TECHINCAL DOCUMENT

Online Shop API

Editor : **Arif Nur Rahman**
Create Date : **08 April 2021**
Revision Date : **-**
Name : **Online Shop API**
Version : **1.0**

Contents

1. Introduction About the System

1.1 System Overview

Online Shop API is an API for online shopping process, this API is easy to use. API process start from login / register, add to cart, ordering placement and payment, tracking status. This API user NodeJS for programming language and MySQL for database system.

1.2 Major Functions

The main functions provided by system are for login and register user, update profile, update password, create, update and delete address, create, update and delete product (only admin), get product data, add product to cart, update cart, delete cart, get checkout detail, order placement / create order, cancel order, confirm payment (upload evidence), get status order, update order (by admin and system) except received process;

1.3 System Environment

Hardware:

- Computer with Core i5 processor and 16 GB RAM.

Software:

- Node JS v14.18.0
- MySQL
- JWT

Node JS Library:

- Express v4.17.1
- Express-async-handler v.1.1.4
- Bcrypt v5.0.1
- Cors v2.8.5
- Date-and-time v2.0.1
- Dotenv v10.0.0
- Jsonwebtoken v8.5.1
- Mysql
- Nodemon v2.0.13
- Uniqid v5.4.0

2. System Development Standards

2.1 System

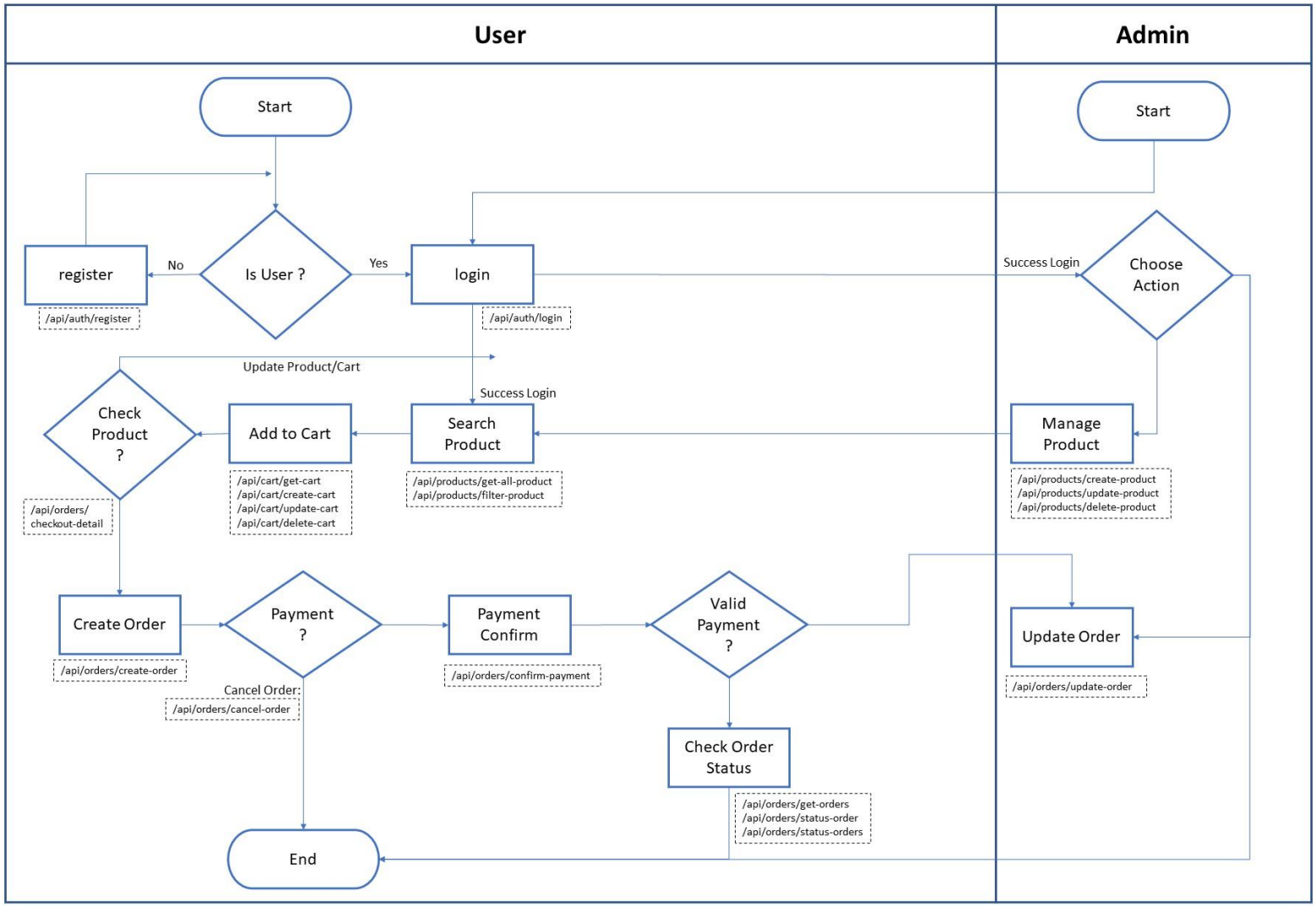
1. Naming variables must have the same name as their type.
2. Naming variables cannot have more than one.
3. Completion of achievements can only be done if the achievement filling time has been opened.
4. Buffering and blocking requirements for files, namely a maximum of 3 files and the file size of each 50MB file.
5. The format of the achievement filling is in accordance with the target format and scoring rules.
6. Classes must be declared in individual files with the controller file name that matches the class name.
7. Configuration to the database can be done in the **db.js** file in the config/ directory.
8. Configure the home/url business user with url: **IP:Port/api/routes-name**.
9. Configuring data information (model) is in the **/models** directory.
10. Configuration process information (controller) is in **/services** directory.
11. Configuration routes information is in **/routes** directory.
12. Start server with command **npm run server (run server.js)**

2.2 Test Case

Standard Test Cases include:

1. Minimum test and documentation of required test cases
2. Documenting test cases, expected results, and test data
3. The method for running the test.
4. Methods for documenting actual results
5. Methods for correcting errors and rerunning tests.

2.3 Flow of Online Shop API



3. API Lists

3.1 Authentication and Profile User

3.1.1 register

Use this API for register new user

Common Data

Name	Description
URL	http://localhost:3000/api/auth/register
Method	POST
Authorization	-

Request Example

JSON

```
{
    "email": "test2@olshop.com",
    "password": "test1234"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": {
        "id": 5,
        "username": "test2@olshop.com",
        "email": "test2@olshop.com",
        "status": 1,
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…"
    }
}
```

Error

Name	Description
Not Valid Email	Email is not valid format
Some Data Empty	Field of email or password is empty
Already Exist	Email is already store in user data

3.1.2 login

Use this API for login

Common Data

Name	Description
URL	http://localhost:3000/api/auth/login
Method	POST
Authorization	-

Request Example

JSON

```
{
    "email": "test2@olshop.com",
    "password": "test1234"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": {
        "id": 5,
        "username": "test2@olshop.com",
        "email": "test2@olshop.com",
        "status": 1,
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…"
    }
}
```

Error

Name	Description
Not Found	User data not found
Incorrect Password	Password user is incorrect

3.1.3 update-password

Use this API for update password user

Common Data

Name	Description
URL	http://localhost:3000/api/auth/update-password
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "oldPassword": "test1234",
    "newPassword": "test4321"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": {
        "id": 5,
        "username": "test2@olshop.com",
        "email": "test2@olshop.com",
        "status": 1,
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…"
    }
}
```

Error

Name	Description
Not Found	User data not found
Incorrect Password	Old Password user is incorrect

3.1.4 update-profile

Use this API for update profile user

Common Data

Name	Description
URL	http://localhost:3000/api/users/update-profile
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "name": "Dummy Account",
    "profile_account": "Profile Account Test",
    "gender": "M",
    "birth_date": "1989-12-12",
    "phone": "081112131415"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": {
        "id": 4,
        "name": "Dummy Account",
        "profile_account": "Profile Account Test",
       "gender": "M",
        "birth_date": "1989-12-12",
        "phone": "081112131415"
    }
}
```

Error

Name	Description
-	

3.1.5 create-address

Use this API for create address user

Common Data

Name	Description
URL	http://localhost:3000/api/users/create-address
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "recipient_name": "Dummy Account",
    "recipient_phone": "081112131415",
    "address_1": "Jl. Dummy No. 03",
    "address_2": "Ciputat Timur, Tangsel",
    "address_3": ""
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success create address"
}
```

Error

Name	Description
-	

3.1.6 update-address

Use this API for update address user

Common Data

Name	Description
URL	http://localhost:3000/api/users/update-address
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "address_id": "6",
    "recipient_name": "Arif Rahman",
    "recipient_phone": "081112131555",
    "address_1": "Jl. Semanggi 4 No. 15 Rt. 003 Rw. 001",
    "address_2": "Kel. Petekeyan, Kec. Tahunan, Kab. Kudus, Prov. Jawa Tengah",
    "address_3": ""
}
```

Response Example

JSON

```
{
    "success": true,
    "data": [
        {
            "address_id": 6,
            "recipient_name": "Arif Rahman",
            "recipient_phone": "081112131555",
            "address_1": "Jl. Semanggi 4 No. 15 Rt. 003 Rw. 001",
            "address_2": "Kel. Petekeyan, Kec. Tahunan, Kab. Kudus, Prov. Jawa Tengah",
            "address_3": ""
        }
    ]
}
```

Error

Name	Description
Not Found	Address data not found

3.1.7 delete-address

Use this API for delete address user

Common Data

Name	Description
URL	http://localhost:3000/api/users/delete-address
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "address_id": "4"
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success deleted address data"
}
```

Error

Name	Description
Not Found	Address data not found

3.1.8 get-address

Use this API for get all address user

Common Data

Name	Description
URL	http://localhost:3000/api/users/delete-address
Method	GET
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON
-

Response Example

JSON

```
{
    "success": true,
    "data": [
        {
            "user_id": 4,
            "profile_id": 4,
            "address_id": 5,
            "recipient_name": "Dummy Account",
            "recipient_phone": "081112131415",
            "address_1": "Jl. Dummy No. 03",
            "address_2": "Ciputat Timur, Tangsel",
            "address_3": ""
        }
    ]
}
```

Error

Name	Description
-	

3.1.9 get-user

Use this API for get data user

Common Data

Name	Description
URL	http://localhost:3000/api/auth/me
Method	GET
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

-

Response Example

JSON


```
{
    "success": true,
    "data": {
        "id": 4,
        "username": "test1@olshop.com",
        "email": "test1@olshop.com",
        "name": "Dummy Account",
        "profile_account": "Profile Account Test",
        "gender": "M",
        "birth_date": "1989-12-12",
        "phone": "081112131415",
        "addressData": [
            {
                "user_id": 4,
                "profile_id": 4,
                "address_id": 5,
                "recipient_name": "Dummy Account",
                "recipient_phone": "081112131415",
                "address_1": "Jl. Dummy No. 03",
                "address_2": "Ciputat Timur, Tangsel",
                "address_3": ""
            }
        ]
    }
}
```


Error

Name	Description
-	

3.2 Product Data

3.2.1 get-all-product

Use this API for get all product data (status active)

Common Data

Name	Description
URL	http://localhost:3000/api/products/get-all-product
Method	GET
Authorization	-

Request Example

JSON

–

Response Example

JSON


```
{
    "success": true,
    "data": [
        {
            "id": 1,
            "name": "Xiaomi Redmi 10 4/64GB - Carbon Gray",
            "description": "6.5 inches, MediaTek Helio G88,RAM 4GB, ROM 64GB;
microSDXC slot, Android 11, Kamera Belakang: 50 MP, Kamera Depan: 8 MP",
            "brand": "Xiaomi",
            "category": "Smartphone",
            "real_price": 2099000,
            "discount_percentage": 0,
            "discount_price": 0,
            "price": 2099000,
            "stock": 10,
            "rating": 0,
            "count_review": 0,
            "image_content": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…"
        }
    ]
}
```

Error

Name	Description
-	

3.2.2 get-product

Use this API for get product data (status active) based on product_id

Common Data

Name	Description
URL	http://localhost:3000/api/products/get-product
Method	POST
Authorization	-

Request Example

JSON


```
{
    "product_id": "1"
}
```


Response Example

JSON


```
{
    "success": true,
    "data": [
       {
            "id": 1,
            "name": "Xiaomi Redmi 10 4/64GB - Carbon Gray",
            "description": "6.5 inches, MediaTek Helio G88,RAM 4GB, ROM 64GB;
microSDXC slot, Android 11, Kamera Belakang: 50 MP, Kamera Depan: 8 MP",
            "brand": "Xiaomi",
            "category": "Smartphone",
            "real_price": 2099000,
            "discount_percentage": 0,
            "discount_price": 0,
            "price": 2099000,
            "stock": 10,
            "rating": 0,
            "count_review": 0,
            "image_content": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…" }]}
```

Error

Name	Description
-	

3.2.3 filter-product

Use this API for get product data (status active) based on keyword

Common Data

Name	Description
URL	http://localhost:3000/api/products/filter-product
Method	POST
Authorization	-

Request Example

JSON

```
{
    "keyword": "Xiaomi Redmi 10 4/64GB"
}
```

Response Example

JSON


```
{
    "success": true,
    "data": [
        {
            "id": 1,
            "name": "Xiaomi Redmi 10 4/64GB - Carbon Gray",
            "description": "6.5 inches, MediaTek Helio G88,RAM 4GB, ROM 64GB;
microSDXC slot, Android 11, Kamera Belakang: 50 MP, Kamera Depan: 8 MP",
            "brand": "Xiaomi",
            "category": "Smartphone",
            "real_price": 2099000,
            "discount_percentage": 0,
            "discount_price": 0,
            "price": 2099000,
            "stock": 10,
            "rating": 0,
            "count_review": 0,
            "image_content": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…" }]}
```

Error

Name	Description
-	

3.2.4 get-inactive-product

Use this API for get all product data (status inactive)

Common Data

Name	Description
URL	http://localhost:3000/api/products/get-inactive-product
Method	GET
Authorization	Bearer Token (Admin JWT Token from Login)

Request Example

JSON

```
-
```

Response Example

JSON


```
{
    "success": true,
    "data": [
        {
            "id": 1,
            "name": "Xiaomi Redmi 10 4/64GB - Carbon Gray",
            "description": "6.5 inches, MediaTek Helio G88,RAM 4GB, ROM 64GB;
microSDXC slot, Android 11, Kamera Belakang: 50 MP, Kamera Depan: 8 MP",
            "brand": "Xiaomi",
            "category": "Smartphone",
            "real_price": 2099000,
            "discount_percentage": 0,
            "discount_price": 0,
            "price": 2099000,
            "stock": 10,
            "rating": 0,
            "count_review": 0,
            "image_content": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9…"
        }
    ]
}
```

Error

Name	Description
Not Auhtorize	JWT Token not authorize for access this route (admin only)

3.2.5 create-product

Use this API for create product

Common Data

Name	Description
URL	http://localhost:3000/api/products/create-product
Method	POST
Authorization	Bearer Token (Admin JWT Token from Login)

Request Example

JSON

```
{
    "name": "Product Test",
    "description": "Description",
    "brand": "Testing",
    "category": "Smartphone",
    "price": "1100000",
    "discount_percentage": "0",
    "stock": "10",
    "rating": "",
    "count_review": "",
    "image_content": "/9j/4R0wRXhpZgAATU0AKgAAAAg…",
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success create Product"
}
```

Error

Name	Description
Not Auhtorize	JWT Token not authorize for access this route (admin only)

3.2.6 update-product

Use this API for update product

Common Data

Name	Description
URL	http://localhost:3000/api/products/update-product
Method	POST
Authorization	Bearer Token (Admin JWT Token from Login)

Request Example

JSON

```
{
    "product_id": "17",
    "name": "Product Update",
    "price": "500000"
}
```

Response Example

JSON


```
{"success": true,
    "data": [
        {
            "id": 17,
            "name": "Product Update",
            "description": "Description",
            "brand": "Testing",
            "category": "Smartphone",
            "real_price": 500000,
            "discount_percentage": 0,
            "discount_price": 0,
            "price": 500000,
            "stock": 10,
            "rating": 0, "count_review": 0,
            "image_content": "/9j/4R0wRXhpZgAATU0AK…"}]}
```

Error

Name	Description
Not Auhtorize	JWT Token not authorize for access this route (admin only)
Not Found	Product data not found

3.2.7 delete-product

Use this API for delete product

Common Data

Name	Description
URL	http://localhost:3000/api/products/delete-product
Method	POST
Authorization	Bearer Token (Admin JWT Token from Login)

Request Example

JSON

```
{
    "product_id": "17"
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success deleted Product data"
}
```

Error

Name	Description
Not Auhtorize	JWT Token not authorize for access this route (admin only)
Not Found	Product data not found

3.3 Shopping Cart Process

3.3.1 create-cart

Use this API for add product to cart

Common Data

Name	Description
URL	http://localhost:3000/api/cart/create-cart
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "product_id": "3",
    "quantity": "1"
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success create Cart"
}
```

Error

Name	Description
Not Found	Product not found
Out of Stock	Quantity product cart more than product stock

3.3.2 update-cart

Use this API for update product at shopping cart

Common Data

Name	Description
URL	http://localhost:3000/api/cart/update-cart
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "cart_id": "5",
    "quantity": "2"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": [
        {
            "cart_id": 7,
            "user_id": 4,
            "product_id": 3,
            "quantity": 2,
            "product_name": "Apple iPhone 12 Pro 64GB, Purple",
            "price": 25000000,
            "image_content": "/9j/2wCEAAEBAQEBAQEBAQE…"
        }
    ]
}
```

Error

Name	Description
Not Found	Cart not found
Out of Stock	Quantity product cart more than product stock

3.3.3 delete-cart

Use this API for delete product at shopping cart

Common Data

Name	Description
URL	http://localhost:3000/api/cart/delete-cart
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "cart_id": "7"
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success deleted Cart data"
}
```

Error

Name	Description
Not Found	Cart not found

3.3.4 get-cart

Use this API for get data from shopping cart

Common Data

Name	Description
URL	http://localhost:3000/api/cart/get-cart
Method	GET
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON
-

Response Example

JSON

```
{
    "success": true,
    "data": {
        "countData": 1,
        "cartData": [
            {
                "cart_id": 8,
                "user_id": 4,
                "product_id": 3,
                "quantity": 1,
                "product_name": "Apple iPhone 12 Pro 64GB, Purple",
                "price": 12500000,
                "image_content": "/9j/2wCEAAEBAQEBAQEBAQE…"
            }
        ]
    }
}
```

Error

Name	Description
-	

3.4 Checkout and Order Process

3.4.1 checkout-detail

Use this API for check order summary before create order

Common Data

Name	Description
URL	http://localhost:3000/api/orders/checkout-detail
Method	GET

Name	Description
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
-
```

Response Example

JSON


```
{
    "success": true,
    "data": {
        "recipientData": [
            {
                "user_id": 4,
                "profile_id": 4,
                "address_id": 5,
                "recipient_name": "Dummy Account",
                "recipient_phone": "081112131415",
                "address_1": "Jl. Dummy No. 03",
                "address_2": "Ciputat Timur, Tangsel",
                "address_3": ""
            }
        ],
        "orderData": {
            "productCount": 1,
            "productData": [
                {
                    "cart_id": 8,
                    "user_id": 4,
                    "product_id": 3,
                    "quantity": 1,
                    "product_name": "Apple iPhone 12 Pro 64GB, Purple",
                    "price": 12500000,
                    "image_content": "/9j/2wCEAAEBAQEBAQEBAQEBAQE…"
                }
            ]
        },
        "totalAmount": 12500000
    }
}
```

Error

Name	Description
Not Found	Cart data not found

3.4.2 create-order

Use this API for create order based on cart data

Common Data

Name	Description
URL	http://localhost:3000/api/orders/create-order
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "address_id": "5",
    "delivery_service": "JNE",
    "note": ""
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success create Order"
}
```

Error

Name	Description
Cart Not Found	Cart data not found
Recipient Not Found	Recipient data not found

3.4.3 get-orders

Use this API for get al order based on user_id

Common Data

Name	Description
URL	http://localhost:3000/api/orders/get-orders
Method	GET
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
-
```

Response Example

JSON


```
{
    "success": true,
    "data": [
        {
            "order_id": 3,
            "order_code": "g2pcafe0kuhg8qnj",
            "total_price": 12500000,
            "sample_product_name": "Apple iPhone 12 Pro 64GB, Purple",
            "sample_product_image": "/9j/2wCEAAEBAQEBAQEBAQEBAQEBAQE…"
            "sample_product_quantity": 1,
            "sample_product_price": 12500000,
            "product_total": 1
        }
    ]
}
```


Error

Name	Description
Not Found	Order data not found

3.4.4 get-order

Use this API for get detail order based on order_id

Common Data

Name	Description
URL	http://localhost:3000/api/orders/get-order
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "order_id": "4"
}
```

Response Example

JSON


```
{
    "success": true,
    "data": {
        "order_id": 4,
        "order_code": "g2pcajkwkuhm0jhl",
        "total_price": 12500000,
        "receipt_number": null,
        "recipient_name": "Dummy Account",
        "recipient_phone": "081112131415",
        "recipient_address": "Jl. Dummy No. 03 Ciputat Timur, Tangsel ",
        "status": "payment_process",
        "note": "",
        "order_date": "2021-10-08",
        "payment_method": "bank_tranfer",
        "bank_name": "BCA",
        "bank_account_name": "7310252527",
        "bank_account_number": "PT. Risyarma Jaya",
        "payment_date": null,
        "delivery_service": "JNE",
        "delivered_date": null,
        "received_date": null,
        "detailOrder": [
            {
                "detail_id": 5,
                "product_id": 3,
                "product_name": "Apple iPhone 12 Pro 64GB, Purple",
                "product_image": "/9j/2wCEAAEBAQEBAQEBAQEB
                "product_price": 12500000,
                "quantity": 1
            }
        ]
    }
}
```


Error

Name	Description
Not Found	Order data not found

3.4.5 cancel-order

Use this API for cancel order based on order_id

Common Data

Name	Description
URL	http://localhost:3000/api/orders/cancel-order
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "order_id": "4"
}
```

Response Example

JSON

```
{
    "success": true,
    "message": "Success deleted Order data"
}
```

Error

Name	Description
Not Found	Order data not found

3.5 Payment and Tracking Order

3.5.1 confirm-payment

Use this API for upload evidence or payment into application

Common Data

Name	Description
URL	http://localhost:3000/api/orders/confirm-payment

Name	Description
Method	POST
Authorization	Bearer Token (JWT Token from Login)

Request Example

JSON

```
{
    "order_id": "3",
    "payment_evidence": "/9j/4AAQSkZJRgABAQAA…"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": "Success upload evidence payment"
}
```

Error

Name	Description
Not Found	Order data not found

3.5.2 update-order

Use this API for update the **payment_date** field and order **status** to be **paid_off** if the evidence is valid or update **status** to **packed** if the package is being packed, or update **recipient_number**, **delivered date** and **status** to delivered if it has been sent, and update **status** to **received** if the package has been received (***Admin or User Access***).

Common Data

Name	Description
URL	http://localhost:3000/api/orders/update-order
Method	POST
Authorization	Bearer Token (Admin & User (only received updated) JWT Token from Login)

Request Example

JSON

```
{
    "order_id": "3",
    "payment_date": "2021-10-12 13:30:06",
    "status": "paid_off"
}
```

Response Example

JSON

```
{
    "success": true,
    "data": "Success updated Order data"
}
```

Error

Name	Description
Not Found	Order data not found
Not Authorize	User token not authorize for access API

3.5.3 status-orders

Use this API for check order data based on status such as **payment_process, paid_off, packed, delivered, received**.

Common Data

Name	Description
URL	http://localhost:3000/api/orders/status-orders
Method	POST
Authorization	Bearer Token (Admin & User (only received updated) JWT Token from Login)

Request Example

JSON

```
{
    "order_id": "3",
    "status": "payment_process"
}
```

Response Example

JSON


```
{
    "success": true,
    "countData": 1,
    "data": [
        {
            "order_id": 1,
            "order_code": "g2pca2fckugqlj0n",
            "status": "delivered",
            "total_price": 27200000,
            "sample_product_name": "Apple iPhone 12 Pro
            "sample_product_image": "/9j/
                feBHrHOTMSQADtdz33+EQVx0HCXPIPitu
             "sample_product_quantity": 2,
             "sample_product_price": 25000000,
             "product_total": 3
        }
    ]
}
```


Error

Name	Description
Not Found	Order data not found

4. Program ID / Description Lists

4.1 Routes Lists

ROUTES FILE	DESCRIPTION
AuthRoutes.js	Route for user access such as login, register, get user data, and update password.
UserRoutes.js	Route for user profile such as update profile, get address, create, update, and delete address user.
ProductRoutest.js	Route for manage product such as create, update, delete product, get product data, product detail, filter / search product.
CartRoutest.js	Route for manage cart such as add product to cart, update product in cart, delete product in cart.
OrderRoutest.js	Route for manage order such as checkout detail, create order based on cart data, cancel order, payment confirmation (upload evidence), update order (status, payment date, received date, delivered date, etc) and tracking order (get data based on status order)

4.2 Functions/Services Lists

ACTION NAME	DESCRIPTION	CONTROLLER
login	Function to login into application (return JWT token for used in another API)	AuthController.js
register	Function to register user into application or API	
getMe	Function to get user / profile data	
updatePassword	Function to update password user	
updateProfile	Function to update profile user	UserController.js
getAddress	Function to get address user (all address)	
createAddress	Function to create address user	
updatedAddress	Function to update address user	
deleteAddress	Function to delete address user	
getAllProduct	Function to get all product in online shop	ProductController.js

getProduct	Function to get detail product based on id	
getInactiveProduct	Function to get all inactive product	
filterProduct	Function to get product based on keyword	
createProduct	Function to create product (by admin)	
updateProduct	Function to update product (by admin)	
deleteProduct	Function to delete product (by admin)	
getCart	Function to get cart data based on user_id	CartController.js
createCart	Function to add product to cart	
updateCart	Function to update product in cart	
deleteCart	Function to delete product in cart	
detailCheckout	Function to get checkout detail before order	OrderController.js
getOrders	Function to get all orders data and 1 sample product from order detail	
getOrder	Function to get order data based on order_id	
createOrder	Function to create order based on user_id and cart data	
updateOrder	Function to update order data such as update status, payment_date, delivered_date, receipt_number, received_date, etc	
deleteOrder	Function to cancel order	
confirmPayment	Function to upload payment evidence	
statusOrders	Function to get order data based on status payment_process / paid_off / packed / delivered / received / canceled	

4.3 Models Lists

ACTION NAME	DESCRIPTION	MODELS
findOne	Function for get user data based on email	User.js
findById	Function for get user data based on id	
getUser	Function to get user data join profile and address based on user_id / id	

getProfile	Function to get profile data based on user_id	
insertData	Function to insert user data	
updateData	Function to update user data	
insertProfileData	Function to insert profile data	
updateProfileData	Function to update profile data	
getAddress	Function to get address data based on user_id	
getAddressOne	Function to get address data based on user_id limit 1	
getAddressById	Function to get address data based on address_id	
insertAddress	Function to insert address data	
updateAddress	Function to update address data	
deleteAddress	Function to delete address data	
getAllProduct	Function to get all product data	Product.js
getProduct	Function to get product data by id	
getInactiveProduct	Function to get all inactive product data	
filterProduct	Function to get product data by keyword	
insertProduct	Function to insert product data	
updateProduct	Function to update product data	
deleteProduct	Function to delete product data	
getCartById	Function to get cart data by cart_id	Cart.js
getCart	Function to get cart data by user_id join with user and product data	
sumCart	Function to get total amount of cart based on user_id join with user and product data	
insertCart	Function to insert product to cart data	
updateCart	Function to update product in cart data	
deleteCartt	Function to delete product in cart data	
getOrderData	Function to get order data based on user_id and group by order_code	Order.js

getOrderMaster	Function to get order master data based on order_id	
getOrderDetail	Function to get order detail based on order_master_id	
getOrderById	Function to get all field from order master based on id	
insertOrder	Function to insert order master data	
insertOrderDetail	Function to insert order detail based on product in cart data	
updateOrder	Function to update order master data	
deleteOrder	Function to update status order master to canceled	
getStatusOrder	Function to get order data based on status and user_id	

4.4 Middlewares Lists

FILE NAME	DESCRIPTION
AuthHandler.js	File or function to verify JWT token from header authorization API

4.5 Libs Lists

FILE NAME	DESCRIPTION
generateToken.js	File or function to generate JWT token based on user_id

4.6 Config Lists

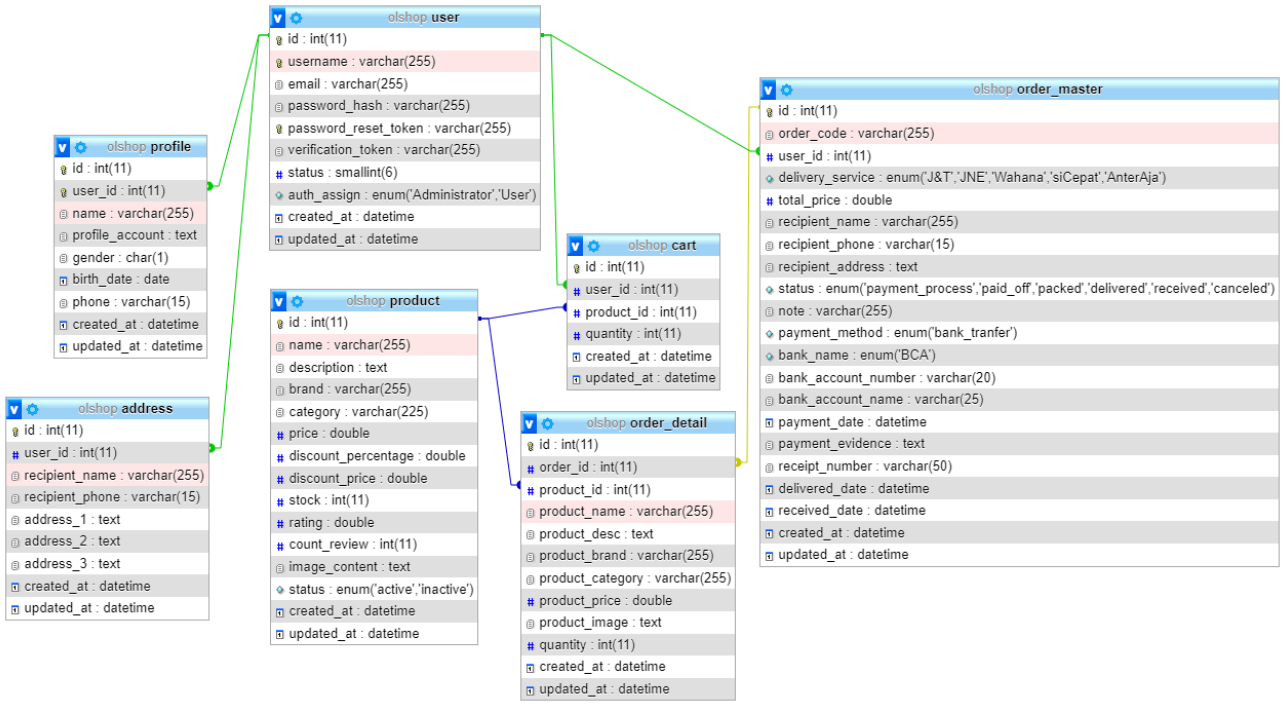
FILE NAME	DESCRIPTION
db.js	File of function for configuration database connection

5. File/Schema Listings

A. List Table

Table Name	Description	Type
user	Table to store user data	Master
profile	Table to store profile user data relation with user table by user.id	Master
address	Table to manage address data relation with user table by user.id	Master
product	Table to manage product data	Master
cart	Table to store and manage cart data relation with user table by user.id and product table by product.id	Transaction
order_master	Table to store and manage order data (master) relation with user table by user.id	Transaction
order_detail	Table to store and manage order data (product detail) relation with order_master table by order_id and product table by product_id	Transaction

B. Relationship



C. Table Detail Description

1. User (user)

   The User table as table to store user data

Field Name	Type	Description	Key
id	int(11)	id user data auto increment	Primary Key
username	varchar(255)	Username user	Unique
email	varchar(255)	Email user	-
password_hash	varchar(255)	Password user (in hash value)	-
password_reset_token	varchar(255)	Password reset token (if request reset / forgot password)	Unique
verification_token	varchar(255)	Verification token (if new register for validation data)	-
status	smallint(6)	Status user	-
auth_assign	enum('Administrator','User')	Assignment type	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

2. Profile (profile)

   The profile table as table to store profile user data relation with user table by user.id

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
user_id	int(11)	User_id form user table	Unique
name	varchar(255)	Name of user	-
profile_account	text	Profile of User	-
gender	char(1)	Gender of user	-
birth_date	date	Birth date of user	-
phone	varchar(15)	Phone of user	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

3. Address (address)

   The address table as table to manage address data relation with user table by user.id

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
user_id	int(11)	User_id form user table	Key / MUL
recipient_name	varchar(255)	Name of recipient	-
recipient_phone	varchar(15)	Phone of recipient	-
address_1	text	Address 1 of recipient	-
address_2	text	Address 2 of recipient	-
address_3	text	Address 3 of recipient	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

4. Product (product)

   The product table as table to manage product data

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
name	varchar(255)	Name of product	-
description	text	Description of product	-
brand	varchar(255)	Brand of product	-
category	varchar(225)	Category of product	-
price	double	Real Price of Product	-
discount_percentage	double	Percentage discount of product	-
discount_price	double	Price discount of product (real price * percentage)	-
stock	int(11)	Stock of product	-
rating	double	Rating of product	-
count_review	int(11)	Count review of product	-
image_content	text	Image/Photo of product in base64 data	-
status	enum('active','inactive')	Status of product	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

5. Cart (Cart)
   The cart table as table to store and manage cart data relation with user table by user.id and product table by product.id

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
user_id	int(11)	User_id form user table	Key / MUL
product_id	int(11)	Product id from product table	Key / MUL
quantity	int(11)	Quantity of product	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

6. Order Master (order_master)
   The product table as table to store and manage order data (master) relation with user table by user.id

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
order_code	varchar(255)	Number of order	-
user_id	int(11)	User_id form user table	Key / MUL
delivery_service	enum('J&T','JNE','Wahana','siCepat','AnterAja')	Type of delivery service	-
total_price	double	Amount total of product	-
recipient_name	varchar(255)	Name of Recipient	-
recipient_phone	varchar(15)	Phone of Recipient	-
recipient_address	text	Address of Recipient	-
status	enum('payment_process','paid_off','packed','delivered','received','canceled')	Status of order	-
note	varchar(255)	Note of order	-
payment_method	enum('bank_tranfer')	Payment method value default is bank transfer	-
bank_name	enum('BCA')	Name of Payment Bank	-
bank_account_number	varchar(20)	Account Number of Payment Bank	-
bank_account_name	varchar(25)	Account Name of Payment Bank	-

Field Name	Type	Description	Key
payment_date	datetime	Date of Payment	-
payment_evidence	text	Evidence of Payment in base64 format	-
receipt_number	varchar(50)	Receipt Number for package	-
delivered_date	datetime	Date of deliver package	-
received_date	datetime	Date of receive package	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-

7. Order Detail (order_detail)
   The product table as table to store and manage order data (product detail) relation with order_master table by order_id and product table by product_id

Field Name	Type	Description	Key
id	int(11)	id data auto increment	Primary Key
order_id	int(11)	id of oder master	Key / MUL
product_id	int(11)	id of product data	Key / MUL
product_name	varchar(255)	Name of product	-
product_desc	text	Description of product	-
product_brand	varchar(255)	Brand of product	-
product_category	varchar(225)	Category of product	-
product_price	double	Real Price of Product	-
product_image	text	Image/Photo of product in base64 data	-
quantity	int(11)	Quantity of product	-
created_at	datetime	Date of created data	-
updated_at	datetime	Date of updated data	-