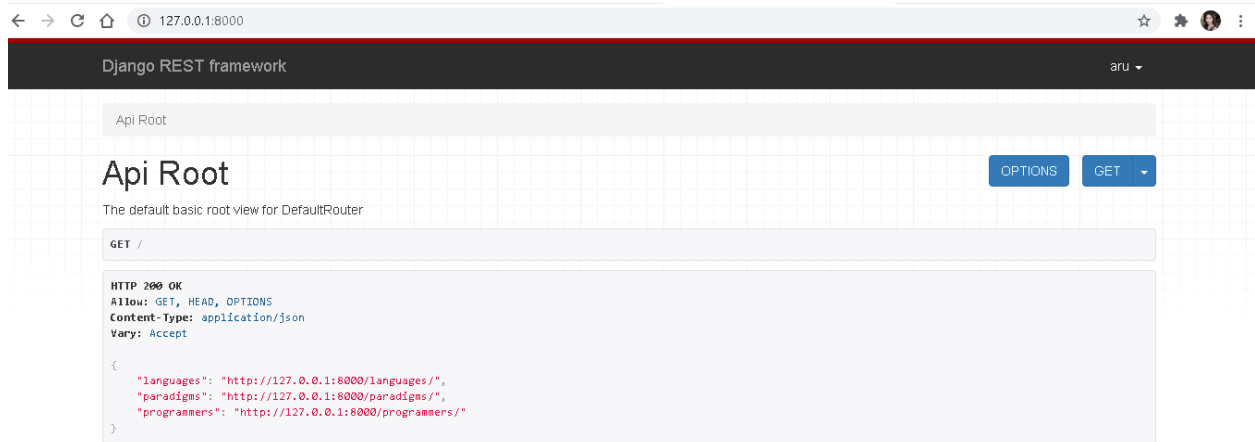
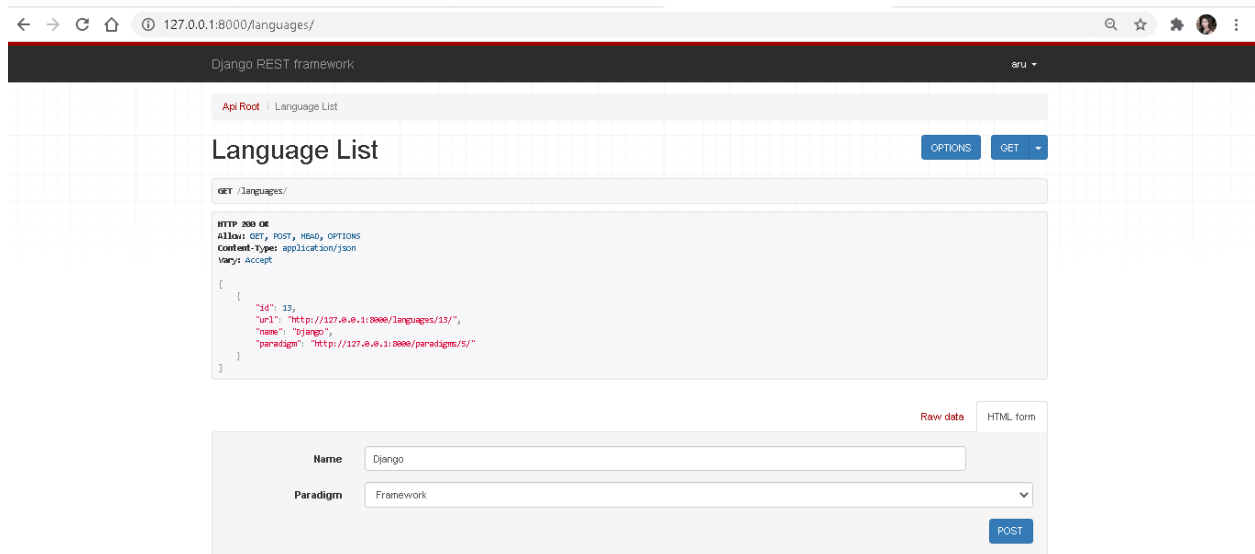


Test the CRUD with APIs



1. Create a new Language using POST /languages Api



Activate Windows

2. Retrieve all Languages using GET /languages Api

The screenshot shows the Django REST framework API browser interface. The browser address bar displays `127.0.0.1:8000/languages/13/`. The page title is "Language Instance". The breadcrumb navigation shows "Api Root / Language List / Language Instance". The main content area displays the HTTP status "200 OK" and the allowed methods "GET, PUT, PATCH, DELETE, HEAD, OPTIONS". The content type is "application/json" and the vary header is "Accept". The JSON response is shown in a code block:

```
{
  "id": 13,
  "url": "http://127.0.0.1:8000/languages/13/",
  "name": "Django",
  "paradigm": "http://127.0.0.1:8000/paradigm/5/"
}
```

Below the JSON response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is active, showing a form with two fields: "Name" (with the value "Django") and "Paradigm" (with a dropdown menu showing "Framework"). A "PUT" button is located at the bottom right of the form.

Activate Windows
Go to Settings to activate Windows.

3. Create all Programmer using POST /languages Api

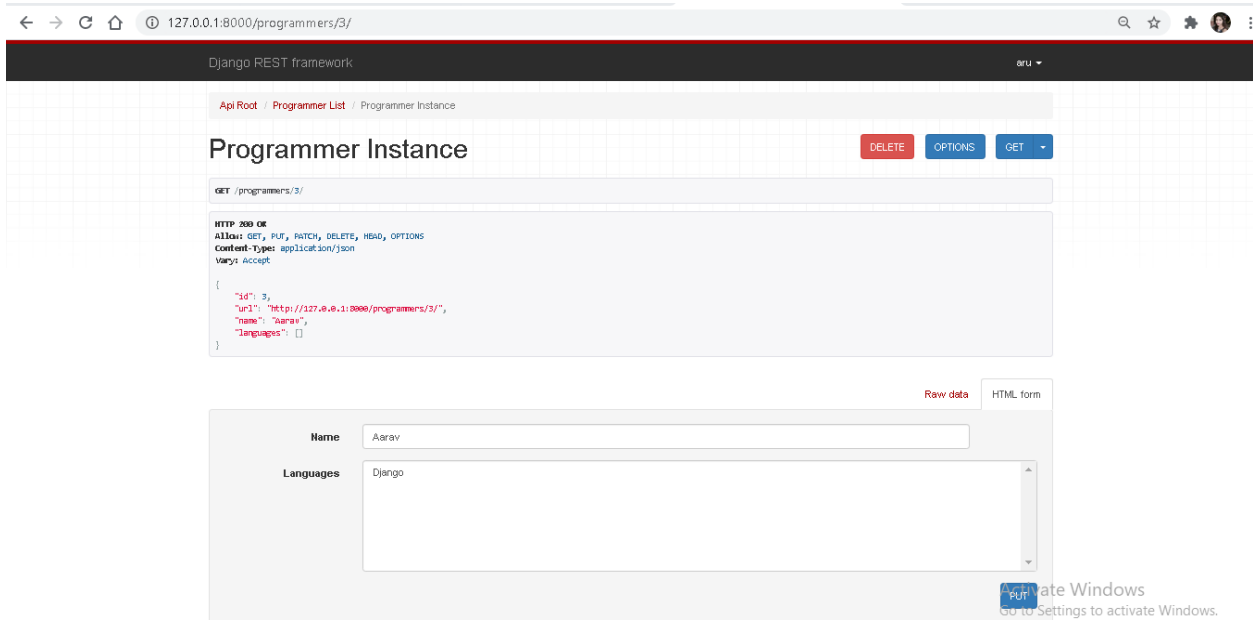
The screenshot shows the Django REST framework API browser interface. The browser address bar displays `127.0.0.1:8000/programmers/`. The page title is "Programmer List". The breadcrumb navigation shows "Api Root / Programmer List". The main content area displays the HTTP status "200 OK" and the allowed methods "GET, POST, HEAD, OPTIONS". The content type is "application/json" and the vary header is "Accept". The JSON response is shown in a code block:

```
[
  {
    "id": 5,
    "url": "http://127.0.0.1:8000/programmers/5/",
    "name": "Anaya",
    "languages": []
  }
]
```

Below the JSON response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is active, showing a form with two fields: "Name" (with an empty text input) and "Languages" (with a text input containing "Django"). A "POST" button is located at the bottom right of the form.

Activate Windows
Go to Settings to activate Windows.

4. Programmer Instance using GET /languages Api

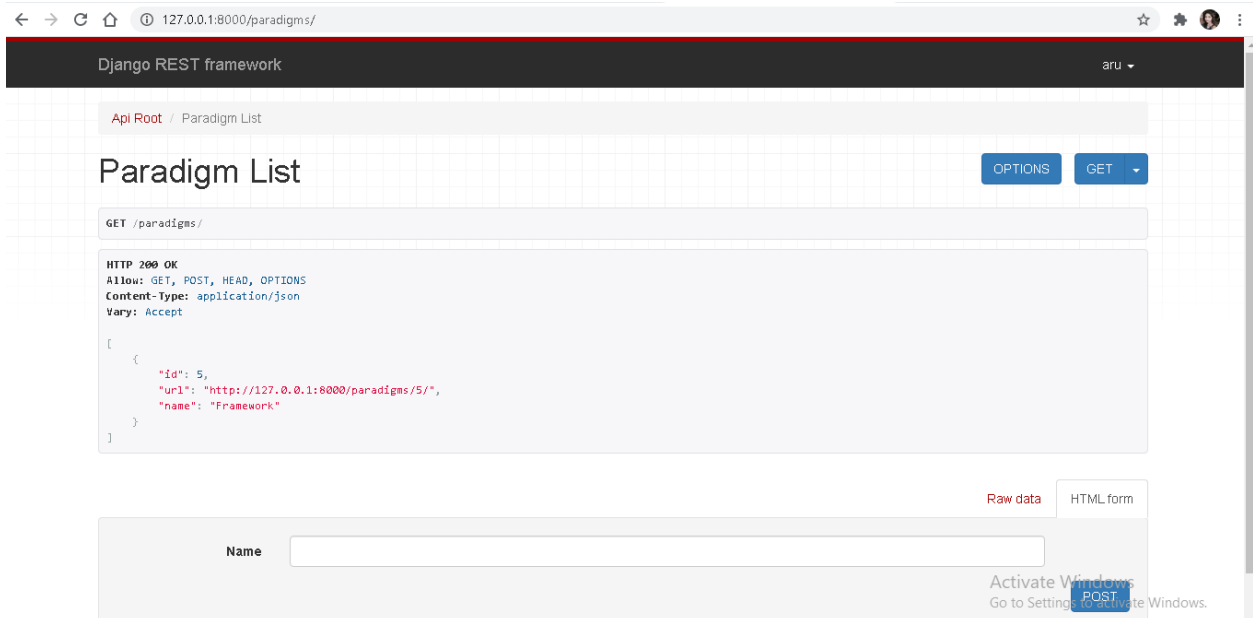


The screenshot shows the Django REST framework API browser interface. The breadcrumb trail is "Api Root / Programmer List / Programmer Instance". The main heading is "Programmer Instance". There are buttons for "DELETE", "OPTIONS", and "GET". The selected method is "GET /programmers/3/". The response is "HTTP 200 OK" with "Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS", "Content-Type: application/json", and "Vary: Accept". The JSON response is:

```
{  "id": 3,  "url": "http://127.0.0.1:8000/programmers/3/",  "name": "Asrav",  "languages": []}
```

 Below the response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is active, showing a form with a "Name" field containing "Asrav" and a "Languages" field containing "Django".

5. Create all Paradigm using POST /languages Api



The screenshot shows the Django REST framework API browser interface. The breadcrumb trail is "Api Root / Paradigm List". The main heading is "Paradigm List". There are buttons for "OPTIONS" and "GET". The selected method is "GET /paradigms/". The response is "HTTP 200 OK" with "Allow: GET, POST, HEAD, OPTIONS", "Content-Type: application/json", and "Vary: Accept". The JSON response is:

```
[  {    "id": 5,    "url": "http://127.0.0.1:8000/paradigms/5/",    "name": "Framework"  }]
```

 Below the response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is active, showing a form with a "Name" field.

6. Update/Delete all Paradigm using PUT / DELETE /languages Api

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/paradigms/5/?format=api`. The page title is "Django REST framework". The breadcrumb navigation shows "Api Root / Paradigm List / Paradigm Instance". The main heading is "Paradigm Instance". To the right of the heading are buttons for "DELETE", "OPTIONS", and "GET". Below the heading, the HTTP method and URL are shown: "GET /paradigms/5/?format=api". The response status is "HTTP 200 OK". The response headers are: "Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS", "Content-Type: application/json", and "Vary: Accept". The response body is a JSON object:

```
{  "id": 5,  "url": "http://127.0.0.1:8000/paradigms/5/?format=api",  "name": "Framework"}
```

. Below the response, there are tabs for "Raw data" and "HTML form". The "HTML form" tab is active, showing a form with a label "Name" and a text input field containing "Framework". To the right of the input field is a "PUT" button. At the bottom right of the page, there is a watermark that says "Activate Windows Go to Settings to activate Windows."

Conclusion

This project aim, how to create Python 3/Django CRUD MySQL example Django Rest Framework for Rest Apis. We also know way to connect a Django application with a MySQL database, create a Django Model, migrate it to a database, write the Views and define Url patterns for handling all CRUD operations.