



# Mantendo seu monolito vivo

## em um ambiente de alta disponibilidade

---

Guilherme Carreiro - Senior Software Engineer at Red Hat

Paulo Martins - Senior Software Engineer at Red Hat

TDC Porto Alegre

28.11.2019



# Mantendo seu monolito vivo

## em um ambiente de alta disponibilidade

---

Guilherme Carreiro - Senior Software Engineer at Red Hat

Paulo Martins - Senior Software Engineer at Red Hat

TDC Porto Alegre

28.11.2019



## Guilherme Carreiro - @karreiro

- São Paulo, SP
- DMN
- Go language
- Data Privacy
- Weekly blogger - [karreiro.com](http://karreiro.com)
- Red Hat / Middleware (Decision Tooling)



bla.dmn — alexa-karreiro

EXPLORER

- ▼ ALEXA-KARREIRO
  - > node\_modules
  - > views
    - .gitignore
    - alex-request-flow.JPG
    - app.json
    - index.js
    - package-lock.json
    - package.json
    - Procfile
    - README.md

NPM SCRIPTS

MAVEN PROJECTS

bla.dmn

Editor Documentation Data Types Included Models

can drive?

age

master\* 0 0 0



## Paulo Martins - @paulovmr

- Campinas, SP
- Computer Scientist
- React
- Foundation Blog - <https://medium.com/kie-foundation>
- Red Hat / Middleware (Foundation)





localhost

Business Central

Menu

admin

Spaces » workspace » project » master



project

TestBuildDeployView Alerts

Assets 2Change Requests 0Contributors 1MetricsSettings

All

1-2 of 21 of 1Import AssetAdd Asset

|   |        |              |                          |                    |
|---|--------|--------------|--------------------------|--------------------|
|  | model  | DMN          | Last modified 1 week ago | Created 1 week ago |
|  | Person | Data Objects | Last modified 1 week ago | Created 1 week ago |

# The Monolith

16Gb of RAM to run a single app 🥰







# Monolith features

- Simple deployment
- Built-in tracing
- Simple packaging
- Easy reusability

# Monolith features

- Deployment coupling
- Code coupling
- Inflexible scalability
- Lack of elasticity



# The Monolith perspective

# The Monolith perspective



# Cloud-native apps and Containers

The Microservices sweet spot

# Linux Containers (LXC)

Namespaces, Control Groups, and  
Copy-on-write filesystem



O'REILLY

# Building Microservices

DESIGNING FINE-GRAINED SYSTEMS



Sam Newman

Building Microservices

Newman

O'REILLY

B. Fernandez

SECURITY PATTERNS  
IN PRACTICE



WILEY

GARY MCGRAW



SOFTWARE  
SECURITY



Addison  
Wesley

The Tangled Web

A Guide to Securing Modern Web Applications

Zalewski





# Containers with Microservices

- Independent deployments
- Flexible scalability
- Horizontal scalability
- High elasticity



# Containers with Microservices

- Clear understanding of the domain
- Tracing
- Orchestration vs. Choreography

# The Cloud-native apps perspective



We want it.

...and nobody is gonna die!



# The Monolith perspective



The Monolith  
perspective

Cloud-native  
apps and  
Containers

# Business Central

Old but gold — not legacy.

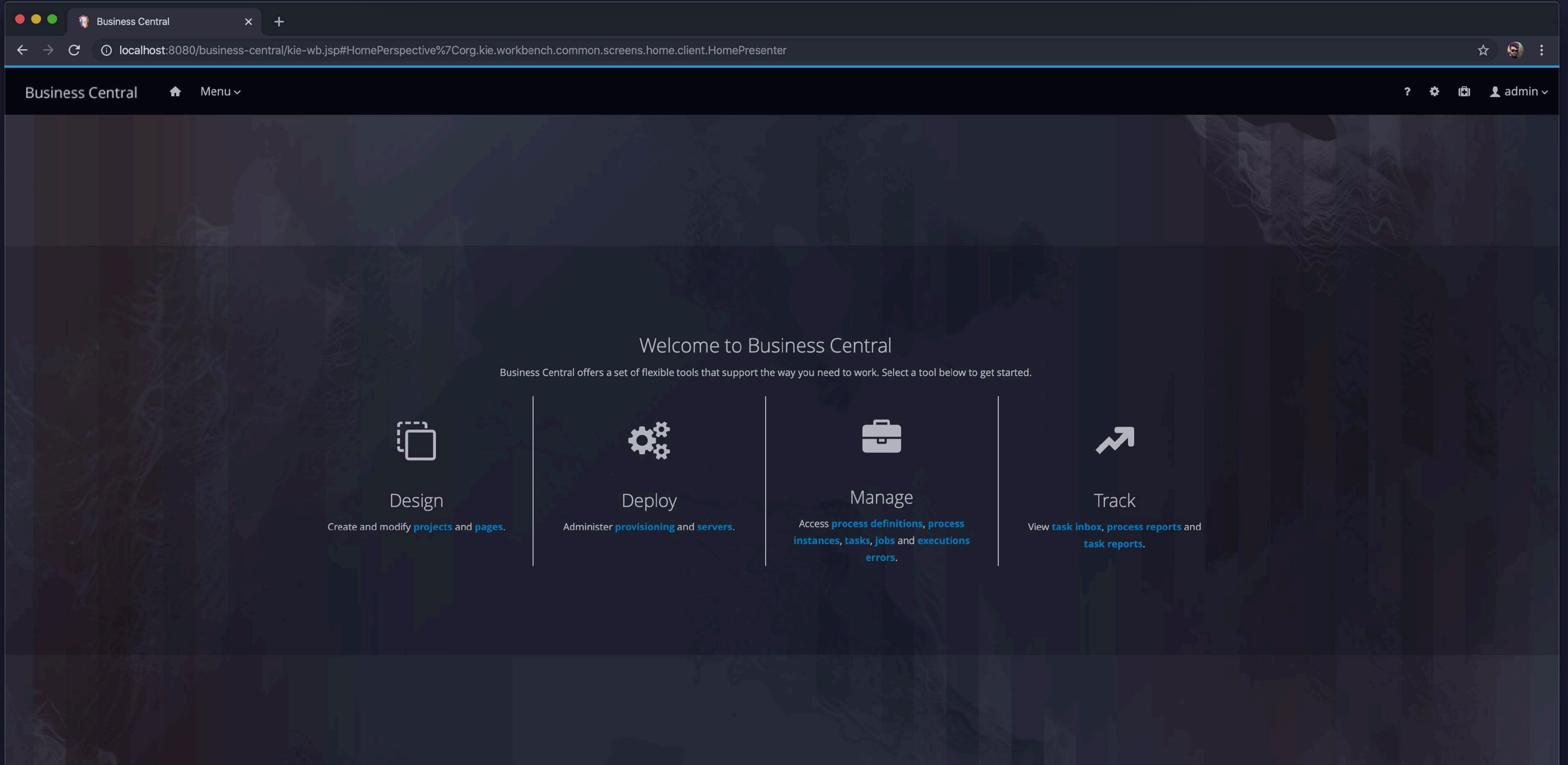
# Business Central

## 188Mb WAR:

- more than 8 years of code base
- more than 1 million of LOC
- more than 30 developers
- more than 150 sub-projects
- **very good balance between coupling and cohesion**



# Business Central platform



Business Central



## Welcome to Business Central

Business Central offers a set of flexible tools that support the way you need to work. Select a tool below to get started.



### Design

Create and modify **projects** and **pages**.



### Deploy

Administer **provisioning** and **servers**.



### Manage

Access **process definitions**, **process instances**, **tasks**, **jobs** and **executions errors**.



### Track

View **task inbox**, **process reports** and **task reports**.

# Business Central filesystem

Business Central

Menu

?

⚙

📁

admin

Spaces » workspace » project » master



project

TestBuildDeployView Alerts

Assets2Change Requests0Contributors1MetricsSettings

All

1-2 of 21 of 1Import AssetAdd Asset

|   |              |                     |               |
|---|--------------|---------------------|---------------|
|  <b>model</b>    | DMN          | Last modified today | Created today |
|  <b>Person</b> | Data Objects | Last modified today | Created today |



# Business Central assets

Business Central

Menu





















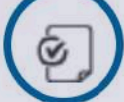


admin

Spaces » workspace » project » master

Add Asset

Cancel

All

|  |   |  |   |
|--|---|--|---|
|  <div>Business Process<br/>Process</div>          |  <div>Business Process (legacy)<br/>Process</div>   |  <div>Data Object<br/>Model</div>                   |  <div>Decision Table (Spreadsheet)<br/>Decision</div>  |
|  <div>DMN<br/>Decision</div>                    |  <div>DRL file<br/>Decision</div>                 |  <div>DSL definition<br/>Decision</div>           |  <div>Enumeration<br/>Model</div>                    |
|  <div>Form<br/>Form</div>                       |  <div>Global Variable(s)<br/>Decision</div>       |  <div>Guided Decision Table<br/>Decision</div>    |  <div>Guided Decision Table Graph<br/>Decision</div> |
|  <div>Guided Decision Tree<br/>Decision</div>   |  <div>Guided Rule<br/>Decision</div>              |  <div>Guided Rule Template<br/>Decision</div>     |  <div>Guided Score Card<br/>Decision</div>           |
|  <div>Package<br/>Others</div>                  |  <div>Score Card (Spreadsheet)<br/>Decision</div> |  <div>Solver configuration<br/>Optimization</div> |  <div>Test Scenario<br/>Decision</div>               |
|  <div>Test Scenario (Legacy)<br/>Decision</div> |  <div>Work Item definition<br/>Others</div>       |  <div>Case definition (legacy)<br/>Process</div>  |   |

# Business Central editors (Data Objects)

The screenshot displays the Business Central web application interface. The browser's address bar shows the URL: `localhost:8080/business-central/kie-wb.jsp#LibraryPerspective%7C$ProjectScreen,AddAssetsScreen,DataModelerEditor?path_uri=default://master@workspace/project/src/main/java/com/workspace/project/Person.java&file_name=Person.java&has_version_support...`. The application header includes the "Business Central" logo, a home icon, a "Menu" dropdown, and a user profile icon labeled "admin". The breadcrumb navigation path is: Spaces » workspace » project » master » Person. The main content area is titled "Person.java - Data Objects" and features a toolbar with buttons for Save, Delete, Rename, Copy, Validate, Download, Latest Version, and Hide Alerts. Below the title bar, there are tabs for "Model", "Overview", and "Source", with "Source" being the active tab. The source code editor displays the following Java code:

```
1 package com.workspace.project;
2
3 /**
4  * This class was automatically generated by the data modeler tool.
5  */
6
7 public class Person implements java.io.Serializable {
8
9     static final long serialVersionUID = 1L;
10
11     private java.lang.Integer id;
12     private java.lang.String name;
13
14     public Person() {}
15
16
17     public java.lang.Integer getId() {
18         return this.id;
19     }
20
21     public void setId(java.lang.Integer id) {
22         this.id = id;
23     }
24
25     public java.lang.String getName() {
26         return this.name;
27     }
28
29     public void setName(java.lang.String name) {
30         this.name = name;
31     }
32
33     public Person(java.lang.Integer id, java.lang.String name) {
34         this.id = id;
35         this.name = name;
36     }
37
38 }
```



# Business Central editors (DMN models)

Business Central

Spaces » workspace » project » master » model

Decision Navigator

Decision Graphs

- model
  - Decision-1
  - InputData-1

Decision Components

Filter by

Enter text

No external models have been included.  
To add an external model to this DMN file, go to the Include Models tab and include a model.

model.dmn - DMN

Model Overview Documentation Data Types Included Models

Save Delete Rename Copy [Icons]

Decision-1

InputData-1

```
graph LR; InputData-1([InputData-1]) --> Decision-1[Decision-1];
```



# Business Central runtime

Business Central

Menu

?

⚙

📁

admin

SERVER CONFIGURATIONS

+ New Server Configuration

server

server

Capabilities:

☒ Decision

☒ Process

☐ Planner

DEPLOYMENT UNITS

+ Add Deployment Unit

com.myspace:aaa:1.0.0-SNAPSHOT

REMOTE SERVERS

[ Group Id:com.myspace | Artifact Id:aaa ]

Start

Stop

Deactivate

Remove

Status

Version Configuration

Process Configuration

No Remote Servers

This Deployment Unit is currently not in use by any Remote Servers.

Refresh

# Monolith-to-Cloud challenges

Here we go!

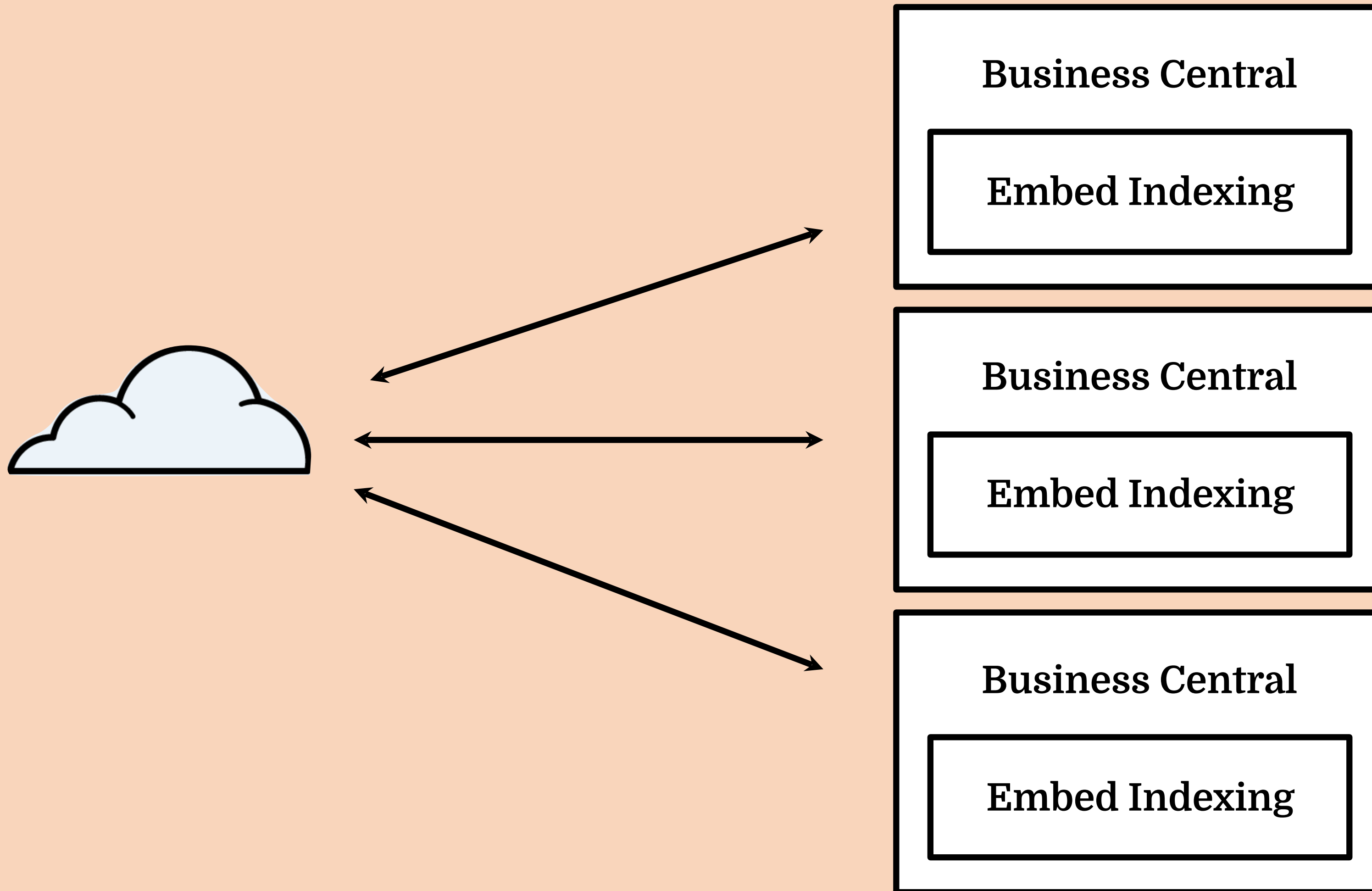
# Monolith-to-Cloud challenges

1. Redundant processing
2. Global lock
3. Keep the programming paradigm



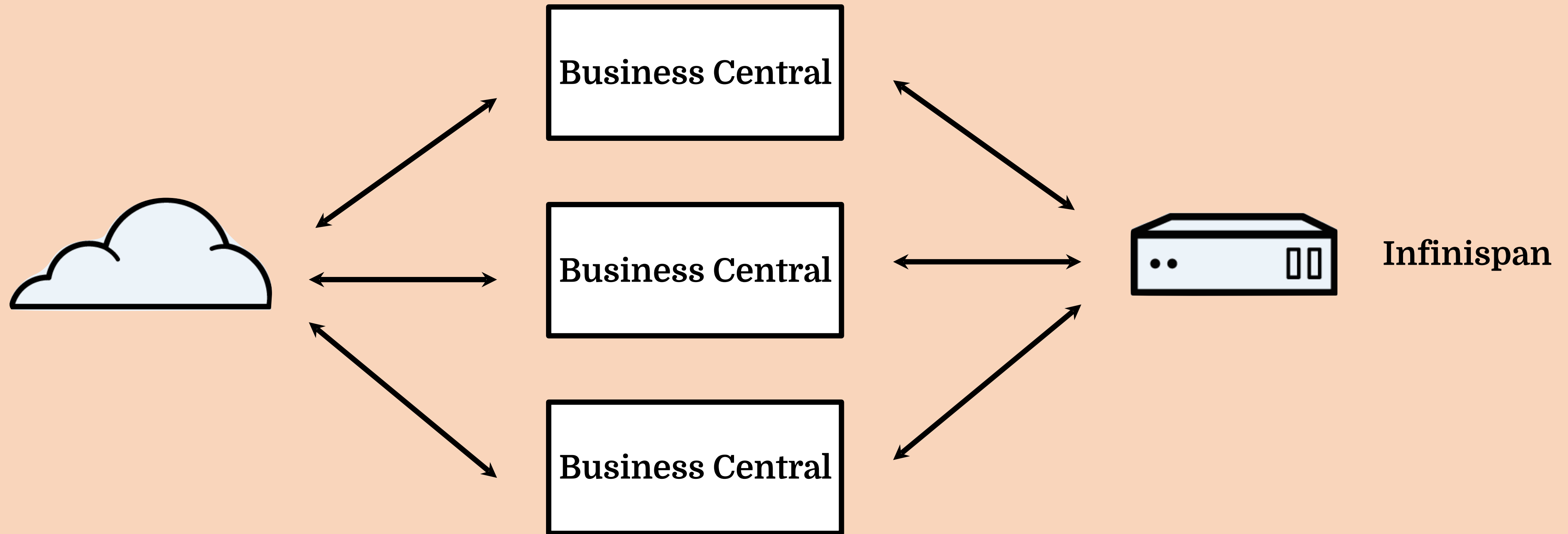
# 1. Redundant processing

DRYing the monolith



# Indexing as a service

Ctrl + Alt + M





# Lighter monolith and fine tuning

Win-win 🏆

## 2. Global lock

Do not stop the world!!!

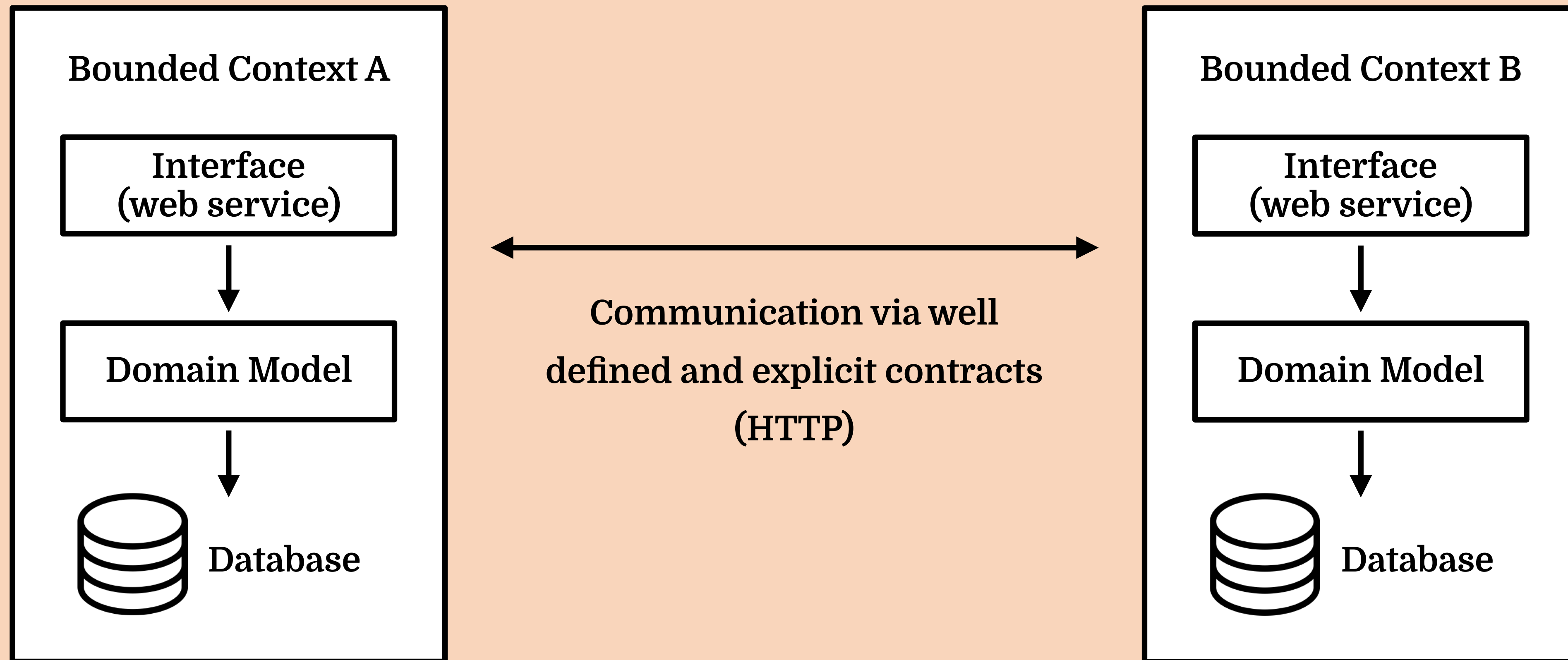
# Bounded Contexts

“A context means a specific responsibility. A bounded context means that responsibility is enforced with explicit boundaries” \*

\* <https://blog.sapiensworks.com/post/2012/04/17/DDD-The-Bounded-Context-Explained.aspx>



# Bounded Contexts



# The locked contexts

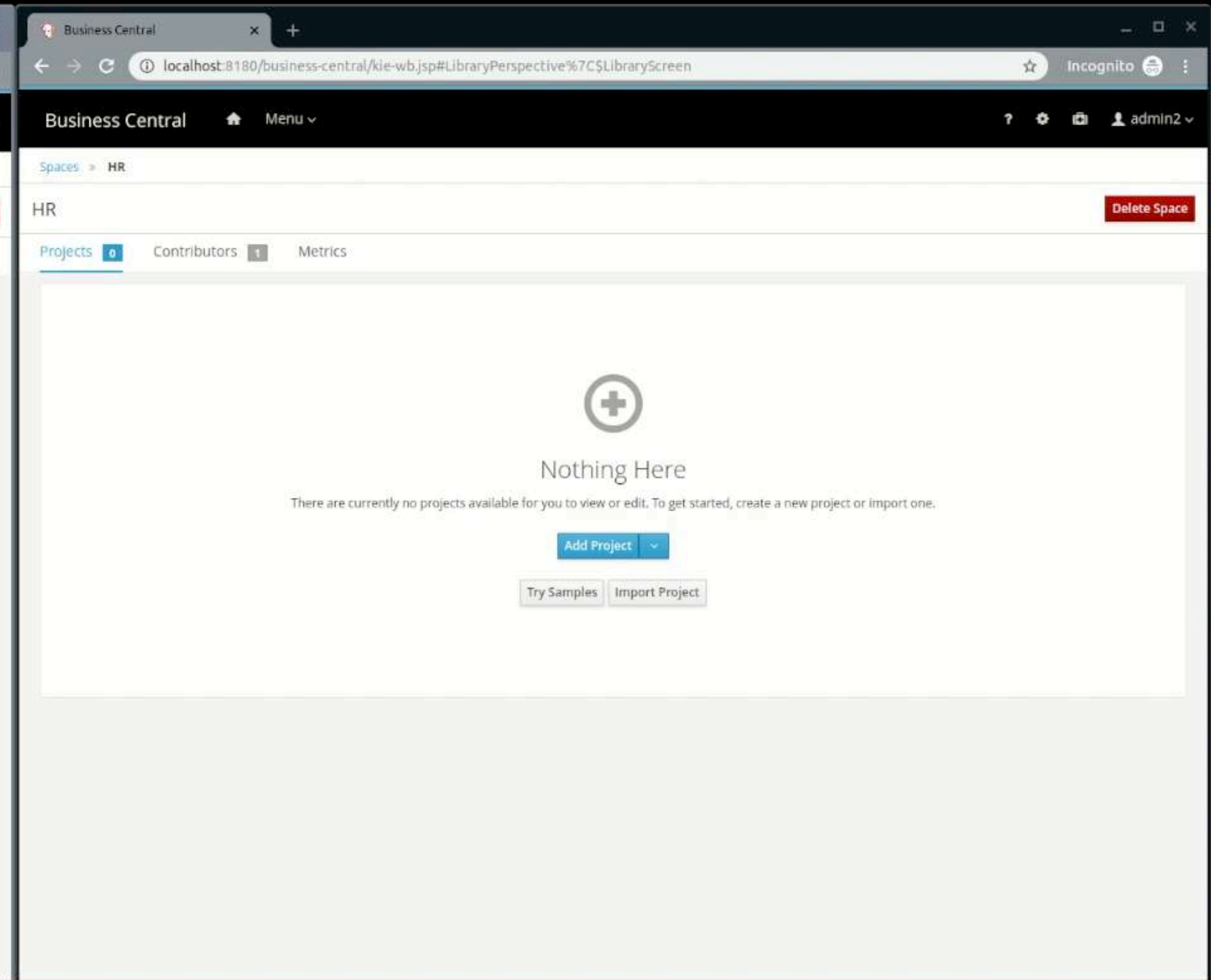
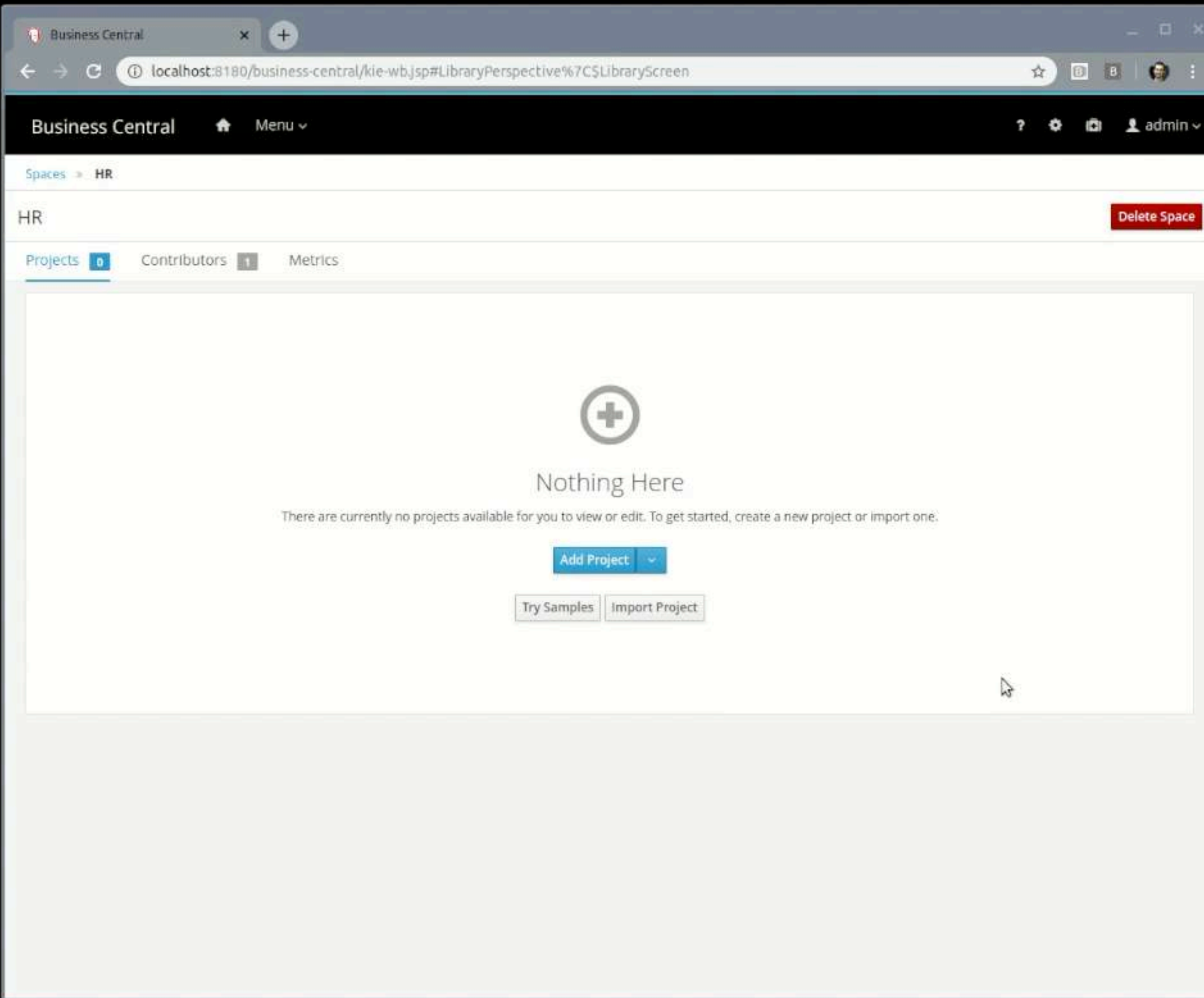
Smaller contexts, smaller locks

# 3. Keep the programming paradigm

The agnostic code base

# http://localhost:8180/...

# http://localhost:8180/...





# CDI Events 101

# CDI Events 101

```
class NewProjectEvent {  
    Project project;  
  
    NewProjectEvent(Project p) {  
        this.project = project;  
    }  
  
    Project getProject() {  
        return this.project;  
    }  
}
```

# CDI Events 101

```
class NewProjectEvent {
    Project project;

    NewProjectEvent(Project p) {
        this.project = project;
    }

    Project getProject() {
        return this.project;
    }
}
```

```
class ProjectsController {
    @Inject
    Event<NewProjectEvent> newProjectEvent;

    void createProject(Project project) {
        fileSystem.save(project);
        newProjectEvent.fire(new NewProjectEvent(project))
    }
}
```

# CDI Events 101

```
class NewProjectEvent {  
    Project project;  
  
    NewProjectEvent(Project p) {  
        this.project = project;  
    }  
  
    Project getProject() {  
        return this.project;  
    }  
}
```

```
class ProjectsController {  
    @Inject  
    Event<NewProjectEvent> newProjectEvent;  
  
    void createProject(Project project) {  
        fileSystem.save(project);  
        newProjectEvent.fire(new NewProjectEvent(project))  
    }  
}
```

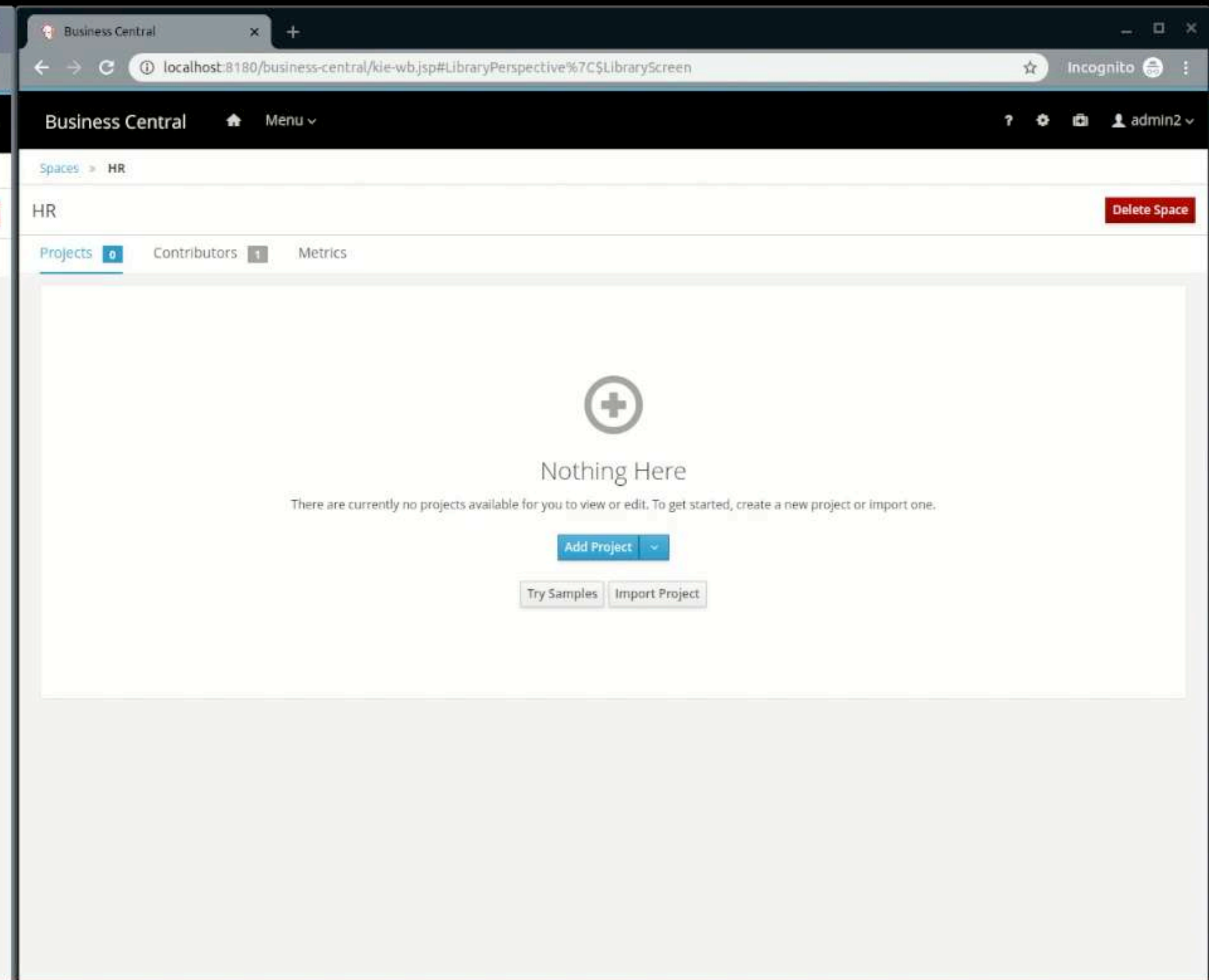
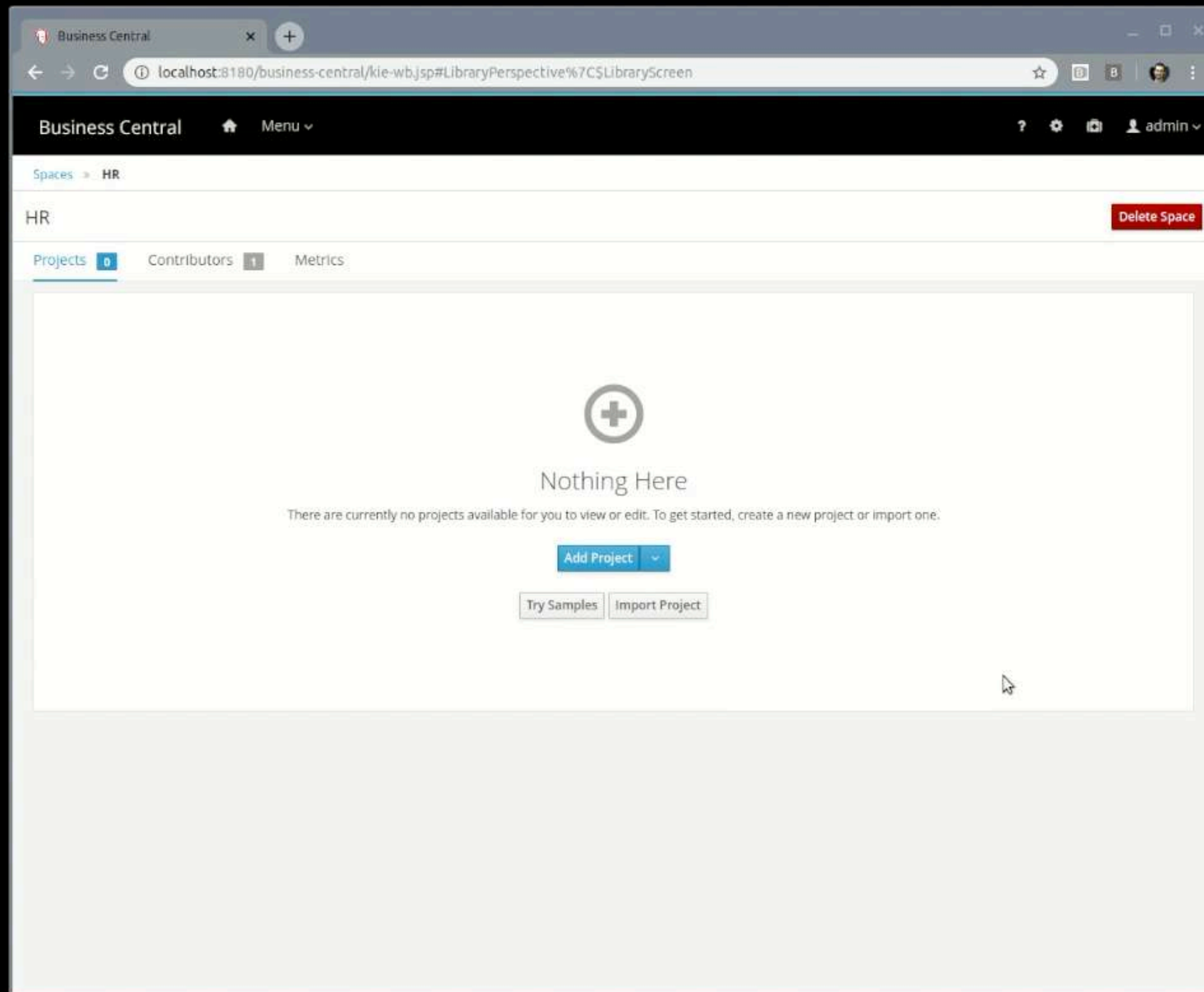
```
class ProjectScreen {  
    onNewProject(@Observes NewProjectEvent e) {  
        updateProjectList(e.getProject())  
    }  
}
```

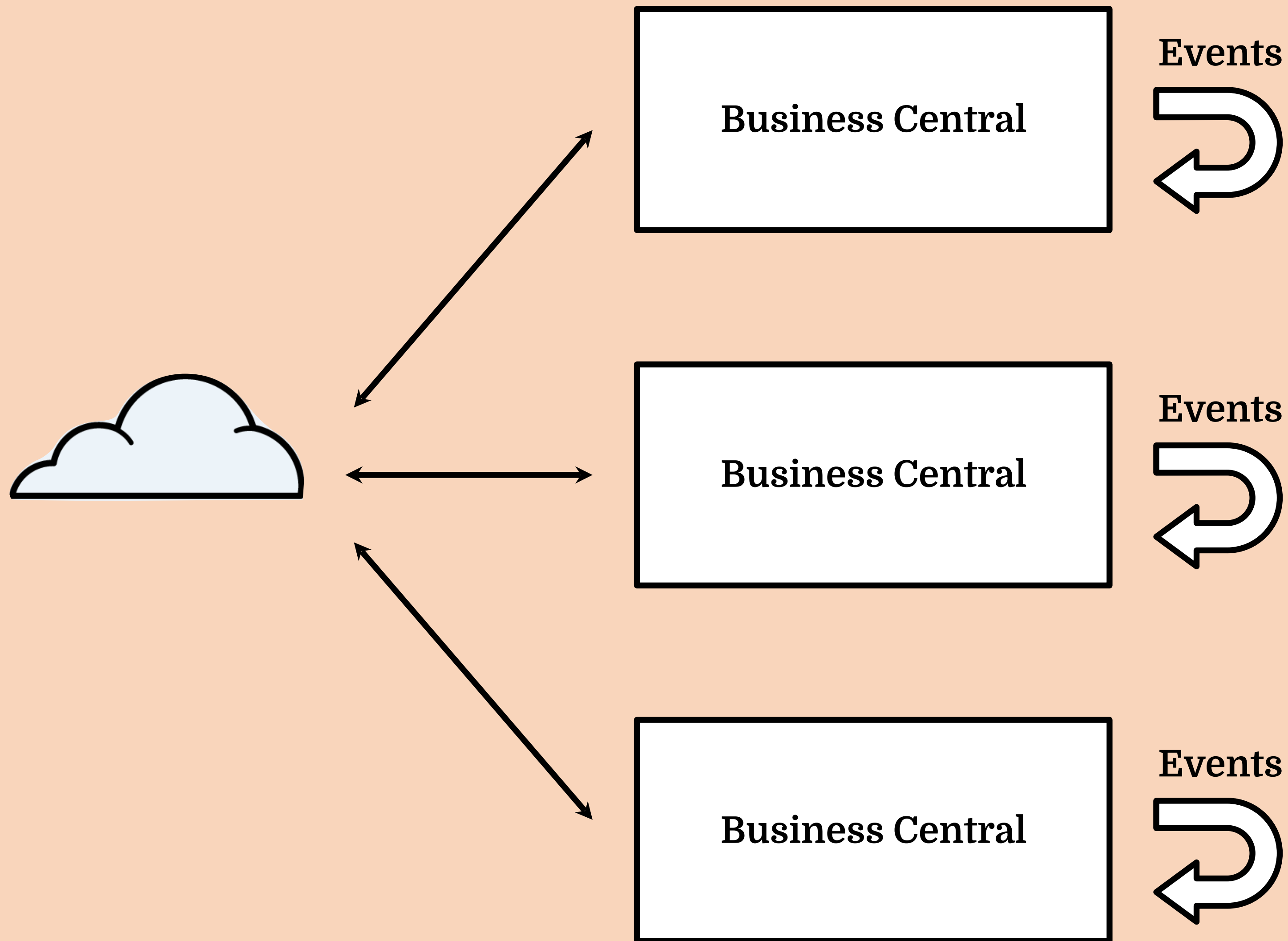




# http://localhost:8180/...

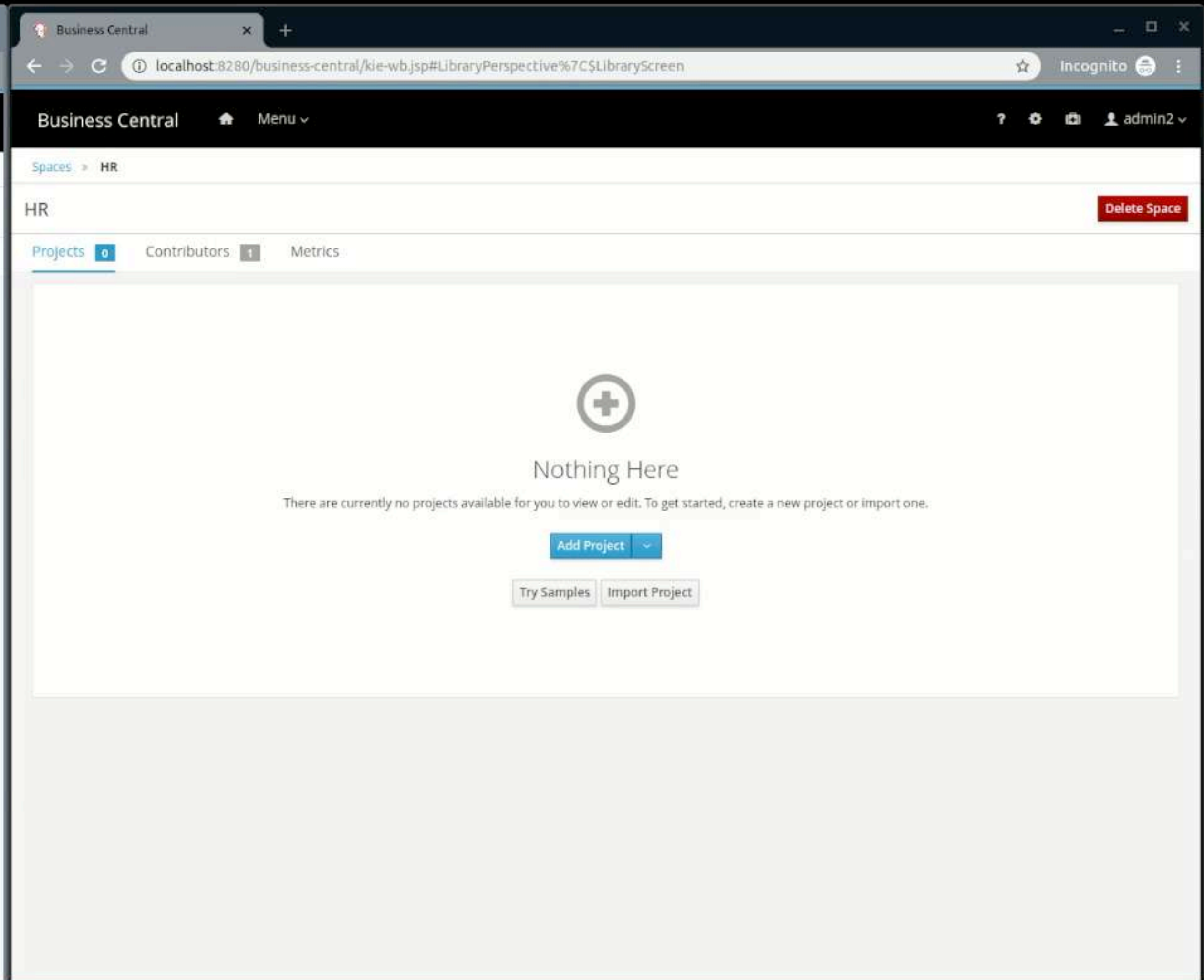
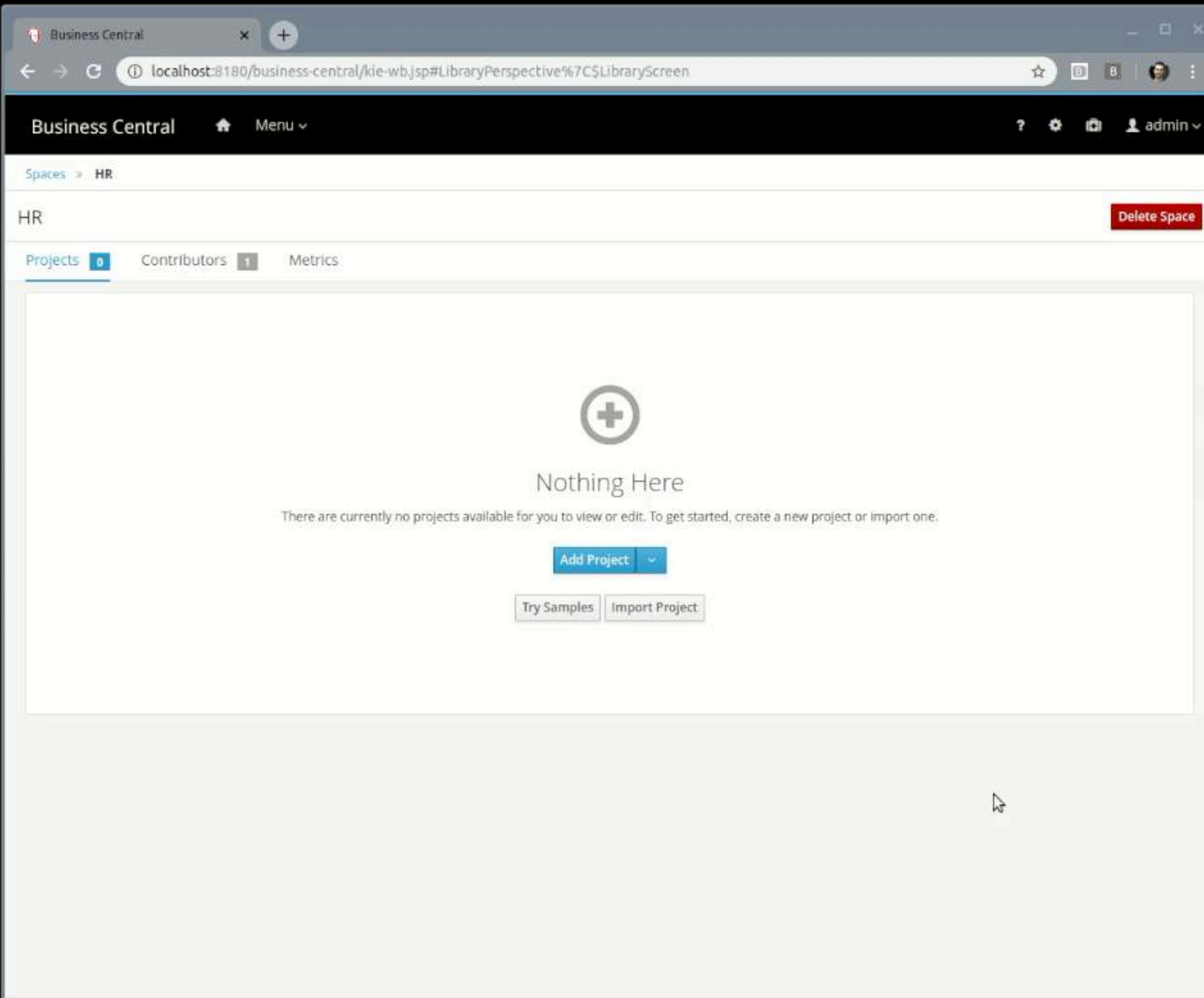
# http://localhost:8180/...





# http://localhost:8180/...

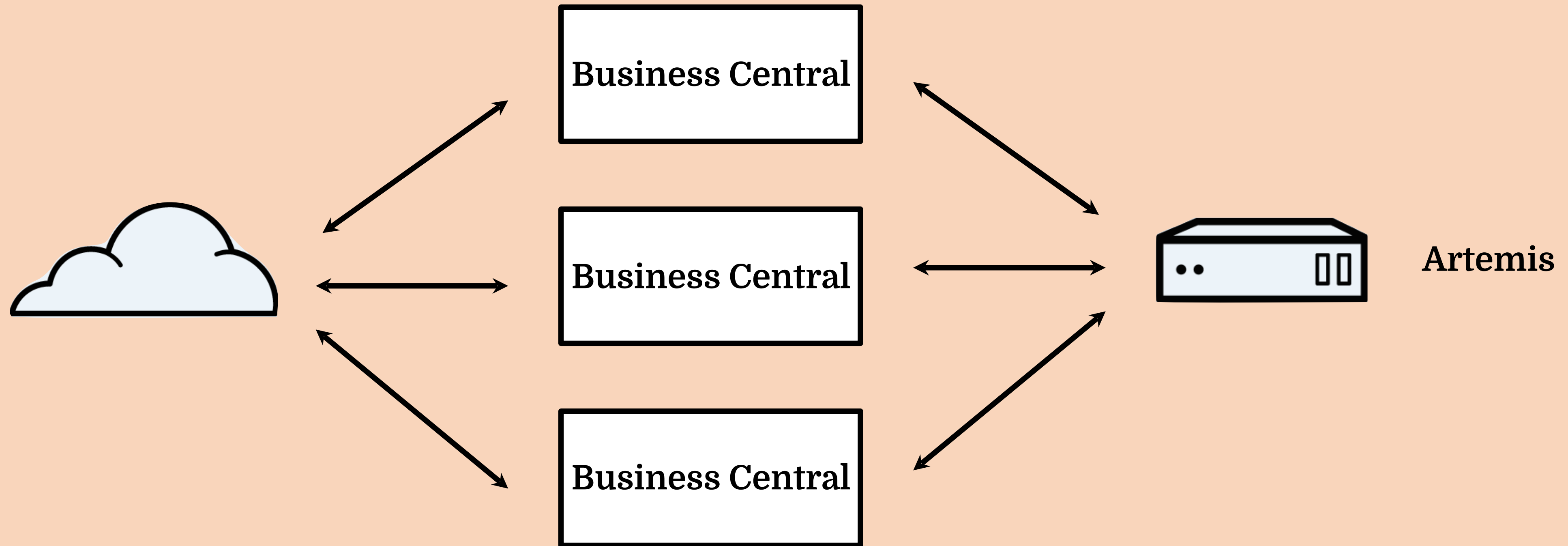
# http://localhost:8280/...





# Distributed CDI Events

- Publish–subscribe pattern



# Distributed CDI Events

- Publish–subscribe pattern
- Clustered events

@Clustered

```
class NewProjectEvent {  
    Project project;  
  
    NewProjectEvent(Project p) {  
        this.project = project;  
    }  
  
    Project getProject() {  
        return this.project;  
    }  
}
```



# Distributed CDI Events

- Publish–subscribe pattern
- Clustered events
- Event propagation

**@ApplicationScoped**

```
class ClusterEventObserver {  
    String nodeId = UUID.randomUUID().toString();  
  
    // ...  
  
    void consumeMessage(Event<Object> eventBus,  
                        EventMessage message) {  
        if (!message.getNodeId().equals(nodeId)) {  
            eventBus.fire(fromJSON(message));  
        }  
    }  
  
    void observeAllEvents(@Observes Object event,  
                          EventMetadata metaData) {  
        if (shouldObserveThisEvent(event, metaData)) {  
            broadcast(event);  
        }  
    }  
    // ...  
}
```

@ApplicationScoped

```
class ClusterEventObserver {  
    String nodeId = UUID.randomUUID().toString();  
  
    // ...  
  
    void consumeMessage(Event<Object> eventBus,  
                        EventMessage message) {  
        if (!message.getNodeId().equals(nodeId)) {  
            eventBus.fire(fromJSON(message));  
        }  
    }  
  
    void observeAllEvents(@Observes Object event,  
                          EventMetadata metaData) {  
        if (shouldObserveThisEvent(event, metaData)) {  
            broadcast(event);  
        }  
    }  
    // ...  
}
```

@ApplicationScoped

class ClusterEventObserver {

String nodeId = UUID.randomUUID().toString();

// ...

```
void consumeMessage(Event<Object> eventBus,
                    EventMessage message) {
    if (!message.getNodeId().equals(nodeId)) {
        eventBus.fire(fromJSON(message));
    }
}
```

```
void observeAllEvents(@Observes Object event,
                      EventMetadata metaData) {
    if (shouldObserveThisEvent(event, metaData)) {
        broadcast(event);
    }
}
```

// ...

}



```
@ApplicationScoped
```

```
class ClusterEventObserver {
```

```
    String nodeId = UUID.randomUUID().toString();
```

```
    // ...
```

```
    void consumeMessage(Event<Object> eventBus,  
                        EventMessage message) {  
        if (!message.getNodeId().equals(nodeId)) {  
            eventBus.fire(fromJSON(message));  
        }  
    }  
}
```

```
    void observeAllEvents(@Observes Object event,  
                          EventMetadata metaData) {  
        if (shouldObserveThisEvent(event, metaData)) {  
            broadcast(event);  
        }  
    }  
    // ...  
}
```

```
@ApplicationScoped
```

```
class ClusterEventObserver {
```

```
    String nodeId = UUID.randomUUID().toString();
```

```
    // ...
```

```
    void consumeMessage(Event<Object> eventBus,  
                        EventMessage message) {  
        if (!message.getNodeId().equals(nodeId)) {  
            eventBus.fire(fromJSON(message));  
        }  
    }  
}
```

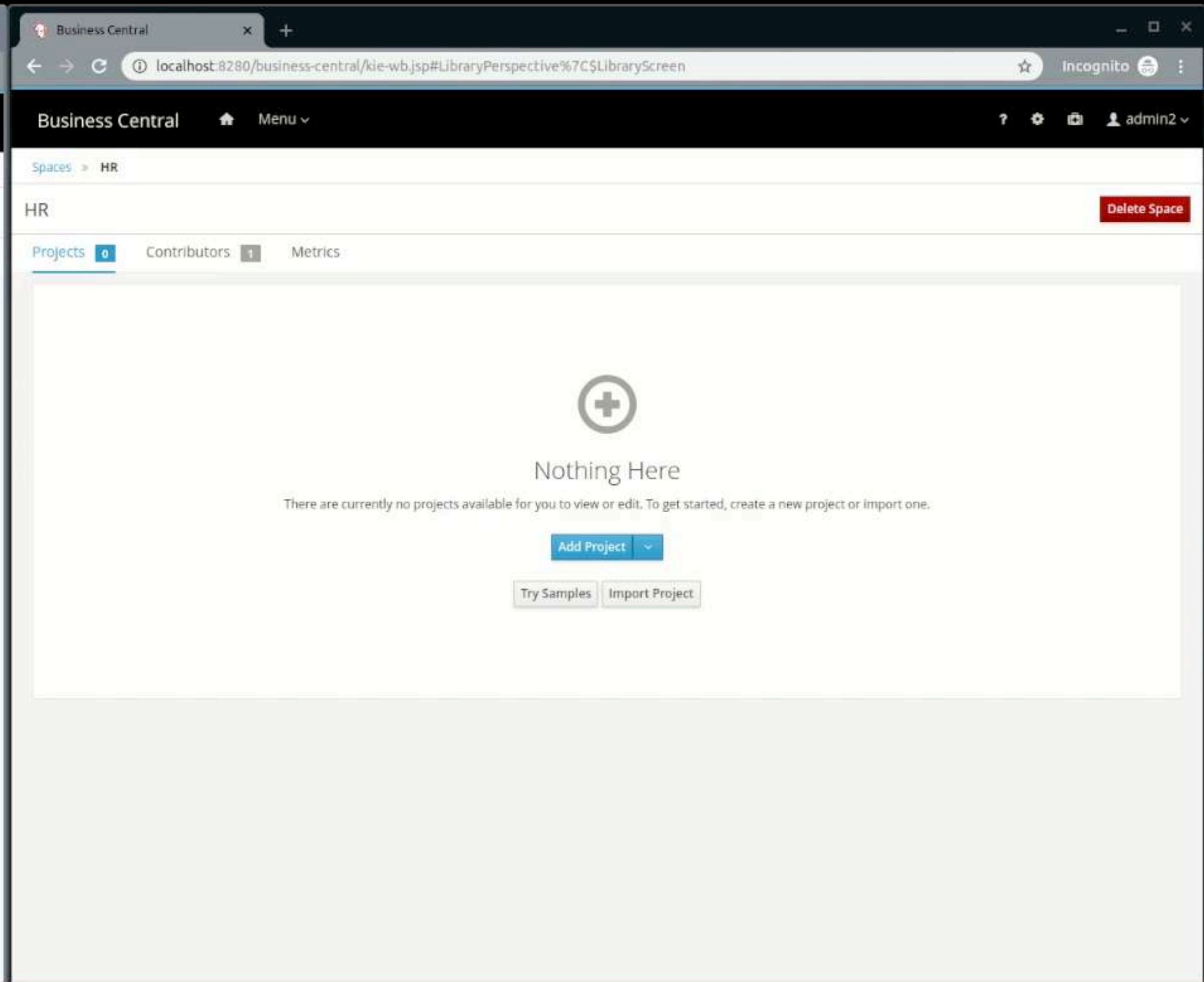
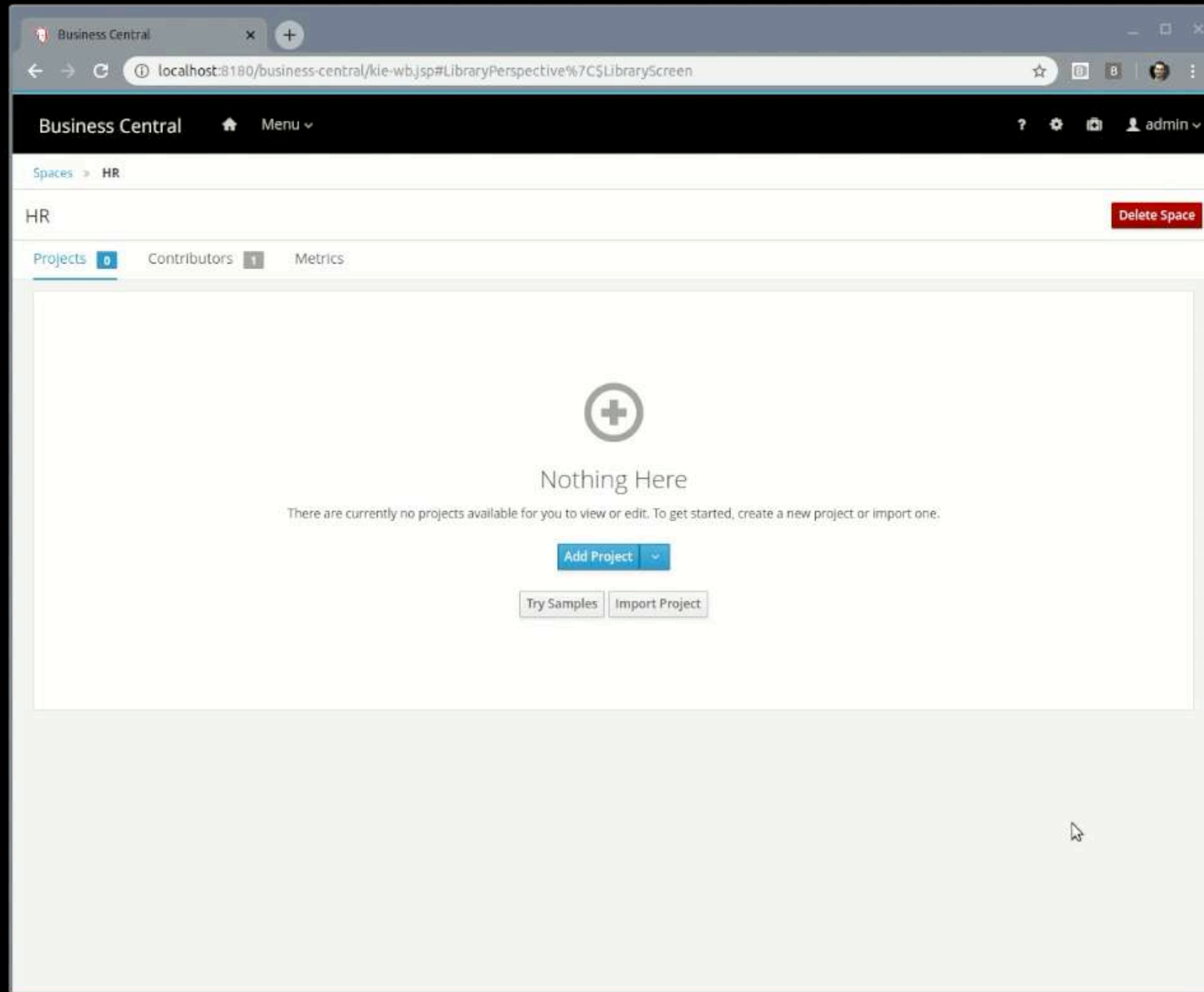
```
    void observeAllEvents(@Observes Object event,  
                          EventMetadata metaData) {  
        if (shouldObserveThisEvent(event, metaData)) {  
            broadcast(event);  
        }  
    }  
}
```

```
    // ...
```

```
}
```

# http://localhost:8180/...

# http://localhost:8280/...





The Monolith  
perspective

Cloud-native  
apps and  
Containers



The Monolith  
perspective

Cloud-native  
apps and  
Containers

Business  
Central

Monolith to  
Cloud  
challenges

# Cloud-ready

The pragmatic good enough.



# Co-Transformation to Cloud-Native Applications

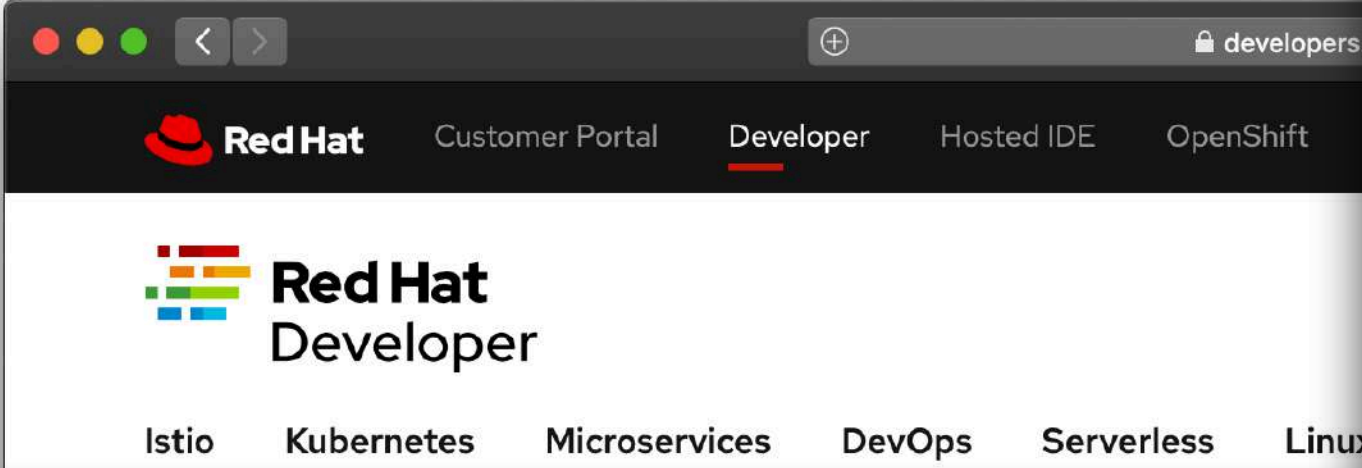
## *Development Experiences and Experimental Evaluation*

Josef Spillner<sup>1</sup>, Yessica Bogado<sup>2</sup>, Walter

<sup>1</sup>Service Prototyping Lab, Zurich University of Applied Sciences

<sup>2</sup>Information and Communication Technology Center, IT Center of Applied Sciences

josef.spillner@zhaw.ch, yessica.bogado@zhaw.ch



Review

# A Brief History of Cloud Application Architectures

Nane Kratzke 

Lübeck University of Applied Sciences, Department of Electrical Engineering and Computer Science  
23562 Lübeck, Germany; nane.kratzke@fh-luebeck.de



# About the Complexity to Transfer Cloud Applications at Runtime and how Container Platforms can Contribute?

Nane Kratzke

Lübeck University of Applied Sciences,  
Center for Communication, Systems and Applications (CoSA), Germany,  
nane.kratzke@fh-luebeck.de

**Abstract.** Cloud-native applications are often designed for only one specific cloud infrastructure or platform. The effort to port such kind of applications into a different cloud is usually a laborious one time exercise. Modern Cloud-native application architecture approaches make use of popular elastic container platforms (Apache Mesos, Kubernetes, Docker Swarm). These kind of platforms contribute to a lot of existing cloud engineering requirements. This given, it astonishes that these kind of platforms (already existing and open source available) are not considered more consequently for multi-cloud solutions. These platforms provide inherent multi-cloud support but this is often overlooked. This paper presents a software prototype and shows how Kubernetes and Docker Swarm clusters could be successfully transferred at runtime across public cloud infrastructures of Google (Google Compute Engine), Microsoft (Azure) and Amazon (EC2) and further cloud infrastructures like OpenStack. Additionally, software engineering lessons learned are derived and some astonishing performance data of the mentioned cloud infrastructures is presented that could be used for further optimizations of IaaS transfers of Cloud-native applications.

**Keywords:** cloud-native application, multi-cloud, elastic platform, container, portability, flexibility, MAPE-AWS, GCE, Azure, OpenStack



# Co-Transformation to Cloud-Native Applications

## *Development Experiences and Experimental Evaluation*

Josef Spillner<sup>1</sup>, Yessica Bogado<sup>2</sup>, Walter

<sup>1</sup>*Service Prototyping Lab, Zurich University of Applied Sciences*

<sup>2</sup>*Information and Communication Technology Center, IT Center*

josef.spillner@zhaw.ch, yessica.bogado@zhaw.ch

### About the Complexity to Transfer Cloud Applications at Runtime and how Container Platforms can Contribute?

Nane Kratzke

Center for Communication, Systems and Applications (CoSA), Germany,

nane.kratzke@fh-luebeck.de

“Cloud-ready apps are applications initially developed to run on static environments, but modified to take advantage of a limited number of cloud features”

### Brief History of Cloud Application Architectures

Nane Kratzke

Lübeck University of Applied Sciences, Department of Electrical Engineering and Computer Science  
23562 Lübeck, Germany; nane.kratzke@fh-luebeck.de

**Abstract.** Cloud-native applications are often designed for only one specific cloud infrastructure or platform. The effort to port such kind of applications into a different cloud is usually a laborious one time exercise. Modern Cloud-native application architecture approaches make use of popular elastic container platforms (Apache Mesos, Kubernetes, Docker Swarm). These kind of platforms contribute to a lot of existing cloud engineering requirements. This given, it astonishes that these kind of platforms (already existing and open source available) are not considered more consequently for multi-cloud solutions. These platforms provide inherent multi-cloud support but this is often overlooked. This paper presents a software prototype and shows how Kubernetes and Docker Swarm clusters could be successfully transferred at runtime across public cloud infrastructures of Google (Google Compute Engine), Microsoft (Azure) and Amazon (EC2) and further cloud infrastructures like OpenStack. Additionally, software engineering lessons learned are derived and some astonishing performance data of the mentioned cloud infrastructures is presented that could be used for further optimizations of IaaS transfers of Cloud-native applications.

**Keywords:** cloud-native application, multi-cloud, elastic platform, containerization, Kubernetes, Docker Swarm, AWS, GCE, Azure, OpenStack

entando

WHAT WE DO TECH

🔗 Back to overview

### What is the Difference between Cloud-Ready

Tuesday, April 17, 2018 | [ENGINEERING\\_TEAM](#)

📖 Engineering



# Cloud-ready

1. Each instance is agnostic of environment
2. Redundant tasks as a service
3. Horizontal scalability





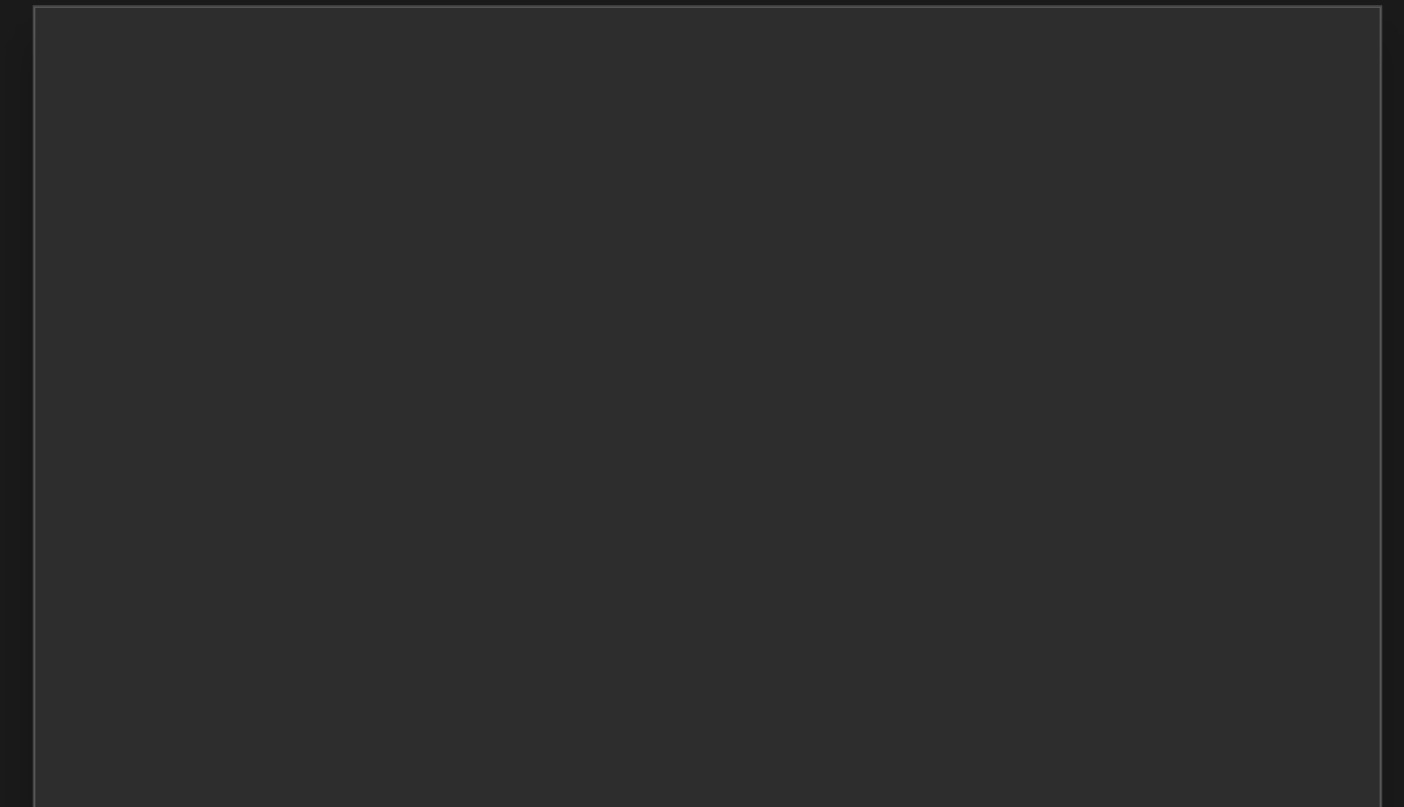
The Monolith  
perspective

Cloud-native  
apps and  
Containers

Business  
Central

Monolith to  
Cloud  
challenges

Cloud-ready  
enough



The Monolith  
perspective

Cloud-native  
apps and  
Containers

Business  
Central

Monolith to  
Cloud  
challenges

Cloud-ready  
enough

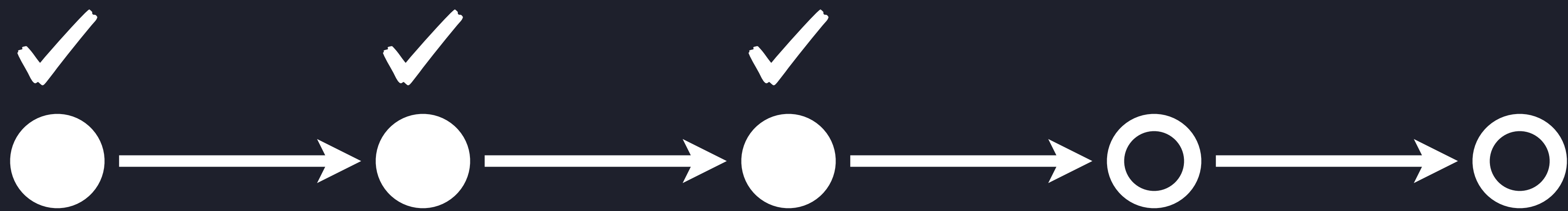
Look at your  
app

# Look at your app

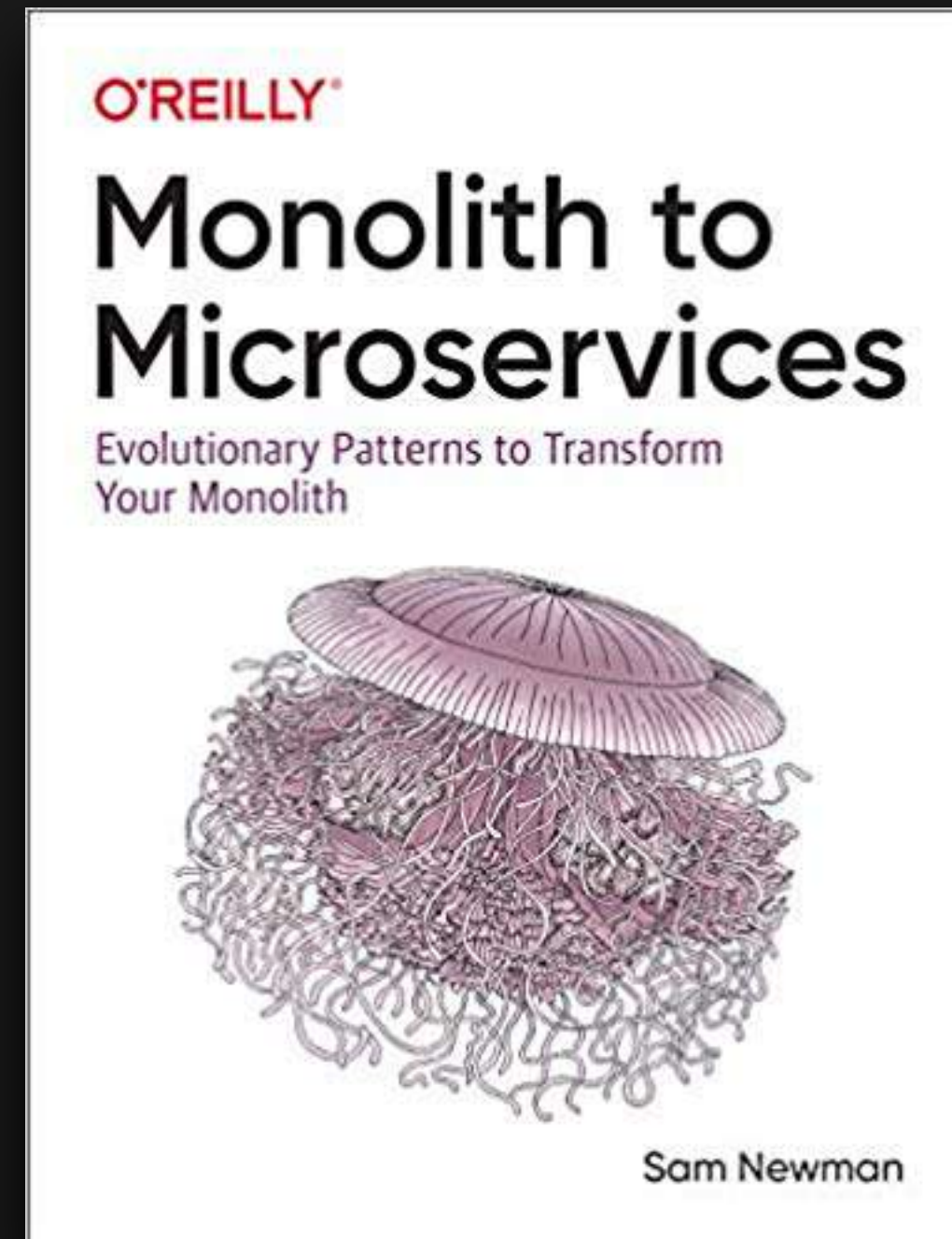
It's not that ugly, neither that good — probably.









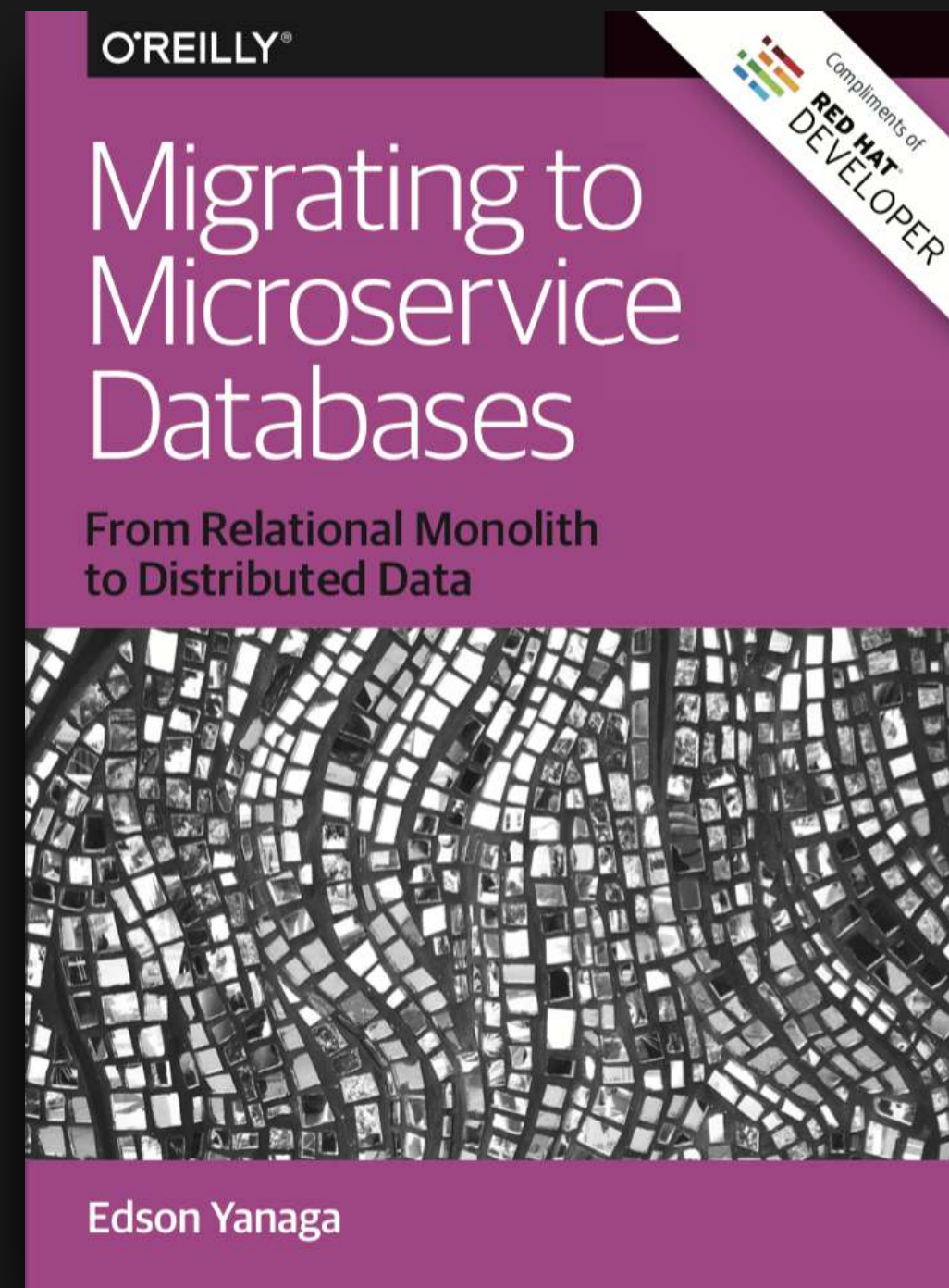


# Monolith to Microservices

by Sam Newman

Release Date: November 2019

<http://shop.oreilly.com/product/0636920233169.do>

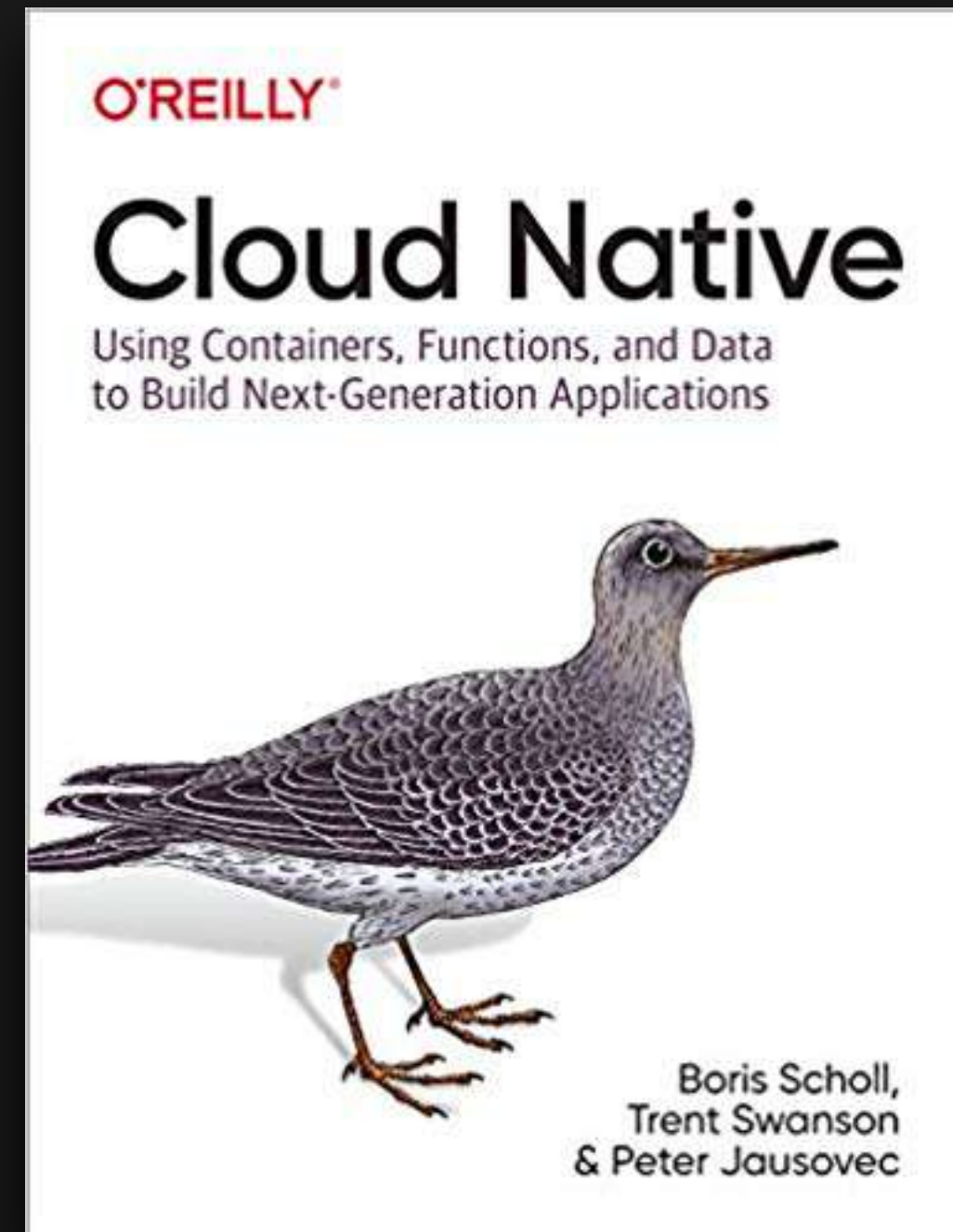


# Migrating to Microservice Databases

by Edson Yanaga

Release Date: April 2017

<https://developers.redhat.com/books/migrating-mi...>



# Cloud Native

by Peter Jausovec, Trent Swanson, Boris Scholl

Release Date: August 2019

<http://shop.oreilly.com/product/0636920261704.do>





THE  
DEVELOPER'S  
CONFERENCE

# Thank you! 🤗

@karreiro - [karreiro.com/talks](https://karreiro.com/talks)

@paulovmr