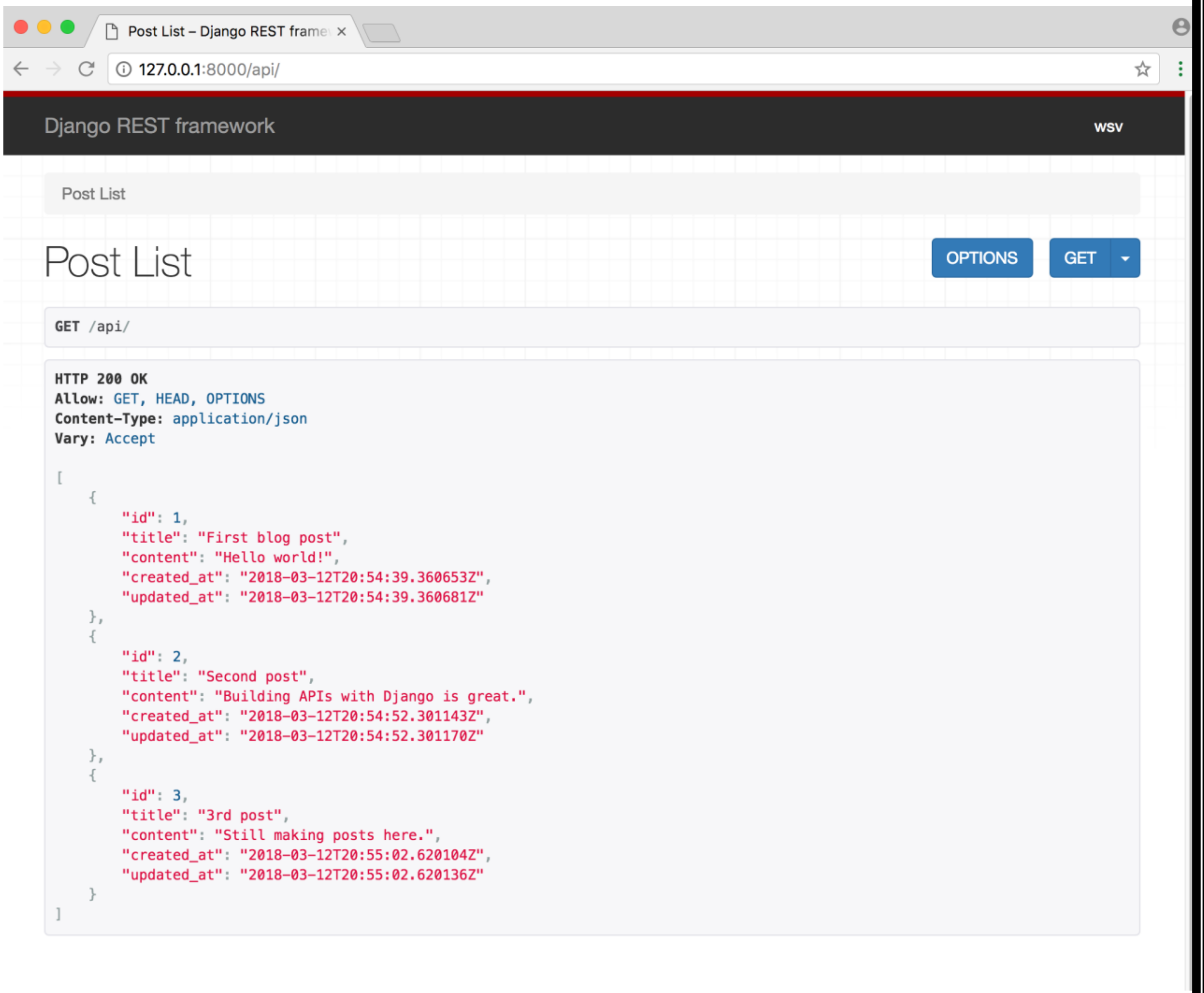
Rest Framework – API
Python
Assessment Test

- **Write a project to demonstrate blog app api using of Django Rest framework**
- **Prepare demonstration of Python blog app api under software development principles and follow coding protocols**
- **The project was built in a MVT concept create an effective interactive interface.**
- **In this application Perform all operations under function based views**
- **Create a comment model in the database for the comment application that the Django ORM will manage.**
    - **Create api which return below data in json format**



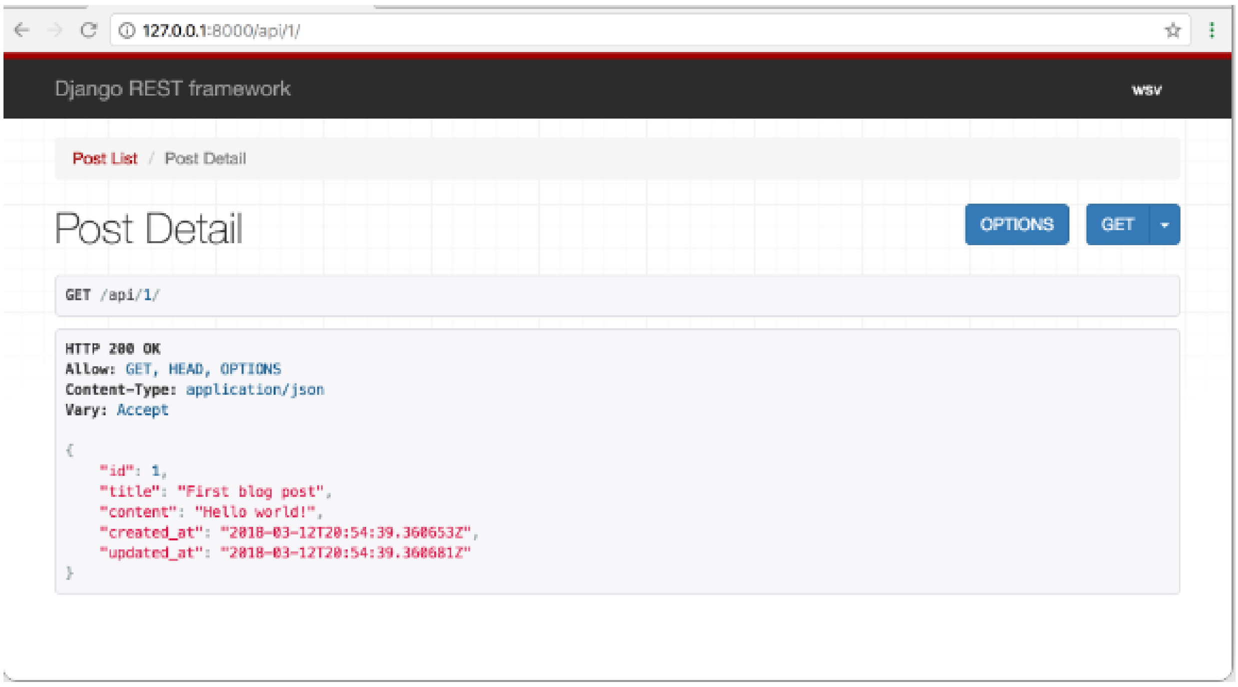
```
GET /api/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "title": "First blog post",
        "content": "Hello world!",
        "created_at": "2018-03-12T20:54:39.360653Z",
        "updated_at": "2018-03-12T20:54:39.360681Z"
    },
    {
        "id": 2,
        "title": "Second post",
        "content": "Building APIs with Django is great.",
        "created_at": "2018-03-12T20:54:52.301143Z",
        "updated_at": "2018-03-12T20:54:52.301170Z"
    },
    {
        "id": 3,
        "title": "3rd post",
        "content": "Still making posts here.",
        "created_at": "2018-03-12T20:55:02.620104Z",
        "updated_at": "2018-03-12T20:55:02.620136Z"
    }
]
```

- **Make sure all api data return in json format – create separate file of json serializes**

- **These are a few key options for a REST API request:**

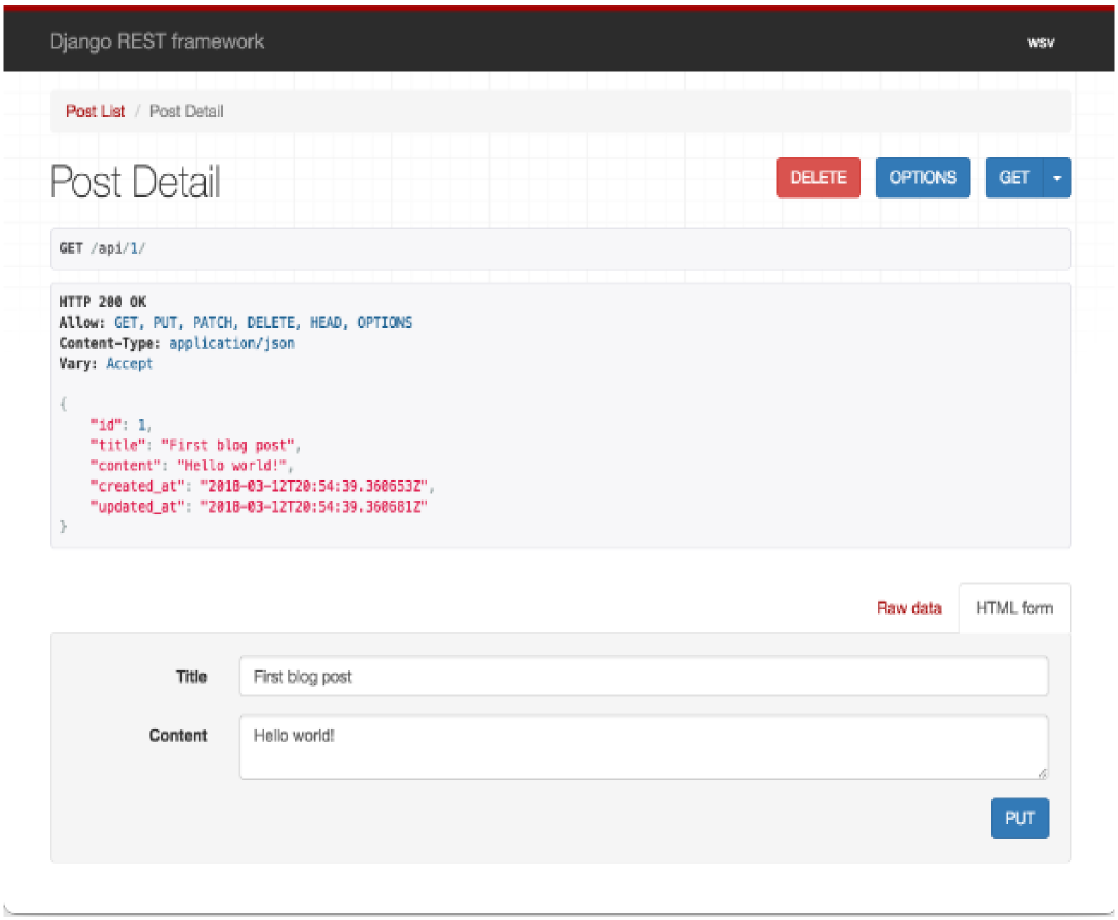
- **GET — the most common option, returns some data from the API based on the endpoint you visit.**

Now go to http://127.0.0.1:8000/api/1/ and you'll see only the data for the first entry.



```
GET /api/1/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 1,
    "title": "First blog post",
    "content": "Hello world!",
    "created_at": "2018-03-12T20:54:39.360653Z",
    "updated_at": "2018-03-12T20:54:39.360681Z"
}
```

- **Cover not just reading/getting content but the full CRUD syntax.**



Post Detail

```
GET /api/1/
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 1,
    "title": "First blog post",
    "content": "Hello world!",
    "created_at": "2018-03-12T20:54:39.360653Z",
    "updated_at": "2018-03-12T20:54:39.360681Z"
}
```

Title	First blog post
Content	Hello world!

- **Admin can able to add and delete records of this system**
- **Manage proper naming conversion – create appropriate objects name.**
- **create separate file for all business logics and make them reusable - use modules concepts for implements above logic**
- **Make sure code prevent from unexpected exception return to the previous screen and accept all details again.**
- **Make sure Database normalize manage in this project work**
- **Developer needs to test this product before launching it into the market**
- **After completion this project upload it on GitHub**
    - **Upload all features in develop branch after completion all features merge it with main branch**