**Varo | Backend Engineer | Take-Home project**

This assignment should take about 2-4 hours to complete.

**TASKS:**

1. **Problem Statement:** As a user, I should be able

* to register into the system with my first name, last name, email and multiple addresses with information like line 1, line 2, city, state, zip.
* to get the list of all the users in the system.
* to search a user by email and confirm if the user exists.
* to edit the address of the user but retain the old address.
* to update user information.
* to delete users from the system but am able to audit the deletion of users.

1. Create an architectural diagram demonstrating the workflow of your solution to the problems below.
2. Create **GRPC / REST API using Kotlin / Java** microservice with distinct proto’s/data model and service with gradle or maven dependency management system.
3. Microservice will support below endpoints
   1. **UpsertUser** API to update user if exists else create a new
   2. **IsUserExists** API which takes input as email and returns appropriate status
   3. **ListUsers** API to return list of all users
   4. **GetUserDetails** API inputs user id and returns user details including address.
   5. **DeleteUser** API which takes user id and marks the user as deleted.
   6. **UpdateAddress** API which takes user id and updates user address such that the old address is archived.
4. Your goal is to architect a solution that will
   1. Create right protoBuf/data model for required API
   2. Demonstrate OOP principles along with separation of concerns.
   3. Demonstrate use of design patterns necessary for the solution.
   4. Plan to include indexes to optimize query performance.
   5. Create unit tests and integration tests to test the system behavior.

**RULES:**

1. Please include the database migration script.
2. You are welcome to use any open source libraries and/or create additional services to accomplish your tasks.
3. You are open to use any database, preferred Postgres
4. Setup database connections with below settings or provide ability to configure connection via env variables url host - localhost, dbname - backend\_test, username - varo, password - varo.

**SUBMISSION:**

1. Include architecture-diagram.png sketch showing high level architecture.
2. Create a README.md with the following:
   1. Instructions on how to start your service(s)
   2. Provide instructions to set up the connection string.
   3. Provide a list of grpcurl / curl API to test services.
3. Zip your challenge and submit or share a Github or any other repo url that is accessible publicly.

**EVALUATION:**

1. We will spin up the service(s) that you will have created.
2. Make grpc/rest curl calls as per the list of api that you have provided.

**EXTENDED GOALS:**

1. Add fault tolerance if the database is down or timeout occurs, so that the changes for user and address are persisted.
2. Plan for scale on how this system can support millions of users.
3. Make the updates to the database auditable.
4. Make the db migrations repeatable and reversible.
5. Dockerize your application to execute with simplicity - provide docker commands if you decide to go.