E-Wallet Application

Requirements or Functionality

1. User Will be able create their E Wallet Account
2. Once the user is created some, money will be credited to their wallet account like 10,20,50 etc.
3. Email Notification will be sent to the user once the user got onboarded successfully
4. User Will be able to transfer some money to other user
5. User will be able to see their wallet balance
6. User will be able to see their recent transactions history

### Microservices

1. User Onboarding Service
2. Wallet Service
3. Notification service
4. Transaction service
5. API gateway service
6. Common module

#### User Onboarding service:

It is responsible to create or onboard user in wallet application

User

Id, name, email, phone, password, userIdentifier(Enum), userIdentifierValue, dob, role(enum), userstatus(enum), createdOn, updatedOn

Properties:

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.url=jdbc:mysql://localhost:3306/{db}

spring.datasource.driver-class=com.mysql.jdbc.Drivert

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

#### Notification Service:

Responsible for sending email notification to the user once the user has created their account

Properties

spring.mail.host=sandbox.smtp.mailtrap.io

spring.mail.port=587

spring.mail.username=e0fa16ca2711d2

spring.mail.password=01d02857310d79

spring.mail.properties.mail.smtp.auth=true

spring.mail.properties.mail.smtp.starttls.enable=true

@Bean

public ConcurrentKafkaListenerContainerFactory<String, String> kafkaListenerContainerFactory() {

ConcurrentKafkaListenerContainerFactory<String, String> factory = new ConcurrentKafkaListenerContainerFactory<>();

factory.setConsumerFactory(consumerFactory());

return factory;

}

JavaMailSender - It is an interface

SimpleMailMessage - class to construct mail object

#### Wallet Service

Responsible for creating user wallet account, and responsible for processing the wallet amount

Wallet Class

Id, userId, phoneNo,userIdentifier,userIdentifierValue, balance

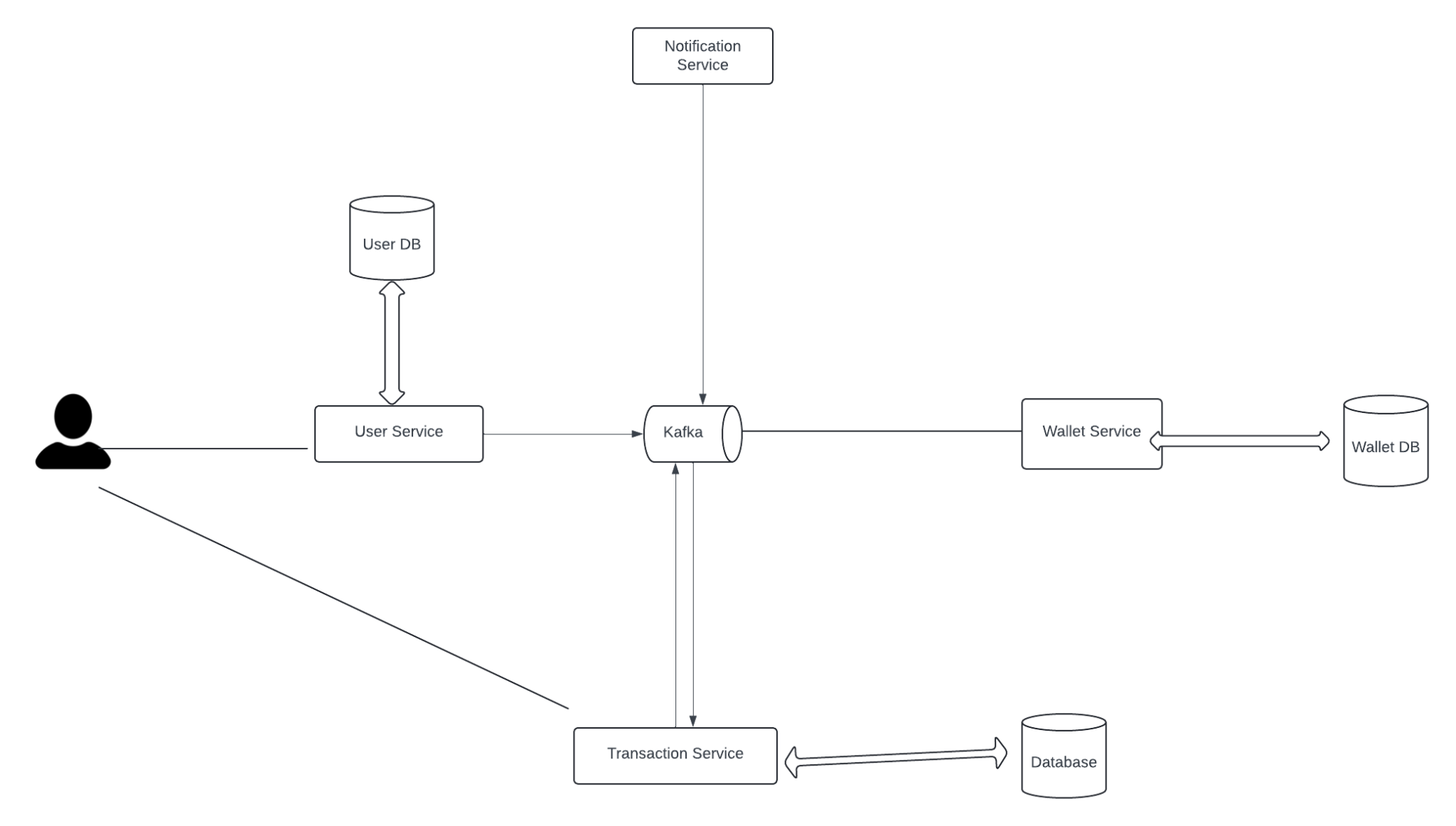
#### Transaction Service

It is responsible for processing the transaction and showing the transaction history of the user

Transaction class

Id, senderId, receiverId, txnId, amount, purpose, txnStatus (enum), txnMessage

**HLD**

****