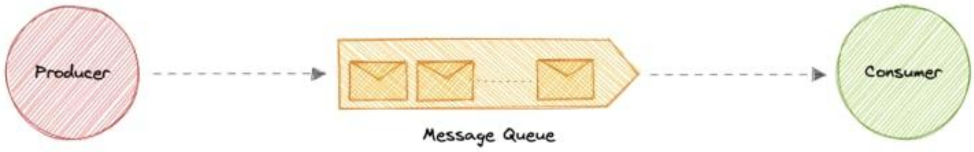
Golang backend assignment

Assignment:

Return your answer as a zip file containing all relevant files _with tests_ (including `.git`, so that we can see your commit history). If you create your own git repo, please make sure it is private, so that the other candidates cannot access your solution.

Design and implement (with tests) a _message queuing system_ using Go programming language and RabbitMQ or Kafka.



We don't value over-engineering.
Provide a readable minimalistic implementation that has understandable split to well-named source files and functions.
Impress us with simplicity, good unit tests and a working solution.

In this simplified scenario the system includes the following parts:

API: Design an API where it should receive a product data and store in the database, below are the parameters that should be passed in the API

- user_id (create users table and primary key of that table)
- product_name
- product_description (text)
- product_images (array of image urls)
- product_price (Number)

Producer: After storing the product details in the database, product_id should be passed on to the message queue.

Consumer: based on the product_id, product_images should be downloaded and compressed and stored in local. After storing, a local location path should be added as an array value in the products table in the compressed_product_images column.

Database Schema:

Users: (data for the table can be added manually)

1. id - int, primary key
2. name - Name of the users
3. mobile - Contact Number of the user
4. latitude - Latitude of the user’s location
5. longitude - Longitude of the user’s location
6. created_at
7. updated_at

Products: (Data should be added from the API Only)

1. product_id - int, primary key
2. product_name - string, Name of the product
3. product_description - text, About your product
4. product_images - array
5. product_price - number
6. compressed_product_images - array
7. created_at
8. updated_at

Testing

The project should contain integration and benchmark tests.
Please add unit tests for your implementation, without them the assignment will be rejected.