FETCH REWARDS ASSIGNMENT

1. How would you deploy this application in production?

To deploy this application in production, we would need to package the application code along with its dependencies into a Docker container. We could use YAML files to define the container and its dependencies, as well as the necessary resources such as volumes, secrets, and environment variables. We would also need to set up monitoring and logging infrastructure to be able to monitor the application's performance and troubleshoot any issues that arise.

2. What other components would you want to add to make this production ready?

To make this application production-ready, we would want to add components such as load balancing and auto-scaling to handle increased traffic and prevent downtime. We would also want to add a caching layer such as Redis to reduce the number of database calls and improve performance. We would also need to set up backups and disaster recovery procedures to ensure that we can recover from any potential data loss or system failure.

3. How can this application scale with a growing dataset?

To scale this application with a growing dataset, we could use partitioning to split the database into smaller, more manageable pieces. We could also use a distributed caching layer such as Apache Ignite or to distribute the cache across multiple servers. We could also optimize database queries and use indexing to improve query performance.

4. How can PII be recovered later on?

In this case, the PII is masked before being stored in the database. Therefore, if there is a need to recover the original PII data, the masked values will have to be reverse engineered to obtain the original values. This process may involve using a mapping table that associates the masked values with the original values. However, recovering PII data should only be done if necessary and should be subject to appropriate security measures and data protection regulations. It is also important to ensure that the recovery process does not compromise the privacy and confidentiality of the data subjects.

5. What are the assumptions you made?

In developing this solution, I assumed that the data in the message queue would be in the correct format and that there would be no unexpected errors or exceptions. I also assumed that the JSON data would fit into memory, and that the number of messages in the queue would not be too large to cause performance issues. Additionally, I assumed that the database schema provided was accurate and that there were no issues with the database connection.

TERMINAL OUTPUT:

```
app % python login_processor.py

 app % docker ps

CONTAINER ID   IMAGE                              COMMAND
       CREATED       STATUS         PORTS
                NAMES

32a13297c160   fetchdocker/data-takehome-postgres
"docker-entrypoint.s…"   24 hours ago   Up 11 minutes
0.0.0.0:5432->5432/tcp                            app-postgres-1

3254a12fa635   fetchdocker/data-takehome-localstack
"docker-entrypoint.sh"   24 hours ago   Up 15 minutes   4510-4559/tcp,
5678/tcp, 0.0.0.0:4566->4566/tcp   app-localstack-1

app % awslocal sqs receive-message --queue-url
http://localhost:4566/000000000000/login-queue


{

    "Messages": [

        {
            "MessageId": "2222bc56-8d68-4a7e-b878-a03cf1467714",

            "ReceiptHandle":
"YWYyZTAzNDgtNTlmMy00OTcyLWI3NDUtOTVhZWI0ODE4NDU1IGFybjphd3M6c3Fz
OnVzLWVhc3QtMTowMDAwMDAwMDAwMDA6bG9naW4tcXVldWUgMjIyMmJjNTYtO
GQ2OC00YTdlLWI4NzgtYTAzY2YxNDY3NzE0IDE2Nzc1NTUzNDAuNDMxODYyNA==
",

            "MD5OfBody": "e4f1de8c099c0acd7cb05ba9e790ac02",

            "Body": "{\"user_id\":
\"424cdd21-063a-43a7-b91b-7ca1a833afae\", \"app_version\": \"2.3.0\",
\"device_type\": \"android\", \"ip\": \"199.172.111.135\", \"locale\":
\"RU\", \"device_id\": \"593-47-5928\"}"

app % docker exec -it app-postgres-1 psql -d postgres -U postgres -p
5432 -h localhost -c "SELECT * FROM user_logins;"



user_id | device_type | masked_ip | masked_device_id | locale |
app_version | create_date

---------+-------------+-----------+-----------------+--------+-------------+-------------

 1234    | phone       | MASKED-1  | MASKED-1234      | en     |
    1 | 2022-01-01

 5678    | tablet      | MASKED-2  | MASKED-5678      | fr     |
    2 | 2022-01-02

 9012    | phone       | MASKED-3  | MASKED-9012      | de     |
    3 | 2022-01-03

 1234    | phone       | MASKED-1  | MASKED-1234      | en     |
    1 | 2022-01-01

 5678    | tablet      | MASKED-2  | MASKED-5678      | fr     |
    2 | 2022-01-02

 9012    | phone       | MASKED-3  | MASKED-9012      | de     |
    3 | 2022-01-03

 1234    | phone       | MASKED-1  | MASKED-1234      | en     |
    1 | 2022-01-01
 5678    | tablet      | MASKED-2  | MASKED-5678      | fr     |
    2 | 2022-01-02

 9012    | phone       | MASKED-3  | MASKED-9012      | de     |
    3 | 2022-01-03

(9 rows)
```