Engineering Challenge

Create a real-time system that emulates the charges calculation and invoice generation for a payment system. The system will be receiving the transaction details with status (approve or reject) in the below format B. Invoicing system will process these transactions, store them in the database, and calculates the charges based on the card type, currency, and client.

Based on the billing cycle of the client, the system will be generating the invoice and send an email to the email address of the client.

For testing purposes, please create an API to trigger the billing cycle for specific or all clients. Transactions should not be billed multiple times to clients.

You are required to create a database using MySQL, APIs (client management (add, update, delete, list), record payment/transaction *add, update, delete, list), generate invoice), timer routine for bill generation, on above requirements and share your assumptions.

Some details on Fees calculation:

- Clients will be charged flat fees per successful / approved transaction.
- For refund transactions, fees collected already will be returned back to the client.
- For refund transactions, if transactions are not billed they will not be charged to the client.
- If the approved transaction, then it will be billed to the client as per the amount of the fee
- If declined transactions, then no fees are to be charged.
- Refund cannot be done for a declined transaction.
- Order id is unique in the system

Grading:

You are expected to build a system in java to handle the above scenarios that fulfill each element.

- Meets all specified requirements
- Is valid runnable code (via CLI or within your IDE)
- Has appropriate usage of design patterns, concurrency, and data structures
- Has extendable architecture
- Log output that allows the accessor to clearly understand and follow your system's operations as it runs in real-time
- Has comprehensive unit testing (covers all major components and flows)
- Has production-quality code cleanliness
- Has production-quality docs on all public functions (as if someone had to work with your code)
- Has a README file that contains
    - Instructions on how to run and test your code in a local environment
    - Any other design choices you would like the interviewers to know
    - Relevant git commit messages.

Code Submission

Kindly share your code in a zip file with build instructions

Client List (Format A)

```
[
{
"id": "1",
"client": "Pizza House", // name of the client
"status":"active", // active or disabled
"billing-interval":"daily", // daily / monthly
"email”:"s@s.com", // email address to receive the bills
"fees-type”: "flat-fee",
"fees": "1.00"
}, {
"id":"2",
"client":"Fitness House",
"status":"active", // active or disabled
"billing-interval":"monthly", // daily / monthly
"email":"s@s.com" // email address to receive the bills
"fees-type": "flat-fee",
"fees": "2.00"
},
.....
]
```

Payment Transactions - Format B

```
[
{
"orderid":"123456",
"datetime":"31-May-2022 15:30 pm",
"ordername":"Cheese Pizza",
"amount":"50.00",
"currency":"USD",
"cardtype": "visa", // visa / master
"status":"approved", // status (approved or declined or refunded)
"client":"Pizza House". // name of the client
}, {
"orderid":"123457",
"datetime":"31-May-2022 15:31 pm",
"ordername":"Fitness Pack",
"amount":"100.00",
"currency":"USD",
"cardtype": "master" // visa / master
"status":"approved", // status (approved or declined or refunded)
"client":"Fitness Zone" // name of the client
}
]
```