Trivia API

REVIEW

CODE REVIEW 9

HISTORY

Meets Specifications

Congratulations 👏✨✨

Dear Learner,

You have passed the project "**Trivia API**" with flying colors.🎯
This is an amazing piece of work you have done here.
It is nice to see you have taken extra efforts to modify the UI for a better user experience.

I went through all your modules and it clearly demonstrates all the functionalities.
The way you have implemented your logic and organized your code was awesome. Great job.
I really appreciate your hard work and dedication to this project.

Keep Learning and keep doing the good work.
Good luck !!!

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.

- 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.

- 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. ❌

Recommendation

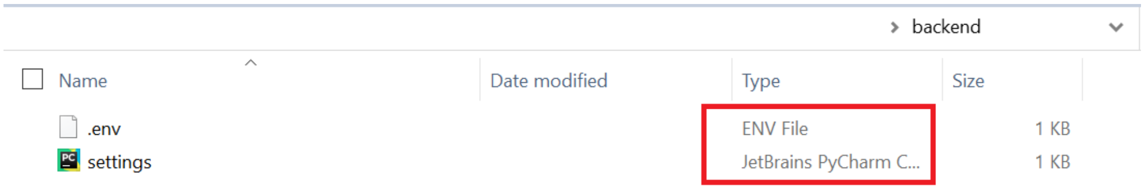
The secrets like the "DB hostname" and "password" in the models.py as well as in test_flaskr.py should be used from environment variables as these are sensitive data.

How to implement

- Install the latest version with:

```
pip install -U python-dotenv
```

- Create a file named .env in the existing directory (i.e in the /backend directory). Also, create a python module named settings.py



```
> backend

Name            Date modified    Type                 Size
.env                             ENV File             1 KB
settings                         JetBrains PyCharm C...  1 KB
```

- You can mention all the sensitive information into your .env file
  for example:-

```
DB_NAME='trivia'
DB_USER='postgres'
DB_PASSWORD='testpass'
```

- Write code to read data from .env file in the settings.py

```
from dotenv import load_dotenv
import os
load_dotenv()
DB_NAME = os.environ.get("DB_NAME")
DB_USER=os.environ.get("DB_USER")
DB_PASSWORD = os.environ.get("DB_PASSWORD")
```

- Finally in **/backend/models.py** and **/backend/test_flaskr.py** you can access them by importing the settings.py module and use them

```
from settings import DB_NAME, DB_USER, DB_PASSWORD
```

Resources

- To store and use the secret variables, it is advised to use the dotenv package.

- To understand the basics of environment variables, kindly follow the adjacent screencast:- 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

Awesome, your README.md file is beautifully documented mentioning about all the request parameters and response body.

Resources

- Anatomy of a README
- python-docstrings

✓

Local files and virtual environment are included in .gitignore file

Nice, Local files and virtual environment are included in .gitignore file

Resources

To know more about .gitignore you can refer to link

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

RESTful principles are followed throughout the project,

- ✓ Including appropriate naming of endpoints
- ✓ Use of HTTP methods GET, POST, and DELETE.

✓

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.

Awesome, brilliant effort here. 👏👏

- [✓] GET questions
- [✓] GET categories
- [✓] DELETE question
- [✓] POST new question
- [✓] POST get questions by category
- [✓] POST get questions based on a search term
- [✓] POST get quiz question

✓

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

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

Further reading

Error handing in Flask

API Testing & Documentation

✓

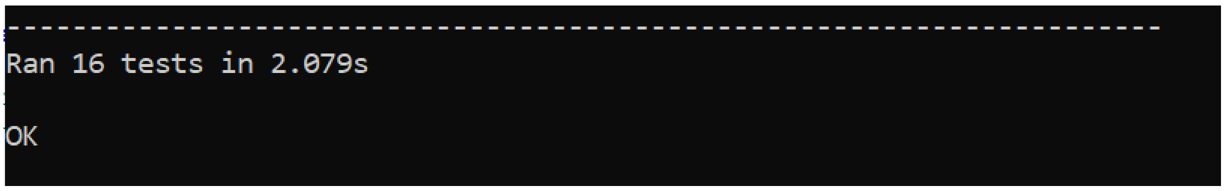
Import and utilize unittest library to test each endpoint for expected success and error behavior. Each endpoint should have at one test for the expected behavior and tests for error handling if applicable.

- ✓ Utilized unittest library to test each endpoint for expected success and error behavior.
- ✓ Each endpoint have at one test for the expected behavior and tests for error handling.

✓

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

Project includes tests to ensure CRUD operations are successful. ✅



```
----------------------------------------------------------------------
Ran 16 tests in 2.079s

OK
```

DOWNLOAD PROJECT

9 CODE REVIEW COMMENTS ›

RETURN TO PATH

Rate this review

START