Streamlined Routing Rule Framework In PG-SVC

Problem:
Eliminate hardcoded routing rules and streamlined to dynamically route payment traffic across vendors and MIDs based on dynamic business-defined parameters.

Initiative:

- Establish a single source of truth for all routing rules.
- Design and implement a flexible routing mechanism in PG-SVC that allows easy addition or removal of decision parameters with minimal code changes.

Execution:

- Identified hardcoded logic throughout the codebase and documented all existing routing flows involving vendors and merchant IDs.
- Collaborated with the product team to validate these flows before migrating them to the new, extendable routing system.
- Evaluated both in-house and go-rules solutions for managing routing logic based on various parameters.
- Chose and implemented the solution with a strong focus on performance and fault tolerance.
- *https://slicepay.atlassian.net/wiki/spaces/ENG/pages/3340240805/TRD+-+Redesign+Transaction+Routing+Config*

Impact:

- Significantly reduced cyclomatic complexity by removing hardcoded logic, making the system more maintainable and scalable.
- Increased agility in modifying percentage-based routing, with changes easily testable in lower environments.
- Reduced the time to introduce new decision parameters from 2–3 days to less than half a day i.e 80% reduction.

Competencies Demonstrated:
Execution Excellence | Technical Excellence

Payment Locking in PG-SVC

Problem:
Enable dynamic display of payment options based on real-time business and user-specific parameters.

Initiative:
Built an extensible solution to control visibility of payment options based on dynamic factors like business-level (eg. IsUpiMergerEnabled) and user-level (eg. txn amount).

Execution:

- Designed a flexible contract for payment locking configurable at the vertical level and dynamically overridable by payment initiators.
- Implemented a system to manage vertical-level defaults with the ability to override settings per transaction.
- https://slicepay.atlassian.net/wiki/spaces/ENG/pages/3371205089/TRD+-+Support+High+value+payments

Impact:

- Simplified the process of adding/removing decision parameters for controlling payment options.
- Reduced time to introduce new parameters by 60–70%.

Competencies:
Execution Excellence | Technical Excellence

Distributed Locking in CBS

[CBS Modernization Track]

Problem:
CBS modules were using instance-level locks to manage concurrent operations, which hindered horizontal scalability. As traffic and load increased, the lack of a distributed coordination mechanism led to contention issues and inconsistent behavior across instances.

Initiative:

To overcome these limitations, we aimed to introduce distributed locking across CBS modules, enabling them to scale horizontally without compromising data integrity or performance.

Execution:

- Implemented Redis-based distributed locking using Redisson to ensure a consistent locking mechanism across instances.
- Designed an extensible and reusable interface with annotation-based support to simplify adoption by other teams.
- Carefully considered edge cases such as lock contention, timeout handling, and recovery scenarios. Defined clear guidelines around lock wait time and expiration to promote consistency and avoid deadlocks.
- Documented implementation and best practices thoroughly, making it easier for other teams to onboard and follow the same standards.
- Conducted extensive concurrency and race condition testing to validate lock behavior under load and real-world conditions.
- Added relevant custom metrics and alerts to track the proper locking and unlocking of locks and to monitor the breaches.
- Shared implementation details and learnings internally (Confluence link).

Impact:

- Enabled seamless horizontal scaling by eliminating dependency on in-memory locks.
- Provided a standardized distributed locking framework that other CBS modules could easily adopt.
- Improved system reliability and concurrency handling across critical CBS workflows.

Competencies:

Execution Excellence | Technical Excellence

HA Enablement in CBS

[CBS Modernization Track]

Overview:
Part of the broader CBS modernization effort to make core banking services highly available, fault-tolerant, and horizontally scalable by enabling them to run seamlessly across multiple pods with zero downtime.

Key Contributions & Impact:

1. **NGINX Configuration for HA Routing:**
   Fine-tuned NGINX setup to support intelligent traffic routing across multiple HA pods, enabling zero-downtime deployments and smooth request distribution.
2. **Graceful Shutdown Enablement:**
   Implemented graceful shutdown handling for CBS Transactions, ensuring no transaction loss or inconsistency during deployments. This reduced operational overhead and eliminated downtime-related issues.
3. **High-Load Validation via Performance Testing:**
   Conducted comprehensive performance and concurrency testing on critical APIs to validate behavior under load. Documented results and learnings for reuse in future migrations and onboarding.
4. **HA-Safe Migration of Critical APIs:**
   Successfully migrated 19+ high-traffic GET APIs (each with >1 RPS) to HA pods post validation, ensuring service reliability and performance parity.
5. **State Management Refactoring:**
   Identified all instance-level atomic variables and transitioned them to Redis-backed shared state, enabling correctness and consistency in a distributed, multi-pod environment.
6. **HA Readiness Assessment:**
   Performed in-depth analysis and safety checks on 25–30 APIs to ensure they were HA-compatible within the CBS Transactions module.
7. **Documentation:**
   Created comprehensive internal documentation to outline the HA testing scope and safe migration practices, enabling smooth future rollouts.
   CBS Transactions HA Test Scope

Competencies Demonstrated:
Ownership | *Execution Excellence* | *Technical Excellence* | *Collaboration*.

Decoupling of CBS Modules

[CBS Modernization Track]

- **API-Driven Dependency Removal:**
  Contributed to the design and development of APIs to eliminate direct dependencies between CBS modules. Enabled clean service-to-service communication and improved system boundaries for better maintainability and control.
- **Stored Procedure & DB Decoupling:**
  Contributed to remove cross-module database access by decoupling stored procedures. Took ownership—created necessary APIs in the dependent module and integrated them—to accelerate the transition and ensure clean separation of concerns.

Competencies Demonstrated:
Ownership | *Execution Excellence* | *Collaboration*.

On-call Efforts & Operational Support

Extended On-call Ownership:
Stepped up during a bandwidth crunch and ensured high availability for on-call support over an extended period (more than a month), significantly reducing operational load on the team and ensuring business continuity.

Key Contributions During On-call:

1. **Optimized High-Load GET API:**
   Identified and optimized a GET API responsible for 90% CPU spike on the CBS RDS reader, preventing potential outages and improving system stability.
   Slack Reference

2. **Resolved M2P Reconciliation Issue:**
   Provided a temporary solution to unblock day to day operations, followed by driving a permanent fix to eliminate the recurring recon issue at the root level.
3. **Fixed Compliance Gap in Card Masking:**
   Identified gaps in masking customer debit card numbers in ATM withdrawal flows, collaborated across product and tech to implement a compliant, long-term fix across affected touchpoints.
4. **Plugged Fraud Risk in NEFT/RTGS Flows:**
   Contributed towards addition of FRM (anti-fraud) checks in NEFT/RTGS flows that were being exploited—mitigated fraud risk on priority in coordination with stakeholders.
5. **Identification and Solutioning of IMPS failures in merchant flow:** Identified and addressed an issue where the FRM (Fraud Risk Management) check needed to be skipped for IMPS in specific merchant use cases.
   - **Execution:**
     - Investigated and identified the root cause of the issue.
     - Collaborated with sister teams and the Product team to validate the problem.
     - Conducted meetings to align on the right solution.
     - Raised the task and ensured it was prioritized and taken up in the sprint.

Competencies Demonstrated:
Ownership | *Execution Excellence* | *Operational Execellence*

Alerting in CBS

Problem: Service degradations were going undetected by our team and were frequently flagged by other teams, highlighting a gap in our observability and responsiveness.

Initiative:
Took ownership to transform our alerting capabilities, with a clear goal: ensure our team is always the *first* to detect, respond to, and act on any downtime or degraded experience—before it impacts users or reaches others.

Execution:

- Integrated comprehensive API-level metrics to monitor response codes and latency trends in real time.
- Smartly segmented alerts based on API traffic volume and time of day (day vs. night), drastically reducing false positives and alert fatigue.
- Introduced business-level alerts, bridging the gap between technical health and real-world impact, ensuring no critical event goes unnoticed.
- https://slice-it.slack.com/archives/C085A8QQYCW/p1742536241498539

Impact:

- Our team has become the *go-to* for system reliability—we are now consistently the first to identify and respond to issues.
- We've moved from reactive firefighting to proactive leadership, flagging critical issues to other teams even before they realize there's a problem.
- Strengthened trust across stakeholders through sharper, faster, and more accurate alerting.

Competency Demonstrated:
Execution Excellence | Technical Excellence | Operational Excellence | Customer Obsession

UPI Flow Optimization

Problem:
The existing UPI flow included unnecessary processes, leading to performance inefficiencies. The goal was to enhance the overall performance by eliminating redundant flows.

Initiative:
Conduct a comprehensive analysis of all API calls made during the UPI flow and take appropriate action to optimize them.

Execution:

- Conducted a thorough analysis of the UPI flow to list all external API calls, capturing their latency, frequency, and identifying potential action items.

- Collaborated with the Product team to review certain flows that appeared unnecessary from a business perspective and removed them accordingly.
- Reviewed the codebase to identify checks irrelevant to the UPI flow that were still being triggered. These were corrected to streamline the logic.
- https://slice-it.slack.com/archives/C085A8QQYCW/p1739988603756479
- UPI internal API calls Analysis.

Impact:

- Eliminated 4 redundant API calls, resulting in a more efficient UPI flow.
- Helped simplify the UPI flow and reduced API spikes caused by unnecessary dependencies.

Competencies Demonstrated:
Customer Obsession | Collaboration | Operational Excellence

ACH Idempotency Check Implementation

Problem:
Recurring clearing mismatches were being triggered due to improper handling of multiple user clicks, leading to duplicate processing. This was a persistent pain point, consuming 30–40 minutes of daily on-call bandwidth and creating operational overhead.

Execution:

- Deep-dived into the ACH flow to understand the root cause of inconsistencies.
- Identified application-level gaps and evaluated potential solutions through internal discussions and analysis.
- Implemented a robust **idempotency check** at the application layer to prevent duplicate processing from multiple user actions.

Impact:

- Eliminated recurring clearing mismatches caused by multiple clicks.
- Saved 30–40 minutes of daily on-call bandwidth.
- Significantly reduced manual operational interventions, improving system reliability and team efficiency.

Competencies Demonstrated:
Technical Excellence | Operational Excellence

On-call Runbook & Documentation

Problem:
Frequent repetition of common tasks and issues during on-call shifts led to inefficiencies and delays in resolution.

Initiative:

- Created comprehensive documentation and a runbook covering recurring issues and standard solutions to streamline on-call support.
- https://slicepay.atlassian.net/wiki/spaces/CBS/pages/3415771125/Oncall+Runbook

Impact:
Empowered team members to independently resolve common issues more efficiently, reducing response time and improving overall team productivity.

Competencies Demonstrated:
Org Building | Collaboration