Homework for applicants:

Description:

1. Implement a C# Web API using .NET or .NET Core using the requirements listed below
2. Showcase the following (each one is optional but nice to have):
   1. Usage of Dependency Injection
   2. Generics, extension methods
   3. Async programming
   4. Multi-threading
   5. Exceptions handling
   6. JSON serialization/deserialization
3. Benefits:
   1. Add Swagger support
   2. Add Postman support
   3. Attributes creation & usage
   4. Middleware and filter usage
   5. Unit tests

Requirements (the more implemented, more score you will get):

Implement an API capable of generating and returning a subsequence from a sequence of Fibonacci numbers. The API should have a controller with an endpoint accepting the following parameters:

1. The index of the first number in Fibonacci sequence that starts subsequence.
2. The index of the last number in Fibonacci sequence that ends subsequence.
3. A Boolean, which indicates whether it can use cache or not.
4. A time in milliseconds for how long it can run. If generating the first number in subsequence takes longer than that time, the program should return error. Otherwise as many numbers as were generated with extra information indicating the timeout occurred.
5. A maximum amount of memory the program can use. If, during the execution of the request this amount is reached, the execution aborts. The program should return as many generated numbers similarly to the way it does in case of timeout reached.

The return from the endpoint should be a JSON containing the subsequence from the sequence of Fibonacci numbers that is matching the input indexes.

The controller that accepts requests could use async pattern. It could schedule the work on two background threads and wait for results asynchronously. The generation of Fibonacci numbers could happen on at least two background threads, where the next number in sequence could be generated on a different thread.

Please bear in mind, there could be many requests landing simultaneously and those could use the same background threads that are executing already.

There could be a cache for numbers, so that subsequent requests can rely on it in order to speed up the Fibonacci numbers generation.

The cache could be invalidated after a time period, where the period is defined in configuration.