Overview

The functionalities implemented including the followings:

- Software architecture
- One entity can have multiple accounts. One account can have multiple wallets of different assets. Each asset is represented as a wallet.
- The client of the ledger should be able to move assets from one wallet to another.
- The client of the ledger should be able to make multiple movements of assets in a single

request.

- All the requests to the ledger have to be executed as “all or nothing”.
- The ledger should be able to manage the lifecycle of an account. For example OPEN

state of an account means postings can happen to/from any wallets of the account.

CLOSED means postings cannot happen to/from any wallets of the account.

- The client of the ledger should be able to change the state of the account from one to

another.

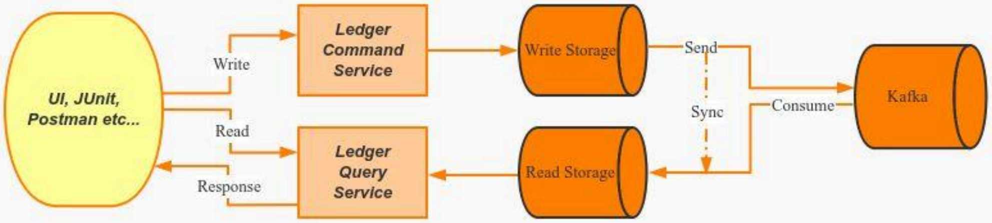
- Similar to the account lifecycle there could be multiple life cycles of postings(movement).

Such as PENDING, CLEARED, FAILED.

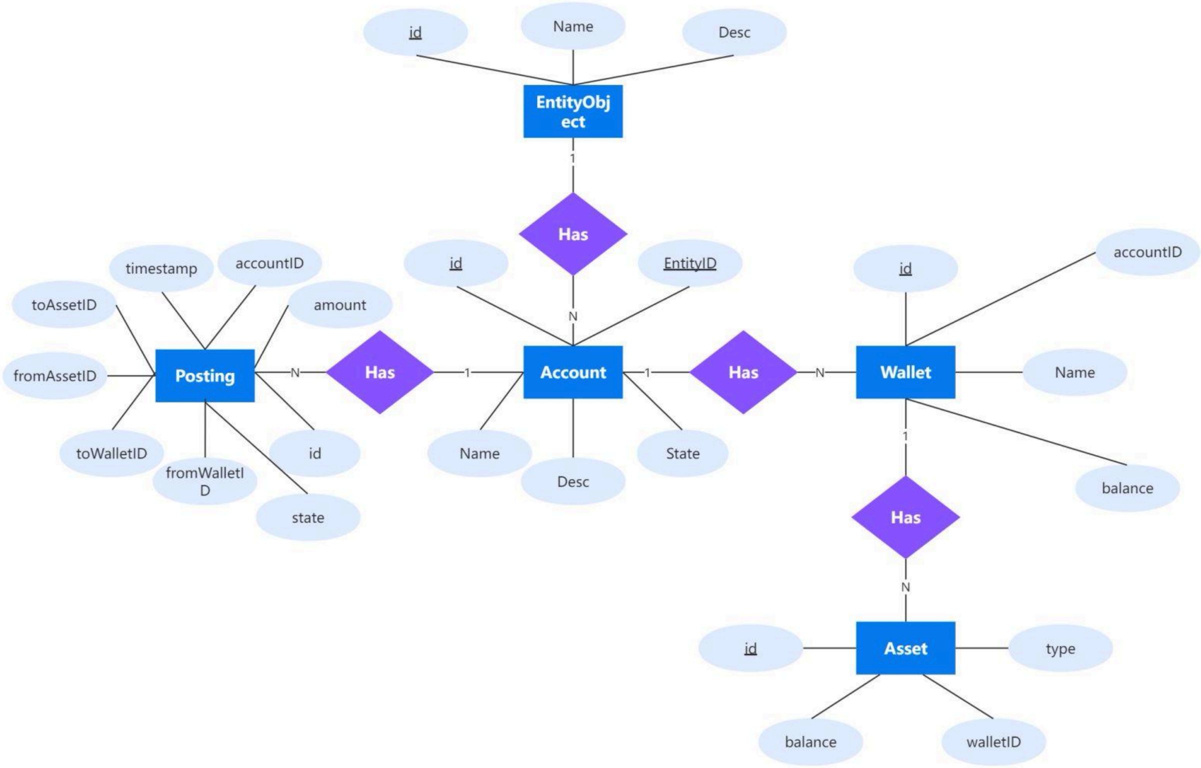
- The client of the ledger should be able to change to postings it has done before.

System Architecture

I used the CQRS pattern to segregate the read and write operations and split the microservice into two services. The first one is the Ledger-Command-Service, responsible for write operations, and the second one is the Ledger-Query-Service, responsible for read operations. Their databases are synchronized using Kafka.



Database Model



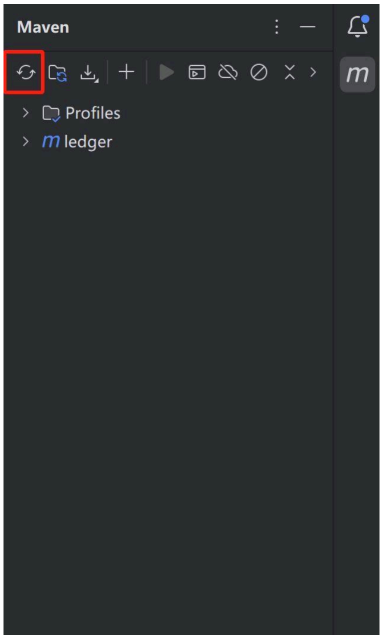
Some Assumptions

1. The wallet balance and asset balance is in dollar, and it is able to make transfer of balance between different types of assets.
2. Assume the balance can't be transferred between different accounts.

Setup Steps

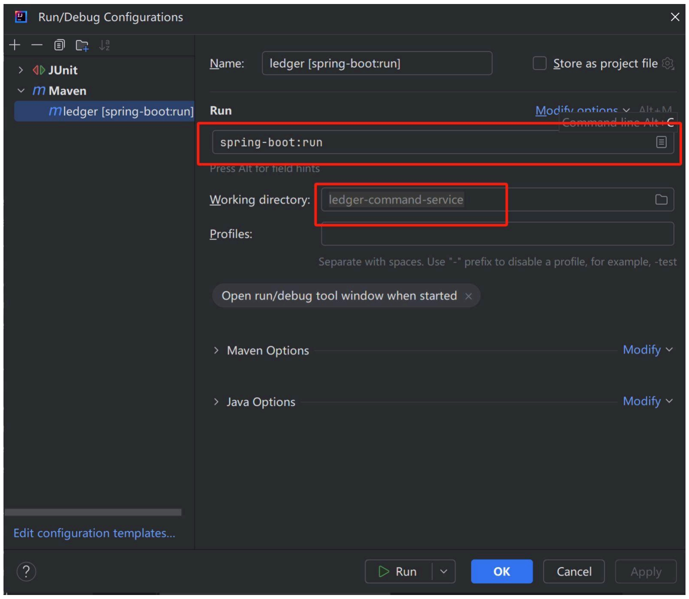
- Open and load maven projects

Open new projects on pom.xml of ledger-command-service and ledger-query-service -> maven tool box -> reload all maven projects.



- Add run configuration

Add maven run configuration for ledger-command-service and ledger-query-service as following:



Name: ledger [spring-boot:run]

Run

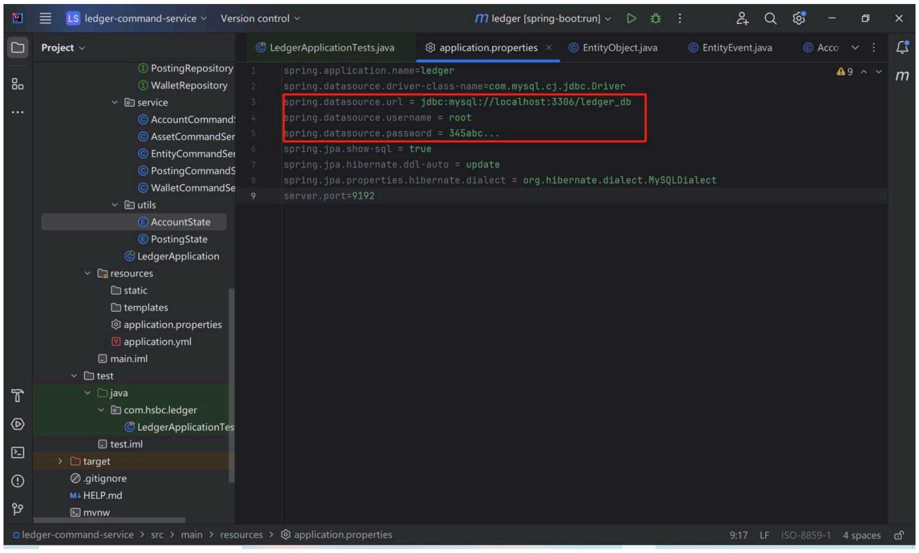
spring-boot:run

Working directory: ledger-command-service

Profiles:

- Change the MySQL configuration

Change the database port, user, password in application.properties for ledger-command-service and ledger-query-service according to your configuration.



```
spring.application.name=ledger
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url = jdbc:mysql://localhost:3306/ledger_db
spring.datasource.username = root
spring.datasource.password = 345abc...
spring.jpa.show-sql = true
spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect
server.port=9192
```

- Add mysql database

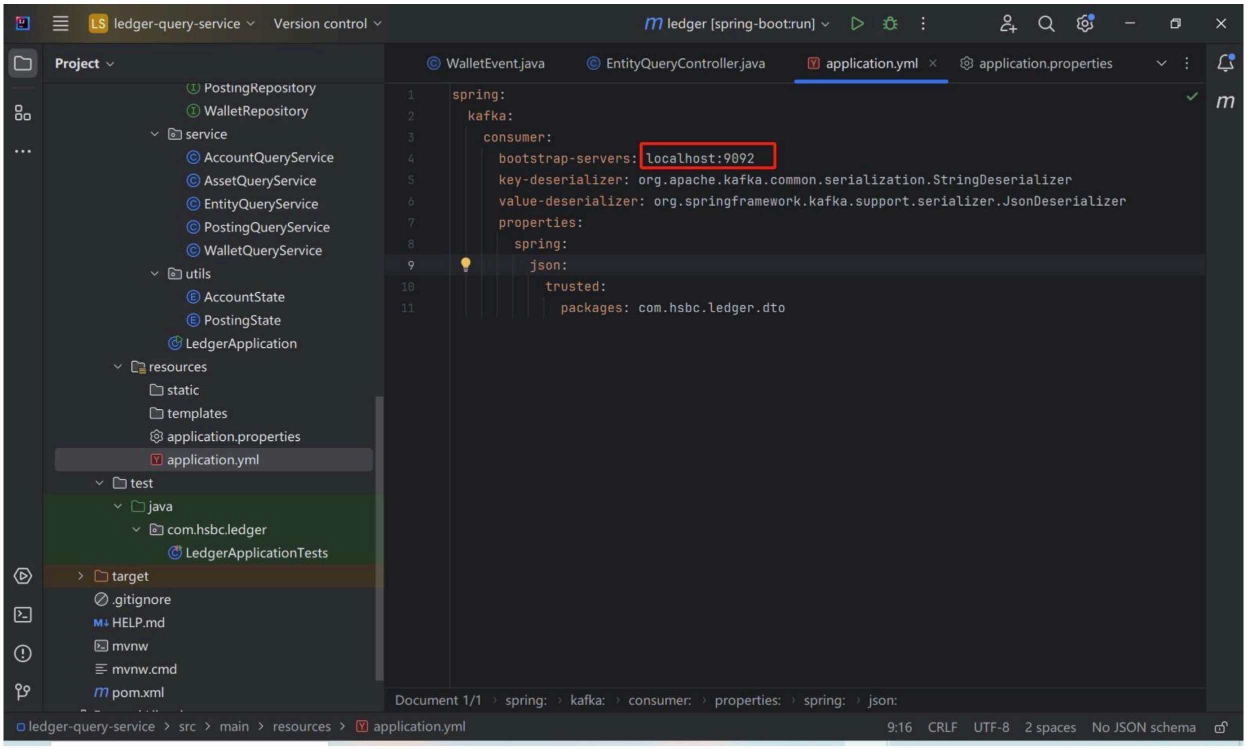
Use mysql workbench or other clients to add the database:

DROP DATABASE IF EXISTS ledger_db;

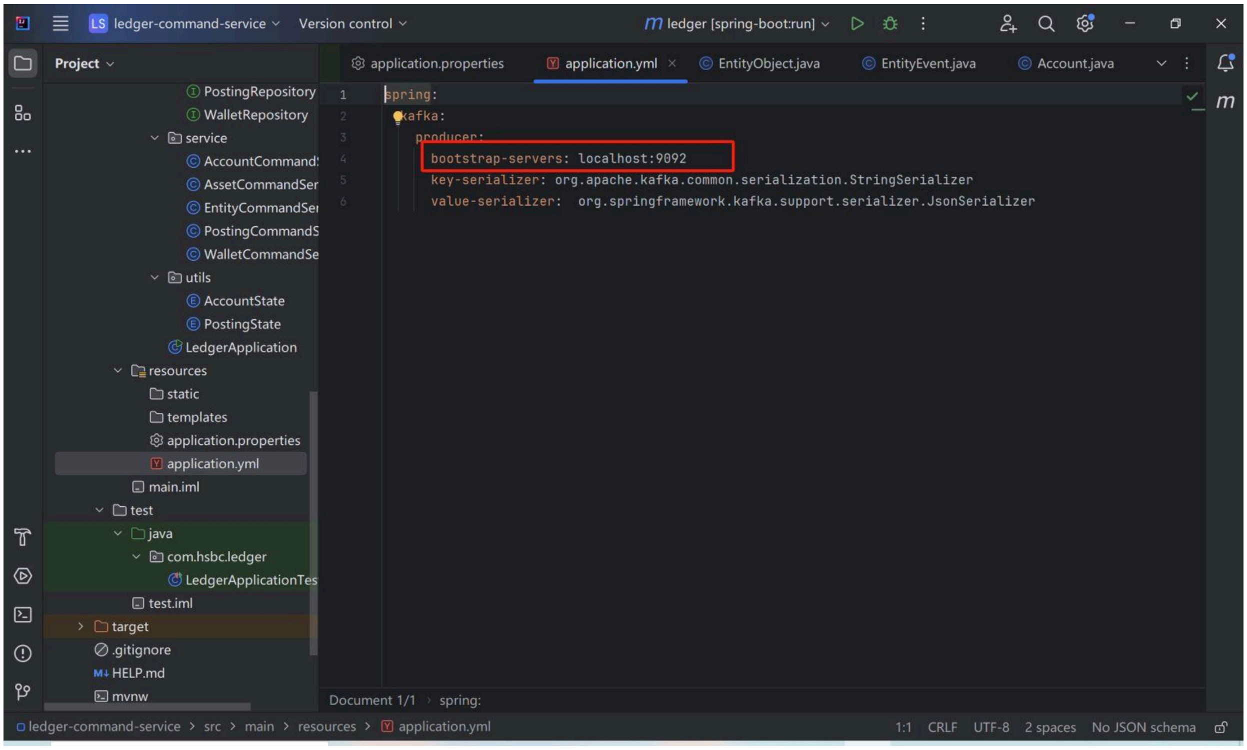
CREATE DATABASE IF NOT EXISTS ledger_db;

- Change Kafka configuration

Change the Kafka port in application.yml for ledger-command-service and ledger-query-service according to your configuration.



```yaml
spring:
  kafka:
    consumer:
      bootstrap-servers: localhost:9092
      key-deserializer: org.apache.kafka.common.serialization.StringDeserializer
      value-deserializer: org.springframework.kafka.support.serializer.JsonDeserializer
      properties:
        spring:
          json:
            trusted:
              packages: com.hsbc.ledger.dto
```



```yaml
spring:
  kafka:
    producer:
      bootstrap-servers: localhost:9092
      key-serializer: org.apache.kafka.common.serialization.StringSerializer
      value-serializer:  org.springframework.kafka.support.serializer.JsonSerializer
```

- Start the kafka server

```
cd <your-kafka-dir>\bin\windows\
```

```
zookeeper-server-start.bat ..\..\config\zookeeper.properties
```

```
kafka-server-start.bat ..\..\config\server.properties
```

- Create kafka topics

```
kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic Account-event-topic
```

```
kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic Wallet-event-topic
```

```
kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic Posting-event-topic
```

```
kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic Asset-event-topic
```

```
kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic Entity-event-topic
```

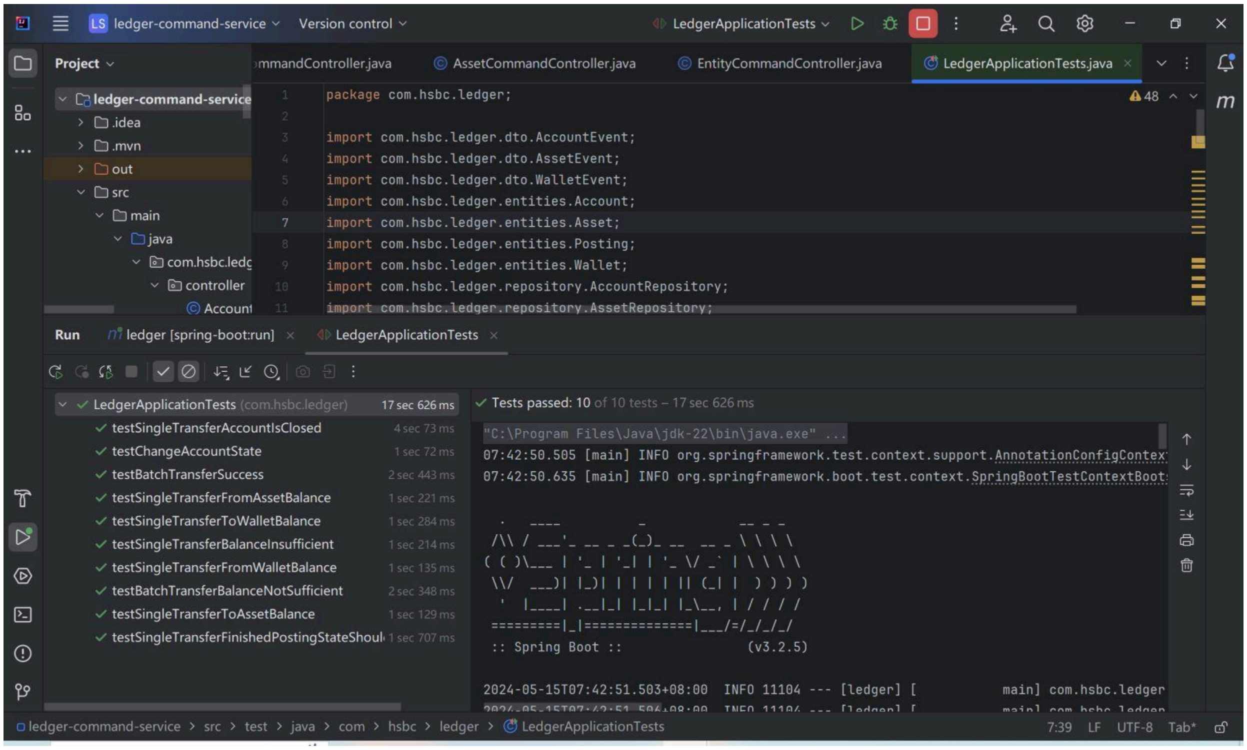
If you are using Linux change the commands accordingly.

- Start the ledger-command-service and ledger-query-service service

Now the ledger-command-service is able to run on port 9192, and the ledger-query-service is able to run on port 9191 by clicking Run 'ledger [spring-boot:run]'

- Run the JUnit tests

After starting the ledger-command-service and the ledger-query-service service, and make sure the Kafka and MySQL running properly in the background, run the JUnit tests in LedgerApplicationTests.java.



```java
package com.hsbc.ledger;

import com.hsbc.ledger.dto.AccountEvent;
import com.hsbc.ledger.dto.AssetEvent;
import com.hsbc.ledger.dto.WalletEvent;
import com.hsbc.ledger.entities.Account;
import com.hsbc.ledger.entities.Asset;
import com.hsbc.ledger.entities.Posting;
import com.hsbc.ledger.entities.Wallet;
import com.hsbc.ledger.repository.AccountRepository;
import com.hsbc.ledger.repository.AssetRepository;
```

```
LedgerApplicationTests (com.hsbc.ledger)            17 sec 626 ms
    testSingleTransferAccountIsClosed                 4 sec 73 ms
    testChangeAccountState                            1 sec 72 ms
    testBatchTransferSuccess                         2 sec 443 ms
    testSingleTransferFromAssetBalance               1 sec 221 ms
    testSingleTransferToWalletBalance                1 sec 284 ms
    testSingleTransferBalanceInsufficient            1 sec 214 ms
    testSingleTransferFromWalletBalance              1 sec 135 ms
    testBatchTransferBalanceNotSufficient            2 sec 348 ms
    testSingleTransferToAssetBalance                 1 sec 129 ms
    testSingleTransferFinishedPostingStateShoul      1 sec 707 ms
```

Tests passed: 10 of 10 tests – 17 sec 626 ms

```
"C:\Program Files\Java\jdk-22\bin\java.exe" ...
07:42:50.505 [main] INFO org.springframework.test.context.support.AnnotationConfigContex
07:42:50.635 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBoot

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.2.5)

2024-05-15T07:42:51.503+08:00  INFO 11104 --- [ledger] [           main] com.hsbc.ledger
2024-05-15T07:42:51.504+08:00  INFO 11104 --- [ledger] [           main] com.hsbc.ledger
```