Implementation Plan: Modular Rate Limiter

Module Structure

- rl-core: Core algorithms, SPI definitions, and internal registry.
- rl-spi-redis: Redis implementation using Lua.
- rl-spi-caffeine: Local memory implementation.
- rl-adapter-spring: Spring Boot Starter, SpEL resolver, and AOP.
- rl-adapter-quarkus: Quarkus Extension and CDI Interceptors.
- rl-adapter-k8s: ConfigProvider for ConfigMap hot-reloading.

Phase 1: Core & Algorithms

- Define RateLimitStorage and ConfigProvider SPIs.
- Implement TokenBucket and SlidingWindowCounter as pure Java functions.
- Unit test algorithms with simulated "virtual time."

Phase 2: Storage & Resilience

- Write the Redis Lua script for atomic bucket updates.
- Implement the CircuitBreaker logic to handle StorageProvider failures.
- Implement the "L1/L2" dual-layer fetcher.

Phase 3: Framework Adapters

- Build the Spring @Aspect and the Quarkus Interceptor.
- Create the KeyResolver that supports SpEL/Jakarta EL.
- Add Auto-Configuration logic for each framework.

Phase 4: K8s & Observability

- Implement the FileWatcher for K8s volume-mounted ConfigMaps.
- Integrate with Micrometer and SmallRye Metrics.