Serverless Quiz Application with AWS DynamoDB and Lambda

A Course Project Report Submitted in partial fulfillment of the course requirements for the award of grades in the subject of

CLOUD BASED AIML SPECIALITY
(22SDCS07R)

by

SRIKARSHA PANEM

(2210030432)

Under the esteemed guidance of

Ms. P. Sree Lakshmi
Assistant Professor,
Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K L Deemed to be UNIVERSITY

Aziznagar, Moinabad, Hyderabad,
Telangana, Pincode: 500075

April 2025

K L Deemed to be UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

This is Certified that the project entitled **"Serverless Quiz Application with AWS DynamoDB and Lambda"** which is a experimental &/ theoretical &/ Simulation&/ hardware work carried out by SRIKARSHA PANEM (2210030432), in partial fulfillment of the course requirements for the award of grades in the subject of **CLOUD BASED AIML SPECIALITY**, during the year **2024-2025**. The project has been approved as it satisfies the academic requirements.

Ms.P.Sree Lakshmi

Course Coordinator

Dr. Arpita Gupta

Head of the Department

Ms. P. Sree Lakshmi

Course Instructor

CONTENTS

Page No.

1. INTRODUCTION

In today's digital era, the demand for scalable, secure, and easily deployable applications has grown exponentially. This project, titled "AWS-Powered Online Quiz System", is a cloud-based web application designed to collect user details, administer quizzes dynamically, evaluate performance, and store user scores—all leveraging the robust ecosystem of Amazon Web Services (AWS). The primary objective of this project is to provide a seamless, interactive quiz-taking experience while ensuring data persistence and backend reliability using serverless cloud infrastructure.

This project enables users to begin a quiz after entering personal details such as name, email, and mobile number. The quiz content—comprising questions, options, and correct answers—is fetched from an external JSON file hosted on AWS S3. This approach allows quiz administrators to update or change quiz content without altering the core HTML or JavaScript logic. The answers submitted by the user are evaluated client-side, and upon completion, the user's score, along with their personal details, is sent to an AWS API Gateway endpoint. This endpoint triggers an AWS Lambda function, which in turn stores the data in DynamoDB, ensuring high availability and scalability.

The application architecture was carefully designed using key AWS services like S3 (for static website hosting and question storage), API Gateway (to manage REST API requests), Lambda (to handle backend logic serverlessly), and DynamoDB (to persist user information). Furthermore, CORS policies and HTTP status management were properly configured to ensure smooth client-server communication without interruptions or cross-origin errors.

This project is an excellent example of how frontend and backend systems can be decoupled and seamlessly integrated using cloud services. It also demonstrates the power of serverless architecture in reducing operational overhead while ensuring cost efficiency and ease of deployment. By the end of this documentation, readers will have a thorough understanding of how modern web applications can be built, deployed, and scaled using cloud-native tools and services.

2. AWS Services Used as part of the project

1) Amazon S3

Used to host the static website files such as index.html, quiz.html, results.html, and the questions.json file.

2) AWS Lambda

A serverless function that receives POST requests containing user details and quiz scores, processes them, and stores them in DynamoDB.

3) Amazon API Gateway

Exposes a REST endpoint that triggers the Lambda function. Handles CORS and request-response formatting.

4) Amazon DynamoDB

NoSQL database used to persist user data and quiz scores in a structured table with email as the primary key.

5) IAM (Identity and Access Management)

Manages roles and permissions between Lambda and DynamoDB.

3. Steps involved in solving project problem statement

Step 1: Design the UI

Three main HTML pages were created:

- index.html: Registration form
- quiz.html: Quiz interface with dynamic questions
- results.html: Display final score

Step 2: Store Questions in JSON

A questions.json file stores quiz data with the following structure:

```
[
  {
    "question": "What is the capital of France?",
    "options": ["Paris", "London", "Berlin", "Madrid"],
    "answer": "Paris"
  },
  ...
]
```

Step 3: Host Files on Amazon S3

All frontend files including questions.json were uploaded to an S3 bucket configured for static website hosting.

Step 4: Create DynamoDB Table

A table named QuizResults was created with:

- email as the partition key
- Attributes: name, mobile, score

Step 5: Write and Deploy Lambda Function

The Lambda function was written in Node.js to:

- Parse request body
- Extract name, email, mobile, and score
- Save them to DynamoDB

Step 6: Configure API Gateway

An HTTP POST method was added under API Gateway to:

- Connect to Lambda
- Allow CORS with headers like:
    - Access-Control-Allow-Origin: *
    - Access-Control-Allow-Methods: POST, OPTIONS

Step 7: Integrate JavaScript with API

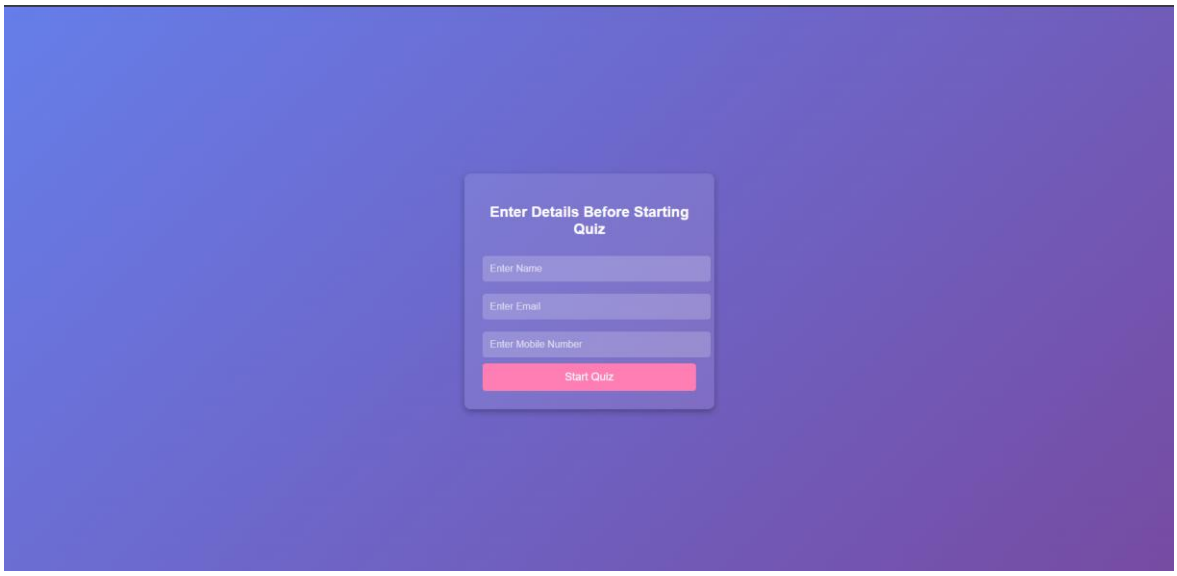
JavaScript in the quiz frontend:

- Calculates score on submit
- Sends a POST request with JSON data to the API endpoint
- Stores response and score in localStorage
- Redirects to results.html

4. Stepwise Screenshots with brief description

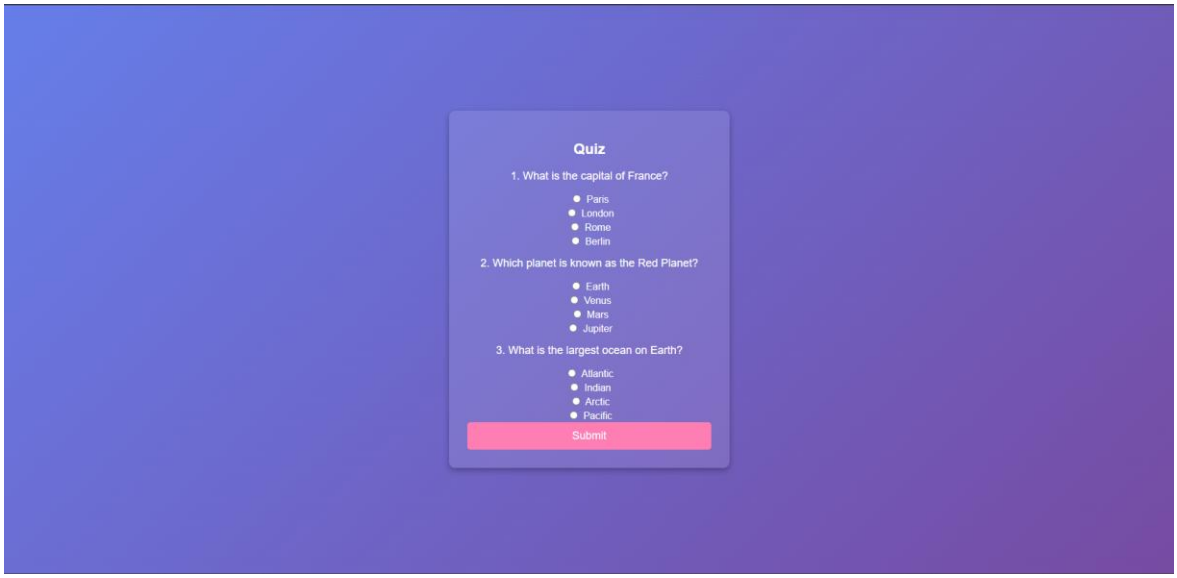
1. User Registration Form (index.html)

Collects name, email, mobile number



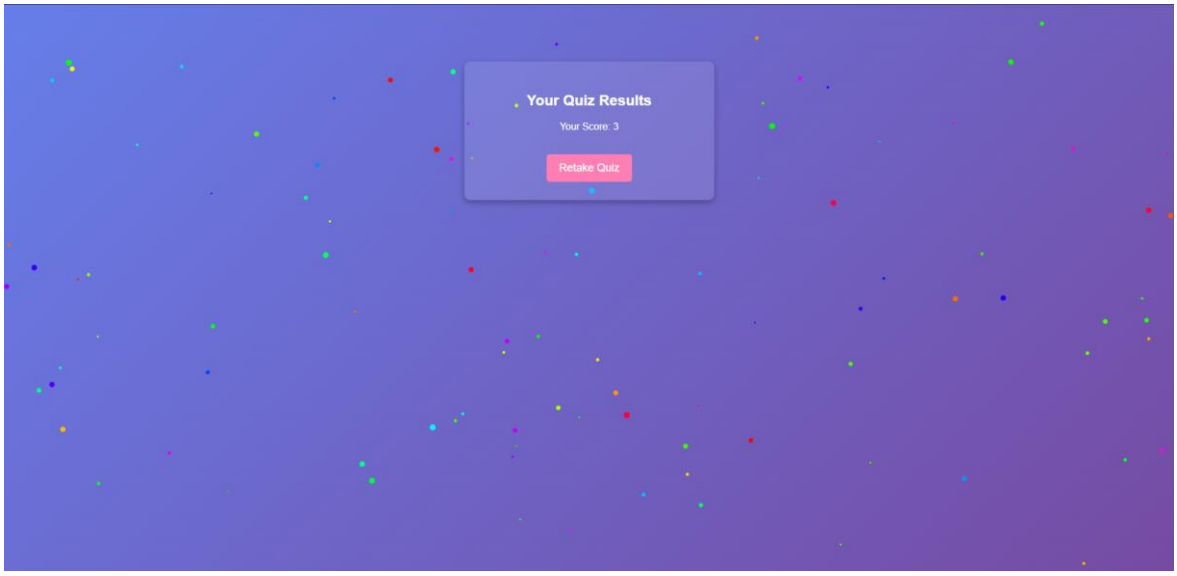
2. Quiz Interface (quiz.html)

- Dynamically loads questions from questions.json
- Validates selected answers
- Calculates and displays score



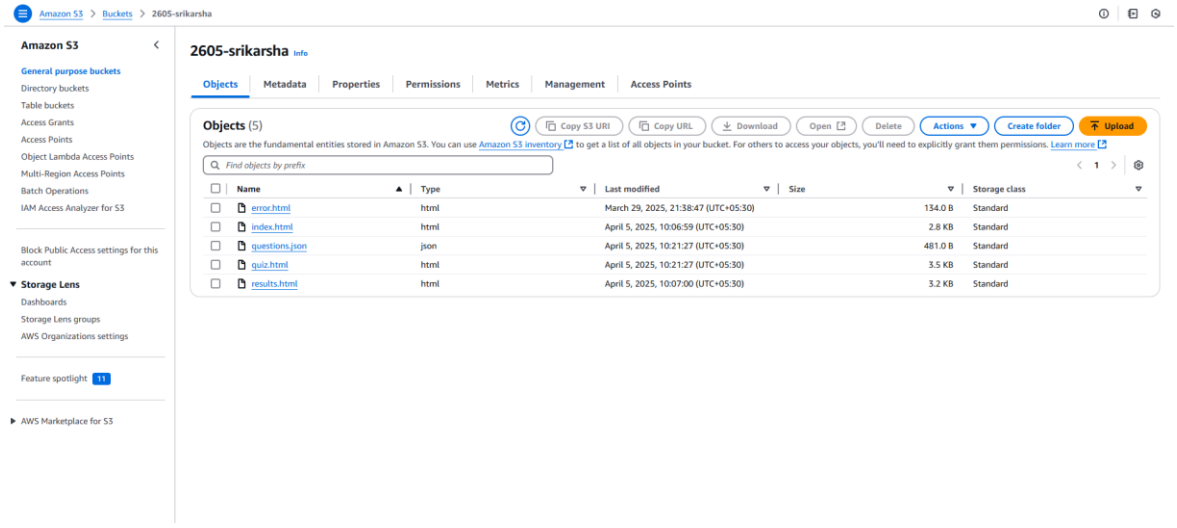
3. Results Page (results.html)

- Retrieves score from local storage
- Shows result and allows quiz retake



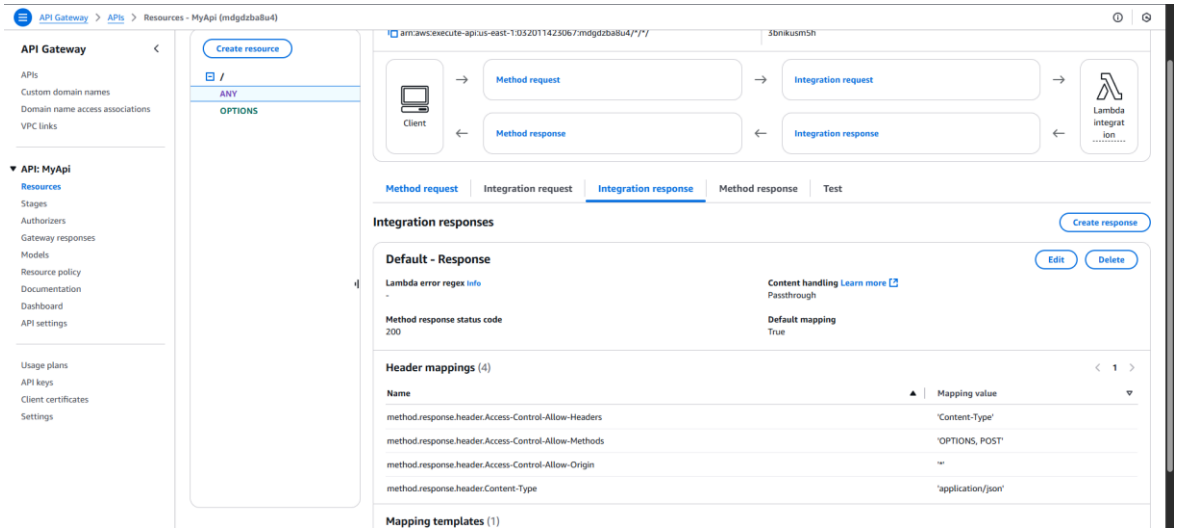
4. S3 Hosting Setup

- Shows the public URL
- Bucket permissions and static hosting config



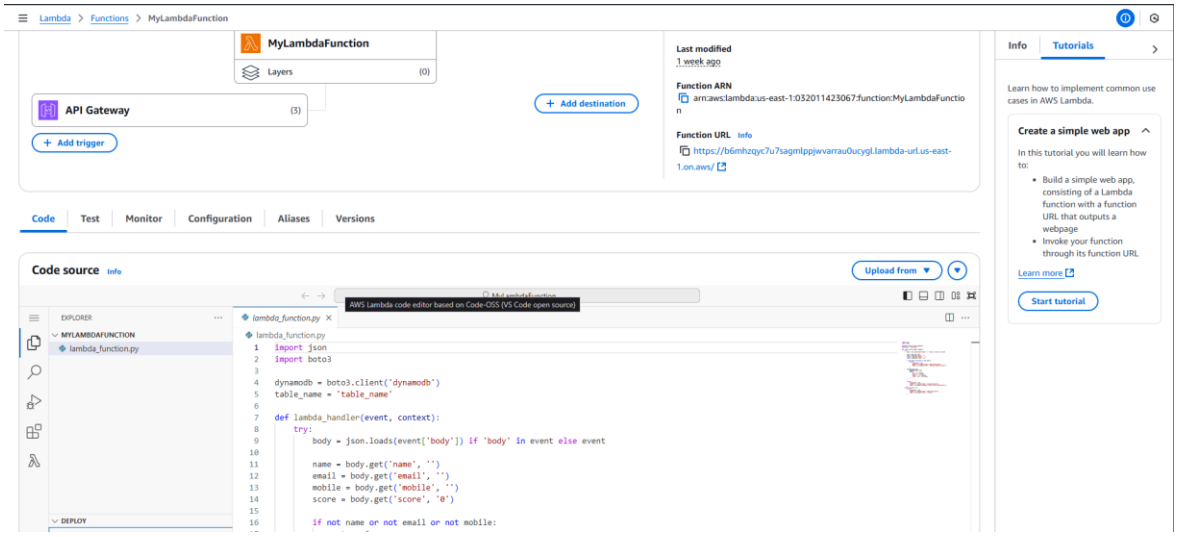
5. API Gateway Setup

- Method Execution → Lambda Integration



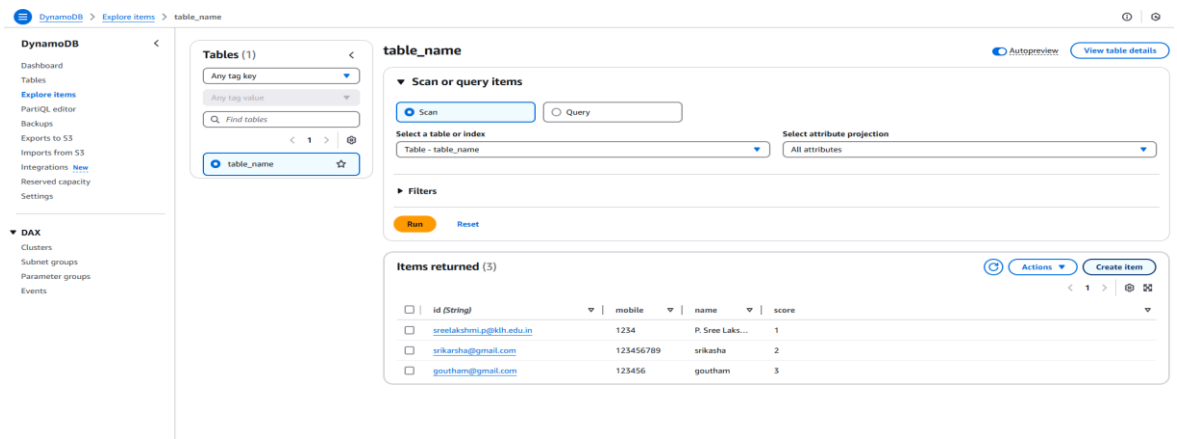
6. Lambda Function

- Code editor showing JSON parsing and DynamoDB insertion



7. DynamoDB Table

- Table structure and sample records



5. Learning Outcomes

- **Full-Stack Serverless Architecture**
  Learned to build a full-stack serverless web application using AWS services without managing any servers. Understood how frontend and backend interact in a serverless environment.

- **Amazon S3 for Static Hosting**
  Gained experience in hosting static websites using S3. Also used S3 to load dynamic JSON data for quiz questions.

- **API Gateway for Backend Integration**
  Learned how to set up REST APIs using API Gateway, including handling methods, CORS, and integration with Lambda.

- **AWS Lambda for Backend Logic**
  Used AWS Lambda to process quiz results and store data. Understood how serverless functions execute code in response to API calls.

- **DynamoDB for Data Storage**
  Gained practical exposure to NoSQL databases. Learned how to store and retrieve structured user data efficiently.

- **CORS and HTTP Handling**
  Configured CORS policies for smooth communication between frontend and backend. Learned the importance of headers in cross-origin requests.

- **JavaScript and Fetch API**
  Enhanced JavaScript skills by using localStorage, Fetch API, and DOM manipulation to build a dynamic quiz experience.

- **IAM Roles and Permissions**
  Understood how to manage secure access between AWS services using IAM. Set appropriate permissions for Lambda to access DynamoDB.

- **Debugging and Deployment**
  Improved problem-solving through hands-on debugging of CORS, redirection, and data handling issues. Deployed using the AWS Console.

- **Scalability and Cloud Benefits**
  Realized the cost and performance benefits of serverless applications. Appreciated how AWS services automatically scale based on usage.

6. Conclusion

The project successfully demonstrates how modern cloud technologies—particularly AWS services—can be effectively used to develop a robust, secure, and scalable serverless quiz application. It provides a seamless user experience by integrating a static frontend hosted on Amazon S3 with dynamic backend processing using AWS Lambda and API Gateway.

Users can register through a responsive web interface, take a quiz whose questions are dynamically loaded from a JSON file, and have their responses evaluated instantly. Their personal information and scores are securely stored in Amazon DynamoDB, ensuring persistence and reliability without the need for managing any server infrastructure.

The use of JSON for storing quiz content offers a flexible approach to modifying or extending the quiz without changing the application code. The project also handled real-world challenges like CORS errors, response handling, and secure data access using IAM roles.

Overall, the application showcases a practical implementation of serverless computing with real-time interactivity, data persistence, and dynamic content. It highlights the power of AWS in simplifying development workflows while ensuring scalability, cost-efficiency, and security—making it a valuable learning experience and a deployable solution.

7. REFERENCES

1. https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteHosting.html
2. https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors.html
3. https://docs.aws.amazon.com/lambda/latest/dg/welcome.html
4. https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html
5. https://docs.aws.amazon.com/s3/