**Introduction to Sales Management API**

Overview

The Sales Management API is designed to facilitate the management of sales-related data within your application. It provides endpoints for managing clients, products, sales transactions, and administrative functions.

**Before you start please read Registration Controller Api documentation first so that you can sign up and sign in to get your jwt token to access other endpoin**t**s. It’s at the last page**

Client Controller API Documentation

**Introduction**

The Client Controller API provides endpoints for managing clients in the system.

**Base URL**

The base URL for all endpoints in the Client Controller API is:

<http://localhost:8084/client>

**Authentication**

This API requires authentication to access certain endpoints. Users must provide valid authentication tokens in the request headers to access protected endpoints.

**Endpoints**

1. Get All Clients

Description

Returns a list of all clients stored in the system.

**Request**

Method: GET

URL: /get-all-clients

Headers: [Authentication headers] (Bearer $Jwt token)

Parameters: None

**Response**

Status Code: 200 OK

Response Body: json

[

{

"id": 1,

"name": "Client Name",

"email": "client@example.com",

...

},

...

]

**Example**

Request:

GET /client/get-all-clients

Response:

Status: 200 OK

[

{

"id": 1,

"name": "Client 1",

"email": "client1@example.com",

...

},

{

"id": 2,

"name": "Client 2",

"email": "client2@example.com",

...

},

...

]

2. Add Client

Description

Adds a new client to the system.

**Request**

Method: POST

URL: /add-client

Headers: [Authentication headers] (Bearer $Jwt token)

Request Body: Json

{

"name": "Michaelzy",

"location": "America",

"lastName": "Anowi"

}

**Response**

Status Code: 201 Created

Response Body: Client added successfully

**Example**

Request:

POST /client/add-client

Content-Type: application/json

{

"name": "Michaelzy",

"location": "America",

"lastName": "Anowi"

}

**Response:**

Status: 201 Created

Client added successfully

3. Update Client

Description

Updates an existing client in the system.

**Request**

Method: PUT

URL: /update-client

Headers: [Authentication headers] (Bearer $Jwt token)

Parameters:

id (query parameter): ID of the client to update

Request Body:

{

"name": "Updated Client Name",

"email": "updatedclient@example.com",

...

}

**Response**

Status Code: 200 OK

Response Body: Client updated successfully

**Example**

Request:

PUT /client/update-client?id=1

Content-Type: application/json

{

"name": "Updated Client Name",

"email": "updatedclient@example.com",

...

}

Response:

Status: 200 OK

Client updated successfully

4. Delete Client

Description

Deletes a client from the system.

Only an authorized User with the role-Admin can delete a client from the system

**Request**

Method: DELETE

URL: /delete-client

Headers: [Authentication headers] (Bearer $Jwt token)

Parameters:

id (query parameter): ID of the client to delete

**Response**

Status Code: 204 No Content

Response Body: None

**Example**

Request:

DELETE /client/delete-client?id=1

Response:

Status: 204 No Content

Product Controller API Documentation

**Introduction**

The Product Controller API provides endpoints for managing products in the system.

**Base URL**

The base URL for all endpoints in the Product Controller API is:

http://your-api-base-url.com/product i.e localhost:8084

**Authentication**

This API requires authentication to access certain endpoints. Users must provide valid authentication tokens in the request headers to access protected endpoints.

Endpoints

1. Get All Products

Description

Returns a list of all products stored in the system.

**Request**

Method: GET

URL: /get-all-products

Headers: [Authentication headers]

Parameters: None

**Response**

Status Code: 200 OK

Response Body: json

Copy code

[

{

"id": 1,

"name": "Product Name",

"description": "Product Description",

...

},

...

]

**Example**

Request:

GET /product/get-all-products

Response:

Status: 200 OK

[

{

"id": 1,

"name": "Product 1",

"description": "Description of Product 1",

...

},

{

"id": 2,

"name": "Product 2",

"description": "Description of Product 2",

...

},

...

]

2. Add Product

Description

Adds a new product to the system.

**Request**

Method: POST

URL: /add-product

Headers: [Authentication headers]

Request Body: json

{

"name": "New Product",

"description": "Description of New Product",

...

}

**Response**

Status Code: 201 Created

Response Body: Product added successfully

Example

Request:

POST /product/add-product

Content-Type: application/json

{

"name": "Fridge",

"description": "Description of New Product",

“InitialQuantity”: 20

...

}

**Response:**

Copy code

Status: 201 Created

Product added successfully

3. Update Product

**Description**

Updates an existing product in the system.

**Request**

Method: PUT

URL: /update-product/{id}

Headers: [Authentication headers]

Parameters:

{id} (path parameter): ID of the product to update

**Request Body:**

{

"name": "Fridge",

"description": "Updated Description",  
 “quantity”: 70

...

}

**Response**

Status Code: 200 OK

Response Body: Product updated successfully

Example

**Request:**

PUT /product/update-product/1

Content-Type: application/json

{

"name": "Updated Product Name",

"description": "Updated Description",

...

}

Response:

Status: 200 OK

Product updated successfully

**4. Delete Product**

**Description**

Deletes a product from the system.

Request

Method: DELETE

URL: /delete-product

Headers: [Authentication headers]

Parameters:

{id} (query parameter): ID of the product to delete

**Response**

Status Code: 204 No Content

Response Body: None

Example

Request:

DELETE /product/delete-product?id=1

Response:

Status: 204 No Content

Sales Controller API Documentation

**Introduction**

The Sales Controller API provides endpoints for managing sales in the system.

Base URL

The base URL for all endpoints in the Sales Controller API is:

http://your-api-base-url.com/sales

**Authentication**

This API requires authentication to access certain endpoints. Users must provide valid authentication tokens in the request headers to access protected endpoints.

Endpoints

1. Get All Sales

**Description**

Returns a list of all sales stored in the system.

**Request**

Method: GET

URL: /get-all-sales

Headers: [Authentication headers]

Parameters: None

**Response**

Status Code: 200 OK

Response Body:

[

{

"id": 1,

"date": "2024-04-10T10:00:00",

"totalPrice": 100.00,

...

},

...

]

Example

Request:

GET /sales/get-all-sales

Response:

Copy code

Status: 200 OK

[

{

"id": 1,

"date": "2024-04-10T10:00:00",

"totalPrice": 100.00,

...

},

{

"id": 2,

"date": "2024-04-11T11:00:00",

"totalPrice": 150.00,

...

},

...

]

2. Add Sale

**Description**

Adds a new sale to the system.

**Request**

Method: POST

URL: /add-sale

Headers: [Authentication headers]

Request Body:

{

"productId": 1,

"quantity": 5,

...

}

**Response**

Status Code: 201 Created

Response Body: Sale added successfully

Example

Request:

POST /sales/add-sale

Content-Type: application/json

{

"sales": {

"client": {

"name": "Wizzy",

"location": "Argentina",

"lastName": "Layman"

}

},

"products": [

{

"name": "Generators",

"salesQuantity": 5

},

{

"name": "Microwave oven",

"salesQuantity": 2

}

]

}

**Response:**

Status: 201 Created

Sale added successfully

3. Update Sale

**Description**

Updates an existing sale in the system.  
Here you can edit just the salequantities and total prices of the sale if they were calculated wrongly. This means the editing is done once a sale has been made

**Request**

Method: PUT

URL: /update-sale/{id}

Headers: [Authentication headers]

Parameters:

{id} (path parameter): ID of the sale to update

Request Body:

{

"saleQuantity": 1,

"total": 10,

...

}  
  
total here signifies totalPrice

**Response**

Status Code: 200 OK

Response Body: Sale updated successfully

Example

Request:

PUT /sales/update-sale/1

Content-Type: application/json

{

"saleQuantity": 1,

"total": 10,

...

}

Response:

Status: 200 OK

Sale updated successfully

4. Delete Sale

Description

Deletes a sale from the system

**Request**

Method: DELETE

URL: /delete-sale/

Headers: [Authentication headers]

Parameters:

{id} (Query parameter): ID of the sale to delete

**Response**

Status Code: 204 No Content

Response Body: None

Example

Request:

DELETE /sales/delete-sale

Response:

Status: 204 No Content

Client Reporting Controller API Documentation

**Introduction**

The Client Reporting Controller API provides endpoints for generating reports related to clients in the system.

**Base URL**

The base URL for all endpoints in the Client Reporting Controller API is:

http://your-api-base-url.com/clients-report

**Endpoints**

1. Generate Client Report

Description

Generates a comprehensive report about clients including total number of clients, top spending clients, client activity, and client location statistics.

**Request**

Method: GET

URL: /clients-client

Headers: None

Parameters:

id (query parameter): ID of a specific client to generate report for the client activity. The response too will generate details of other clients in the differenct sections   
i.e total number of clients, top spending clients, clients location statistics

**Response**

Status Code: 200 OK

Response Body: JSON object containing various report data including:

totalNumberOfClients: Total number of clients in the system

topSpendingClients: List of top spending clients

clientActivity: Client activity for a specific client (if id parameter provided)

clientsLocationStatistics: Statistics about clients' locations

**Example**

Request:

GET /clients-report/clients-client?id=1

Response:

Status: 200 OK

{

"totalNumberOfClients": 100,

"topSpendingClients": [

{

"name": "Client A",

"totalSpending": 500.00

},

{

"name": "Client B",

"totalSpending": 450.00

},

...

],

"clientActivity": {

"name": "Client X",

"activity": "High"

},

"clientsLocationStatistics": {

"country": {

"USA": 50,

"Canada": 30,

"UK": 20

},

"city": {

"New York": 25,

"Los Angeles": 20,

"Toronto": 15,

...

}

}

}

Sales Reporting Controller API Documentation

**Introduction**

The Sales Reporting Controller API provides endpoints for generating reports related to sales in the system.

**Base URL**

The base URL for all endpoints in the Sales Reporting Controller API is:

<http://your-api-base-url.com/sales-report>

**Endpoints**

1. Generate Sales Report

**Description**

Generates a comprehensive report about sales within a specified date range, including the total number of sales, total revenue, top selling products, and top performing sellers.

**Request**

Method: GET

URL: /sales-report

Headers: None

Query Parameters:

startDate (required): Start date of the report period (format: YYYY-MM-DD)

endDate (required): End date of the report period (format: YYYY-MM-DD)

**Response**

Status Code: 200 OK

Response Body: JSON object containing various report data including:

totalNumberOfSales: Total number of sales within the specified date range

totalRevenue: Total revenue generated from sales within the specified date range

topSellingProducts: List of top selling products within the specified date range

topPerformingSellers: List of top performing sellers within the specified date range

**Example**

Request:

GET /sales-report?startDate=2024-01-01&endDate=2024-03-31

Response:

Status: 200 OK

{

"totalNumberOfSales": 1000,

"totalRevenue": 50000.00,

"topSellingProducts": [

{

"productId": 1,

"productName": "Product A",

"totalSales": 150,

"totalRevenue": 7500.00

},

{

"productId": 2,

"productName": "Product B",

"totalSales": 120,

"totalRevenue": 6000.00

},

...

],

"topPerformingSellers": [

{

"sellerId": 1,

"sellerName": "Seller X",

"totalSales": 250,

"totalRevenue": 12500.00

},

{

"sellerId": 2,

"sellerName": "Seller Y",

"totalSales": 200,

"totalRevenue": 10000.00

},

...

]

}

Product Reporting Controller API Documentation

**Introduction**

The Product Reporting Controller API provides endpoints for generating reports related to products in the system.

**Base URL**

The base URL for all endpoints in the Product Reporting Controller API is:

<http://your-api-base-url.com/product-report>

**Endpoints**

1. Generate Product Report

Description

Generates a report containing information about products, such as sales performance, inventory status, and pricing analysis.

**Request**

Method: GET

URL: /product-report

Headers: None

**Response**

Status Code: 200 OK

Response Body: JSON array containing product reports, where each report contains:

productId: ID of the product

productName: Name of the product

salesCount: Total number of sales for the product

totalRevenue: Total revenue generated by the product

inventoryStatus: Current status of the product's inventory

pricing Analysis: Analysis of the product's pricing

**Example**

Request:

GET /product-report

Response:

Status: 200 OK

[

{

"productId": 1,

"productName": "Product A",

"salesCount": 100,

"totalRevenue": 5000.00,

"inventoryStatus": "……",

"pricingAnalysis": "……."

},

{

"productId": 2,

"productName": "Product B",

"salesCount": 80,

"totalRevenue": 4000.00,

"inventoryStatus": "…………",

"pricingAnalysis": "……."

},

...

]

Admin Page Controller API Documentation

**Introduction**

The Admin Page Controller API provides endpoints for managing API users with administrative privileges.

Base URL

The base URL for all endpoints in the Admin Page Controller API is:

<http://your-api-base-url.com/admin-page>

**Endpoints**

1. Retrieve API Users List

Description

Retrieves a list of API users with administrative privileges.

**Request**

Method: GET

URL: /admin-page

Headers: None

**Response**

Status Code: 200 OK

Response Body: JSON array containing information about API users with administrative privileges.

**Example**

Request:

GET /admin-page

Response:

Copy code

Status: 200 OK

[

{

"id": 1,

"username": "admin1",

"email": "admin1@example.com",

"role": "Admin"

},

{

"id": 2,

"username": "admin2",

"email": "admin2@example.com",

"role": "Admin"

},

...

]

2. Save API User

Description

Saves a new API user with administrative privileges.

**Request**

Method: POST

URL: /admin-page/save-api-user

Headers: Content-Type: application/json

Request Body: JSON object containing details of the new API user to be saved.

Response

Status Code: 204 No Content

**Example**

Method: POST

URL: /admin-page/save-api-user

{

"username": "admin1@gmail.com",

“address": "Abuja",

“mobile”: “93933..”

“fullName”: “okoy…”

}

Remember as you are saving the apiUser, The records automatically get created for the Seller too. So in general using the method automatically creates a Seller.

3. Update User

Description

Updates an existing API user with the specified email.

**Request**

Method: PUT

URL: /admin-page/update-user

Query parameter {email} - String

Headers: Content-Type: application/json

Request Body: JSON object containing updated details of the API user.

Response

Status Code: 200 OK

Response Body: JSON object containing updated details of the API user.

4. Update User Role to Admin

Description

Updates the role of an existing API user with the specified email to "Admin".

**Request**

Method: PUT

URL: /admin-page/update-to-admin/

Query parameter {email} - String

**Response**

Status Code: 200 OK

Response Body: JSON object containing updated details of the API user.

5. Update User Role to Seller

Description

Updates the role of an existing API user with the specified email to "Seller".

**Request**

Method: PUT

URL: /admin-page/update-to-seller

Query parameter {email} – String

**Response**

Status Code: 200 OK

Response Body: JSON object containing updated details of the API user.

6. Delete User

Description

Deletes an existing API user with the specified username.

**Request**

Method: DELETE

URL: /admin-page/delete

Query parameter {email} – String

**Response**

Status Code: 204 No Content

History Controller Api Documentation

**Overview**

The History Controller provides endpoints for retrieving historical data, such as sales history or other activities logged in the system.

**Base URL**

The base URL for all endpoints in the History Controller is:

http://your-api-base-url.com/history

**Endpoints**

1. Retrieve All History

Description

Retrieves a list of all historical records stored in the system.

**Request**

Method: GET

URL: /history

Headers: None

Response

Status Code: 200 OK

Response Body: JSON array containing historical records.

Example

Request:

GET /history

Response:

Status: 200 OK

[

{

"id": 1,

"timestamp": "2024-04-01T10:00:00",

"activity": "Sale processed for client XYZ",

"user": "admin"

},

{

"id": 2,

"timestamp": "2024-04-01T11:00:00",

"activity": "Product added to inventory",

"user": "seller"

},

...

]

Registration Controller Api Documentation

**Overview**

The Registration Controller provides endpoints for user registration and token confirmation in the system.

**Base URL**

The base URL for all endpoints in the Registration Controller is:

http://your-api-base-url.com/registration i.e localhost:8084  
The number depends on the port number set in the application.properties  
Make sure to set your database up correctly and make the connections there

**Endpoints**

1. User Registration

Description

Registers a new user in the system.

**Request**

Method: POST

URL: /registration

Headers: None

Request Body: JSON object containing user registration data.

**Response**

Status Code: 201 Created if the user is successfully registered.

Response Body: Empty.

Example

Request:

POST /registration

Content-Type: application/json

{

"username": "admin1",

“address": "Abuja",

“mobile”: “93933..”

“fullName”: “okoy…”

}

Response:

Status: 201 Created