**ExamQuizTitle**

**API Summary**

* **API URL:** <https://1f7vc4um1i.execute-api.us-east-1.amazonaws.com/v1/fetchQuizTitle>
* **API Name:** quizTitle
* **Resource Name:** fetchQuizTitle
* **Stage:** v1

**Functionality**

The fetchQuizTitle API retrieves quiz IDs and titles associated with a user's email address from MongoDB, based on a JWT token provided for authentication.

**Request**

* **Method:** POST
* **Headers:**
  + Authorization: JWT token for authentication
  + Content-Type: application/json (expected)
* **Body:**
  + JSON object containing userEmail field

**Responses**

* **Success Response (200 OK):**
  + Returns an array of objects containing quizId and quizTitle for each quiz found.
  + Example:

json

Copy code

{

"quizzes": [

{

"quizId": "611e8827f4b30f001f78581d",

"quizTitle": "Sample Quiz 1"

},

{

"quizId": "611e8827f4b30f001f78581e",

"quizTitle": "Sample Quiz 2"

}

]

}

* **Error Responses:**
  + **400 Bad Request:**
    - If request body is empty or contains invalid JSON.
    - If userEmail is missing in the request body.
  + **401 Unauthorized:**
    - If JWT token is missing or invalid.
  + **500 Internal Server Error:**
    - If there's an error querying the database.

**Payload Example**

headers

Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2NjcxOWJjMWZhOWQ3ZjBmZDllNDVkODYiLCJlbWFpbCI6InNoaXZhbmlzaW5naDAzMTIwMkBnbWFpbC5jb20iLCJpYXQiOjE3MjAzNjA1NDEsImV4cCI6MTcyMDM3ODU0MX0.4t9kb4SQEH-GgeCu4oZ8T5\_TlSuGeMvK2G8vHZbdHyw

Content-Type: application/json

Body:

{

"userEmail": "shivanisingh031202@gmail.com"

}

**Response Example**

json

Copy code

HTTP/1.1 200 OK

Content-Type: application/json

{

    "quizzes": [

        {

            "quizId": "66719bf95b96cfa4a177d81c",

            "quizTitle": "science"

        },

        {

            "quizId": "6673f66d9b7621163cfab9d9",

            "quizTitle": "Shivani"

        },

        {

            "quizId": "66741599499dee1e64baa87b",

            "quizTitle": "science"

        },

        {

            "quizId": "667415a2499dee1e64baa87f",

            "quizTitle": "Computer"

        },

}

CODE:

const jwt = require('jsonwebtoken');

const mongoose = require('mongoose');

const Quiz = require('./question'); // Import your Quiz model

const JWT\_SECRET\_KEY = process.env.JWT\_SECRET\_KEY;

const MONGODB\_URI = process.env.MONGODB\_URI;

if (!JWT\_SECRET\_KEY || !MONGODB\_URI) {

throw new Error('Environment variables JWT\_SECRET\_KEY and MONGODB\_URI must be set.');

}

mongoose.connect(MONGODB\_URI);

mongoose.connection.on('error', err => {

console.error('MongoDB connection error:', err);

});

mongoose.connection.once('open', () => {

console.log('MongoDB connected');

});

// CORS headers for different request types

const corsHeaders = {

'Access-Control-Allow-Origin': 'https://student.exambuilder.online', // Allow all origins for simplicity

'Access-Control-Allow-Headers': 'Content-Type, Authorization',

'Access-Control-Allow-Methods': 'GET, POST, OPTIONS'

};

// Preflight request handler

const handlePreflight = (event) => {

return {

statusCode: 200,

headers: corsHeaders,

body: ''

};

};

exports.handler = async (event, context) => {

// Handle preflight requests

if (event.httpMethod === 'OPTIONS') {

return handlePreflight(event);

}

if (!event.body) {

return {

statusCode: 400,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ message: "Empty request body." })

};

}

const token = event.headers.Authorization;

if (!token) {

return {

statusCode: 401,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ error: 'Unauthorized: Missing token' }),

};

}

let decoded;

try {

decoded = jwt.verify(token, JWT\_SECRET\_KEY);

} catch (error) {

console.error('Error verifying token:', error);

return {

statusCode: 401,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ error: 'Unauthorized: Invalid token' }),

};

}

let requestBody;

try {

requestBody = JSON.parse(event.body);

} catch (error) {

console.error('Error parsing request body:', error);

return {

statusCode: 400,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ error: 'Bad Request: Invalid JSON' }),

};

}

const userEmail = requestBody.userEmail && requestBody.userEmail.toLowerCase();

if (!userEmail) {

return {

statusCode: 400,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ error: 'Bad Request: Missing userEmail in request body' }),

};

}

try {

const quizzes = await Quiz.find({ creatorEmail: userEmail })

.select('\_id quizTitle')

.exec();

const response = {

quizzes: quizzes.map(quiz => ({

quizId: quiz.\_id,

quizTitle: quiz.quizTitle

}))

};

return {

statusCode: 200,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify(response)

};

} catch (err) {

console.error('Database query error:', err);

return {

statusCode: 500,

headers: { 'Access-Control-Allow-Origin': 'https://student.exambuilder.online' },

body: JSON.stringify({ error: 'Internal Server Error' }),

};

}

};