

Rest API Using Django Rest Framework

1. Statement:

Develop a rest api (Django Rest framework) to fetch details of Student marks record based on his identity number .

2. Technologies/Library/Tools:

- Python
- Django
- Django Rest Framework
- Postman: For Testing API

3. Screenshots:

Marks record of All student

The screenshot shows a Postman interface for a REST client. The request is a GET to `http://127.0.0.1:8000/student_marks/api/students_record/`. The response is a 200 OK status with a response time of 308 ms and a size of 498 B. The response body is a JSON array containing two student records.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 [{"created": "2020-06-30T10:56:32.481145Z",
2   "unique_id": 1,
3   "data_structure": 50,
4   "database": 60,
5   "web_technology": 70,
6   "cloud": 90,
7   "data_science": 80},
8  {"created": "2020-06-30T12:19:01.730711Z",
9   "unique_id": 2,
10  "data_structure": 90,
11  "database": 80,
12  "web_technology": 95,
13  "cloud": 70,
14  "data_science": 80}]
```

Student Record

Student Record

GET /student_marks/api/students_record/

HTTP 200 OK

Allow: OPTIONS, GET

Content-Type: application/json

Vary: Accept

```
[
  {
    "created": "2020-06-30T10:56:32.481145Z",
    "unique_id": 1,
    "data_structure": 50,
    "database": 60,
    "web_technology": 70,
    "cloud": 90,
    "data_science": 80
  },
  {
    "created": "2020-06-30T12:19:01.730711Z",
    "unique_id": 2,
    "data_structure": 90,
    "database": 80,
    "web_technology": 95,
    "cloud": 70,
    "data_science": 80
  }
]
```

Student Marks record on the basis of unique ID

GET

http://127.0.0.1:8000/student_marks/api/students_record/1

Send

Params

Authorization

Headers (7)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies (1)

Headers (7)

Test Results

Status: 200 OK

Time: 232 ms

Size: 357 B

Save

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "created": "2020-06-30T10:56:32.481145Z",
3   "unique_id": 1,
4   "data_structure": 50,
5   "database": 60,
6   "web_technology": 70,
7   "cloud": 90,
8   "data_science": 80
9 }
```

Student Record / Student Record By Prn

Student Record By Prn

GET /student_marks/api/students_record/1/

HTTP 200 OK

Allow: OPTIONS, GET

Content-Type: application/json

Vary: Accept

```
{
  "created": "2020-06-30T10:56:32.481145Z",
  "unique_id": 1,
  "data_structure": 50,
  "database": 60,
  "web_technology": 70,
  "cloud": 90,
  "data_science": 80
}
```

Request Limiting Mechanism

```
REST_FRAMEWORK = {
    'DEFAULT_THROTTLE_CLASSES': [
        'rest_framework.throttling.AnonRateThrottle',
        'rest_framework.throttling.UserRateThrottle'
    ],
    'DEFAULT_THROTTLE_RATES': {
        'anon': '10/min',
        'user': '10/min'
    }
}
```

Limitation of request is 10 per minute if it goes beyond that the response is like given below.

The screenshot shows a REST client interface with a GET request to `http://127.0.0.1:8000/student_marks/api/students_record/2`. The response status is 429 Too Many Requests. The response body in JSON format is:

```
{
  "detail": "Request was throttled. Expected available in 44 seconds."
}
```

The interface includes tabs for Params, Authorization, Headers (7), Body, Pre-request Script, Tests, and Settings. The Body tab is selected, showing the JSON response. The status bar indicates the status, time (24 ms), size (319 B), and a Save button.

The total requests to the service

The screenshot shows the Django administration interface for the `request_counters` app. The page title is "Django administration". The breadcrumb trail is "Home > Studentmarksapi > Request_counters > request_counter object (1)". The page content is titled "Change request_counter".

The form has two fields:

- Service name:**
- Counter req:**

At the bottom of the form is a red button labeled "Delete".

