Disable OSIV from PG-SVC

Note : OSIV (Open Session in View) is a design pattern in Hibernate that keeps the database session open during the entire web request to allow lazy loading of data.

Problem: The scalability of the pg service was compromised due to the OSIV pattern. By keeping the database session open throughout the entire web request, database connections can be hogged, resulting in reduced performance and the inability to handle a high number of concurrent requests.

Initiative: Disable OSIV in the application to improve scalability and performance.

Execution:

- Test all application flows rigorously to identify issues caused by disabling OSIV.
- Fix the identified issues and write the missing test cases.
- Perform a load test of the service, as this is an application-level configuration.
- Load Test Document

Impact:

- Reduced usage time of database connections.
- Ability to serve much more traffic with the same maximum number of configured connections.
- Prevent incoming requests from experiencing increased latency or receiving 5xx errors due to database connections being hogged by ongoing requests, especially under increased load or when external parties increase latency.

Competency Meet: Technical Excellence

Native UPI Intent Support / Razorpay S2S

Problem: To reduce transaction costs, improve UPI payment experience by having inhouse upi intent support, and to have an alternative vendor for payments.

Initiative: Integrate with a new vendor, Razorpay, for UPI payments and to use S2S flow instead of SDK flow for inhouse improved experience.

Execution:

- Introduced a routing layer between vendors in the payment gateway (PG) based on the payment method.

- Redesigned the order creation flow (LLD) to separate the vendor layer from the business layer, making it extensible for easier integration with vendors with minimal code changes. Previously, business logic was tightly coupled with the vendor.
- Documents

Impact:
- Integration of Razorpay S2S resulted lakhs, totaling around 2 crores to date.
- Reduced development time required for future vendor integration.

Competency Meet : Execution & Technical Excellence

E-NACH Mandate

Problem: To enhance disbursement for the borrow product by introducing personal loans, with e-mandate registration as authentication and compulsory step.

Initiative: Introduce a new domain in pg-svc to support and streamline e-mandate registration and execution flows.

Execution:
- Designed the mandate module with both Low-Level Design (LLD) and High-Level Design (HLD) considerations, incorporating various payment methods for mandate registration and supporting multiple vendors.
- Developed the module with contextual retries in mind.
- Integrated the system with NPCI
- TRD Documents:
    - Document-1
    - Document-2

Impact:
- Personal loan project was launched successfully.
- Successfully registered around 1000 E-NACH mandates with NPCI till now.

Competency Meet : Execution & Technical Excellence, Domain knowledge

UPI Mandate Registration & Execution

Problem: Enhance user experience, target higher-risk user segments to increase borrow disbursement, and reduce collection operations.

Initiative: Implement a new method for mandate registration via UPI in the mandate module.

Execution:

- Designed and implemented the UPI-based mandate registration flow.
- Developed support for UPI mandate creation via Payment Links and In-app flows.
- Developed a system to handle both coupled and decoupled UPI mandate execution flows.
- Developed mandate execution flows with support for retries at various critical steps, such as Pre-Debit Notification and Debit Junctures.
- TRD Documents :
    - Document-1
    - Document-2
    - Document-3

Impact:

- Successfully registered around 13k mandates till now.
- Autoload mandate flow was migrated to newer flow.
- Enhanced customer experience as money is auto-debited and customers don't need to come to slice app.

Competency Meet : Execution Excellence, Domain knowledge

Note:

The initiation of mandate execution involves two steps

1. Sending Pre Debit Notification 24 hrs before the actual debit of money.
2. Initiating a debit transaction

- Coupled flow: vendor does both the above steps
- Decoupled flow: flow where we can call both the above steps individually.

Success Rate Improvement of UPI Mandate

Problem: Low success rate of mandate registration.

Initiative: Identify and address the root causes of the low success rate across systems.

Execution:

- Collaborated with cross-functional teams to understand the technical and experiential gaps causing the lower success rate of UPI mandates.
- Identified the root causes and implemented fixes to enhance the user experience.

Impact:

- Increased the UPI success rate from an average of **30% to 80%** across different flows. Thread - Link
- As the autoload flow was migrated to newer flow, the success rate of autoload increased as well.

Competency Meet : Execution Excellence, Domain Knowledge

Operation Reduction & Faster Resolution of Oncall Queries

Problem:

- Blocker while testing due to dependencies on third parties.
- Repeated queries and self queries leading to take lot of on call bandwidth

Initiative :

- Initiative for Faster testing and removing dependencies on external parties in non-prod environments
- Started Initiative to document runbooks for day-to-day on call queries facilitating easier, faster resolution.

Execution:

- Mocked the NPCI third party behavior with the help of central mocker service.
- Documented Runbooks

Impact :

- Achieved a **100% reduction** in queries during testing of PL flows.
- Lesser repeated queries and
- **Efficiency Gains:** Saved significant QA hours during feature testing and release regression. This ensured faster product shipment, especially when NPCI UAT experienced downtime.

Competency Meet: Execution Excellence, Innovation and Creativity

Race Condition Resolution

Problem: In the legacy code of pg-svc, multiple race conditions are causing payment status inconsistencies across systems, leading to monetary losses and increased operational effort to resolve these issues manually.
Initiative: Identify and fix the root cause of the race conditions.

Execution:

- **Root Cause Identification:** Debugged the code to determine the source of the race conditions.
- **Resolution:**

    - Addressed the issue of reading before taking a pessimistic lock, which was causing dirty reads and lost update problems.
    - Identified that taking the lock with a two-step operation was problematic, leading to two threads successfully acquiring the lock. This was corrected by using atomic operations instead.

Impact:

- Almost all flows are now free from race conditions.
- This has resulted in significant savings by reducing monetary losses and the operational effort required to manually resolve these issues.

Competency Meet : Technical Excellence & Customer obsession

Conducted the Learning Session for Clean Code.

Problem: Need to increase code quality standards and encourage team members to use best practices.

Initiative: Conduct clean code learning sessions and promote best practices.

Execution: Held weekly clean code sessions, fostering healthy discussions on new topics and best principles.

Impact:

- Improved code quality across the team. Team awareness of using the best practices has increased.
- Lesser bugs due to newer code.

Competency Meet : Innovation & Creativity and Collaboration

Mentoring

Initiative: Quicker Mentoring of new members in the team.
Execution:

- Helped the newer members in day to day tasks and operations.
- Encouraged to take decisions and gave the context regarding the svc.
- Documented each flow PG System for quicker mentoring and understanding of newer members in the team. Document

Impact:

- Successfully onboarded one SDE-2's and two SDE-1's to pg-svc

- leading them to become confident members of the team in day to day operations and decision making.

Competency Meet : Org Building and collaboration

Payout SVC

Note: Galleon svc was a microservice before the product launch, supporting the slice older products like disbursement of emergency cash to paytm banks and option to buy GIFT vouchers (EGV) like amazon, myntra voucher, etc .

Problem: Improve the efficiency and scalability of the Galleon SVC by segregating it into two new microservices: Sahukar SVC (orchestrator of borrow product) and Payout SVC (to disburse money).

Initiative:

- Segregate the Galleon SVC into Sahukar SVC and Payout SVC.
- Develop a lightweight, stateless Payout Service for efficient interaction with third-party vendors.
- Evolve the Payout SVC from stateless to stateful, ensuring it has its own interface for interacting with Sahukar SVC for borrow loan products.
- Evolve the service meeting new compliance requirements and supports multi-tenant and multi-vendor capabilities.

Execution:

- Designed and implemented the Low-Level Design (LLD) and High-Level Design (HLD) for the Payout SVC.
- Managed end-to-end service responsibilities, ensuring improvements in scalability and availability for 7-8 months single handedly.
- Developed features for the Payout SVC, including order management, asynchronous processing of orders, third-party interactions with Juspay and Razorpay, aggressive polling with exponential backoff, and webhook status updates to clients.
- Later, Proposed and developed Payout SVC V2 to meet compliance requirements, ensuring data segregation for multiple tenants and enabling easier onboarding of new tenants and vendors with minimal code changes.
- Onboarded and mentored SDE-2 and SDE-1 to the Payout SVC team.
- Integrated ICICI for streamlining money disbursal processes, boosting system performance, and achieving significant cost savings.
- Introduced delayed SQS queues in Payout, improving success rates and order completion times, and later adopted by Sahukar SVC.
- TRD Documents:

- https://slicepay.atlassian.net/wiki/spaces/ENG/pages/2976612718/Payout+Revamp
- https://slicepay.atlassian.net/wiki/spaces/ENG/pages/2954068449/Recon+for+Borrow+Transaction+System

Impact:

- Enabled the Payout SVC to disburse around 600-700 crores each month, becoming the backbone of the borrow product for money disbursement.
- Improved success rate of Payout SVC from 80% to around 98% by establishing reconciliation processes and identifying and addressing gaps in the system and vendor interactions.
- Enhanced system performance and compliance adherence, leading to significant cost savings and streamlined money disbursal processes.

Competency Meet : Execution & Technical Excellence

Circuit Breaker

Problem: Build a more resilient and fault-tolerant microservices architecture that can gracefully handle and recover from service failures or degradation.

Initiative: Implement a circuit breaker mechanism.

Execution :

- Implemented circuit breakers for internal clients.
- Contributed to an organization-wide initiative to enhance system resilience and fault tolerance.
- Implementation Details

Impact :

- Improved overall system stability and reliability by preventing cascading failures and ensuring smoother recovery from service disruptions.

Competency Meet: Technical Excellence

Routing based on MIDs

Problem: Achieve cost savings for the payment gateway by onboarding new Axis Bank MIDs for the unique combination of vendor and merchant ID.

Initiative: Develop a system to support routing between multiple MIDs for a unique combination of vendor and merchant ID.

Execution:

- Designed and developed a routing layer to support traffic routing between multiple MIDs for a given vendor and merchant IDS.
- TRD Document

Impact: Achieved cost savings of approximately **30-35k monthly.**

Competency Meet : Execution & Technical Excellence

Success after failure cases

Problem: Improve customer experience by addressing scenarios where customer payments initially fail and are later moved to success.

Initiative: Develop a solution to manage these cases effectively with the product
Execution: Demonstrated customer obsession by collaborating with the product team during non-working hours to enhance the user experience.
Impact:

- Enhanced customer experience for 1.5k payments per day during peak hours, totaling approximately 30k payments to date.
- Reduced the number of customer queries to the support team, minimizing the need for manual intervention.

Competency Meet : Execution Excellence & Customer obsession