

Shop Bridge API

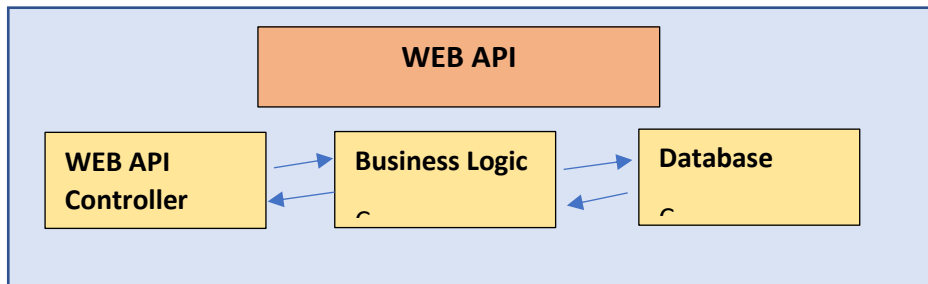
Database: SQL Server

Back end: .NET 6 Web API

ORM Frameworks used: Entity Framework core, Dapper

Required Software: SQL Server 19, Visual Studio 2022

Architecture Pattern followed:



Implementation Details:

1. Created the entity models and the dbContext class for ShopBridge API
2. Generated tables using Code First Approach in Entity Framework Core
3. Used migration to seed initial data for product and category table
4. Implemented Create, Update and Delete operations using Entity Framework core ORM.
5. Used Dapper for the GET operations as it is faster than Entity Framework Core and effective to implement different filter operations.
6. Used OAuth to implement Role-based access control for the endpoints.

Dependencies:

1. Microsoft.EntityFrameworkCore. -Version 7.0.2
2. Microsoft.EntityFrameworkCore.Tools -Version 7.0.2
3. Microsoft.EntityFrameworkCore.SqlServer – Version 7.0.2
4. Dapper - version (2.0.123)

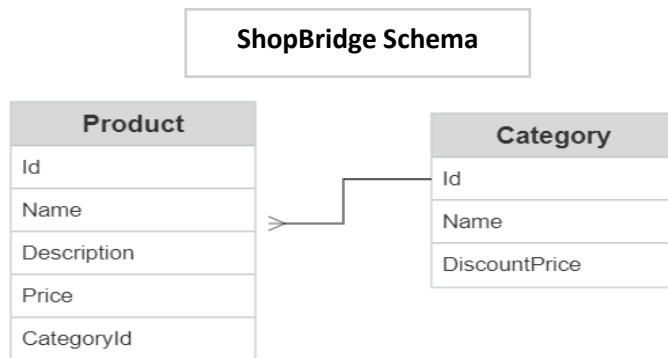
Other Nuget Packages used:

1. Fluent Validation (v11.4.0) – for validation the add and update request data
2. AutoMapper (v12.0.1) – to map between the entity model and the DTOs.
3. Newtonsoft.json (v13.0.2) – to handle json serialization operations

Database: ShopBridgeDatabase

Tables created: Product, Category

ER Diagram:



API Endpoints:

Get Products

1. Filter by category

Request URL

https://localhost:7069/api/Product?Categories=Books

Server response

Code

Details

200

Response body

```
[
  {
    "productId": 1,
    "productName": "Sapiens - By Yuval Noah Harari",
    "description": "A Brief History of Human Kind",
    "price": 460,
    "discountPrice": 450.8,
    "categoryName": "Books"
  },
  {
    "productId": 2,
    "productName": "Atomic Habits - By James Clear",
    "description": "An Easy and Proven way to create good habits",
    "price": 350,
    "discountPrice": 343,
    "categoryName": "Books"
  },
  {
    "productId": 3,
    "productName": "The Psychology of Money - By Morgan Housel",
    "description": "Timeless lessons on wealth, greed and Happiness",
    "price": 420,
    "discountPrice": 411.6,
    "categoryName": "Books"
  },
  {
    "productId": 4,
    "productName": "The Psychology of Money - By Morgan Housel",
    "description": "Timeless lessons on wealth, greed and Happiness",
    "price": 420,
    "discountPrice": 411.6,
    "categoryName": "Books"
  }
]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 05 Feb 2023 06:18:18 GMT
server: Kestrel
```

2. Common Search

Request URL

```
https://localhost:7069/api/Product?SearchTerm=New
```

Server response

Code	Details
200	<p>Response body</p> <pre>[{ "productId": 13, "productName": "New Product - book", "description": "", "price": 250, "discountPrice": 245, "categoryName": "Books" }, { "productId": 15, "productName": "Test Product - Furniture", "description": "New", "price": 950, "discountPrice": 0, "categoryName": "Furniture" }]</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 05 Feb 2023 06:36:42 GMT server: Kestrel</pre>

3. Sort by Price in Descending order

Request URL

```
https://localhost:7069/api/Product?SortProperty=2&SortDirection=DESC
```

Server response

Code	Details
200	<p>Response body</p> <pre>[{ "productId": 7, "productName": "Homeify Sofa set", "description": "Seating Capacity - 8", "price": 40000, "discountPrice": 0, "categoryName": "Furniture" }, { "productId": 6, "productName": "Wooden TV Unit", "description": "Perfect TV Unit for your home", "price": 18000, "discountPrice": 0, "categoryName": "Furniture" }, { "productId": 10, "productName": "Godrej Refrigerator", "description": "Environment friendly Double door refrigerator", "price": 18000, "discountPrice": 0, "categoryName": "Home Appliances" }, { "productId": 9, "productName": "XYZ Wet Grinder", </pre>

4. Simple Pagination

Curl

```
curl -X 'GET' \
  'https://localhost:7069/api/Product?PageSize=1&PageNumber=5' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7069/api/Product?PageSize=1&PageNumber=5
```

Server response

Code	Details
200	<p>Response body</p> <pre>[{ "productId": 5, "productName": "Wooden Wall Shelf", "description": "Decorative set of shelves", "price": 2000, "discountPrice": 0, "categoryName": "Furniture" }]</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 05 Feb 2023 06:39:00 GMT server: Kestrel</pre>

Get By Id

Request URL

```
https://localhost:7069/api/Product/2
```

Server response

Code	Details
200	<p>Response body</p> <pre>{ "productId": 2, "productName": "Atomic Habits - By James Clear", "description": "An Easy and Proven way to create good habits", "price": 350, "discountPrice": 343, "categoryName": "Books" }</pre> <p>Response headers</p> <pre>content-type: application/json; charset=utf-8 date: Sun, 05 Feb 2023 06:20:32 GMT server: Kestrel</pre>

Create Product

1. Given invalid request

Curl

```
curl -X 'POST' \
  'https://localhost:7069/api/Product' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productName": "Test Product 12",
    "description": "",
    "price": 0,
    "categoryName": "Furniture"
  }'
```

Request URL

https://localhost:7069/api/Product

Server response

Code	Details
422	Error: response status is 422

Response body

```
Validation failed:
-- Price: 'Price' must be greater than '0'. Severity: Error
-- CategoryName: Please specify a valid category Severity: Error
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Sun, 05 Feb 2023 06:26:45 GMT
server: Kestrel
```

2. Given valid request

Curl

```
curl -X 'POST' \
  'https://localhost:7069/api/Product' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productName": "Test Product 12",
    "description": "",
    "price": 1200,
    "categoryName": "Furniture"
  }'
```

Request URL

https://localhost:7069/api/Product

Server response

Code	Details
201	Response headers

Response headers

```
content-length: 0
date: Sun, 05 Feb 2023 06:28:01 GMT
server: Kestrel
```

Update Product

1. Invalid Put Request body

Curl

```
curl -X 'PUT' \
  'https://localhost:7069/api/Product/10015' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productName": "string",
    "description": "string",
    "price": 0,
    "categoryName": "string"
  }'
```

Request URL

https://localhost:7069/api/Product/10015

Server response

Code	Details
422	Error: response status is 422

Response body

```
Validation failed:
-- ProductName: 'Product Name' must be between 10 and 50 characters. You entered 6 characters. Severity: Error
-- Price: 'Price' must be greater than '0'. Severity: Error
-- CategoryName: Please specify a valid category Severity: Error
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Sun,05 Feb 2023 06:30:50 GMT
server: Kestrel
```

2. Invalid Product Id:

Curl

```
curl -X 'PUT' \
  'https://localhost:7069/api/Product/10015' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productName": "Test Product 12",
    "description": "",
    "price": 1200,
    "categoryName": "Furniture"
  }'
```

Request URL

https://localhost:7069/api/Product/10015

Server response

Code	Details
404	Error: response status is 404

Response body

```
Product not found for the given Id
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Sun,05 Feb 2023 06:32:03 GMT
server: Kestrel
```

3. Valid Put Request

Curl

```
curl -X 'PUT' \
  'https://localhost:7069/api/Product/1001' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productName": "Test Product 12",
    "description": "New Product",
    "price": 1200,
    "categoryName": "Furniture"
  }'
```

Request URL

<https://localhost:7069/api/Product/1001>

Server response

Code	Details
204	<p>Response headers</p> <pre>date: Sun, 05 Feb 2023 06:33:44 GMT server: Kestrel</pre>

Delete Product

1. Invalid Product Id

Curl

```
curl -X 'DELETE' \
  'https://localhost:7069/api/Product/10015' \
  -H 'accept: */*' \
```

Request URL

<https://localhost:7069/api/Product/10015>

Server response

Code	Details
404	<p>Error: response status is 404</p> <p>Response body</p> <pre>Product not found for the given Id</pre> <p>Download</p>

2. Valid Product Id

Curl

```
curl -X 'DELETE' \
  'https://localhost:7069/api/Product/1001' \
  -H 'accept: */*' \
```

Request URL

<https://localhost:7069/api/Product/1001>

Server response

Code	Details
204	<p>Response headers</p> <pre>date: Sun, 05 Feb 2023 06:35:11 GMT server: Kestrel</pre>

Implementation In Progress

Add role based authorization to allow only users with **ProductAdmin** Role to access the CREATE, UPDATE and DELETE endpoints.

Users with **ProductUser** can only access the GET endpoints.

Roles Created in OAuth:

The screenshot shows the OAuth console interface for the 'dev-xfx4kbp81aq1x4dj' application in 'Development' mode. The left sidebar contains navigation links: Getting Started, Activity (EARLY), Applications, Authentication, Organizations, User Management (selected), Users, Roles, Branding, Security, Actions, Auth Pipeline, Monitoring, Marketplace, Extensions, Settings, and Get support. The main content area is titled 'ProductAdmin' with Role ID 'ro1_Jc2kj8cRyug4Zi2u'. It has tabs for Settings, Permissions (selected), and Users. A warning message states: 'Warning! Your API ShopBridge API does not have RBAC enabled. Roles and permissions will not be evaluated during the authorization transaction. You can enable this on the API settings page under Access Settings.' Below this, a text block says: 'Add Permissions to this Role. Users who have this Role will receive all Permissions below that match the API of their login request.' There is an 'Add Permissions' button. A table lists the permissions:

Permission ^	Description	API	
create:products	create products permission	ShopBridge API	
delete:products	Delete permission	ShopBridge API	
get:products	Fetch permission	ShopBridge API	
update:product	Update permission	ShopBridge API	

The screenshot shows the OAuth console interface for the 'dev-xfx4kbp81aq1x4dj' application in 'Development' mode. The left sidebar is the same as the previous screenshot. The main content area is titled 'ProductUser' with Role ID 'ro1_w8sV1tex8380eDr'. It has tabs for Settings, Permissions (selected), and Users. A warning message states: 'Warning! Your API ShopBridge API does not have RBAC enabled. Roles and permissions will not be evaluated during the authorization transaction. You can enable this on the API settings page under Access Settings.' Below this, a text block says: 'Add Permissions to this Role. Users who have this Role will receive all Permissions below that match the API of their login request.' There is an 'Add Permissions' button. A table lists the permissions:

Permission ^	Description	API	
get:products	Fetch permission	ShopBridge API	

Created Users and Assigned them roles:

[← Back to Users](#)

JA

janani@shopbridge.com

user_id: auth0|63deb5598bfabce77af606ff

Actions ▾

Details

Devices

History

Raw JSON

Authorized Applications

Permissions

Roles

All Roles assigned to this User.

Assign Roles

Name	Description	Assignment
ProductAdmin	Access to Product module	Direct

[← Back to Users](#)

JA

jay@shopbridge.com

user_id: auth0|63deb582641e1d30b85c62dc

Actions ▾

Details

Devices

History

Raw JSON

Authorized Applications

Permissions

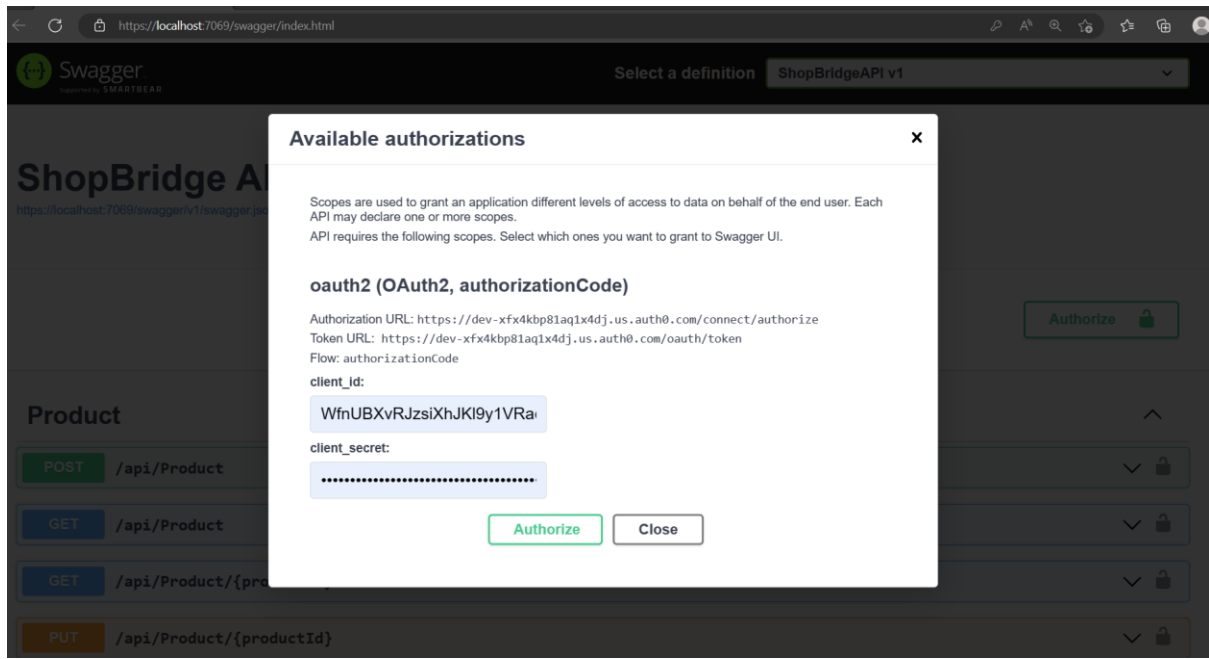
Roles

All Roles assigned to this User.

Assign Roles

Name	Description	Assignment
ProductUser	General user	Direct

Incorporated OAuth2 Swagger Login Flow:



Configuring the OAuth Authorization in .net Application:

```
builder.Services.AddAuthorization(options =>
{
    options.AddPolicy("CreateAccess", policy =>
        policy.RequireClaim("permissions", "create:products"));
    options.AddPolicy("UpdateAccess", policy =>
        policy.RequireClaim("permissions", "update:product"));
    options.AddPolicy("DeleteAccess", policy =>
        policy.RequireClaim("permissions", "delete:products"));
    options.AddPolicy("GetAccess", policy =>
        policy.RequireClaim("permissions", "get:products"));
});
```

```

// Fetch product by Id
[Authorize(Roles = "ProductUser")]
//[Authorize(Policy = "GetAccess")]
[HttpGet("{productId:int}")]
[ProducesResponseType(typeof(Product), StatusCodes.Status200OK)]
[ProducesResponseType(typeof(string), StatusCodes.Status404NotFound)]
[ProducesResponseType(typeof(string), StatusCodes.Status500InternalServerError)]
0 references
public async Task<ActionResult> GetAsync(int productId)
{
    try
    {
        if (productId == 0)
        {
            return BadRequest(ErrorMessages.InvalidProductId);
        }

        var result = await _getProductService.GetByIdAsync(productId);
        return Ok(result);
    }
    catch (EntityNotFoundException ex)
    {
        _logger.LogError(ex.Message);
        return NotFound(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError, ex.Message);
    }
}

```

I am facing some issues with validating the JWT token generated from the login flow in the ShopBridge API. I may need to spend some more time on this to make it work.