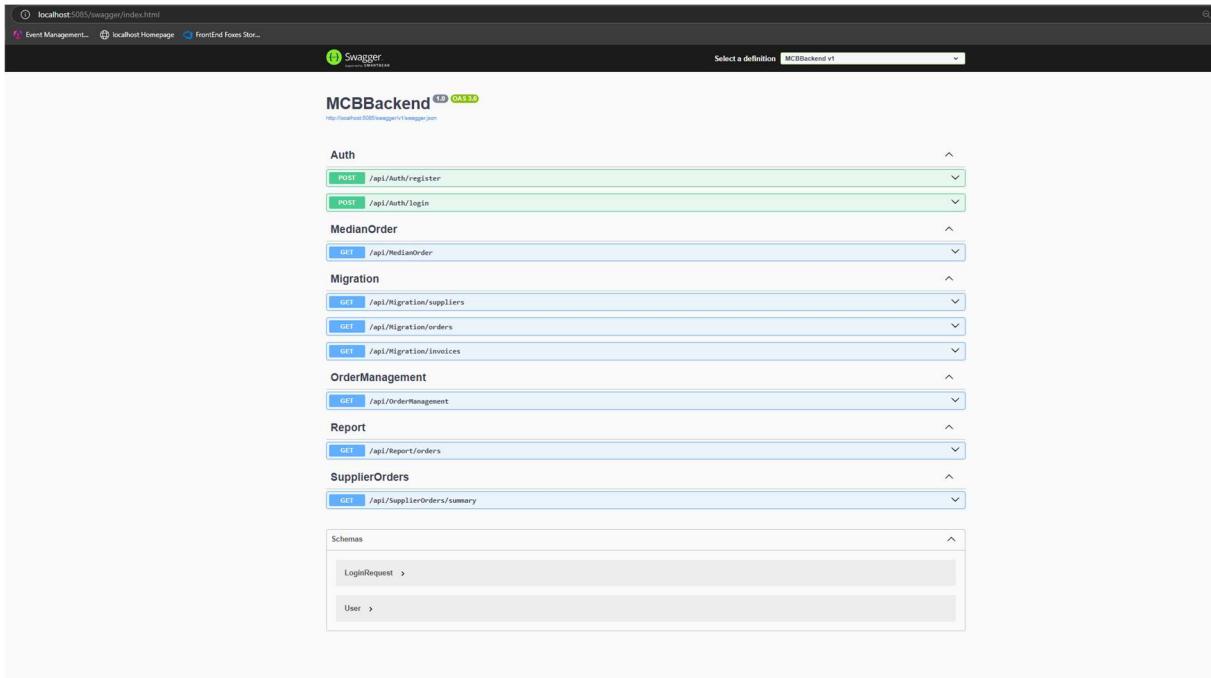


Informative Report on MCB Backend Development

1. Introduction

This report details the development of the **MCBBackend** system, which aims to provide a structured and automated solution for managing **Purchase Orders (POs)**, **Invoices**, and **Supplier Transactions** for **Moris Ltd**, a financial institution. The backend was designed to migrate data from an **MSSQL database**, normalize it, and expose **RESTful APIs** for data retrieval and reporting.



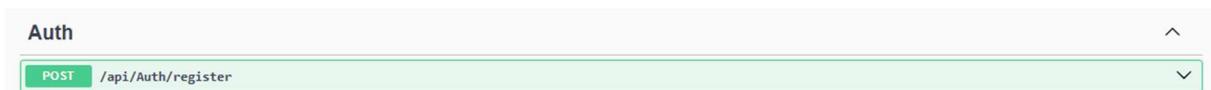
2. API Endpoints and Authentication

The **MCBBackend** exposes multiple API endpoints, categorized as follows:

2.1 Authentication APIs

To secure access, the backend provides JWT-based authentication for user registration and login.

- **POST /api/Auth/register** → Registers a new user.



Registering a user

Auth

POST /api/Auth/register

Parameters

No parameters

Request body

```
{ "username": "Abdurrahman Gurib", "password": "P@ssw0rd!", "role": "User" }
```

Cancel Reset

application/json

Execute

Responses

Code	Description	Links
200	OK	No links

User Registered Successfully

Auth

POST /api/Auth/register

Parameters

No parameters

Request body

```
{ "username": "Abdurrahman Gurib", "password": "P@ssw0rd!", "role": "User" }
```

Cancel Reset

application/json

Execute Clear

Responses

Curl

```
curl -X 'POST' \
'http://localhost:5085/api/Auth/register' \
-H 'Accept: */*' \
-H 'Content-Type: application/json' \
-d '{ "username": "Abdurrahman Gurib", "password": "P@ssw0rd!", "role": "User" }'
```

Request URL

http://localhost:5085/api/Auth/register

Server response

Code	Details
200	Response body { "message": "User registered successfully" } Download Response headers content-type: application/json; charset=utf-8 date: Wed, 06 Sep 2023 21:02:34 GMT server: Kestrel transfer-encoding: chunked

Responses

Code	Description	Links
200	OK	No links

Unit Test: Validation if re-register the same user

Auth

POST /api/Auth/register

Parameters

No parameters

Request body

application/json

```
{
  "username": "Abdurrahman Gurib",
  "password": "P@ssw0rd!",
  "role": "User"
}
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \
'http://localhost:5085/api/Auth/register' \
-H 'accept: */*' \
-H 'Content-Type: application/json' \
-d '{
  "username": "Abdurrahman Gurib",
  "password": "P@ssw0rd!",
  "role": "User"
}'
```

Request URL

http://localhost:5085/api/Auth/register

Server response

Code Details

400 Error: Bad Request
Undocumented

Response body

Username already exists

Response headers

```
content-type: text/plain; charset=utf-8
date: Mon, 24 Feb 2025 21:04:16 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code Description Links

200 OK No links

Server response status 400, and indicate Username already exists

Responses

Curl

```
curl -X 'POST' \
'http://localhost:5085/api/Auth/register' \
-H 'accept: */*' \
-H 'Content-Type: application/json' \
-d '{
  "username": "Abdurrahman Gurib",
  "password": "P@ssw0rd!",
  "role": "User"
}'
```

Request URL

http://localhost:5085/api/Auth/register

Server response

Code Details

400 Error: Bad Request
Undocumented

Response body

Username already exists

Response headers

```
content-type: text/plain; charset=utf-8
date: Mon, 24 Feb 2025 21:04:16 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code Description Links

200 OK No links

- **POST /api/Auth/login** → Authenticates users and generates a JWT token.

```
POST /api/Auth/login
```

Log in the user:

POST /api/Auth/login

Parameters

No parameters

Request body

```
{
  "username": "Abdurrahman Gurib",
  "password": "P@ssw0rd!"
}
```

application/json

Cancel Reset

Execute

Responses

Code	Description	Links
200	OK	No links

User Logged Successfully

POST /api/Auth/login

Parameters

No parameters

Request body

```
{
  "username": "Abdurrahman Gurib",
  "password": "P@ssw0rd!"
}
```

application/json

Cancel Reset

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:5005/api/Auth/login' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "username": "Abdurrahman Gurib",
    "password": "P@ssw0rd!"
}'
```

Request URL

<http://localhost:5005/api/Auth/login>

Server response

Code	Details
200	<p>Response body</p> <pre>{ "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpWC39.eyJlbm1xdWVfbmFtZSI6IkF1ZHycmFobAfFuIEd1cm11iwiM9sZSI6I1VzZXI1LClJyM10jE3NDM0MzEzNTksImV4cCI6MTC0MDQzMjI10SwidWF0IjoxNzQwNDM0MzU5LCjpc3M1O1JodHRwczovL2xvY2FsaG9zdDo1M0lkX1h1YAVXV1joiaHR0cHM6Ly9sb2NhbmhvC3QmNTAwM539.IUDf6Pm6N7FuxTCzY31bQjOEI40p1jNaxt4oSiFAVKA", "refresh_token": "2110c883-2f9c-4a84-97e5-a99dd3d1a064" }</pre> <p>Download</p> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Mon, 24 Feb 2025 21:09:19 GMT server: Kestrel transfer-encoding: chunked</pre>

Responses

Code	Description	Links
200	OK	No links

Security Measures

- ◆ **JWT Authentication:** Protects endpoints from unauthorized access.
- ◆ **Role-Based Access Control (RBAC):** Ensures restricted access.
- ◆ **SQL Injection Prevention:** Parameterized queries used in stored procedures.
- ◆ **Data Validation:** Input sanitization for secure data handling.

JWT Token Security:

Signed Token: The JWT is signed using HMAC-SHA256 with a secret key.

Token Expiration: The JWT has a short expiration time (15 minutes) to limit its validity period.

Issuer & Audience Claims: These ensure the token is valid and intended for your API.

Refresh Token Mechanism:

Refresh Token Generation: A GUID is used as a refresh token.

Refresh Token Storage: With the addition of SaveRefreshToken, the refresh token is stored securely.

Token Renewal: The refresh token allows obtaining a new JWT after expiration without re-authenticating.

Appsettings.json [JWT, SQL Connection, Tokens]

```
{  
  "ConnectionStrings": {  
    "DefaultConnection": "Server=4JPRJX3\\SQL2022DE;Database=MCB_Assignment_2025;Trusted_Connection=True;TrustServerCertificate=True;"  
  },  
  "Jwt": {  
    "Key": "A0D3C9F1E6B8G5H7I2K4J6L8M9N0P1Q3",  
    "Issuer": "https://localhost:5001",  
    "Audience": "https://localhost:5001",  
    "AccessTokenExpirationMinutes": 15,  
    "RefreshTokenExpirationDays": 7  
  },  
  "Logging": {  
    "LogLevel": {  
      "Default": "Information",  
      "Microsoft": "Warning",  
      "Microsoft.Hosting.Lifetime": "Information"  
    }  
  },  
  "AllowedHosts": "*"  
}
```

Unit Test: Validation if Username does not exist

POST /api/Auth/login

Parameters

No parameters

Request body

application/json

```
{ "username": "Noor-Ul-Haqq Gurib", "password": "P@ssw0rd!" }
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \ 'http://localhost:5085/api/Auth/login' \ -H 'accept: */*' \ -H 'Content-Type: application/json' \ -d '{ "username": "Noor-Ul-Haqq Gurib", "password": "P@ssw0rd!" }'
```

Request URL

<http://localhost:5085/api/Auth/login>

Server response

Code Details

401 Unauthorized

Undocumented

Error: Unauthorized

Response body

Invalid credentials

Download

Response headers

```
content-type: text/plain; charset=utf-8
date: Mon, 24 Feb 2025 21:11:39 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code Description Links

200 OK No links

Unit Test: Validation if Username is registered, and password is wrong

The screenshot shows a POST request to the endpoint `/api/Auth/login`. The request body is a JSON object:

```
{
  "username": "Abdurrahman Gurib",
  "password": "MCB@Software_Engineer2025Test"
}
```

The response details show a **401 Unauthorized** status with the message **Invalid credentials**. The response headers include:

```
content-type: text/plain; charset=utf-8
date: Mon, 24 Feb 2025 21:14:05 GMT
server: Kestrel
transfer-encoding: chunked
```

There is also a table for responses with a single entry for **OK**.

2.2 Data Management APIs

These APIs retrieve key data related to **suppliers, orders, invoices, and reports**.

The screenshot shows three GET requests under the **Migration** section:

- `GET /api/Migration/suppliers`
- `GET /api/Migration/orders`
- `GET /api/Migration/invoices`

2.2.1 Migration APIs

These endpoints fetch data migrated from the **BCM_ORDER_MGT** table:

- **GET /api/Migration/suppliers** → Retrieves supplier details.

Migration

GET /api/Migration/suppliers

Parameters

No parameters

Responses

Code	Description	Links
200	OK	No links

Migration

GET /api/Migration/suppliers

Parameters

No parameters

Responses

Execute

Code	Description	Links
200	OK	No links

Successfully migrated data to BDO.Suppliers Table

Migration

GET /api/Migration/suppliers

Parameters

No parameters

Responses

Execute

Curl

```
curl -X 'GET' \
  'http://localhost:5085/api/Migration/suppliers' \
  -H 'accept: */*'
```

Request URL

http://localhost:5085/api/Migration/suppliers

Server response

Code	Details
200	Response body

```
[{"supplierID": 1, "supplierName": "BACOASY CO. LTD.", "supplierContactName": "Kerry Parker", "supplierAddress": "68, Marock Lane, - , Vacoas, Mauritius", "supplierTown": "Vacoas", "supplierContactNumber1": "57841266", "supplierContactNumber2": "6028010", "supplierEmail": "bparker@inetnet.mu"}, {"supplierID": 2, "supplierName": "ENTELLO LTD", "supplierContactName": "Nimby", "supplierAddress": "998, Bistroop Street, Marlon, Pamplemousses, Mauritius", "supplierTown": "Pamplemousses", "supplierContactNumber1": "2428641", "supplierContactNumber2": "57841698", "supplierEmail": "nimeyey-t.mu"}, {"supplierID": 3, "supplierName": "FIRELAND BROS.", "supplierContactName": "Amelia Bridney", "supplierAddress": "Main court bldg, Nerling road, Moka, Mauritius", "supplierTown": "Moka", "supplierContactNumber1": "5948 0015", "supplierContactNumber2": "5948 0015"}, {"supplierID": 4, "supplierName": "GARIBOLDI LTD", "supplierContactName": "Dawn", "supplierAddress": "123, Main street, Mauritius", "supplierTown": "Mauritius", "supplierContactNumber1": "5555 5555", "supplierContactNumber2": "5555 5555"}]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Monday, 26 Dec 2025 21:21:42 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	No links

- **GET /api/Migration/orders** → Retrieves orders from suppliers.

GET /api/Migration/orders

Parameters

No parameters

Responses

Code	Description	Links
200	OK	No links

GET /api/Migration/orders

Parameters

No parameters

Responses

Code	Description	Links
200	OK	No links

Successfully migrated data to BDO.Orders Table

GET /api/Migration/orders

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
'http://localhost:5085/api/Migration/orders' \
-H 'accept: */*'
```

Request URL

<http://localhost:5085/api/Migration/orders>

Server response

Code Details

Code	Details
200	Response body

```
[
  {
    "orderId": 1,
    "originalOrderRef": "PO0001",
    "orderDate": "2024-01-01T00:00:00",
    "orderTotalAmount": 10000,
    "orderDescription": "Vehicle Spare parts",
    "orderStatus": "Closed",
    "supplierName": "PEGASUS LTD"
  },
  {
    "orderId": 2,
    "originalOrderRef": "PO0002",
    "orderDate": "2024-01-10T00:00:00",
    "orderTotalAmount": 50000,
    "orderDescription": "Purchase of Car",
    "orderStatus": "Open",
    "supplierName": "MOTOMAY CORP."
  },
  {
    "orderId": 3,
    "originalOrderRef": "PO0003",
    "orderDate": "2024-01-24T00:00:00",
    "orderTotalAmount": 57300,
    "orderDescription": "Computer screens",
    "orderStatus": "Closed",
    "supplierName": "DIGISAY CO. LTD."
  }
]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 24 Feb 2025 21:24:10 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	No links

- **GET /api/Migration/invoices** → Retrieves invoices linked to orders.

The screenshot shows two views of the same API endpoint. The top view is a standard API documentation page with sections for Parameters, Responses, and a Try it out button. The bottom view is a modal window with an Execute button instead of the Try it out button. Both views show the same endpoint details: GET /api/Migration/invoices, no parameters, and a 200 OK response.

Successfully migrated data to BDO.Invoices Table

This screenshot shows the results of executing the GET /api/Migration/invoices endpoint. It includes:

- Curl:** A terminal command to execute the request: `curl -X 'GET' \ 'http://localhost:5085/api/Migration/invoices' \ -H 'accept: */*'`
- Request URL:** `http://localhost:5085/api/Migration/invoices`
- Server response:**
 - Code:** 200
 - Details:** Response body
 - Content:** JSON array of invoice objects. The first object has an invoice ID of 1, order ID of 1, and a description of "Part payment for vehicle spare parts".
- Response headers:** Content-type: application/json; charset=utf-8, date: Mon, 24 Feb 2025 21:25:39 GMT, server: Kestrel, transfer-encoding: chunked
- Responses:** 200 OK

2.2.2 Order Processing APIs

- **GET /api/OrderManagement** → Retrieves structured **purchase order and invoice** details.

OrderManagement

GET /api/OrderManagement

Parameters

No parameters

Responses

Code	Description	Links
200	OK	No links

OrderManagement

GET /api/OrderManagement

Parameters

No parameters

Responses

Code	Description	Links
200	OK	No links

Displaying all raw data from BCM_ORDER_MGT which are not normalized.

OrderManagement

GET /api/OrderManagement

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5085/api/OrderManagement' \
  -H 'accept: */*'
```

Request URL

<http://localhost:5085/api/OrderManagement>

Server response

Code Details

Code	Details
200	Response body

```
[
  {
    "ORDER_REF": "P00001",
    "ORDER_DATE": "03-JAN-2024",
    "SUPPLIER_NAME": "PEGASUS LTD",
    "SUP_CONTACT_NAME": "Georges Neeroo",
    "SUP_ADDRESS": "40, Ferney Way, Mission Road, Curepipe, Mauritius",
    "SUP_CONTACT_NUMBER": "+461 5041 57412545",
    "SUP_EMAIL": "geonee@pegasus.mru",
    "ORDER_TOTAL_AMOUNT": "100000",
    "ORDER_DESCRIPTION": "Vehicle Spare parts",
    "ORDER_STATUS": "Closed",
    "INVOICE_REFERENCE": "-",
    "INVOICE_DATE": "-",
    "INVOICE_STATUS": "-",
    "INVOICE_AMOUNT": "-",
    "INVOICE_DESCRIPTION": "-"
  },
  {
    "ORDER_REF": "P00001-1",
    "ORDER_DATE": "03-JAN-2024",
    "SUPPLIER_NAME": "PEGASUS LTD",
    "SUP_CONTACT_NAME": "Georges Neeroo",
    "SUP_ADDRESS": "40, Ferney Way, Mission Road, Curepipe, Mauritius",
    "SUP_CONTACT_NUMBER": "+461 5041 57412545",
    "SUP_EMAIL": "geonee@pegasus.mru",
    "ORDER_TOTAL_AMOUNT": "-"
  }
]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 24 Feb 2025 21:29:03 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	No links

2.2.3 Reporting APIs

- **GET /api/Report/orders** → Provides a **summary report** of all purchase orders grouped by supplier region.

The screenshot shows three separate requests for the `/api/Report/orders` endpoint. Each request includes a 'Report' header, a 'GET' method, and the URL `/api/Report/orders`.
1. The first request shows the 'Parameters' section with 'No parameters' and the 'Responses' section with a 200 OK status.
2. The second request shows the 'Parameters' section with 'No parameters' and the 'Responses' section with a 200 OK status.
3. The third request shows the 'Parameters' section with 'No parameters' and the 'Responses' section with a 200 OK status. This request has a large blue 'Execute' button at the bottom of its panel.

Provides a **summary report** of all purchase orders grouped by supplier region.

This screenshot provides a detailed view of the `/api/Report/orders` endpoint. It includes:
- A 'Report' header with 'GET' and the URL `/api/Report/orders`.
- A 'Parameters' section with 'No parameters'.
- A 'Responses' section with a 'Curl' example:

```
curl -X 'GET' \
  'http://localhost:5005/api/Report/orders' \
  -H 'Accept: */*'
```


- A 'Request URL' field containing `http://localhost:5005/api/Report/orders`.
- A 'Server response' section showing a JSON response body:

```
[{"Region": "Camp Hill", "OrderReference": "11", "OrderStatus": "Open", "SupplierName": "Lamboni Stat Inc.", "OrderTotalAmount": "43,200.00", "OrderStatus": "Closed", "InvoiceTotalAmount": "43,200.11", "InvoiceTotalAmount": "43,200.00", "Action": "No Action"}, {"Region": "Camp Hill", "OrderReference": "12", "OrderStatus": "Open", "SupplierName": "Lamboni Stat Inc.", "OrderTotalAmount": "6,800.00", "OrderStatus": "Closed", "InvoiceTotalAmount": "6,800.00", "InvoiceTotalAmount": "6,800.00", "Action": "No Action"}, {"Region": "Curepipe", "OrderReference": "13", "OrderStatus": "Open", "SupplierName": "Pegasus Ltd.", "OrderTotalAmount": "2,656.00", "OrderStatus": "Closed", "InvoiceTotalAmount": "2,656.00", "InvoiceTotalAmount": "2,656.00", "Action": "No Action"}]
```


- A 'Response headers' section showing:

```
content-type: application/json; charset=utf-8
date: Mon, 24 Feb 2025 21:32:56 GMT
server: Kestrel
transfer-encoding: chunked
```


- A 'Responses' section with a 200 OK status.

- **GET /api/SupplierOrders/summary** → Lists all suppliers with their total number of orders and amounts per month.

SupplierOrders

GET /api/SupplierOrders/summary

SupplierOrders

GET /api/SupplierOrders/summary

Parameters

No parameters

Try it out

Responses

Code	Description	Links
200	OK	No links

SupplierOrders

GET /api/SupplierOrders/summary

Parameters

No parameters

Cancel

Execute

Responses

Code	Description	Links
200	OK	No links

Lists all suppliers with their total number of orders and amounts per month.

SupplierOrders

GET /api/SupplierOrders/summary

Parameters

No parameters

Cancel

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'http://localhost:5085/api/SupplierOrders/summary' \
-H 'accept: */*'
```

Request URL

http://localhost:5085/api/SupplierOrders/summary

Server response

Code Details

200 Response body

```
[{"Month": "April 2024", "SupplierName": "EMTELLO LTD", "SupplierContactName": "Megan Hemby", "SupplierContactNumber1": "+2426641", "SupplierContactNumber2": "+57841698", "TotalOrders": 1, "OrderTotalAmount": 21000}, {"Month": "August 2024", "SupplierName": "NOTOMAY CORP.", "SupplierContactName": "Stevens Seernah", "SupplierContactNumber1": "+5794-2513", "SupplierContactNumber2": "-", "TotalOrders": 1, "OrderTotalAmount": 5819625}, {"Month": "August 2024", "SupplierName": "PEGASUS LTD", "SupplierContactName": "Georges Neeroo", "SupplierContactNumber1": "+461-5841", "SupplierContactNumber2": "+57412545", "TotalOrders": 1, "OrderTotalAmount": 265000}],
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Mon, 24 Feb 2025 21:35:26 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code Description Links

200 OK No links

- **GET /api/MedianOrder** → Fetches the **median order** based on the order total amount.

The screenshot shows three related API documentation pages for the **MedianOrder** endpoint:

- MedianOrder**: Shows the **GET /api/MedianOrder** method with no parameters and a 200 OK response.
- MedianOrder**: Shows the **GET /api/MedianOrder** method with no parameters, a "Try it out" button, and a 200 OK response.
- MedianOrder**: Shows the **GET /api/MedianOrder** method with no parameters, an "Execute" button, and a 200 OK response.

Fetches the **median order** based on the order total amount.

This screenshot shows an expanded view of the **MedianOrder** API endpoint documentation:

- GET /api/MedianOrder**
- Parameters**: No parameters
- Responses**:
 - Curl**: curl -X 'GET' \ 'http://localhost:5085/api/MedianOrder' \ -H 'accept: */*' (with a copy icon)
 - Request URL**: http://localhost:5085/api/MedianOrder
 - Server response**:

Code	Details
200	Response body <pre>{ "orderReference": "3", "orderDate": "24 Jan 2024", "supplierName": "DIGISAY CO. LTD.", "orderTotalAmount": "57,380.00", "orderStatus": "Closed" }</pre> Response headers <pre>content-type: application/json; charset=utf-8 date: Mon, 24 Feb 2025 21:37:56 GMT server: Kestrel transfer-encoding: chunked</pre>
 - Responses**:

Code	Description	Links
200	OK	No links

3. Database Schema and Migration

The raw **BCM_ORDER_MGT** table was **normalized** into multiple relational tables:

3.1 Table Structure

1. **Suppliers** (Suppliers Table)
 - o Stores supplier details, including name, contact information, and location.
2. **Orders** (Orders Table)
 - o Stores purchase order details, linked to **Suppliers** via SupplierID.
3. **Invoices** (Invoices Table)
 - o Stores invoice details, linked to **Orders** via OrderID.

3.2 Data Migration

A **PL/SQL Procedure** was developed to extract, clean, and store the normalized data.

4. Stored Procedures for Reporting

4.1 Orders Summary Report (sp_ReportOrdersSummary)

This procedure extracts and formats purchase order data per **supplier region** with:
Order Reference (removes PO prefix)
Order Period (YYYY-MM)
Formatted Amounts (99,999,990.00)
Invoice Aggregation (STRING_AGG(InvoiceReference, '|'))
Action Field (To verify, To follow up, No Action)

4.2 Median Order Calculation (sp_ReportMedianOrder)

Fetches **the median purchase order** based on total amount, with:
Uppercased Supplier Name
Date Formatting (DD-MMM-YYYY)
Pipe-delimited Invoice References

4.3 Supplier Orders Summary (sp_ReportSupplierOrdersByMonth)

Extracts monthly order totals for suppliers:
Month (January 2024)
Supplier Contact Name & Numbers (formatted 5999-9999)
Total Orders & Total Order Amount

5. Security and Performance Considerations

5.1 Security Measures

- ◆ **JWT Authentication:** Protects endpoints from unauthorized access.
- ◆ **Role-Based Access Control (RBAC):** Ensures restricted access.
- ◆ **SQL Injection Prevention:** Parameterized queries used in stored procedures.
- ◆ **Data Validation:** Input sanitization for secure data handling.

5.2 Performance Optimization

- ◆ **Indexes on Foreign Keys (SupplierID, OrderID)**
- ◆ **Stored Procedures Instead of Ad-Hoc Queries**
- ◆ **Efficient Aggregation Using STRING_AGG()**

6. Tools and Technologies Used

- ◆ **Database:** MSSQL SQL
- ◆ **Backend Framework:** ASP.NET Core
- ◆ **Authentication:** JWT
- ◆ **Frontend:** Angular (for dashboard visualization)
- ◆ **Version Control:** GitHub (<https://github.com/Abdurrahman-gurib/MCB-Assignment-2025>)
- ◆ **Testing:** Unit Testing for APIs and Database Queries