AnyMind Group Limited (Thailand)
689 Bhiraj Tower at Emquartier, 34th Floor, Sukhumvit Road,
Khlongtan Nuea,
Wattana, Bangkok 10110

Project for Back-end Engineer at AnyMind Group

Deadline:

- 5 days

Project details:

You manage a POS integrated e-commerce platform which offers a **point system** and has a vast collection of payment methods integrated. Some payment methods require a commission fee from the payment provider thus you do not want to provide too much discount on a product if the customer selects that payment method. At the same time, you would want to **control the points given per purchase based on the payment methods** to minimize loss. The following is the list of possible payment methods, rates, points, and additional actions that will be performed based on the payment method.

Payment Method	Possible final price (price_modifier)	Points	Additional items
CASH	price * 0.9 ~ price	price * 0.05	
CASH_ON_DELIVERY	price ~ price * 1.02	price * 0.05	Should accept the following courier services only ● YAMATO ● SAGAWA
VISA	price * 0.95 ~ price * 1	price * 0.03	Should accept and store last 4 digits of the card
MASTERCARD	price * 0.95 ~ price * 1	price * 0.03	Should accept and store last 4 digits of the card
AMEX	price * 0.98 ~ price * 1.01	price * 0.02	Should accept and store last 4 digits of the card
JCB	price * 0.95 ~ price * 1	price * 0.05	Should accept and store last 4 digits of the card
LINE_PAY	price	price * 0.01	
PAYPAY	price	price * 0.01	
POINTS	price	0	

GRAB_PAY	price	price * 0.01	
BANK_TRANSFER	price	0	Should accept and store bank and account number information
CHEQUE	price * 0.9 ~ price	0	Should accept and store bank and cheque number information

Shops and online stores will integrate with your system. To make a payment, the following requests and response should be provided

Example 1

Request	Response

`{`
`    "customerId": "12345",`
`    "price": "100.00",`
`    "priceModifier": 0.95,`
`    "paymentMethod": "MASTERCARD",`
`    "datetime": "2022-09-01T00:00:00Z",`
`    "additionalItem": {`
`        "last4": "1234"`
`    }`
`}`
	`{`
	`    "finalPrice": "95.00",`
	`    "points": 3`
	`}`

Calculation

```
finalPrice = 100 * 0.95 = 95
points = 100 * 0.05 = 5
```

Example 2

Request	Response

`{`
`    "customerId": "12345",`
`    "price": "100.00",`
`    "priceModifier": 1,`
`    "paymentMethod": "CASH_ON_DELIVERY",`
`    "datetime": "2022-09-01T00:00:00Z",`
`    "additionalItem": {`
`        "courier": "YAMATO"`
`    }`
`}`
	`{`
	`    "finalPrice": "100.00",`
	`    "points": 5`
	`}`

Calculation
```
finalPrice = 100 * 1 = 100
points = 100 * 0.05 = 5
```

If the input is invalid, your system should respond in the following format. You are free to design how the error message/object should look like
```
{
    "error": "<please design a suitable error>"
}
```

You should also allow the users of your e-commerce system to see **how much sales were made within a date range broken down into hours**. Your system should be able to show a list of sales and the points given out to the customer.

Request	Response

`{`
`    "startDateTime":`
`"2022-09-01T00:00:00Z",`
`    "endDateTime":`
`"2022-09-01T23:59:59Z"`
`}`
	`{`
	`    "sales": [`
	`        {`
	`            "datetime":`
	`"2022-09-01T00:00:00Z",`
	`            "sales" "1000.00",`
	`            "points": 10`
	`        },`
	`        {`
	`            "datetime":`
	`"2022-09-01T01:00:00Z",`
	`            "sales" "2000.00",`
	`            "points": 20`
	`        },`
	`        {`
	`            "datetime":`
	`"2022-09-02T00:00:00Z",`
	`            "sales" "5000.00",`
	`            "points": 75`
	`        },`
	`        ...,`
	`        {`
	`            "datetime":`
	`"2022-09-01T23:00:00Z",`
	`            "sales" "7000.00",`
	`            "points": 30`
	`        }`
	`    ]`
	`}`

You decided to create a web API (`` `GraphQL`or`gRPC` ``) and use any frameworks/technologies/libraries that might help you.

Your e-commerce system is starting to get more traction, which means you need to make sure that your system is able to handle many concurrent requests. You also want to show your friends that you are an awesome BE developer. It means that:

- You have to use PostgreSQL as the database
- You have to use Git
- Your code has to be clear
- You know good coding practices and patterns
- Your API can handle incorrect data
- Your architecture is extendable/testable
- Your system can be executed easily
- You can consider multi-thread, many servers
- Your system should be tested
- Your system should be able to scale with newer payment methods

Create a repository and publish an instruction for your fellow BE dev friends on how to launch your server.
Good luck!