SW4BAD mandatory assignment 3: **Authentication and MongoDB**

Remember to read the whole assignment before you start working and before you hand in, so you are sure you hand in the correct files.

A. Authentication and authorization

In this part, you will increase the security of your solution to assignment 2. You should add login, register endpoints, and implement logic to satisfy the following requirements below. The following authorization logic should be implemented:

1. There are 4 roles in the system: Admin, Manager, Cook, Cyclist
2. Login and Query 2 are the only endpoints accessible to anonymous users.
3. Query 1 is only available to Managers.
4. Query 5 is only available to Cooks.
5. Query 6 is only available to Cyclists
6. Query 5 and 6 endpoints only show data related to the currently authenticated user.
7. Admin users can use all query endpoints.

Your solution should have a way to seed the database with an Admin user. To simplify your testing, you can seed one test user for each of the other roles.

Technology requirements: You must use Microsoft Identity with JWT.

Deliverable: As part of C. - see later

B. Log to MongoDB

In this part you must add logging to your web API. All POST, PUT and DELETE requests must be logged to a MongoDB database.

Technology requirements: you must use Serilog to MongoDB Using Configuration.
Hints: https://github.com/ChangemakerStudios/serilog-sinks-mongodb
https://www.adamrussell.com/asp-net-core-logging-with-serilog-to-mongodb-using-configuration
https://www.youtube.com/watch?v=lgujcmQGuVQ

Deliverable: As part of C. - see later

C. View and search in logs

In this part you must add to your solution an endpoint that can be used to search the mongo database for log entries. Only admin users should be able to use this endpoint. It should be possible to filter for a specific user, a time interval and/or type of operation (POST, PUT DELETE).

Deliverable: A cleaned and zipped .NET Solution into a filed named BAD_MA3_Solution_grp<grp-no>.zip (e.g., **BAD_MA4_Solution_grp42.zip**)

D. Functional and security testing

In this part you will test your API using Postman. Your testing should cover the following requirements:

1. Each of the query endpoints must be tested at least once.
2. The tests check if the authorization logic is well implemented.

Deliverable: Postman collection file(s) to be included in the zip produced in C.

E. Release

In this part you must write a Docker file that will containerize your ASP solution, and a docker compose file that will spin up your API container and the two databases.

Deliverable: Docker and compose files to be included in the zip produced in C.

F. Bonus (Optional)

Automate the execution your postman tests with docker compose

Hint: https://learning.postman.com/docs/collections/using-newman-cli/newman-with-docker

Optional Deliverable: The same with the added docker compose.

Hand-in "formalities"

The same rules of Assignment 2 apply.