Asp .Net Core Authentication

Implement authentication and authorization with role-based access control in an ASP.NET Core API for an order and sales management system:

1. **Setup Authentication and Authorization:**
   - Install necessary NuGet packages like `Microsoft.AspNetCore.Authentication.JwtBearer` and `Microsoft.AspNetCore.Authorization`.

2. **Define Roles**:
   - Determine the roles needed for your system (e.g., Admin, Manager, Salesperson, Customer).
   - Define these roles in your application, either through an enum or database.

3. **Implement Authentication:**
   - Configure JWT authentication in `Startup.cs` to issue JWT tokens upon successful authentication.
   - Implement user authentication using Identity framework or any custom authentication mechanism.
   - Return a JWT token upon successful authentication.

4. **Implement Authorization:**
   - Define policies based on roles in `Startup.cs` or using policy-based authorization attributes.
   - Apply authorization checks on controllers or actions using `[Authorize]` attribute or `AuthorizeAttribute` with specific policies.
   - Use JWT claims to enforce authorization based on roles.

5. **Role Assignment:**
   - Implement a way to assign roles to users (e.g., Admin assigns roles to users).
   - Create API endpoints for role management.

6. **Secure Endpoints:**
   - Secure API endpoints based on roles using `[Authorize(Roles = "RoleName")]` attribute or policy-based authorization.
   - Define what actions each role can perform (e.g., only Admin can access certain endpoints).

7. **Test**:
   - Test authentication by obtaining a JWT token through login API endpoint.
   - Test authorization by attempting to access API endpoints with different roles to ensure proper access control.

8. **Error Handling:**
   - Implement error handling for unauthorized access attempts (e.g., return 401 Unauthorized status code).

9. **Logging and Monitoring:**
   - Implement logging and monitoring to track authentication and authorization events for auditing purposes.

10. **Documentation:**
    - Document the authentication and authorization process for future reference and maintenance.

Break down of the roles and corresponding endpoints for an order and sales management system:

1. **Roles:**
   - **Admin:** Has full access to all functionalities of the system.
   - **Manager:** Manages sales teams, products, and customer accounts.
   - **Salesperson:** Creates and manages orders for customers.
   - **Customer:** Views order history and manages their own account details.

2. **Endpoints:**

   - **Authentication:**
     - `/api/auth/login`: POST request to authenticate users and obtain JWT token.
     - `/api/auth/register`: POST request to register new users.

   - **User Management:**
     - `/api/users`: GET request to retrieve all users (Admin only).
     - `/api/users/{id}`: GET request to retrieve a specific user details (Admin only).
     - `/api/users/{id}`: PUT request to update user details (Admin only).
     - `/api/users/{id}`: DELETE request to delete a user (Admin only).

   - **Role Management:**
     - `/api/roles`: GET request to retrieve all roles (Admin only).
     - `/api/roles/{id}`: GET request to retrieve a specific role details (Admin only).
     - `/api/roles/{id}`: PUT request to update role details (Admin only).
     - `/api/roles/{id}`: DELETE request to delete a role (Admin only).

   - **Product Management:**
     - `/api/products`: GET request to retrieve all products.
     - `/api/products/{id}`: GET request to retrieve a specific product details.
     - `/api/products`: POST request to create a new product (Manager only).
     - `/api/products/{id}`: PUT request to update product details (Manager only).
     - `/api/products/{id}`: DELETE request to delete a product (Manager only).

   - **Order Management:**

  - `/api/orders`: GET request to retrieve all orders (Manager and Salesperson).
  - `/api/orders/{id}`: GET request to retrieve a specific order details (Manager and Salesperson).
  - `/api/orders`: POST request to create a new order (Salesperson only).
  - `/api/orders/{id}`: PUT request to update order details (Salesperson only).
  - `/api/orders/{id}`: DELETE request to delete an order (Salesperson only).

- **Customer Management:**
  - `/api/customers`: GET request to retrieve all customers (Manager only).
  - `/api/customers/{id}`: GET request to retrieve a specific customer details (Manager only).
  - `/api/customers`: POST request to create a new customer (Manager only).
  - `/api/customers/{id}`: PUT request to update customer details (Manager only).
  - `/api/customers/{id}`: DELETE request to delete a customer (Manager only).
  - `/api/customers/{id}/orders`: GET request to retrieve all orders for a specific customer (Manager and Salesperson).

Each endpoint is secured with appropriate authorization checks based on roles to ensure that only authorized users can access them.