Meets Specifications

Hi there!
Congratulations on successfully completing this project!
I really commend you for this project. You have demonstrated great skills, and proven knowledge in API design, API documentation, and Test driven-development.
All the best for next your submission
Happy learning and Stay Udacious!

Code Quality & Documentation

✓ **The code adheres to the PEP 8 style guide and follows common best practices, including:**

- **Variable and function names are clear.**
- **Endpoints are logically named.**
- **Code is commented appropriately.**
- **The README file includes detailed instructions for scripts to install any project dependencies, and to run the development server.**
- **Secrets are stored as environment variables.**

✓ **README includes:**

- **Instructions for how to install project dependencies and start the project server.**
- **Detailed documentation of API endpoints and expected behavior, using the format taught in the course:**
  - **METHOD Url**
    - **Request parameters**
    - **Response body**

Great job listing API documentation.

✓ **Local files and virtual environment are included in .gitignore file**

Handling HTTP Requests

✓ **RESTful principles are followed throughout the project, including appropriate naming of endpoints, use of HTTP methods GET, POST, and DELETE.**

Routes perform CRUD operations on the psql database

The RESTful principles have been followed throughout the project such as client-server relationship, stateless nature of the server code and uniform interface. The routes perform the CRUD operations on the psql database successfully.Good work getting this correct!

✓ **Complete all TODO flags in `backend/app.py`:**

- **[ ] Endpoint to handle GET requests for questions, including pagination (every 10 questions). This endpoint should return a list of questions, number of total questions, current category, categories.**
- **[ ] Endpoint to handle GET requests for all available categories.**
- **[ ] Endpoint to DELETE question using a question ID.**
- **[ ] Endpoint to POST a new question, which will require the question and answer text, category, and difficulty score.**
- **[ ] Create a POST endpoint to get questions based on category.**
- **[ ] Create a POST endpoint to get questions based on a search term. It should return any questions for whom the search term is a substring of the question.**
- **[ ] Create a POST endpoint to get questions to play the quiz. This endpoint should take category and previous question parameters and return a random questions within the given category, if provided, and that is not one of the previous questions.**

All the endpoints have been completed with the required response bodies upon passing the required requests .Great job!

✓ **Project handles common errors using the `@app.errorhandler` decorator function to format an API friendly JSON error response**

Passes all provided tests related to error handling

All the common errors have been handled in the code including the client side and server side errors. You may have a look at the Werkzeug HTTPException module(https://werkzeug.palletsprojects.com/en/1.0.x/exceptions/) to handle the errors in a leaner way.

API Testing & Documentation

✓ **Import and utilize unittest library to test each endpoint for expected success and error behavior. Each endpoint should have at one test for the expected behavior and tests for error handling if applicable.**

The unit test library has been used to test for each endpoint for the success and failure cases. Good work!

✓ **Project includes tests to ensure CRUD operations are successful and persist accurately in the database for GET, POST, PUT and DELETE HTTP requests.**

All the tests pass signifying successful CRUD operations using the relevant HTTP requests on the test database.Great job!