

# RestoranKu - API

## RestoranKu API

Below is the list of APIs used by frontend side, to get list of restaurants, create, edit, or delete a restaurant.

---

### POST Login

`http://127.0.0.1:8000/api/login`

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1` ).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

#### Body urlencoded

---

email	admin@restoranku.com
-------	----------------------

password	password
----------	----------

---

### POST Register

`http://127.0.0.1:8000/api/register`

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1` ).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

#### HEADERS

---

Accept	application/json
--------	------------------

**Body** urlencoded

---

<b>email</b>	admin3@restoranku.com
<b>password</b>	password
<b>password_confirmation</b>	password
<b>name</b>	admin3

---

## GET Get List Restoran

http://127.0.0.1:8000/api/restaurants

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1` ).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

---

## GET Get data

http://127.0.0.1:8000/api/restaurants/2

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1` ).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

---

## POST Post data



http://127.0.0.1:8000/api/restaurants

This is a POST request, submitting data to an API via the request body. This request submits JSON data, and the data is reflected in the response.

A successful POST request typically returns a `200 OK` or `201 Created` response code.

## AUTHORIZATION Bearer Token

---

Token

<token>

**Body** raw (json)

---

json

```
{
  "name": "Add your name in the body",
  "address": "address",
  "phone_number": "081234567",
  "note": "note",
  "open_times": [
    {"day_start": "1", "day_end": "7", "time_start": "07:00", "time_end": "19:00"}
  ]
}
```

---

## PUT Update data



http://127.0.0.1:8000/api/restaurants/6

This is a PUT request and it is used to overwrite an existing piece of data. For instance, after you create an entity with a POST request, you may want to modify that later. You can do that using a PUT request. You typically identify the entity being updated by including an identifier in the URL (eg. `id=1`).

A successful PUT request typically returns a `200 OK`, `201 Created`, or `204 No Content` response code.

## AUTHORIZATION Bearer Token

---

Token

<token>

**Body** raw (json)

---

json

```
{
  "name": "Add your name in the body",
  "address": "address",
  "phone_number": "081234567",
  "note": "note",
  "open_times": [
    {"day_start": "1", "day_end": "7", "time_start": "07:00", "time_end": "19:00"}
  ]
}
```

```
}
```

---

## DELETE Delete data



`http://127.0.0.1:8000/api/restaurants/5`

This is a DELETE request, and it is used to delete data that was previously created via a POST request. You typically identify the entity being updated by including an identifier in the URL (eg. `id=1`).

A successful DELETE request typically returns a `200 OK`, `202 Accepted`, or `204 No Content` response code.

### AUTHORIZATION Bearer Token

---

Token	<token>
-------	---------