LogiPack

Lab. 4 - Introdução à Engenharia de Software

Leonardo Almeida • Pedro Rodrigues • Rafael Gonçalves • Rafael Remígio

Licenciatura em Engenharia Informática
2022/2023

universidade de aveiro

Introdução

Objetivo: propor, conceptualizar e implementar uma solução de software *multi-layer*, de classe empresarial e com foco na web.

Proposta: sistema de gestão de transportes de mercadorias, destinado às entidades que operam na área da Logística (cadeias de abastecimento).

Equipa

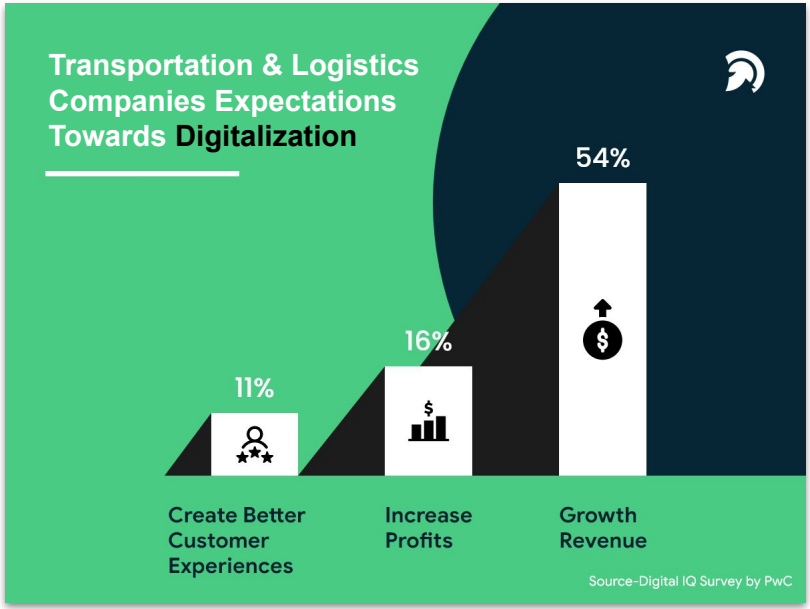
Papel	**Aluno**
Team Manager (*Coordinator*)	Pedro Rodrigues (102778)
Product Owner	Leonardo Almeida (102536)
Architect	Rafael Remígio (102435)
DevOps Master	Rafael Gonçalves (102534)
Developers	Todos os supracitados

Planeamento

Conceito do produto

Visão

- Perante a **transformação digital**, as empresas tiveram de mudar de paradigma, havendo cada vez mais negócios *eCommerce*.
- Novos S.I. não se destinam apenas a clientes finais, mas também a entidades prestadoras de serviços ou fornecedoras de produtos.
- Na Logística, as plataformas digitais podem ajudar as empresas a gerir melhor as suas cadeias de abastecimento, aumentando a **eficácia** e **eficiência** das suas operações.



Fonte: *Questionário da PwC, uma das maiores multinacionais de consultoria e auditoria do mundo.*

Muitos dos sistemas que partilham estes objetivos são exclusivos das suas redes de distribuição.
A LogiPack é uma **solução genérica**, passível de ser integrada em qualquer cadeia de abastecimento.

Levantamento de requisitos

Importância

- Permite entender as necessidades dos utilizadores.
- Permite definir objetivos do projeto.
- Permite identificar possíveis problemas, na fase de planeamento.

Ferramentas

- Atores, casos de utilização e *user stories*.
- Personas.
- Cenários.

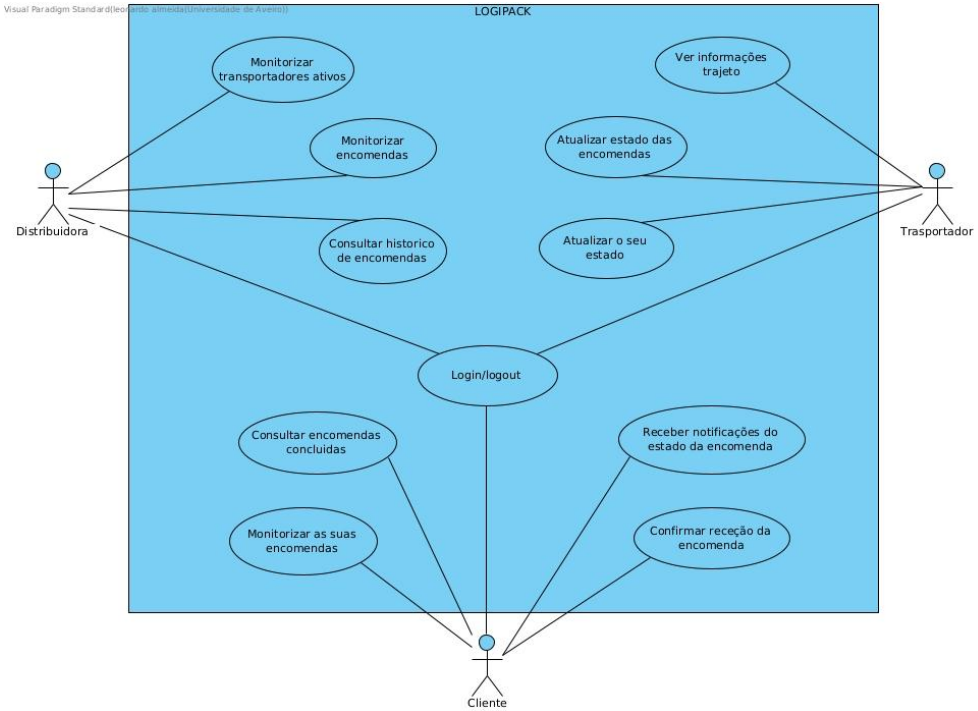
Resultado

- Garantir que a aplicação tem o comportamento esperado.
- Organizar o seu desenvolvimento.
- Reduzir riscos.

Atores, Casos de utilização e *User stories*

Objetivo:

- Entender o tipo de entidades que vão usar a aplicação.
- Dividir o projeto em partes mais pequenas, geríveis e “focadas”.
- Entender o contexto e o benefício de cada funcionalidade.



Personas

Objetivo:

- Entender as necessidades, expectativas e comportamentos dessas entidades.

Administrador da distribuidora

Factos sobre o Daniel:

- Tem 10 anos de experiência na Logística.
- É organizado e atento aos detalhes.
- Valoriza a eficiência e a satisfação dos clientes, estando constantemente à procura de melhorar os processos da empresa.

Motivações para usar o sistema:

- Acompanhar as frotas de entrega.
- Coordenar os transportadores.
- Atender às necessidades dos clientes, com celeridade, a fim de manter a boa reputação dos serviços da distribuidora.
- Ver estatísticas em tempo real.

Nome	Daniel Santos
Idade	35 anos
Sexo	Masculino
Profissão	Gestor de operações (*Head of Operations*) Setor: Logística
Estado civil	Casado
Localização	Lisboa

Cenários

Objetivo:

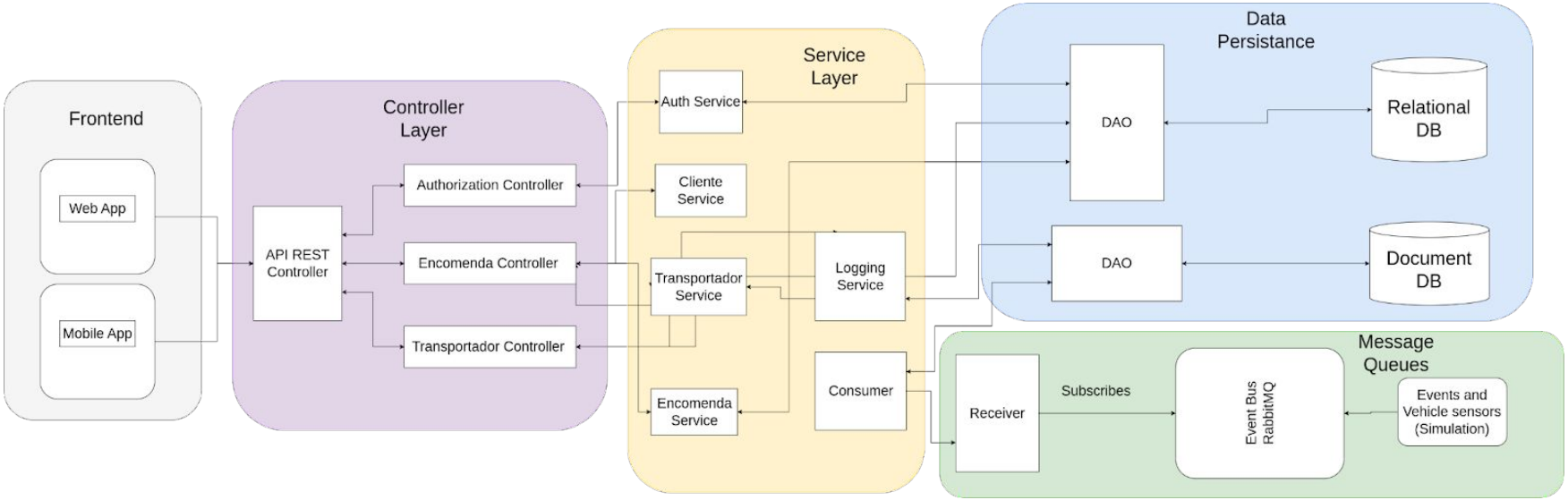
- Identificar problemas no uso da aplicação.
- Priorizar e validar requisitos.

Daniel Santos (administrador da distribuidora)

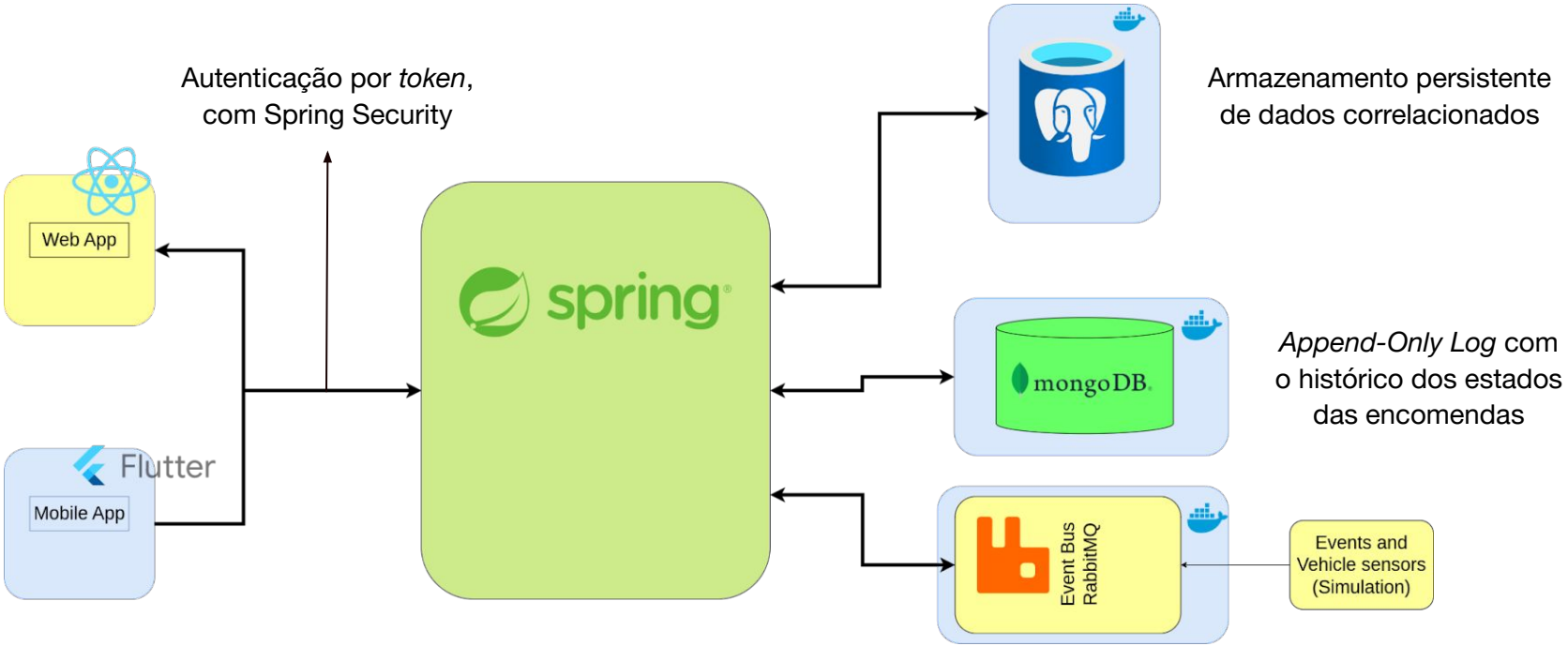
O Daniel está a preparar-se para um dia atarefado, no centro de distribuição. Inicia sessão na LogiPack, para consultar a lista de encomendas, prontas a ser expedidas. Constata que há uma carga adicional, que não estava prevista no início da semana, e que o sistema atribuiu automaticamente uma nova rota a um transportador inativo.

Arquitetura

Vista geral



Tecnologias utilizadas



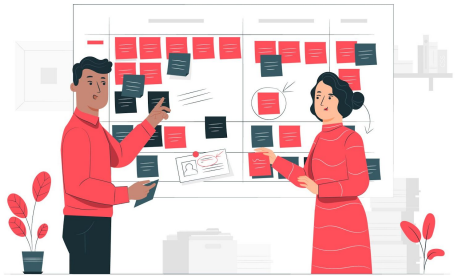
Desenvolvimento

Gestão do projeto

Na gestão do projeto, foram utilizados diferentes conceitos da filosofia *Agile*, sendo os mais relevantes:

- ***User Story*****,** como unidade de planeamento;
- ***Backlog***, para priorizar as tarefas a realizar, bem como a sua atribuição;
- ***Story Points***, para quantificar em termos de esforço, dificuldade e tempo necessário para concluir uma tarefa.
    - O sistema de *story points* utilizado consiste na sequência de números 0, 2, 4 e 8, onde 0 representa uma tarefa de esforço mínimo e 8 de esforço elevado.

A atribuição e pontuação das tarefas foi, sempre que possível, efetuada perante a presença de todos os elementos do grupo.



Gestão do *Backlog*

Para a gestão do *backlog*, a plataforma escolhida foi o ***Atlassian Jira***.

Principais motivos da escolha desta plataforma:

- Vasto suporte para a filosofia *Agile*, incluindo a *Scrum Methodology*
- Elevada transparência
- Flexibilidade
- Acessibilidade/facilidade
- Integração com o GitHub

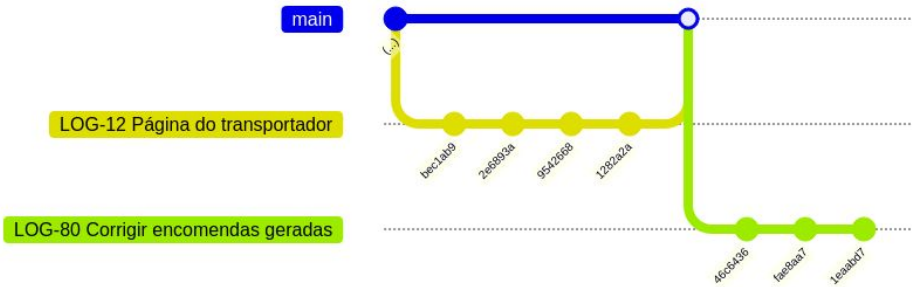
Repositório

Controlo de versões: Git

Hospedagem do código-fonte: GitHub

Feature-branching workflow:

- O desenvolvimento do projeto foi orientado a *branches*.
- A integração (*merge*) de novas funcionalidades na *main* foi quase sempre antecedida por um ***pull request***, cuja aprovação era habitualmente da responsabilidade do DevOps *master*.

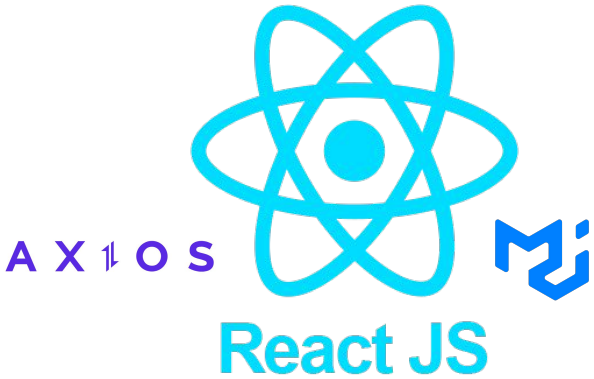


Aplicação web

Para o desenvolvimento da camada *View*, recorremos à biblioteca JavaScript **ReactJS**.

As principais razões pelas quais escolhemos ReactJS foram:

- Desenvolvimento rápido e fácil de interfaces ricas e interativas
- Componentes reutilizáveis
- *Data-Binding* proficiente
- Comunidade larga e ativa
- Boa documentação



Para interação com a API, utilizámos a biblioteca JavaScript **Axios**.

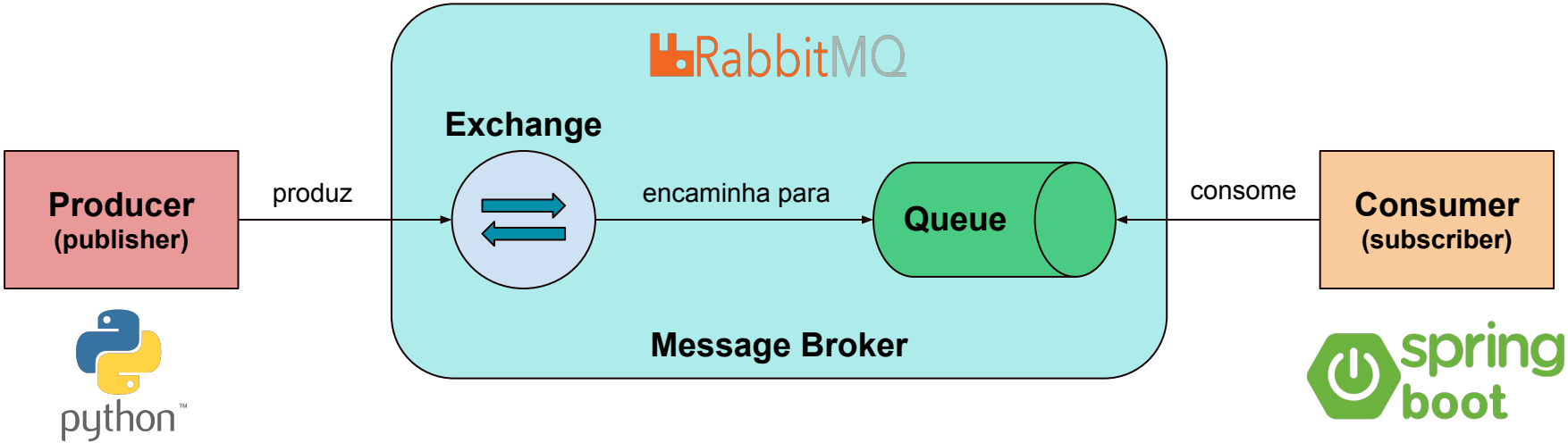
Usufruímos de componentes *React* disponibilizados pela biblioteca **Material UI**.

Aplicação móvel

Para o desenvolvimento da Mobile App, optámos pelo Flutter, pelas seguintes razões:

- *Cross-Platform codebase.*
- Componentes Widget agilizam o desenvolvimento da UI.
- Testagem rápida com *Hot Reload* e boas ferramentas de *debugging*.
- Experiência prévia com a *framework*.

Geração de dados



Backend

Construção da API

Abordagem RESTful:

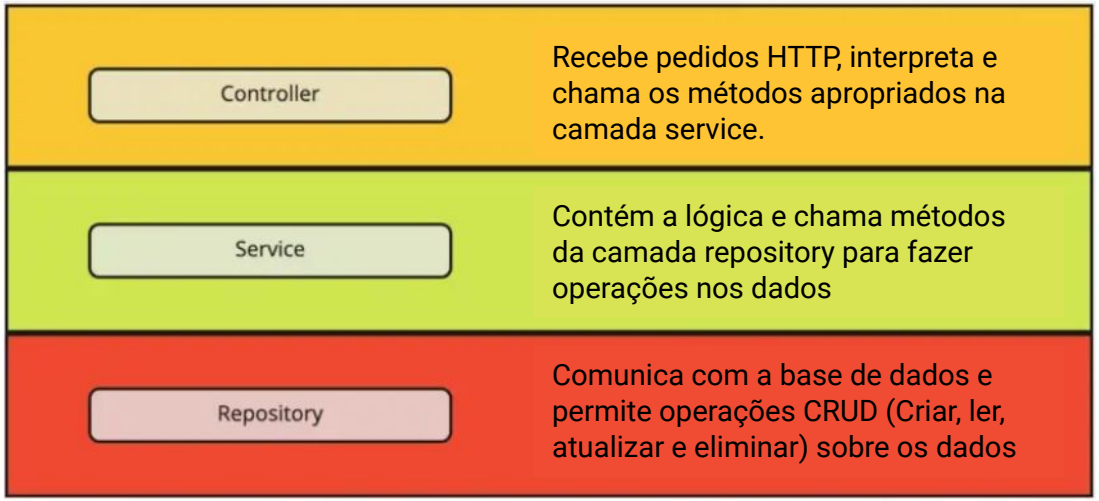
- GET, POST, PUT, DELETE
- Retorno em JSON
- Permite filtros

Implementação:

- Controller
- Service
- Repository

Documentação e testes:

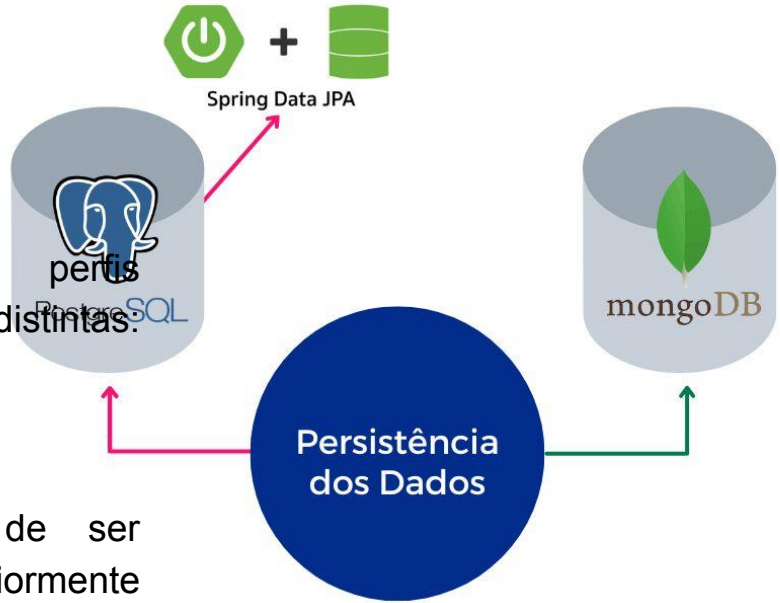
- Postman



POSTMAN

spring boot

Persistência dos Dados

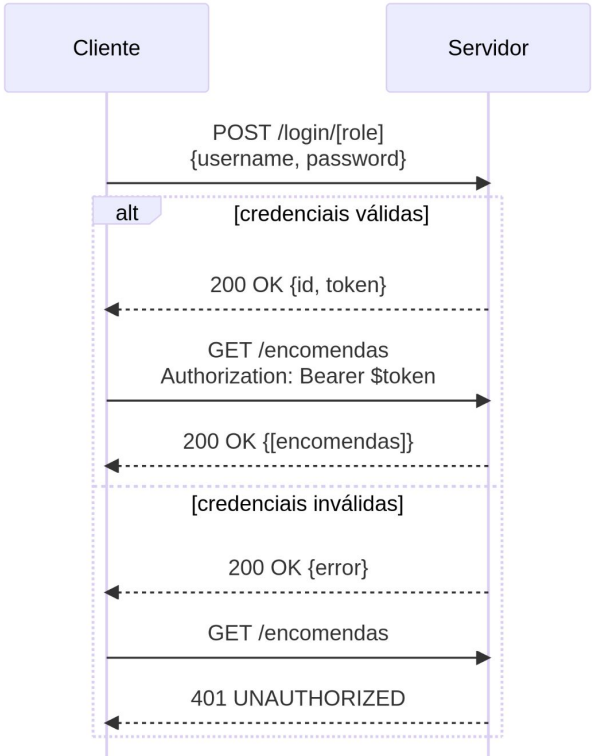


Porquê duas bases de dados?

O sistema desenvolvido apresenta dois perfis distintos com necessidades distintas:

1. Perfil onde os dados apenas necessitam de ser armazenados, de modo a serem acedidos posteriormente através da REST API.
2. Perfil que exige uma inserção contínua de dados de um mesmo tipo, isto é, pertencentes a uma mesma entidade.

Segurança no acesso à API



- Com **Spring Security**, é possível definir em que circunstâncias devem ser aceites pedidos HTTP à API.
- Recorremos ao protocolo de autorização **OAuth2**, cujo funcionamento depende de:
  - *Authorization server*: emite tokens.
  - R*esource server*: aloja os recursos protegidos e responde a pedidos de acesso.
- No nosso sistema, ambos os servidores são da responsabilidade da framework Spring.
- Um **JSON Web Token (JWT)** é um mecanismo seguro, compacto e *self contained* de transmitir informação entre duas partes, na forma de um objeto JSON.
- No nosso sistema, é assinado digitalmente com um pares de chaves pública/privada (RSA).

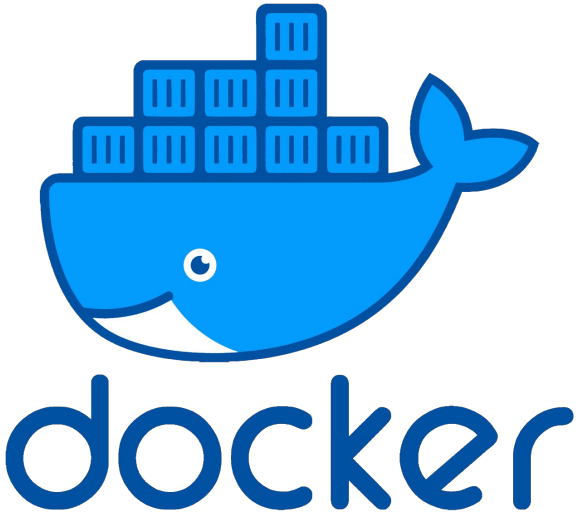
Deployment

Baseado em *containers*

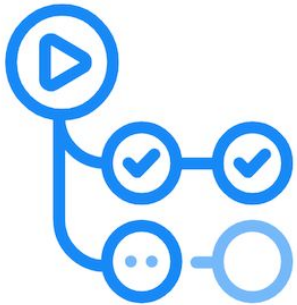
O **Docker** é um mecanismo de **virtualização**, que atua sobre o SO e visa o encapsulamento das camadas de uma aplicação em *containers*, com as suas dependências e configurações, o que facilita a sua distribuição e execução em qualquer ambiente, incluindo o de produção.

Apresenta ainda outros benefícios, tais como:

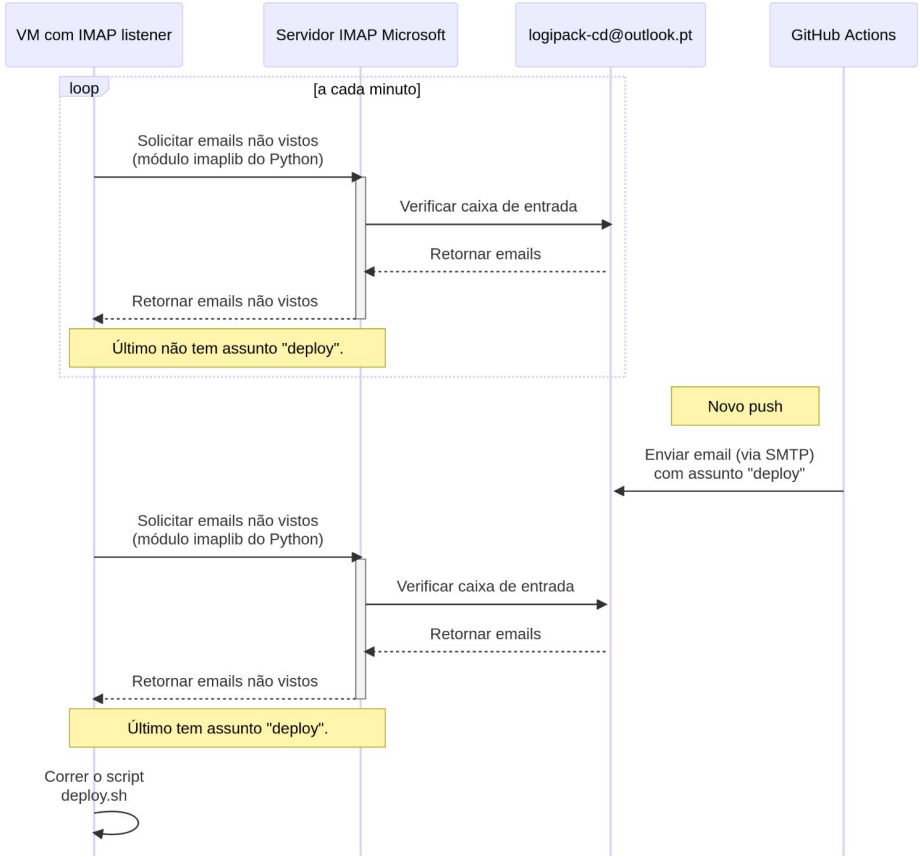
- Portabilidade
- Isolamento
- Escalabilidade
- Eficiência (em comparação com as VMs)
- Segurança



Continuous Deployment



GitHub Actions



Demonstração