# Projeto BatsWorksMatriz Calculator

## Sumario

1 - Introdução - Pagina 3

2 - Colaboradores - Pagina 4

3 - Diagrama - Pagina 5

4 - Requisitos - Pagina 6

5 - Tecnologias Utilizadas - Pagina 8

6 - Descrição e Estrutura - Pagina 9

7 - Rweferencias - Pagina 11

## Introdução

A **Calculadora de Matrizes** é um aplicativo projetado para facilitar a realização de operações matemáticas com matrizes, oferecendo uma interface gráfica simples e intuitiva. Este software foi desenvolvido para atender às necessidades de estudantes, professores e profissionais que lidam com cálculo matricial no dia a dia, seja em disciplinas acadêmicas, áreas de engenharia ou ciência de dados.

O principal objetivo da aplicação é permitir a execução rápida e precisa de operações como soma, subtração, multiplicação e divisão de matrizes. Além disso, a ferramenta suporta diferentes tamanhos de matrizes, variando de 2x2 a 6x6, com a possibilidade de alterar dinamicamente o tamanho conforme necessário.

Por meio de recursos adicionais, como a validação de entradas, exibição detalhada de resultados e opções de copiar os resultados para a área de transferência, o software busca não apenas resolver cálculos, mas também melhorar a experiência de aprendizado e trabalho dos usuários.

Essa calculadora foi projetada com ênfase em eficiência, usabilidade e portabilidade, podendo ser executada em qualquer sistema operacional que suporte Java. É uma solução ideal para quem precisa de uma ferramenta prática e confiável para trabalhar com matrizes.

## Colaboradores

Felipe Batista da Silva\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_RA: 821244065

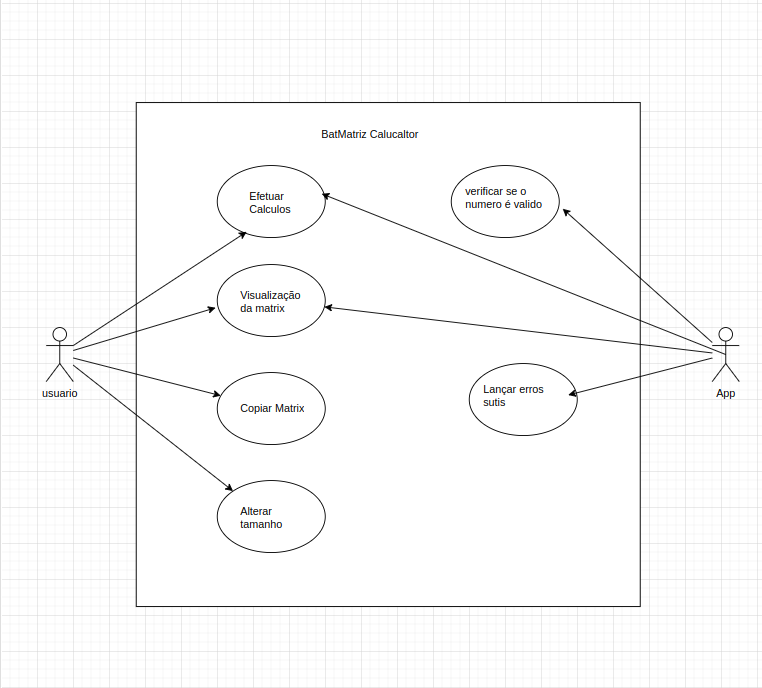
Vitor Henrique Fonseca da Silva\_\_\_\_\_\_\_\_RA: 823214255

Diogo Dagnino\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_RA: 823130326

Matheus Rossi\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_RA: 823134151

## Diagrama

Diagrama de Caso de uso



## Requisitos

### Requisitos Funcionais

1. **Operações Matemáticas:**

Somar duas matrizes de tamanho compatível.

Subtrair duas matrizes de tamanho compatível.

Multiplicar duas matrizes de tamanho compatível.

Dividir elementos das matrizes (se aplicável).

1. **Entrada e Validação de Dados:**

Permitir entrada de valores para todas as células das matrizes.

Validar que os valores inseridos são números válidos.

Exibir mensagens de erro para entradas inválidas.

1. **Tamanhos de Matrizes:**

Suportar diferentes tamanhos de matrizes (2x2 até 6x6).

Atualizar dinamicamente os campos de entrada ao mudar o tamanho.

1. **Limpeza e Reset:**

Botão para limpar todas as células das matrizes.

Resetar as matrizes para valores padrão (ex.: 0).

1. **Exibição de Resultados:**

Mostrar os resultados das operações em uma área dedicada (JTextArea).

Permitir copiar os resultados para a área de transferência.

1. **Persistência Temporária:**

Manter os valores inseridos ao realizar operações ou mudar de tamanho.

1. **Interface Gráfica:**

Separação visual clara entre matrizes e resultados.

Elementos interativos como botões e menus de fácil acesso.

### Requisitos Não Funcionais

1. **Desempenho:**

O cálculo das operações deve ser realizado em menos de 1 segundo para matrizes de até 6x6. A interface deve atualizar sem travamentos durante operações ou mudanças de tamanho.

1. **Usabilidade:**

A interface deve ser intuitiva e acessível para usuários leigos.

Elementos visuais como botões e campos de texto devem ser claramente identificáveis.

1. **Portabilidade:**

O aplicativo deve funcionar em sistemas operacionais Windows, Linux e macOS com suporte a Java

1. **Robustez:**

O sistema deve lidar com erros de entrada, como valores nulos ou inválidos, sem fechar insperadamente.

Mensagens de erro devem ser claras e direcionar o usuário para correções.

1. **Manutenibilidade:**

O código deve ser modular e seguir boas práticas de programação para facilitar aualizações futuras.

A lógica das operações deve ser separada da interface gráfica (ex.: usar classes como OerationListener).

1. **Escalabilidade:**

O sistema deve ser projetado para suportar a adição de novos tamanhos de matrizes ou operações no futuro.

1. **Segurança:**

Validar que as entradas são seguras, evitando ataques de injeção de código ou problemas com valores extremos.

1. **Estilo Visual:**

Seguir um padrão visual consistente com cores, fontes e estilos apropriados para uma ferramenta educacional ou acadêmica.

## Tecnologias Uilizadas

* Java 14 - 17 - 21
* Java Swing
* Maven 3.9
* IntellijIDEA
* Eclipse
* Draw.IO

### Descrição da Estrutura do Código

#### Raiz do Projeto

* **POM.XML:**

Arquivo de configuração do Maven, contendo as dependências do projeto, plugins, e configurações gerais de build e execução.

* **README.md:**

Documento que descreve o projeto, seu propósito, e instruções para configuração e execução.

#### src/main/java:

Diretório principal contendo os arquivos fonte do projeto. Segue o padrão de organização de pacotes utilizado pelo Maven.

##### Pacotes

1. **com.batsworks.matrix**

Pacote base que organiza todas as classes relacionadas ao projeto.

1. **enums**

Contém classes enumeradas que definem constantes ou tipos fixos utilizados no projeto.

Exemplo: Tipos de operações suportadas na calculadora de matrizes (soma, subtração, etc.).

1. **Factory**

Função: Facilitar a criação de objetos complexos ou dependentes, promovendo modularidade e reutilização de código.

1. **Frame**

Contém as classes que gerenciam a interface gráfica (GUI) da aplicação.

Exemplo: Frames ou painéis que exibem as matrizes, botões, e resultados.

1. **listeners**

Classes que implementam os *listeners* de eventos (como cliques em botões ou alterações nos campos de texto).

Função: Gerenciar interações do usuário com a interface gráfica.

1. **Styles**

Possui classes para estilização da interface gráfica, como configurações de fontes, cores, e tamanhos. Função: Padronizar a aparência do software.

1. **utils**

Classes utilitárias com métodos auxiliares ou ferramentas genéricas.

Exemplo: Validação de entradas, manipulação de matrizes, ou cálculos específicos.

1. **MainApplication**

Classe principal do projeto, responsável por inicializar e executar a aplicação.

Contém o método main.

#### src/main/resources

Diretório para arquivos de recursos estáticos.

Exemplo: Arquivos de configuração, imagens, ou propriedades utilizadas no projeto.

#### target

* Diretório gerado automaticamente pelo Maven durante o processo de build.
* Contém os arquivos compilados e o artefato final do projeto (como um .jar ou .war).

## Referencias

**Repositorio**: <https://github.com/Felipeb26/A3-Projeto-Matriz>

### Outras referencias

**Java Swing:**

<https://www.youtube.com/watch?v=4BRUmU-ETRk>

**Calculo de Matrizes com Java:**

<https://www.youtube.com/watch?v=aLmr1waxc10>

[https://www.youtube.com/watch?v=fThDTZNaxNA&list=PLGsRpMr9gXDcxU55vBG3mYAW5-N5W9yHv](https://www.youtube.com/watch?v=aLmr1waxc10)