

Document Technical Dating Apps

Author : Moh. Radyatama Suryana

Email : mohradyatama24@gmail.com

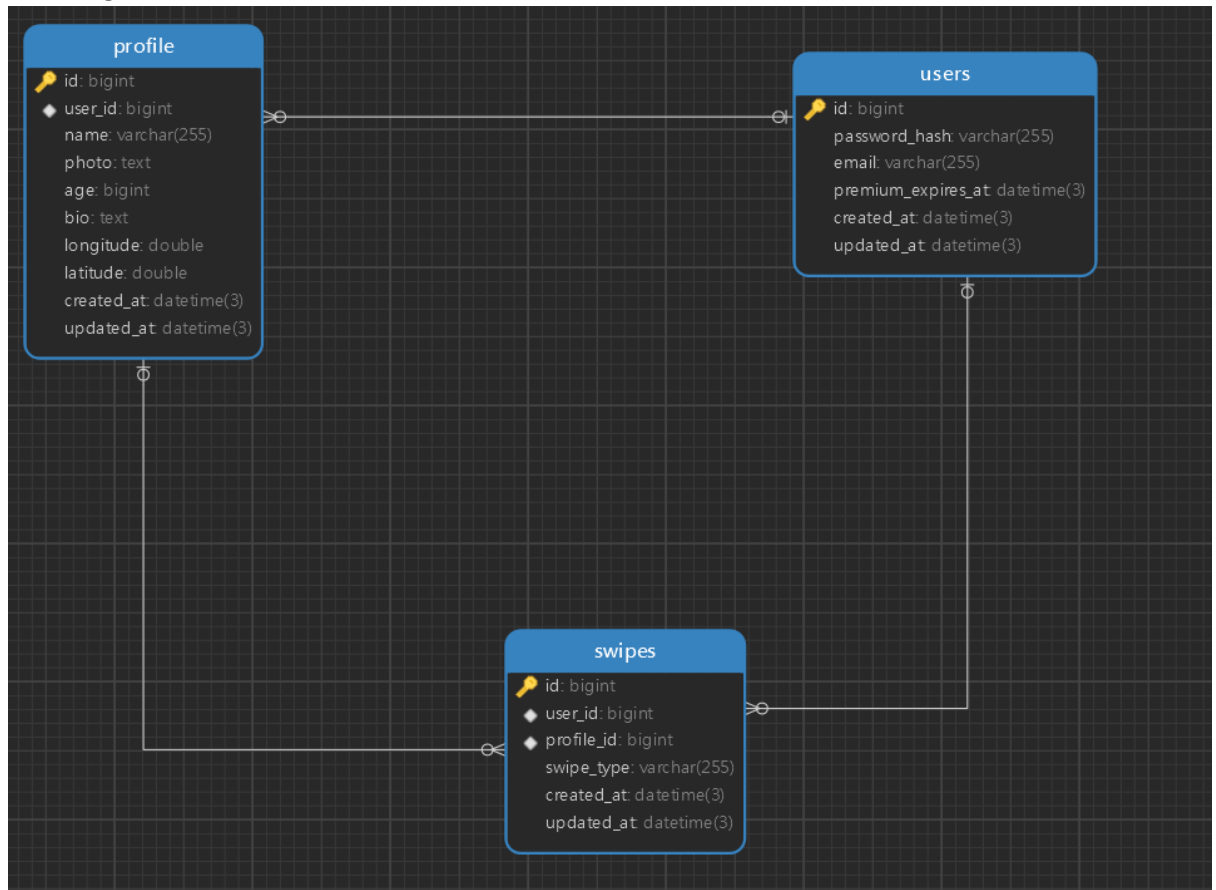
1. Url github repository: <https://github.com/radyatamaa/dating-apps-api>
2. List Of Functionality :
 - a. User Register
Endpoint: POST /api/v1/user/register
Description: This endpoint allows new users to sign up for the app. The user provides a name, age, bio, photo, email, and password. The password is hashed before being stored in the database for security.
 - b. User Login
Endpoint: POST /api/v1/user/login
Description: This endpoint allows users to log in by providing their email and password. If the credentials are valid, a JWT token is returned for authenticated access to other endpoints.
 - c. Purchasing Premium
Endpoint: POST /api/v1/user/purchase-premium
Description: This endpoint allows users to purchase premium packages that unlock additional features (e.g., no swipe quota, verified label). The purchase is processed and the user's premium status is updated, needed ` Authorization: Bearer <JWT_TOKEN>` for the update status premium.
 - d. Update Location of Current User
Endpoint: PUT /api/v1/profile/location
Description: This endpoint allows users to update their current location. This location data can be used to show nearby profiles, needed ` Authorization: Bearer <JWT_TOKEN>` for the update current location.
 - e. Profile Viewing
Endpoint: GET /api/v1/profile
Description: This endpoint allows users to view a list of profiles with pagination and longitude latitude request for nearby profile, they can swipe on. It enforces the daily swipe limit (10 profiles per day for non-premium users) and ensures the same profile does not appear twice in the same day (if pass the profile will doesn't appear at the same day , if liked will doesn't appear again), needed ` Authorization: Bearer <JWT_TOKEN>` for the profile viewing.
 - f. Swiping
Endpoint: POST /api/v1/swipe/profile
Description: This endpoint allows users to swipe left (pass) or right (like) on a profile. It records the swipe action and enforces the daily limit for non-premium users (like + pass each profile max 10 profile per day), needed ` Authorization: Bearer <JWT_TOKEN>` for the profile viewing.

3. List of the tech stacks has been used :

- a. Golang : Go is chosen for its high performance and efficiency as a statically typed, compiled language. It excels in handling concurrent operations, making it suitable for applications requiring real-time interactions, like a dating app. Go's built-in support for concurrency using goroutines and channels simplifies managing multiple user interactions simultaneously. Additionally, Go's simple syntax enhances code readability and maintainability, reducing complexity and easing the onboarding process for new developers.
- b. Beego Framework : Beego organizes the codebase in a structured manner, facilitating easier management and scalability. It comes with built-in features such as ORM, session management, and logging, which expedite development and reduce the reliance on additional third-party libraries. Additionally, Beego benefits from a growing community and comprehensive documentation, providing valuable support and best practices, and also least issues instead of others framework like gin/echo base on github.
- c. Clean Architecture : Clean Architecture is adopted to ensure the separation of concerns and maintainability of the codebase. It promotes a clear distinction between the business logic and the infrastructure, making the system more modular and testable. By organizing the code into layers, Clean Architecture facilitates easier updates and scalability, as changes in one layer have minimal impact on others. This approach also enhances the readability and understandability of the code, allowing developers to quickly grasp the system's structure and functionality.
- d. Go Standard Project Layout : The Go Standard Project Layout is chosen to ensure a consistent and well-organized structure for the codebase. This layout follows widely accepted conventions in the Go community, making it easier for developers to navigate and understand the project. By adhering to this standard, the project benefits from improved maintainability, scalability, and collaboration, as developers can quickly familiarize themselves with the project's structure. This layout also facilitates the application of best practices and simplifies the integration of tools and libraries commonly used in the Go ecosystem.
- e. Database MySQL : MySQL is chosen for its reliability, stability, and robustness as a widely-used relational database. It excels at handling structured data and complex queries. MySQL's support for ACID (Atomicity, Consistency, Isolation, Durability) properties ensures data integrity, which is crucial for transactional applications like a dating app. Additionally, MySQL's ability to scale both vertically and horizontally makes it a suitable choice for applications that need to handle increasing amounts of data and users.
- f. Redis Cache : Redis is selected for its exceptional performance as an in-memory data structure store, offering fast read and write operations. This makes it ideal for

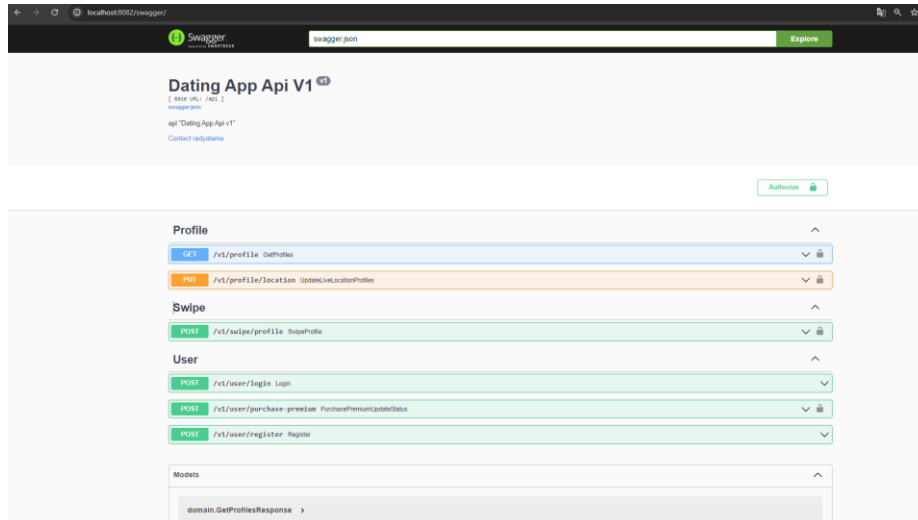
caching frequently accessed data, reducing the load on the database and improving response times. Redis's support for complex data types such as strings, hashes, lists, sets, and sorted sets enables the implementation of various caching strategies. Additionally, Redis is well-suited for session management, thanks to its fast access speeds and ability to handle large volumes of data with low latency. Its Pub/Sub feature further enhances its utility, making it useful for implementing real-time features like notifications and messaging within the app.

4. ERD Diagram :



5. Test:

- a. Open the app swagger <http://localhost:8082/swagger/>



The Accept-Language to manage response language, by default is `english`, for available is `en/id`, en = English, id = Bahasa.

Name	Description
Accept-Language string (header)	lang

Accept-Language

Example for lang id

Curl

```
curl -X 'POST' \
  'http://localhost:8082/api/v1/user/login' \
  -H 'accept: application/json' \
  -H 'Accept-Language: id' \
  -H 'Content-type: application/json' \
  -d '{
    "email": "mohdedytama24@gmail.com",
    "password": "passwords"
  }'
```

Request URL

<http://localhost:8082/api/v1/user/login>

Server response

Code Details

400

Error: Bad Request

Response body

```
{
  "code": "ERROR-API-028",
  "message": "email password salah",
  "data": null,
  "errors": null,
  "request_id": "8d7bf1a1-28e5-4788-aef8-d1249ee15552",
  "timestamp": "2024-06-06 15:34:35"
}
```

Download

b. Register your account

- success

The screenshot displays a REST client interface with a light green background. The top section is a form for constructing an HTTP request. It includes fields for 'Accept-Language' (set to 'lang'), 'photo' (a file named 'profile_face.jpeg'), 'name' (set to 'radyatama'), 'age' (set to '25'), 'bio' (set to 'Software Developer'), 'email' (set to 'mohradyatama24@gmail.com'), and 'password' (set to 'passvord'). Below the form are 'Execute' and 'Clear' buttons. The 'Responses' section shows the result of the request, with a dropdown menu set to 'application/json'. The 'Curl' section contains the equivalent curl command. The 'Request URL' is 'http://localhost:8082/api/v1/user/register'. The 'Server response' section shows a '200' status code. The 'Response body' is a JSON object indicating a successful registration.

Accept-Language
string
(header)
lang
Accept-Language

photo * required
file
(formData)
file
[Pick File] profile_face.jpeg

name * required
string
(formData)
name
radyatama

age * required
integer
(formData)
age
25

bio * required
string
(formData)
bio
Software Developer

email * required
string
(formData)
email
mohradyatama24@gmail.com

password * required
string
(formData)
password
passvord

Execute Clear

Responses Response content type application/json

Curl

```
curl -X 'POST' \
  'http://localhost:8082/api/v1/user/register' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'photo=profile_face.jpeg;type=image/jpeg' \
  -F 'name=radyatama' \
  -F 'age=25' \
  -F 'bio=Software Developer' \
  -F 'email=mohradyatama24@gmail.com' \
  -F 'password=passvord'
```

Request URL

http://localhost:8082/api/v1/user/register

Server response

Code	Details
200	<p>Response body</p> <pre>{ "code": "OK", "message": "operation successfully executed.", "data": null, "version": "v1", "request_id": "799d25cc-58a6-4f62-9675-3448b6d8a7b", "timestamp": "2024-06-06 14:56:40" }</pre>

Response headers

- error user already exist

Curl

```
curl -X 'POST' \
  'http://localhost:8082/api/v1/user/register' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'photo=profile_face.jpg;type=image/jpeg' \
  -F 'name=radystana' \
  -F 'age=25' \
  -F 'bio=Software Developer' \
  -F 'email=mohradystana24@gmail.com' \
  -F 'password=password'
```

Request URL

http://localhost:8082/api/v1/user/register

Server response

Code	Details
400	<p>Error: Bad Request</p> <p>Response body</p> <pre>{ "code": "ERROR-API-004", "message": "Invalid request, errors arise when your request has invalid parameters.", "data": null, "errors": [{ "field": "Email", "message": "Email already exist." }], "request_id": "238e188-897f-452b-8ae4-59f7b9927581", "timestamp": "2024-06-06 14:57:40" }</pre> <p>Response headers</p>

- error validation

Curl

```
curl -X 'POST' \
  'http://localhost:8082/api/v1/user/register' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'photo=profile_face.jpg;type=image/jpeg' \
  -F 'name=radystana' \
  -F 'age=25' \
  -F 'bio=Software Developer' \
  -F 'email=mohradystana24' \
  -F 'password=password'
```

Request URL

http://localhost:8082/api/v1/user/register

Server response

Code	Details
400	<p>Error: Bad Request</p> <p>Response body</p> <pre>{ "code": "ERROR-API-004", "message": "Invalid request, errors arise when your request has invalid parameters.", "data": null, "errors": [{ "field": "Email", "message": "Email invalid format email." }], "request_id": "6883a8e-097f-425d-8d2f-2b9c82acb981", "timestamp": "2024-06-06 14:58:36" }</pre> <p>Response headers</p>

- success

```
Curl
curl -X 'POST' \
  'http://localhost:8082/api/v1/user/login' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "email": "mohvadyas24@gmail.com",
    "password": "passwords"
  }'
```

Request URL

```
http://localhost:8082/api/v1/user/login
```

Server response

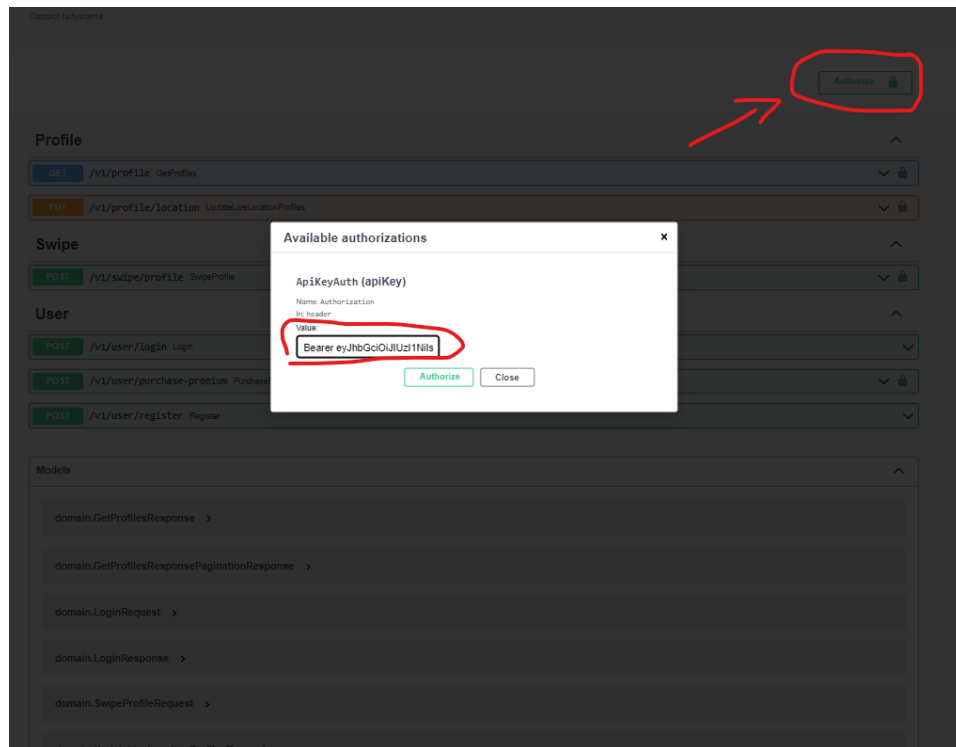
Code	Details
400	Error: Bad Request

Response body

```
{
  "code": "ERROR-API-628",
  "message": "Invalid Email and Password",
  "data": null,
  "errors": null,
  "request_id": "88923885-68ca-4277-a294-116935c926c0",
  "timestamp": "2024-06-06 15:02:49"
}
```

Response headers

- d. Put your token with Bearer <Token> in the Authorize swagger then click Authorize then close



- success

[illegible]

- error token

Curl

```
curl -X 'PUT' \
  'http://localhost:8082/api/v1/profile/location' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "latitude": -6.5241888,
    "longitude": 106.7941888
  }'
```

Request URL

```
http://localhost:8082/api/v1/profile/location
```

Server response

Code

Details

401

Undocumented

Error: Unauthorized

Response body

```
{
  "code": "ERROR-API-018",
  "message": "the token is missing, please filled in the request.",
  "data": null,
  "errors": null,
  "request_id": "3f51271e-2039-4a9b-ac6c-38347ae38df5",
  "timestamp": "2024-06-06 15:09:36"
}
```

Download

Response headers

```
content-length: 225
content-type: application/json; charset=utf-8
date: Thu, 06 Jun 2024 08:09:36 GMT
x-request-id: 3f51271e-2039-4a9b-ac6c-38347ae38df5
```

Responses

- With pagination

- ```
Curl curl -X 'GET' \
 http://localhost:8082/api/v1/profile?pageSize=5&page=1&latitude=-6.5241088&longitude=106.7941888 \
 -H 'accept: application/json' \
 -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFPbC6ImVhZThhdGFtYTIQOGdYdiIsImNvbmV5SisIdm4iOiJlbnR5cyJyYXNpdjE3MTI0IjBzb2NhbgVhc3Q6ODAwMisiImpdaSI6IjE3MTIwZjE3MTI0IiwiaWF0IjoxNjc3OTYxMDk5fQ.joxnzE3NjYyOYWLCPjc3MI01sb2NhhGvc3Q6ODAwMisiImpdaSI6IjE3MTIwZjE3MTI0IiwiaWF0IjoxNjc3OTYxMDk5fQ
```

Request URL  
http://localhost:8082/api/v1/profile?pageSize=5&page=1&latitude=-6.5241088&longitude=106.7941888

Server response

| Code | Details                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 200  | <p>Response body</p> <pre>{   "code": "OK",   "message": "operation successfully executed.",   "data": {     "data": [       {         "id": 29,         "name": "Okzts6IKNs6ddF",         "photo": "https://fastly.picsum.photos/id/660/536/354.jpg?hmac=r1c36NcjocyX8aPwVv-b2M6nmTjnUV56Y2YKxmckG4",         "age": 21,         "bio": "dummy",         "verified": false,         "distance": "2532km"       },       {         "id": 20,         "name": "69bClG025dgEivX",         "photo": "https://fastly.picsum.photos/id/660/536/354.jpg?hmac=r1c36NcjocyX8aPwVv-b2M6nmTjnUV56Y2YKxmckG4",         "age": 21,         "bio": "dummy",         "verified": false,         "distance": "7971km"       },       {         "id": 17,         "name": "8NKKcQBLaFVVixi",         "photo": "https://fastly.picsum.photos/id/660/536/354.jpg?hmac=r1c36NcjocyX8aPwVv-b2M6nmTjnUV56Y2YKxmckG4",         "age": 21,</pre> |

- Success

[illegible]

- Error validation

[illegible]

- Error swipe pass/like quota has run out max 10 profile (like + pass) per day

```
Curl curl -X POST \
'http://localhost:8082/api/v1/swipe/profile' \
-H 'accept: application/json' \
-H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWpbC1EmlvaChZlhdGFtYTYTCmQ6YWJsZWV5LS1mV4cCI6NTczNmNwLmF0eSjxwNzE3MjYyMDYVLCJpc3MlOjlsb2hhbG9vc3Q6ODAAAHlicImpbaS16IjE3MTc2NjJ9' \
-H 'Content-Type: application/json' \
-d '{
 "profile_id": 8,
 "swipe_type": "LIKE"
}'
```

< [icon]

Request URL  
http://localhost:8082/api/v1/swipe/profile

Server response

| Code | Details            |
|------|--------------------|
| 400  | Error: Bad Request |

Response body

```
{
 "code": "ERROR-API-829",
 "message": "max swipe or like is 10 you couldn't continue , please purchase premium for unlimited swip and like",
 "data": null,
 "errors": null,
 "request_id": "14e75a8b-f9db-4380-aazf-fc9b592e971",
 "timestamp": "2024-06-06 15:24:12"
}
```

[icon] Download

Response headers

```
access-control-allow-credentials: false
access-control-allow-headers: Origin,Accept,Content-Type,Authorization
access-control-allow-methods: GET,POST
access-control-allow-origin: *
content-length: 273
content-type: application/json; charset=utf-8
date: Thu, 06 Jun 2024 08:24:12 GMT
server: beegoServer:2.0.6
x-request-id: 14e75a8b-f9db-4380-aazf-fc9b592e971
```

- POST

/v1/user/purchase-premium

PurchasePremiumUpdateStatus

Parameters

Cancel

| Name                                  | Description                                  |
|---------------------------------------|----------------------------------------------|
| Accept-Language<br>string<br>(header) | lang                                         |
|                                       | <input type="text" value="Accept-Language"/> |

Execute

Clear

Responses

Response content type application/json

Curl

```
curl -X 'POST' \
 'http://localhost:8082/api/v1/user/purchase-premium' \
 -H 'accept: application/json' \
 -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bm9pcCI6ImlvaGZlbnJhbGFYIiwiaWF0IjE0NjUwMjUwMD0.OTQ0MjUwMjUwMD0.OTQ0MjUwMjUwMD0' \
 -d ''
```

Request URL

http://localhost:8082/api/v1/user/purchase-premium

Server response

Code Details

200

Response body

```
{
 "code": "OK",
 "message": "operation successfully executed.",
 "data": null,
 "errors": null,
 "request_id": "3fa6dbf8-8813-4cc3-b03d-093ba66483c5",
 "timestamp": "2024-06-06 15:26:46"
}
```

Download

Response headers

```
access-control-allow-credentials: false
access-control-allow-headers: Origin,Accept,Content-Type,Authorization
access-control-allow-methods: GET,POST
access-control-allow-origin: *
content-length: 195
content-type: application/json; charset=utf-8
date: Thu, 06 Jun 2024 08:26:46 GMT
server: bogaoserver:2.0.0
x-request-id: 3fa6dbf8-8813-4cc3-b03d-093ba66483c5
```

```
Curl -X "POST" \
http://localhost:8082/api/v1/user/login' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
 "email": "mohradyatama24@gmail.com",
 "password": "password"
}'
```

Request URL

http://localhost:8082/api/v1/user/login

Server response

| Code | Details                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 200  | <p>Response body</p> <pre>{   "code": "OK",   "message": "operation successfully executed.",   "data": {     "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFrpbCIGlmalvaDzhldmhGFtYTY1BQ6dYVwlsLmVybSIsImV4cCI6IjE4ODcxNzcxMzcxZC80TDAwIiwiaWF0IjoxNDkxMzE3OTYyYVJlc3p3cm10IjIzb2NhbmVhc3Q0MDA4MHIsImphbiSiOiJEMTMtZWJlI2PDADOTU0RUduMDAlIChwc9maek1K2lk1jozMScidkl1jozM08.rZhWAPDUhtd2XD-sPGvIEgzQFZ_-GGS8axeVto6A7Q",     "expired_at": "2024-06-07 15:38:06.6954451 +0700 +07 m+=88574.112882601",     "user": {       "id": 11,       "email": "mohradyatama24@gmail.com",       "name": "radyatama",       "photo": "http://localhost:8082/external/storage/asf8wbX0ti.jpeg",       "age": 25,       "bio": "Software Developer",       "password": "16.79q1888",       "latitude": -6.3270100,       "verified": true     }   },   "errors": null,   "request_id": "37aa2e18-607c-4acfc-9e93-da7a92a6193b",   "timestamp": "2024-06-06 15:38:00" }</pre> |

Response headers