Payment System

xxx

April 2023

the original idea is simple

- pay-in flow: from customer to service
- pay-out flow: from service to seller

1 Step 1

Understand the Problem and Propose High-level Design
design two flows and think of how to scale it
should be simple

2 Step 2

Propose High-level Design

- end user
- payment service
- record payment service db
- record ledger db
- record user wallet db
- payment executor
- link to external payment service provider (PSP)

API design

- POST payment
- get by specific payment id

database use relational DB to ensure ACID

3 Step 3

Design Deep Dive
PSP Integration
payment service links to PSP, during this interaction what should we show to users?

- user click checkout (payment not started)
- enter payment service
- communicate with PSP to create payment nonce
- store the returned token
- display payment page to user browser
- user starts payment
- redirect to completion page

reconciliation
every night PSP confirms with payment DB (ledger and wallet db)
to confirm all payments are correctly recorded
Message queue
from payment service to multiple others
e.g. ledger, wallet, payment service
Error handling
if transient error, use a queue to store it and retry