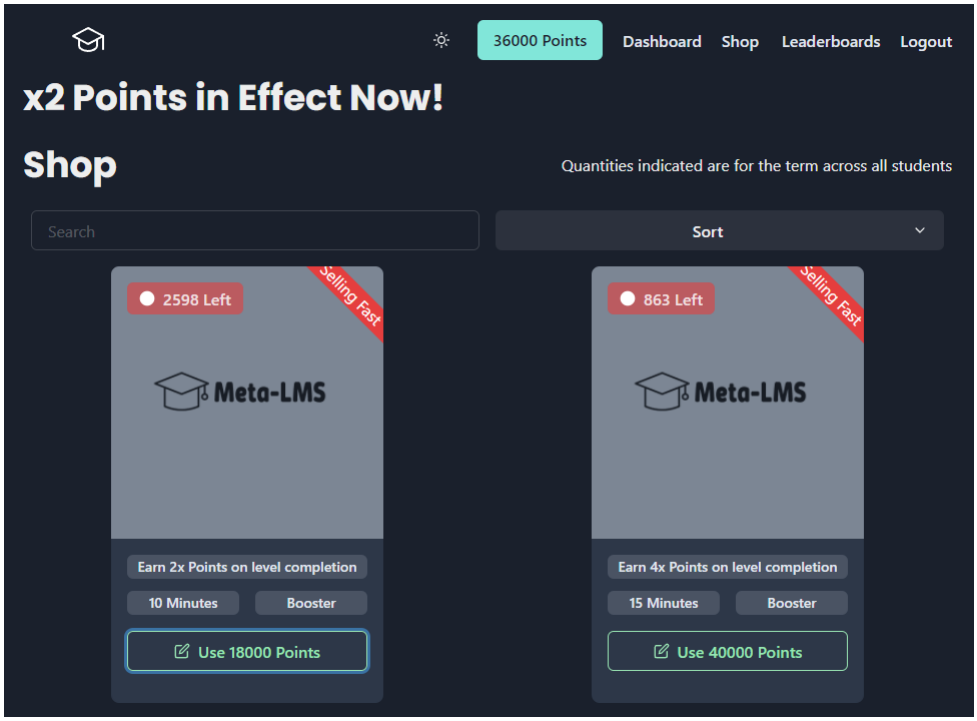

Figure 5.98: Shop Item In Effect

Purpose

The shop feature allows students to spend their points on boosters to increase the rate of earning points.

What was Implemented

There are 2 boosters available which are x2 points earned and x4 points earned with 10 minutes and 15 minutes duration respectively. These can be easily customised by platform admins and shop items are by default visible to all users on the gamification platform. Users can only buy 1 booster at a time and must use it immediately. Once a booster is used, there will be a banner on the top of the platform to notify users that the booster is in effect, this banner will remain until the effective duration is over. While the booster is in effect, the points that user earned will be multiplied by the multiplier of the booster used.

How it was Implemented

Within the backend server, we have a table setup with items. By default only 2 times with the attributes: item name, item type, item duration, item multiplier, item cost. These attributes will change how the booster behaves and the cost required to attain the booster. This information is retrieved when the user navigates to the shop page and the platform will display the items which are available for purchase in a grid list format with an option to buy. In the current implementation, the usage built is only on the item type is booster, it is possible to add other game item types to increase the variety of rewards for points.

Considerations

- Value of boosters vs the value of points needs to be balanced to drive correct behaviours and competitive environment

5.7.7 Leaderboards

Design

Students can access the leaderboards through the Leaderboards link in the navigation bar. The leaderboards page will display the user's progress through each of the topic groups that they are part of. It will display a mixture of prescriptive advice for the student to progress further in the platform and provide high level rankings details for the top 3 students in the topic group run to foster a competitive goal for students.

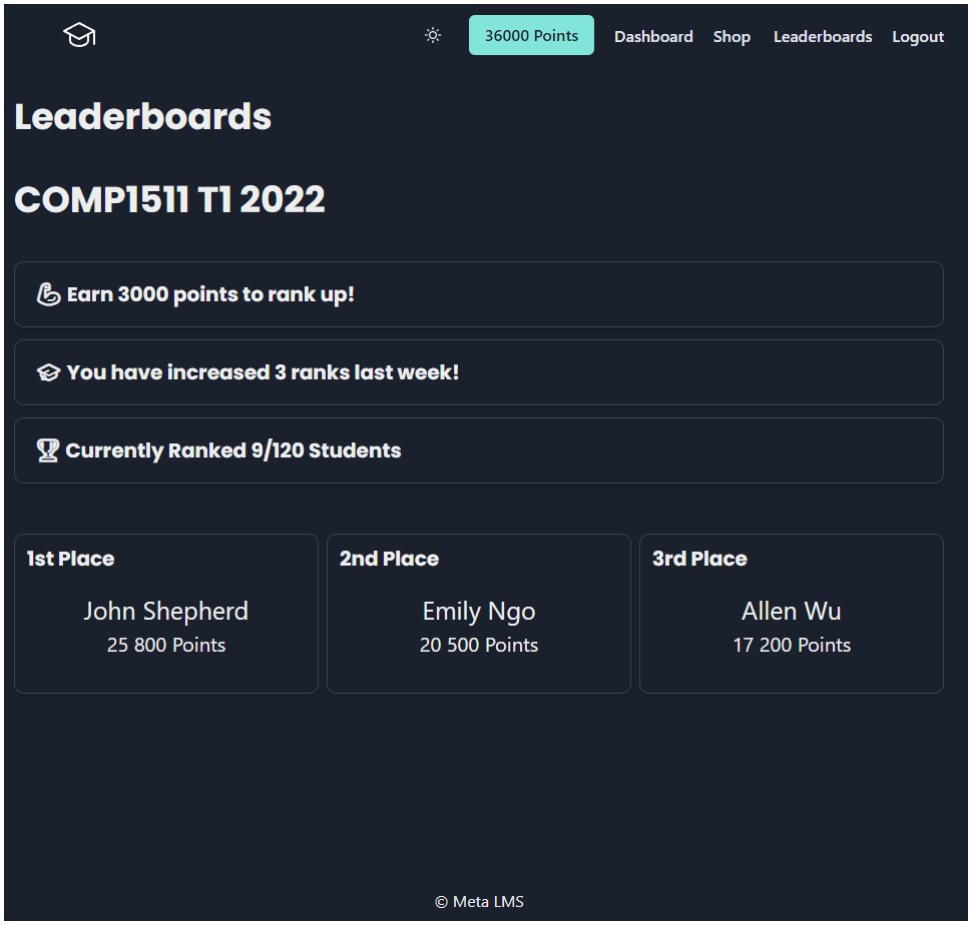

Figure 5.99: Student Leaderboards

Purpose

Provide students with prescriptive advice to enhance their learning while providing statistics on how they are progressing compared to their peers. This feature aims to create a competitive environment but avoid most of the negative connotations of ranking.

What was Implemented

A centralised organisation of statistics that displays to the student their rank in the topic group, how far they are to rank up and also the top 3 students in the topic group. This feature shows these details per topic group so it is more specific and targeted.

How it was Implemented

The platform first ranks the user in their topic group based on the amount of points earned on the platform. It then finds out the difference in points to the user in the next higher rank and displays the difference to the student. The system also pulls the details like name, points and ranking of the top 3 students. This is done for each of the topic groups that the student is enrolled in and is displayed one after another in a list format.

Considerations

- Avoid negative connotations by using a more objective approach to display facts that foster competition
- Provide prescriptive advice for students to improve their progress in the platform to break the goal of improvement into small achievable steps.

5.7.8 Creating and Editing Content in Levels

Design

This feature is only accessible by users with role of Staff. Upon entering the gamification platform, users with the assigned role of Admins will be directed to a Levels Dashboard which will display a grid list of all the levels they own / created. There is a button to add a New Level.

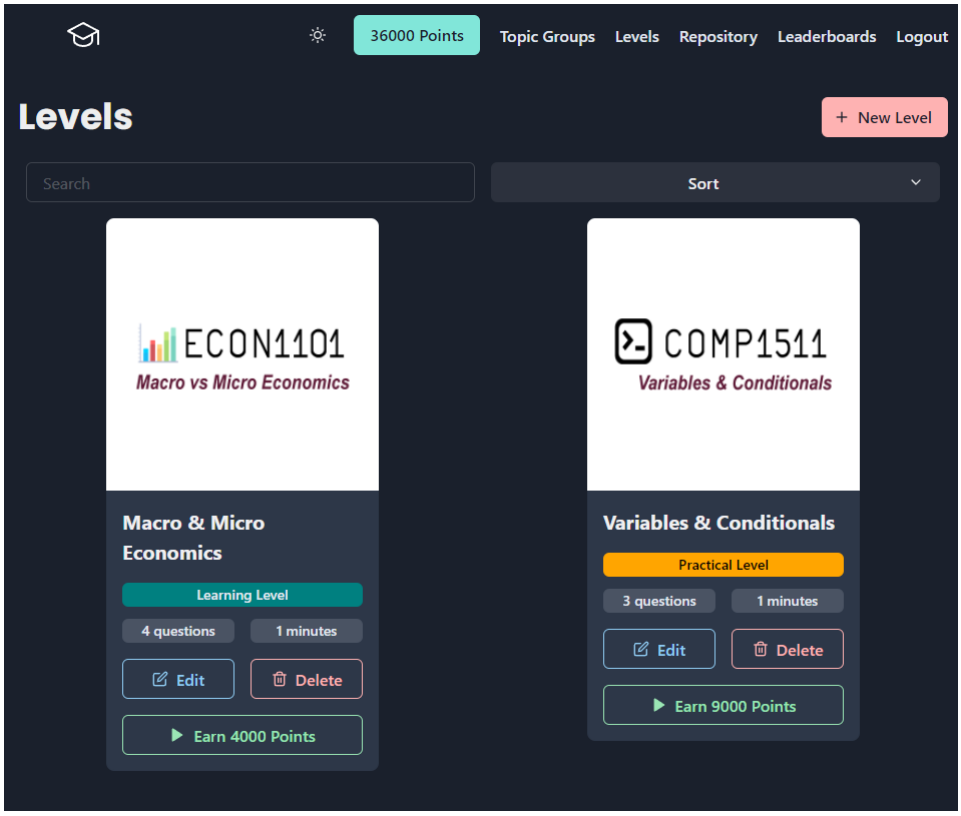

Figure 5.100: Admin Levels Dashboard

Upon clicking on the New Level button, admins are asked to provide a name for the level in a prompt. Once admin has entered a name for the level, they will be directed to Edit Level Page.

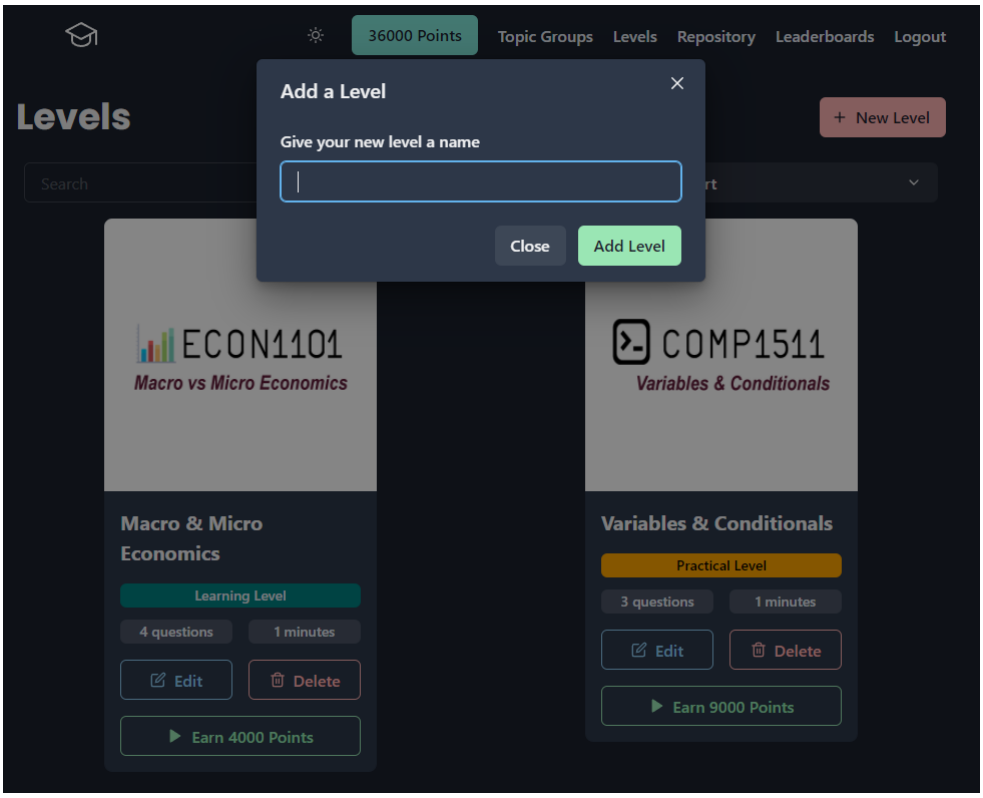

Figure 5.101: Create new Level

Admins are then directed to edit information about the level and its content. This page can be accessed on a new or existing level. Here the admin will add information like level name, week number (determines when in the term the level is visible to students), level type (Knowledge, Practice or Challenge), level format (Level or Live Kahoot Style Quiz) and a Thumbnail for the main image of the level. Below this is a functionality to edit questions that users will see within the level. Clicking Add a question, the user will see questions added with default 10s time available and 1000 points available. These questions can also be easily reordered by dragging the hamburger icon on the left of the tile to move the questions in order with the first question being at the top and last at the bottom.

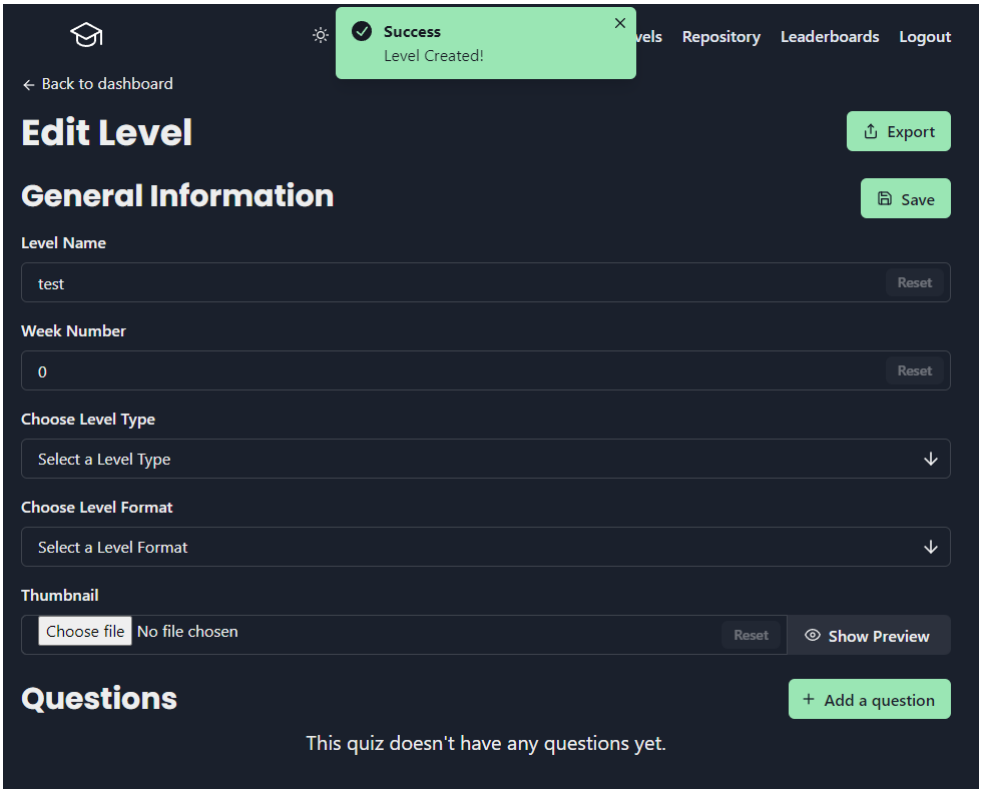

Figure 5.102: Edit Level

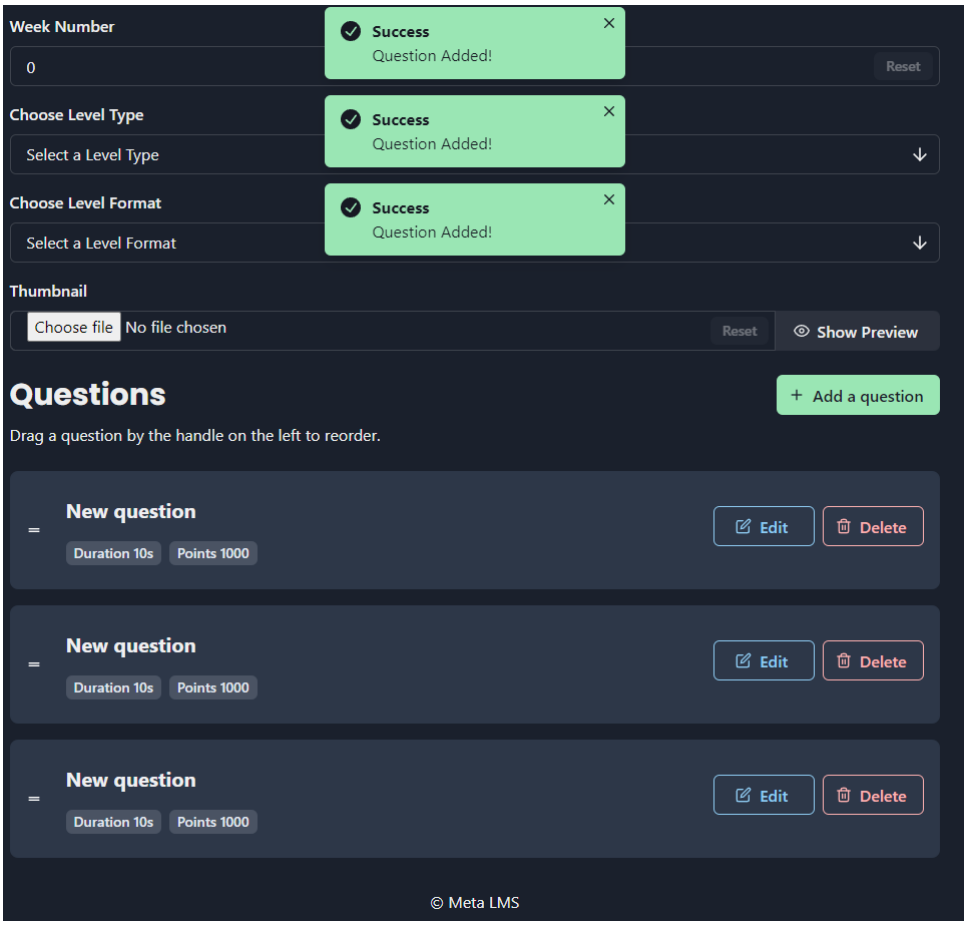

Figure 5.103: Add Questions

Upon clicking edit question, the user will see the edit question page. This page is specific to a single question and allows admin to customise the text for the question, the points available to be earned, the time available (duration) for the question, add any helper media like images or videos and determine the possible answers as well as selecting the correct answers with the green check boxes (can be multiple).

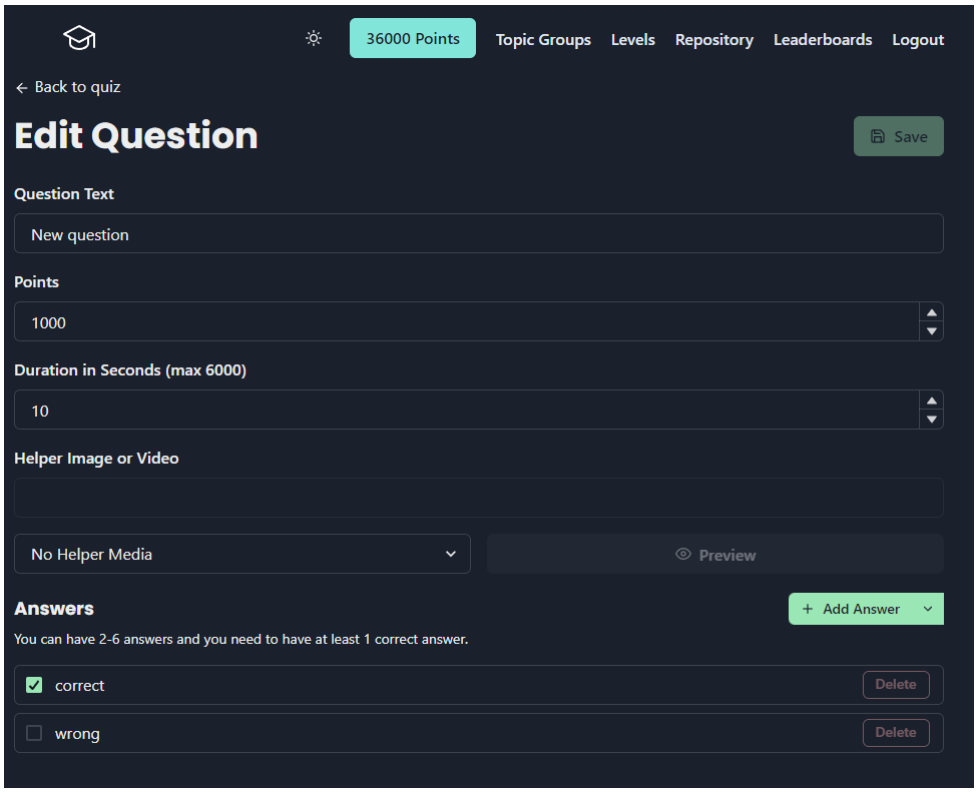

Figure 5.104: Edit Question

Purpose

Creating and Editing content for levels form the foundations to allow customisation to enable different content to be created on the platform for students to learn. This feature allows admins to easily create new content and edit past content.

What was Implemented

A capability to create new levels or edit levels that is only accessible to staff. Edit question allows admin to amend various atrributes of the level and apply it immidiately by saving. It also allows admin to determine the number and order of questions that will appear when students playthrough the level. The edit question page allows the admin to set the question text, add helper image or video, set the points that can be earned, set duration of the question and also set the answers for the question.

How it was Implemented

When a new level is requested by the admin, the backend creates a new record in the database for an empty level with the name specified. When opening the edit level page for a level, the backend retrieves all the attributes relating to the level and the array of question objects using the level id. The attributes are inserted into the UI inputs which can be amended by the admin. Once admin has finished their changes, clicking save on the top right will send the new data to the backend to update database records. The User Interface also loops through each question object in question array, displaying the question text, points available and time allowed. Admins have an option to Edit or Delete specific questions, clicking Edit will bring them to edit question page and delete will remove the question selected from the array, thus removing it from the level.

On the Edit Question page, similar to edit levels page, the platform pulls the attributes for the specific question using the question id. It populates the existing data into the UI fields and allows admins to amend its content. The second part of the page allows admins to change the possible answers that will appear (up to 6) and also select which answers are correct (could be multiple). Admins can also change the text for each answer. When an admin has made all their changes, clicking save on the top right will send this updated question state to the backend to be stored. The changes are immidiate and students will see the changes.

Considerations

- Organisation of Levels as parent to questions helped to simplify this flow. Initial design of questions as an attribute instead of part of array of objects caused problems with implementation.
- A design choice was made to not limit the number of questions for a level which could be an obstacle for some types of levels, instead the control is given to admins to manage the max number of questions as they see fit. The platform is capable to handle up to a few hundred questions before performance issues will appear on the edit pages.

5.7.9 Manage Topic Groups and Repository

Design

Admins can access the Topic Groups link in the top navigation bar. They will be presented with a grid list of topic groups they have admin rights for.

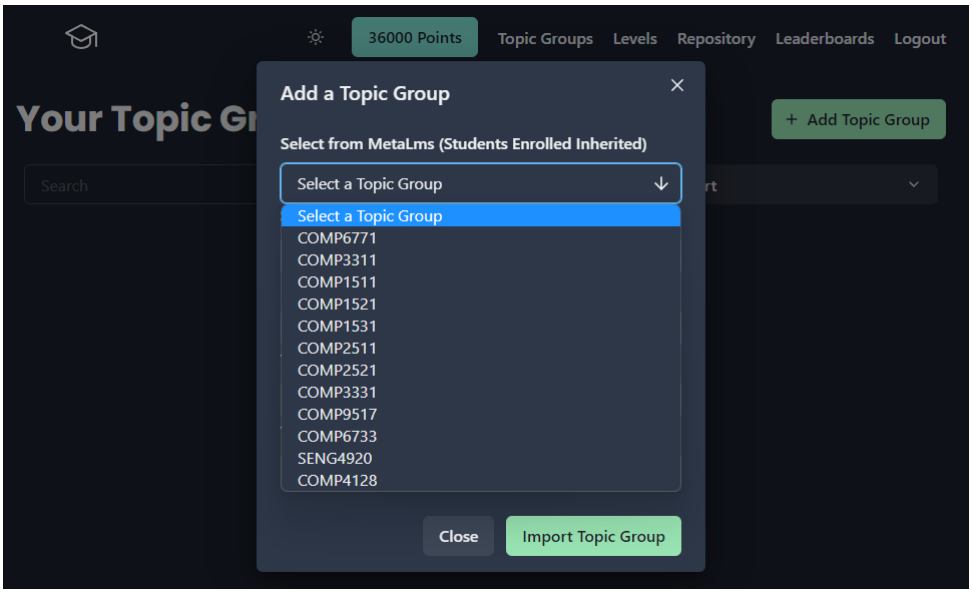

Figure 5.105: Manage Topic Groups

Upon clicking Add Topic Group, admins can see a list of available topic groups in the metalms system. They can select it to inherit the student permissions of the topic groups as meta-data like topic group descriptions into the gamification platform. Admins can then add additional information on the topic group like start / end date, term and year to make the topic group active. Once admin clicks import topic group, the students assigned in the wider metalms will have access to the levels within topic groups they are enrolled into.

Admins can add or remove levels they own / created by clicking the Edit Levels Button on the specific topic group tile. They will be presented with a list of levels available to be added or removed. The system detects if the level is already present in topic group and allow the admin to remove and otherwise it will allow the admin to add.

Admins can access the repository through the Repository Link in the top navigation bar. Once on the repository page, admins can see a list of all levels created by all admins on the platform. From there admins can search and find any levels (set of questions for a single topic) which they can add into their topic group.

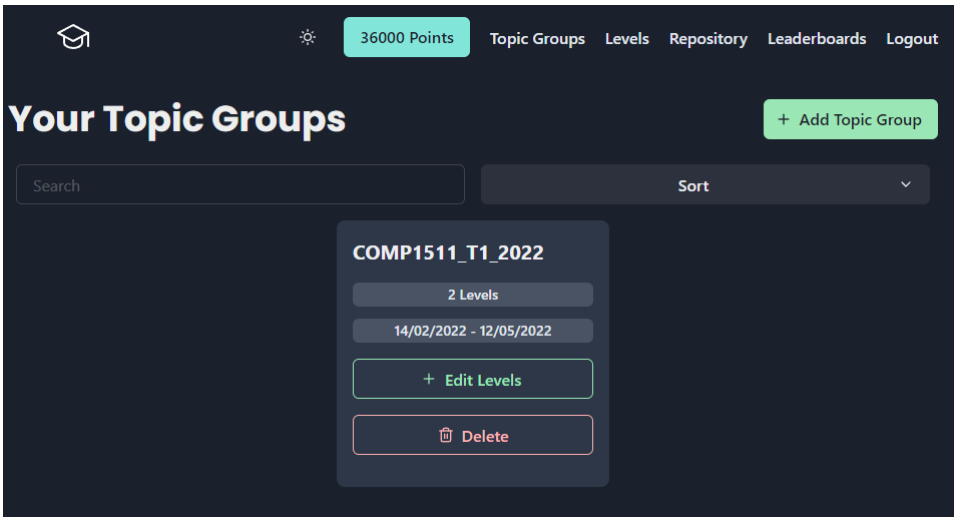

Figure 5.106: Importing a Topic Group

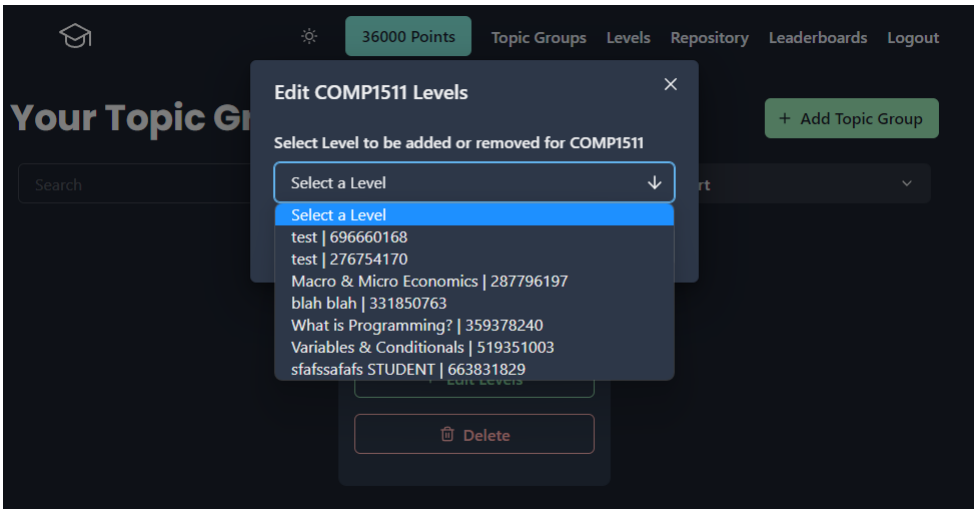

Figure 5.107: Add or Remove Levels from Topic Group

Purpose

Manage Topic Groups page allows admins to retrieve the topic groups from metalms that they need to actively manage. The topic group will help admins to make levels visible to students who have been enrolled into the topic group on the metalms. The repository acts as a centralised hub for all levels on the platform to be found, it allows any admin to search for levels which are set of questions for specific topics. This enables levels to be reused by admins.

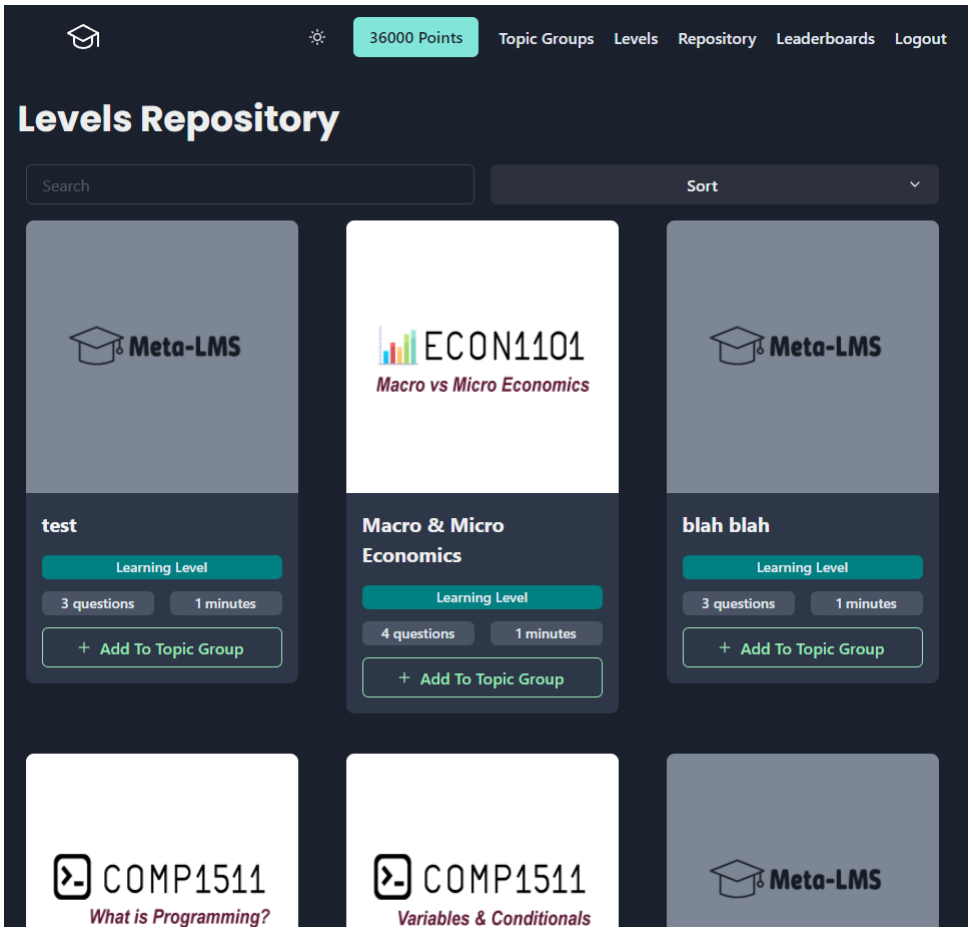

Figure 5.108: Repository

What was Implemented

Importing of metadata on topic groups, assigning levels to topic groups and removal of topic groups from gamification (does not affect metalms). Repository displays all levels created with a search and sort function to allow admins to easily find and reuse content.

How it was Implemented

When selecting add new topic group, the platform pulls the latest list of topic groups available on the metalms. This is presented to user as a list of topic groups. Once user selected a topic group, the system will retrieve all metadata relating to the topic group based on the topic group id and it will appear on the topic groups page for users as a new tile with some metadata displayed. From the tile, admins are able to edit or delete the topic group (does not affect metalms). Clicking on edit levels will retrieve all the levels that the admin owns or have created. After selecting a level on this prompt,

the platform uses the level id to identify if the level is already assigned to the topic group. If level has already been assigned, it will display a remove button instead of an add button. When user clicks on the add or remove button, the platform sends a request to add or remove level from topic groups using the level id and topic group id. Deleting topic group will send the topic group id to backend to remove topic group from gamification section of the database and this will not affect the topic group on the metalms.

Considerations

- Destructive changes like delete of topic group will not affect metalms. This was a concious decision as deletion of topic groups should occur in the enrolment section on the main Metalms page.
- Initial design was to have both topic groups and levels in the repository but it was complex to provide visibility over both topic groups and levels at the same time. Ultimately implemented with levels which is the smallest module to increase flexibility for admins.

5.8 Backend

The backend can be given a walk through using OpenAPI which is a specification for machine readable interfaces, producing and consuming restful API services.

5.8.1 OpenAPI

The OpenAPI documentation is accessible through the backend server. There are separations based on each component in the LMS.

Purpose

The purpose of the OpenAPI specification is to show a list of endpoints for each component including inputs and output responses.

What Was Implemented

A user with access to the OpenAPI document can view specific endpoints through their sections.

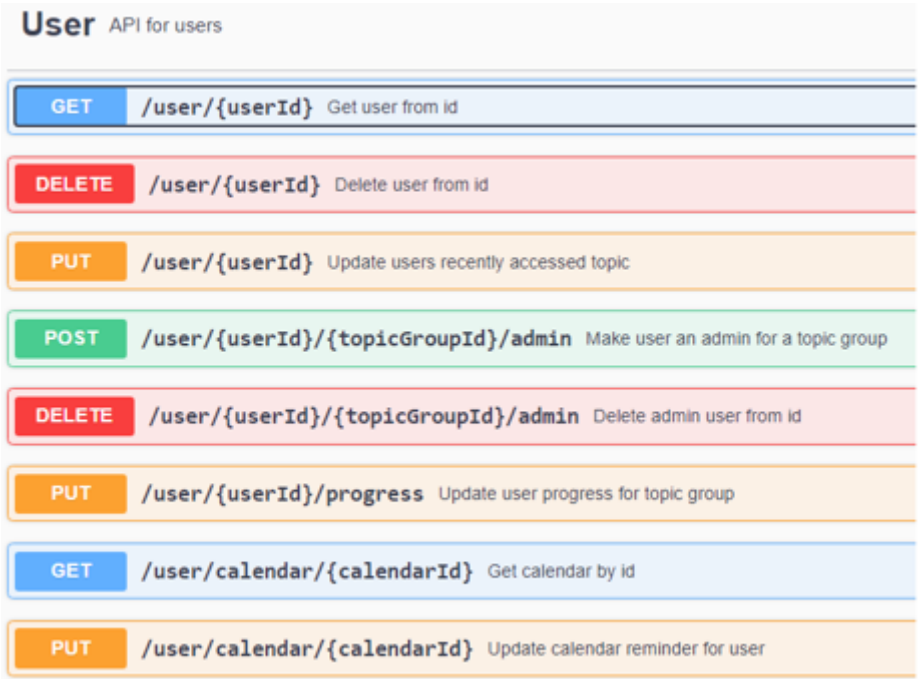

Figure 5.109: OpenAPI endpoints

The user can click on any endpoint and then view the required inputs and sample outputs of the endpoint. Then once the user has clicked on an endpoint, they are able to enter parameters to get a response tailored to their inputs.

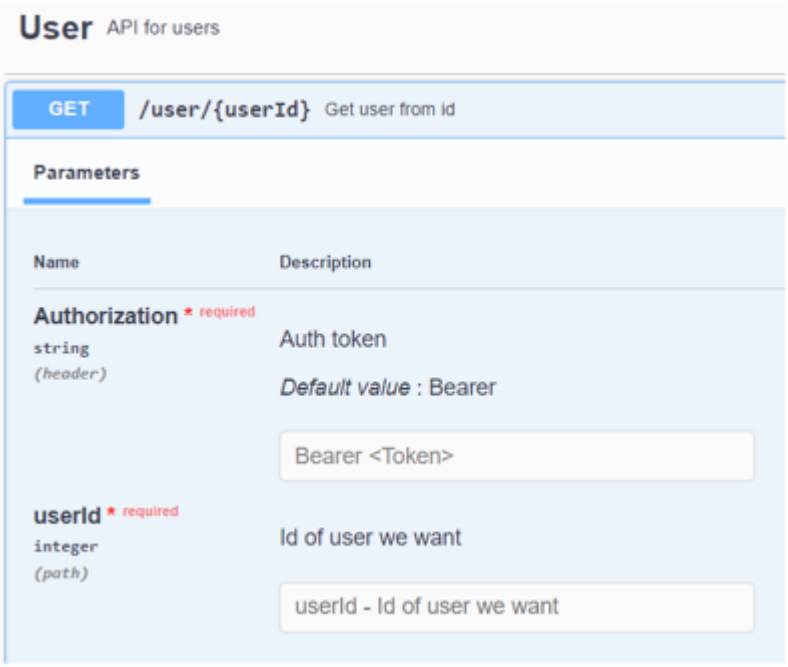

Figure 5.110: Endpoint Parameters

Figure 5.111: Output of endpoint

How It Was Implemented

This component was implemented using OpenAPI specification, which is a specification for visualising RESTful webservices. The APIs are created using NodeJS and Express and then visualised in the OpenAPI specification, and the backend handles the input parameters given through OpenAPI. Additionally, when the user clicks the execute button the request is sent to the backend and then the output response can be viewed in the relevant endpoint in OpenAPI.

Considerations

Instead of having a single list of endpoints, the endpoints were separated based on the project features. This allows users to quickly identify their desired endpoint and sections.

Chapter 6

Analysis

6.1 Accounts and Enrollments

6.1.1 Login and Sign Up

Functional and Non-Functional Requirements

The requirements laid out in the project plan for the login and sign up feature were as follows:

Users can register and login High

AS A	User
I WANT TO	Create a new account and log in to my existing account
SO THAT	I can use the LMS
Acceptance Criteria:	
- The user can successfully sign up to the LMS - The user can use an account previously made to log in	

Staff can create accounts on behalf of students Low

AS A	Staff Member
I WANT TO	Create accounts on behalf of students
SO THAT	I can streamline the formation of a course for new students
Acceptance Criteria:	
- The staff can create a batch of student accounts - Students can use these accounts to log in	

The first requirement was achieved very early in the project to all acceptance criteria, as it was high priority, while the second was not completed as part of the Thesis, however

will be listed to be completed in future work. It not completed due to the prioritisation of other advanced features such as the login code invite redirect causing this feature to be demoted in priority.

Usability Tests

There was one usability test conducted in relation to this feature. Three students and one teacher completed this and all the remainder of the tests for the accounts and enrollment feature.

Login and Sign up - Task 1	**Description**
Task	Register for an account and sign in
Starting State	The login page
Successful Completion Criteria	- Navigated to the sign up form from the login page - Correctly fills all fields - Submits the form once complete
Benchmark	30 seconds

All four of the users were able to very easily complete this test within the allotted time. This can be attributed to the simple design that is comparable to other login and sign up flows that exist elsewhere on the internet.

Performance and Accessibility

Google Lighthouse was used to benchmark the performance and accessibility of all of the features within accounts and enrollments. Google lighthouse uses automated tests to asses the response time of the page, and then analyses the code executed to build the page to find what may be causing a page to take a long time to respond and render. It also uses this same analysis to determine whether best accessibility practices have been used such as using appropriate alt-tags and aria labels where possible, and ensuring that components have proper contrast.

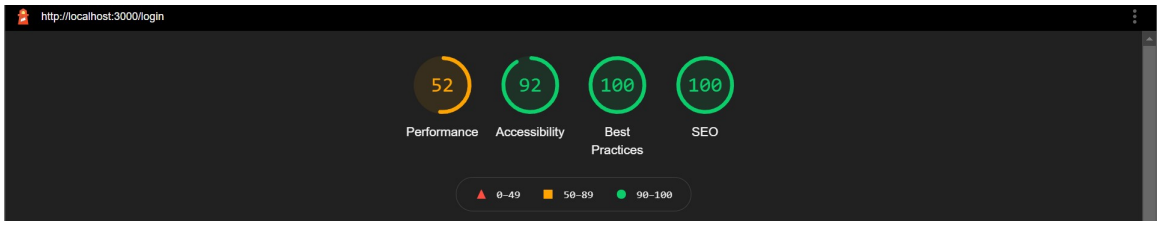

Figure 6.1: Google Lighthouse Results for the Login and Sign Up Page

The login page scored a 52/100 for performance and a 92/100 for accessibility. The performance score is primarily due to the non-minified state of the JavaScript. Minifying JavaScript is a process whereby the files that make up a web-page are processed to reduce file size by removing white-space and reducing all the code down to one line. As the Meta LMS is still in development, minifying the JavaScript is not feasible, as

it makes the code for the site non-human readable and thus non-editable. The other performance impact is unused JavaScript.

The accessibility score was not a perfect 100 due to a HTML container that is used to control the spacing on the page having an incorrect aria label. This can cause users with screen readers to have a hard time navigating the web page, however this can be easily remedied by refactoring the page's code.

Timeline

From the timeline set out in the project plan, the login and sign up feature was planned to be finished by week 4 of term 2. The front-end component of this feature was completed by this time, however the back-end component of this feature took an additional 6 weeks to finish. The reasons for which will be discussed in the following challenges section.

Challenges Faced

The extended timeline for completing the login and sign up feature was due to unforeseen complexities that resulted from developing authentication features in parallel with the rest of the LMS, meaning that the ever changing specification of the other features caused the nature of the authentication process to need to change a number of times through development. This could have been resolved by developing authentication before any of the other features, however this would have caused delays for the other portions of the LMS.

6.1.2 Account Management

Functional and Non-Functional Requirements

The requirements laid out in the project plan for the login and sign up feature were as follows:

Users can view and update their details Medium

AS A	User
I WANT TO	View and update my details
SO THAT	I can ensure my information is up to date
Acceptance Criteria:	
- Users can access a page that shows their account information - Users can edit their information from this page	

Staff can manage other users Low

AS A	Staff member
I WANT TO	Manage other users
SO THAT	I can ensure student's accounts contain the correct information
Acceptance Criteria:	
- Staff can access a page that shows a students account information - They can edit this information from this page	

The first requirement was successfully achieved to all acceptance criteria, however the second was lowered in priority in favor of advanced enrollment features and was subsequently not completed. However, the current accounts system was developed with these features in mind, and it is very feasible for the current system to be updated to support this functionality with future work.

Usability Tests

There was two usability tests conducted in relation to this feature. Three students and one teacher completed them.

Account Management - Task 1	**Description**
Task	Change your email
Starting State	The Site Dashboard
Successful Completion Criteria	- Navigated to the account management screen from the dashboard - Sets the account management page to edit mode - Fills in the new email and current password - Successfully submits the changes
Benchmark	90 seconds

Account Management - Task 2	**Description**
Task	Change your password
Starting State	The account management page
Successful Completion Criteria	- Sets the account management page to edit mode - Fills in the new password and confirm password fields - Enters current password - Successfully submits the changes
Benchmark	90 seconds

The purpose of test 1 was to determine whether users were easily able to find the account management page and then decipher the editing system. All 4 of the participants managed to successfully find the accounts page with no additional help, however one took considerably longer to find the drop-down menu on the dashboard. This can be attributed to the lack of an indication that clicking on the users name and image in the top right will display a drop-down menu, which can be rectified by adding a down

arrow icon next to the text, or an underline to the text to indicate its clickable. Once on the account page, all users very quickly found the edit button and filled the email field. 3 of the 4 participants tried submitting the form without confirming their password which resulted in an error. All of the participants successfully completed the task, but 2 of the 4 did not finish within the benchmark time, due to not finding the drop-down and missing the confirm password portion of the form. The confirm password issue could be resolved with more visibility to the required nature of that field for the form, perhaps by adding a subheading above that input, as the current implementation of the red asterisks next to the text fields label was easily overlooked by participants.

The second test was completed much more successfully by all participants. After having already used the edit feature, all participants correctly filled the form to change both fields, remembering to confirm their own password, and intrinsically understanding to confirm their new password before submitting. This speaks to the learn-ability of the account management feature.

Performance and Accessibility

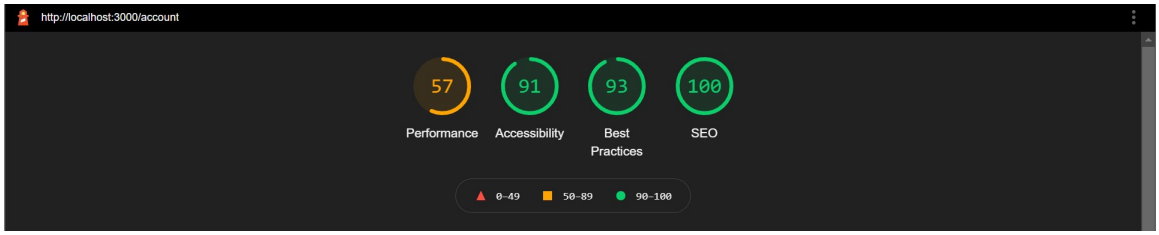

Figure 6.2: Google Lighthouse Results for the Account Management Page

The accounts page received near identical scores to the login and sign up page, with the same criticisms for each of the criteria. The accessibility score was a 91/100 due to some divs in the page that were used for styling not having the correct aria labels, and the performance was mostly impacted by non-minified JavaScript. Both of these can be easily rectified.

Timeline

The timeline originally defined when planning was that users would be able to view their account information by Week 6 Term 2, and edit it by Week 4 Term 3. However, both of these features were delayed to being worked on from between Week 2 Term 3 and Week 5 Term 3. This was done to ensure that the enrollment features that were more essential to the functionality of the LMS as a whole being were completed first.

Challenges Faced

One of the primary challenges faced while developing this feature was the ability to add profile images. We did not want to store these images within the database, as that would increase the size of any payloads containing these images to be extremely

large, causing the entire site to slow down, but we did not have the time or resources to develop a separate online storage for these images within the project. Thus, as a temporary measure, users are able to enter a link to an image to be used as their profile image. This is not a good solution from a user experience perspective, but it is functional for the time being. However, due to the nature of the current system pointing to a URL that stores an image, future work can be done to quite easily add an online store for images once more resources to fund online storage are available.

6.1.3 Enrollment Dashboard - Staff

Functional and Non-Functional Requirements

The requirements laid out in the project plan for the staff enrollment dashboard feature were as follows:

Staff can enrol and un-enroll students on their behalf High

AS A	Staff member
I WANT TO	Enrol and un-enroll students from a course
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that shows all students enrolled in a course - They can enroll new students or un-enroll existing students from the course	

Staff can generate course invite codes High

AS A	Staff member
I WANT TO	Generate course invite codes
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that allows them to generate and manage course invite codes - They can add restrictions to code like number of uses and time valid - They can manage existing codes - When a student uses the code they can enroll in a course	

Staff can set courses to be open enrolment or invite/code only High

AS A	Staff member
I WANT TO	Set courses to be open enrolment or invite/code only
SO THAT	I can effectively manage a course's enrollments
Acceptance Criteria:	
- Staff can access a page that allows them to manage whether a course is publically search-able - They can control whether a course is public	

All 3 of these requirements were successfully completed to all acceptance criteria. They were all made high priority once development commenced due to the importance of them to the functioning of the broader LMS.

Usability Tests

There were three usability tests conducted in relation to this feature. One teacher completed them.

Staff Enrollment Dashboard - Task 1	**Description**
Task	Delete an existing invite code and generate a new one with limited uses, then copy it to the clipboard
Starting State	A course's dashboard
Successful Completion Criteria	- Navigates to the enrollment dashboard - Identifies the code to delete and deletes it - Sets the number of uses for the new code - Generates the new code and saves it to the clipboard
Benchmark	60 seconds

Staff Enrollment Dashboard - Task 2	**Description**
Task	Set the state of the course to search-able
Starting State	A course's enrollment dashboard
Successful Completion Criteria	- Identifies the search-ability toggle - Set the course to search-able
Benchmark	10 seconds

Staff Enrollment Dashboard - Task 3	**Description**
Task	Manually enroll a student, un-enroll that student
Starting State	A course's enrollment dashboard
Successful Completion Criteria	- Identifies the manual enrollment box and enters a students ID - Identifies that same student in the student list - Successfully un-enrolls that student
Benchmark	60 seconds

Each of the tests were designed to test one of the three key enrollment features. Task 1 focuses on the invite code feature. The participant was easily able to navigate to the enrollment dashboard, and found the code to delete very quickly. They also very quickly identified the delete button and successfully deleted the code. They took a little while to analyse the code generation form and read the sub-headings, but generated the new code and copied it to clipboard with no difficulty. An interesting note was that

they used the copy button from within the table list rather than the one at the top of the page next to the code. When asked why, they responded by saying that they did not realise the top one was there. This can be addressed by increasing the size of that button and perhaps making it a more visible color.

Task 2 was completed very quickly. After a cursory screening of the page, the participant found and toggled the search-ability switch well within the benchmark time. Task 3 was also completed with ease, the participant very quickly found the text input, enrolled the student with the given ID, and were able to easily find the student with the corresponding student ID that was just enrolled, and used the un-enroll button to un-enroll them well within the benchmark time. A comment the participant made was that when they have a larger course size, it may be more difficult to find a student in the student list, and suggested perhaps a search bar or more advanced filtering for the student list.

Performance and Accessibility

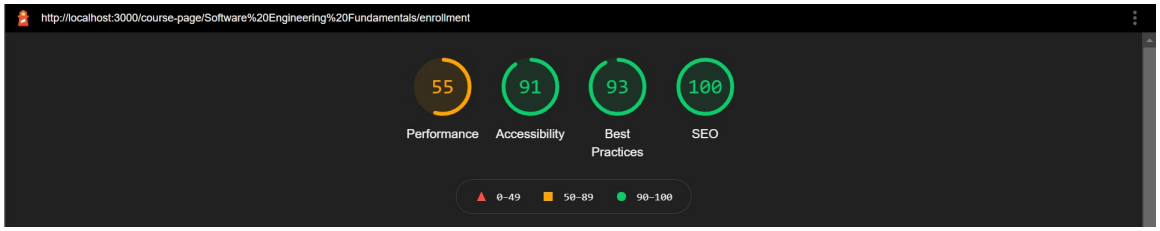

Figure 6.3: Google Lighthouse Results for the Staff Enrollment Dashboard

The staff enrollment dashboard received near identical scores to the login and sign up page, with the same criticisms for each of the criteria. The accessibility score was a 91/100 due to some divs in the page that were used for styling not having the correct aria labels, and the performance was mostly impacted by non-minified JavaScript. Both of these can be easily rectified.

Timeline

In the original timeline, this was broken out into standard and advanced enrollment features, with the standard features being manual enrollment and advanced being everything else. Standard features were planned on being completed in Week 9 of Term 2, and was completed on schedule. Advanced features were planned on being finished in Week 4 of Term 3, by which the code enrollment was completed, but the search functionality wasn't completed until week 7.

Challenges Faced

This feature contained the bulk of the feature work for the accounts and enrollment feature, due to the fact that a lot of the back-end architecture for the student enrollment features had to be considered and developed as a part of the staff enrollment features. As a result, many decisions had to be made regarding the structure of the database, and

how student and staff accounts would interact with entries for courses and topics. The eventual decision was to create a new table that contained the course id, the user id, the date enrolled and the users progress through the course to represent an enrollment. This format gives us the flexibility to be able to search enrollments per user, and per course, as well as gain some basic information about these enrollments very easily.

Another challenge faced was the architecture for the invite code feature. It became difficult to ensure that none of the codes in the database were expired, leading to potential misleading information on the enrollment dashboard and potential erroneous enrollments. To combat this, every time an enrollment code is accessed, its validity is checked, and also, while the staff enrollment dashboard is open, the invite code table is reloaded every 5 minutes to force the back-end to check the validity of all the enroll codes and to give the front-end the most up to date information about the number of uses and time remaining for each of these codes.

6.1.4 Enrollment Dashboard - Student

Functional and Non-Functional Requirements

The requirements laid out in the project plan for the student enrollment dashboard feature were as follows:

Students can enrol themselves in courses with a code High

AS A	Student
I WANT TO	Enrol myself in courses with a code
SO THAT	I can enroll in a new course
Acceptance Criteria:	
- Students have access to an enrollment page where they can enroll in a course with an invite code - After submitting the code, they are enrolled in the corresponding course	

Students can search for open courses High

AS A	Student
I WANT TO	Search for open courses
SO THAT	I can enroll in a new course
Acceptance Criteria:	
- Students have access to an enrollment page where they can search for courses - After searching, a list of courses will be shown to the student - Students can interact with this list to enroll in a course	

Both of these requirements were successfully completed to all acceptance criteria. They were all made high priority once development commenced due to the importance of them to the functioning of the broader LMS.

Usability Tests

There were three usability tests conducted in relation to this feature. Three students completed them.

Student Enrollment Dashboard - Task 1	**Description**
Task	Enroll with an invite code
Starting State	Site dashboard
Successful Completion Criteria	- Correctly navigates to the student enrollment dashboard - Enters the invite code and successfully enrolls
Benchmark	60 seconds

Student Enrollment Dashboard - Task 2	**Description**
Task	Search for and enroll in a course
Starting State	Student enrollment dashboard
Successful Completion Criteria	- Enters a search term and searches successfully - Uses search results to enroll in a course
Benchmark	30 seconds

Student Enrollment Dashboard - Task 2	**Description**
Task	Creates a new account and enrolls in a course from an invite code
Starting State	Browser new tab page
Successful Completion Criteria	- Uses the invite link to navigate to the site - Successfully creates a new account - Enrolls in the course
Benchmark	90 seconds

The first test was designed to determine the ease of finding the student enrollment page, and the ease of understanding of the invite code enrollment format. All 3 participants very quickly found the enrollment page, figured out where to enter the invite code, and enrolled the course well within the benchmark time. The second test, designed to perform the same analysis of the search, was also very successfully completed without any issues by all 3 participants. The third task was designed to test the user flow of a new user to the site. 2 of the 3 participants navigated to the site directly from the invite link, signed up and enrolled very easily, while the third instead first navigated to the site themselves, made a new account and once they had made a new account and logged in, went to the invite link and enrolled. When asked why they followed this path through the app instead of directly going to the invite link, they stated that they did not think it would work because they needed to make a new account. The time taken between the 2 approaches the participants took was not hugely different, and as such

this test exposed the flexibility of the invite code system to the workflows of different users.

Performance and Accessibility

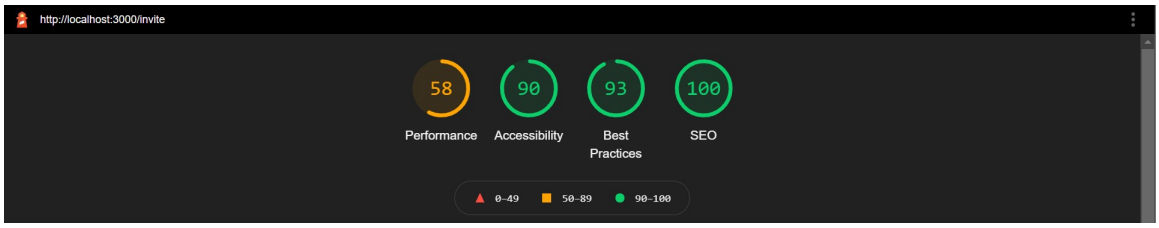

Figure 6.4: Google Lighthouse Results for the Student Enrollment Dashboard

The student enrollment dashboard received near identical scores to the login and sign up page, with the same criticisms for each of the criteria. The accessibility score was a 90/100 due to some divs in the page that were used for styling not having the correct aria labels, and the performance was mostly impacted by non-minified JavaScript. Both of these can be easily rectified.

Timeline

Due to the interconnected nature of the staff and student enrollment features, the student features were completed on the same timeline ass the staff features. In the original timeline, this was broken out into standard and advanced enrollment features, with the standard features being manual enrollment and advanced being everything else. Standard features were planned on being completed in Week 9 of Term 2, and was completed on schedule. Advanced features were planned on being finished in Week 4 of Term 3, by which the code enrollment was completed, but the search functionality wasn't completed until week 7.

Challenges Faced

The primary challenge faced was creating an effective user-flow for both existing and new users to enroll using an invite code. It was found out very early during internal testing that although the invite codes worked very well for existing users, the process for a new user was less than ideal, as they would have to follow an invite link, then make an account, then re-enter the invite link in their browser. This process did not feel smooth and lead to a poor user experience. The solution was to implement a redirect system that allowed the login page to redirect to the student enrollment dashboard, if the link that took them to the login page was an invite link. This change greatly improved the new user experience.

Another challenge was the quality of search results returned to student users. Initially, the search only compared itself to course names and course codes, however it was found that this method did not help students who did not know the exact name of the course they were looking for, as it did not allow for more general searching. To fix this, the

search was expanded to also search course outlines. This allows the search to be more powerful and give better results, however the cost is that the search takes a lot longer to complete. Although the time for search is a downside, it was decided that it was a worthwhile trade-off due to how much it improved the quality of the search results.

6.2 Topic Tree

The topic tree feature will be examined based on a framework to determine how well the implemented feature has achieved its intended purpose and its functionality.

Therefore the analysis includes four sections:

- Functional Requirements - how well does the system fulfil the functional requirements set out in the project approach;
- Non-functional Requirements - how well does the system achieve the non functional requirements including performance and accessibility;
- Scalability and Efficiency - how well does the feature perform with many, many topics and topic groups;
- Feedback from potential users of the feature - what do users such as lecturers, tutors and students think about the feature and how could it be improved?

6.2.1 Functional requirements

The functional requirements of the feature were analysed for completedness. Each functional requirement has a priority attached to it, and the functionality of the system was assessed against this priority.

In total, 16 requirements out of 19 were fully completed. 1 high requirement, 1 medium requirement and 1 low requirement were not completed at all. 10 high requirements, 5 medium requirements and 1 low requirement were completed.

The following explains the requirements that were not completed and why they were not completed.

Users can search for specific resources - High Priority
Currently the topic tree feature allows users to search through the topic tree for specific topics, but does not allow users to search through the entire topic tree for a specific

resource. This is because topic resources are already grouped by topic, and there was no need to have to search the entire topic tree for a specific resource. Also, tags were added as a feature where users can specify different synonyms for a topic, improving searchability. Topic resources is also commonly given generic names, such as "Lecture Slides". This is not descriptive of the content, and would have proved resource searchability difficult.

Users can delete a topic group that they've created - Medium Priority
Users can currently create topic groups, but cannot delete them. This is due to integration issues found with deleting topics, as topic groups are an integral part of the entire system. Deleting topic groups could create database integrity issues, and there was not enough time to fix these issues.

Users can export data from topics and course material - Low Priority
This requirement was classed low priority as the system was designed to be the only learning management system for an entire faculty or university. Exporting data from topics and course material would have proved useful for exporting into another learning management system, but there was again not enough time to implement this feature either.

6.2.2 Non functional requirements

The following details the analysis undertaken to examine the accessibility, security and performance of the feature.

6.2.3 Performance

Performance of the system could be improved, with the main topic tree graph view loading a little slowly. Google Lighthouse was used to analyse performance, and performance differs depending on the database server used i.e. if a remote server was used, the topic tree takes longer to load the graph.

Google Lighthouse is a tool used to measure page performance and accessibility, and gives a score out of 100 after running its tests. It waits until there is no more movement on the graph and measures the time taken, however the page is interactive before movement stops on the graph due to nodes slowly moving outwards from where they loaded.

As shown in the report, a score of 32/100 was given for the topic tree's performance. Restructuring of the database and less API requests are strategies that can be used to improve the performance of the topic tree, as well as a different graphing library as d3

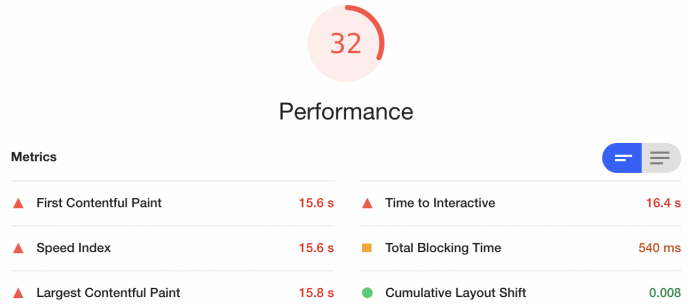

Figure 6.5: Performance report of the topic tree

is not an optimised library for handling graphs.

6.2.4 Accessibility

Accessibility of the feature is good, as the Chakra UI library was used to ensure that alt and aria tags were used. Colour contrast of the feature could be improved, but is still quite good with contrasting blue and white colours used.

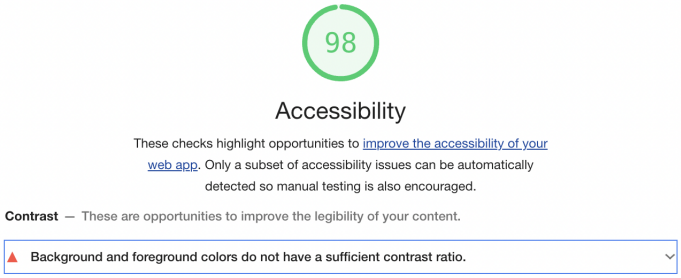

Figure 6.6: Accessibility report of the topic tree

A score of 98 was achieved for accessibility. Google Lighthouse accesses tags and HTML elements used, as well as contrast ratios to assess accessibility.

6.2.5 Scalability Testing

This section focuses on how well the feature copes against tens and hundreds of topic groups and a large amount of data. As the feature will be used with many, many topic groups, scalability testing is performed to ensure the feature is usable.

To assess the feature, each topic group will contain 3 topics to simulate a real world topic group. The feature will be tested with 3, 5, 10, 20, 50, 100, and 300 topic groups. The School of CSE currently runs around 100 courses, and so as this topic tree is designed to be used by a faculty or school, 300 topic groups is the limit that will be tested. Ideally, disciplines will be implemented in the future which will create separate topic trees for each school or faculty.

Each test will be assessed by measuring the performance using Google Lighthouse i.e. the time to finish loading the website.
The user experience will also be assessed, with comments made after each test referring to whether managing the number of topics is efficient or not.

Results

Number of Topic Groups	Time Taken to Interactive (sec)
3	3.13
5	3.16
10	3.20
20	3.43
50	6.05
100	10.11
300	26.47

As shown in the above results, some issues arise with more than 50 topic groups. The performance is quite good but can be significantly improved, especially when there are more than 50 topic groups. According to a BBC article on Nordstorm, a popular shopping website, they believe a page load time of 2.5 seconds or less strikes a good balance between functionality and speed. By minimising the amount of data and database requests needed to load the feature, the speed of the topic tree can be dramatically improved.

Up to 20 topic groups are also shown on the graph very well, as shown in the screenshot below.

However, once 50 or more topic groups are shown, the topic groups start to cluster together. This makes it quite difficult for users to navigate around the topic tree, as shown below as well.

The issue is then accelerated with more and more topic groups displayed on the topic tree as well.

This can be improved by altering the force algorithm used to separate the nodes, which was designed by the d3 module that this feature uses to display the graph.

Figure 6.7: 20 topic groups on the topic tree

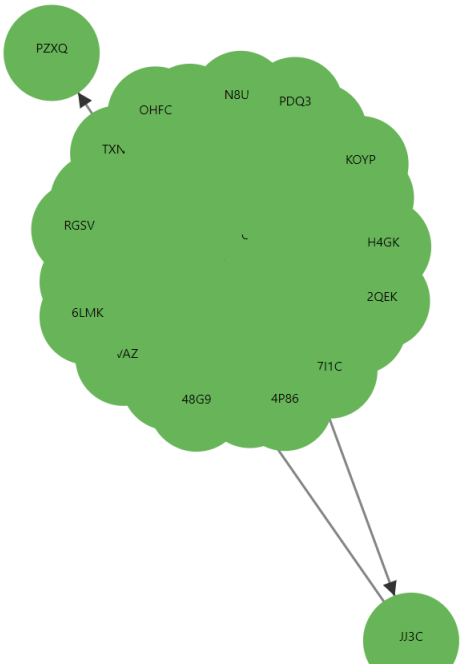

Figure 6.8: 100 topic groups on the topic tree

6.2.6 Feedback from Potential Users

Feedback from students, tutors and lecturers have been collected to analyse the user experience of the system and what could be improved.

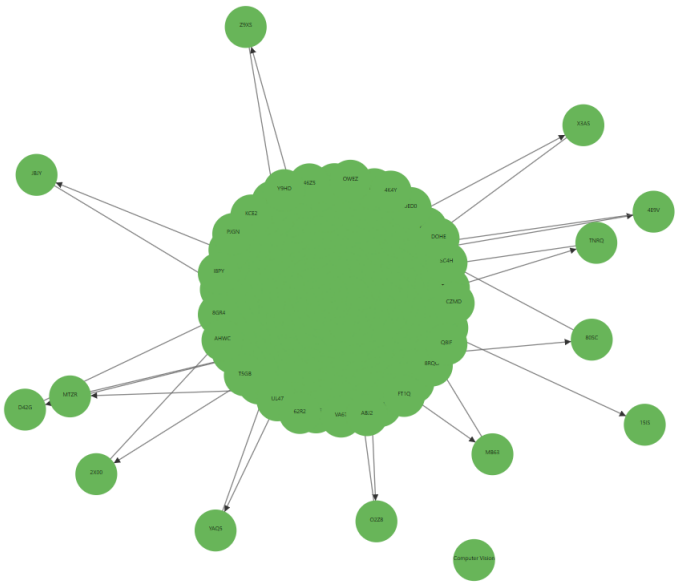

Figure 6.9: 300 topic groups on the topic tree

Audience

Feedback was collected from a range of fields including:

- Computer Science and Engineering;
- Humanities and Languages;
- and Business and Economics.

This range of different areas provides context on how usable the feature would be in their subject area or faculty, and also provides a diverse range of users.

Questions
These users were asked various questions before a demonstration of the feature, and then asked more questions after the demonstration as well. The questions are as follows:
Prior to Demonstration

1. What learning management systems have you used previously?
2. (Academics only) Have you experienced a situation where you have tried to reuse content? If so, what was it?

People from UNSW all answered the first question with Moodle and some answered WebCMS. One user from the University of Sydney answered Canvas, as they use a different learning management system.
Academics from the school of Computer Science and Engineering at UNSW answered the second question with "Yes", and most have explained that it was quite difficult having to create new content for the same topic in their courses. However, academics in Humanities and Languages explained that it was very easy to reuse course content in Moodle, as they only need to create a new offering of the course and have no need to reuse content from other courses.

After Demonstration

1. Is the system easy to use? What UI issues did you have with it?
2. (Academics Only) Would you use the system to help organise content better?
3. Is the terminology of the system such as topics and topic groups easy to understand?
4. Would you use the graph or list view of the topic tree more?
5. What are some positive aspects of the topic tree?
6. What could be improved?
7. Do you have any final comments or questions?

Most users commented that the system is quite easy to use, however there are some UI issues with the system. Most also agreed that it was quite difficult to see what the topic group was when expanded into their individual topics. There were also various small UI issues throughout the feature, but these did not affect the functionality or the ease of use of the topic tree significantly.

Many academics from the faculty of Computer Science and Engineering commented that the system would be extremely useful for managing their courses, as it allows reusability of content and specific topics can be set as prerequisites eg. Git can be set as a prerequisite for COMP6080 instead of having to reteach it.

However, academics from the faculty of Languages and Arts commented that they would have little use of the system in helping them organise content better, as they were so used to using Moodle which allows an easy way to offer the same course in a different year. As the content is quite linear i.e. the first topic must be completed before attempting the second topic, etc., the system may be of better use for Computer Science and Economics for example instead where the prerequisite topic tree is more

complicated.

Many users also responded that they would rather rename 'topic groups' to 'courses' instead, to allow easier use of the system and faster adaptability of the system. However, other users also commented that this may not be an issue as there are many terms within university that are confusing to both students and academics as well such as "course", "major", and "offering".

Similar trends were observed where users who would use the system to its full potential i.e. users from the Computer Science faculty, would use the graph view more than the list view more. However, academics from the School of Languages and Arts would prefer to use the list view. One user responded that it was good to have an option to use either as when looking for a course, it is faster to use the list view instead of the graph view but when looking at topics overall, the graph view proves more useful.

Many users found the structure of Content, Practice, Preparation and Assessment quite useful, as it can fit many learning models well. The various features such as searching the topic tree and adding tags to various topics also proved useful. Accessibility was another positive aspect of the topic tree, but the algorithm that picks colours to distinguish the groups of topics when expanded sometimes picks colours that are inaccessible.

Finally, some academics commented that a permission system is needed, as especially in faculties such as Arts and Economics, lecturers tend to not share content with each other due to copyright issues. As all topic groups would be shown on the topic tree, this would be a major issue without a permissions system available.

Overall, the feedback of the system was positive and most academics and tutors agreed that the topic tree feature would help create more structure to content and improve the reusability of coursework at university.

Limitations

There are various limitations with this structure of the survey, where a demonstration is shown and then questions are asked. A usability test would have made it easier to gauge the UI issues of the system, however due to the limited amount of time academics had to complete the survey, a demonstration structure was adopted instead. UI issues with the feature were still observed, and the feedback given was of high quality.

6.3 Course Pages

The main aspects that the features of the course pages that will be analysed and discussed are:

- Functional Requirements - have the features outlined been completed?
- Non-functional Requirements - how well do the features complete their objectives?
- Usability Tests - how do users utilise the features and what do they think of it?
- Challenges faced - what were the difficulties faced?

6.3.1 Functional Requirements

As stated before, there were a total of 14 functional requirements. Within these requirements there were 6 high priorty requirements, 3 medium priority requirements and 5 low priority requirements. In total only 9 of the functional requirements were completed. The 5 features that were not complete due to being low priority.

Completed

- Users can click on the links in the sidebar to be directed to the corresponding component;
- Users can view the announcments within the dashboard;
- Users can view the course content;
- Administrators of the course can create announcments within the dashboard;
- Users can make comments on announcments;
- Administrators can upload the course outline onto the course outline page;
- Users can download the course content;
- Administrators can edit the sidebar to change what components are linked;
- Course content is catergorised based on how the content is catergorised in the topic tree;
- Administrators can edit the announcements made in the dashboard;
- Users can intereact with widgets to enchance their experience with the LMS;
- Users can toggle on or off specific widgets.

Uncompleted

- Users can view the course outline of a course;
- Administrators can add course content into the page by selecting content from the topic tree;
- Administrators can upload the course outline onto the course outline page;
- Administrators can edit the sidebar to change what components are linked;
- Users can toggle on or off specific widgets.

The course outline feature was removed from the final design as it was not highly prioritised and other features being prioritised. The feature for administrators to add course content onto a course was streamlined, so that administrators can add content on the topic tree and have those changes be reflected on the course content page. The last feature that was not complete was the functionality of allowing users to toggle on or off specific widgets. This was due to the lack of priority for this feature and thus was not implemented.

Apart from the outlined requirements, there were also other functionality that were introduced:

- Users can view the courses that they are enrolled in;
- Users can select a course that they would want to view the course pages for;
- Users can view the most recent announcement;
- Users can view the progress of their courses;

These functional features were added early in Thesis B as they are the features that would make up the course selection page.

6.3.2 Non-Functional Requirements

The non-functional requirements that will be used to analyse the course pages are:

- Performance;
- Accessibility.
- Responsiveness;

Performance

Google Lighthouse was utilised to quantify the performance of the different aspects within the course pages. Google Lighthouse provides a simple way to test various aspects of a page, for example the performance, accessibility and assessing whether the website complies with best practices. The course pages that will be analysed are comprised of:

- Course selection page;
- Course dashboard.
- Course content page;

The performance of all of these pages all range within 40-50, indicating that there are a variety of further optimisations to better the performance of each page.

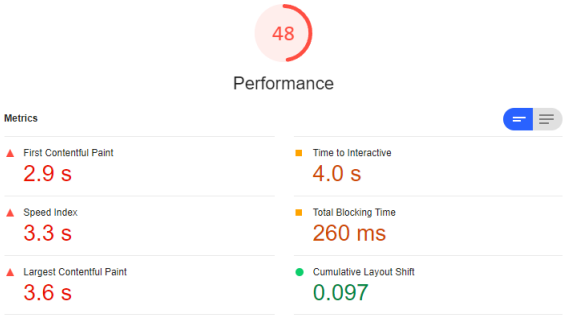

Figure 6.10: Performance report of the course selection page

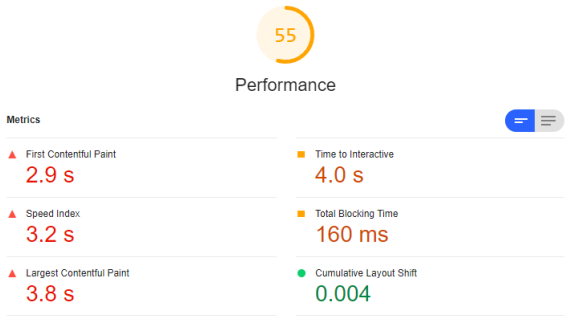

Figure 6.11: Performance report of the course dashboard page

As shown, the performance of the course selection page is 48/100, the course dashboard being 55/100 and the course content page being 41/100. This would be due to the variety of features within the page which would require numerous API calls. Possible optimisations could include, restructuring the use of API calls and database schema to improve fetch times and also minifying javascript files which would further improve

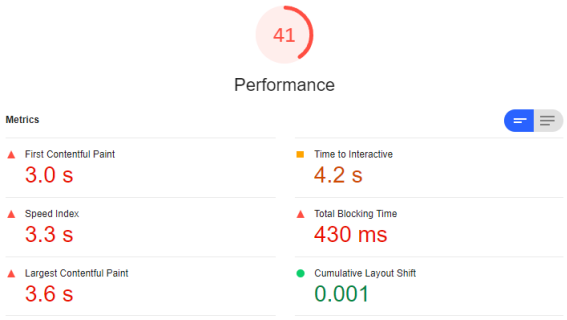

Figure 6.12: Performance report of the course content page

performance as this test was run on a development server. For the course selection page, there are a variety of features, such as the most recent announcement, course progress or most recently accessed topic. This page could be optimised by having the associated data be attached to a user, making only 1 API call needed. The course dashboard involves a variety of API calls, which include getting the announcement data, the user who made the announcement and other metadata. The course pages can be improved through redesigning the topic schema. The main concern was the multiple API calls needed to get the topic data and the corresponding files. If a topic contained a large file, then the API call to get the topic data would take longer.

Accessibility

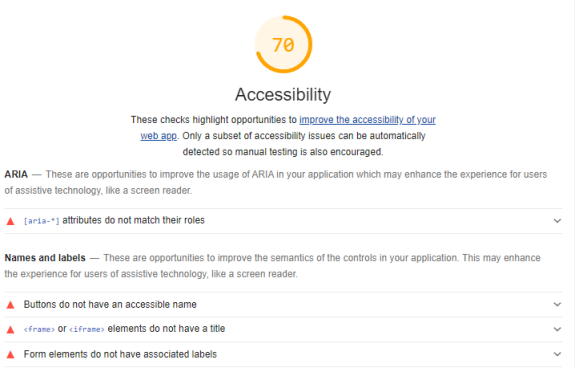

Figure 6.13: Accessibility report of the course selection page

As shown the accessibility of the course pages range within 70-80 and as such is one aspect that could be improved on with minor changes. With the use of Chakra UI, many aspects of accessbility within frontend design was already handled. For example aria labels and the roles of certain HTML elements were handled with Chakra UI. However some aspects such as colour contrast and certain HTML semantics can be improved. For example some buttons do not have accessible names for screen readers to be able to read.

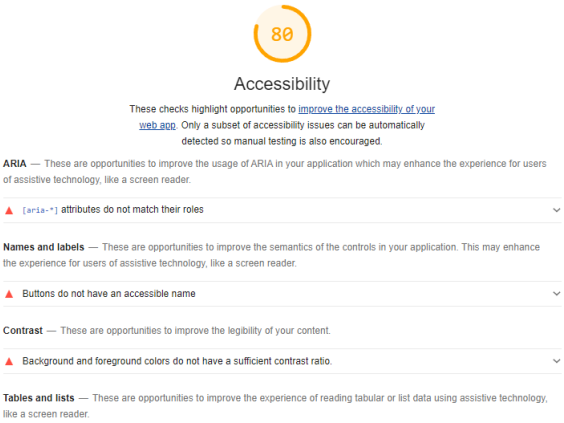

Figure 6.14: Accessibility report of the course dashboard page

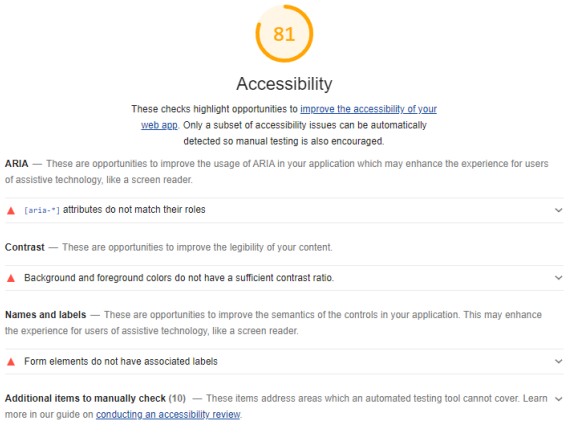

Figure 6.15: Accessibility report of the course content page

Responsiveness

The responsiveness of the course pages were one aspect of design that was deliberately designed. Each page has 3 intermediate stages, which correspond to screen widths, with the breakpoints being 990px and 765px. The main aspect of responsiveness is to provide the same functionality regardless of the screen size. This however has not been fully achieved within the course pages. The corresponding component affecting the responsiveness are the 2 sidebars, the navigation sidebar and widgets bar. As shown in the figures, the medium and small screen sizes do not have any widget functionality.

A way to provide this functionality would be to allow users to select an option when clicking on the account button on the top right of the screen which would open up the widgets bar. In this design, the functionality of the widgets bar would still be present within smaller screens and also larger screens, however is less accessible for users to navigate.

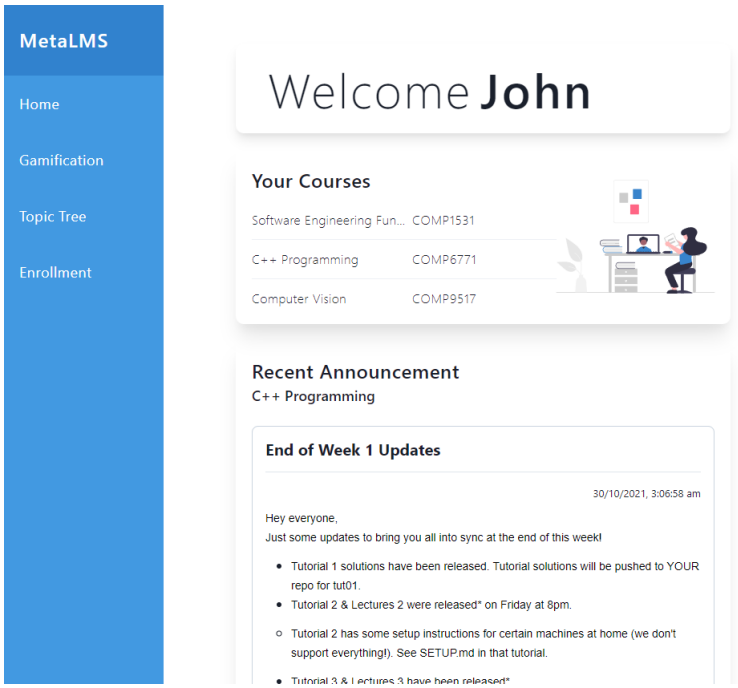

Figure 6.16: Medium screen size for the course selection page

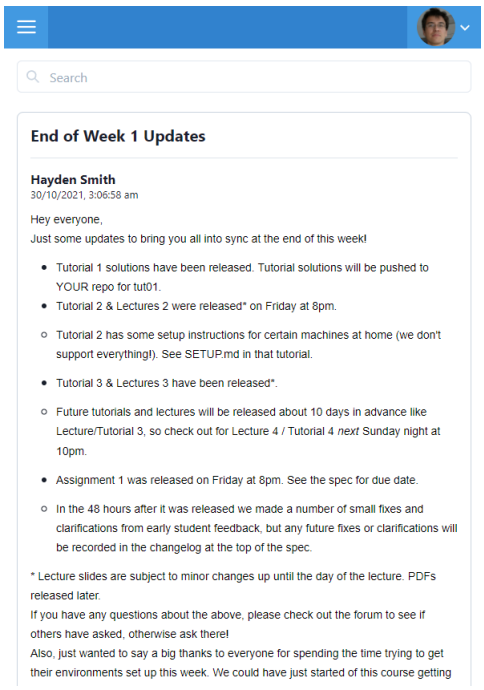

Figure 6.17: Small screen size for the course dashboard page

6.3.3 Usability Tests

The usability tests were conducted on a variety of UNSW students primarily studying computer science and engineering. For the usability testing a series of tasks were utilised to simulate several use cases for regular users and administrators. The following tasks were presented to the individuals in which the usability tests were conducted on:

1. Login as john smith.
2. Can you tell me what courses you are enrolled in?
3. Can you tell me the progress of the courses?
4. Can you identify anything else that is shown on the page?
5. Can you make a reminder on the calendar and delete it?
6. Can you tell me what the reminder section is for and what do you think of it?
7. Go to the C++ programming course home page and to the content of that course.
8. Can you find and open the while loops pdf file and signify that you have completed it?
9. Tell me the perquisites of the topic that the while loops pdf file is in.
10. On the same page can you navigate to another course?
11. Login as jane doe.
12. Can you identify what is shown on the page?
13. Go the C++ Programming page and create an announcement with the title: 'Week 2 updates' and content: 'Continue with your work and finish variable topic content for next week'.
14. Edit the newly created announcement so the content is 'Finish while loops content this week and next week we will announce a new assignment'.
15. Delete the announcement.

The "Ask a question" functionality on the course announcements page allows users to make a direct post to the course forums in relation to the announcement they were reading. When created via this method, a forum post will be automatically tagged with "Announcements" which could be used for filtering. This feature was tested in another separate usability tests, which were conducted by different participants. The task associated with this feature was to have the user to ask a question that is linked to a particular announcement.

Before beginning each individual usability test, each participant was given context of the topic groups, topics and topic files. Specifically the relationship between courses

and topic groups were explained, along with how topics contain 4 types of material (preparation, content, practice, assessments). Most users could navigate through all of the tasks with ease. However there were certain aspects that took longer than expected to complete or did not fully understand. These were:

- Not understanding the which announcement is for which particular course in the course selection page
- Not understanding the most recently accessed feature
- Not noticing the difference between the course selection page and course dashboard/content page.
- Difficulty to distinguish the separate topics and content types in the course content page
- Difficulting in how to ask a question that is linked to a particular announcement

These results highlight UI deficiency in providing a clear outline of allowing users to understand the functionality. For the first 2 points, help buttons or tooltips could be implemented to provide the user with the ability to search how the feature is intended to be used. The 3rd point involves the lack of unique distinction between the course selection and a particular course page. A possible solution would be to change the side bar colors when a user enters a particular course page. This would be similiar to WebCMS3 as it uses colours to distinguish different courses from each other. Another supplement to this solution to provide more distinction that a user can return back to the course selection page would be to highlight the return button. Lastly the accordion within the course content page would need to be updated to further distinguish items within a topic. This can be done by increasing the margin for each item or having different font weights for topics or topic content. In general the course dashboard had no complaints as it was a familiar feature for users to utilise. Some users also conveyed some concern with understanding the relationship between topic groups and courses, as they would not have understood how to utilise the MetaLMS without the background context. In this aspect, the MetaLMS would benefit with a general help page to assist users with understanding the several relationships. When given the scenario to ask a question to the forums that is linked to an announcement, some users were confused about how they could achieve this. This shows that despite the use of a question mark button and a tooltip to explain its purpose, the functionality is still unclear to some users. The main piece of feedback, however, was that this feature would be much more useful if the post created contained a direct link to the announcement. This would make it easier to trace the post back to the announcement. This functionality is something that could be built in future iterations of this component.

6.3.4 Challenges faced

The main challenges that were encountered in developing the course pages were:

- Understanding the semantic use of Chakra UI;
- Developing the specifications for the required API endpoints.

One of the main challenges was understanding how to utilise the frontend framework, Chakra UI. Chakra UI provides a unique method of enabling responsiveness and as such required time to fully understand. In combination with understanding how to utilise Chakra UI and its several components, developing the specifications and data for the required API endpoints was an aspect that was found to be challenging. Understanding the required data that needed to be stored was a large challenge that had to be overcome in order to progress with the development of the course pages. With the main course selection page containing a variety of features, a variety of endpoints had to be developed. For example the course progress had to be stored for each user, along with what topics they had completed. These challenges were solved eventually through thorough experimentation and reading through documentation.

6.4 Lectures and Tutorials

The lectures and tutorials component will be assessed on based on the following:

- Functional Requirements
- Non-functional Requirements

6.4.1 Functional Requirements

The functional components for the lectures and tutorials component have been satisfactorily completed as 14 out of 14 requirements have been completed. Listed below are the functional requirements for this feature.

Lectures and Tutorials

1. Users can toggle lectures and tutorials subsection visibility

Video Streaming

1. Users can watch a live lecture
2. Users can watch a live tutorial
3. Instructors can stream a live lecture
4. Instructors can stream a live tutorial

File System

1. Users can search for a file
2. Instructors can upload files to tutorials subsection
3. Instructors can upload files to lectures subsection
4. Users can download a recorded lecture
5. Users can download a recorded tutorial
6. Users can download tutorial files
7. Users can download file files
8. Users can view tutorial files
9. Users can view lecture files

6.4.2 Non-Functional Requirements

The non-functional requirements for this component have been completed to an adequate level, as 3/3 requirements have been satisfied.

Non-Functional Requirements

1. Usability - The feature must be efficient and effective
2. Performance - The feature must operate within a reasonable time frame
3. Aesthetics - The feature must be visually pleasing
4. Learnabilty - The feature must be easy to learn and use
5. Accessibility - The feature must accommodate varying users

Usability

The lectures and tutorials component is efficient and effective to use as there is a minimal amount of clicks to complete a desired task. Additionally, the component is efficient as each action is performed relatively well and quickly.

Performance

The component suffers from an occasional bug where an uploaded file does not render immediately. As a result this requirement is somewhat completed. Otherwise, the other functions such as deleting, occasional uploading, updating weeks are relatively quick in performance.

Aesthetics

The aesthetics of the component are adequately pleasing as the colours and spacing are well put. Additionally, the inclusion of icons and hover effects also add to the aesthetics of the component. However, there is room for improvement in this requirement such as allowing for different colours and fonts for the component to add for visual effects.

Learnability

The learnability requirement is sufficiently satisfied as the functionalities of the buttons are clearly labelled and inferred by the icons. For instance, the red trash icon indicates a delete functionality for the feature. Moreover, as stated in the usability tests most participants were able to complete the tasks within 5-10 seconds which indicates the learnability requirement being satisfied.

Accessibility

As for the lectures and tutorials feature, there were no features implemented that took into account accessibility. In the future, there could be additional work such as a screen reader and a font changer to accommodate all users.

6.4.3 Usability Tests

This subsection will depict the results from usability tests conducted focusing on the lectures and tutorials feature of the LMS. This was tested on 5 anonymous participants whom are familiar with learning management systems such as Moodle and Canvas.

Results

The results of the usability tests are conveyed below:

- 5 out of 5 participants were able to complete all of the tasks given

- The time taken to complete each task was on average 5-10 seconds which indicates that the feature is easy to understand and use

Feedback

Some feedback provided by the participants were:

- The lectures and tutorials components could be combined into a single page since both components were similar
- The placement of the add new weeks was ambiguous and could be better placed on the page
- There should be a visual representation of the videos in question for aesthetic reasons
- An in-browser document viewer would save users time if they wanted to look at only certain parts of a file
- Some customisation features for the pages is desired
- There should be a feature to drag multiple files to upload to the relevant week
- A file repository system for the course would be great as the goal of the Meta LMS was for re usability

6.4.4 Challenges

The challenges faced when implementing the lectures and tutorials component was understanding how to use the backend endpoints to create the frontend for this feature. As there was unfamiliarity with integrating both, this led to a delay in completing this feature. Additionally, there was also an issue with how to link to an external video provider to host lectures and tutorials. Moreover, there were some issues with using the ChakraUI on the frontend which also resulted in a delay in the feature completion date.

6.5 Assessments

The Assessments feature will be examined based on the following:

1. Functional Requirements - whether the system provide enough functionality for the user to complete their task

2. Non-functional Requirements - whether it satisfies the goals of the feature and performs operations in a timely manner
3. Usability Tests - if users can navigate through the feature and be able to complete tasks.

6.5.1 Functional Requirements

The order of requirements that were implemented was based on the priority level of each requirement - [Must Have] being a core feature and "Could have" being the lowest priority.

In total, 9 requirements out of 30 were fully completed. This resulted in 8 out of 14 [Must Have] features being implemented. Below are the mentioned completed requirements:

Quiz Creation [1-3, 9]

1. Course lecturers can create a quiz [Must Have]
2. Course lecturers can add questions to a quiz [Must Have]
3. Course lecturers can create different types of quiz questions (multiple-choice, short answer, checkboxes) [Must Have]
4. Course lecturers can set up a timer or due date for a quiz [Must Have]
5. Course lecturers can add a question to a question bank [Should Have]

Viewing quiz results and feedback [1, 4]

1. Students can view results against their answers [Should Have]
2. Students can view an explanation of the correct answer if answered incorrectly [Could Have]

The following explains the requirements that were not completed and why they were not completed:

Quiz creation [5-7, 8, 10], Quiz modification/removal [3, 4], Quiz usage [4]

1. Course lecturers can add "drag and drop" type questions [Could Have]
2. Course lecturers can add "connect the pairs" type questions [Could Have]

3. Course lecturers can add media (audio or video) into a question as the entire question or as a supplementary material to the question [Could Have]
4. Course lecturers can create a question bank [Should Have]*
5. Course lecturers can import a question from a question bank [Should Have]
6. Course lecturers can remove a question bank [Should Have]
7. Course lecturers can remove a question from a question bank [Should Have]
8. Students can manually save their progress at any time [Should Have]

These were not completed due to the core question types (multiple-choice, short answer and checkboxes) unexpectedly taking a lot of time to implement due to bugs and the nature of the answers being formatted differently (with multiple choice and checkboxes being given answers that you must select from and short answer being a free answer the student must enter). There were difficulties when allowing the quiz creator to switch between question types and structuring the data storing those choices to work with all question types.

*For simplicity sake, instead of allowing the course lecturer to create a question bank, the default is a single question bank exists to store all questions that want to be used in future quizzes.

Viewing quiz results and feedback [2, 3]

1. Students, course lecturers and admins can see how many students selected each answer for a question [Should Have]
2. Students can view topic or lecture the question derives from [Should Have]

Poll feature and Re-usability

These were not implemented due to the Quiz portion taking more time than expected and multiple re-designs of how to store the data for each question. However, there are placeholder entry points on the frontend and API endpoints that available for the re-usability portion (importing questions from question bank).

6.5.2 Non-functional Requirements

The non-functional requirements used is heavily inspired by the Jakob Nielsen's 10 usability heuristics that's used to evaluate the usability of user interfaces.

1. **Efficiency** - users are able to perform tasks without taking too many steps
2. **Learnability** - new and returning users are able to quickly learn how to use and interact with the feature on the go
3. **Performance** - operations within the feature are completed within a timely manner, resulting in a smooth process
4. **Consistency** - components, colour themes and formats are consistent across pages
5. **Aesthetic and minimalist design** - the user interface is appealing and simple while providing the expected features
6. **Re-usability** - users are able to re-use components and content they've previously made

Efficiency

From what the user interface provides, there are multiple options provided for the same action (such as expanding a question by clicking on the Question accordion header or by using the Question List tool on the left column on the Quiz Creation page) as well as useful tools. This allows the user to perform specific tasks faster and more efficiently.

Learnability

This cannot be tested with returning users since no usability test was done in early iterations of the feature, however can be evaluated with new users using the results from the usability tests which is in the next section.

Performance

Due to the frontend of the Assessments feature being partially connected to the backend due to the API calls being denied for unknown reasons, the performance cannot be evaluated properly.

Consistency

The colours themes used are consistent across pages, however the formatting and components differ per page. On the quiz creation page, there are 3 columns while the quiz usage page has 2 columns and the quiz review submission page has 1 column, where the content is vertically aligned in the center. This may be satisfactory, but could be improved upon future iterations.

There is also a combination of icons and words being used for icons which might make the user interface more confusing or difficult to navigate.

Aesthetic and minimalist design

The user interface looks somewhat appealing, however can have improvements made to be more appealing, such as the quiz details section on the quiz creation page. The design for the quiz creation page is not minimalistic due to the extra ability to remove, edit and add new answers. The design for the quiz usage and review submission page is quite minimalistic.

Re-usability

There is currently no re-usability support and so this requirement has not been completed.

6.5.3 Usability tests

The following are results from the usability tests focusing on the student-related pages (Quiz Submission and Review Submission pages), which was conducted on 6 participants who are currently or were students that have used at least one learning management system before.

Results

Below is a summary of the results recorded while the participants were completing the tasks specified in the usability test plan:

- 2 out of 4 participants were able to complete all tasks with no misclicks, errors or detours
- The main cause of errors was due to the outline of input boxes and selection buttons (eg. radio and checkbox buttons) being very light and weakly contrasting with the white background
- Each task was completed on average between 10-20 seconds by each participant which suggests the interface was usable and quick to learn for being first time users.

Feedback from participants

The main pieces of feedback were:

Quiz Submission page:

- Selection options and input boxes were too light and difficult to see
- At the last question, have a "Submit quiz" button appear next to the "Previous Question" button to maintain the 2 button format (that was "Previous Question" and "Next Question" for middle questions)

Quiz Review Submission page:

- Moving between questions one at a time may be better so students can focus on reviewing a particular question
- Providing links to any referenced materials (eg. lecture slide PDF) would allow faster revision and be convenient
- Have a box around the Answer Explanations section to indicate that it is not part of the original question
- Have a cross at the end of incorrect answers and a tick at the end of correct answers

6.5.4 Challenges faced

The main challenges when implementing the Assessments feature was creating a generic data structure that allows multiple question types to be supported, and providing the ability to change the question type (when creating a question) without causing unexpected issues. This slowed down the progress and as a result, left no time in implementing the re-usability portion of the Quiz feature, and the Poll feature altogether. Due to this, the data structure was re-designed multiple times before it was finalised and was working consistently. This resulted in significant modifications to the Quiz API endpoints being done later than expected (roughly midway through Term 3) and due to technical issues with API requests always being rejected, there was not enough time to implement it into the final implementation.

6.6 Forums

The forums feature will be examined based on a framework to determine how well the implemented feature has achieved its intended purpose and its functionality.

The feature analysis includes three sections:

1. **Functional Requirements** - does the forum component meet the functional requirements defined in the project approach?
2. **Non-Functional Requirements** - does the forum component meet the non-functional requirements defined in the project approach?
3. **Usability Tests** - what do potential users of the system think about the forums and how could it be improved?

This section will also contain a reflection on how well milestones were met compared to the timelines set at the start of this project, as well as the challenges faced throughout this project.

6.6.1 Functional Requirements

The functional requirements of the forums are evaluated based on whether or not they were implemented in the system. A requirement is considered complete if most or all of its acceptance criteria have been met. The priority of the requirement has also been taken into consideration. If a "Must Have" criteria has not been completed, then the requirement also cannot be considered completed. Requirements and criteria marked as "Won't Have" are not considered in this evaluation.

Below is the list of functional requirements and their acceptance criteria, each marked with their status of completion.

1. Users can view a list of forum posts (Completed)
    - Users can see all forum posts in a table (Completed)
    - Users can see all pinned posts in a separate table (Completed)
2. Users can clearly see which posts have been read and actioned (Completed)
    - Users can easily find answered posts (Completed)
    - Users can sort posts by number of replies or comments (Completed)
    - Users can see which posts have been endorsed (Completed)
3. Users can add a post to the forum (Completed)

- Users can add a post with a title and description (Completed)
- Users can format their post description (Completed)
- Users can optionally attach files (Partially Completed)
- Users can optionally attach tags (Completed)
- Users can optionally attach links (Completed)

4. Users are able to search for a forum post (Completed)

    - Users can search a post's description or title (Completed)

5. Users can filter through forum posts (Completed)

    - Users can filter forum posts using pre-defined tags (Completed)
    - Users can filter posts by those linked to announcements (Completed)
    - Users can filter posts by those that are answered (Completed)
    - Users can filter posts by those that are unanswered (Completed)
    - Users can filter posts by those that are endorsed (Completed)

6. Users are able to sort the forum posts (Completed)

    - Users can sort forum posts by different headings in the post table (Completed)

7. Users can view the individual post details for a chosen post (Completed)

    - Users can navigate to a post page for an individual post (Completed)
    - Users can easily see the post title, description, tags, author and data created (Completed)
    - Users can see all the responses to the post (Completed)
    - Users can see all the comments on the post (Completed)

8. Users can share a forum post with others (Partially Completed)

    - Users can copy a link to a forum post (Completed)

- Users can directly share a forum post to other platforms (Not Completed)

9. Users can upvote a forum post (Completed)

    - Users can upvote a post on the forum overview page (Completed)
    - Users can upvote a post on the forum post page (Completed)
    - Users can see the number of upvotes a post has (Completed)

10. Users can edit their post (Completed)

    - Users can edit a post after it has been posted (Completed)

11. User can delete their post (Completed)

    - Users can delete a post after it has been posted (Completed)

12. Users can comment on a post (Completed)

    - Users can leave a comment on a forum post (Completed)
    - Users can format their comment (Completed)
    - Users can edit their comment after it has been posted (Completed)
    - Users can delete their comment after it has been posted (Completed)

13. Staff can manage the tags for the topic group (Completed)

    - Staff can add new tags (Completed)
    - Staff can remove tags (Completed)
    - Staff are unable to create duplicate tags (Completed)
    - Students are unable to create tags (Completed)

14. Staff are able to pin posts to the top of the forums (Completed)

    - Staff can pin posts to the top of the forums (Completed)

- Staff can unpin posts to the top of the forums (Completed)
- Staff are unable to pin and unpin posts (Completed)

15. Staff can reply to a forum post (Completed)

    - Users can leave a reply on a forum post (Completed)
    - Users can format their reply (Completed)
    - Users can edit their reply after it has been posted (Completed)
    - Users can delete their reply after it has been posted (Completed)

16. Staff can endorse a post or comment (Completed)

    - Staff can endorse a post (Completed)
    - Staff can unendorse a post (Completed)
    - Staff can endorse a comment (Completed)
    - Staff can unendorse a comment (Completed)

17. Staff can directly link or embed materials to the forum post (Partially Completed)

    - Staff can add a direct link to a reply (Completed)
    - Staff can embed course materials in a reply (Not Completed)

In total, 15 out of the 17 functional requirements were completed. The remaining two functional requirements were only partially completed. Each of these functional requirements had one acceptance criteria that was not implemented however, both of these were lower priority tasks. Overall, all of the high and mid priority requirements were implemented into the system.

Below is a list of incomplete requirements and acceptance criteria, and the reason behind why they were not completed.

Users can share a forum post with others

Users can directly share a forum post to other platforms (Could have)

The original idea was that the share button on individual posts would provide users with various platforms, like Facebook Messenger and WhatsApp, in which they could select to send a direct link to the forum post to. As this was a lower priority task, time constraints meant that it was unable to be completed on time. Furthermore, since this functionality could be achieved by using the copy link to clipboard button that was built early in the implementation process and pasting the link to share it, it was decided that the focus should be shifted to ensure higher priority tasks are completed instead.

Staff can directly link or embed materials to the forum post

Staff can embed course materials in a reply (Could have)

The ability for staff to directly embed course materials into their reply to a forum post was a low priority feature that would've been nice to have. This would've allowed staff to easily provide students who are asking for help with references and materials that could assist them. Similar to the previous requirement, there was not enough time to build this feature and since staff already had the option to include a link to resources in their reply, this criteria was considered unnecessary.

Users can add a post to the forum

Users can optionally attach files (Should have)

While this requirement has been marked complete, it is important to note that the current state of the forums only allows for images to be attached to posts. This criteria has been marked as partially complete as ideally, documents and videos should be able to be attached as well. This was unable to be done, however, due to time constraints and lack of knowledge in handling these file types. However, since a user can still add a new post, and screenshots are the most popular type of attachment in forums, this overall requirement has been marked as completed.

6.6.2 Non-functional Requirements

This section contains an evaluation of how performant and accessible the forum component is.

Using Google Lighthouse, tests were run and the forum component was given a score out of 100 for performance and accessibility. Google Lighthouse also produced a report

with more details on ways in which the forums could be improved to produce a higher score.

Performance

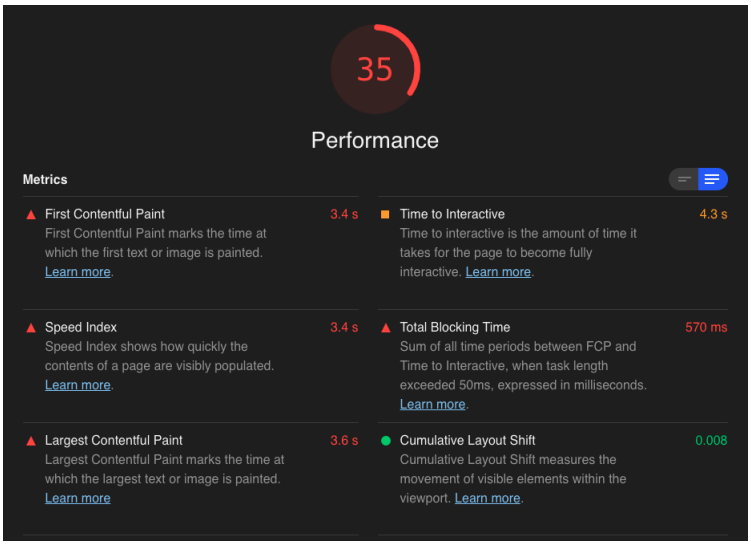

Figure 6.18: Performance Report for Forums Overview Page

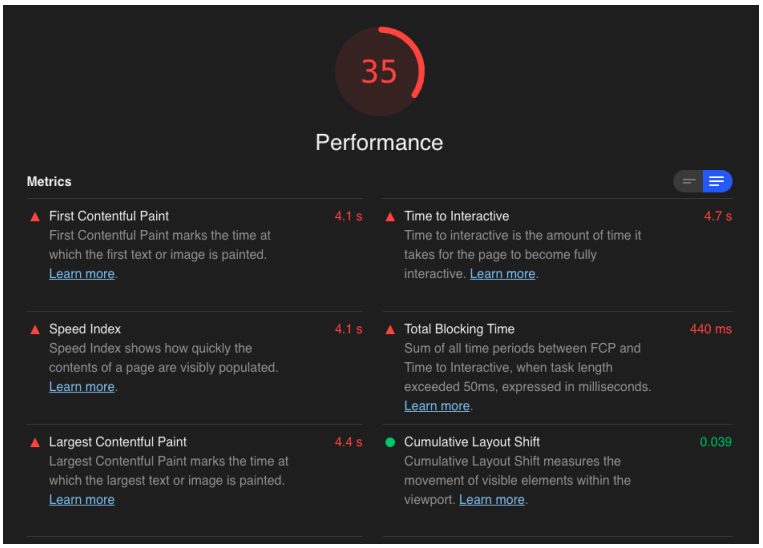

Figure 6.19: Performance Report for Forums Post Page

As seen from the figures above, the performance score for the forums are quite low, with a 35 out of 100 for both the overview and post pages. Compared to the standards set by high-performing websites, the metrics measured show that the contents of the forum pages are slow to render and become interactive. For a user with reasonable internet speeds, the slower renders would not be too noticeable. The performance

issues would mainly become a problem for users who have slow or throttled internet connections. Despite this, performance is still an important factor when building a website and should definitely be improved in future iterations of the forum. Refactoring the backend API and database to have a heavier focus on performance would be required to achieve this. The report also suggests that reducing the amount of unused JavaScript could significantly reduce the load times. This should definitely be addressed in future iterations.

On a positive note, the report shows that the forums pass the "Cumulative Layout Shift" metric. This means that once the page has loaded, there is very minimal movement within the viewport of the user. This is vital as it reduces the likelihood of misclicks for users.

Accessibility

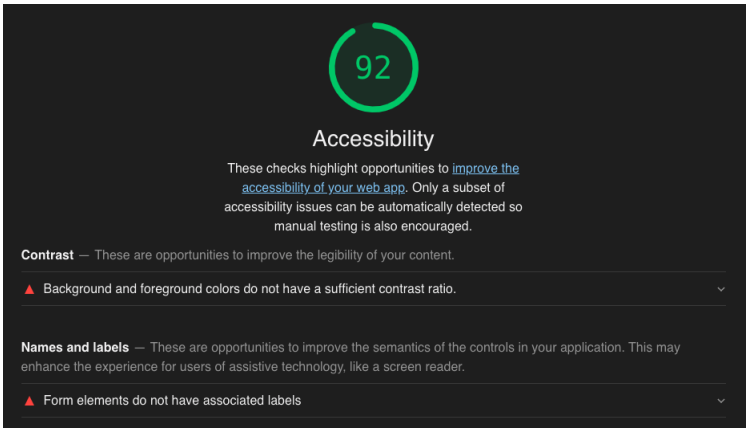

Figure 6.20: Accessibility Report for Forums Overview Page

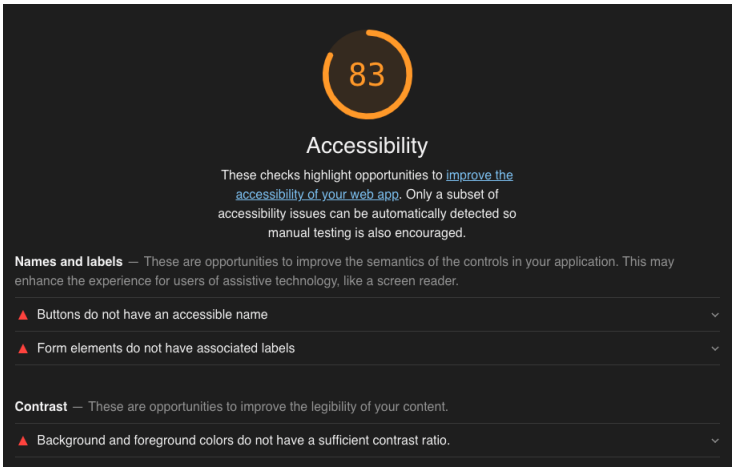

Figure 6.21: Accessibility Report for Forums Post Page

As seen from the figures above, the accessibility score for the forums was relatively high, with a 93 out of 100 for the overview page and a 83 out of 100 for the post pages. A high accessibility score is essential for ensuring that all users, including those with disabilities, are able to use the system. The score received for the forums is most likely due to the use of the Chakra-UI library which automatically applies a lot of the tags required for accessibility. Even though these scores are quite high, the report suggests some easy improvements that can be made in future iterations to further improve the accessibility of the site.

Some of the colours used on both the overview and post pages do not have enough contrast with the background. This means that people who are visually impaired may have trouble reading the words. Since the background colour of the forums is white, making all the text darker on both pages would easily solve this problem.

A screenreader is a popular device for some people with disabilities as it can read out the components and text on a page. As a result, it is important that all images, forms and buttons have alternative text that describes its purpose that can be read out by the device. The report shows that on the post page, some of the buttons do not contain this alternative text. In order to ensure that the button can be read out to users, an accessible name should be added to these buttons. Labels should also be added to each of the input fields on the page, including those in the "Add Post" modal and in the responses and comments section of the post page.

6.6.3 Usability Testing

Simple tests were run with tutors and students to evaluate the usability of the forums component and receive feedback on ways in which the system could be improved.

Audience

In order to get a wholistic understanding of the usability of the system, participants from various fields were used to test the student's perspective of the forums component. This is to test that the forums are appropriate for use in all types of courses. These fields included:

- Engineering;
- Medicine;
- Science; and,
- Art and Design.

Student's Perspective

To test the forums from a student's perspective, student participants were first asked to look at the overview and post page and provide their initial thoughts.

For the forum overview page, most participants stated that on first look, they liked the way in which the posts were laid out. They mentioned that the layout was pretty straight forward and easy to understand. They also liked that the pinned posts were separated and located at the top of the page.

Some users, however, found that the page looked quite cluttered. This was mainly seen in the search and filter areas at the top of the page. This could be improved by making the search and filter areas collapsible so that students can hide them from view when not being used. They also mentioned that the indentation on the left of the post table looked weird and was a waste of horizontal space. When viewing the overview page from a staff perspective, this would be where the pin icon shows however this is not the case for students since they are unable to pin posts themselves. The indentation could easily be fixed by removing the spacing or by showing the pin buttons but not making them clickable for students.

For the forum post page, all participants were initially confused by the responses and comments section. They were unable to understand, just from looking at the page, why there were two separate sections. When looking at a post that has some comments and replies, this confusion was cleared up a bit for some participants. Some users were also confused by how the comments were displayed, and were unsure if the input field was for creating a new comment or to reply to the comment above. They suggested adding placeholder text to the input field to make this clearer.

The student participants were then given various scenarios to test out different features of the forums. For each of the scenarios, the students were asked to rate the process required to complete the task out of five, one being difficult and five being simple.

The first scenario asked the students to add a new post with a tag to the forums. All participants found this process very simple and intuitive, giving the process a five out of five. Participants found that it was easy to work out how to add a post due to the large call-to-action on the overview page. They also liked that the button opened up a modal instead of making them navigate to a different page. One participant suggested moving the tag selection to the top of the "Add Post" modal since it is easy to miss when it is at the bottom of the form. This is something that will definitely be implemented in future iterations of this project. Additionally, some students noticed and interacted with the edit button on the newly created post. The main feedback that arose from this was that students would like to also be able to edit the title and tags for the post.

The second scenario required participants to filter the posts by a specific tag. All participants found this process simple, giving it a five out of five for usability. Some students used the dropdown menu to find the tag to filter by while others typed the tag name into the filter bar. This shows that having both options is useful to users and it

isn't redundant to have both. Some users liked the fact that the filter displayed at the top of the page and wasn't hidden away on the side or in a menu, while others thought it cluttered up the overview page a bit. Due to the way it was built, the filter does not persist between pages meaning that if a user filters by a tag and then goes through each of the results, they would have to re-filter the posts each time they return to the overview page. This frustrated some participants as they expect the filter to remain when they navigate back from a post.

Participants were then asked to find a specific post and leave a reply. In order to complete this task, some participants used the search bar, some used the sort functionality in the table and some just scrolled until they found the post. This proves that users interact with the forums in different ways and shows that none of this functionality is useless. Despite the method they used to find the post, all users gave this process a five out of five. All participants also found it very easy to leave a comment and didn't have much feedback for that process.

Participants were also asked to edit their previous comment. They all gave this process a five out of five for usability, with the size and positioning of the buttons making it easy to work out how to complete this process. The only feedback received for this was that the participant expected the timestamp on the comment to be updated to reflect when they made the change.

Participants also found that it was extremely easy to upvote a post, with them all successfully completing the scenario and giving it a five out of five. They appreciated the fact that they didn't have to click into a post to upvote it if they didn't want to, they could just upvote it from the overview page. For this scenario, the participants were also asked about how they would rate the usefulness of this feature. This was to get an understanding of whether or not the feature would actually be beneficial to users. Pretty much all of the participants said that they personally would not upvote a post themselves however, they would definitely use the number of upvotes a post has to determine which ones were worth looking at. As a result, on average, they gave the upvoting feature a 4.5 out of five in terms of usefulness.

Finally, the students were asked to share a post. Most users were easily able to complete this task but some were confused about the icon on the share button and expected it to have a different functionality to sharing. One user also didn't notice there was a share button and instead, copied the url from the url bar of the browser. As a result, this process was given an average score of 4.5 out of five for usability. When asked about the usefulness of this feature, most participants were unsure if it provided much value to the system since the url could just be copied from the browser. They suggested that implementing the ability to share posts to specific platforms would make it more useful. This is a feature that was planned however, was not achieved due to time constraints.

Overall, all student participants found that the system was easy and intuitive to use. Most enjoyed the layout and design however, there is still improvements that could be made to please more users.

Staff's Perspective

Due to having limited time with the staff members, the usability tests for the staff-only features of the forum mainly consisted of showing the participants the overview and post pages and letting them interact with it whilst providing their thoughts and feedback.

The feedback for the forum overview page was mainly positive from all participants. Most participants liked the fact that they could pin posts to the top, and mentioned that it was a feature that they would definitely use. Some participants commented on the pinned posts also showing in the list of all posts and while they personally did not like this concept, they said that it didn't hinder their understanding of which posts were and weren't pinned.

There were some mixed reviews on upvotes with some participants saying that they liked the feature and would definitely use it to work out which posts require their attention first. Others said that they prefer to just scroll and answer posts chronologically and therefore, wouldn't use the feature.

Some features that aren't included in the forums that participants said they would like to see included a "new" status on forum posts that they haven't seen yet. They said that this would be the most useful in working out which posts to look at first. Participants also suggested having a way to identify if a post on the overview page had been replied to by them or a different staff member. However, they also noted that while this would be a nice feature, they preferred the existing, clean interface of the post tables compared to one with too many columns.

When it comes to tags, some participants liked the feature while some were impartial to it. Those who liked the tags thought that it was a good way to categorise and group posts together. Some participants, however, thought tags may be unnecessary if the search functionality of the forums is good enough. They were also concerned about whether or not students would actually use the tags, and whether they would be used correctly.

Like the student participants, the staff were also initially confused about the difference between responses and comments. This issue could potentially be fixed by renaming the responses section to "Official Responses", or by adding a description of what each section means below the title.

Participants also had mixed feelings about endorsing posts and comments. Some mentioned that it felt too easy to endorse a post and suggested adding a confirmation message to ensure that the staff member hasn't misclicked the button. Others were just unsure about how endorsing could be helpful to them however, once it was explained further, they were able to see the value of the feature. This shows that there needs to be more explanation around what it means to endorse a post and how this can benefit the user. This could be done by adding a tutorial tooltip to the endorse button when it is first clicked by the user.

Overall, the staff participants were happy with the forums and liked all the functionality it provided.

6.6.4 Timeline

Throughout the duration of this project, many of the originally planned tasks were able to be completed ahead of schedule. This allowed for additional features like endorsed posts, enhanced filter and upvoting, to also be implemented. Overall, majority of the important features were completed with enough time to conduct sufficient analysis of the system.

6.6.5 Challenges

Most of the challenges faced throughout this project occurred when working with third-party packages and libraries.

The initial rich-text editor package for React was quite difficult to work with as there was poor documentation and plugins had to be added to handle some functionality. For example, one of the requirements for this feature was that users could embed links in their forum posts however, the original editor needed a plugin to make this possible. The lack of documentation made it difficult to get these plugins installed and working as expected. As a result, a different editor that had these functionalities already built in was used instead.

Another package that was difficult to work with was `react-table`. Some of the columns in the table, like the upvote column and the pin column, displayed elements other than just text. At first, it was difficult to work out how to achieve this however, researching the issue revealed documentation that explained how to display custom elements in the body of the table.

While most of the milestones for the forums component were able to be completed on time, there were some features that couldn't be fully implemented until close to the end of the development process due to dependencies on other LMS features. This meant that some of the analysis and evaluation processes had to also be delayed. This, however, is a challenge typically faced when developing a large system in parallel and was able to be overcome through open communication and discussion with other team members. Despite this, all the main requirements for the forums were eventually able to be completed before the due date.

6.7 Gamification

The gamification feature will be examined based on a framework to determine how well the implemented feature has achieved its intended purpose and its functionality.

The analysis includes four sections:

- Functional Requirements - does the gamification feature fulfil the functional requirements set out in the project approach?
- Non-functional requirements - how well does the feature achieve the non functional requirements including performance and accessibility
- Scalability - how well does the feature perform with many topic groups, levels and users accessing the feature concurrently?
- Feedback from potential users of the feature - what do users such as lecturers, tutors and students think about the feature and how could it be improved?

6.7.1 Functional requirements

The functional requirements of the feature were analysed for completedness for both students, admins and backend.

Requirements for Students

7 out of 8 functional requirements for students have been completed.

As a student, I need to be able to view all my enrolled topic groups and access weekly activities - Incomplete requirement

The current implementation allows students to access the levels assigned to them through the topic groups they are enrolled in but it does not explicitly display the topic groups they are enrolled in on the gamification platform. The implementation of weekly activities was not completed as well due to time constraints. This is a low priority requirement as it is an additional feature on top of the core functions of the platform. This would be classified under future work for the platform.

Requirements for Academics

4 out of 6 functional requirements for academics have been fully completed with 1 functional requirement partially completed.

As an Academic, I need to be able to create new timed questions of types (MCQ, True / False, Coding) - Partially completed requirement

The question types of MCQ and True / False style questions have been implemented, however due to time constraints, the Coding type was not implemented. This was a decision made halfway through thesis C to omit this as it is a large complex piece of work but completion of other requirements was more important than this specific part of the requirement.

As an Academic, I need to be able to customise the rewards awarded to students on completion of questions in modules. (Rewards Management) - Incomplete requirement

Currently implemented is only points awarded on completion of a level. This requirement relates to badges and achievements that admins can customise for their own topic groups. This requirement was not completed due to time constraints and would similarly be classified as future work for the platform.

Requirements for Backend

8 out of 8 functional requirements for backend of gamification have been fully completed. All backend requirements support the frontend requirements and this was prioritised throughout the thesis to ensure that frontend requirements are able to be implemented.

6.7.2 Non functional requirements

Performance and Accessibility

There is room for improvement in terms of performance in the platform's current implemented state. There are no major performance issues that users will be able to observe but rather slighly longer page load times due to some heavy overheads of javascript that is being loaded on the platform. This is placing unecessary stress on the system that could be cleaned up as part of future works. Code clean up has been limited due to the time constraints and in hindsight, more time should have been allocated to refine and ensure code base is as clean as possible to decrease loading times.

The implemented platform scores well for accessibility due to the Chakra UI framework automatically adjusting to device size, allowing compatibility with screen readers and neatly tagging user interface elements approriately.

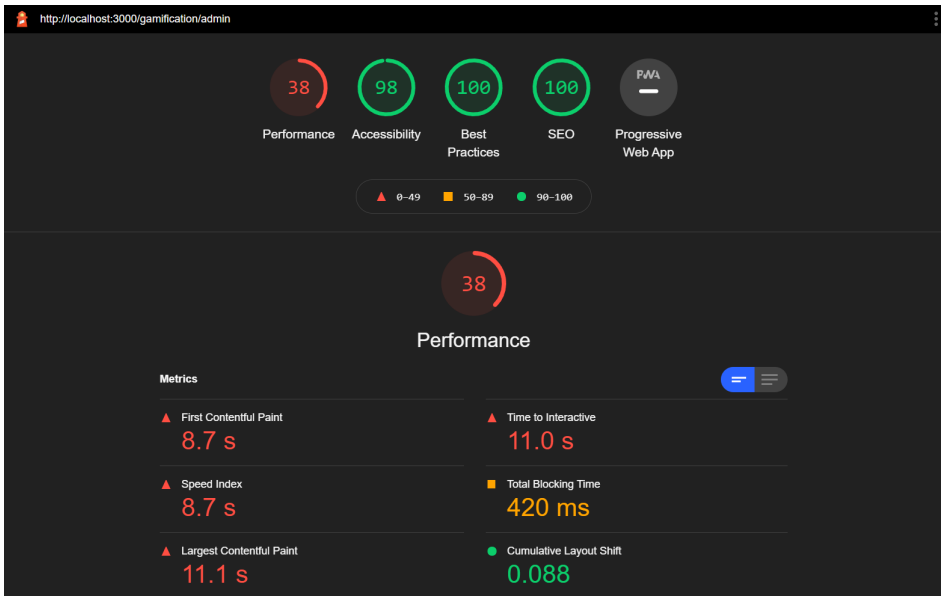

Figure 6.22: Google Light House Report

User Interface is easy to navigate

The gamification platform pages have been based on the same core fundumentals of listing items in a grid list. This makes every page feel familiar for users and provide a consistent experience throughout. It also provides the flexibility in displaying different data objects using customised tiles throughout the platform. Overall this has been a great design choice and has helped to create a simple and easy to understand journey for both students and admins.

Reusability

User Interface design elements have been reused extensively. Examples include the grid list interface that is used across topic groups, levels, student dashboard, shop and repository. This decision has reduced the amount of development effort can is easily extendable to other use cases on the platform. The edit question and edit level pages are based on the same reused elements, simplifying and standardizing the look and feel for form submissions. Prompts and alerts are also reused to reduce development efforts. The repository page was designed to directly support reusabilty of levels across the whole platform by allowing admins to search for levels and add it into their own topic groups. Overall, the reusability score for the gamification platform is high and it has helped greatly in maximising the requirements implemented in the amount of time available.

6.7.3 Scalability Testing

The gamification platform scales well, the user interface are setup with grid lists which have no limits. This means that there is no limit on how many levels, topic groups or shop items on the platform. The only potential bottleneck could be the backend api server but that can be easily fixed through extra deployment of backend docker containers for load balancing.

Similarly, if there was a bottleneck in the frontend when there are hundreds of users using the frontend concurrently, the platform can be scaled by increasing the number of docker containers and using nginx as load balancer to increase capacity. The features have been designed to support this form of scaling which is the industry standard. Each docker container can handle about 80 - 100 concurrent users and multiplying the number of docker containers will scale the platform up.

6.7.4 Feedback from Potential Users

Students

"The gamification platform has been an interesting take on learning modules, it is more exciting and the competitive aspect is really awesome!"

"The platform should limit how many times people can run through a level, that will increase competitiveness and sense of urgency."

Overall, across the students I have explained the platform to, the feedback has been positive, there are small details that they like to see to make it more robust and more competitive.

Admins

"It provides tutors more ways to test knowledge while encouraging students to actively do their best"

"This will definitely help to achieve learning outcomes especially in the online format."

Overall, across the tutors and lecturers I have explained the platform to, the feedback has been positive, they believed that this would add a way of learning and admin customisation that currently does not exist.

6.8 Backend

The backend component will be assessed on based on the following:

- Functional Requirements
- Non-functional Requirements

6.8.1 Functional Requirements

The functional requirements for the backend component have been completed. The priorities of each requirement are classified as equally important, as each category is general.

In total 8/8 functional requirements have been completed for both generalised API and database categories. This completion of the requirements is displayed below.

API

1. Users can retrieve data
2. Users can change data
3. Users can delete data
4. Users can create data

Database

1. Admins can view database entries
2. Admins can edit database entries
3. Admins can create database entries
4. Admins can delete database entries

6.8.2 Non-Functional Requirements

The non-functional requirements for this component have been completed to an adequate level, as 3/3 requirements have been satisfied.

Non-Functional Requirements

1. Usability - The feature must be efficient and effective
2. Performance - The feature must process queries within a reasonable timeframe
3. Learnabilty - The feature must be easy to learn and use

Usability

From the backend each endpoint is effective at completing its task and efficient to use. The backend is efficient as users simply need to enter inputs and execute to receive the response output.

Performance

The backend does somewhat process queries within a reasonable timeframe if the size of the database is small although the backend can be improved to better match this requirement.

Learnability

There is an OpenAPI specification that allows users to easily understand the required inputs and outputs of each endpoint. Additionally, learnability is further supported with sample responses and inputs provided on the OpenAPI specification.

6.8.3 Challenges

The challenges when implementing the backend was setting up the backend with APIs and databases, as a lack of experience made this challenging. For instance, understanding how to create Restful APIs, integration of NodeJS with PostgreSQL and also incorporating the visualisation of the APIs via OpenAPI specification. Additionally, devising a method to locally store the files on the system was also challenging as there was no usage of cloud storage methods. Although there were some challenges to the backend implementation, the backend is arguably completed and adheres to the scope of the Meta LMS.

Chapter 7

Conclusion

7.1 Accounts and Enrollments

The main contributions of the accounts and enrollment feature are the ability for all users to interact with the LMS in a way that is personal to them, so that they can have the best possible learning experience. It also provides tools to staff moderate their courses as efficiently and effectively as possible.

7.1.1 What would you have done differently

The primary feature that I would implement differently would be the method of authentication for users. Currently, when a user is signed in, the back-end sends a JSON Web Token to the front-end to represent that user's session, and this is stored in the browsers local storage. While this is functional, it is not particularly secure as bad actors could easily intercept these payloads, or extract these tokens from the browsers storage quite easily. An alternate authentication method that I would employee instead would be using a standard such as OAuth 2 [Par]. OAuth 2 is the current industry standard protocol for authentication, and it has a much higher focus on client developer simplicity, while also being far more secure than using JSON Web Tokens. Implementing this authentication method would greatly increase the security of the authentication system and thus the quality of the user experience. Choosing the OAuth 2 approach early in the design process would have benefited the LMS as a whole greatly.

Another component of the accounts and enrollments feature that would be approached differently would be inclusion of more account types than just staff and student. Being able to assign account types per course rather than site wide would help allow for complex enrollment scenarios such, as a student who also tutors a course, or a staff member who lectures in one course and administrates another. Adding further depth to the account types by adding more roles and making them on a per-course basis rather

than site wide would also allow for a more enriched permissions system, restricting access to certain parts of the LMS further while broadening access to other parts. For example, perhaps only course administrators should have access to the enrollments dashboard, while all staff from tutors to lecturers to the course administrator should be able to endorse comments on the forum. Considering these further depths of account types from earlier in the design and development process would have deeply enriched the functionality of the LMS.

7.1.2 Future work

Due to the time and resource restricted nature of the Thesis, while the core functionality of the accounts and enrollments system was completed, there are many potential future additions that would greatly increase the user experience with this feature. These features include:

1. Allowing student users to un-enroll themselves from a course, rather than only staff being able to un-enroll students.
2. Allowing staff users to manage student accounts on behalf of the students.
3. Adding deeper enrollment features such as the ability to import/export a CSV list of students into or from a course to speed up the enrollment process.
4. Allowing users to download and delete all information relating to them from the LMS as a privacy tool.
5. Upgrading authentication to OAuth 2.
6. Modifying the current account types system to include more account types (Course administrator, lecturer, tutor).
7. Upgrading the current account type system to allow account types to be on a per-course basis rather than site-wide.

7.2 Topic Tree

The aims of this thesis as discussed in the introduction, to allow course instructors to easily curate and manage content. The topic tree allows instructors to do just this, by providing more structure to content, and by providing a method to create and manage topic groups and topics themselves. Instructors can easily view prerequisites between topics and reuse content by cloning topics from other topic groups and setting specific topics as prerequisites for other topics. All of these features improve the reusability and management of content for instructors.

The main contributions of the topic tree include:

1. A new method of creating and managing courses with the concept of topics and topic groups;
2. Providing further structure to content by organising them by topics;
3. Improving reusability by allowing topics in other topic groups to be set as pre-requisites and by introducing the cloning feature;
4. Providing adaptability to many learning models with files separated into four key sections;
5. A functioning proof of concept topic tree that is well integrated into an overall learning management system.

7.2.1 Challenges

The main challenge of the topic tree included developing the graph interface. This interface uses the D3 library, which allows for greater flexibility in developing data graphs, but with little support or documentation online. This made it quite difficult to develop the topic tree.

Other challenges included working with the rest of the team to integrate the topic tree with the rest of the system, as the rest of the team needed to know how topics and topic groups worked early on to develop their own features. Setting up topics and topic groups in the database was also quite difficult, due to the graph nature of the topic tree, as all this data had to be collated and loaded into the topic tree on first load with minimal impact to performance. Finally, some features listed in future work would involve working heavily with the rest of the team, such as tracking user progress required enrolment to be completely finished to retrieve students' enrolment data, and disciplines involved changing the database structure significantly which many API end-points depended on, and so were not completed by the end of the thesis.

7.2.2 Future Work

Future work for the topic tree includes:

1. Third party integration with YouTube, The Box, Moodle, etc.;
2. Methods to export topic and topic group data to other platforms;

3. Disciplines i.e. grouping topic groups together to improve performance and reduce information overload, especially as different schools or faculties mostly do not have prerequisites from other faculties;
4. Graph performance and smoothness improvements;
5. Student progress shown on the topic tree.

7.3 Course Pages

7.3.1 What would you have done differently

The main things that would have been done differently are:

- Focusing more on the implementation of the course selection page
- Possibly restructuring the way the pages were implemented
- Restructuring some of the specifications for the backend API endpoints

The course selection page was added in Thesis B, it was after realising that a user would need a page to navigate to different courses. This crucial element of the MetaLMS should have been emphasised in the development as it would be a central area of navigation. Other things that could be done would be to improve the performance or allow for a less coupled system and provide more flexibility.

7.3.2 Future Work

Future features that could be implemented are:

- Making the widgets bar more customisable
- Implementing the course outline page
- Adding more features to the course selection page

The current design of the widgets bar has 2 features, the calendar and reminders section. In future implementation, more features could be added and the widgets bar could be made to be more customisable and allow for users to choose what widgets to have on the widgets bar. The course selection page could also include more features and customisability as well. The features in the course selection page are separated into boxes which could possibly allow for a more flexible design.

7.4 Lectures and Tutorials

The contributions of the lectures and tutorial component include:

1. Allowing users to access lectures and tutorial files
2. Users to view lectures or tutorials videos
3. Assisting users in searching for files related to lectures and tutorials

7.4.1 What would you have done differently

There would be some research on how to incorporate YouTube's API to render the lecture videos and playlists within the lectures and tutorials component itself. Additionally, an early discussion should have been made to determine which feature should store relevant files to the course in order to avoid overlapping features.

7.4.2 Future Work

Some future work for the lectures and tutorials components include:

1. Visualise playlist of lecture/tutorial videos on the Meta LMS
2. Allow lecturers to start/end lecture from the lectures/tutorials page

7.5 Assessments

The main contributions of the Assessment feature include:

1. Providing convenient quiz creation, usage and submission
2. Offering useful tools for both staff and students
3. Aiding students in revising and improving by showing explanations for questions

7.5.1 What would you have done differently

What would have been done differently when approaching the Assessments feature would have been to make early designs of the user interface and database schema to include every feature - both core and extra features. This would have saved a lot of time and countless re-designs since any additions or deletions could be easily done at any time. The lack of this has caused a large delay to the plans for the Assessments feature and as a result, leaves the Poll feature as future work and the Quiz feature incomplete.

7.5.2 Future Work

Potential future works for this thesis include:

1. Ability to import existing questions
2. Add more question types
3. Create a statistics page on student attempts
4. More extensive feedback when a student views their submission
5. Integration with other features
6. Improve user interface to be more appealing and easier to use

7.6 Forums

As mentioned in the project approach, the main goal of the forums component is for it to be the central location for conversation and discussion between students and educators for each course. The aim was to implement all the required and desired features into the LMS' forum so that staff would no longer have to turn to a third-party forum services. Aside from the standard features of viewing and adding posts, comments and replies, the additional features built that helped contribute to this goal include:

- Advanced search, filtering and sorting capabilities to assist in finding specific posts;
- Methods for both students and staff to assist in bringing attentions to posts, like upvoting, endorsing and pinning posts; and,
- The ability to copy a direct link to a post to assist in sharing.

7.6.1 What would you have done differently

If this project were to be re-done, the main thing that would've been done differently is conducting more research on different third-party libraries instead of going ahead with the first one found. For React in particular, there are many third-party packages that essentially provide the same service but with different features. Researching would allow for the package with the most desired features to be chosen. Choosing a package with sufficient documentation could also prevent developers from wandering around in the dark when trying to implement it into the project. Overall, choosing the correct package from the start would've prevented a lot of refactoring of the code, therefore saving time in the development process and allowing more features to be built on time.

7.6.2 Future Work

Future work for the forums component includes:

- Ability to attach documents and files to posts;
- Ability to add attachments to replies and comments;
- More share options so that links can be shared directly to other platforms;
- Introduce natural language processing to parse post titles and descriptions as they are being written and generate suggested tags; and,
- Advanced querying that searches existing posts as users type out titles and descriptions and suggests similar posts to reduce duplicates.

7.7 Gamification

Overall, the gamification feature's design, implementation and testing phases have been successful. Taking a role based approach to requirements have prioritised users throughout the implementation phase. Throughout the thesis, there were many design and implementation considerations and tradeoffs that had to be made. The decisions made have always prioritised user experience and the gamification principles outlined in the design phase. The gamification feature has been completed on time according to timeline outlined in Thesis B. There were extra unexpected work that caused time constraints during the integration phase, as a result, a couple of lower priority requirements were removed from scope for this thesis and classified under future work. The feedback received from both students and admins have been extremely positive and has made the implementation of gamification in learning systems worthwhile.

7.7.1 What would you have done differently

- Prioritised the requirements more specifically upfront and have contingency plans due to time constraints
- Provide more buffer time to tackle unexpected requirements that emerge during implementation
- Get user feedback throughout the implementation instead of just at the end to help guide some of the features more clearly
- Started integration work with other features early to ensure that features are more seamless
- Plan time according to priorities better and reuse more components to reduce the implementation effort

7.7.2 Future work

- Customise Rewards like badges and achievements for students
- Coding game type in levels
- Limit number of times a level can be played
- Provide prescriptive analytics across multiple topic groups
- Allow admins to view statistics of their levels
- More item types in the shop

7.8 Backend

The contributions of the backend component include:

1. Establishing the server side of the Meta LMS
2. Assisting users in understanding endpoints
3. Storing data for users and students

7.8.1 What would you have done differently

For the backend, a more refined file hierarchy would have helped save time in refactoring and fixing code. This is because previously all of the code was in one file which eventually became large and difficult to maintain and fix. Additionally, utilising query optimisation techniques and better structuring of tables would allow for better performance in the backend.

7.8.2 Future Work

Some future work for the backend component would include:

1. Scaling the backend to account for more users in the system
2. Optimisation to improve backend performance

7.9 Overall Conclusion

The LMS thesis project aimed to create a meta LMS that corrects the problems found in current LMS's in order to improve the educative experience for instructors and students. That is, the meta LMS will offer reusable content, better course management features, an improved user experience to further promote and enhance the teaching and learning experience for users. As discussed in the report, this LMS will include some quality-of-life features such as a search bar, polished UI, and a feature to reuse course materials in order to achieve the aims outlined in the report.

7.9.1 What would we have done differently - DevOps

The primary issue that arose during development that we would have done differently if given the chance, would be to force more rigorous DevOps practices to ensure the project remains on track during the entire development, and that all developers are on the same page over the course of the development timeline. There are a number of key ways we could have improved this proccess and they are as follows:

- Adopting a specific agile development methodology such as the trunk based approach [Zet]
- Incorporating regular, development focused stand-ups into the development process
- Having a project management tool such as Jira or Trello to ensure a much more streamlined development of the project

Trunk-based Development

A key area where we struggled during development was that due to the turbulent nature of the teams schedules, often development work would take long periods of time. An unfortunate side-effect of this is that this caused some feature branches to exist for a lot longer than expected, and by the time the feature work was completed and the branches

merged back into the master branch, they were extremely out of date with other portions of the app, leading to a disproportionate amount of development time being spent resolving merge conflicts and performing testing within other modules, rather than on developing new features that enhance the LMS. The trunk-based development approach [Zet] enforces guidelines that force feature branches to be smaller, and not exist for too long before being merged back into master. This means that feature work is done in small chunks over one or two days rather than in large components taking multiple weeks making them not only less likely to break the app on merge, but also making them much easier to test. Another core component of the trunk-based approach is the inclusion of pull requests (PRs) into the development cycle. By incorporating PRs in our development, it would allow for multiple team members to independently test and verify each feature, leading to less bugs and errors, as well as compatibility issues down the track. Using a development approach like this would have greatly increased the productivity of the team over the course of development, potentially leading to a more feature rich product.

Regular Stand-ups

Over the course of development, we had weekly meetings to discuss the overall progress of the project. While this was an extremely beneficial tool in ensuring consistent progress was being made, this meeting were not particularly development focused, and rather focused more holistically on the project. The inclusion of specifically development focused stand-ups in the development process may have allowed us to more efficiently allocate development time to certain features if they were falling behind, or identify potential application wide issues and make application wide decisions far early in the process. This may have allowed us to more efficiently manage our time and resources, and perhaps deliver a more feature rich and well integrated product.

Project Management Tools

The final area in which we would have done things differently is the use of a project management tool such as Jira to more clearly visualise both the progress of the project while it was in development, and the work still needing to be completed. Over the course of the project, we split the required work by feature. Instead of this, we could have treated these features as epic stories, and broken these epic stories into smaller user stories representing pieces of development work. These user stories could then very easily be catalogued within a tool such as Jira as tickets. This would have allowed us to then break up the work based on each team members skill sets, assigning tickets to people based on the type of development it involved rather than the feature they existed within. This may have potentially increased the speed of development due to less time needed to be spent up-skilling over the course of the project.

7.9.2 What would we have done differently - Technology

While the technology stack used successfully delivered a highly functional LMS, during development it became apparent that we made a few decisions on technologies to be used

that while they were functional, were not the optimal solution for our requirements. These two main technology decisions we would change if we were to do the project again are as follows:

- Using TypeScript instead of vanilla JavaScript for our primary programming language
- Using MongoDB instead of PostgreSQL for our database

Typescript

While the type-less nature of JavaScript makes it extremely flexible and easy to use, it also leaves it prone to many type errors that using other languages avoid. TypeScript, however, gives JavaScript the benefits of strict typing while still keeping the benefits of its ease of use and power on the web. By using TypeScript in our project, we could have greatly reduced the time spent fixing type related errors, while also being able to more strongly define the transfer of different types of data between the front-end and back-end using types. Although this approach would have required a lot of the team to spend time learning how to use TypeScript, this time would most likely be less than the time spent instead diagnosing and fixing type related errors over the course of the project, making TypeScript a great choice if we were to complete this project again.

MongoDB

Although PostgreSQL is a very powerful database and it worked effectively throughout the course of the project, a lot of development time on the back-end was spent formatting both the data we recieved from the database to be compatible with out JavaScript application, and formatting the data generated by our application to be compatible with the database. MongoDB, on the other hand, uses a JSON-like data format that is highly compatible with JavaScript applications that would have greatly reduced development time for database interactions. Much like TypeScript, the trade-off required for time spent up-skilling to use MongoDB instead of PostgreSQL would most likely have been less than the time it would save not having to spend time converting the data sent to and recieved from our PostgreSQL database.

7.9.3 Additional features

The features listed in this section are are additional features that will be implemented in the future. These features have been compiled into a list as below:

- Assignments
- Attendance and Grading
- Blogs/Wikis/Discussions

- Notifications
- Third Party Integration/Data Migration
- Inbox/Messaging

Bibliography

[Bha20] Apoorva Jayesh Bhagchandani. *Building a Meta Learning Management System*. Bachelor of Engineering in Software Engineering (Honours) Thesis A, School of Computer Science and Engineering, The University of New South Wales, 2020.

[Cana] Canvas. Canvas. https://www.instructure.com/en-au. Accessed: 2021-04-12.

[Canb] Canvas. What is Canvas Commons? https://community.canvaslms.com/t5/Commons/What-is-Canvas-Commons/ta-p/1788. Accessed: 2021-04-25.

[D2L] D2L. LMS For Schools. https://www.d2l.com/en-apac/schools/products/core/. Accessed: 2021-04-22.

[Edma] Edmodo. Edmodo. https://new.edmodo.com. Accessed: 2021-03-17.

[Edmb] Edmodo. Introducing JumpStart Activity Studio on Edmodo. https://go.edmodo.com/introducing-jumpstart-activity-studio-on-edmodo. Accessed: 2021-03-17.

[Ell12] Ryann Ellis. Field Guide to Learning Management. https://web.archive.org/web/20140824102458/http://www.astd.org/~/media/Files/Publications/LMS_fieldguide_20091, 2012. Accessed: 2014-08-24.

[eTh] eThink. What is Moodle? The Comprehensive Moodle Learning Management System (LMS) Guide. https://ethinkeducation.com/what-is-moodle-guide/. Accessed: 2021-03-16.

[Gooa] Google. Lighthouse - Tools for Web Developers - Google Developers. https://developers.google.com/web/tools/lighthouse. Accessed: 2021-04-23.

[Goob] Google. Lighthouse Performance Scoring. https://web.dev/performance-scoring/. Accessed: 2021-04-25.

[Mooa] Moodle. Backup and Restore FAQ. https://docs.moodle.org/310/en/Backup_and_restore_FAQ. Accessed: 2021-03-16.

[Moob] Moodle. Course Backup. https://docs.moodle.org/310/en/Course_backup. Accessed: 2021-03-16.

[Mooc] Moodle. Course Restore. https://docs.moodle.org/310/en/Course_restoreQ. Accessed: 2021-03-16.

[Mood] Moodle. Moodle LMS. https://moodle.com/lms/. Accessed: 2021-03-16.

[NT20] Suphachabhar Nigrodhananda and Ki Fung Kelvin Ting. *Building a Meta Learning Management System*. Bachelor of Engineering in Software Engineering (Honours) Thesis A, School of Computer Science and Engineering, The University of New South Wales, 2020.

[Opea] OpenLearning. About openlearning. https://web.archive.org/web/20121013202446/https://www.openlearning.com/About. Accessed: 2012-10-03.

[Opeb] OpenLearning. Pricing plans for top education business - OpenLearning. https://solutions.openlearning.com/pricing. Accessed: 2021-04-23.

[Opec] OpenLearning. What information can be exported from and where? - Statistics and Exports - OpenLearning Help Community. https://help.openlearning.com/t/365w1v/what-information-can-be-exported-and-from-where. Accessed: 2021-04-23.

[Par] Aaron Parecki. Oauth 2. https://oauth.net/2/. Accessed: 2021-11-20.

[Pro] ProductPlan. Moscow Prioritization. https://www.productplan.com/glossary/moscow-prioritization/. Accessed: 2021-04-30.

[SCO] SCORM.com. Scorm Explained: In Depth Review of the SCORM eLearning Standard. https://scorm.com/scorm-explained. Accessed: 2021-03-20.

[Tiw] Anand Vardhan Tiwary. Why Universities Should Use An LMS. https://elearningindustry.com/why-universities-should-use-lms. Accessed: 2021-04-25.

[Uni] Charles Sturt University. Innovative curriculum - Engineering. https://www.csu.edu.au/engineering/curriculum. Accessed: 2021-04-25.

[UNS] UNSW. Understanding Learning Processes. https://www.teaching.unsw.edu.au/understanding-learning-processes. Accessed: 2021-11-19.

[web] web.dev. Lighthouse accessibility scoring. https://web.dev/accessibility-scoring/. Accessed: 2021-04-26.

[Wik] Wikipedia. Edmodo - Wikipedia. https://en.wikipedia.org/wiki/Edmodo. Accessed: 2021-03-17.

[Zet] Kev Zettler. Trunk-based development. https://www.atlassian.com/continuous-delivery/continuous-integration/trunk-based-development. Accessed: 2021-11-23.

[Zho20] (Tammy) Xue Qing Zhong. *Building a Meta Learning Management System*. Bachelor of Engineering in Software Engineering (Honours) Thesis A, School of Computer Science and Engineering, The University of New South Wales, 2020.