

Nama : Fajar Alfiantino

NPM : 202310072

Kelas : TI-20-A Pagi

Mata Kuliah : Microservices

Tabel ProgramStudy

Create:

The screenshot shows two applications side-by-side. On the left is a database client (DBeaver) connected to a MySQL database. It displays the 'programstudy' table with columns: id, code, departement_id, and descri. The table is currently empty. On the right is an API client (Postman) showing a POST request to 'localhost:8080/api/programstudy'. The request body is a JSON object:

```
{  "code": "TI-PA",  "name": "Teknologi Informasi",  "description": "SI",  "departement_id": 1,  "faculty_id": 1,  "program_id": 1,  "is_active": 1}
```

. The response is a 200 OK status with a JSON body:

```
{  "id": 1,  "name": "Teknologi Informasi",  "description": "SI",  "code": "TI-PA",  "program_id": 1,  "faculty_id": 1,  "departement_id": 1,  "is_active": true}
```

Read:

The screenshot shows two applications side-by-side. On the left is a database client (DBeaver) connected to a MySQL database. It displays the 'programstudy' table with columns: departement_id, id, code, departement_id, is_active, name, and program_id. The table contains two rows:

departement_id	id	code	departement_id	is_active	name	program_id
1	S1		1	1	Teknologi Informasi	1
1	S1		1	1	Sistem Informasi	2

. On the right is an API client (Postman) showing a GET request to 'localhost:8080/api/programstudy'. The response is a 200 OK status with a JSON body:

```
{  "id": 1,  "name": "Teknologi Informasi",  "description": "SI",  "code": "TI-PA",  "program_id": 1,  "faculty_id": 1,  "departement_id": 1,  "is_active": true}
```

Read ById:

The screenshot displays a web application interface on the left and an API client interface on the right.

Web Application Interface:

- URL: localhost:127.0.0.1
- Database: fak_apr_sandbox
- Table: programstudy
- Query: `SELECT * FROM "programstudy"`
- Results: Showing rows 0 - 1 (2 total, Query took 0.0013 seconds)
- Table Data:

departement_id	description	faculty_id	is_active	name	program_id
1	S1	1	1	Teknologi Informasi	1
1	S1	1	1	Sistem Informasi	2

API Client Interface:

- Collection: api-ibk
- Environment: Environments
- Mock Servers: Mock Servers
- Monitors: Monitors
- Flows: Flows
- History: History
- Request: GET localhost:8080/api/programstudy/2
- Response: 200 OK, 177 ms, 298
- Body (Pretty):

```
{
  "id": 2,
  "name": "Sistem Informasi",
  "description": "SI",
  "code": "SI-PA",
  "program_id": 2,
  "faculty_id": 1,
  "departement_id": 1,
  "is_active": true
}
```

Update:

The screenshot displays a web application interface on the left and an API client interface on the right.

Web Application Interface:

- URL: localhost:127.0.0.1
- Database: fak_apr_sandbox
- Table: programstudy
- Query: `SELECT * FROM "programstudy"`
- Results: Showing rows 0 - 1 (2 total, Query took 0.0013 seconds)
- Table Data:

departement_id	description	faculty_id	is_active	name	program_id
1	S1	1	1	Teknologi Informasi	1
1	S1	1	1	Sistem Informasi	2

API Client Interface:

- Collection: api-ibk
- Environment: Environments
- Mock Servers: Mock Servers
- Monitors: Monitors
- Flows: Flows
- History: History
- Request: PUT localhost:8080/api/programstudy
- Response: 200 OK, 209 ms, 298
- Body (Pretty):

```
{
  "id": 2,
  "name": "SI-PA",
  "description": "SI",
  "code": "SI-PA",
  "program_id": 2,
  "faculty_id": 1,
  "departement_id": 1,
  "is_active": true
}
```

Delete:

The screenshot displays a web application interface on the left and a REST client on the right. The web application shows a table with one row of data. The REST client shows a DELETE request to the endpoint `localhost:8080/api/programstudy/2` with a response body of `1`.

Web Application Interface:

- Showing rows 0 - 0 (1 total, Query took 0.0007 seconds.)
- Query: `SELECT * FROM 'programstudy'`
- Table structure: `id`, `code`, `departement_id`, `description`
- Table data:

id	code	departement_id	description
1	TLPA	1	S1

REST Client:

- Method: DELETE
- URL: `localhost:8080/api/programstudy/2`
- Response Body: `1`

Tabel Students

Create:

The screenshot displays a web application interface on the left and a REST client on the right. The web application shows a table with two rows of data. The REST client shows a POST request to the endpoint `localhost:8080/api/students` with a response body of `1`.

Web Application Interface:

- Showing rows 0 - 1 (2 total, Query took 0.0021 seconds.)
- Query: `SELECT * FROM 'students'`
- Table structure: `id`, `departement_id`, `firstname`, `lastname`
- Table data:

id	departement_id	firstname	lastname
1	1	Fajar	Alfantino
3	1	Keisha	Salsabila

REST Client:

- Method: POST
- URL: `localhost:8080/api/students`
- Request Body (JSON):

```
{  "department_id": 1,  "firstname": "Keisha",  "lastname": "Salsabila",  "middlename": "Amalia",  "npm": "282338915",  "program_id": 1}
```
- Response Body: `1`

Read:

The screenshot displays a web application interface on the left and a REST client on the right. The web application shows a table of students with columns: id, departement_id, firstname, and lastname. The table contains two rows: (1, Fajar, Alfiantino) and (3, Kaisha, Nabila). The REST client shows a GET request to localhost:8080/api/students, returning a JSON array of two student objects.

id	departement_id	firstname	lastname
1	1	Fajar	Alfiantino
3	1	Kaisha	Nabila

```
1 {  
2   "npm": "282318072",  
3   "firstname": "Fajar",  
4   "middlename": null,  
5   "lastname": "Alfiantino",  
6   "program_id": 1,  
7   "departement_id": 1,  
8   "id": 1  
9 }  
10 [  
11   {  
12     "npm": "282318072",  
13     "firstname": "Kaisha",  
14     "middlename": "Amalia",  
15     "lastname": "Nabila",  
16     "program_id": 1,  
17     "departement_id": 1,  
18     "id": 3  
19   }  
20 ]
```

Read ById:

The screenshot displays the same web application interface on the left, but the REST client on the right shows a GET request to localhost:8080/api/students/1. The response is a single JSON object representing the student with id 1.

id	departement_id	firstname	lastname
1	1	Fajar	Alfiantino
3	1	Kaisha	Nabila

```
1 {  
2   "npm": "282318072",  
3   "firstname": "Fajar",  
4   "middlename": null,  
5   "lastname": "Alfiantino",  
6   "program_id": 1,  
7   "departement_id": 1,  
8   "id": 1  
9 }
```

Update:

The screenshot shows a web browser on the left and the Postman API client on the right. The browser displays a table of students with columns: id, departement_id, firstname, and lastname. The table contains two rows: (1, 1, Fajar, Allantino) and (3, 1, Keisha, Nabila). The Postman interface shows a PUT request to localhost:8080/api/students/. The request body is a JSON object: {"id": 3, "departement_id": 1, "firstname": "Keisha", "lastname": "Nabila", "middlename": "Amalia", "name": "2002310015", "program_id": 1}. The response status is 200 OK.

id	departement_id	firstname	lastname
1	1	Fajar	Allantino
3	1	Keisha	Nabila

```
{
  "id": 3,
  "departement_id": 1,
  "firstname": "Keisha",
  "lastname": "Nabila",
  "middlename": "Amalia",
  "name": "2002310015",
  "program_id": 1
}
```

Delete:

The screenshot shows a web browser on the left and the Postman API client on the right. The browser displays a table of students with columns: id, departement_id, firstname, and lastname. The table contains two rows: (1, 1, Fajar, Allantino) and (3, 1, Keisha, Nabila). The Postman interface shows a DELETE request to localhost:8080/api/students/3. The response status is 200 OK.

id	departement_id	firstname	lastname
1	1	Fajar	Allantino
3	1	Keisha	Nabila