Personalized Quiz Maker: Novel feature for Alby learning management system

Lys Kang , '24

Personalized Quiz Maker: Novel feature for Alby learning management system

E90 Project Report
Spring 2024

Lys Kang
Supervisor: Allan Moser

Abstract

This report details the development of a personalized quiz feature for the Alby learning management platform, designed to enhance student learning by generating quizzes tailored to individual performance data. The project involved creating an algorithm to analyze dynamic student data and identify areas for improvement and then using OpenAI's GPT-4 to generate the multiple choice quizzes with elaborative feedback. Developed as a serverless application with AWS tools, the feature operates independently from the main platform. Despite the absence of applicable professional standards and constraints like limited data availability, the project met its functional requirements, demonstrating the potential to improve educational outcomes through customized, adaptive quizzes. Future work will aim to optimize quiz generation times and explore the development of a custom AI model.

Table of Contents

I. Introduction

A. Project Overview

1. Purpose and Objectives

This report discusses the development and implementation of a new feature that generates personalized multiple-choice quizzes for a learning management platform named Alby. This software engineering project aims to provide an individualized learning experience for students by using their specific performance data to create subject-specific quizzes. In the scope of the project, an algorithm was developed to analyze the student data collected and identify areas that require revision. Generative artificial intelligence is then employed to generate the quizzes, which adapt the contents dynamically based on the student's learning trajectory. The objective of this project is to continuously adapt and offer a tailored learning experience that reflects each student's individual learning and evolving needs.

2. Constraints, Requirements, and Standards

For this software engineering project, several constraints had to be taken into consideration that included the need to develop a serverless application that leverages AWS tools and services, while not being integrated into the main app and limited data available. This is further elaborated in section 1C. The requirements developed for this project included fully developing the functional feature for the platform, which was successfully achieved. No professional standards or codes governed the design of this project, as it is a software feature being developed for a startup, and no relevant standards were found or applicable in this case.

3. Background Information

This project aimed to develop software for a short multiple-choice quiz that would be effortless to take daily to embed efficient study habits early in the student's educational journey. By actively and repeatedly quizzing the students in a low-pressure manner, our approach aims to alleviate exam-related stress and panic and promote active learning, as opposed to passive methods, which promote deeper understanding and more closely resemble real test conditions. This approach not only enhances knowledge retention but also boosts confidence and exam readiness, ensuring students are better prepared and less anxious when facing exams.

Power Tools & Pedagogical Motivation

The development of personalized educational tools, particularly in the ed-tech sector, has garnered significant attention in recent years. This trend includes advancements in adaptive systems, such as personalized quizzes, which tailor content and pacing to individual learners' needs.

The project incorporates elements of retrieval practice and spaced repetition, techniques proven to enhance learning. Retrieval practice is a cognitive strategy that involves recalling information from memory, thereby strengthening its storage and retrieval in the long term. This technique has been shown to improve academic performance across various subjects and disciplines (Roediger & Karpicke, 2006; Karpicke & Blunt, 2011). Spaced repetition, on the other hand, is a learning strategy that involves reviewing material at increasing intervals over time. This approach has been demonstrated to improve long-term memory retention and reduce the amount of time spent studying (Ebbinghaus, 1885; Bahrick & Phelps, 1975).

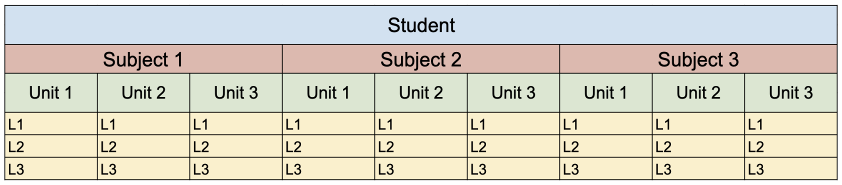
By integrating these pedagogical strategies within a generative AI framework, the software offers a bespoke educational experience that dynamically adjusts content based on a student's demonstrated learning trajectory. By focusing on individual student weaknesses, the project aims to streamline the learning process, making it more efficient and effective. This individualized approach to learning not only caters to the unique learning needs of each student but also optimizes their study time, leading to improved learning outcomes.

Alby, Active Learning Buddy

Alby Learning Management Platform is an educational technology startup founded by my sister and me during our summer break. The primary objective in building the platform is to create an all-in-one learning management system that integrates cutting-edge research in study methods with a personalized, engaging, and adaptive learning experience for both teachers and students. The driving motivation behind the development of this company is to help student's study and learn more efficiently. Students, particularly those in middle and high school, often grapple with the overwhelming amount of material they need to study, leading to panic and stress. Therefore, this project's idea was inspired to address the challenges in education by providing tailored and targeted learning solutions, specifically for middle and high school students.

The platform operates by allowing educators to upload their class slides along with a concept list, which outlines the main ideas covered in each lesson. The Alby system then generates a lesson summary, typically in PDF format, that provides an overview of the lesson's content. Additionally, the platform creates a multiple-choice quiz tailored specifically to the material from that particular lesson. Upon completion of the quiz, the system automatically grades the student's performance and offers detailed feedback on each question, highlighting the correct answers and explaining why certain choices are incorrect. The primary motivation underlying the design and implementation of the platform's features are how they are short and ease-of-use, therefore allowing learners to cultivate effective study strategies, studying little everyday and consequently retaining more information, rather than reliance on last-minute cramming.

Alby was built on the premise that the classes the students were structured in the following manner illustrated in the diagram bla. Alby's educational model is based on the structured class system, illustrated in the diagram below. The diagram depicts the breakdown of each subject (such as Chemistry, Biology, and Math) into smaller units, each of which is comprised several lessons delivered by the teacher on a daily basis.



Student								
Subject 1			Subject 2			Subject 3		
Unit 1	Unit 2	Unit 3	Unit 1	Unit 2	Unit 3	Unit 1	Unit 2	Unit 3
L1	L1	L1	L1	L1	L1	L1	L1	L1
L2	L2	L2	L2	L2	L2	L2	L2	L2
L3	L3	L3	L3	L3	L3	L3	L3	L3

Diagram bla: Student Class Hierarchy

The primary focus of the development of this personalized quiz is for it to be a new feature for the Alby platform: a personalized quiz that incorporates both current and past material from various lessons within a subject. This approach aims to enhance students' learning experiences by adapting quiz content based on their demonstrated weaknesses in previous lessons.

B. Methodology

This section outlines the methodology employed in the development of a software engineering project aimed at generating personalized multiple-choice quizzes. The approach is structured to prioritize the creation of a functional yet simple initial version of the system, known as the Minimum Viable Product (MVP). This methodology integrates principles from Agile and Lean development strategies, tailored to facilitate iterative design and continuous improvement based on user feedback.[1] [2]

Iterative Development Approach

- Incremental Development Cycles: The project was segmented into small, manageable tasks, allowing for rapid development, testing, and integration within short cycles. With this approach it allowed me to be able to adapt quickly and fix any issues that arose in the big picture of the product.
- Build-Measure-Learn Feedback Loop: Each iteration is designed to be more nuanced and an expanded version of the system's previous capabilities based on the bigger systems constraints (Alby as a learning management platform) and feedback from students.

Application of Supporting Methodological Frameworks

- Agile Development Practices: The project adopts Agile inspired methodologies, particularly the use of iterative approach to development, which was suitable for small projects like this with evolving requirements.
- Lean Development Principles: In this project, it also adopted lean inspired strategies which aims to eliminate non-essential features for the initial steps and

[1] Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Publishing Group. [The Lean Startup](https://theleanstartup.com/).
[2] Beck, K., Beedle, M., et al. (2001). *Manifesto for Agile Software Development*. [Agile Manifesto](https://agilemanifesto.org/).

processes, thereby focusing on getting full system development and satisfying basic requirements.

C. Constraints and Assumptions

This section elaborates on the constraints and specifications of the project, more specifically into the software specifications and data constraints. Additionally, this section will also give an overview of the utilized platforms and their roles within the project's infrastructure.

<u>Software Specifications</u>

A primary constraint for this project is the existing architecture of Alby's core business logic, which is constructed on and hosted via Amazon Web Services (AWS). AWS is a subsidiary of Amazon that offers a suite of on-demand cloud computing platforms and services. Given that the existing system is serverless, the integration of new features necessitates adherence to a similar serverless framework.

Key AWS services utilized in this project include:

- AWS Lambda: This service enables the execution of code without the need to manage servers, facilitating the deployment of serverless applications. Lambda functions serve as the backbone of the application, handling core computational tasks.
- AWS Step Functions: These allow for the orchestration of complex workflows involving multiple Lambda functions. Step Functions provide a structured way to manage the sequence of actions and data flow between functions, enabling the creation of a cohesive pipeline.
- Amazon S3 (Simple Storage Service): This scalable object storage service is used for storing and retrieving large amounts of data, including documents and media files. It supports the application's data handling needs by providing a reliable and accessible data storage solution.

The one service not directly involved in this project is AWS EC2 (Elastic Compute Cloud), which provides resizable computing capacity in the cloud. Although EC2 hosts the main components of Alby, the existing infrastructure, this project focuses on extending functionality rather than altering the core computational environment.

Platform Descriptions and Utilization

The platforms chosen for this project are primarily AWS services due to their robust scalability, reliability, and integration capabilities, which are critical for supporting a serverless application architecture. The interconnected use of Lambda, Step Functions, and S3 forms a flexible and powerful framework that aligns with the serverless design principle, reducing the overhead associated with traditional server management and allowing for a focus on feature development and user experience enhancement. This setup ensures that the solution is both scalable and maintainable.

Data Constraints

The project is constrained by several key data limitations, largely due to the platform's newness and small scale:

- Limited Data Collection: The platform collects only the most recent score from lesson-specific multiple-choice quizzes, without storing historical scores. This limitation restricts the ability to monitor a student's progress over time.
- Concept Data: Data collected includes a list of concepts associated with each lesson, provided by educators. While crucial for aligning quizzes with lesson content, this data is limited to what is initially uploaded with the lesson materials.
- Absence of a Question Bank: There is no centralized repository for quiz questions, preventing the reuse or systematic analysis of questions for future enhancements. This absence hinders the ability to refine and personalize quizzes based on past interactions.

Database Technologies:

- MongoDB: Utilized for storing quiz scores, MongoDB's flexibility and scalability are beneficial for handling diverse data types.

- DynamoDB: Employed for storing lists of concepts, DynamoDB offers fast performance and scalability within the AWS ecosystem, though it limits the system's database technology options.

II. Project Development

A. Development Timeline:

The following section will delve into the planning stage of this project, detailing the approaches employed to develop the project. This section is broken down into 5 sections: Design, Development, Local Testing, Cloud Implementation, and Optimization and Integration.

Design Phase:

This segment describes how the initial process started in the development of the project. Below describes the general approach taken to brainstorming the pipeline of the project. This part of the project took approximately 1-2 days to complete.

- Initial Concept and Planning:
    - The project began with conceptualizing the key features through brainstorming sessions, resulting in the ideation of the overall structure and functionality of the software.
- Flowchart Development:
    - Based on the initial data analysis and educational structure assumptions, a flowchart was drafted to visualize the process flow of the quiz application. This visual representation served as a blueprint for the subsequent development, outlining the flow and interactions within the quiz application.

Development Phase:

In this section, the approach and steps taken when developing the software is described and why said choices were made. The three points below were being built concurrently as need be. This portion of the project took approximately 2-3 weeks to complete.

- Full Pipeline Approach:
    - The software was built using a holistic approach, where the entire application was developed in full iterations rather than modular block construction. This strategy allowed for continuous integration and testing of features, ensuring consistency and functionality throughout the development process.
- Local Implementation:
    - Functions were coded as Python scripts designed to operate locally. This script manipulated CSV files that simulated data akin to what would be stored in a production environment.
- Synthetic Data Simulation: To mimic real-world data interactions
    - Lesson Scores Table: This table was structured to reflect data stored in a MongoDB database, aiming to simulate how lesson scores might be stored and retrieved.
    - Results CSV: Created to replicate a DynamoDB format, ensuring that the data structure remained consistent with potential cloud implementations.
    - Review History Tracking: An additional table was designed to track user interaction history, a critical component for enhancing the educational adaptability of the platform.

<u>Local Testing:</u>

This section outlines the process of cleaning up the python scripts written to first make sure they are working as they should and secondly modify the scripts to have a structure that would be closer to the ones for lambda. This portion of the project took 1 week to complete.

- Local Testing and Iterations:
    - The developed pipeline was tested in the local setup to ensure its effectiveness and reliability. Each test led to refinements, enhancing the robustness of the data handling and processing capabilities.
- Preparation for Cloud Migration:

    - With the goal to deploy the application on the cloud using AWS Lambda functions, the existing Python scripts were adapted to align with cloud functionalities.

<u>Cloud Implementation:</u>

Below the process of transitioning the local python scripts to the cloud environment is outlined. Although not addressed to a large amount of detail, there were various other smaller issues and alternatives to the development of the project. However, they are not addressed here since in the bigger scale of the project they weren't as significant. This implementation portion of the project took 1 month to complete.

- Lambda Function Development:
    - Transitioning to the cloud began with converting the locally tested Python functions into AWS Lambda functions. This step required minimal adjustments due to the foresighted design choices made during the local development phase.
- Exploration of Database Options:
    - There was a period of exploration into using relational databases was undertaken to better align with the structured nature of educational data. However, practical constraints and the project's scope led to the continuation with the established CSV and S3 bucket approach.
- Lambda Testing and Integration:
    - Each Lambda function was tested independently to ensure they performed as expected within the cloud infrastructure. These tests were documented with screenshots included in the project report.

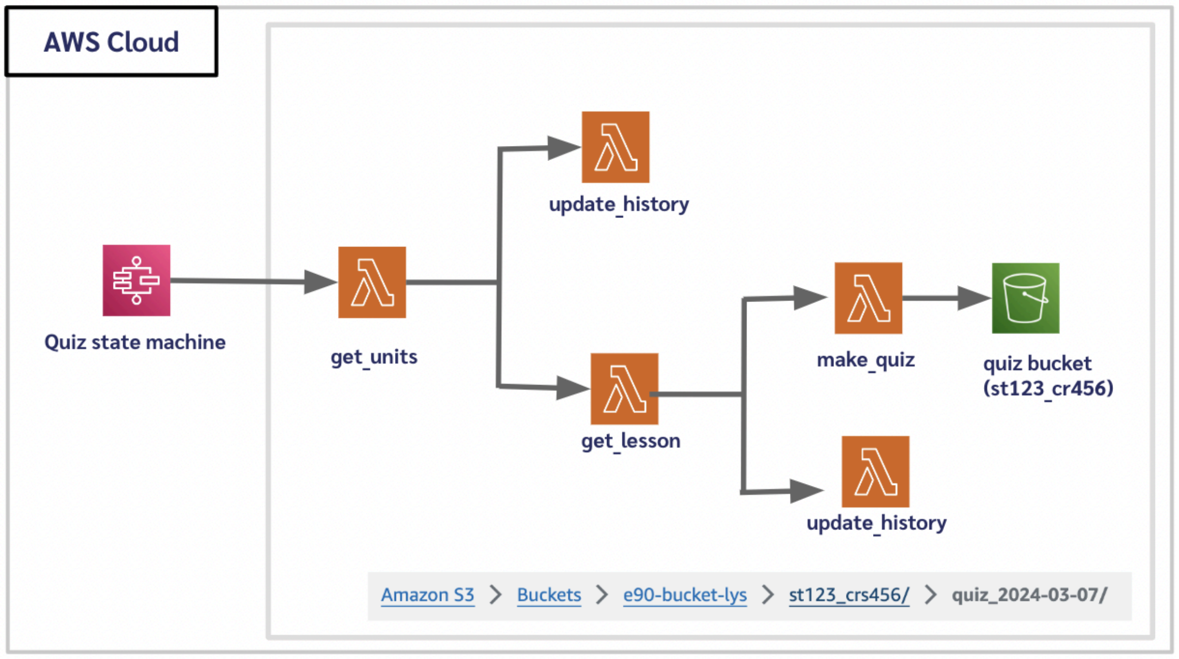
<u>Optimization and Integration:</u>

This final portion of the project development was more about tweaking and modifying the quiz generation process to make it more robust and dynamic. It was also in this part of the development process that feedback was requested from potential users. This part of the project took 2 weeks to complete.

- Enhancing Quiz Logic:
    - The quiz generation logic was refined to dynamically update content based on students' most recent interactions with lessons, ensuring that the quizzes remained relevant and engaging.
- Frontend Compatibility and Final Testing:
    - The final quizzes were tested against the frontend interface to ensure seamless integration. Issues related to automatic updates were identified, but deemed outside the current project scope due to the required extensive backend modifications.

B. Pipeline built on AWS

Below is a diagram illustrating the final pipeline constructed using AWS. The diagram depicts the relationships and the order between functions. It's important to note that the outputs of one function serve as inputs to the next, as indicated by the directional arrows.


AWS Cloud
update_history
Quiz state machine
get_units
make_quiz
quiz bucket (st123_cr456)
get_lesson
update_history
Amazon S3 > Buckets > e90-bucket-lys > st123_crs456/ > quiz_2024-03-07/

Note: The bottom gray bar represents the directory structure of the developed software within the designated S3 bucket.

1. Descriptions of functions:

In the following section, the functions depicted in the above diagram are detailed, with an outline of their specific functionalities. For each function, the inputs are specified, and the operations they perform are described, such as the algorithms executed. Additionally, examples of the CSV tables that were accessed and modified during the process are included.

<u>Functions:</u>

Key:

- Student ID: stID (in diagram __ above would be 123)
- Course ID: crsID (in diagram __ above would be 456)

Get_units:

Inputs:

- stID
- crsID

Definition:

1. Fetches a CSV file from AWS S3 based on a path determined by student ID and course ID: bucketstID_crsID_reviewhistory.csv
2. Reads the CSV content and parses rows into a list of dictionaries, focusing on 'unit_id', 'unit_name', and 'review_date' fields.
3. Identifies the oldest unit reviewed and most recently reviewed unit.
4. returns a list of dictionaries

Ex. Return output

```
[
 {
  "unit_id": "u103",
  "unit_name": "Unit 3: Stoichiometry and Chemical Reactions"
 },
 {
  "unit_id": "u105",
  "unit_name": "Unit 5: Acids, Bases, and Salts"
 }
]
```

Update_history:

Inputs:

- stID

- crsID
- units (list of dictionaries)

Definition:

1. Fetches a CSV file from AWS S3 based on a path determined by student ID and course ID: reviewhistory.csv
2. Iterates through the units list and updates the units 'review_date' field to the current date
3. Writes the updated records and uploads the updated reviewhistory.csv data back to the original S3 location, overwriting the old file.

Sample reviewhistory.csv

unit_id	unit_name	review_date
u101	Unit 1: Atomic Structure and the Periodic Table	2024-04-08
u102	Unit 2: Chemical Bonding and Molecular Geometry	2024-04-08
u103	Unit 3: Stoichiometry and Chemical Reactions	2024-04-05
u104	Unit 4: Solutions and Solubility	2024-04-16
u105	Unit 5: Acids, Bases, and Salts	2024-04-17

Get_lessons:

Inputs:

- stID
- crsID
- units (list of dictionaries)

Definition:

1. Fetches a CSV files from AWS S3: results.csv and newlessons.csv
2. For each unit in the input,
    a. Searches newlessons.csv and finds the lesson with the lowest quiz score & prioritizes lessons that haven't been reviewed in the last 3 days
3. Searches results.csv to find records matching the worst-performing lesson's ID and extracts a list of associated concepts.
4. returns a comma-separated string of all the concepts identified from worst-performing lessons across the provided units.

Ex. Return Output

```
{
  "student_id": "st123",
  "course_id": "crs456",
  "lessons": [
    "l201",
    "l502"
  ],
  "conceptlist": "Definition and Types of Ionic Bonds, Characteristics of Ionic
Compounds, Definition and Types of Covalent Bonds, Polarity of Molecules,
Electronegativity Difference and Bond Type, Arrhenius Theory of Acids and Bases,
Brønsted-Lowry Theory of Acids and Bases, Lewis Theory of Acids and Bases,
Comparison of Acid-Base Theories, Applications of Acid-Base Theories"
}
```

Sample results.csv row

LessonId	APILessonId	LessonConceptList	LessonContentFilepath	LessonContentFolder	LessonName	UnitId
l101	new_id_1	[{"S":"Introduction to Atomic Theory"},{"S":"Historical Models of the Atom"} ...	path/to/atomic_content	path/to/atomic_folder	Atomic theory and structure	u101

Sample newlessons.csv

unit_id	lesson_id	quiz_score	last_reviewed
u103	l301	85	2024-04-12
u103	l302	75	2024-04-16
u103	l303	70	2024-04-12
u104	l401	92	2024-04-12
u104	l402	89	2024-04-19
u105	l501	94	2024-04-22
u105	l502	91	2024-04-24

Update_review_lesson:

Inputs:
- stID
- crsID
- lessons_to_update (list of lesson IDs to update)

Definition:
1. Reads the newlessons.csv file from the designated S3 bucket using the student and course IDs to construct the file path.
2. Iterates through each record (row) in the newlessons.csv and searches for lesson_ids in the lessons_to_update list
3. When it finds the desired lesson_ids, it updates the last_reviewed field with the current date.
4. Writes the updated records and uploads the updated newlessons.csv data back to the original S3 location, overwriting the old file.

Make_quiz:

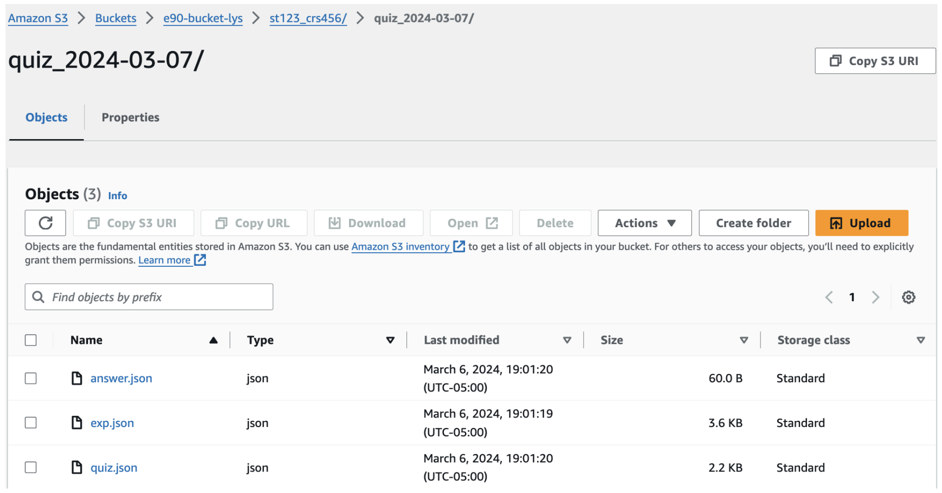
Inputs:

- stID
- crsID
- conceptlist (list of concepts to be tested)
- OpenAI API key

Definition:

1. Retrieves data necessary for quiz generation: student ID, course ID, concept list, and OpenAI API key.
2. Cleans up potential errors by double-checking the quiz generated by OpenAI, specifically looking for and fixing any incomplete formatting, such as missing brackets.
3. Generates a quiz with OpenAI by providing the concepts and instructions to OpenAI's language model, receiving a multiple-choice quiz in JSON format.
4. Formats the quiz output into three versions for different purposes:
    - exp.json: Includes questions with explanations for each answer, useful for studying.
    - answer.json: Contains only the correct answers, designed for easy grading.
    - quiz.json: Comprises only the questions and options, intended for distribution to students.
5. Uploads the three quiz files (exp.json, answer.json, quiz.json) to an AWS S3 bucket, organizing them into a folder named using the student ID, course ID, and date

Sample output file storage


Amazon S3 > Buckets > e90-bucket-lys > st123_crs456/ > quiz_2024-03-07/

quiz_2024-03-07/

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
answer.json	json	March 6, 2024, 19:01:20 (UTC-05:00)	60.0 B	Standard
exp.json	json	March 6, 2024, 19:01:19 (UTC-05:00)	3.6 KB	Standard
quiz.json	json	March 6, 2024, 19:01:20 (UTC-05:00)	2.2 KB	Standard

C. Evaluation

In this section of the report, I assess the extent that the software developed was able to accomplish its objectives. The section below also discusses the feedback received and potential areas were identified for future improvements.

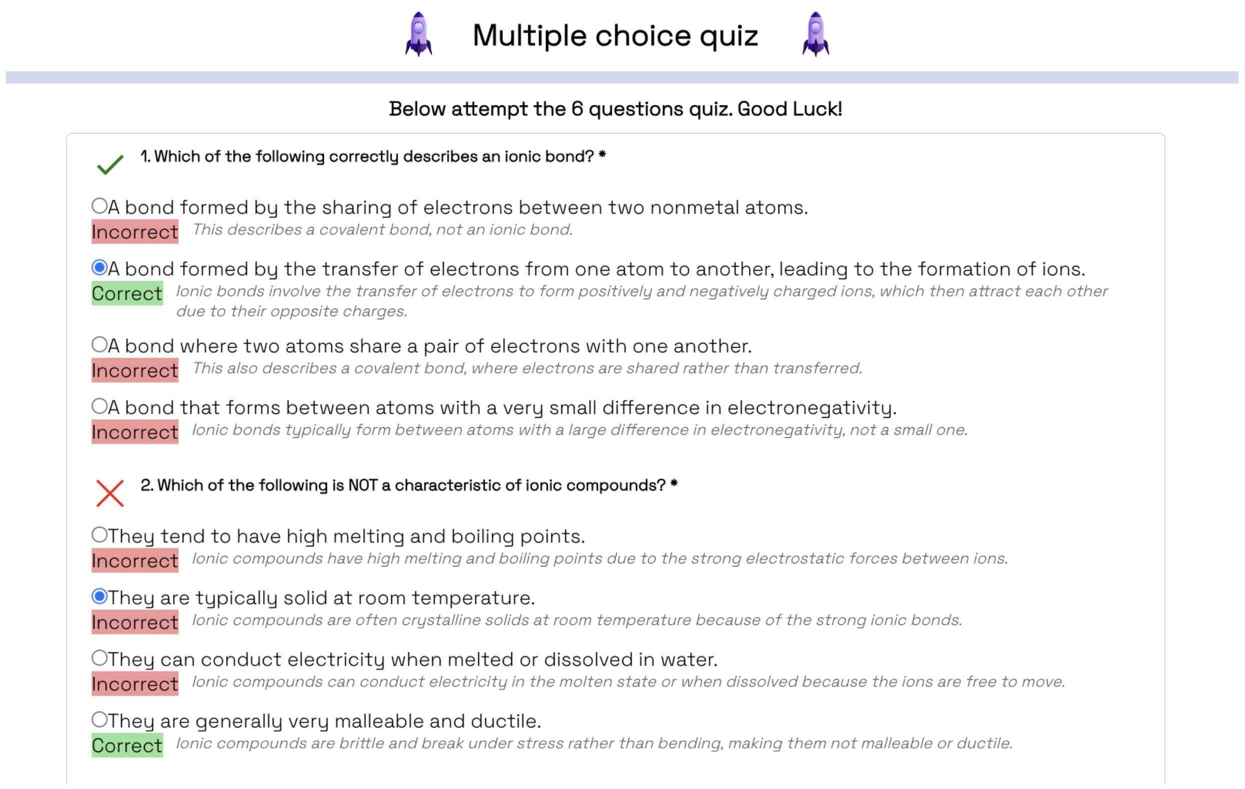
1. Performance Assessment and Feedback

Over the course of the semester, I successfully developed software that mostly achieved the objectives outlined in the beginning of the project within its constraints. While the quiz maker feature has yet to be integrated into the Alby platform, only minor modifications are required in both the software code and the Alby platform's backend. The personalized quiz maker was adeptly designed using the same cloud framework, utilizing data already captured by the platform. It effectively engaged OpenAI's GPT-4 API and implemented checks to ensure the consistent production of dynamically adapted quizzes, formatted according to individual student performance data. Furthermore, the output of the quizzes was rigorously tested to confirm compatibility with the expected input requirements for rendering and displaying the quizzes on the website.

Given the time constraints, limited feedback was received. However, one piece of feedback—suggesting that both units and lessons be dynamically selected beyond merely targeting the weakest areas—was incorporated during the Optimization and Integration phase of the project. Another comment, which remains unaddressed, highlighted that the quiz generation process takes approximately 1.5 minutes and pointed out that it is considered a long wait time from the user's standpoint. Therefore, enhancing the efficiency of this process is a priority for future iterations. Additionally, I observed that while the generated questions adequately covered the required concepts, it would be interesting to have overlapping units to have integrated questions that combine concepts from both units within a single multiple choice question.

Moving forward, the implementation phase will need to focus on deploying the feature in a live environment to gather data on its performance and impact on student engagement and learning outcomes. This will be crucial for validating the effectiveness of the personalized quizzes and for making necessary adjustments to optimize the feature's functionality and user experience.

Image of an example Quiz Generated by Software in the Current Frontend Display


Multiple choice quiz

Below attempt the 6 questions quiz. Good Luck!

✓ 1. Which of the following correctly describes an ionic bond? *

○ A bond formed by the sharing of electrons between two nonmetal atoms.
Incorrect *This describes a covalent bond, not an ionic bond.*

● A bond formed by the transfer of electrons from one atom to another, leading to the formation of ions.
Correct *Ionic bonds involve the transfer of electrons to form positively and negatively charged ions, which then attract each other due to their opposite charges.*

○ A bond where two atoms share a pair of electrons with one another.
Incorrect *This also describes a covalent bond, where electrons are shared rather than transferred.*

○ A bond that forms between atoms with a very small difference in electronegativity.
Incorrect *Ionic bonds typically form between atoms with a large difference in electronegativity, not a small one.*

✗ 2. Which of the following is NOT a characteristic of ionic compounds? *

○ They tend to have high melting and boiling points.
Incorrect *Ionic compounds have high melting and boiling points due to the strong electrostatic forces between ions.*

● They are typically solid at room temperature.
Incorrect *Ionic compounds are often crystalline solids at room temperature because of the strong ionic bonds.*

○ They can conduct electricity when melted or dissolved in water.
Incorrect *Ionic compounds can conduct electricity in the molten state or when dissolved because the ions are free to move.*

○ They are generally very malleable and ductile.
Correct *Ionic compounds are brittle and break under stress rather than bending, making them not malleable or ductile.*

III. Conclusion

In conclusion, this software project report details the development of a quiz generation feature for a learning management system and designed to enhance educational experiences through AI-driven technologies. The software, developed as an AWS state machine and uses OpenAI's GPT-4 API, dynamically generates quizzes tailored to student performance and stores them into their designated directory in an S3 bucket.

Throughout the semester, substantial progress was made towards achieving the project's objectives. While full integration with the Alby platform and databases like DynamoDB remains incomplete, only minor modifications are necessary to fully operationalize the quiz maker feature. The incorporation of feedback to dynamically select units and lessons, beyond merely focusing on weak areas, has significantly improved the project's functionality. However, the current quiz generation time of 1.5 minutes has been identified as a potential hindrance to user experience, suggesting a need for optimization in future iterations.

Looking ahead, future developments may include transitioning from OpenAI's GPT-4 to a custom-tailored model. This shift aims to enhance processing efficiency and align quiz content more closely with specific curricular needs, further personalizing the educational tool and potentially revolutionizing how students interact with learning material.

IV. Appendices

Link to git repository: https://github.swarthmore.edu/lkang1/E90_Personalized_quiz.git