

Universidade Presbiteriana Mackenzie  
Ciência da Computação - Laboratório de Engenharia de Software

## Projeto - Laboratório de Engenharia de Software

Francisco Losada Totaro - 10364673

Pedro Henrique L. Moreiras - 10441998

São Paulo

2025

# Sumário

<b>Capítulo 1:</b>	<b>3</b>
Introdução	3
<b>Capítulo 2:</b>	<b>4</b>
Problema e oportunidade	4
Produto de Software	4
Clientes e usuários	4
Etapas de construção	4
Critérios de qualidade	4
<b>Capítulo 3</b>	<b>6</b>
Requisitos funcionais	6
Requisitos não funcionais	6
<b>Capítulo 4</b>	<b>7</b>
Wireframes	7
<b>Capítulo 5</b>	<b>10</b>
Casos de Uso	10
Diagrama de Classe:	11
Diagramas de Sequência	12
<b>Capítulo 6</b>	<b>14</b>
Descrição da arquitetura do sistema (modelo)	14
Ferramentas que serão utilizadas	14
<b>Capítulo 7</b>	<b>16</b>
Desenvolvimento.	16
<b>Capítulo 8</b>	<b>17</b>
Resultados.	17
<b>Capítulo 9</b>	<b>18</b>
Conclusão	18
Trabalhos Futuros	18

# Capítulo 1:

## Introdução

O CarbonFlight é um programa que, utilizando informações de rotas aéreas e suas emissões de carbono, calcula a rota do aeroporto "A" para o aeroporto "B" mais benéfica para o meio ambiente.

O usuário seleciona o aeroporto de partida e o de destino, o programa calcula e indica a rota com menor impacto para o meio ambiente. Além disso, será mostrado as outras rotas possíveis, com suas emissões e, possivelmente, seus tempos de viagem e custo.

O projeto será desenvolvido em Java, de maneira multidisciplinar em conjunto com Grafos e Interação Humano-Computador.

## **Capítulo 2:**

### **Problema e oportunidade**

Em geral, os transportes aéreos, assim como os terrestres, emitem uma quantidade significativa de carbono. Nos transportes aéreos, essa quantidade pode ser influenciada por diversos fatores, desde a quantidade de passageiros presentes até o modelo específico do avião. Porém, mesmo com esses fatores em mente, é possível chegar a um mesmo destino com uma menor transmissão de carbono, apenas alterando uma rota. A partir disso, foi possível enxergar uma oportunidade em desenvolver um produto que ajudasse na redução de emissão de carbono, mostrando para o usuários os trechos de rotas aéreas e um ponto A até um ponto B.

### **Produto de Software**

O produto será um software que o usuário pode colocar seu aeroporto de partida e destino, e receberá sugestões das rotas com menores emissões de carbono.

### **Clientes e usuários**

- Passageiros ou pessoas que se preocupam com o meio ambiente e estão interessadas em planejar viagens com menor impacto ambiental.
- Agências de turismo ou profissionais que desejam comparar alternativas sustentáveis.
- Usuários leigos, que apenas desejam conhecer as possíveis rotas entre dois aeroportos.
- Gestores de empresas com interesse em reduzir os gastos de carbono como métrica de sustentabilidade.

### **Etapas de construção**

A construção consiste em algumas etapas. A primeira foi a ideação do produto, de maneira que ele, além de ser uma ideia original, seguisse os requisitos das três matérias em que ele será desenvolvido em conjunto.

Após a ideação, é necessário entendermos para quem o produto vai ser feito, seus requisitos, pensar nas personas, cenários, entre outras métricas.

Depois partimos para a modelagem leve do sistema, com diagramas de caso de uso, sequência e classes, e fizemos os primeiros protótipos de baixa fidelidade.

### **Critérios de qualidade**

Os principais critérios de qualidade para o produto são, inicialmente, mostrar as principais rotas disponíveis entre dois pontos escolhidos pelo usuário, destacando o total

de carbono emitido por essa rota, tendo como principais variáveis o modelo da aeronave, seu gasto de combustível, número de passageiros, distância, quantidade de paradas e tempo de viagem, entre outros fatores.

Um outro critério é o User Experience, que precisa ser fácil e intuitiva de ser utilizada por qualquer pessoa que tenha interesse no produto.

## Capítulo 3

### Requisitos funcionais

Requisitos	Prioridade
Dar ao usuário a opção de selecionar uma rota (definindo aeroporto de partida e destino).	4
Mostrar o cálculo da emissão de carbono e as principais características da rota ao usuário.	5
Mostrar o histórico de pesquisa do usuário (Mostrando as rotas pesquisadas anteriormente).	3
Dar diferentes alternativas de rotas, baseada na rota definida pelo usuário.	4
Botão de ajuda, mostrando como funciona o aplicativo	3

### Requisitos não funcionais

Requisitos	Prioridade
Tempo de resposta, no máximo 5 segundos, do aplicativo para mostrar o cálculo de emissão e as principais informações da rota	4
Utilizar API com dados confiáveis sobre a emissão de carbono	4
Histórico das rotas pesquisadas salvas em nuvem	3
Interface intuitiva e de fácil utilização	3
Disponibilidade de 24 horas, do uso do sistema	2

# Capítulo 4

## Wireframes

Para os protótipos, buscamos um design mais simples e intuitivo, mostrando apenas as informações cruciais que o usuário busca a utilizar nosso programa. A figura 1 é a tela inicial, onde podemos selecionar o aeroporto de partida e de destino, buscar a melhor rota e consultar seu histórico de pesquisa. A figura 2 e a figura 3 são a tela de resultados, onde você pode buscar mais informações sobre cada rota e a figura 4 o histórico.



The wireframe shows the home screen of the CarbonFlight application. It features a light beige background. In the top left corner, the text "CarbonFlight" is displayed in a bold, black, sans-serif font. In the top right corner, there is a small circular icon containing a question mark. Below the header, there are two input fields for airport names. The first is labeled "Aeroporto de Partida" and the second is labeled "Aeroporto de Destino". Both fields have a placeholder text "Nome...". Below these fields are two buttons: "Buscar Melhor Rota" and "Histórico de Pesquisa". The "Histórico de Pesquisa" button has a right-pointing chevron icon.

FIGURA 1



FIGURA 2



FIGURA 3





FIGURA 4

## Capítulo 5

Para um melhor entendimento do fluxo do nosso aplicativo, desenvolvemos diferentes diagramas que foram desenvolvidos utilizando a plataforma Figma. Na figura 1, realizamos um diagrama de caso de uso, onde mostramos a interação do usuário com as principais funções da aplicação. Na figura 2, desenvolvemos um diagrama de classes, onde mostramos um fluxo melhor do aplicativo, além de mostrarmos as devidas funções e seus atributos de cada classe principal do aplicativo. Por fim, realizamos dois diagramas de sequência, na figura 3, mostramos as interações na hora de realizar a busca de uma rota e na figura 4, as interações de consultar o histórico de rotas.

### Casos de Uso

=

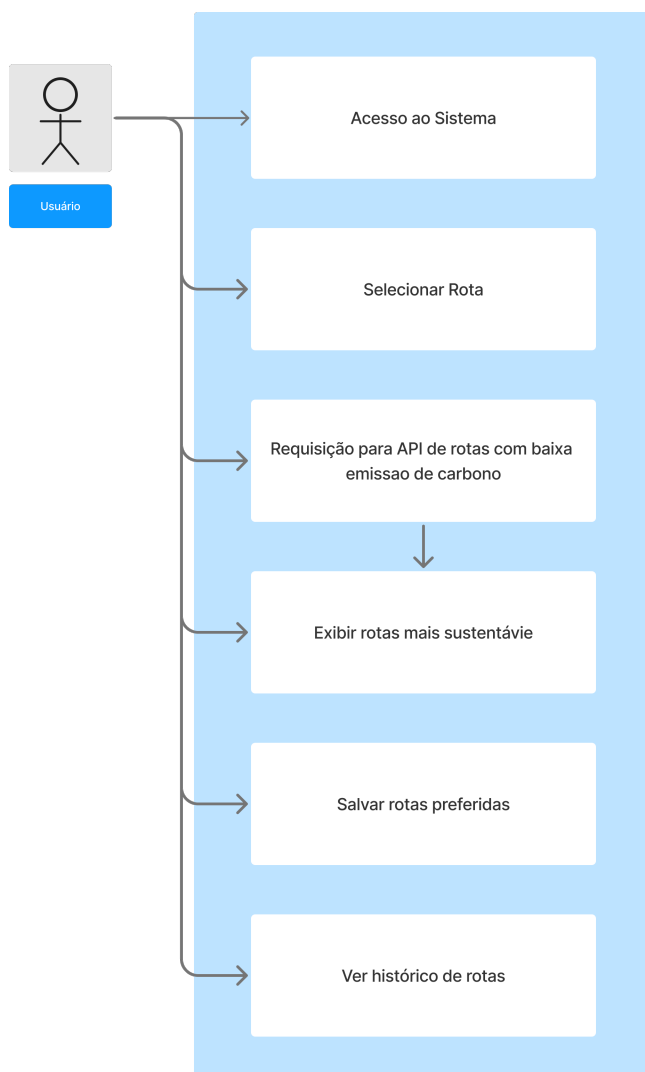


FIGURA 1

## Diagrama de Classe:

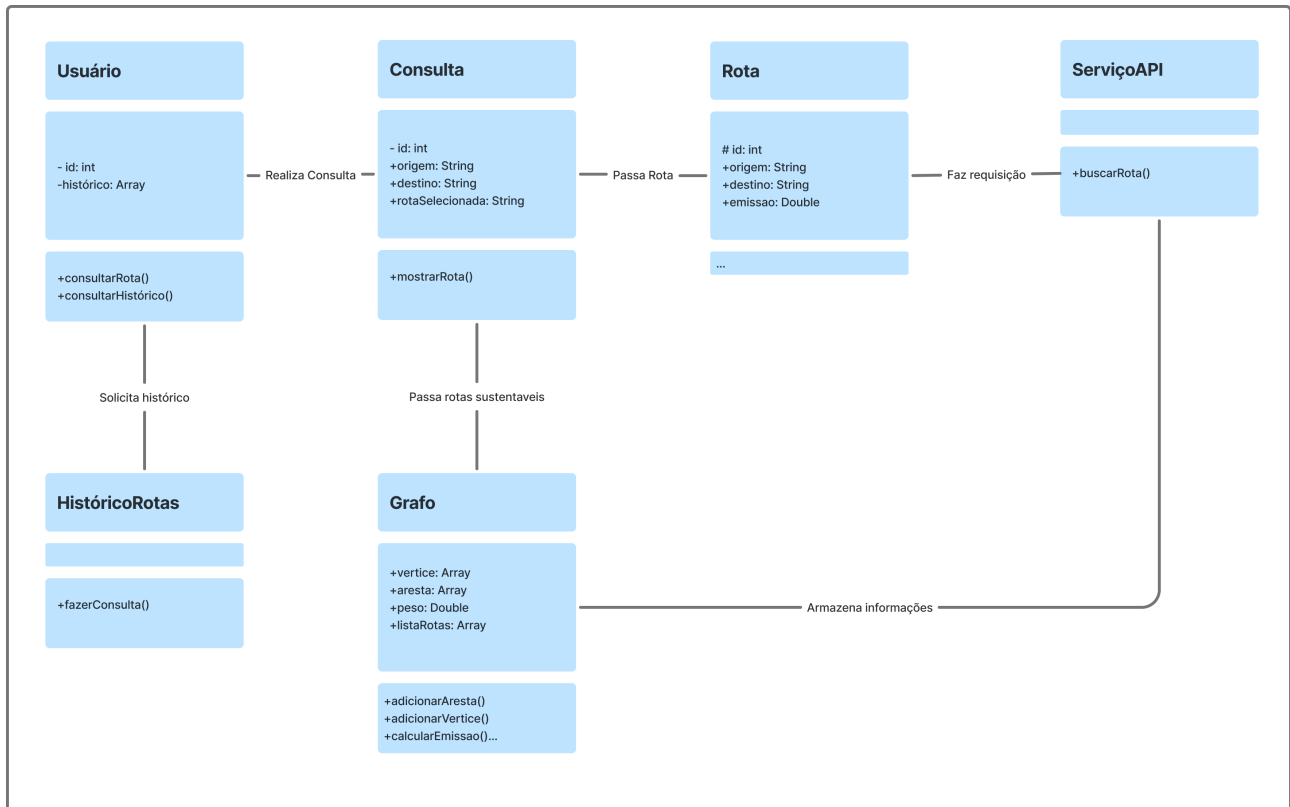


FIGURA 2

## Diagramas de Sequência

### Consultar Rota

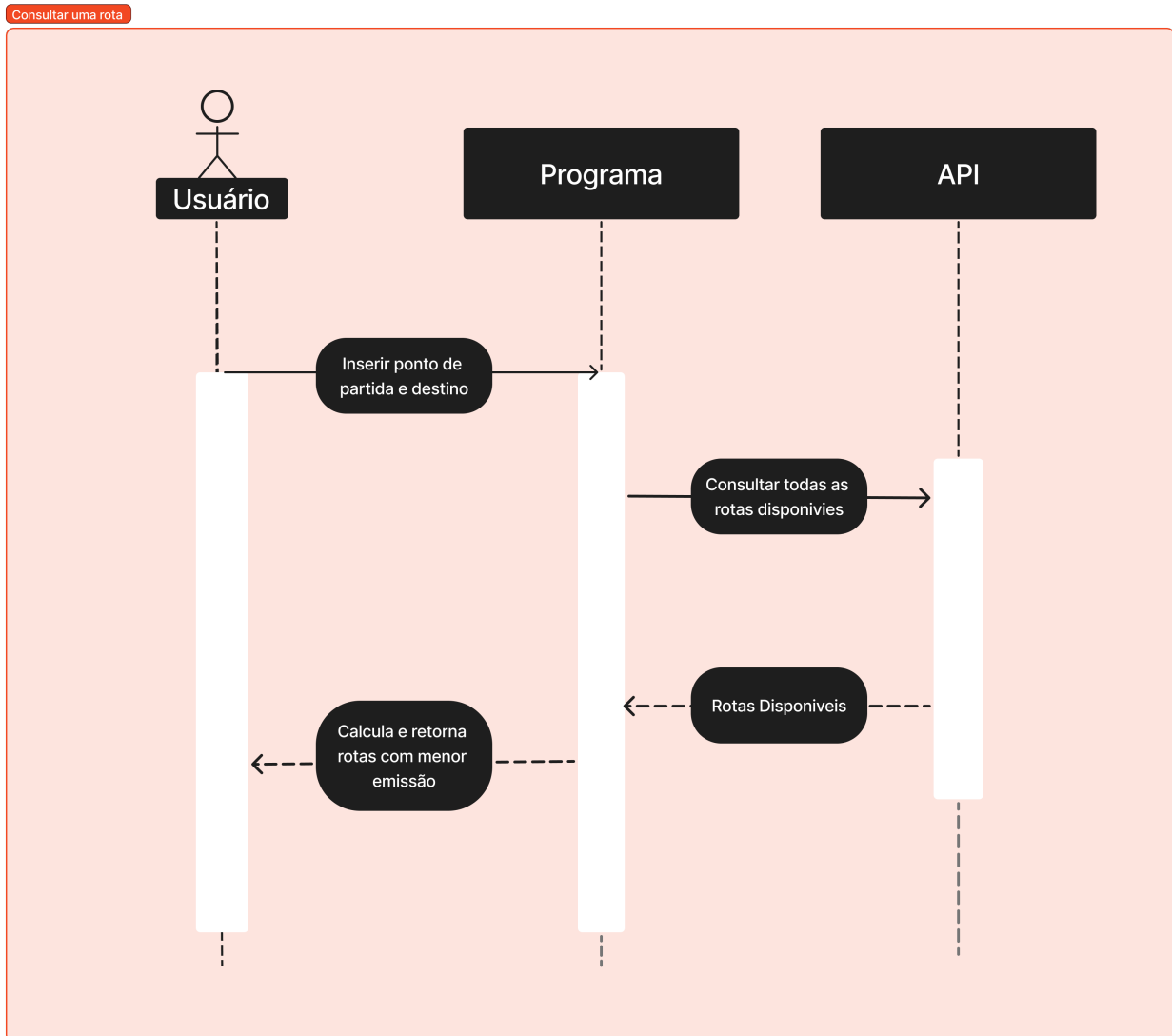


FIGURA 3

Consultar Histórico:

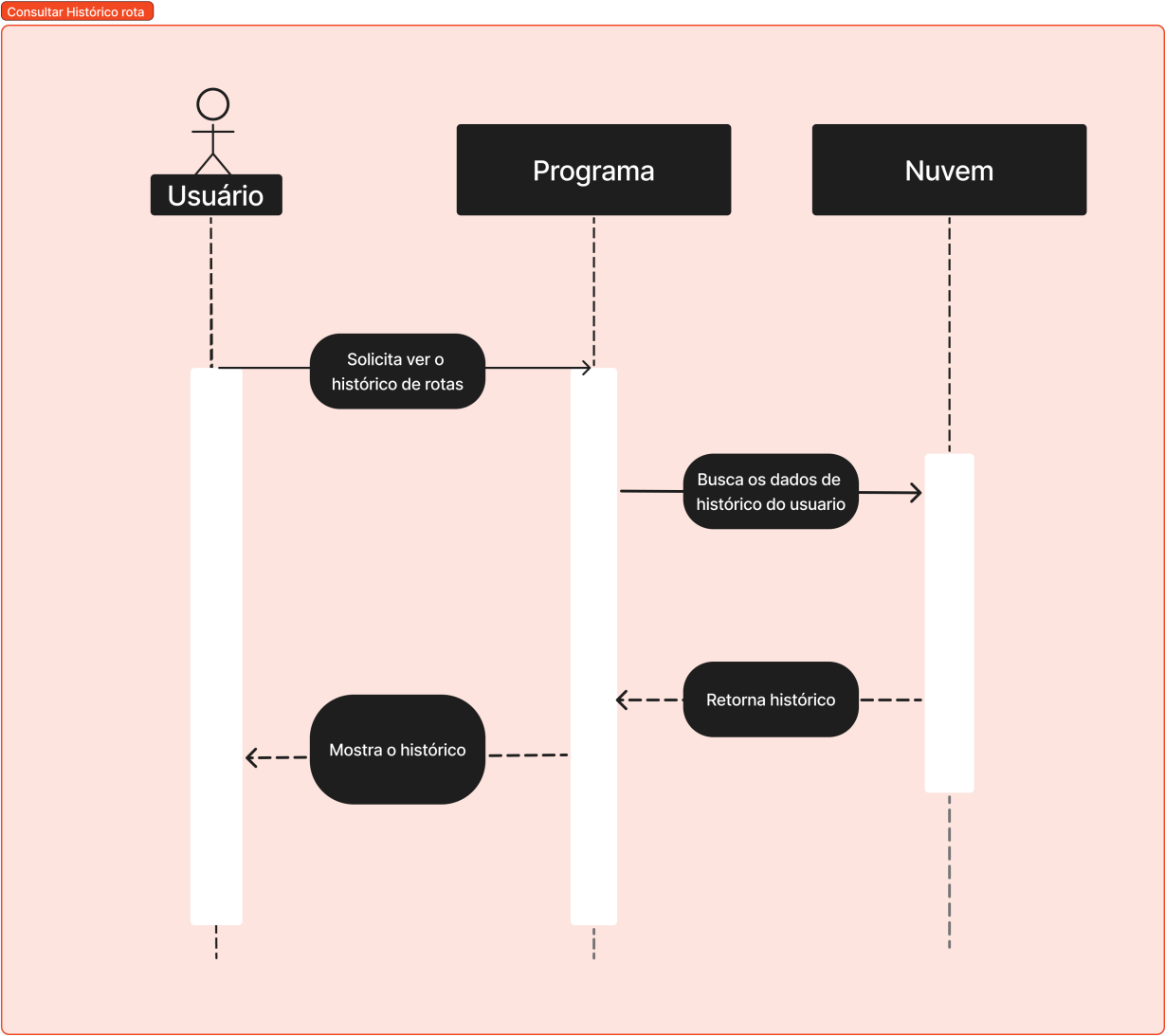


FIGURA 4

# Capítulo 6

## Descrição da arquitetura do sistema (modelo)

Para a parte de arquitetura do sistema, iremos utilizar uma arquitetura em camadas, o MVC. Escolhemos ela pela nossa familiaridade, além de ser de fácil aplicação, ter cada camada bem definida, o que facilita o desenvolvimento e manutenção do código.

## Ferramentas que serão utilizadas

O programa será uma página web, com o backend em Java com Spring Boot para integrar com o front em HTML, CSS e JavaScript. A parte dos protótipos e a dos diagramas foram feitos no Figma.

Para o CI/CD iremos utilizar o GitHub Action, com o seguinte Pipeline:

name: CarbonFlight CI/CD Pipeline

on:

push:

branches: [ "main" ] # roda quando fizer push na main

pull\_request:

branches: [ "main" ] # roda em PRs para a main

jobs:

build-test-deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout código

uses: actions/checkout@v4

- name: Configurar JDK 24

uses: actions/setup-java@v4

with:

distribution: 'temurin' # pode ser zulu, adopt, etc.

java-version: '24'

- name: Compilar e rodar testes

run: mvn clean verify --batch-mode

- name: Buildar pacote

run: mvn package -DskipTests

- name: Salvar artefato gerado

uses: actions/upload-artifact@v4

with:

name: meu-jar

path: target/\*.jar

- name: Deploy (exemplo simples via SCP)

if: github.ref == 'refs/heads/main'

run: |

echo "Fazendo deploy..."

scp -o StrictHostKeyChecking=no target/\*.jar \

\${{ secrets.SERVER\_USER }}@\${{ secrets.SERVER\_HOST }}:/caminho/do/

deploy/

## Capítulo 7

### **Desenvolvimento.**

Descrever o processo de construção do sistema proposto.

Descrever as tecnologias utilizadas para o desenvolvimento do sistema. Como o sistema foi modularizado. Indicar se alguma documentação poderia sofrer modificação.

Capítulo 2. O Projeto 5



## Capítulo 8

### Resultados.

Apresentar as características e funcionalidades da aplicação desenvolvida. Apresentar “prints” de telas das funcionalidades principais. Fazer uma análise crítica do sistema desenvolvido, apontando os pontos fortes e fracos. Incluir (novamente) o link do projeto no GitHub; não esquecer de subir os códigos da aplicação no repositório.

## **Capítulo 9**

### **Conclusão**

#### **Trabalhos Futuros**

Escrever a conclusão do projeto, revisando os pontos principais descritos até essa seção. Apontar quais melhorias no sistema poderiam ser exploradas em trabalhos futuros. Apontar trabalhos futuros que podem ser derivados do sistema proposto.