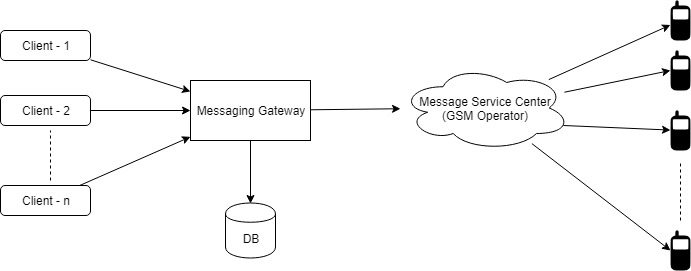
**2-) Bulk Message Delivery Application**

Design a bulk message gateway application using OOP principles. See the following image for the platform structure.



You are required to write an application providing an API to which multiple clients will connect and make requests (e.g. HTTP) for message deliveries and your platform will be forwarding messages to Message Service Center (GSM Operator). Your application must have the following features:

* An API (service class eg. MessageService) which will handle incoming requests.
* Add/remove new customers (clients).
* By using the API, connected clients will make a message delivery request by providing originating address (cell phones will see that number as the message sender), a list of recipients, message content etc. Your application must validate the message content and length (e.g. requests with empty/null messages must return error, maximum message length can be 1024 characters etc.). Your API must return the id of each request to the requesting client so that clients will make queries (get reports) about their own requests with their corresponding ids.
* Since the API will support multiple clients, clients should authenticate themselves to the API and each delivery request must be associated to the requesting client.
* Deliveries must have a start and end time. In other words, the API should support scheduling the delivery for a later time, so that platform will start forwarding messages to GSM network at that time. Likewise, deliveries should have an expiration time, meaning that no more delivery for that request will be made after that time.
* API should support cancelling deliveries which are unfinished or have not started. Delivery attempt to cell phones should not be made for cancelled deliveries.
* API should support retrying deliveries for failed attempts (e.g. number unreachable, network error) with a maximum of 3 retries and wait time of 1 minute between each. Retries should be done automatically according to the response code received from Message Service Center and there should be 1 minute of wait time between each retry.
* Each delivery request details (request date, message content, recipient list etc.) made by the clients must be stored in the database.
* Result of each submission request made by the platform towards cell phones must be stored in the database.
* Each client should be able to query for their own requests (request date, request status (Completed/Aborted etc.), number of recipients for the requests etc.) and delivery results. A client should not be able to query other clients’ requests.
* API should provide methods for the following queries:
  + Number of requests made to the API per client within a time interval.
  + Number of successful/failed messages sent to Message Service Center within a time interval.
  + Number of failed messages per error code within a time interval.
  + Number of failed messages per error code and per client within a time interval.
* Number of recipients for each request must have a limit of 10 messages.
* Each client must have a daily message quota and it should be defined separately for each client in the database.
* Provide necessary classes and database scripts for the application including a user manual explaining how to run/test the application.
* Use line/block comments in your code where necessary.
* Validate user input to handle invalid requests.
* You can use a self-contained database such as SQLite or H2.
* Use the following class to simulate the message submission to Message Service Center.

/\*\*

\* Message Service Center interface which simulates a GSM operator.

\*/

**public** **class** MessageServiceCenter {

/\*\*

\* Sends a message submission request to the GSM operator's network. Later GSM operator

\* will forward the request to the recipient and returns the result code of the operation.

\*

\* Available result codes:

\* 0: Success

\* 1: Number unreachable

\* 2: Network error

\* 3: Unknown error

\*

\* **@param** senderAddress The number which will be shown on recipient's cell phone.

\* **@param** destinationNumber Recipient's cell phone number.

\* **@param** messageBody Message text to be sent to the recipient.

\* **@return**

\*/

**public** **int** submitMessage(String senderAddress, String destinationNumber, String messageBody) {

// For simplicity return a random number between 0 and 3 as a result code

**return** (**int**) (Math.*random*() \* 4);

}

}

* You are not required to provide a Restful/SOAP API, you can write a console application to make requests to the API.

You are required to submit a working code including all class(es) and a user manual explaining how to test the code. Make sure that your solution handles invalid inputs as well.