Quiz Master: Project Abstract

Group name: NULL Pointers

Group:

Parth Patel

Dev Patel

Shailen Sutradhar

Gautam Santhanu Thampy

Date: February 11th, 2025

Class: CMPE272 Sec 03

Prof: Andrew Bond

Table of Contents

Introduction

Purpose

QuizMaster is designed to an application that enables users to assess their knowledge and proficiency on a specific topic of their choice. Its core functionality is to generate quizzes based on user-provided content.

Users can upload a PDF document containing content for which they would like to test their knowledge in. QuizMaster will generate a quiz tailored to this content and provide it to the user in a simple, user-friendly interface. Upon completion, users can receive instant grading, feedback on correct and incorrect answers, and the ability to retake the quiz as many times as they like.

Furthermore, when logged in/signed up, QuizMaster will also allow users to save quizzes for future retakes.

Intended Audience

The QuizMaster tool is designed to empower the executives and students across schools, universities, enterprises and a portion of general audience.

- Students can upload lecture slides and textbooks to generate tests helping them prepare for exams and be able to test their knowledge on a topic.
- Enterprises can integrate the tool in their employee recruitment, training and assessment.
- Universities can use QuizMaster to automate composing examination questions and reducing the risk of cheating with the ability to generate unique questions for each test.
- General audience can utilize the tool to generate quizzes from books, documents and research papers.

Technical stack

Frontend: React, Typescript, Tailwindcss

Backend: AWS Lambda, API Gateway

File storage: Amazon S3

Database: AWS DynamoDB

LLM: Google Gemini

Authorization: Firebase

CI/CD: GitHub Actions, Code Build

Version control: GitHub, Git

Reasoning for selected tech stack:

React:

- Component based code for front-end makes it more readable and code is reusable for future uses
- Responsive and faster to load
- Better documentation and resources to create a user-friendly interface

AWS Lambda:

- Affordable cost due to its serverless function and only queries made to the function are charged
- Easier integration with other AWS services
- Faster

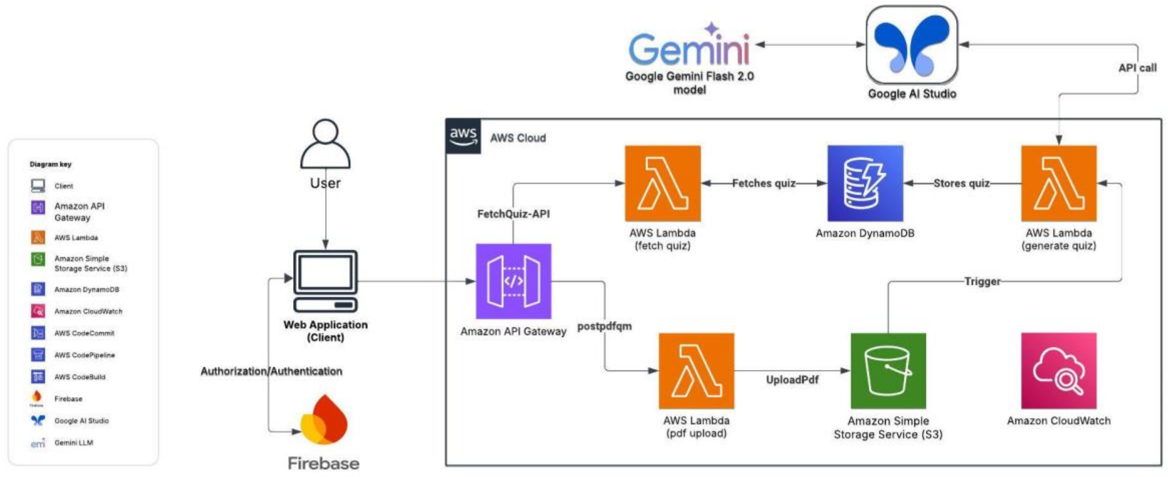
AWS API Gateway:

- Secure deployment of Rest APIs
- Scalable to handle large number traffic loads
- Seamless integration with AWS Lambda

Code Build:

- Offers free 100 build minutes/month
- Allows automated software release through multiple environments

Architecture & High-Level Design



Link to [Lucid Chart]()

Definitions and acronyms

AWS: Amazon Web Services. A cloud computing platform that offers a variety of services, including storage, computing and content delivery.

EC2: Amazon Elastic Compute Cloud. A web service that let users run application in the Amazon Web Services cloud.

S3: Amazon Simple Storage Service. A cloud-based object storage service that allows users to store and retrieve data.

LLM: Large Language Model. A type of artificial intelligence that can understand and generate human language.

API Gateway: Application Programming Interface Gateway. A software tool that manages API calls between clients and backend services.

DynamoDB: Amazon DynamoDB. A cloud-based database service.

CI/CD: Continuous Integration and Continuous Deployment/Delivery. A software development practice that uses automation to speed up and improve the software development process.

Git: Global Information Tracker. A free, open-source version control system.

Project risks

Accessibility: The application must be accessible by the users all time. For server or some other failure, the user might not be able to access the website at all. Besides, the saved data might be inaccessible for the user.

Responsiveness: The webpage might not be responsive across devices with different screen sizes which might lead to broken layout, unreadable texts and a poor user interface.

File Handling: The application will not function if the user uploads a file too big for the LLM or a file of inappropriate type.

Performance: It might take long for the quiz to be generated which will lead to inefficiency. Sometimes, failure to generate quizzes can also happen.

Security: The authentication can fail, which will lead to unauthorized access. There can be breaches in stored quiz data leading to privacy and security concerns.

Incorrect or Unethical Quiz: The AI model might generate unrealistic or inappropriate questions and incorrect answers. It might also generate questions out of the given material.

Scalability: A high number of users trying to access the webpage can slow down the application and cause it to crash.

Development process

1. Requirements gathering (completed)
   a. Coming up with features
   b. Designing the website
   c. Choosing an LLM
2. Creating the frontend (in progress)
   a. Creating a home page
   b. Creating a quiz page
   c. Creating authorization
   d. Creating backend
   e. Testing
3. Creating an organization on AWS for team collaboration
4. Creating storage (S3 and DynamoDB)
5. Setting up API gateway and Lambda functions
   a. Setting up
   b. Implementing
   c. Testing
6. Setting up Gemini account
7. Enabling Code Deploy and EC2 for deployment
   a. Provisioning
   b. Testing

Milestones & Time schedule

Tasks to Accomplish	Total weeks	Complete by
Requirements gathering Research which LLM model to use, and setup an account with it (ChatGPT/Gemini/Deep Seek etc.) Creating an AWS organization account for collaboration	1	Feb 10th
Develop a functional home page. Creating an S3 bucket Setting up a Dynamo database	2	Feb 24th
Milestone #1 Team meeting + Testing + Next steps		
Create the Quiz frontend component	2	March 10th

Implement the pdfUploadToS3 Lambda function Create NoSQL mock data for DynamoDB Test connection to AWS lambda Test trigger events to S3 Test responsiveness of frontend		
Milestone #2 Team meeting + Testing + Next steps		
Setup Firebase and backend for authorization Do prompt engineering with LLM of choice. Test authorization with Gmail, and new email Test example PDFs with prompts	1	March 17th
Milestone #3 Team meeting + Testing + Next steps		
Implement the Lambda function responsible for populating DynamoDB Implement backend for fetching data from DynamoDB Test populate dynamo lambda function Test fetching data from dynamo lambda function	2	March 31st
Milestone #4 Team meeting + Testing + Next steps		
Provisioning EC2 and Code Build for deployment End-to-end testing	2	April 14th
Milestone #5		

Team meeting + Final Testing + Presentation planning		
Creating a presentation Assigning slides to each member Practice presenting	Remaining time: ~ 1 Month	May 12th
PRESENT PROJECT IN CLASS		

Organization

Project group

Name	Role
Parth Patel	Frontend development Client-side backend develop Data fetch Lambda Development Testing
Shailen Sutradhar	S3 Lambda Development Provisioning resources Testing
Dev Patel	Code deployment Provisioning resources Testing
Gautam Santhanu Thampy	LLM Lambda development Prompt engineering Testing

Communication

Github

Link: https://github.com/ParthPatel00/QuizMaster