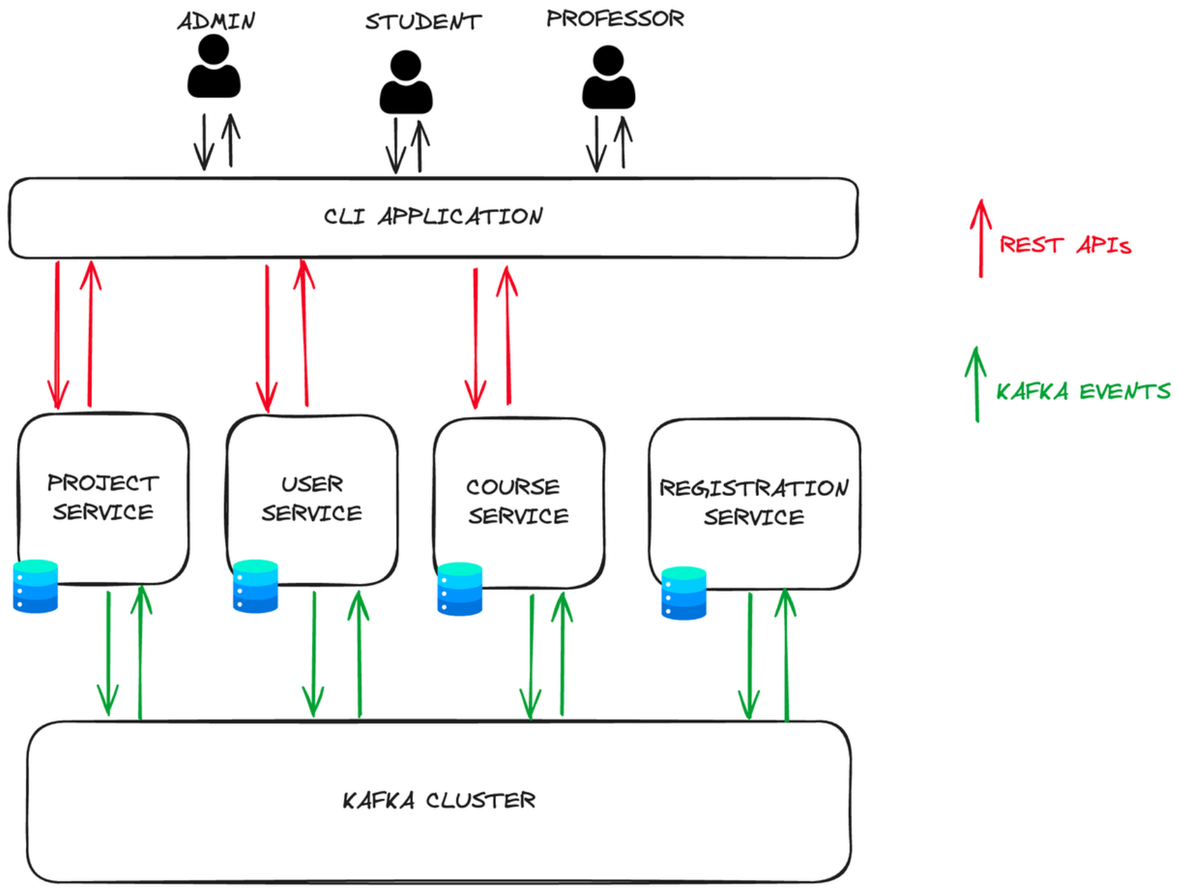
# Online Services for Continuous Evaluation


# Assumptions and Guidelines

- Services do not share state, only events over Kafka topics
- Services may crash at any time => need fault recovery procedure
- State is implemented using in-memory data structures
- Kafka topics cannot be lost
- No authentication system
- Sum of grades is sufficient when > 50

# High Level Architecuture Diagram



# CLI Application and Commands

| User Role | Commands |
|---|---|
| Admin | `create-course --id=<id> --name=<name>`<br>`get-courses`<br>`delete-course --id=<id>`<br>`create-student --id=<id>`<br>`create-professor --id=<id>` |
| Student | `get-courses`<br>`enroll --course-id=<id>`<br>`submit-solution --course-id=<id> --project-id=<id> --submission-id=<id> --solution=<solution>`<br>`get-course-projects --course-id=<id>`<br>`get-project-submissions --course-id=<id> --project-id=<id>`<br>`get-submission-grades --course-id=<id> --project-id=<id> --submission-id=<id>` |
| Professor | `get-courses`<br>`create-project --id=<id> --course-id=<course-id> --name=<project-name>`<br>`get-sub --course-id=<id> --project-id=<id>`<br>`grade --course-id=<id> --proj-id=<id> --sub-id=<id> --grade-id=<grade-id> --grade=<grade>` |

Table 1: CLI Commands for Different User Roles

# Microservices Overview

| Microservice | Endpoint | Method | Description | End-User |
|---|---|---|---|---|
| **Project Service** | /courses/:course-id/projects/create | POST | Creates a new project for the given course | Professor |
| | /courses/:course-id/projects/:project-id/submit | POST | Submit solution for project of a course | Student |
| | /courses/:course-id/projects/:project-id/submissions/:submission-id/grade | POST | Grade a solution submission of a student | Professor |
| | /courses/:course-id/projects | GET | Retrieve projects of a course | Student |
| | /courses/:course-id/projects/:project-id/submissions | GET | Retrieve solution submission of a project | Professor |
| | /courses/:course-id/projects/:project-id/submissions/:submission-id/grades | GET | Retrieve grade of a solution submitted by a student | Student |
| **User Service** | /users/student/create | POST | Creates a new student | Admin |
| | /users/professor/create | POST | Creates a new professor | Admin |
| **Course Service** | /courses | GET | Retrieves all courses | Admin, Professor, Student |
| | /courses/create | POST | Creates a new course | Admin |
| | /courses/:course-id/delete | DELETE | Deletes a specific course | Admin |
| | /courses/:course-id/enroll | POST | Enrolls student in a specific course | Student |
| **Registration Service** | Not applicable | N/A | Does not expose REST endpoints | N/A |

Table 2: REST endpoints exposed by microservices

- Each service, except for the Registration, communicates with the end-user by exposing REST endpoints
- Each service keeps in-memory state; no database!

# Kafka Cluster Configuration

**Topic**: Student

**Producer**(s): User Service
**Consumer**(s): Course Service

**Topic**: Professor

**Producer**(s): User Service
**Consumer**(s): Project Service

**Topic**: Course

**Producer**(s): Course Service
**Consumer**(s): Project, Reg. Service

**Topic**: Enrollment

**Producer**(s): Course Service
**Consumer**(s): Project Service

**Topic**: Project

**Producer**(s): Project Service
**Consumer**(s): Reg. Service

*Every microservice that produces on a topic also consumes on the same topic.*

***Reason:*** *Recover its state during a crash.*

**Topic**: Submission

**Producer**(s): Project Service
**Consumer**(s): Reg. Service

**Topic**: Grade

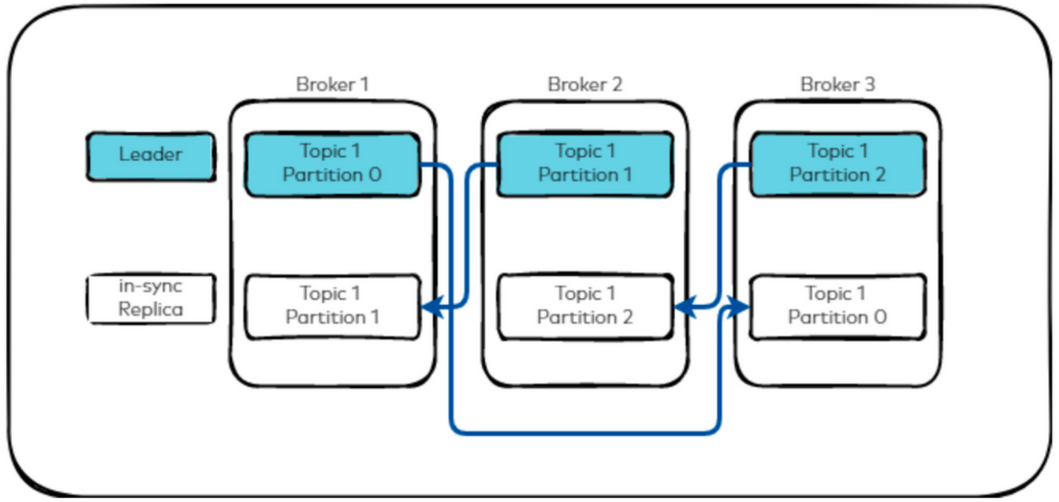
**Producer**(s): Project Service
**Consumer**(s): Reg. Service

# Design Choices

- Easy to build a server
- Goroutines
- Concurrency capabilities

kafka

- 7-day-old message are deleted
- Replication factor of 3



# Implementation Details 1 of 2

confluent-kafka-go

- Good integration with Confluent Cloud

+

kafka-wrapper-go

- Shared code between microservices

---

## Producer

- Headers: action type (add, delete, etc...)
- Kafka decides the optimal partition based on the current partition load

# Implementation Details 2 of 2

## Consumer

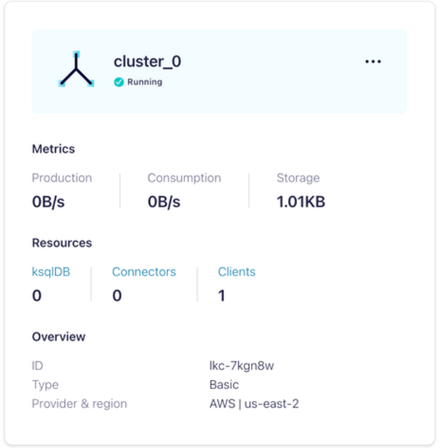
- Consumer group set to “evaluation-sys-<microservice-name>”
- Each microservice has their consumer group => allows different microservices to consume the events on the same topic in parallel
- “auto.offset.reset” configuration set to earliest => read from earliest uncommitted message
- Seek to beginning of each assigned partition => **Fault Recovery**

# Deployment

aws for the microservices

| Environment name | Health | Application name | Platform | Domain | Running versions | Tier na... |
|---|---|---|---|---|---|---|
| Eval-sys-course-env-docker | Ok | eval-sys-course | Docker running on 64bit Amazon Linux 2023 | Eval-sys-course-env-docker.eba-ij33i5hc.eu-north-1.elasticbeanstalk.com | eval-sys-course-version-2 | WebServer |
| Eval-sys-project-env-docker | Ok | eval-sys-project | Docker running on 64bit Amazon Linux 2023 | Eval-sys-project-env-docker.eba-tx9pz6g2.eu-north-1.elasticbeanstalk.com | eval-sys-project-version-2 | WebServer |
| Eval-sys-registration-env-docker | Ok | eval-sys-registration | Docker running on 64bit Amazon Linux 2023 | Eval-sys-registration-env-docker.eba-rwrpvejf.eu-north-1.elasticbeanstalk.co... | eval-sys-registration-version-3 | WebServer |
| Eval-sys-user-env-docker | Ok | eval-sys-user | Docker running on 64bit Amazon Linux 2023 | Eval-sys-user-env-docker.eba-qj3fh5wc.eu-north-1.elasticbeanstalk.com | eval-sys-user-version-8 | WebServer |

CONFLUENT for the Kafka cluster



# Thank you!

# Q&A