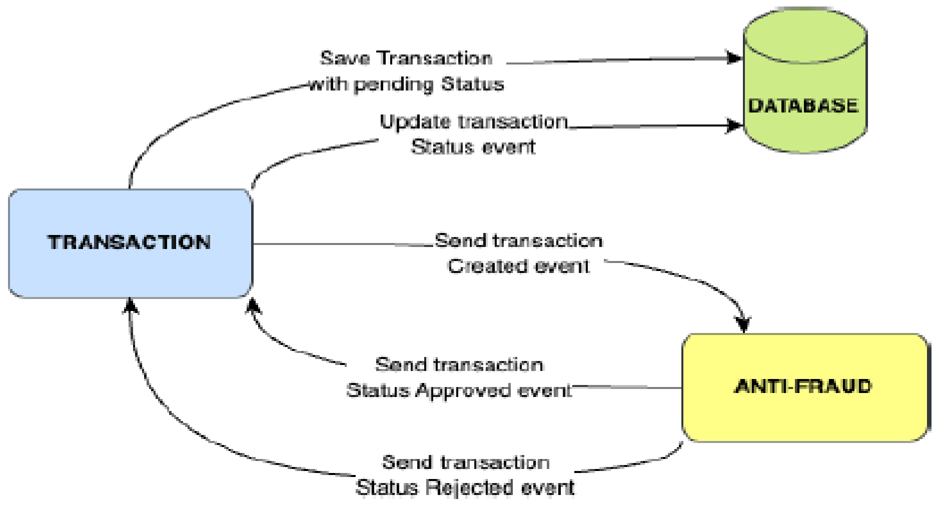
Anti-fraud

Every time a financial transaction is created it must be validated by our anti-fraud microservice and then the same service sends a message back to update the transaction status. For now, we have only three transaction statuses:

1. pending
2. approved
3. rejected

The following are the rejection criteria:

- Every transaction with a value greater than 2000.
- Accumulated per day is greater than 20000.


Save Transaction with pending Status
DATABASE
Update transaction Status event
TRANSACTION
Send transaction Created event
ANTI-FRAUD
Send transaction Status Approved event
Send transaction Status Rejected event

Tech Stack

1. .Net 8
2. Any database
3. Kafka

We do provide a `Dockerfile` to help you get started with a dev environment.

You must have two resources:

1. Resource to create a transaction that must contain:

```
{
  "sourceAccountId": "Guid",
  "targetAccountId": "Guid",
  "tranferTypeId": 1,
  "value": 120
}
```

2. Resource to retrieve a transaction

```
{
  "transactionExternalId": "Guid",
  "createdAt": "Date"
}
```

Send us your challenge

When you finish your challenge, reply to us with your solution. There are no limitations to the implementation, you can follow the programming paradigm, modularization, and style that you feel is the most appropriate solution.

If you have any questions, please let us know.

Sugerencias

- Recomendaciones técnicas:
    - Desarrolle la solución empleando NET 8.
    - Tenga presente utilizar arquitectura Hexagonal.
    - Incluir pruebas unitarias.
    - Utilice algún mecanismo de trabajo para comunicación entre los 2 microservicios “Transaction” y “Anti-Fraud”, estos mecanismos deberían garantizar la transaccionalidad o recuperación frente a fallos.
    - Para que todo se pueda probar de manera integrada se sugiere utilizar un docker-compose para que el ejercicio se pueda ejecutar y habilitar un entorno de desarrollo.
    - Se sugiere que en el README agregue algún diagrama explicando la arquitectura que está empleando.