Self-Healing Microservices with Recovery Framework

Introduction

- Microservices architecture is gaining popularity, but failures can occur.
- This presentation introduces a self-healing library to enhance microservice resilience.

Challenges of Microservices

- Independent deployment: Failures can isolate services and impact overall functionality.
- Manual recovery: Requires developer intervention, leading to downtime and delays.

Introducing the Self-Healing Library

- Core components:
  - @Before & @AfterReturning aspects: Capture data for recovery points.
  - Recovery service class: Manages recovery logic and triggers.
  - RECOVERY POINT TABLE: Stores data for potential recovery attempts.

How it Works

1. **Capture Recovery Points (Aspects):**
   - @Before: Intercepts controller methods before execution.
   - Stores relevant data (parameters, state) for potential recovery.
2. **Recovery Logic (Service Class):**
   - **Controller API:** Triggers manual recovery or marks entries for recovery.
   - **Spring Scheduler:**
     - Periodically checks the RECOVERY POINT TABLE.
     - Identifies recovery-worthy entries based on criteria (time, attempts).
     - Invokes appropriate recovery logic using stored data.
3. **RECOVERY POINT TABLE:**
   - Stores:
     - Unique identifier
     - Creation timestamp
     - Data for recovery
     - Metadata (service method name)
     - Optional - Recovery attempt count

Benefits

- Improved system resiliency: Automatic recovery attempts minimize downtime.
- Reduced manual intervention: Faster recovery without developer involvement.
- Increased fault tolerance: Microservices adapt to failures and maintain functionality.

Considerations

- **Idempotency:** Ensure recovery logic avoids unintended side effects on retries.
- **Error Handling:** Implement robust error handling during recovery attempts.

- **Recovery Strategies:** Define strategies based on failure types (retry, rollback, notify).
- **Configuration:** Allow customization of scheduling, retries, and recovery logic.
- **Logging and Monitoring:** Track recovery attempts and monitor performance.

Conclusion

- This self-healing library empowers microservices to self-heal from failures.
- Increased resilience leads to a more reliable and robust microservices architecture.

Additional Slides (Optional)

- Code snippets demonstrating aspect usage and recovery logic.
- Example recovery scenarios for different failure types.
- Configuration options for the self-healing library.

Note:

- Feel free to customize the content and level of detail based on your audience and presentation time.
- Consider adding visuals (diagrams, screenshots) to enhance clarity.
- Practice your delivery and be prepared to answer any questions from your audience.