QuizCrate Backend (Postman-Based)

QuizCrate – A RESTful backend API for online quiz management and scoring system.

# Objective

To build a backend system for:  
- Creating and storing quizzes  
- Attempting quizzes  
- Automatically evaluating answers and scoring  
- Storing quiz attempts  
- Viewing past attempts  
  
Testing is done using Postman, not a frontend UI.

# Technologies Used

- Node.js with Express.js  
- MongoDB with Mongoose  
- Postman for API testing

# MongoDB Collections

- users (Optional or static)  
- quizzes (Each quiz with questions, options, correct answers)  
- attempts (Stores answers, score, user, and quizId)

# Folder Structure

project-root/  
├── models/  
│ ├── Quiz.js  
│ └── Attempt.js  
├── routes/  
│ └── quiz.js  
├── server.js  
└── .env

# API Endpoints and Postman Test Cases

## Register User

Method: POST  
URL: /api/quiz/create  
Body (JSON):

const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({

name: String,

email: { type: String, unique: true },

password: String

});

module.exports = mongoose.model('User', userSchema);

**Example:**{

"name" : "ravi" ,

"email": "ravi56@gmail.com",

"password": "123456"

}  
  
  
**NOTE:**User passwords are **not stored as plain text** in the database.  
Instead, they are **securely hashed using the** bcrypt **library** before saving.  
This prevents password leaks even if the database is compromised.  
During login, the entered password is compared with the stored hashed password using bcrypt.compare().  
This is a standard and secure approach for authentication in web applications.

**CODE:**  
const bcrypt = require('bcrypt');

const hashedPassword = await bcrypt.hash(req.body.password, 10);

user.password = hashedPassword;

await user.save();

const isMatch = await bcrypt.compare(req.body.password, user.password);  
  
Original Password:123456  
Hashed Password:$2b$10$xCYSEB0Wtv0bTh/TpZSRZ.y0S.Rw9CgzyfgwVdPAuOheoS8EHJws2

## 2. Login User {

## "email": "ravi56@gmail.com",

## "password": "123456"

## }

## CODE router.post('/login', async (req, res) => {

## const { email, password } = req.body;

## try {

## // 1. Check if user exists

## const user = await User.findOne({ email });

## if (!user) {

## return res.status(400).json({ msg: 'Invalid credentials' });

## }

## const isMatch = await bcrypt.compare(password, user.password);

## if (!isMatch) {

## return res.status(400).json({ msg: 'Invalid credentials' });

## }

## res.status(200).json({

## msg: 'Login successful',

## user: {

## id: user.\_id,

## name: user.name,

## email: user.email

## }

## });

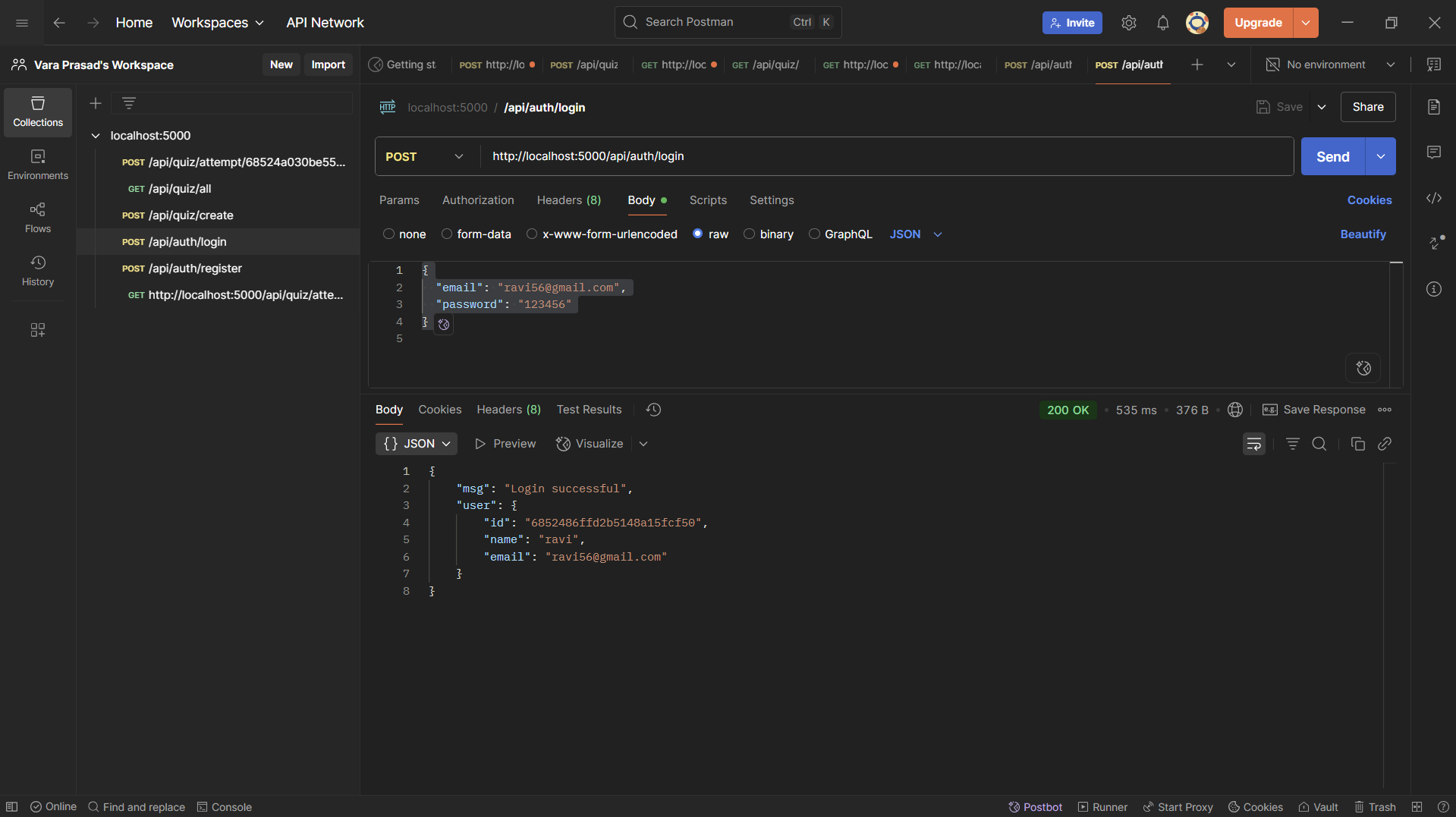
## } catch (err) {

## console.error(err.message);

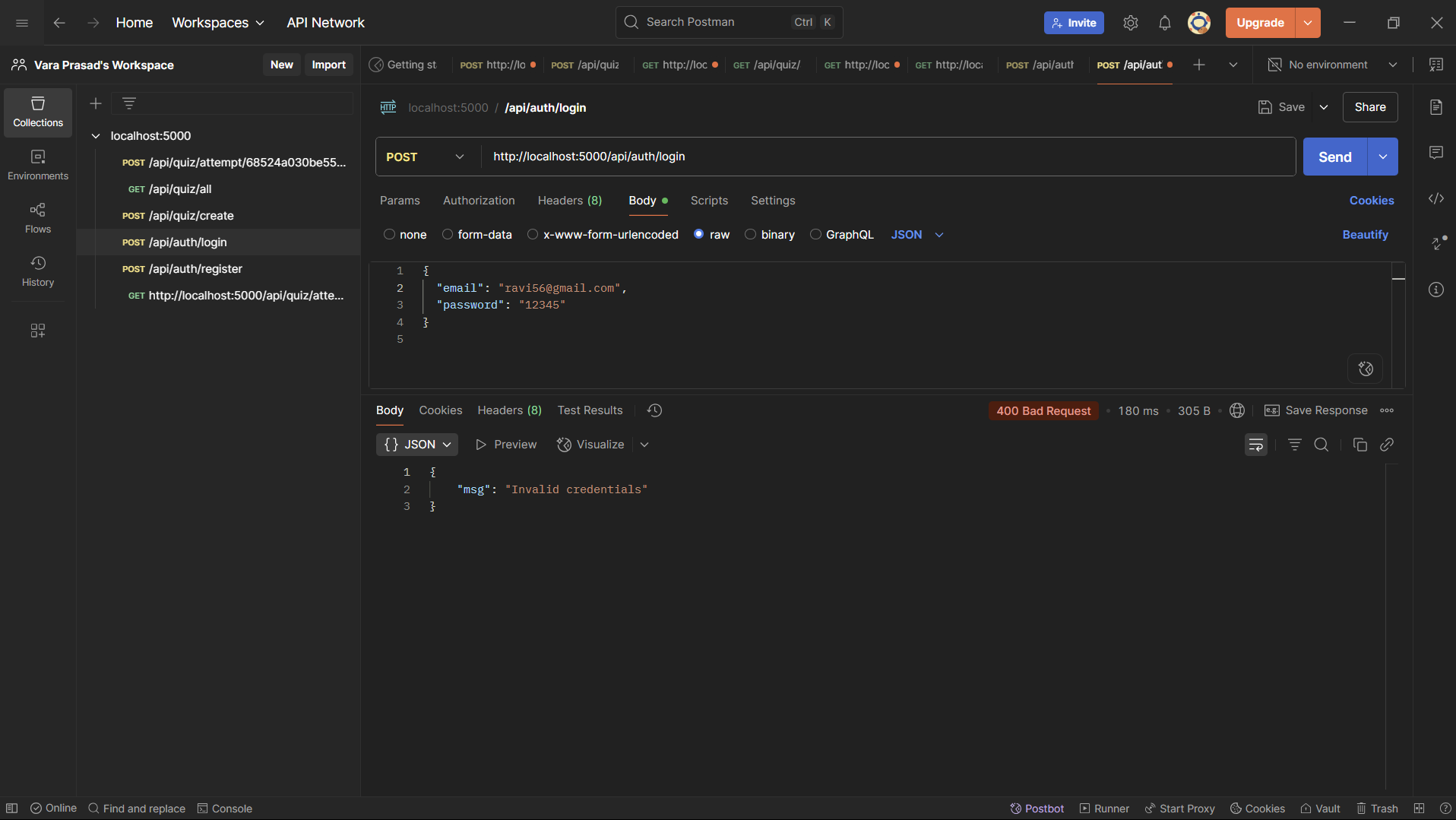
## res.status(500).send('Server error');

## }

## });

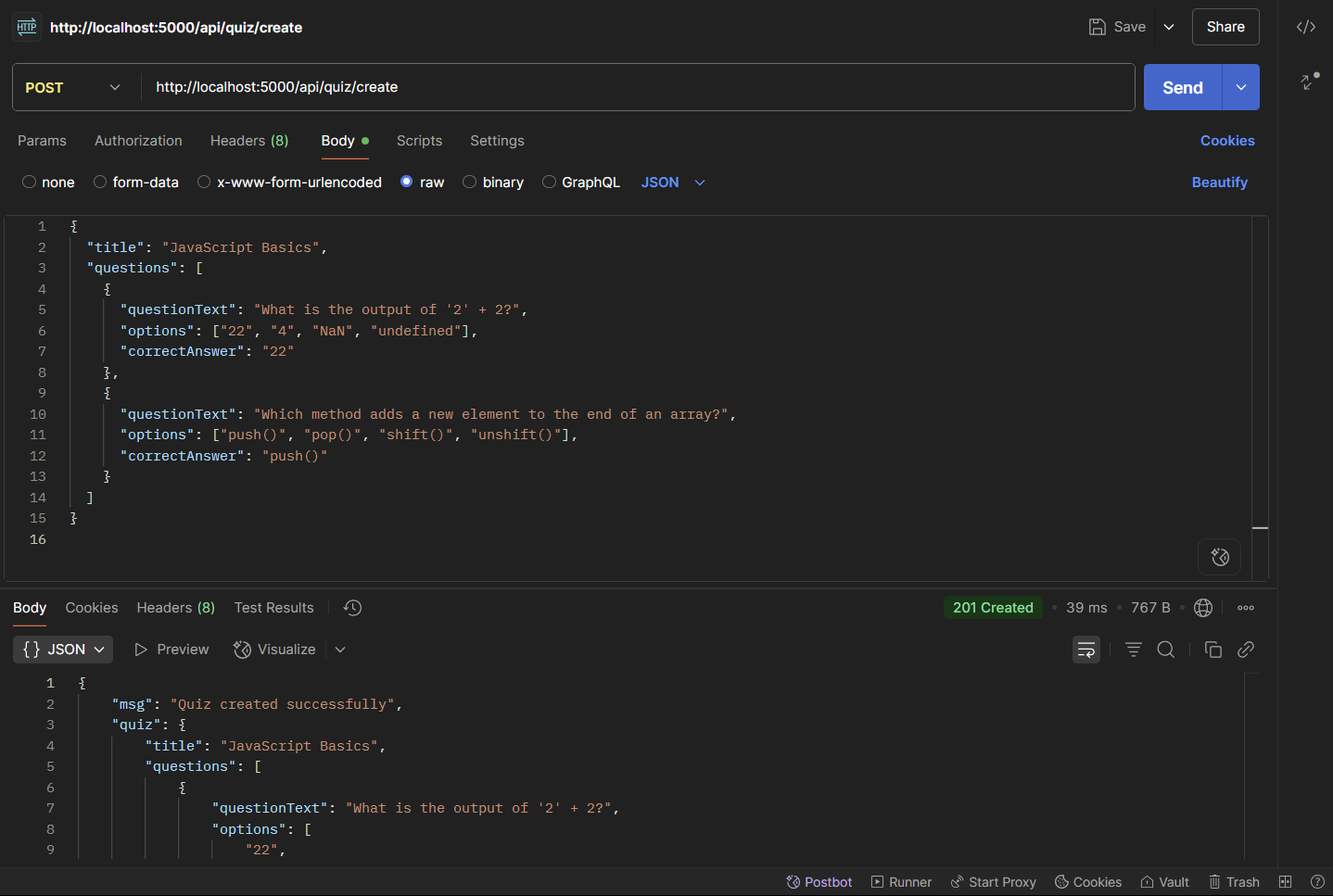
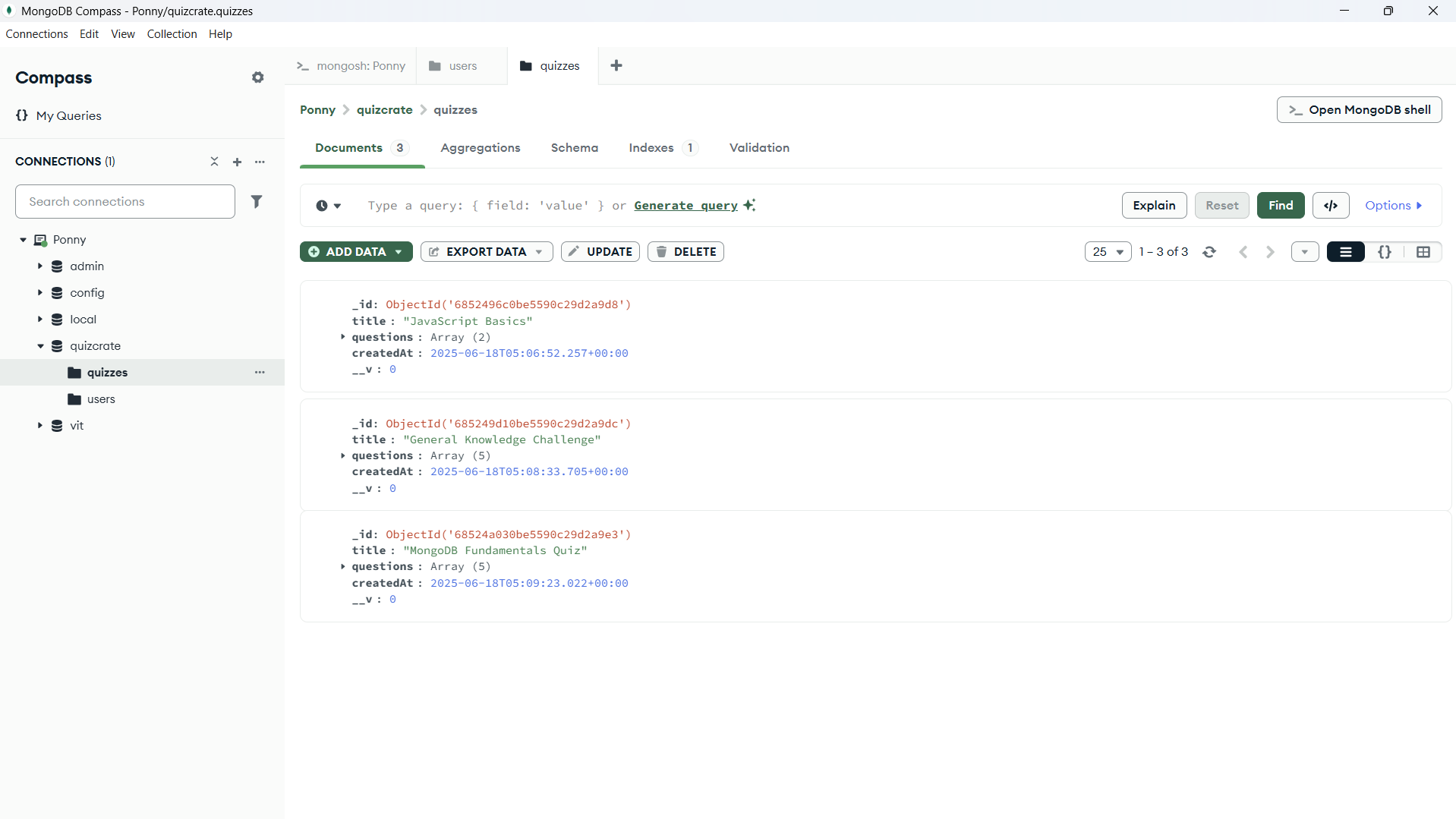


**Invalid Credentials:**



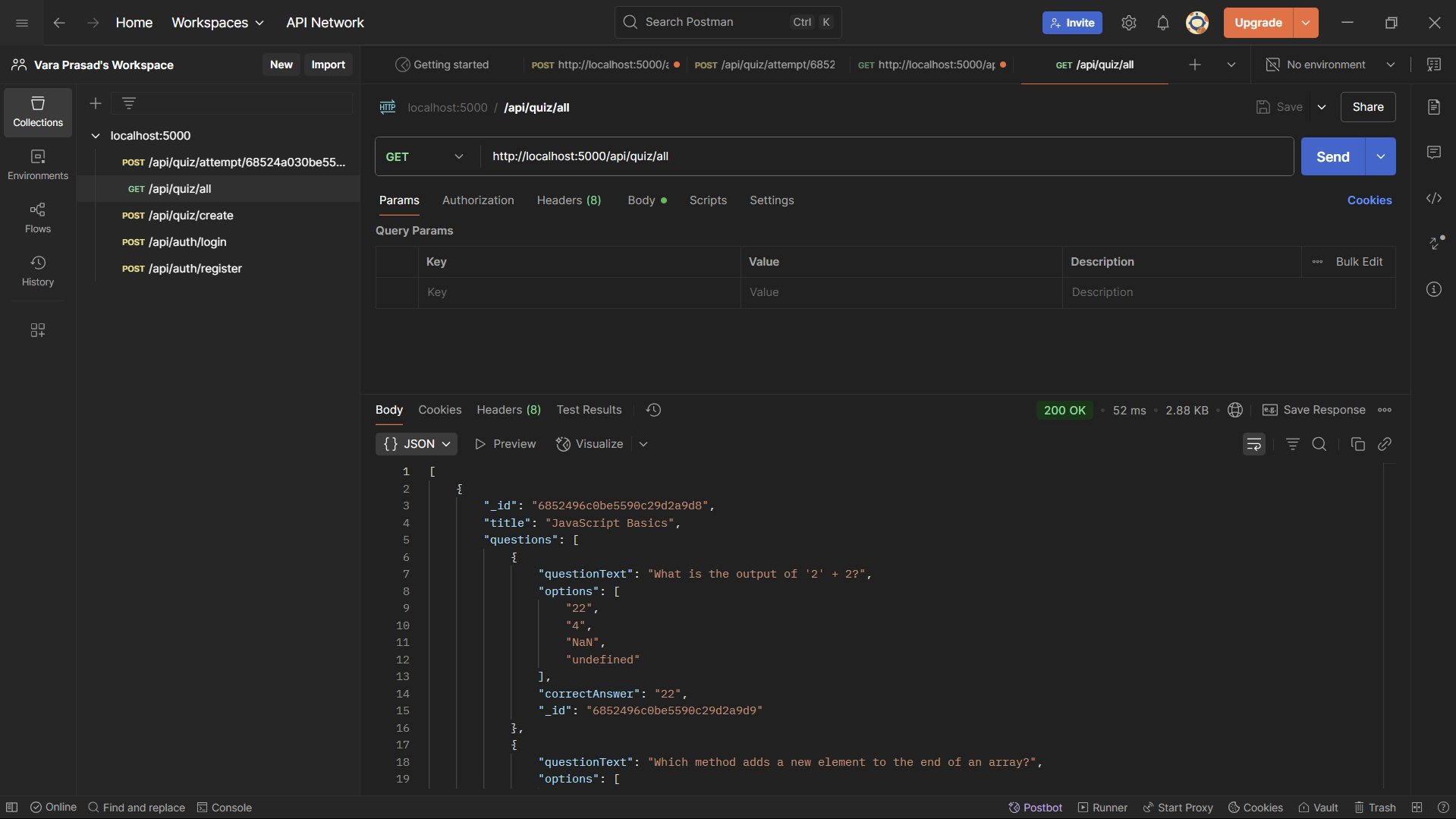
## 3. Create Quiz

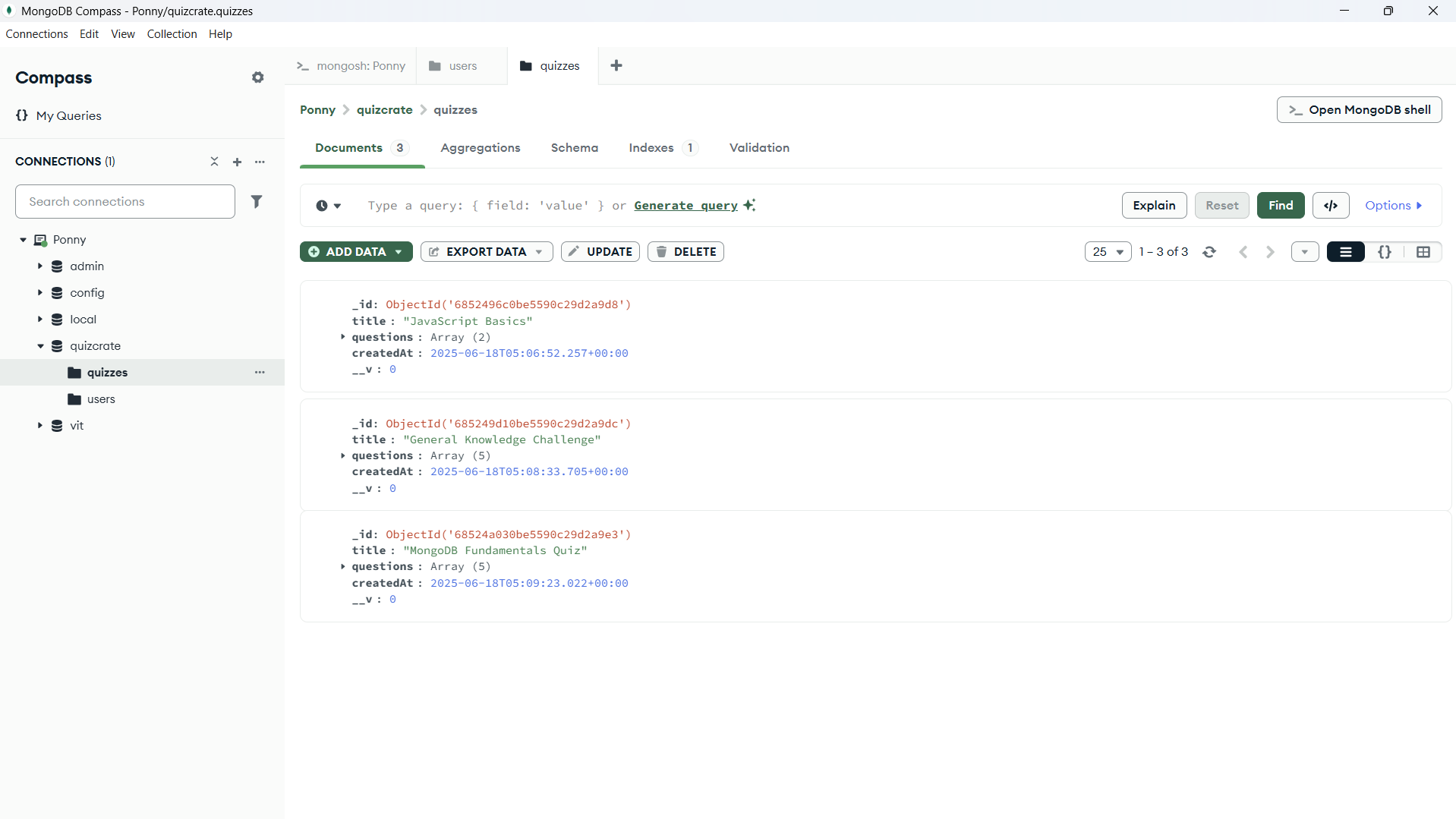
Method: POST  
URL: /api/quiz/create  
Body (JSON):  
{  
 "title": "General Knowledge Quiz",  
 "questions": [  
 {  
 "questionText": "What is the capital of France?",  
 "options": ["London", "Berlin", "Paris", "Rome"],  
 "correctAnswer": "Paris"  
 },  
 {  
 "questionText": "2 + 2 = ?",  
 "options": ["3", "4", "5", "6"],  
 "correctAnswer": "4"  
 }  
 ]  
}

## 4. Get All Quizzes

Method: GET  
URL: /api/quiz/all

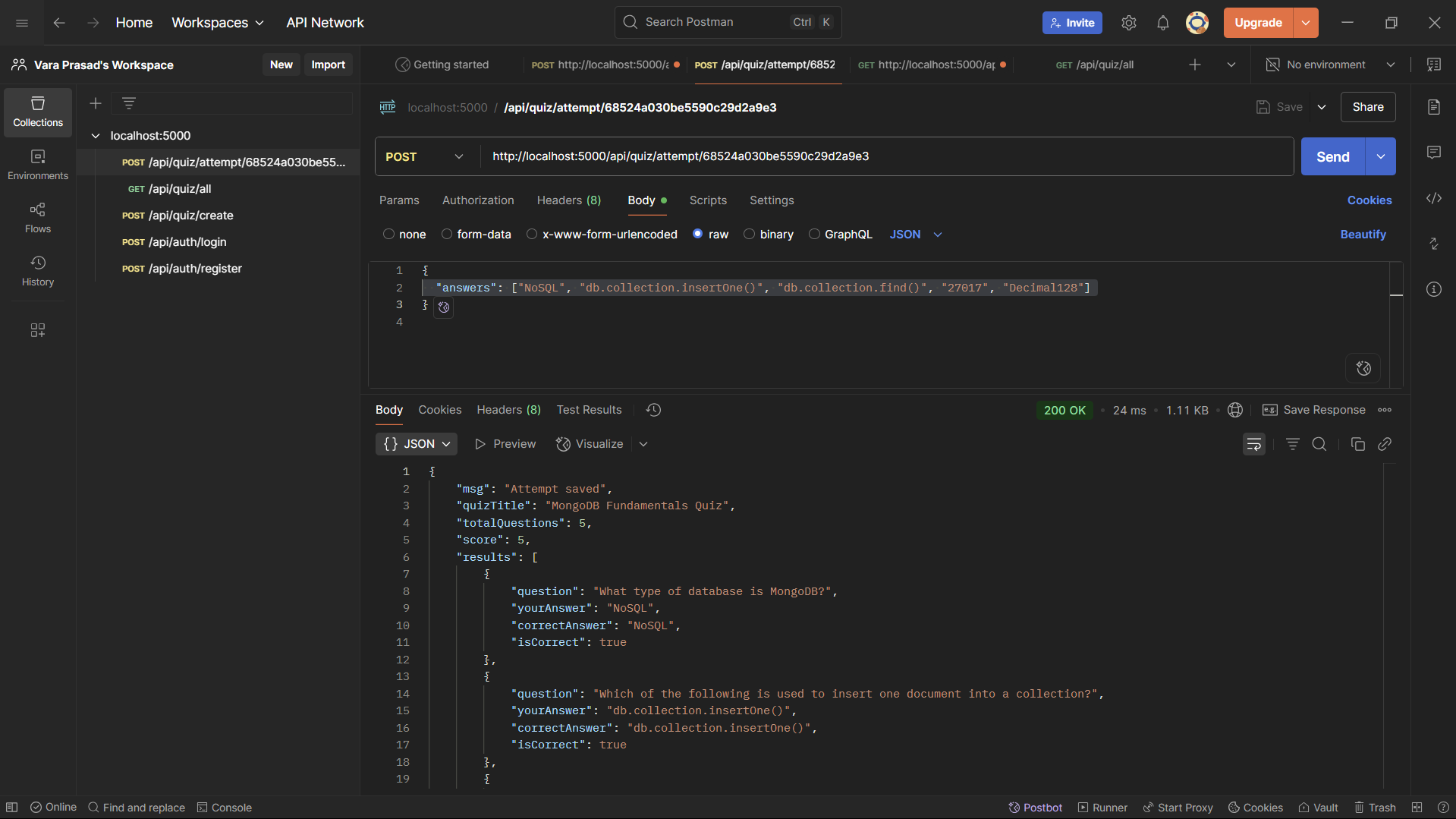


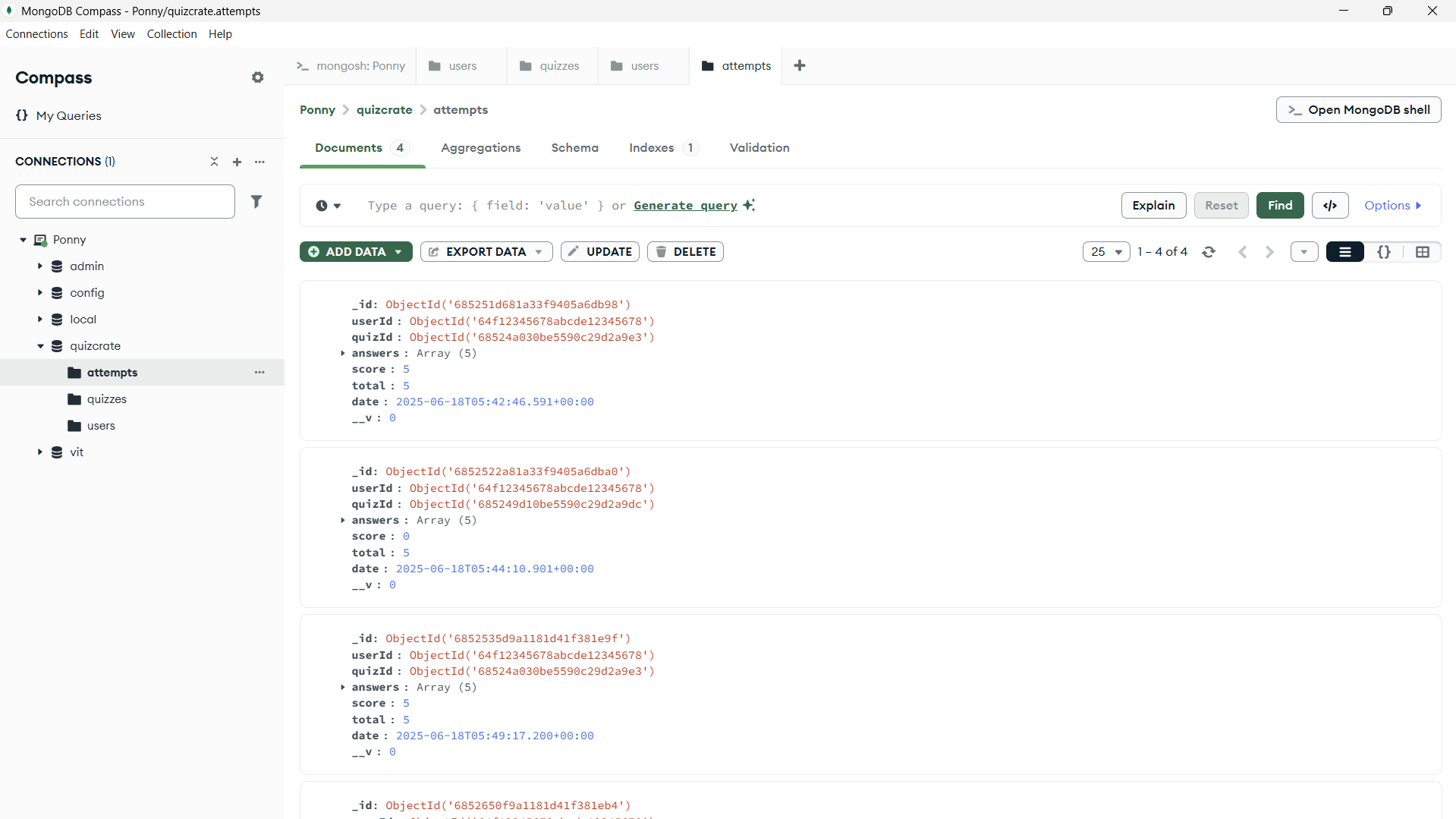


## 5. Attempt a Quiz

Method: POST  
URL: http://localhost:5000/api/quiz/attempt/68524a030be5590c29d2a9e3  
Body (JSON):  
{  
 "answers": ["NoSQL", "db.collection.insertOne()", "db.collection.find()", "27017", "Decimal128"]

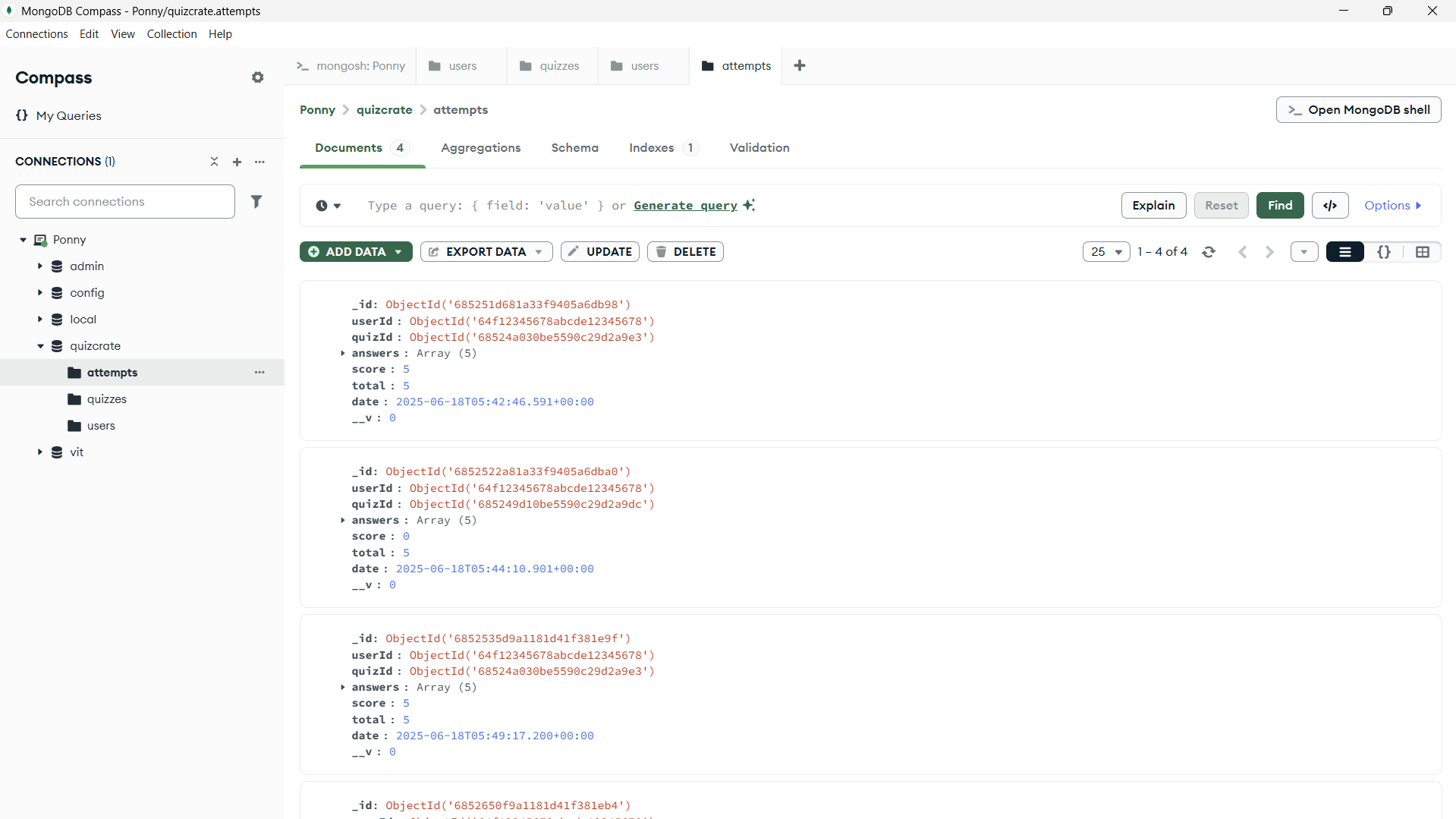
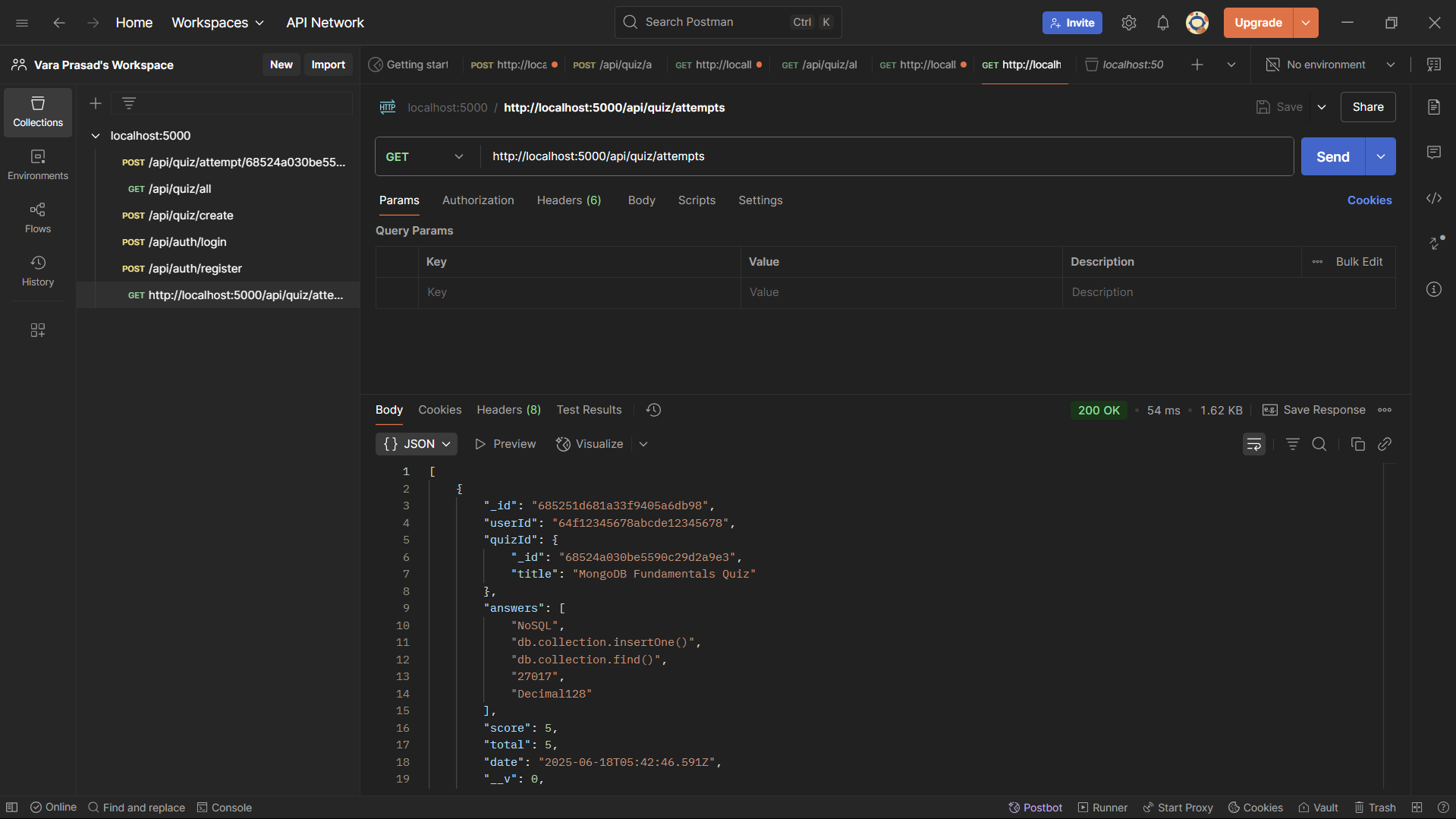
}





## 6. View Past Attempts

Method: GET  
URL: /api/quiz/attempts



## 7.Invalid QuizId

# Sample Test Results Summary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case No. | API Endpoint | Method | Input Provided | Expected Output | Result |
| TC-1 | /register | POST | Email,name,password | User registered | ✅ |
| TC-2 | /login | POST | Email,password | Log in success | ✅ |
| TC-3 | /create | POST | Valid quiz data | 201 Created with quiz JSON | ✅ |
| TC-4 | /all | GET | None | Array of quizzes | ✅ |
| TC-5 | /attempt | POST | Correct answers | Score = full marks | ✅ |
| TC-6 | /attempts | GET | None | List of user attempts | ✅ |
| TC-7 | /attempt | POST | Missing quizId | 404 or error | ✅ |