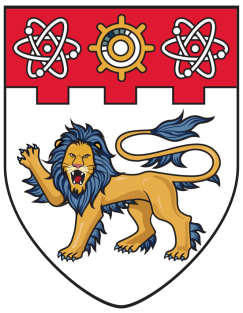

NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

NANYANG TECHNOLOGICAL UNIVERSITY

SC/CE/CZ4052 CLOUD COMPUTING

PROJECT REPORT

(Topic 4: Cloud Computation by Crowdsourcing)

Group Members:

Tan Jun Wei (U2021891K)

Lau Xin Wei (U2020986E)

Tan Ye Quan (U2021135G)

Table of Contents

1. Introduction

The intersection of technology and education continues to produce innovative solutions to age-old problems, one of which is choosing the right academic modules. With the rapid evolution of cloud-based platforms and the principle of crowdsourcing, there is an opportunity to enhance how students select their courses. Our project focuses on developing an online school modules review system that utilises these technologies to gather and disseminate student feedback on academic courses.

2. Problem Statement

Selecting academic modules without sufficient information can lead to poor academic experiences and outcomes for students. Currently, there is a lack of comprehensive, accessible, and organised information about course structures, teaching methods, and workload, which are crucial for making informed decisions. Our project addresses this gap by creating a platform where students can share detailed reviews of their completed modules. This crowdsourced information will assist future students in making better-informed decisions about which modules to take, based on previous students' real-life experiences and insights.

3. Solution Design

The development of our online school modules review system required careful consideration of both frontend and backend components, along with the selection of appropriate cloud services for deployment. Below, we detail the architecture and technologies utilised for each aspect of the application.

3.1. System Architecture

The NTUMods system leverages a combination of cloud services from Amazon Web Services (AWS) and Microsoft Azure to host and manage the application's frontend and backend components, along with DevOps tools that aid in infrastructure management

and continuous integration and delivery. Figure 1 shows the system architecture for the NTUMods.

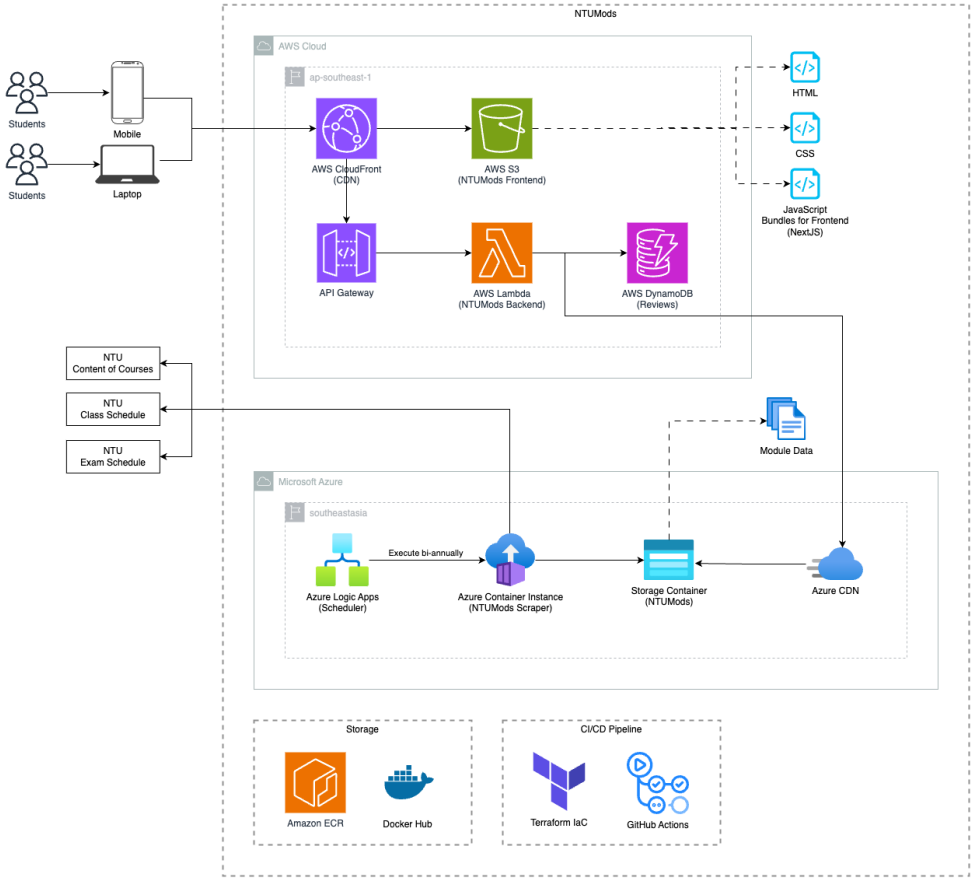

Figure 1: NTUMods System Architecture

Amazon Web Services (AWS) Integration

The NTUMods system utilises various AWS services to host and manage the application's frontend and backend components:

1. **Elastic Container Registry (ECR):** A fully managed Docker container registry, to store and manage my application's container images securely. The backend images hosted in ECR are utilised in conjunction with AWS Lambda for deployment and execution of the serverless backend functions.

2. **AWS CloudFront:** CloudFront serves as the content delivery network (CDN) to distribute the frontend assets with low latency and high transfer speeds. It ensures that students using the NTUMods system experience quick load times

for the web application, regardless of their geographical location.

3. **AWS S3:** Amazon Simple Storage Service (S3) is used to store and serve static web resources for the NTUMods frontend. This includes HTML, CSS, and JavaScript files that are bundled using Next.js, a React framework.

4. **AWS API Gateway:** This service acts as the entry point for the backend, managing and directing incoming HTTP requests to the appropriate Lambda functions and enabling a secure way to connect the client side with the serverless backend.

5. **AWS DynamoDB:** DynamoDB is a NoSQL database service that stores and retrieves the data needed by the application. It offers fast and predictable performance with seamless scalability.

Microsoft Azure Integration

In addition to AWS, the NTUMods system also incorporates services from Microsoft Azure:

1. **Azure Logic Apps:** A cloud service that helps you schedule, automate, and create workflows to integrate services. This service is used to orchestrate and automate the processes associated with retrieving and updating the module data from NTU resources.

2. **Azure Container Instance:** A container on demand service that allows users to run containers directly. The container instance runs the NTUMods Scraper script, a critical component responsible for extracting NTU's course content, class schedules, and exam schedules. This service is set to execute biannually, ensuring the module information is up-to-date.

3. **Azure Storage Container:** Module data extracted by the NTUMods Scraper is stored in Azure Storage Containers, providing a secure and scalable solution for housing the data before distribution.

4. **Azure CDN:** Integrated with Azure Storage, Azure CDN caches the module data close to users, reducing latency and load times, thereby accelerating content delivery and enhancing the user experience.

DevOps and Infrastructure Management

The NTUMods system also incorporates the following suite of tools to automate deployment and ensure consistency across environments:

1. **Terraform Infrastructure-as-Code (IaC):** The NTUMods system's cloud infrastructure is defined and provisioned using Terraform, enabling consistent and automated deployments across both AWS and Azure platforms.

2. **Docker Hub:** Docker Hub serves as the central image repository for the docker images for NTUMods Scraper. These images are then pulled by Azure Container Instances for the NTUMods Scraper, assuring version control and management of container deployments.

3. **GitHub Actions:** Integrated into the NTUMods development pipeline, GitHub Actions automates the CI/CD processes. Each code push or pull request initiates workflows to build and deploy the application, promoting a streamlined, efficient development lifecycle.

3.2. Module Scraper

The module scraper is developed using the Go programming language which offers numerous advantages for web scraping tasks. Go provides robust features for handling concurrent operations which is essential for efficient data extraction across multiple sources. This capability is particularly useful when dealing with multiple HTTP requests as it significantly reduces the time required for data collection by running these processes in parallel.

The operational workflow of the scraper begins with extracting a list of modules offered for the current academic year and semester. Following this initial step, the scraper makes HTTP requests to the NTU portals (specifically, the Content of Courses, Class Schedule, and Exam Schedule) to gather necessary information. Utilising the Go's concurrency system, the scraper concurrently processes data from these three distinct websites. This approach not only optimises the speed of data retrieval but also ensures that the data collected is up-to-date and accurate, aligning with the latest academic offerings.

Once the data is collected, it is automatically uploaded to a Microsoft Azure Storage Container for secure and scalable storage. In addition to the detailed data, a moduleList.json file is created and uploaded. This file serves as a lightweight summary of all courses offered in the semester, providing an easily accessible reference for further analysis or reporting needs.

3.3. Frontend Development

The frontend architecture of this project is powered by Next.js (an open-source web development framework) to deliver a responsive and user-friendly interface. The frontend facilitates efficient client-side navigation, allowing users to experience smooth transitions and quick load times across different sections of the application. To enhance code reliability and maintainability, TypeScript is utilised for static typing, improving both code readability and robustness.

The frontend also utilised Tailwind CSS and Ant Design component library to streamline the user interface development. Tailwind CSS enables rapid-styling with its utility-first approach, while Ant Design provides a suite of ready-to-use components that enhance the overall design and usability of the application.

The user interface is designed to facilitate seamless interaction and easy navigation, ensuring that users accessing the application from various devices and screen sizes have a uniformed experience. Each module page within the application is designed to present detailed information in a structured and easily digestible format, and includes interactive elements such as module index selection, timetable integration, and review functionalities enhance user engagement and utility.

Furthermore, the frontend features a timetable planner component that enables users to visually organise and manage their module schedules effectively. This planner not only supports the addition and modification of modules in a user's timetable but also integrates seamlessly with the detailed module pages, providing a cohesive and interactive user experience. This comprehensive integration of front-end technologies and thoughtful UI design collectively optimise the functionality and accessibility of the application, placing it an essential tool for users managing their academic commitments.

3.4. Backend Development

The backend is built on top of a robust set of server-side technologies to ensure efficient and reliable performance. Node.js serves as the runtime environment, supporting asynchronous events and scalable network applications. Express.js is used to construct the web server, providing a streamlined framework for routing and middleware functionalities that simplify the development of server-side logic.

For database management, Amazon's DynamoDB is used due to its flexibility and scalability, critical attributes for handling varying loads and data types. As a NoSQL database, DynamoDB supports the efficient storage and retrieval of structured module data, user profiles, and records of reviews and replies; to ensure fast access and high availability. Moreover, DynamoDB offers built-in features to maintain strong consistency, guaranteeing that all users view the most current data without delays. This is vital for user interactions and data integrity across the application.

Additionally, the RESTful APIs are developed to facilitate seamless client-server communication. These APIs allow the frontend components to interact effectively with the backend, enabling features such as retrieving module information, managing user sessions, and storing review data. This integration ensures that the application remains responsive and capable of supporting a high level of user interaction and data management, catering to the needs of a dynamic user base.

4. System Features

4.1. Module Search and Overview

Students can search for specific modules and access detailed information including an overview of the module's content, objectives, and learning outcomes. The module search also provides in-depth details of the module, such as the exam schedule, prerequisites for the module and many more. Figure 2 shows an example of the module detail page.

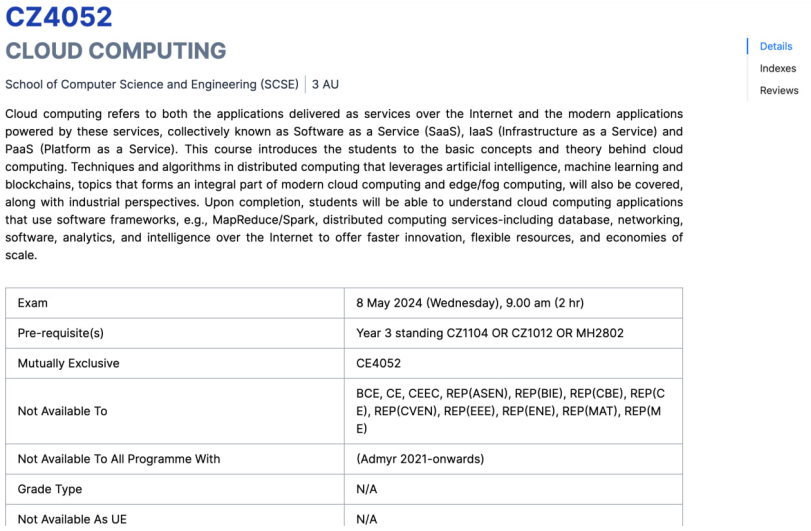

Figure 2: Module detail page for CZ4052

4.2. Timetable and Index Information

The application provides information on the various class timings for each index available for a module, allowing students to select indexes that best fit their schedules. Figure 3 shows an example of the interface for a module offering multiple index.

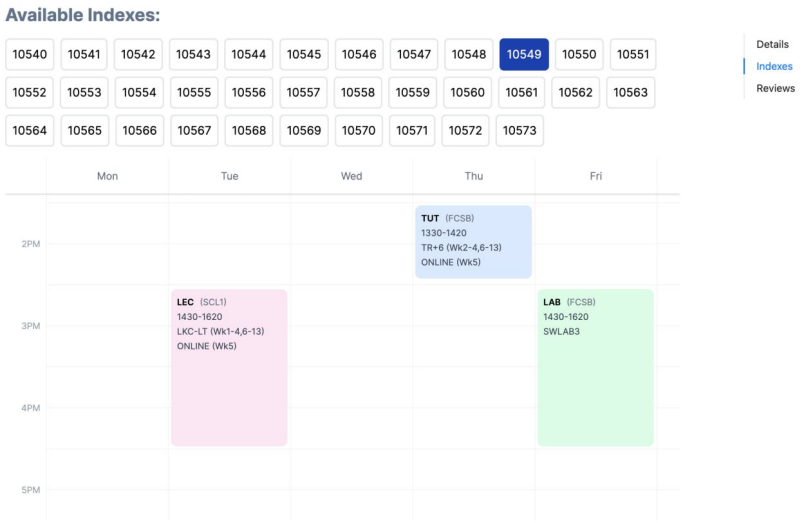

Figure 3: Index selection and Viewing on Timetable

4.3. Comment Section

A dedicated area for student reviews provides insights into the modules from those who have previously enrolled, offering a platform for honest feedback and shared

experiences. This form of crowdsourcing allows students to make more informed decisions when planning for their modules. Figure 4 shows an example of the reviews and discussion section.

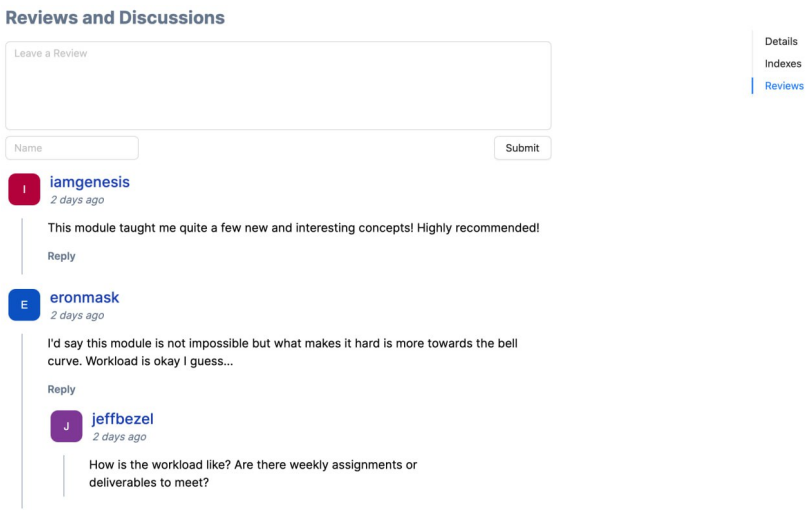

Figure 4: Reviews and Discussions section

4.4. Timetable Planner

Students can use the application to add modules to a personalised timetable section, aiding in efficient scheduling and conflict resolution. Figure 5 shows the timetable planner page.

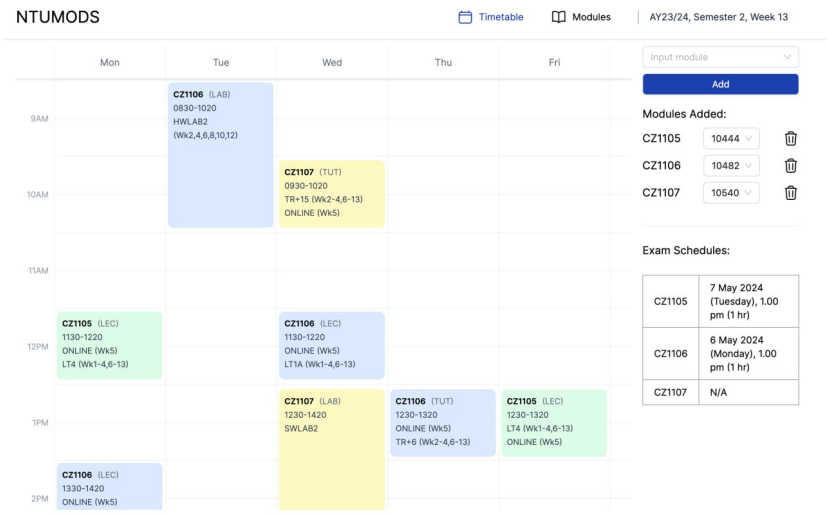

Figure 5: Timetable Planner

When there is a clash in timings between different modules, the student will also be notified in the user interface. This ensures that the students can make more informed decisions when planning for their modules. Figure 6 illustrates an example when user selects an index that clashes with another module.

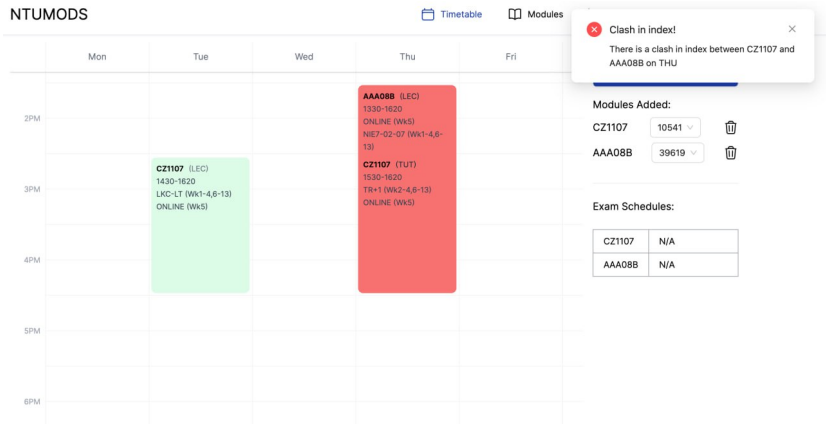

Figure 6: Error notification on clashing index

These features collectively aim to create a user-friendly, informative, and interactive experience, empowering students to make well-informed decisions regarding their academic paths.

5. Future Works

5.1. Optimising Course Update Schedules

One of the primary limitations faced by the NTUMods system is the uncertainty regarding the schedule of content updates by NTU. The content for courses is not consistently available, and the timing of when new content is added remains unpredictable. For instance, the current ”Acad Yr 2023 Special Term” does not have a single module offered for that semester. This variability presents challenges in determining the optimal moments to run the automated scraping scheduler. Currently, the NTUMods system does not support the Special Term due to the lack of available module information during these periods. This gap in the service could lead to incomplete data provisioning for users seeking information about the Special Term offerings.

To address this limitation, future work on the NTUMods system could involve the

implementation of a more dynamic scheduler. The scheduler should be capable of detecting changes or updates to the NTU Content of Courses page in real-time or on a more frequent basis. This enhancement would ensure that the scraper is activated promptly when new data becomes available, thereby maintaining the system's relevance and accuracy throughout the academic year, including the Special Term.

Moreover, although the current NTUMods Scraper does not support the Special Term, it is architected with extensibility in mind. The existing codebase is structured such that it can be readily updated to include Special Term content once it is available for scraping.

5.2. Crowdsourced Campus Venues

Another promising direction for the NTUMods system is the integration of a venue and availability feature for campus facilities. Leveraging the data obtained from scraping module details, including class schedules, the system could provide a dynamic timetable indicating when lecture theatres, tutorial rooms, and seminar rooms are not in use. This would offer students a valuable resource for identifying alternative study spaces during periods of underutilisation.

To further enhance the usability of this feature, a crowdsourcing component could be introduced. Students could contribute to the system by mapping the precise location of these rooms using GPS coordinates, which would facilitate easier navigation through apps like Google Maps. Additionally, integrating with NTU's own map application could provide more accurate internal navigation within the campus.

A social feature could also be explored, where students recommend and share locations they've found conducive for studying. This peer-sourced information could be moderated and added to the system, providing a diverse array of study spots based on actual student experiences and preferences. This approach not only improves the utility of the NTUMods system but also fosters a collaborative community atmosphere among students seeking optimal study environments.

6. Conclusion

In conclusion, our crowdsource-as-a-service platform offers a comprehensive solution for NTU students navigating their module selections. By providing access to detailed module information and invaluable insights from seniors who have firsthand experience, we empower students to make informed decisions tailored to their academic goals and preferences. Our platform not only streamlines the module selection process but also fosters a sense of community and collaboration among students. Moving forward, we remain committed to enhancing the user experience and expanding our service to further support students in their academic journey. With our crowdsource-as-a-service, students can confidently embark on their educational path equipped with the knowledge and resources they need for success.

6.1. Relevant Links

- NTUMods Demo URL: `http://20.247.173.107`
- GitHub Repository: `https://github.com/yequan99/NTUMods`