System Design GPay/ Payements app

Functional Requirements:

1) Create account, link to UPI gateway

2) Select contact from added users and initiate payement
3) Optional - Scan QR code
4) Payement can be buyer initiated or merchant initiated.
5) pay amount through transaction
6) Notification sent to sender/reciever about transaction status.

Non Functional: 1) Payement should be consistent over available / low latency/ ACID compliant
2) Notification can have reasonable delay
3) Account creation should be highly available over consistent.

APIs :
1) UPI_ID registerUser(email, phone, phone pin)
2) loginFirstLevel(phone_no, Pin)
3) SessionToken loginSecondLevel(UPI_ID, pass)
4) UPI_ID getUPI_ID(sessionToken, phoneNo, phonePIN)
4) payToUser(sessonToken, sender_UPI_id, reciever_UPI_id, amount)
5) saveTransaction(sessionToken, sender_id, reciever_id, amount,
time, transaction_status)
6) notifyMessage(sender_id, transaction_id, transact_status, amount,reciever)

Transaction History	
PK	**Transact ID**
	Sender UPI ID
	Reciever UPI ID
	Amount
	Date/Time

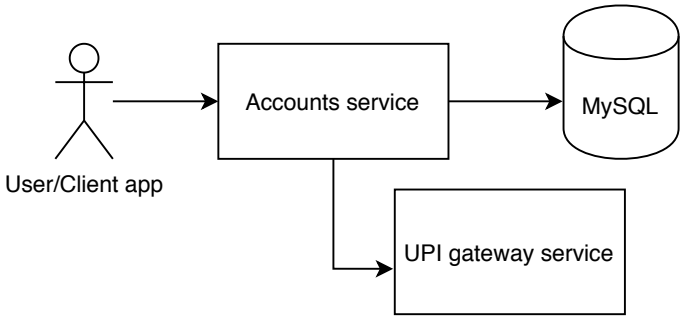
User Accounts	
PK	**Phone No**
FK	**UPI ID**
	Name
	Email

UPI Details	
PK	**UPI ID**
	Acnt No
	Name
	Encrypted Passwd

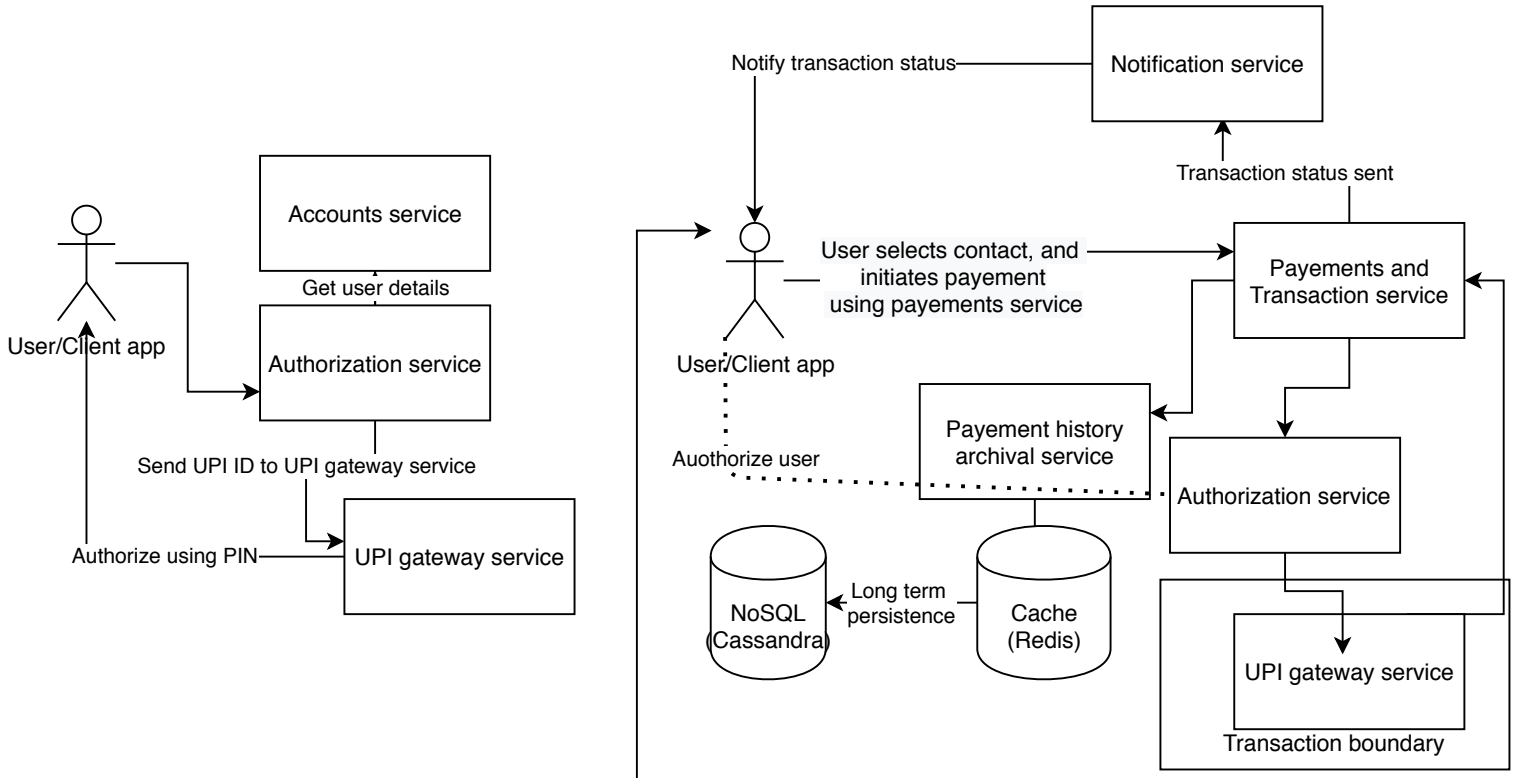
Merchant Accounts	
PK	**Merchant ID**
	Row 1
	Row 2
	Row 3

User Registration

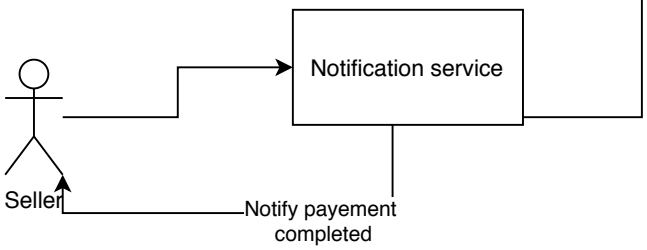
1) Use 2FA to register
2) User sends details to accounts service
3) The accounts service sends detail to notification service to generate OTP
4) User uses OTP for first level auth
5)User then provides bank account details to the accounts service
6) Accounts service sends details to UPI gateway.
7) UPI gateway registers user and provides UPI ID.
8) Accounts service saves user details and UPI ID to DB.



User initiated Payement



Seller initiated Payement



Key Points

1) Use load balancer to distribute load, prefer sticky session to maintain session info only on one server. Needs to be consistent over available.
2) Prefer 2FA for authorization to app, then 2FA to initiate transaction
3) Transaction history can be saved to cache temporarily, then persisted for long term to NoSQL. Thats because the operation is write once and frequent.