ta med videre

- prøve litt parprogrammering
- hva ønsker vi få ut av PR (og hva alternativer som finnes)
- team arbeid > solo arbeid?
- se litt på bruk av JPA
- se litt på bruk av docker images for test (evt. testcontainers)
- kotlin og spring boot 3 er ganske ryddig
- spring boot 3 er ganske ryddig
- se på problem details (sb3)
- se på assert API for validering av requests (sb3)
- kanskje les Clean Architecture (RCM)
- spark arkitekten
- er Snyk plugin lov og burde vi bruke den?

day 2

-Events

- keys: **UUIDs**, integers, nanoids
- enkelt i starten (e.g. lagre til postgres)
- Exactly-once er umulig/vanskelig, heller "at least once"
- Bruk idempotency keys (+ at least once)
- Husk: er distribuert, uforvnetet ting skjer (tapte transaksjoner, ugyldig state, etc.)

Sikkerhetskultur i Origo

- sikkerhetstesting (demystifisering)
- security champion

Java security ("Don't get burned")

- SQL injection
- XML parsing (skru av doctype, XXE)
- Hashing (argon2id)
- Encrypting (aes, secure mode, >128 bits)
- YAML parsing (bruk ⩾ 2.x)
- snyk plugin?

-Quarkus for spring

Arkitektur kaos

- Keep it simple
- Forståelig, utvidbart, fungerer og tør endre >>> andre krav

- Package by feature og isoler (en public interface, resten package-private)

JVM bottleneck pga. type-caching i L2

- JDK 22?
- Agent for discovery

Apache Flink (real-time data)

Apache Kafka and Apache Pulsar

Kafka @ politiet

Microtjenester

- enklere å refaktorere bort noe
- ADR
- Rådprosessen

day 1

Data-driven stateless UI

- data ⇒ uiData ⇒ render (e.g. vdom)
- frakoble data fra UI - bare en mengde data, kan rendres på forskjellige måter
- tester subset av data ⇒ render (pure functions)

PR eller CI

- alternativer til PR
  - code reviews (etterkant)
  - parprogrammering
  - synkron gjennomgang ("hei kan vi se på dette sammen" / direkte PR)
- PR har verdi, men bryter flyt til CI - og CI er viktigere (?)

Parprogrammering for raskere flyt

- kutter steg pga. ikke nødvendig lengre (PR, test-miljø, ...)
- feature-toggling

Java 17++

- Loom - virtual threads
- Valhalla - custom primitive typer, type extensions
- Amber?

Microfrontends are the shit
Hidden security features of the JVM

-JVM startup
-Cloud Native design & principles

-Payment settlement ("how we settle millions of payments at norwegian air")
IPv6

- Stateless (= routing + MAC)
- Stateless DHCPv6

Dynamic DNS

- SRV records

Java collections

- don't use synchronized wrappers

Spring-Boot 3++

- Native images - built into build setup (via initializer at least)
- Client mappings built in (for edge-services, e.g. kombiner med graphql mappinger)
- Assertions on the inside of controllers (non-annotations), throws Exceptions
- Crud repositories (same as before)
- ProblemDetail(s) - common error format
- Observation integration (new in spring-boot 3) for metrics/tracing
- spring-gateway (same as before)

Maven quirks

- consumer > parent > *
- explicit direct dependency > dependency management > transitive dependency
- last in tree wins over earlier in tree
- use maven enforcer
- update maven (3.8.8 or 3.9.3)

Arkitekt 1.0 ⇒ 2.0

- 1.0 removed by independent and self-supplied teams
- 2.0 ⇒ a generalist that should be able to communicate with all corners, e.g. law, economics, software developers and infracstructure engineers

-Frontend frameworks

- Next.js
- Remix

Ergosplit

- ZSA moonlander
- shop.beekeeb.com (?)
- Colemak (adjusted)