IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

Alkindar José Ferraz Rodrigues	SP3029956
Carolina de Moraes Josephik	SP3030571
Fabio Mendes Torres	SP3023184
Gabriely de Jesus Santos Bicigo	SP303061X
Leonardo Naoki Narita	SP3022498
Mariana da Silva Zangrossi	SP3030679

Prova de Conceito Lixt

São Paulo - SP - Brasil

29 de Junho de 2021

IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

Alkindar José Ferraz Rodrigues	SP3029956
Carolina de Moraes Josephik	SP3030571
Fabio Mendes Torres	SP3023184
Gabriely de Jesus Santos Bicigo	SP303061X
Leonardo Naoki Narita	SP3022498
Mariana da Silva Zangrossi	SP3030679

Prova de Conceito Lixt

Prova de conceito da aplicação Lixt da disciplina de Projeto Integrado I no 1° semestre de 2021.

Professor: Ivan Francolin Martinez

Professor: José Braz de Araujo

IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

Tecnologia em Análise e Desenvolvimento de Sistemas

PI1 - Projeto Integrado I

São Paulo - SP - Brasil

29 de Junho de 2021

Lista de abreviaturas e siglas

API	Application Programming Interface - Citado em 7
APK	*Android Package* - Pacote Android - Citado em 6
AWS	*Amazon Web Services* - Serviços Web da Amazon - Citado em 4
HTTPS	*Hyper Text Transfer Protocol Secure* - Protocolo de transferência de hipertexto seguro - Citado em 4
IPA	*iOS application file* - Arquivo de aplicação iOS - Citado em 6

Sumário

1 Infraestrutura

Visando implementar a aplicação conforme descrita no Desenho de Porjeto, foi desenvolvida uma infraestrutura usando os serviços disponibilizados pela *Amazon Web Services* (AWS). Mas, num desvio em relação aquele documento, a equipe optou por usar a ferramenta Github Actions para implementar a pipeline de Deploy. Usando esta pipeline, uma imagem de Docker é gerada com o executável do aplicativo, isolado em um ambiente de execução e com a porta de conexão exposta.

Esta imagem é, então, enviada ao AWS Elastic Container Registry, que mantém um histótico destas imagens, e a partir deste registry, uma tarefa é iniciada e colocada num container, contruídos de modo a complementar a imagem (expondo as portas esperadas pela imagem e pela aplicação, por exemplo). Este container é gerado e gerenciado de forma automática, tendo politicas de segurança associados a ele.

A partir deste ponto, o container se comunica com duas outras ferramentas: um cluster serverless de banco de dados aurora, compatível com MySQL 5, no AWS Relational Database Service, serviço que gerencia as instâncias de banco de dados de forma automática, e permite acesso a elas a partir de grupos de segurança específicos. Nestes grupos de segurança estão os containers da aplicação Back-end, o que permite a comunicação entre ambas as aplicações.

No outro lado da aplicação Back-end, o AWS Elastic Load Balancer oferece um ponto de comunicação entre o a rede privada, onde o Back-end e o banco de dados se encontram, e a internet externa. Este serviço redireciona as requisições *Hyper Text Transfer Protocol Secure* (HTTPS) externas entre todas as instâncias da aplicação que estejam rodando.

Alguns pontos a ser melhor desenvolvidos na infraestrutura são:

- Manter a instâncias da aplicação Back-end ativas e estáveis por um período maior, de modo que pelo menos uma possa responder a qualquer momento;
- Usar o protocolo HTTPS, como está estabelecido nos requisitos de segurança;
- Implementar outras pipelines, para escalar a aplicação e realizar o rollback a uma versão anterior;
- Isolar os ambientes de teste e de produção;
- Usar o serviço de gerenciamento de segredos para armazenar senhas de banco de dados;

- Proteger a branch master do repositório do Github, para que esta aceite novos commits apenas a partir de Pull Requests, uma vez que um commit novo nesta é o trigger para a execução da pipeline no Github Actions.

2 Tecnologias

Nesse capítulo, serão abordados as principais tecnologias e seus casos de uso no sistema.

2.1 Front-end

Para o desenvolvimento do Front-end da aplicação estamos utilizando a linguagem Javascript, a Framework React-Native e estilizando a interface com a biblioteca Native-Base. Além destas tecnologias, utilizamos as seguintes ferramentas da Framework Expo:

- Expo CLI:
  - Possibilita a criação de novos projetos de React-Native com estrutura básica inclusa;
  - Possibilita a execução da aplicação em modo de desenvolvimento com recarregamento rápido, viabilizando a emulação no navegador, em dispositivos móveis (a partir do Expo Go) e em emuladores de dispositivos móveis;
  - Possibilita fazer o Build dos arquivos binários *Android Package* (APK) e *iOS application file* (IPA);
- Expo Go:
  - Possibilita a execução da aplicação em desenvolvimento em dispositivos móveis antes do Deploy

2.2 Back-end

Para o desenvolvimento do Back-end da aplicação, foi utilizado a linguagem Java com o Framework Spring e todo seu ecossistema de tecnologia disponível.

As principais tecnologias doBack-end são:

- Lombok: Automatiza métodos amplamente utilizados no java, tais como *getters*, *setters* e construtores.
- Hibernate: Permite a conexão com o banco de dados através de uma abstração de alto nível.

- Hibernate-Spatial: É um complemento do hibernate, permitindo a integração com os dados do tipo espacial, permitindo geolocalização.
- Spring-Data: É uma ferramenta do ecossistema Spring que permite uma forma mais prática de comunicar a Application Programming Interface (API) com o banco de dados através do hibernate, fornecendo uma abstração mais prática do que a do Hibernate clássico.
- Flyway: É um versionador de scripts de banco de dados.
- OAuth2: É uma tecnologia que fornece um servidor de autenticação e autorização, garantindo alta segurança e alta escalabilidade.
- Spring Security: É uma ferramenta do ecossistema Spring que fornece recursos necessários para proteger uma API.
- Javamail: É um plugin que permite o envio de email dentro da nossa própria API.
- Swagger: É um plugin que permite a documentação da API.

Glossário

Back-end	Um sistema *back-end* é aquele que se encontra na camada de servidor, em uma aplicação de arquitetura cliente-servidor. Sua principal função é fornecer informações e capacidade de processamento a aplicação cliente. - Citado em 4, 6
branch	Uma ramificação em controles de versão são ponteiros para as alterações feitas nos arquivos do projeto - Citado em 5
Build	Termo usado para identificar uma versão compilada de um programa - Citado em 6
commit	Ato de enviar e guardar, ou seja, enviar dados ou códigos para armazenamento em um banco de dados ou em um sistema de controle de versão - Citado em 5
Deploy	Processo pelo qual a aplicação é implantada em ambiente de prpdução, e está disponível para os usuários finais. - Citado em 4, 6
Framework	Conjunto de ações e estratégias que possuem um ojeto e objetivo em específico; Na programação, é um conjunto de pacotes e bibliotecas que abstarai alguma função complexa, geralmente de nível mais baixo, e sobre o qual uma aplicação pdoe ser contruída. - Citado em 6
Front-end	Um sistema *front-end* é aquele que encontra na camada cliente, em uma aplicação de arquitetura cliente-servidor. Sua principal função, no escopo deste projeto, é atuar como interface gráfica para o usário, coletar dados e enviá-los para o *back-end*. - Citado em 6
pipeline	Série de passos automatizados que devem ser implementados para entregar uma nova versão de um software - Citado em 4, 5
rollback	Processo de retornar um banco de dados ou um programa para um estado anterior, geralmente utilizado em recuperação de um estado antes de erros acontecerem - Citado em 4