

# Application Modernization Sample

## From Java EE in 2010 to Cloud-Native in 2021

Niklas Heidloff

Developer Advocate, IBM

@nheidloff

# Why ?

1. More Agility
2. Better User Experiences
3. Reduced Costs



Watch on YouTube

# Strategies

# Strategies

## 1. Retire

# Strategies

1. Retire
2. Don't touch

# Strategies

1. Retire
2. Don't touch
3. Lift and shift

# Strategies

1. Retire
2. Don't touch
3. Lift and shift
4. Containerize
5. Refactor

# Sample Application

**Electronic and Movie Depot**

Shop Cart Order History Account

Entertainment ▾ Electronics

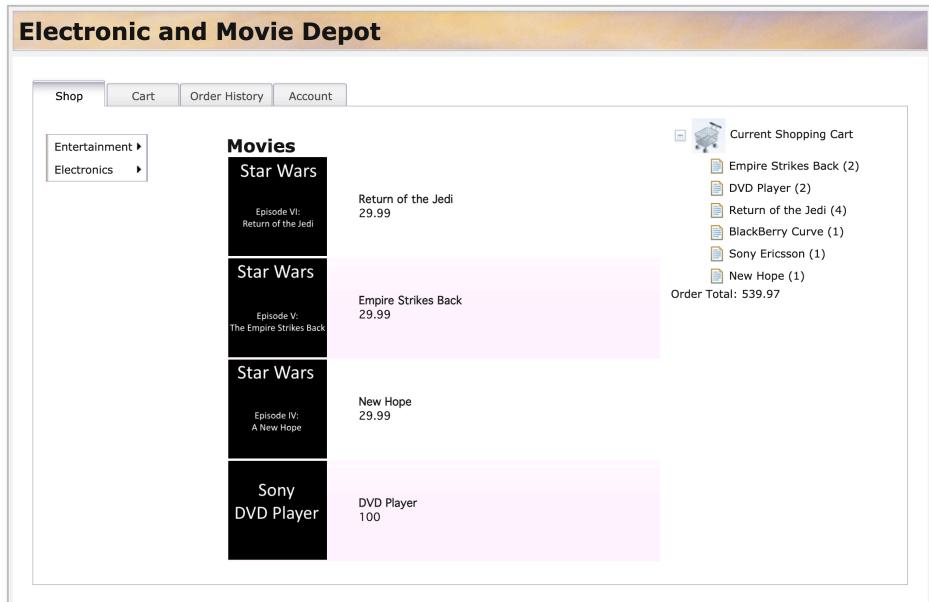
**Movies**

- Star Wars  
Episode VI:  
Return of the Jedi  
29.99
- Star Wars  
Episode V:  
The Empire Strikes Back  
29.99
- Star Wars  
Episode IV:  
A New Hope  
29.99
- Sony  
DVD Player  
100

**Current Shopping Cart**

- Empire Strikes Back (2)
- DVD Player (2)
- Return of the Jedi (4)
- BlackBerry Curve (1)
- Sony Ericsson (1)
- New Hope (1)

Order Total: 539.97



Electronic and Movie Depot Catalog - Movies

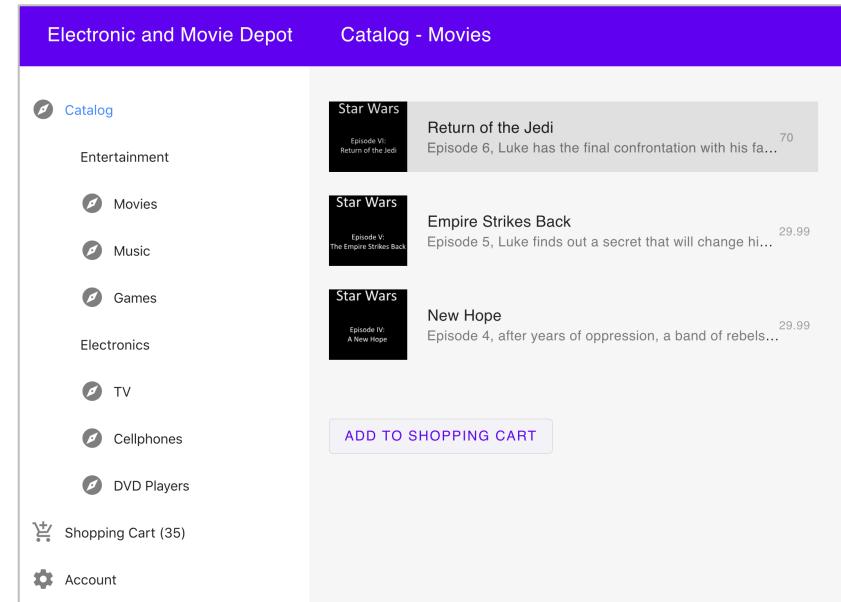
Catalog Entertainment Movies Music Games Electronics TV Cellphones DVD Players Shopping Cart (35) Account

**Star Wars**  
Episode VI:  
Return of the Jedi  
29.99

**Star Wars**  
Episode V:  
The Empire Strikes Back  
29.99

**Star Wars**  
Episode IV:  
A New Hope  
29.99

**Add To Shopping Cart**



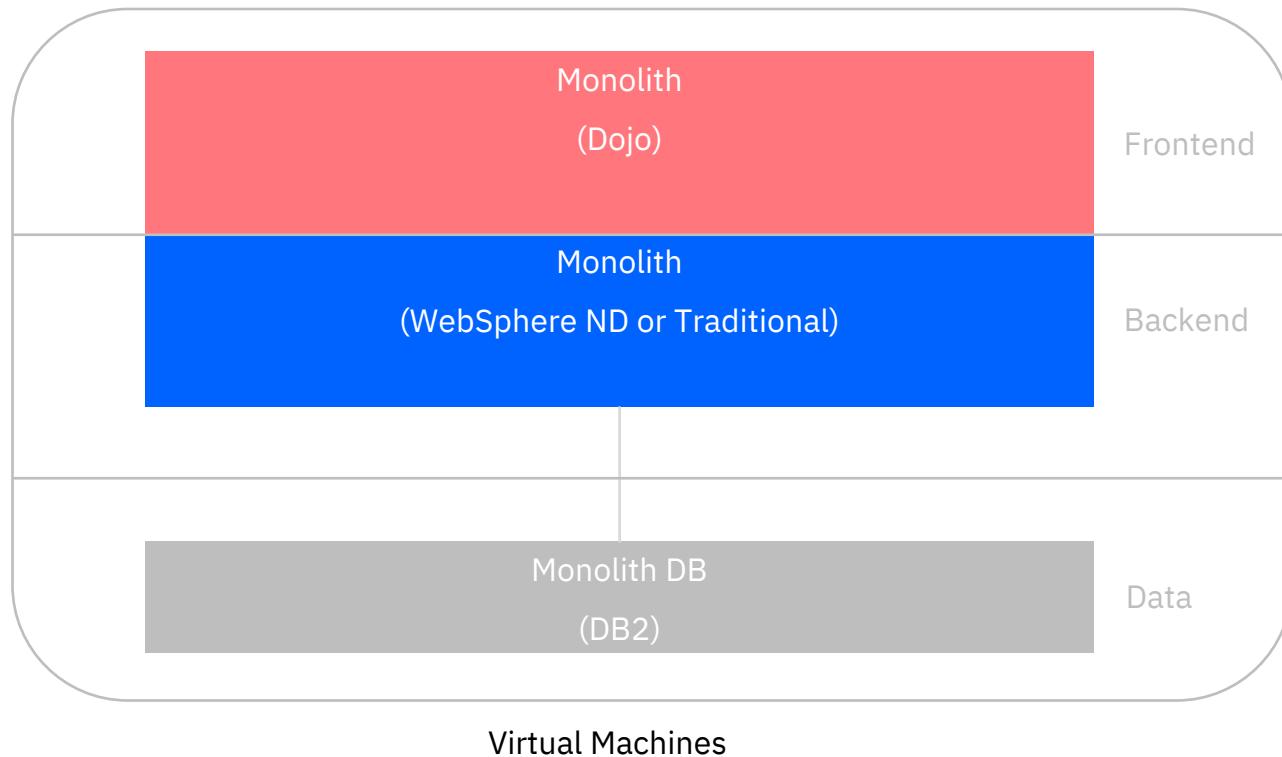
# Application Modernization is a Journey.

## Take Baby Steps!

# Steps

1. WebSphere Traditional 8.5.5 in VM
2. WebSphere Traditional 9 in container
3. WebSphere Liberty monolith
4. Separated frontend
5. Open Liberty monolith
6. Strangled Service and Open Liberty monolith
7. Strangled Service and Quarkus monolith
8. Micro Frontends

# Starting Point: Monolith running in VMs



# IBM Cloud Transformation Advisor



IBM Garage Methodology



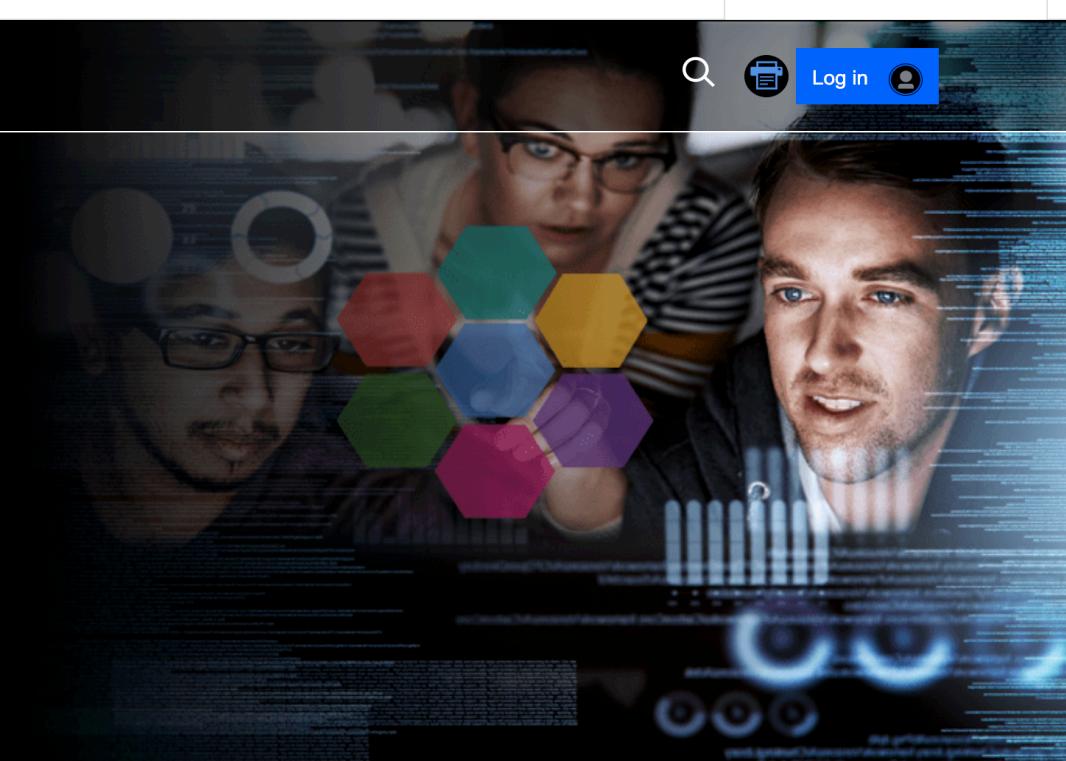
Log in



## IBM Cloud Transformation Advisor

You can use IBM Cloud Transformation Advisor to analyze your on-premises workloads for modernization. Determine the complexity of your applications, estimate the development cost to move to the cloud, and get recommendations for the best target environment. Run Transformation Advisor Local on your desktop until you have a Red Hat OpenShift cluster deployed.

[Try Transformation Advisor Local](#)



# IBM WebSphere Application Migration Toolkit



My Marketplace Add Content More ▾

Home / Marketplace / Tools (1292) / IBM WebSphere Application Server Migration Toolkit

**MARKETS**

**SEARCH**

Search Advanced Search

Search



★ 17

3



## IBM WebSphere Application Server Migration Toolkit



Details

Screenshots

Metrics

Errors

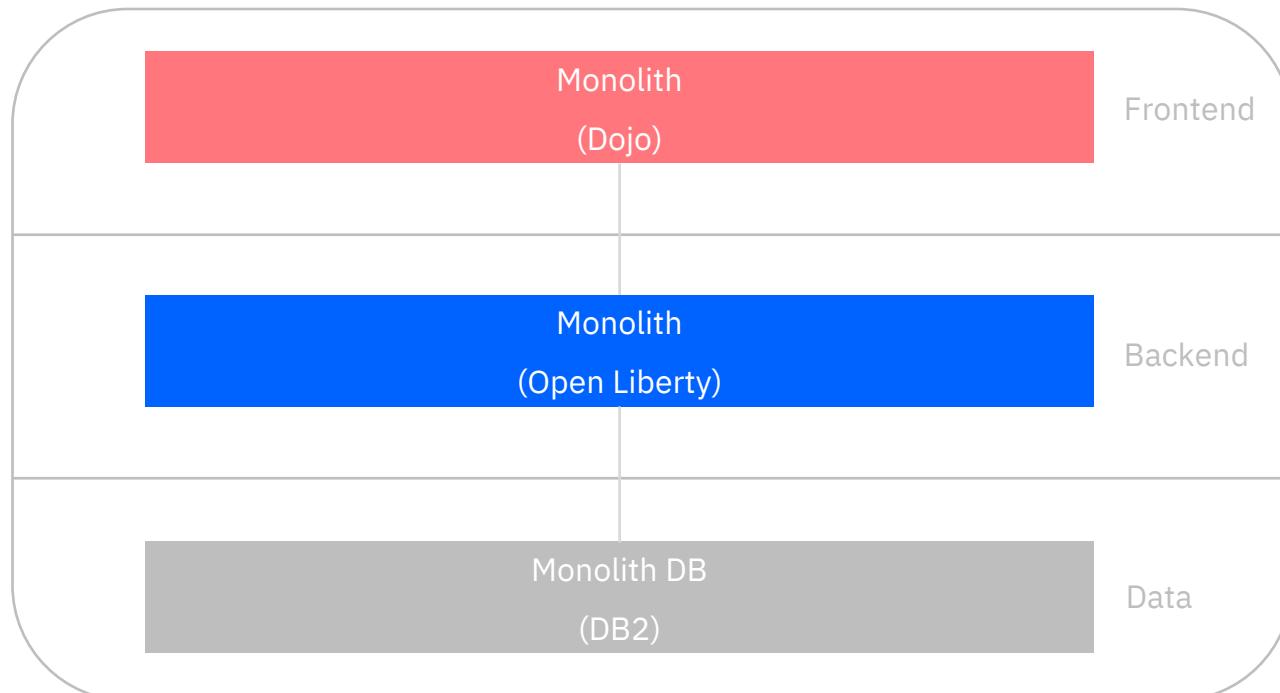
External Install Button

The IBM® WebSphere® Application Server Migration Toolkit is a suite of tools and knowledge collections that enables your organization to quickly and cost-effectively migrate to WAS Liberty and WebSphere Application Server V8, V8.5.5 or V9, whether from a previous version of WebSphere Application Server or competitive application servers including Apache Tomcat Server, JBoss Application Server, Oracle Application Server, and Oracle® WebLogic Server. It features cloud migration rules for running your application on Liberty for Java on IBM Bluemix and other third-party PaaS environments such as Cloud Foundry. This Eclipse Marketplace solution combines all our Eclipse-based migration tools in one easy installation. The WebSphere Version to Version Application Migration Tool enables organizations deploying to Liberty or traditional WebSphere Application Server to more easily migrate applications from WebSphere Application Server V5.1 and later. Upgrading to a more recent version of WebSphere Application Server is of course less time-consuming than migrating from another application server, given that IBM has made significant investments in upward compatibility, configuration, and management process upgrades, as well as API preservation and consistency between versions. However, in some cases, applications must be changed in order to support or exploit new levels of industry standard specifications delivered with new versions of WebSphere Application Server. Go to the [WebSphere migration tools download page](#) to download this tool and the command line binary scanner.

### MORE LIKE THIS

- IBM WebSphere Application Server Migration Toolkit- WAS Liberty
- IBM WebSphere

# Step 5: Open Liberty Monolith



OpenShift / Kubernetes

# Huge Advantage of Jakarta Enterprise Edition:

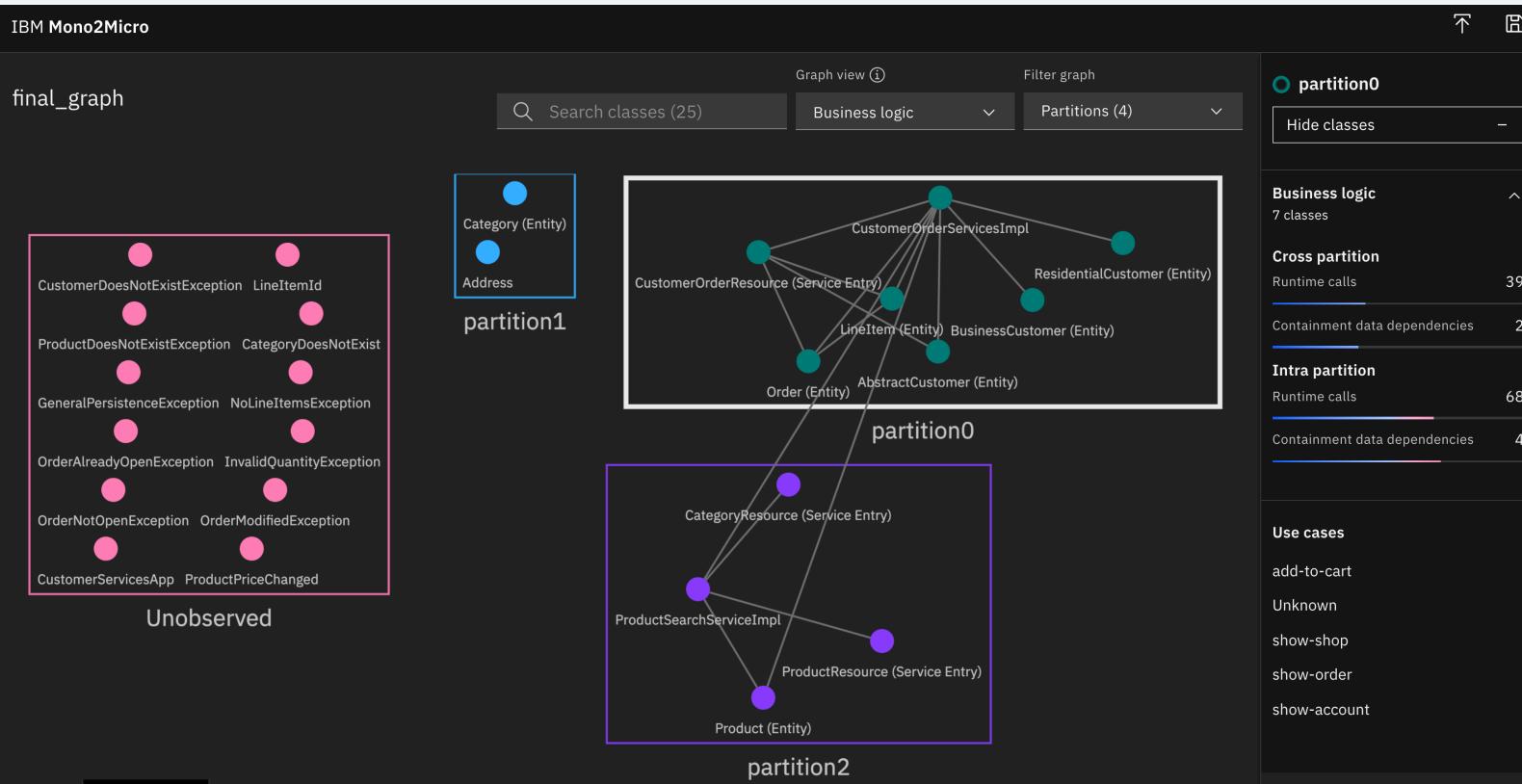
## Upwards Compatibility!

# Monoliths are not evil !

Use the Strangler Pattern  
to modernize Applications  
in Stages !

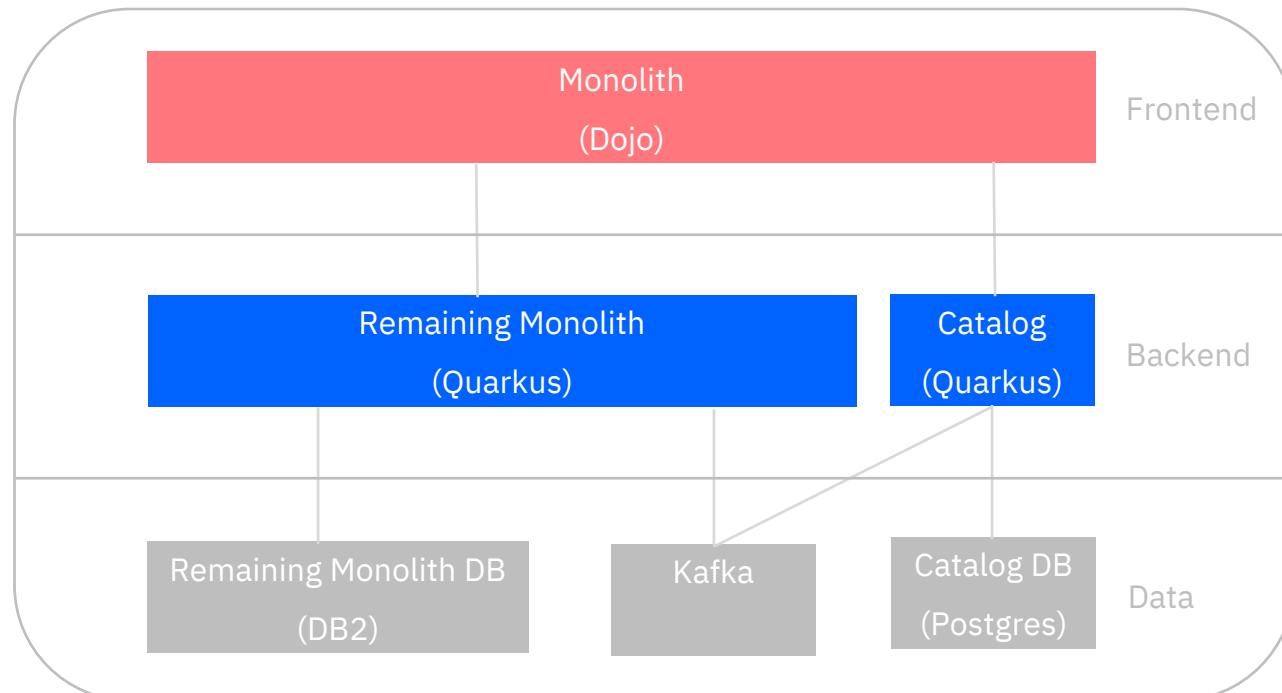
# Don't build distributed Monoliths !

# IBM Mono2Micro



Use event driven  
architectures to de-couple  
microservices !

# Step 7: Strangled Catalog Service and Quarkus Monolith

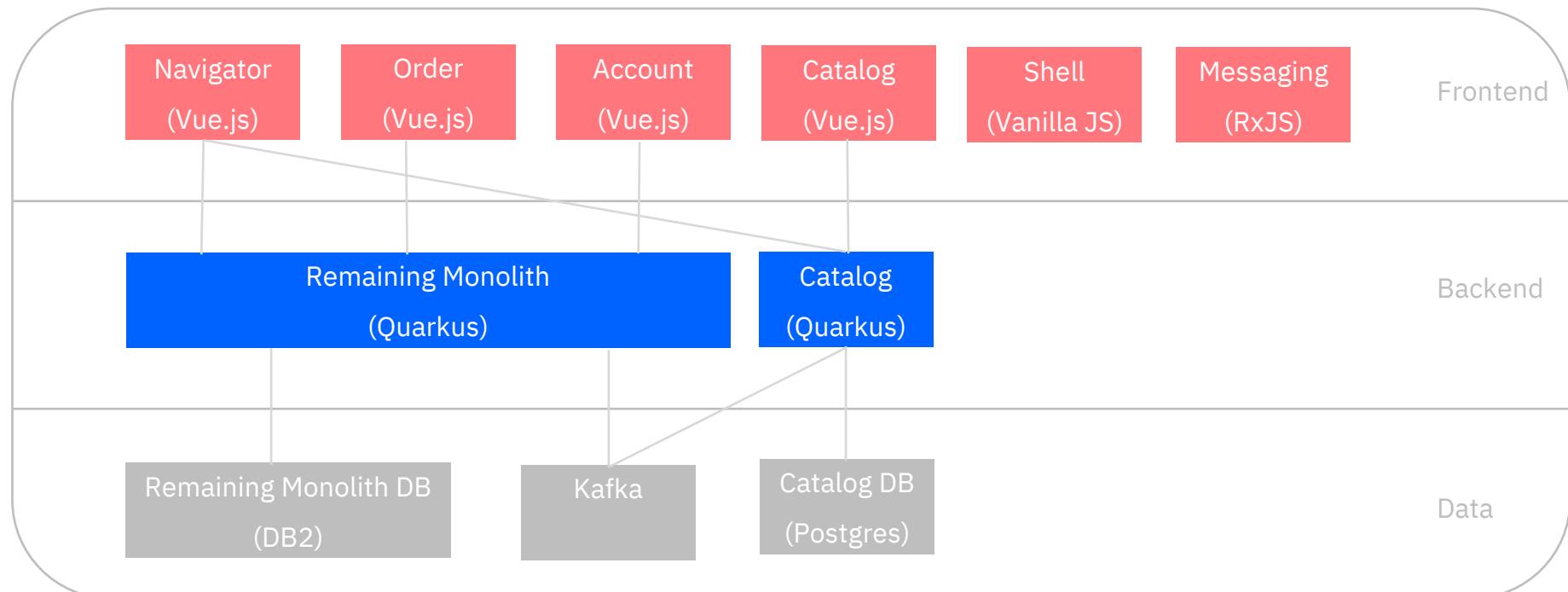


OpenJ9 uses less memory.

Reactive technologies  
require less resources.

# Don't build microservices with monolithic frontends

# Step 8: Micro Frontends



# Step 8: Micro Frontends

Electronic and Movie Depot

The application interface is divided into several sections:

- Left Sidebar:** Contains icons and labels for Catalog, Entertainment, Movies, Music, Games, Electronics, TV, Cellphones, Navigator (Vue.js), Shell (Vanilla JS), and Account.
- Main Content Area:** Titled "Catalog - Movies", it displays a list of Star Wars movies with their titles, episode numbers, descriptions, and prices.
  - Star Wars: Episode VI: Return of the Jedi - 29.99
  - Star Wars: Episode V: The Empire Strikes Back - 29.99
  - Star Wars: Episode IV: A New Hope - 29.99
- Bottom Navigation:** Includes tabs for Catalog (Vue.js), Order (Vue.js), and Account (Vue.js).

Shell  
(Vanilla JS)

Messaging  
(RxJS)

Try out the example  
yourself !

# Documentation

The screenshot shows a web browser window with the URL [github.com/IBM/application-modernization-javaee-quarkus](https://github.com/IBM/application-modernization-javaee-quarkus). The page title is "Documentation". Below the title, there is a paragraph of text followed by a bulleted list of links.

I've written a series of blogs about this project:

- [Application Modernization and Rabbits](#)
- [Project Overview Slides](#)
- [10 Reasons why Enterprises should modernize Applications](#)
- [Improving operational Efficiency through Application Modernization](#)
- [Modernizing Java EE Applications with WebSphere Liberty](#)
- [Step-by-Step Instructions how to use Transformation Advisor](#)
- [Modernizing Applications with new User Experiences](#)
- [Moving from WebSphere Liberty to Open Source with Open Liberty](#)
- [Increasing Productivity for legacy Liberty Applications](#)
- [Don't build distributed Monoliths!](#)
- [Strangler Pattern Example](#)
- [Step-by-Step Instructions for Mono2Micro](#)
- [Event driven Architectures for loosely coupled Microservices](#)
- [Using Quarkus for building reactive Applications](#)
- [Using Micro Frontends in Microservices based Architectures](#)
- [Developing Micro Frontends with Single-Spa](#)
- [Developing loosely coupled Micro Frontends via RxJS](#)
- [Workshop: Modernizing IBM WebSphere Applications](#)
- [Running Liberty Applications with Db2 locally](#)
- [Running legacy Java Applications locally](#)
- [Application Modernization Resources on IBM Developer](#)

# Get Started

← → ⌂ [github.com/IBM/application-modernization-javaee-quarkus](https://github.com/IBM/application-modernization-javaee-quarkus)

## TL;DR

If you want to run the modernized application locally, you can invoke the following commands. All you need is a local Docker installation and the git CLI.

Notes:

- Docker requires 12 GB memory and 8 CPUs
- It takes roughly 15 - 20 minutes to start everything

```
$ git clone https://github.com/nheidloff/application-modernization-javaee-quarkus.git && cd application-modernization-javaee-quarkus
$ ROOT_FOLDER=$(pwd)
$ sh ${ROOT_FOLDER}/scripts-docker/build-and-run.sh
```

The 'build-and-run.sh' script will launch the following containers.

CONTAINER ID	IMAGE	STATUS	NAMES\PORTS
c3312ecd5dd3	storefront-mf-navigator	Up 2 minutes	storefront-mf-navigator@0.0.0:8501->80/tcp
053446cf8b69	storefront-mf-catalog	Up 2 minutes	storefront-mf-catalog@0.0.0:8502->80/tcp
9149015f94a9	storefront-mf-account	Up 2 minutes	storefront-mf-account@0.0.0:8502->80/tcp
d71a15f941a3	storefront-mf-messaging	Up 2 minutes	storefront-mf-messaging@0.0.0:9001->80/tcp
7bae069e9ec0	storefront-mf-order	Up 2 minutes	storefront-mf-order@0.0.0:8504->80/tcp
88322643c884	storefront-mf-shell	Up 2 minutes	storefront-mf-shell@0.0.0:8800->80/tcp
8819989980d7	storefront-frontend	Up 41 minutes	storefront-frontend@0.0.0:9081->9080/tcp, 0.0.0.0:9444->9443/tcp
e8ddbb6d1bc1	proxy-nginx	Up 41 minutes	proxy-nginx@0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp
977fe986e67	storefront-backend-quarkus	Up 41 minutes	storefront-backend-quarkus@0.0.0:9082->8080/tcp, 0.0.0.0:9445->9443/tcp
e45993f1b7e3	storefront-catalog-reactive	Up 45 minutes	storefront-catalog-reactive@0.0.0:9778/tcp, 9779/tcp, 0.0.0.0:9083->8082/tcp
f8693fb7b6	storefront-db2	Up 45 minutes	storefront-db2@0.0.0:5000->5000/tcp, 22/tcp, 0.0.0.0:50000->50000/tcp, 55000/tcp, 60006-60007/tcp, 0.0.0.0:60000->60000/tcp
f5b2a8bf4a97	strimzi/kafka:0.19.0-kafka-2.5.0	Up About an hour	kafka@0.0.0:9092->9092/tcp
37244d91652d	strimzi/kafka:0.19.0-kafka-2.5.0	Up About an hour	zookeeper@0.0.0:2181->2181/tcp
6166623645f1	postgres	Up About an hour	postgres@0.0.0:5432->5432/tcp

# More to come !

@nheidloff