

How to develop your first cloud-native Applications with Java

Niklas Heidloff
Developer Advocate, IBM
@nheidloff

“Microservices are a software development technique [...] that structures an application as a collection of loosely coupled services.”

Wikipedia

What are cloud-native Applications?

Elasticity

→ App stays responsive

Continuous delivery

→ DevOps

New Options → New Challenges



“Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.”

kubernetes.io



kubernetes

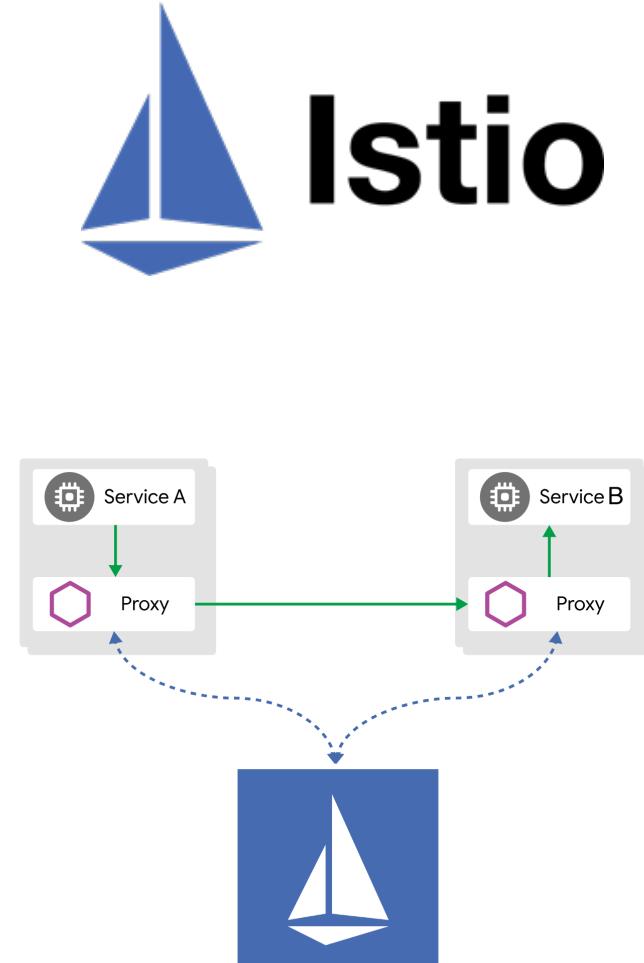
@nheidloff

#IBMDeveloper github.com/ibm/cloud-native-starter

“Istio is an open platform for providing a uniform way to integrate microservices, manage traffic flow across microservices, enforce policies and aggregate telemetry data.”

github.com/istio/istio

@nheidloff



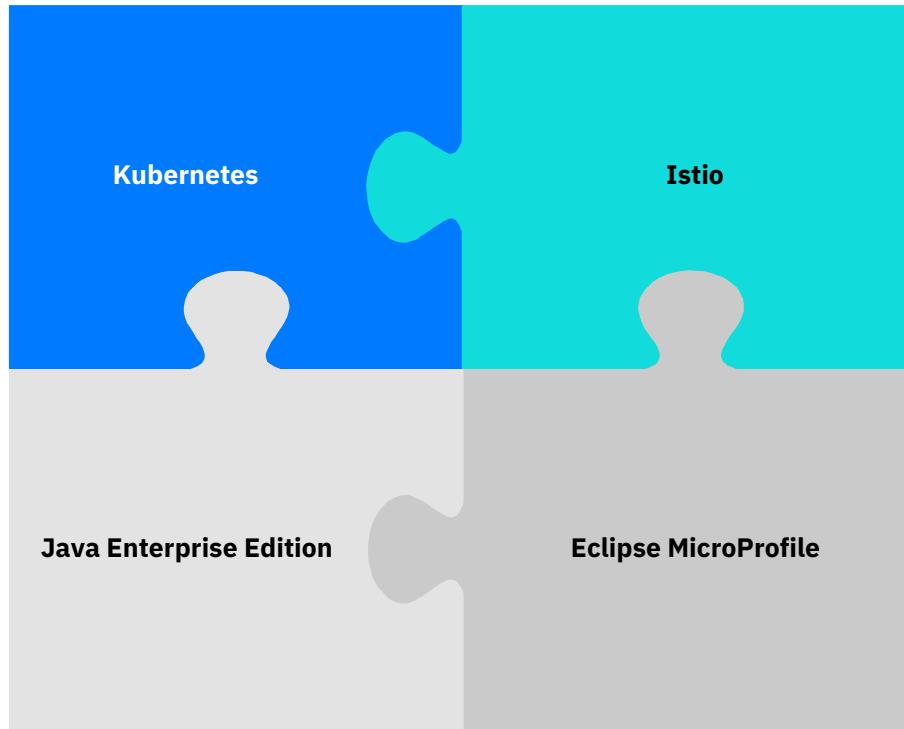
“Optimizing Enterprise Java for a Microservices Architecture.

[...] by innovating [...] with a
goal of standardization.”

microprofile.io



How to use all Pieces together?



Leverage platforms as
much as possible.

Use frameworks for app
specific logic.

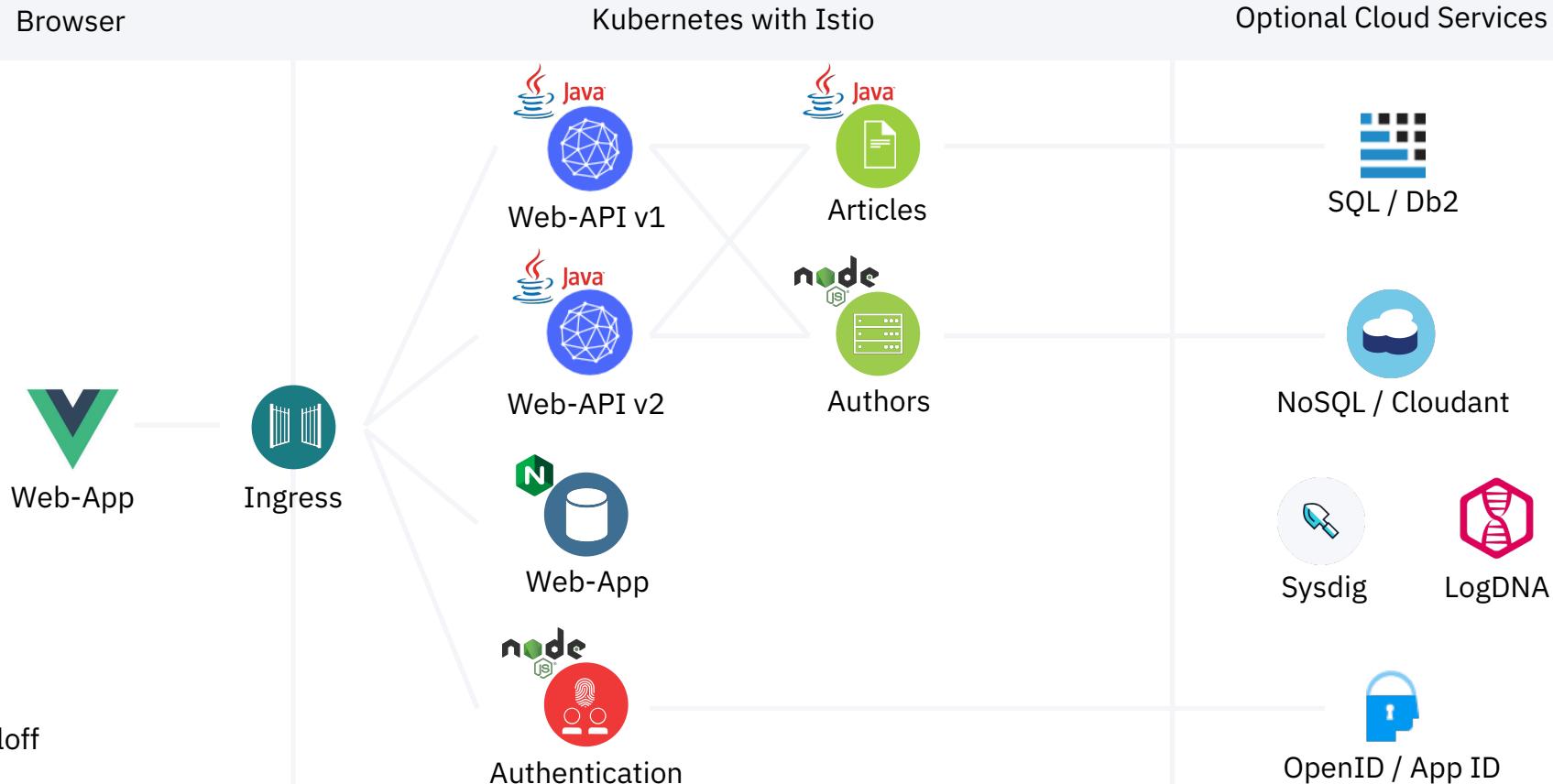
Design principles
for the end-to-
end example
'cloud-native-
starter'

Use only open-
source
components for
the core services
of the application

Make the first
time experience
as simple as
possible

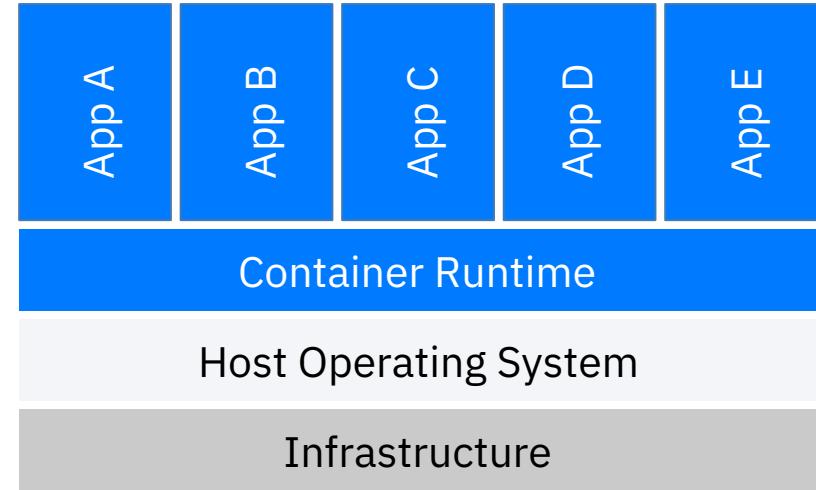
Be able to run the
application in
different
environments

Architecture: End-to-End Example ‘cloud-native-starter’

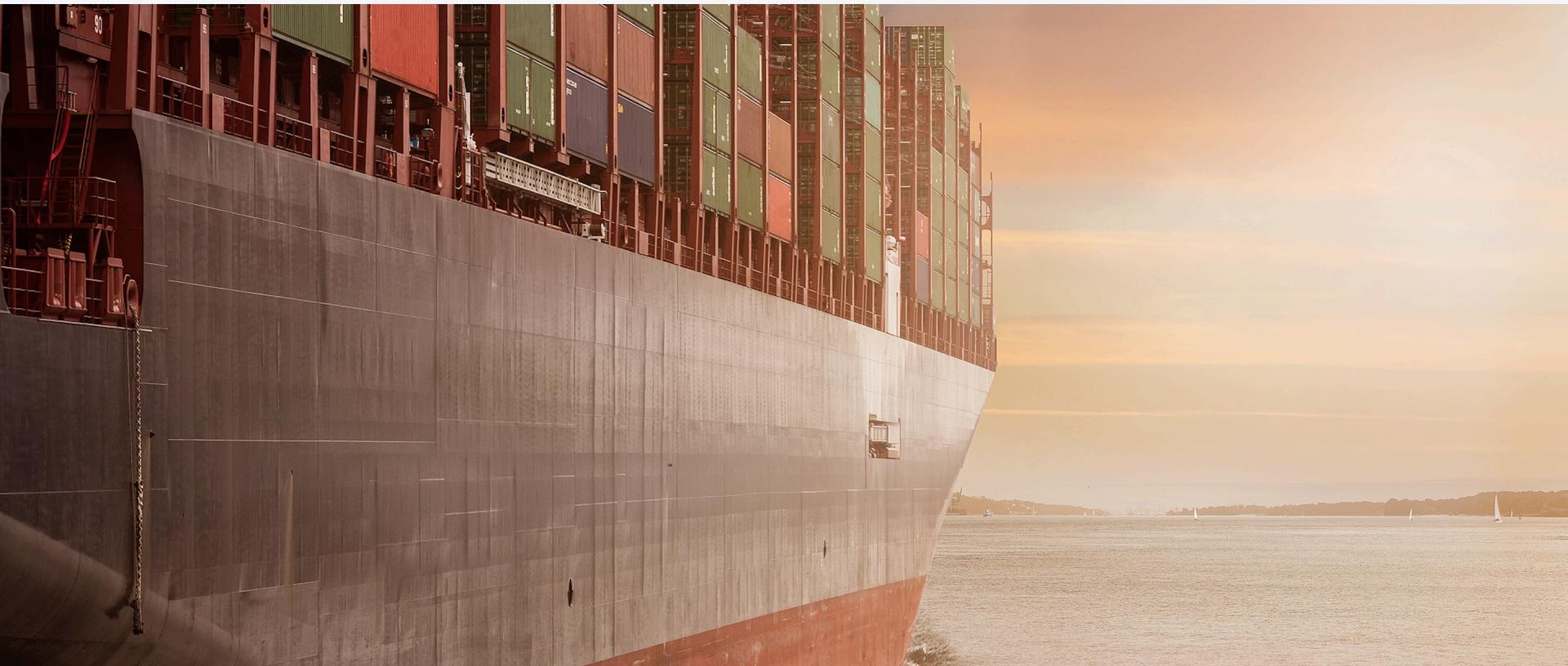


“A container image is a lightweight, standalone, executable package of software that includes everything needed to run an application.”

docker.com



Portable Containers



Java Image

Open source stack

OpenJ9 0.14.0

OpenJDK 11.0.3+7 from AdoptOpenJDK

Open Liberty 19.0.0.5

MicroProfile 2.2

Dockerfile

```
FROM open-liberty:microProfile2-java11
```

```
COPY liberty/server.xml /config/
```

```
COPY target/authors.war /config/apps/
```

Example Application

Cloud Native Starter

user@demo.email ▾

Articles



Title

[Debugging Microservices running in Kubernetes](#)

[Dockerizing Java MicroProfile Applications](#)

[Install Istio and Kiali on IBM Cloud or Minikube](#)

[Three awesome TensorFlow.js Models for Visual Recognition](#)

[Blue Cloud Mirror Architecture Diagrams](#)



Author

Niklas Heidloff

Niklas Heidloff

Harald Uebele

Niklas Heidloff

Niklas Heidloff



Twitter

[@nheidloff](#)

[@nheidloff](#)

[@harald_u](#)

[@nheidloff](#)

[@nheidloff](#)



Blog

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

Example Application – REST APIs

Browser

Kubernetes with Istio



Exposing REST APIs

JAX-RS

Java API for RESTful Web Services

GetArticles.java

```
@RequestScoped
@Path("/v1")
@OpenAPIDefinition(info = @Info(title = "Web-API Service",
|   version = "1.0", description = "Web-API Service APIs"))
public class GetArticles {

    @Inject
    com.ibm.webapi.business.Service service;

    @GET
    @Path("/getmultiple")
    @Produces(MediaType.APPLICATION_JSON)
    @APIResponses(value = {
        @APIResponse(responseCode = "200",
            description = "Get most recently added articles",
            content = @Content(mediaType = "application/json",
                schema = @Schema(type = SchemaType.ARRAY,
                    implementation = Article.class))),
        @APIResponse(responseCode = "500",
            description = "Internal service error") })
    @Operation(summary = "Get most recently added articles",
        description = "Get most recently added articles")
    public Response getArticles() {
```

Exposing REST APIs

Open API (formerly Swagger)

API description format for REST APIs

GetArticles.java

```
@RequestScoped
@Path("/v1")
@OpenAPIDefinition(info = @Info(title = "Web-API Service",
|   version = "1.0", description = "Web-API Service APIs"))
public class GetArticles {

    @Inject
    com.ibm.webapi.business.Service service;

    @GET
    @Path("/getmultiple")
    @Produces(MediaType.APPLICATION_JSON)
    @APIResponses(value = {
        @APIResponse(responseCode = "200",
            description = "Get most recently added articles",
            content = @Content(mediaType = "application/json",
                schema = @Schema(type = SchemaType.ARRAY,
                    implementation = Article.class))),
        @APIResponse(responseCode = "500",
            description = "Internal service error") })
    @Operation(summary = "Get most recently added articles",
        description = "Get most recently added articles")
    public Response getArticles() {
```

Exposing REST APIs

Open API (formerly Swagger)

API description format for REST APIs

The screenshot shows the Open Liberty Web-API Service interface. At the top, there's a header with the Open Liberty logo and the service name. Below the header, it says "Web-API Service 1.0 OAS3". A sub-header reads "Web-API Service APIs". Under "Server", the URL "http://192.168.99.100:31380/web-api" is listed. The main content area is titled "default". It shows a "POST /v1/create" endpoint for creating a new article. Below that is a "GET /v1/getmultiple" endpoint for getting most recently added articles, with a description "Get most recently added articles". Under "Parameters", it says "No parameters". There are "Execute" and "Clear" buttons. The "Responses" section is collapsed. In the "Curl" section, there's a command: `curl -X GET "http://192.168.99.100:31380/web-api/v1/getmultiple" -H "accept: application/json"`. The "Request URL" is `http://192.168.99.100:31380/web-api/v1/getmultiple`. The "Server response" section shows a table with a single row for code 200. The "Code" column is "200" and the "Details" column is "Response body". The response body is a JSON array containing two objects, each representing an article. The first object has fields: id (1555051929394), title (Example Java App running in the Cloud via Kubernetes), url (<http://heidloff.net/article/example-java-app-cloud-kubernetes>), authorName (Niklas Heidloff), authorBlog (<http://heidloff.net>), and authorTwitter (@nheidloff). The second object has a similar structure but its details are cut off.

Code	Details
200	Response body

```
[{"id": "1555051929394", "title": "Example Java App running in the Cloud via Kubernetes", "url": "http://heidloff.net/article/example-java-app-cloud-kubernetes", "authorName": "Niklas Heidloff", "authorBlog": "http://heidloff.net", "authorTwitter": "@nheidloff"}, {"id": "1555051929395", "title": "Another Example Article", "url": "http://heidloff.net/article/another-example-article", "authorName": "John Doe", "authorBlog": "http://john-doe.com", "authorTwitter": "@johndoe"}]
```

Consuming REST APIs

MicroProfile Rest Client

Type-safe approach to invoke RESTful services

AuthorsService.java

```
@RegisterProvider(ExceptionMapperArticles.class)
public interface AuthorsService {

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Author getAuthor(String name) throws NonexistentAuthor;
}
```

AuthorsServiceDataAccess.java

```
public class AuthorsServiceDataAccess implements AuthorsDataAccess {
    public AuthorsServiceDataAccess() {}

    static final String BASE_URL = "http://authors:3000/api/v1/";

    public Author getAuthor(String name) throws NoConnectivity, NonexistentAuthor {
        try {
            name = URLEncoder.encode(name, "UTF-8").replace("+", "%20");
            URL apiUrl = new URL(BASE_URL + "getauthor?name=" + name);
            AuthorsService customRestClient;
            customRestClient = RestClientBuilder.newBuilder().baseUrl(apiUrl)
                .register(ExceptionMapperAuthors.class).build(AuthorsService.class);

            Author output = customRestClient.getAuthor(name);
            return output;
        } catch (NonexistentAuthor e) {
            e.printStackTrace();
            throw new NonexistentAuthor(e);
        } catch (Exception e) {
            throw new NoConnectivity(e);
        }
    }
}
```

Consuming REST APIs

MicroProfile Rest Client

Type-safe approach to invoke RESTful services

AuthorsService.java

```
@RegisterProvider(ExceptionMapperArticles.class)
public interface AuthorsService {

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Author getAuthor(String name) throws NonexistentAuthor;
}
```

AuthorsServiceDataAccess.java

```
public class AuthorsServiceDataAccess implements AuthorsDataAccess {
    public AuthorsServiceDataAccess() {}

    static final String BASE_URL = "http://authors:3000/api/v1/";

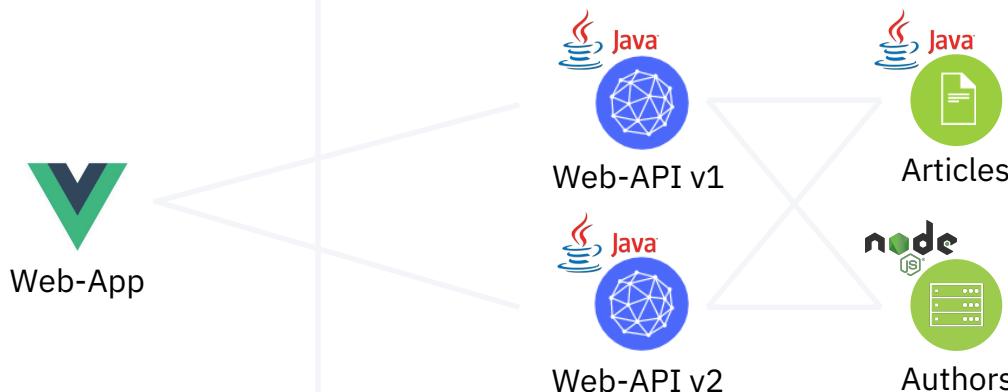
    public Author getAuthor(String name) throws NoConnectivity, NonexistentAuthor {
        try {
            name = URLEncoder.encode(name, "UTF-8").replace("+", "%20");
            URL apiUrl = new URL(BASE_URL + "getauthor?name=" + name);
            AuthorsService customRestClient;
            customRestClient = RestClientBuilder.newBuilder().baseUrl(apiUrl)
                .register(ExceptionMapperAuthors.class).build(AuthorsService.class);

            Author output = customRestClient.getAuthor(name);
            return output;
        } catch (NonexistentAuthor e) {
            e.printStackTrace();
            throw new NonexistentAuthor(e);
        } catch (Exception e) {
            throw new NoConnectivity(e);
        }
    }
}
```

Example Application – Traffic Management

Browser

Kubernetes with Istio



Traffic Management – Web API Version 1

Cloud Native Starter

Articles



Title

[Debugging Microservices running in Kubernetes](#)

[Dockerizing Java MicroProfile Applications](#)

[Install Istio and Kiali on IBM Cloud or Minikube](#)

[Three awesome TensorFlow.js Models for Visual Recognition](#)

[Blue Cloud Mirror Architecture Diagrams](#)



Author

Niklas Heidloff

Niklas Heidloff

Harald Uebel

Niklas Heidloff

Niklas Heidloff



Twitter

[@nheidloff](#)

[@nheidloff](#)

[@harald_u](#)

[@nheidloff](#)

[@nheidloff](#)



Blog

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

Traffic Management – Web API Version 2

Cloud Native Starter

Articles



Title

[Debugging Microservices running in Kubernetes](#)

[Dockerizing Java MicroProfile Applications](#)

[Install Istio and Kiali on IBM Cloud or Minikube](#)

[Three awesome TensorFlow.js Models for Visual Recognition](#)

[Blue Cloud Mirror Architecture Diagrams](#)

[Three Minutes Demo of Blue Cloud Mirror](#)

[IBM announced Managed Istio and Managed Knative](#)

[Developing and debugging Microservices with Java](#)

[Recent Java Updates from IBM](#)

[Blue Cloud Mirror — \(Don't\) Open The Doors!](#)



Author

Niklas Heidloff



Twitter

[@nheidloff](#)



Blog

[Blog](#)

Niklas Heidloff

[@nheidloff](#)

[Blog](#)

Harald Uebele

[@harald_u](#)

[Blog](#)

Niklas Heidloff

[@nheidloff](#)

[Blog](#)

Harald Uebele

[@harald_u](#)

[Blog](#)

Traffic Management

Example: 80% / 20% splitting

ingress.yaml

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtualservice-ingress-web-api-web-app
spec:
  hosts:
  - "*"
  gateways:
  - default-gateway-ingress-http
  http:
  - match:
    - uri:
        prefix: /web-api/v1/getmultiple
      route:
      - destination:
          host: web-api
          subset: v1
          weight: 80
      - destination:
          host: web-api
          subset: v2
          weight: 20
```

Traffic Management Demo

kiali

Namespace: default

Graph [?](#)

Display [?](#) Edge Labels [?](#) Graph Type Versioned app [?](#) Find... [x](#) Hide... [x](#) [?](#) Fetching Last min [?](#) Every 5 sec [?](#)

Apr 11, 11:25:29 ... Apr 11, 11:26:29

The graph illustrates the traffic flow between four services: web-app, web-api, articles, and authors. The traffic is categorized by version (v1 or v2) and percentage.

- web-app (v1)**: Directly receives 100% traffic from its own gateway.
- web-api (v1)**: Receives 80% traffic from its gateway and 20% traffic from the web-app gateway.
- articles (v1)**: Receives 16.7% traffic from the web-api gateway and 83.3% traffic from the authors gateway.
- authors (v1)**: Receives 91.2% traffic from the web-api gateway and 8.8% traffic from the web-app gateway.

Gateway traffic distribution:

- istio-ingressgateway (istio-system): 14.3% to web-api, 85.7% to web-app.

Namespace: default

Current Graph:
6 apps
4 services
11 edges

HTTP Traffic (requests per second):

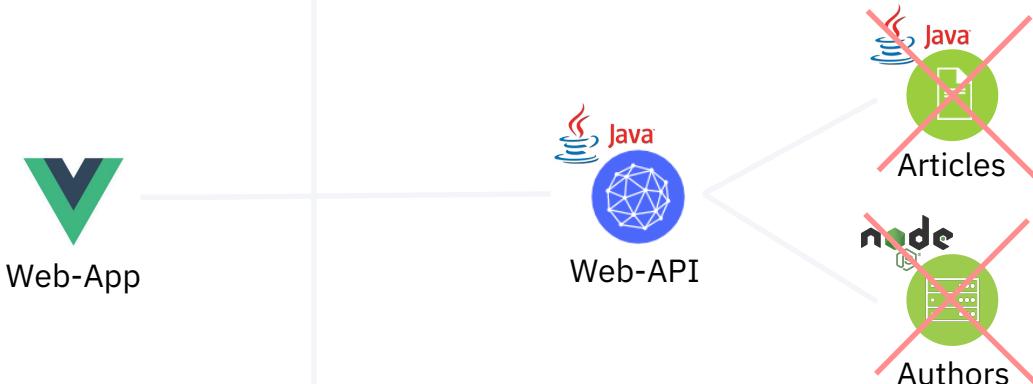
Total	%Success	%Error
6.65	100.00	0.00

Legend: OK (green), 3xx (blue), 4xx (orange), 5xx (red)

Example Application – Resiliency

Browser

Kubernetes with Istio



Web-App

Resiliency

Authors service not available

Usage of default values

Service.java

```
private List<Article> lastReadArticles;

public List<Article> fallbackNoArticlesService() {
    return lastReadArticles;
}

@Fallback(fallbackMethod = "fallbackNoArticlesService")
public List<Article> getArticles() throws NoDataAccess {

    List<Article> articles = new ArrayList<Article>();
    List<CoreArticle> coreArticles = new ArrayList<CoreArticle>();

    try {
        coreArticles = DataAccessManager.getArticlesDataAccess().getArticles(5);
    } catch (NoConnectivity e) {
        throw new NoDataAccess(e);
    }

    for (int index = 0; index < coreArticles.size(); index++) {
        CoreArticle coreArticle = coreArticles.get(index);
        Article article = new Article(coreArticle.id, coreArticle.title,
            coreArticle.title, coreArticle.author);
        try {
            Author author;
            author = DataAccessManager.getAuthorsDataAccess().getAuthor(coreArticle.author);
            article.authorBlog = author.blog;
            article.authorTwitter = author.twitter;
        } catch (Exception e) {
            article.authorBlog = "";
            article.authorTwitter = "";
        }
        articles.add(article);
    }
}
```

Resiliency

Articles service not available

MicroProfile fallback annotation

Service.java

```
private List<Article> lastReadArticles;

public List<Article> fallbackNoArticlesService() {
    return lastReadArticles;
}

@Fallback(fallbackMethod = "fallbackNoArticlesService")
public List<Article> getArticles() throws NoDataAccess {

    List<Article> articles = new ArrayList<Article>();
    List<CoreArticle> coreArticles = new ArrayList<CoreArticle>();

    try {
        coreArticles = DataAccessManager.getArticlesDataAccess().getArticles(5);
    } catch (NoConnectivity e) {
        throw new NoDataAccess(e);
    }

    for (int index = 0; index < coreArticles.size(); index++) {
        CoreArticle coreArticle = coreArticles.get(index);
        Article article = new Article(coreArticle.id, coreArticle.title,
        coreArticle.title, coreArticle.author);
        try {
            Author author;
            author = DataAccessManager.getAuthorsDataAccess().getAuthor(coreArticle.author);
            article.authorBlog = author.blog;
            article.authorTwitter = author.twitter;
        } catch (Exception e) {
            article.authorBlog = "";
            article.authorTwitter = "";
        }
        articles.add(article);
    }
}
```

Resiliency Demo

Cloud Native Starter

user@demo.email ▾

Articles



Title

[Debugging Microservices running in Kubernetes](#)

[Dockerizing Java MicroProfile Applications](#)

[Install Istio and Kiali on IBM Cloud or Minikube](#)

[Three awesome TensorFlow.js Models for Visual Recognition](#)

[Blue Cloud Mirror Architecture Diagrams](#)



Author

Niklas Heidloff

Niklas Heidloff

Harald Uebele

Niklas Heidloff

Niklas Heidloff



Twitter

[@nheidloff](#)

[@nheidloff](#)

[@harald_u](#)

[@nheidloff](#)

[@nheidloff](#)



Blog

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

[Blog](#)

Resiliency Demo

Cloud Native Starter

user@demo.email ▾

Articles



Title

[Debugging Microservices running in Kubernetes](#)

[Dockerizing Java MicroProfile Applications](#)

[Install Istio and Kiali on IBM Cloud or Minikube](#)

[Three awesome TensorFlow.js Models for Visual Recognition](#)

[Blue Cloud Mirror Architecture Diagrams](#)



Author

Niklas Heidloff

Niklas Heidloff

Harald Uebele

Niklas Heidloff

Niklas Heidloff



Twitter



Blog

Authentication and Authorization

OpenID Connect

Identity layer on top of the OAuth 2.0 protocol

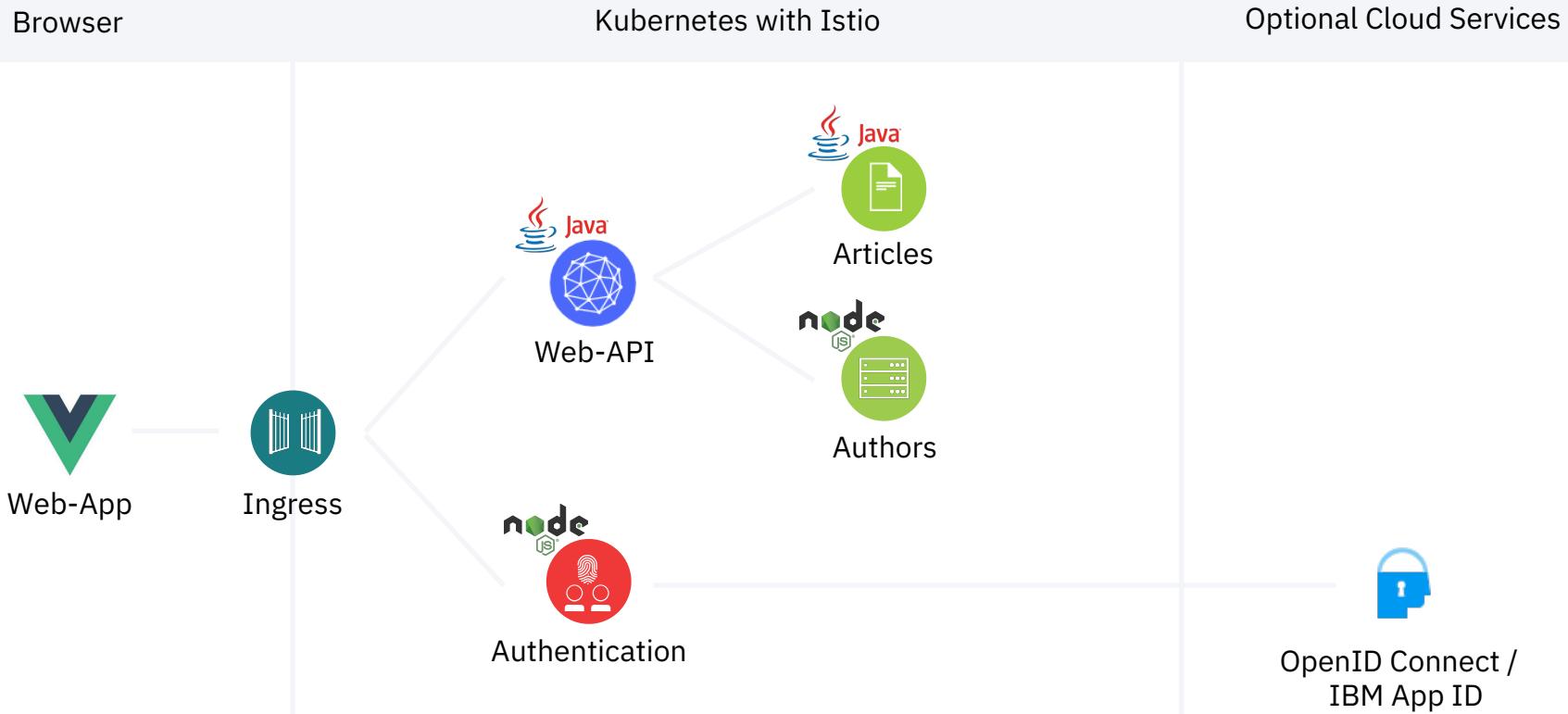
IBM App ID

IBM service to authenticate users and protect APIs

policy.yaml

```
apiVersion: "authentication.istio.io/v1alpha1"
kind: "Policy"
metadata:
  name: "protect-web-api"
spec:
  targets:
    - name: web-api
  origins:
    - jwt:
        issuer: "https://us-south.appid.cloud.ibm.com/oauth/v4/xxx"
        jwksUri: "https://us-south.appid.cloud.ibm.com/oauth/v4/xxx/publickeys"
        trigger_rules:
          - included_paths:
              - exact: /web-api/v1/create
  principalBinding: USE_ORIGIN
```

Example Application – Authentication and Authorization



Authentication



Email:

A text input field for entering an email address. It features a small icon of a person in a grey box on the left and the placeholder text "user@demo.email" on the right.

Password:

A password input field. It features a small icon of a lock in a grey box on the left and a series of five dots representing the password on the right.

[Forgot Password?](#)

Login

Don't have an account? [Sign up](#)

Authorization

Cloud Native Starter

user@demo.email ▾

Create new Article

Title:

Example Java App running in the Cloud via Kubernetes

URL:

<http://heidloff.net/article/example-java-app-cloud-kubernetes>

Author:

Niklas Heidloff

Submit

Show Articles

Authorization

Cloud Native Starter

user@demo.email ▾

Articles

Title	Author	Twitter	Blog
Example Java App running in the Cloud via Kubernetes	Niklas Heidloff	@nheidloff	Blog
Debugging Microservices running in Kubernetes	Niklas Heidloff	@nheidloff	Blog
Dockerizing Java MicroProfile Applications	Niklas Heidloff	@nheidloff	Blog
Install Istio and Kiali on IBM Cloud or Minikube	Harald Uebele	@harald_u	Blog
Three awesome TensorFlow.js Models for Visual Recognition	Niklas Heidloff	@nheidloff	Blog

Network

Filter Hide data URLs

All XHR JS CSS Img Media Font Doc WS Manifest Other

Name	Headers	Preview	Response	Timing
auth	General			
callback?code=w...				
loginwithtoken?na...				
app.9cd06f5a.css				
chunk-vendors.73...				
app.bfe2e0c6.js				
chunk-vendors.3e...				
getmultiple				
create	Response Headers			
getmultiple				

Request URL: http://192.168.99.100:31380/web-api/v1/create
Request Method: POST
Status Code: 201 Created
Remote Address: 192.168.99.100:31380
Referrer Policy: no-referrer-when-downgrade

access-control-allow-credentials: true
access-control-allow-headers: origin, content-type, accept, aut...
access-control-allow-methods: GET, POST, PUT, DELETE, OPTIONS,
access-control-allow-origin: *
content-language: en-US
content-length: 182
content-type: application/json
date: Fri, 12 Apr 2019 06:52:09 GMT
server: istio-envoy
x-envoy-upstream-service-time: 346
x-powered-by: Servlet/4.0

Request Headers

⚠ Provisional headers are shown

Accept: application/json, text/plain, */*
Authorization: Bearer eyJhbGci...
KLTrMTQ2NGZnLmNKOTM1NDM5Ii...
Iy0jM1ljk5NCi...InZlcii6NH0.e...
3VKLmlibS5jb20vb2f1dGvdjQvN...
ZTY0liw1iXVkiobImVUOYxNDc2L...

Authorization

Via MicroProfile

server.xml

```
<server description="OpenLiberty Server">

    <featureManager>
        <feature>webProfile-8.0</feature>
        <feature>microProfile-2.1</feature>
        <feature>usr:opentracingZipkin-0.31</feature>
        <feature>monitor-1.0</feature>
        <feature>mpJwt-1.1</feature>
        <feature>appSecurity-3.0</feature>
    </featureManager>

    <httpEndpoint id="defaultHttpEndpoint" host="*"
        httpPort="9080" httpsPort="9443"/>

    <mpMetrics authentication="false"/>

    <mpJwt id="jwt"
        issuer="https://us-south.appid.cloud.ibm.com/oauth/v4/xxx"
        jwksUri="https://us-south.appid.cloud.ibm.com/oauth/v4/xxx/publickeys"
        userNameAttribute="sub"
        audiences="ALL_AUDIENCES"/>

    <sslDefault sslRef="RpSSLConfig"/>
    <ssl id="RpSSLConfig" keyStoreRef="defaultKeyStore"
        trustStoreRef="validationKeystore"/>
    <keyStore id="defaultKeyStore" location="keystore.jceks"
        type="JCEKS" password="secret" />
    <keyStore id="validationKeystore" location="/config/key.jks"
        type="jks" password="changeit"/>
</server>
```

Authorization

Via MicroProfile

Manage.java

```
@RequestScoped  
@Path("/v1")  
public class Manage {  
  
    @Inject  
    private JsonWebToken jwtPrincipal;  
  
    @POST  
    @Path("/manage")  
    @Produces(MediaType.APPLICATION_JSON)  
    @Operation(summary = "Manage app", description = "Manage app")  
    public Response manage() {  
        System.out.println("com.ibm.web-api.apis.Manage.manage");  
        System.out.println(this.jwtPrincipal);  
  
        String principalEmail = this.jwtPrincipal.getClaim("email");  
        if (principalEmail.equalsIgnoreCase("admin@demo.email")) {  
            JsonObject output = Json.createObjectBuilder()  
                .add("message", "success").build();  
            return Response.ok(output).build();  
        }  
    }  
}
```

Observability

Tracing	Microservices vs monolith	Chained invocations	Kubernetes
Logging	→ Higher complexity		→ 1 service = N pods
Monitoring	→ Ephemeral		
Metrics			
Healthchecks			

Example Application – Distributed Logging

Browser

Kubernetes with Istio



Distributed Logging

The screenshot shows a distributed logging interface with the following elements:

- Left Sidebar:** Icons for Find a View, EVERYTHING, VIEWS, CloudNative, and various monitoring and configuration tools.
- Top Bar:** Filter dropdowns for CloudNative, All Tags, All Sources, 4 Apps, and All Levels.
- Log List:** A list of log entries from different services. The entries are:
 - May 16 14:44:23 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:23.317] [INFO] authors_service - 200 - for: Niklas Heidloff (not retained)
 - May 16 14:44:23 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:23.318] [INFO] authors_service - ::ffff:127.0.0.1 - - "GET /api/v1/getauthor?name=Niklas%20Heidloff HTTP/1.1" 200 - "" "Apache-CXF/3.2.6" (not retained)
 - May 16 14:44:24 articles-76bbfcdfb7-mqxm4 articles com.ibm.articles.apis.GetArticles.getArticles (not retained)
 - May 16 14:44:24 web-api-v1-fc5f475c5-f5vlp web-api com.ibm.web-api.apis.GetArticles.getArticles (not retained)
 - May 16 14:44:24 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:24.367] [INFO] authors_service - Query for: Niklas Heidloff (not retained)
 - May 16 14:44:24 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:24.523] [INFO] authors_service - 200 - Cloudant read successful (not retained)
 - May 16 14:44:24 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:24.523] [INFO] authors_service - ::ffff:127.0.0.1 - - "GET /api/v1/getauthor?name=Niklas%20Heidloff HTTP/1.1" 200 - "" "Apache-CXF/3.2.6" (not retained)
 - May 16 14:44:24 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:24.530] [INFO] authors_service - Query for: Niklas Heidloff (not retained)
 - May 16 14:44:24 authors-6d48cff748-gjzh8 authors [2019-05-16T12:44:24.683] [INFO] authors_service - 200 - Cloudant read successful (not retained)
- Bottom Bar:** Includes a search input, a 'Jump to timeframe' button, and navigation icons for LIVE, LITE, and other functions.

Example Application – Distributed Logging

Browser

Kubernetes with Istio



Distributed Logging

The screenshot shows a distributed logging interface with the following interface elements:

- Left Sidebar:** Icons for Find a View, EVERYTHING, VIEWS, CloudNative, and various monitoring metrics.
- Top Bar:** Filter dropdowns for CloudNative, All Tags, All Sources, 4 Apps, and All Levels.
- Log List:** A list of log entries. The first five entries are from the `web-api` service, and the subsequent entries are from the `articles` service. All logs show the same error message: "Cannot connect to authors service (not retained)".
- Log Selection:** The log entries for the `articles` service are highlighted with a blue rectangular selection box.
- Bottom Bar:** Includes a search bar, a "Jump to timeframe" button, and other navigation controls.

```
CloudNative ▾
All Tags ▾ All Sources ▾ 4 Apps ▾ All Levels ▾

Cannot connect to authors service (not retained)
May 16 14:48:48 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:48 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:48 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:48 articles-76bbfcdfb7-mqxm4 articles com.ibm.articles.apis.GetArticles.getArticles (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api com.ibm.web-api.apis.GetArticles.getArticles (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:49 web-api-v1-fc5f475c5-f5vlp web-api stderr [err] com.ibm.webapi.business.getArticles:
    Cannot connect to authors service (not retained)
May 16 14:48:49 articles-76bbfcdfb7-mqxm4 articles com.ibm.articles.apis.GetArticles.getArticles (not retained)
```

Tracing

OpenTracing

Vendor-neutral APIs and instrumentation for distributed tracing

Jaeger and Zipkin

Open source distributed tracing systems

server.xml

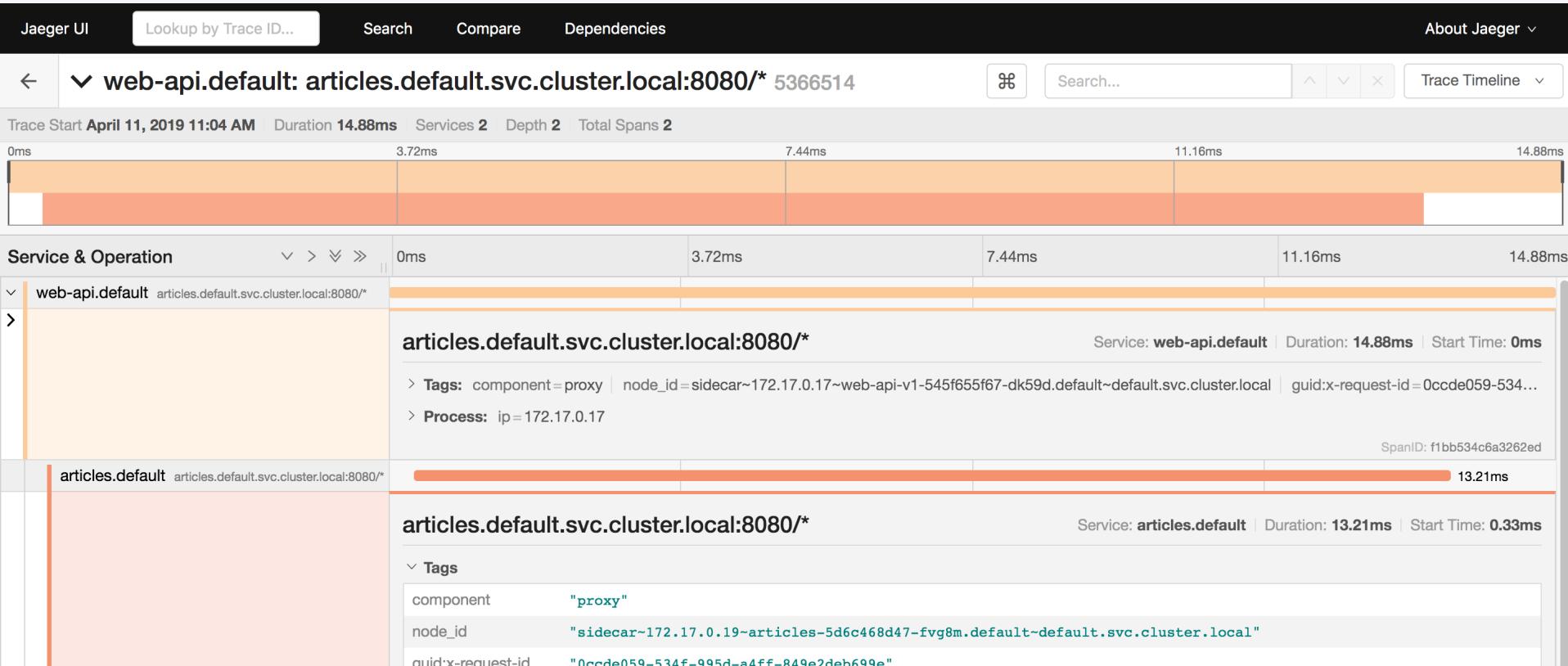
```
<?xml version="1.0" encoding="UTF-8"?>
<server description="OpenLiberty Server">

    <featureManager>
        <feature>webProfile-8.0</feature>
        <feature>microProfile-2.1</feature>
        <feature>usr:opentracingZipkin-0.31</feature>
    </featureManager>

    <httpEndpoint id="defaultHttpEndpoint" host="*"
        httpPort="8080" httpsPort="9443"/>

</server>
```

Distributed Tracing



Metrics

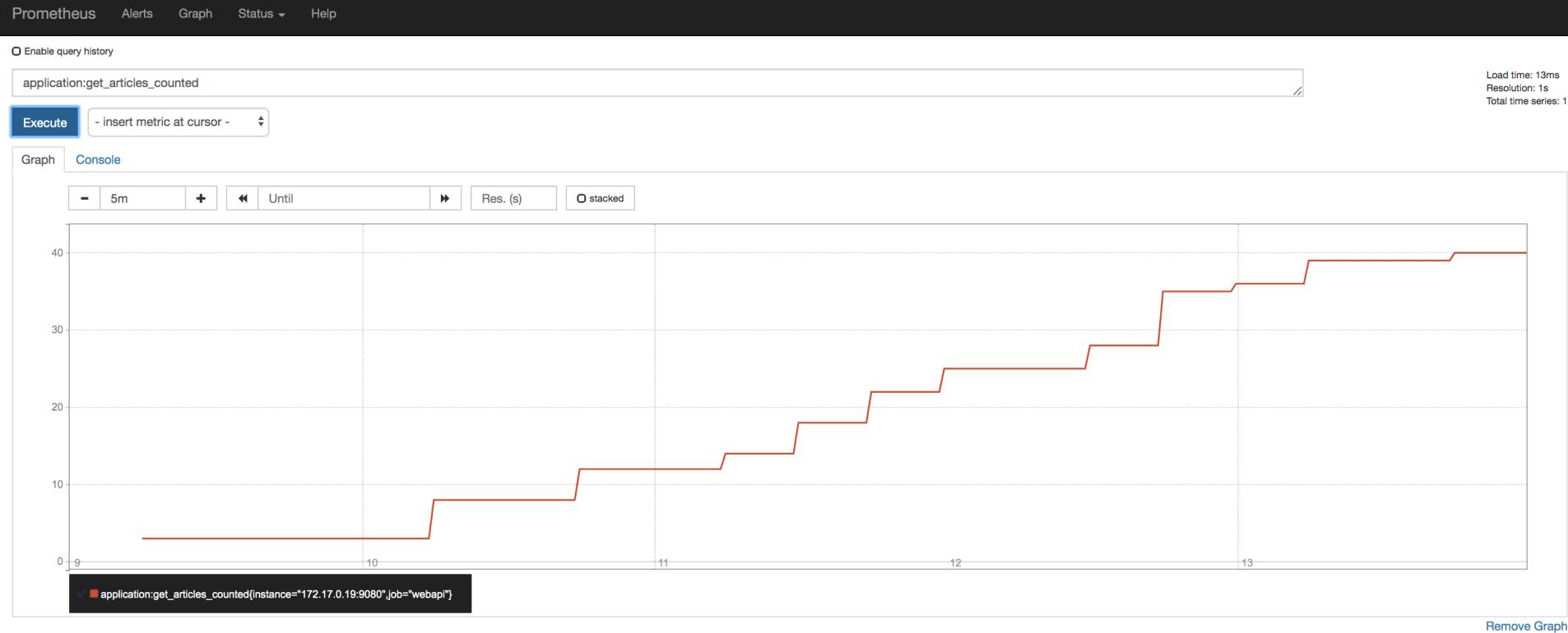
Prometheus

Monitoring system and time series database

GetArticles.java

```
@Timed(name = "getArticlesTimed",
        absolute = true,
        displayName = "web-api /getmultiple timer",
        description = "Time taken by getArticles")
@Counted(name = "getArticlesCounted",
        absolute = true,
        displayName = "web-api /getmultiple count",
        description = "Number of times getArticles has been invoked",
        monotonic = true)
@Metered(name = "getArticlesMetered",
        displayName = "web-api /getmultiple frequency",
        description = "Rate the throughput of getArticles")
@GET
@Path("/getmultiple")
@Produces(MediaType.APPLICATION_JSON)
public Response getArticles() {
```

Metrics



Healthchecks

MicroProfile Health

Liveness probes and readiness probes

HealthEndpoint.java

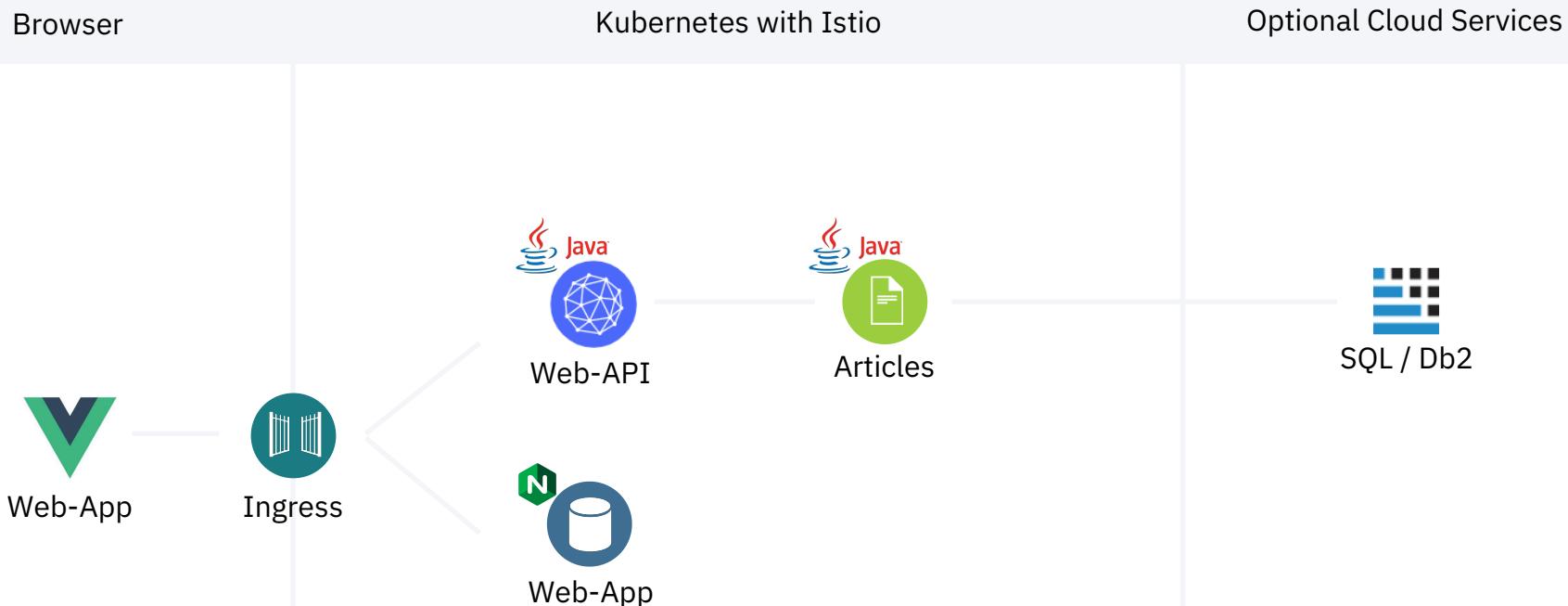
```
@Health
@ApplicationScoped
public class HealthEndpoint implements HealthCheck {

    @Override
    public HealthCheckResponse call() {
        return HealthCheckResponse.named("web-api").withData("web-api", "ok").up().build()
    }
}
```

Service.yaml

```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: web-api-v1
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: web-api
        version: v1
    spec:
      containers:
        - name: web-api
          image: web-api:1
          ports:
            - containerPort: 9080
      livenessProbe:
        exec:
          command: ["sh", "-c", "curl -s http://localhost:9080/"]
        initialDelaySeconds: 20
      readinessProbe:
        exec:
          command: ["sh", "-c", "curl -s http://localhost:9080/health | grep -q web-api"]
        initialDelaySeconds: 40
  restartPolicy: Always
```

Persistence via JPA



Persistence

Via JPA

server.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="OpenLiberty Server">

    <featureManager>
        <feature>webProfile-8.0</feature>
        <feature>microProfile-2.1</feature>
        <feature>usr:opentracingZipkin-0.31</feature>
    </featureManager>

    <httpEndpoint id="defaultHttpEndpoint" host="*"
                  httpPort="8080" httpsPort="9443"/>

    <library id="DB2JCCLib">
        <fileset dir="${shared.resource.dir}"
                 includes="jcc*.jar"/>
    </library>

    <dataSource id="articlejpadatasource"
               jndiName="jdbc/articlejpadatasource">
        <jdbcDriver libraryRef="DB2JCCLib" />
        <properties.db2.jcc databaseName="BLUDB"
                             portNumber="50000"
                             serverName="DB2-SERVER"
                             user="DB2-USER"
                             password="DB2-PASSWORD" />
    </dataSource>
</server>
```

Persistence

Via JPA

ArticleEntity.java

```
@Entity(name = "Article")
@Table(name = "Article")
@NamedQuery(name = "Article.findAll", query = "SELECT e FROM Article e")
@NamedQuery(name = "Article.findArticle", query = "SELECT a FROM Article a WHERE "
    + "a.title = :title AND a.url = :url AND a.author = :author")
public class ArticleEntity implements Serializable {
    private static final long serialVersionUID = 1L;

    @GeneratedValue(strategy = GenerationType.AUTO)
    @Id
    @Column(name = "articleId")
    private int id;

    @Column(name = "articleTitle")
    private String title;

    @Column(name = "articleUrl")
    private String url;

    @Column(name = "articleAuthor")
    private String author;

    @Column(name = "creationDate")
    private String creationDate;

    public ArticleEntity() {
}
```

Configuration

Via MicroProfile

kubernetes.yaml

```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
|   name: articles
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: articles
        version: v1
    spec:
      containers:
        - name: articles
          image: articles:1
          ports:
            - containerPort: 8080
          env:
            - name: samplescreation
              valueFrom:
                configMapKeyRef:
                  name: articles-config
                  key: samplescreation
            - name: inmemory
              valueFrom:
                configMapKeyRef:
                  name: articles-config
                  key: inmemory
          restartPolicy: Always
---
kind: ConfigMap
apiVersion: v1
metadata:
|   name: articles-config
data:
  samplescreation: CREATE
  inmemory: USE_IN_MEMORY_STORE
```

Configuration

Via MicroProfile

CoreService.java

```
@ApplicationScoped
public class CoreService {

    private static final String CREATE_SAMPLES = "CREATE";
    private static final String USE_IN_MEMORY_STORE = "USE_IN_MEMORY_STORE";

    @Inject
    @ConfigProperty(name = "inmemory", defaultValue = USE_IN_MEMORY_STORE)
    private String inmemory;

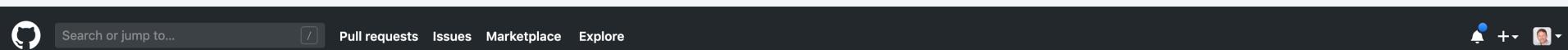
    @Inject
    @ConfigProperty(name = "samplescreation", defaultValue = "dontcreate")
    private String samplescreation;

    @Inject
    private DataAccessManager dataAccessManager;

    @PostConstruct
    private void addArticles() {
        if (inmemory.equalsIgnoreCase(USE_IN_MEMORY_STORE)) {
            if (samplescreation.equalsIgnoreCase(CREATE_SAMPLES))
                addSampleArticles();
        }
    }
}
```

Try out the end-to-end
microservices example
cloud-native-starter!

Documentation



Cloud Native Starter for Java/Jakarta EE based Microservices on Kubernetes and Istio

This project demonstrates several Java EE and Istio key functionality.

- [Containerized Java EE Microservices](#)
- [Exposing REST APIs](#)
- [Consuming REST APIs](#)
- [Traffic Routing](#)
- [Resiliency](#)
- [Authentication and Authorization](#)
- [Metrics](#)
- [Health Checks](#)
- [Configuration](#)
- [Distributed Logging and Monitoring](#)
- [Persistence with Java Persistence API \(JPA\)](#)

IBM Cloud Kubernetes Service including Istio and Knative

IBM Cloud Search resources and offerings... Catalog Docs Support Manage Niklas Heidloff's Account ⚙️ 📜

Clusters / niklas-heidloff-cns

 niklas-heidloff-cns • Normal

[Web Terminal \(beta\)](#) [Kubernetes Dashboard ↗](#) [Connect via CLI](#) ⋮

Access [Overview](#) Worker Nodes Worker Pools Add-ons

Summary

Cluster ID	401c8d4144a744f6978c68a12c8335c5
Master Status	Ready
Kubernetes version	1.12.7_1548
Zones	hou02
Owner	niklas_heidloff@de.ibm.com
Resource group	default
Key protect (Beta)	Enable
IAM pullsecrets	Enabled
Public service endpoint URL	https://c5.dal12.containers.cloud.ibm.com:31446 Disable

Worker Nodes 1



1	Normal
0	Warning
0	Critical
0	Pending

Summary

Get the code →



Leverage platforms as much as possible

Use frameworks for app specific logic

IBM loves open source

Kubernetes and Istio
OpenJ9 & AdoptOpenJDK
MicroProfile
Open Liberty

IBM Developer

developer.ibm.com

IBM Cloud Lite account

ibm.biz/nheidloff

