Practical Exercise – Back-End Developer

Imagine that you have launched a start-up company, with the intention of selling a robotic road-monitoring solution to Transmax. You have created two pieces of hardware:

EagleBot™

- The EagleBot is a robotic drone which can hover above a road segment and use image processing algorithms to monitor the traffic conditions. Its primary function is to report traffic throughput, but it can additionally report on traffic statistics.

EagleRock™

- The EagleRock is a charging and maintenance hub for a group of EagleBots. The EagleRock also serves as an operations control centre. Presently your start-up is only two employees, so you work out of the EagleRock.

Your intention is to sell the solution to Transmax, so that they will own and operate three EagleBots as well as an EagleRock operations hub. Please complete the following tasks for Transmax to assess the viability of your start-up's solutions.

Task 1: EagleRock Service and APIs (mandatory)

Develop a simple back-end service (ideally a containerized C#/.NET Core app) that provides the capability for EagleBots to send data to EagleRock, composed of:

1. a REST API that EagleBots may periodically call to transfer traffic data to the EagleRock
2. a Redis (or equivalent) in-memory data store that the received traffic data is cached into
3. a second (aggregate) REST API that a web application might use to get the current location, status and latest traffic data from all active EagleBots via the Redis cache
4. appropriate unit tests to validate that each component operates as expected

The traffic data payload should be serialized/de-serialized using an industry-recognized, self-describing, language-independent format that can easily be extended. In its first iteration, the content is expected to include the following data elements:

- EagleBot unique identifier
- Current location/co-ordinates of the EagleBot
- Timestamp for the data exchange
- Road name under inspection
- Direction of traffic flow
- Rate of traffic flow
- Average vehicle speed

Task 2: EagleRock Data Stream (for bonus points)

Extend the EagleRock service to offer an event-driven data stream for an additional type of consumer, by means of:

1. received EagleBot data also being written to a RabbitMQ topic (or similar publish/subscribe-based queue) that external applications can read from

In Conclusion

Think of this like a MasterChef challenge: get as close as you can with the time and tools you have available, but if you don't complete it perfectly we still want to see how close you got and what your code looks like. We are looking for the following:

- Skills in C#/,NET Core application design and implementation
- Skills in utilizing cloud service components such as in-memory caches and queues
- Skills in defining and implementing data payloads, and APIs such as REST
- Skills in writing unit tests