UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA - 2018.1

Technical Report - IF1007 Microsserviços
PORQUIN

Bruno Filho (brgccf)
Matheus de La Rocque (mlrbc)
Vinícius de Moraes (vrm)

Sumário

Sumário **2**

Introdução **3**

Arquitetura **4**

Projeto **5**

Execução **10**

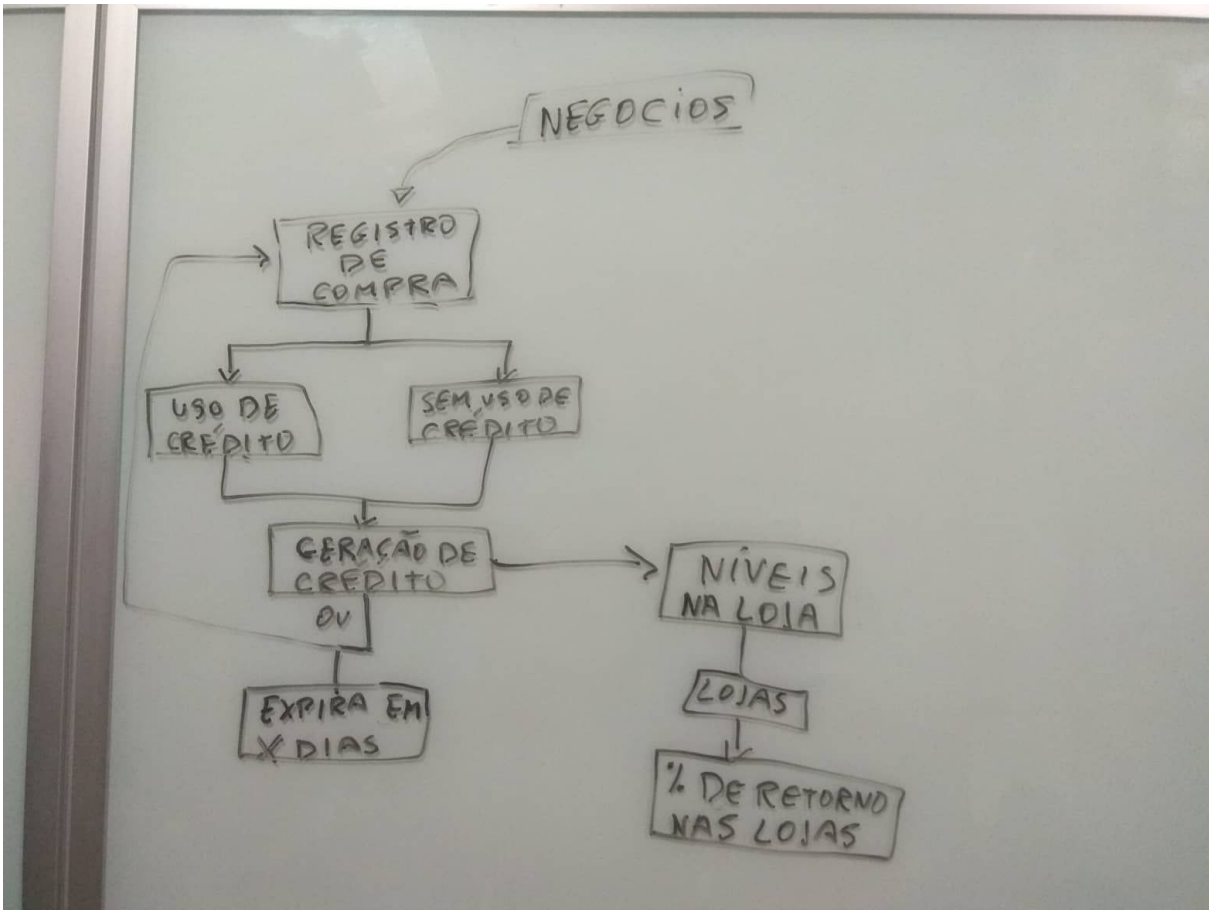
Referências **11**

Introdução

Projeto que utiliza de base a proposta da startup Porquin, voltada à fidelização de clientes de varejistas de moda feminina. A proposta se baseia no conceito de “créditos para compras seguintes”, no qual quando um consumidor realiza uma compra numa das lojas parceiras Porquin, uma parte do valor gasto (porcentagem) na compra volta para sua carteira virtual (aplicativo) em forma de crédito, disponibilizado para ser utilizado na compra seguinte em forma de descontos. Quanto mais um mesmo consumidor realiza compras num mesmo estabelecimento parceiro, maior é sua taxa de créditos retornada no aplicativo, estimulando-o a continuar comprando na mesma loja envolvida.

Arquitetura

Segue abaixo um MindMap que detalha a ideia da arquitetura geral do projeto:



A divisão do projeto foi feita da seguinte forma:

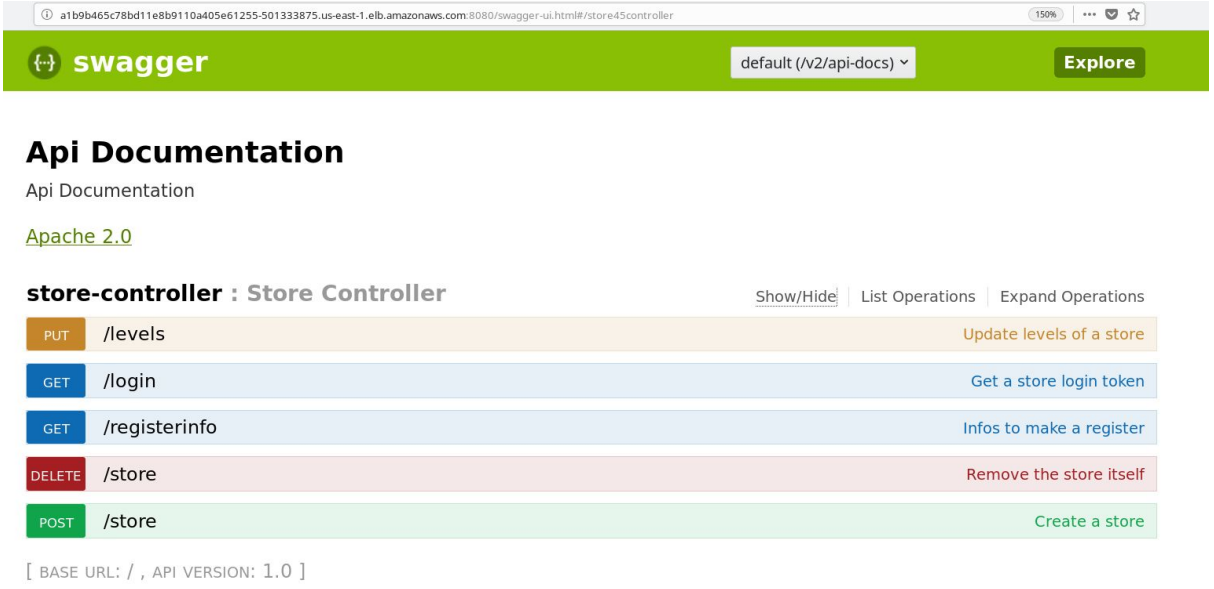
- Foi criado um microsserviço específico para lidar com os usuários e suas respectivas carteiras (onde ficam armazenados seus créditos), o creditService.
- Foi criado outro microsserviço específico para lidar com as lojas, informação de níveis e porcentagens de retorno, o storeService.

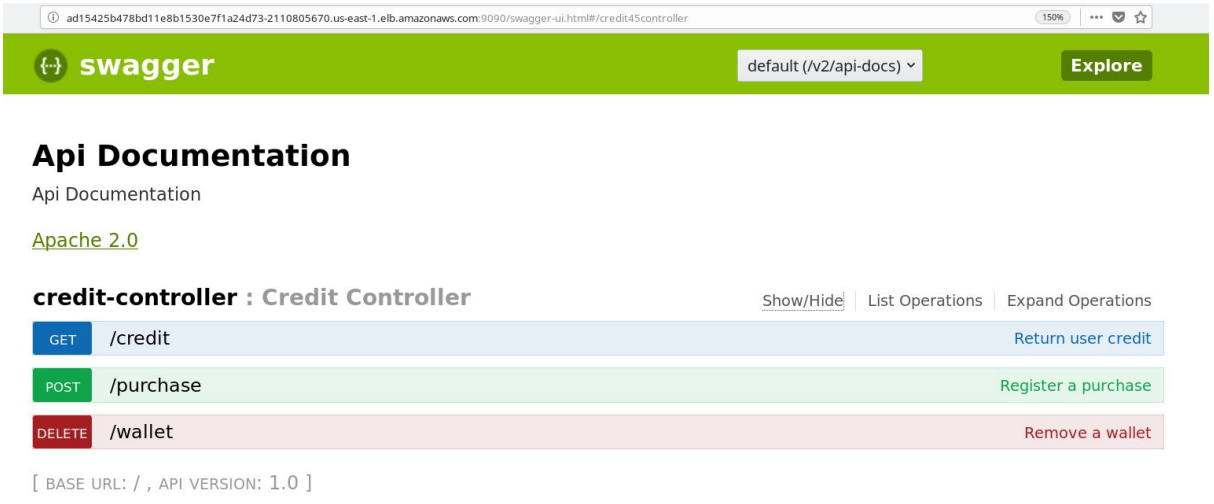
Quando um registro de compra é feito no creditservice, o mesmo consulta o storeservice, para buscar informações sobre os níveis da loja e poder calcular quanto de crédito aquela compra vai gerar para o usuário.

Projeto

Cada microserviço tem 3 arquivos de configuração de ambiente (application-*.yml), prod, test, dev. Em ambiente de produção eles utilizam o mongoDB da mLab e caso contrário tentam se conectar em um local.

Para a documentação dos microsserviços utilizamos o Swagger:





Inicialmente rodamos o Kubernetes localmente com o minikube:

Workloads

Workloads Statuses

Deployments

Pods

Replica Sets

Deployments

Name	Labels	Pods	Age	Images
creditservice	k8s-app: creditservice	3 / 3	15 hours	mlrbc/creditservice
storeservice	k8s-app: storeservice	3 / 3	15 hours	mlrbc/storeservice

Pods

Name	Node	Status	Restarts	Age
creditservice-b6cdff6fb-7gmb4	minikube	Running	1	15 hours
creditservice-b6cdff6fb-m69nj	minikube	Running	1	15 hours
creditservice-b6cdff6fb-wl7tm	minikube	Running	1	15 hours
storeservice-d79f8954-hgps9	minikube	Running	1	15 hours
storeservice-d79f8954-v55l4	minikube	Running	1	15 hours
storeservice-d79f8954-zb9kh	minikube	Running	1	15 hours

Replica Sets

Name	Labels	Pods	Age	Images
creditservice-b6cdff6fb	k8s-app: creditservice pod-template-hash: 627899296	3 / 3	15 hours	mlrbc/creditservice
storeservice-d79f8954	k8s-app: storeservice pod-template-hash: 83594510	3 / 3	15 hours	mlrbc/storeservice

Discovery and Load Balancing

Services

Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
kubernetes	component: apiserver provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	16 hours

Config and Storage

Secrets

Name	Type	Age
default-token-ggcnp	kubernetes.io/service-account-token	16 hours

Creation Time: 2018-06-24T06:40 UTC
Selector: k8s-app: creditservice
Strategy: RollingUpdate
Min ready seconds: 0
Revision history limit: 10
Rolling update strategy: Max surge: 25%, Max unavailable: 25%
Status: 3 updated, 3 total, 3 available, 0 unavailable

New Replica Set

Name	Labels	Pods	Age	Images
creditservice-b6cdff6fb	k8s-app: creditservice pod-template-hash: 627899296	3 / 3	15 hours	mlrbc/creditservice

Old Replica Sets

There is nothing to display here
This Deployment does not have any old replica sets

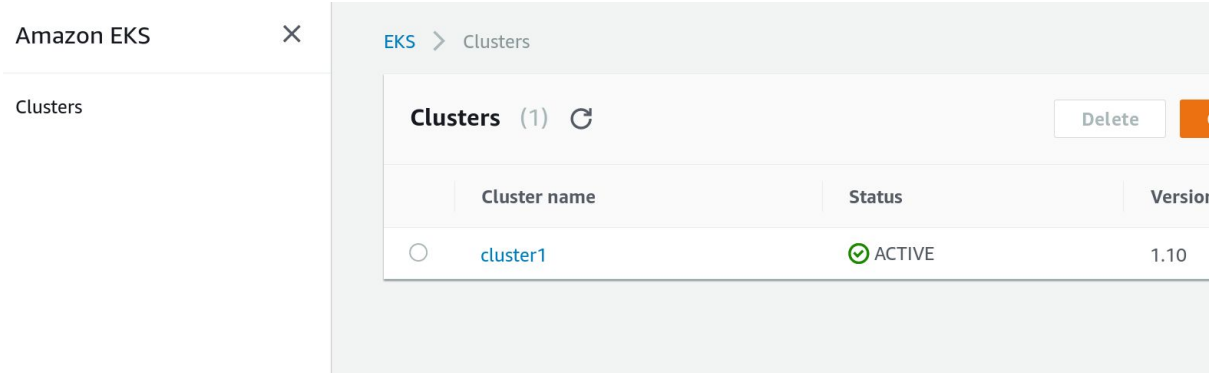
Horizontal Pod Autoscalers

There is nothing to display here
There are currently no Horizontal Pod Autoscalers targeting this Deployment.

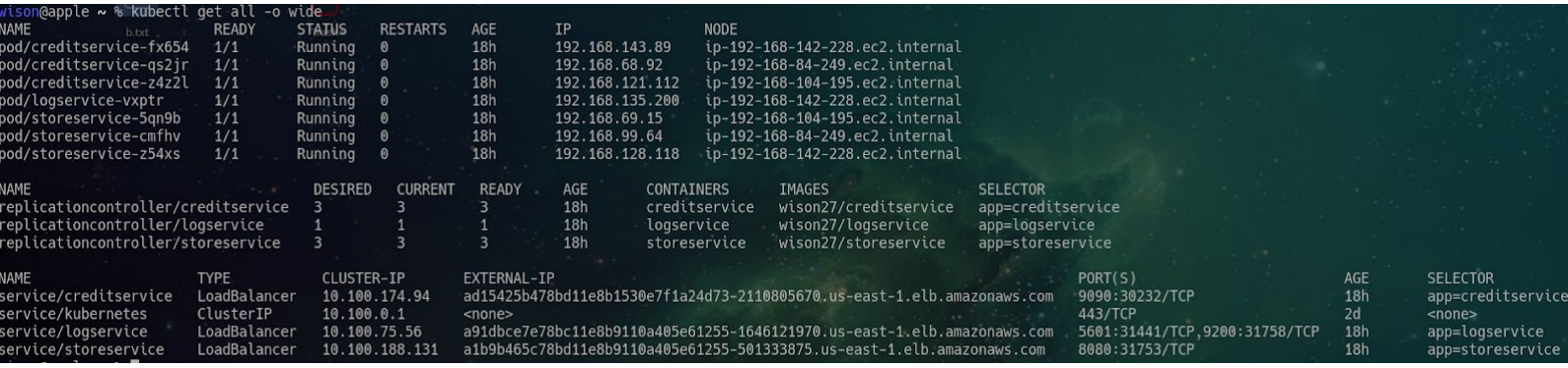
Events

Message	Source	Sub-object	Count	First seen	Last seen
Scaled up replica set creditservice-b6cdff6fb to 3	deployment-controller	-	1	2018-06-24T05:45 UTC	2018-06-24T05:45 UTC
Scaled up replica set creditservice-b6cdff6fb to 3	deployment-controller	-	1	2018-06-24T06:40 UTC	2018-06-24T06:40 UTC

Porém depois conseguimos configurá-lo no EKS da Amazon, criando um cluster com 3 máquinas do EC2 como hosts:



E então pelo kubectl enviamos jsons para criar o replication controller com seus pods e um service para o mesmo[1]:

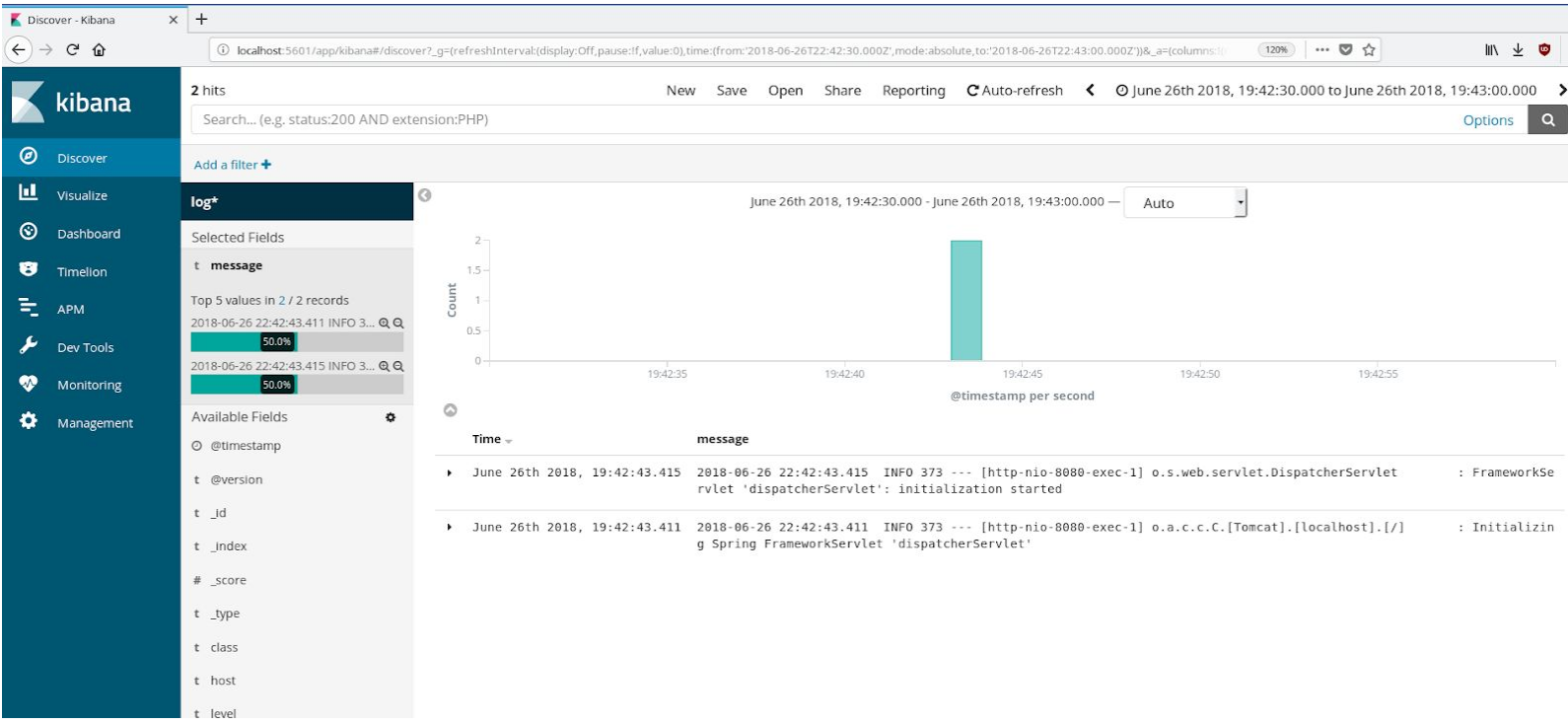


O acesso a eles ainda está disponível em:
StoreService ->
http://a1b9b465c78bd11e8b9110a405e61255-501333875.us-east-1.elb.amazonaws.com:8080/swagger-ui.html
CreditService ->
http://ad15425b478bd11e8b1530e7f1a24d73-2110805670.us-east-1.elb.amazonaws.com:9090/swagger-ui.html

Para log e monitoramento utilizamos o conjunto denominado ELK, composto por Elasticsearch, Logstash e Kibana.

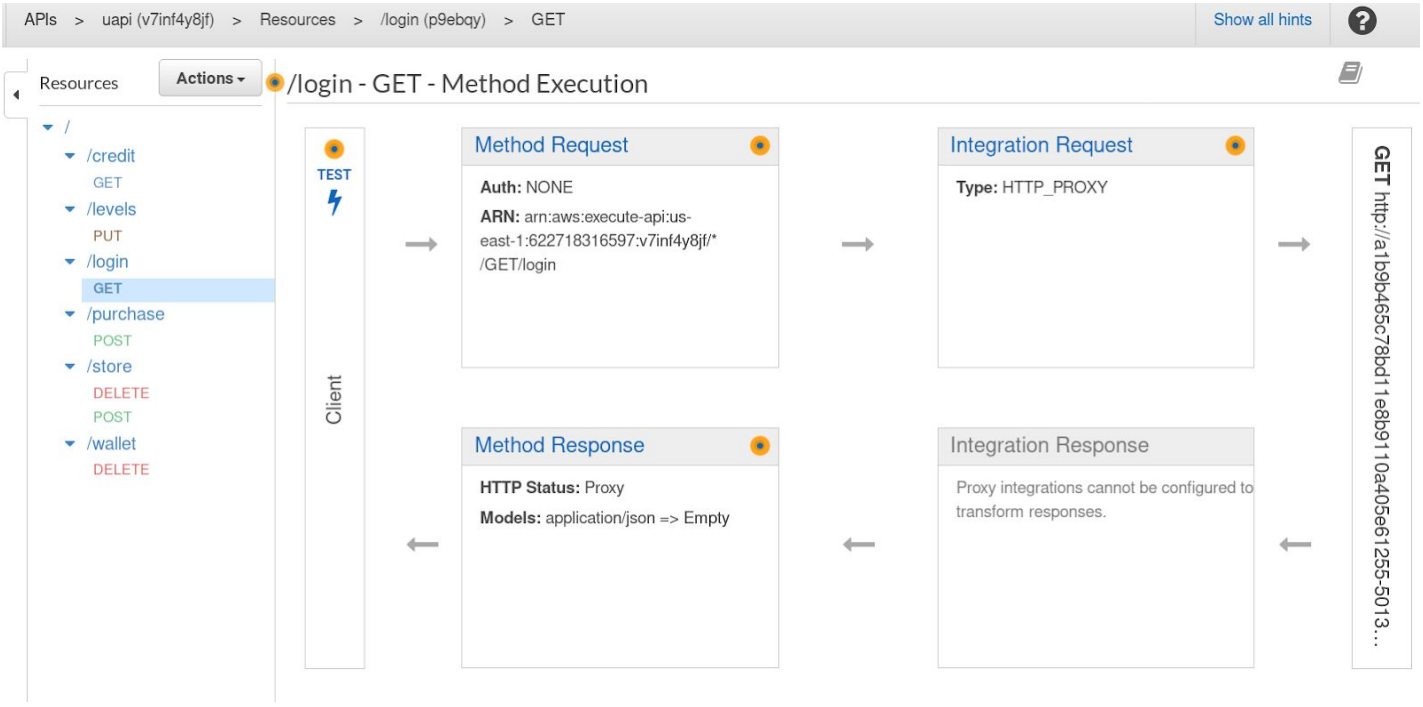
[1] https://github.com/if1007/storeService/tree/master/eks, https://github.com/if1007/creditService/tree/master/eks

Criamos um container denominado logservice que contém o Elasticsearch e o Kibana. E então colocamos o logstash em conjunto com o packetbeat nos containers do storeService[2] e creditService[3], dessa forma conseguimos guardar logs e monitorar cada instância dos containers. Porém não conseguimos rodar o projeto corretamente no EKS, mesmo utilizando como base do logservice o container oficial do elasticsearch[4], pois o mesmo requer que se altere algumas configurações default do kernel linux na máquina host[5].



Para a API Gateway utilizamos o Amazon API Gateway, que permite que tudo seja configurado pela dashboard deles:

[2] https://github.com/if1007/storeService/blob/master/log.sh
[3] https://github.com/if1007/creditService/blob/master/log.sh
[4] https://github.com/if1007/logservice/blob/master/Dockerfile#L1
[5] https://www.elastic.co/guide/en/elasticsearch/reference/current/vm-max-map-count.html, https://github.com/docker-library/elasticsearch/issues/111



Estando disponível em: https://v7inf4y8jf.execute-api.us-east-1.amazonaws.com/test/

Execução

Foram criados scripts para otimizar a execução dos projetos de várias maneiras. Todas as instruções para rodá-los localmente com (prod) e sem docker (dev), os testes (test) e ainda subí-los facilmente no EKS (se houver um arquivo kubeconfig do cluster configurado conforme a especificação do EKS[6]) encontram-se atualizadas nos seus respectivos README.md:

https://github.com/if1007/storeService
https://github.com/if1007/creditService
https://github.com/if1007/logservice

[6] https://docs.aws.amazon.com/pt_br/eks/latest/userguide/getting-started.html

Referências

Abaixo seguem objetos de consulta utilizados ao longo do desenvolvimento do projeto para auxílio e solução de dúvidas.

- Java

https://stackoverflow.com/questions/2607289/converting-array-to-list-in-java
http://www.javarepl.com/term.html

- Eclipse

https://stackoverflow.com/questions/10722773/import-existing-gradle-git-project-into-eclipse/44715981#44715981

- Spring Boot

https://spring.io/guides/gs/actuator-service/
https://stackoverflow.com/questions/23027315/does-application-yml-support-environment-variables/47527525#47527525
https://emmanuelneri.com.br/2017/05/14/profiles-no-spring-boot/

- Spring Boot Profile Config

https://stackoverflow.com/questions/46331059/reading-application-yml-in-service
https://www.mkyong.com/spring-boot/spring-boot-profile-based-properties-and-yaml-example/

- Requisições HTTP

https://spring.io/guides/gs/consuming-rest/
https://stackoverflow.com/questions/42592440/how-to-change-response-http-header-in-get-request-by-spring-resttemplate
https://stackoverflow.com/questions/10308452/how-to-convert-the-following-json-string-to-java-object
https://stackoverflow.com/questions/23205213/how-to-extract-http-status-code-from-the-resttemplate-call-to-a-url

- Swagger

https://github.com/swagger-api/swagger-core/wiki/annotations
https://dzone.com/articles/spring-boot-swagger-ui
http://www.baeldung.com/swagger-2-documentation-for-spring-rest-api

- Testes com Spring Boot

https://www.youtube.com/watch?v=8S8o46avgAw
https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-testing.html
https://springframework.guru/unit-testing-spring-mvc-spring-boot-1-4-part-1/

https://www.petrikainulainen.net/programming/gradle/getting-started-with-gradle-integration-testing/
https://stackoverflow.com/questions/24199393/how-to-send-a-mock-object-as-json-in-mockmvc/24200590#24200590
https://stackoverflow.com/questions/18336277/how-to-check-string-in-response-body-with-mockmvc/18336481

- Api Gateway

https://docs.aws.amazon.com/pt_br/apigateway/latest/developerguide/api-gateway-create-api-as-simple-proxy-for-http.html
https://spring.io/guides/gs/gateway/

- EKS

https://docs.aws.amazon.com/pt_br/eks/latest/userguide/getting-started.html

- Dockerfile

https://stackoverflow.com/questions/19537645/get-environment-variable-value-in-dockerfile
https://docs.docker.com/engine/reference/builder/#workdir

- Docker

https://www.mundodocker.com.br/link-entre-containers-parte-2/
https://docs.docker.com/docker-cloud/builds/push-images/
https://stackoverflow.com/questions/38118791/can-t-delete-image-with-children
https://stackoverflow.com/questions/37782505/is-it-possible-to-show-the-workdir-when-building-a-docker-image
https://gist.github.com/bastman/5b57ddb3c11942094f8d0a97d461b430

- Kubernetes

https://www.mundodocker.com.br/kubernetes-parte-i/
https://www.mundodocker.com.br/kubernetes-parte-ii/
https://churrops.io/2018/06/19/kubernetes-criando-um-cluster-simples-em-cloud-com-o-kubeadm/
https://www.youtube.com/watch?v=1ENtPSKjD2I&t=16s
https://github.com/vinicius3w/if1007-Microservices/blob/master/lectures/if1007-microservices-13-withnotes.pdf
https://aws.amazon.com/pt/kubernetes/

- ELK

https://howtodoinjava.com/microservices/elk-stack-tutorial-example/
https://www.elastic.co/guide/en/beats/packetbeat/current/packetbeat-getting-started.html