#ASLI ENGINEERING

Implementing Distributed Transactions

SWIPE

BY

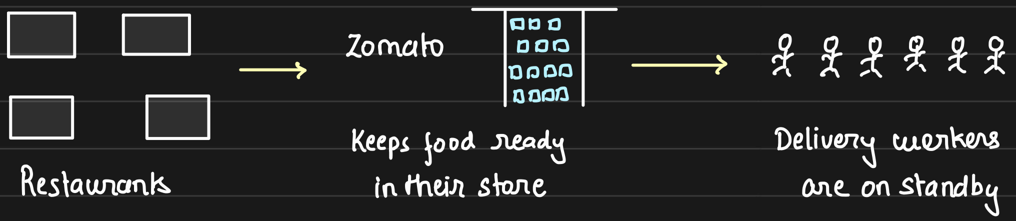
ARPIT BHAYANI

Implementing Distributed Transactions

Zomato's 10-min Food Delivery

Distributed Transaction: A transaction that span over multiple physical systems, machines or computers.

Scenario: 10 min food delivery by Zomato


Zomato
Restaurants
Keeps food ready in their store
Delivery workers are on standby

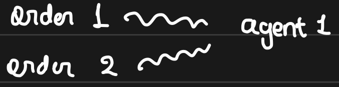
To guarantee that the food is delivered under 10 mins

Zomato should accept order only when

1. Food is available in the store
2. Delivery partner is available to deliver

One order
- One agent
- One food item

A user should not see "Order Placed" if it cannot be fulfilled



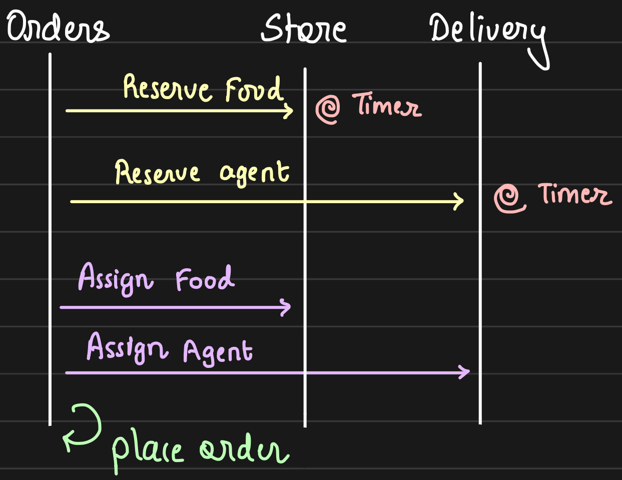
[Both were successful]

Two Phase Commit :

Split the entire flow in 2 phases

1. Prepare - Reserve
2. Commit - Assign

> Note: To keep things simple we are not implementing timer



Schema : Delivery Service

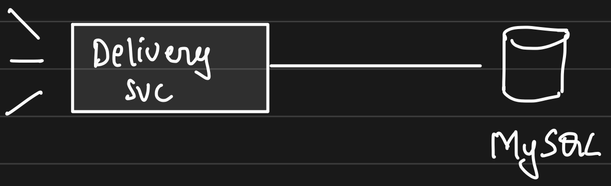
agents

id	is_reserved	order_id
1	False	NULL

↑ is reserved

↑ currently serving order

is_reserved = True
↓
agent is reserved for some order



Schema: Store Service

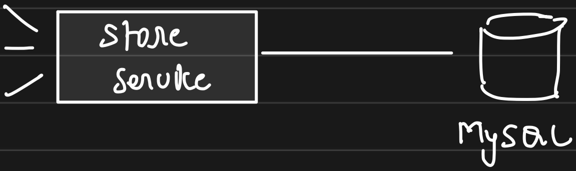
Food

id	name
1	Burger

Packets

Individual food items in the store

id	food_id	is_reserved	order_id
100	1	False	NULL



API End points : Delivery Service

/delivery/agent/reserve → reserve a delivery agent

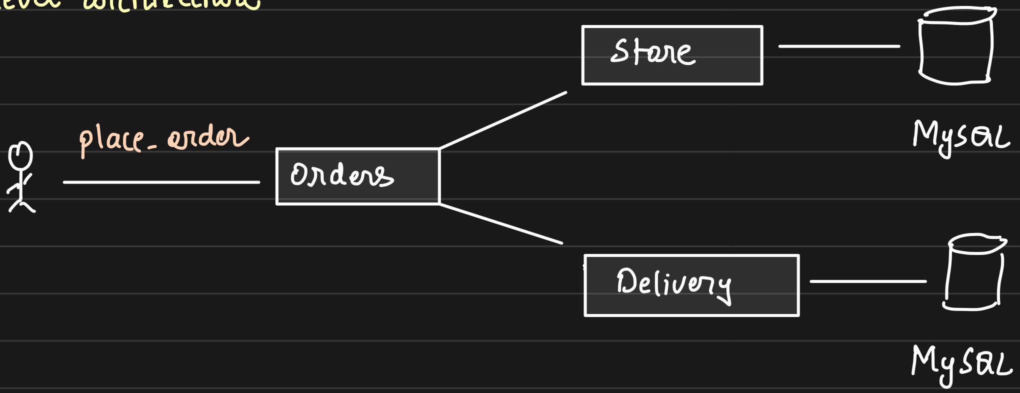
/delivery/agent/book → assign delivery agent to an order

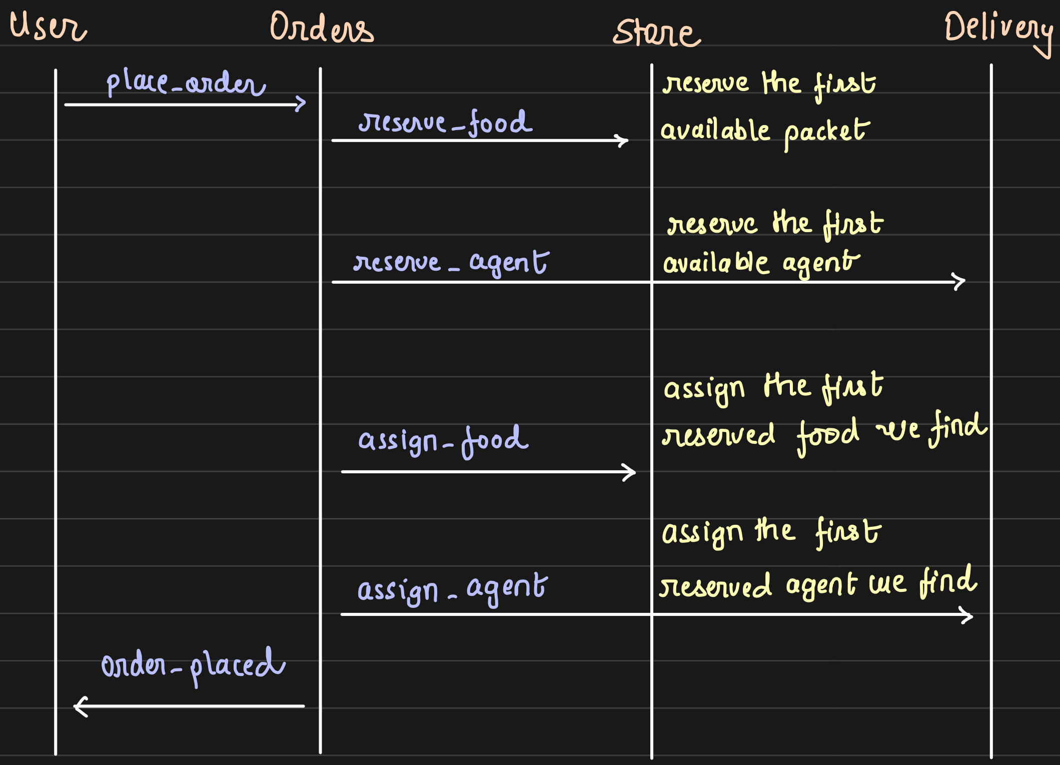
API Endpoints : Store Service

/store/food/reserve → reserve a packet

/store/food/book → assign a packet to an order

High level architecture




User
Orders
Store
Delivery
place_order
reserve_food
reserve the first available packet
reserve the first available agent
reserve_agent
assign the first reserved food we find
assign_food
assign the first reserved agent we find
assign_agent
order_placed

Ensuring no inconsistent data

To focus on data consistency

Note: Because we are not implement auto-freeup (timer), once a food packet ore an agent is reserved, they remain blocked forever.

But as we get expiry, the problem solves itself.

- Finding & reserving food happens atomically
- Finding & reserving an agent happens atomically
- Assing packet or an agent happens atomically

Four core updates are individually atomic.

Concurrent transactions has zero impact.

- To place the order, all 4 should succeed
- If any service fails, orders service will revoke the reservation
- By adding reservation stage → we ensure we have the necessary capacity