Assessment (Interview Pre-Assignment)

Scenario

Design an architecture with **at least two microservices** that communicate with each other, and build a simple **phonebook** application.

Assessment

Expected Functions

- Create a person in the phonebook
- Remove a person from the phonebook
- Add a contact info entry to a person
- Remove a contact info entry from a person
- List people in the phonebook
- Get a person's **details including contact info**
- Request a **report** that produces statistics of people by their **location**
- List the reports produced by the system
- Get the **details** of a specific report

Technical Design

People

The system should allow a **theoretically unlimited** number of people. Each person can have an **unlimited** number of contact info entries.

Required data fields:

- UUID
- First Name
- Last Name
- Company
- Contact Info
    - Type**:** Phone Number, E-mail Address, Location
    - Content

Report

Report requests work **asynchronously**. When a user requests a report, the system should handle this work **in the background**, in an ordered manner without creating bottlenecks. Once the report is completed, the user should be able to observe its status as **“Completed”** via the endpoint that **lists reports**.

A report contains:

- Location
- Number of people in that location
- Number of phone numbers in that location

Report data structure:

- UUID
- Requested At (timestamp)
- Report Status (Preparing, Completed)

Technical Expectations

Technologies to Use

- .NET Core
- Git
- PostgreSQL or MongoDB
- Kafka or a similar Message Queue system

Constraints & Requirements

- Develop the project on Git with frequent commits
- Use master and development branches and version tagging
- Minimum 60% unit testing code coverage
- Include a migration setup to create the database
- Provide a README.md describing how to run the project
- Services must communicate over HTTP via REST or GraphQL
- The asynchronous part of reporting must use a message queue (or equivalent)