**Nodejs Micro service course content**

**Duration: 5 Days**

**Prerequisite**

**Participants should have strong knowledge on JavaScript, ECMA standards and Nodejs**

**Lab Setup**

**Nodejs 16.x**

**Jet brains Web storm 2022.x / Visual Studio Code**

**Mongodb 4.x**

**Docker Desktop with Kubernetes**

**Day 1 and Day2**

* Service-based architecture (done)
* From monolithic application to microservice practitioner code examples transition. (done)
* Services-oriented architecture code example. (done)

A screenshot of a computer

Description automatically generated

* Microservices: building blocks (done)
* Microservices: Solver of problems architecture (done)
* Microservices and cloud-native
* Microservices Core Advance Concepts
* Microservices Advanced Concepts

A screenshot of a computer

Description automatically generated with medium confidence

Day 3

**Creating a Service Registry**

* Setting up the registry
* Setting up endpoints in Express
* Registering and deregistering services
* Creating and testing the registration route
* Unregistering services
* Versioning and load balancing
* Querying the registry
* Removing expired services

**Splitting Up a Monolith into Microservices**

* Setting up the ecommerce service
* Registering the ecommerce service
* Unregistering a service on shutdown
* Adding the service logic
* Using the microservice from the main app
* Cleaning up the main app
* Creating an image serving endpoint

**Adding Fault Tolerance and Resilience**

* What happens if a service fails?
* Understanding circuit breakers
* Building a circuit breaker with Node
* Using the circuit breaker
* Using a cache to bridge outages
* Caching images

**Using Queues with Node.js**

* Factoring out the feedback service
* Using queues
* Setting up RabbitMQ
* Queuing feedback
* Consuming and storing feedback
* Asynchronous Communications (Kafka), (Rabbitmq)

Day 4

* Logging and tracing in a microservices architecture (ELK, Jaegar)
* Hybrid architectures: Hierarchy and service-based
* Making Architecture Choices
* Testing for microservices
* Secure microservices (JWT)
* Monitor microservices (Prometheus and Grafana)

Day 5

* Design considerations
* The tradeoffs
* An argument for edge services
* Embracing DevOps (Docker and Kubernetes)
* CI /Continuous Delivery as a requirement
* GitLab/GitHub
* Rancher (Kubernetes) – Harbor (Repo)