

{ DEV }




# Projeto Gerenciament o de Estoque





## Visão Geral do Projeto


- **Descrição:** O projeto Gerenciamento de Estoque A3 é um sistema desenvolvido para otimizar o controle e a gestão de estoques em empresas de médio e grande porte. Ele oferece uma plataforma integrada para gerenciamento eficiente de produtos, pedidos e inventários.
  - **Objetivos:** Simplificar o processo de gestão de estoque, aumentar a eficiência operacional e reduzir custos associados ao controle de inventário.
- 



## Arquitetura e Tecnologias Utilizadas


Arquitetura: Utiliza uma arquitetura baseada em microsserviços para escalabilidade e flexibilidade. Componentes principais incluem backend em Java Spring Boot, e banco de dados PostgreSQL

### Tecnologias Utilizadas:

- Back-end: Java Spring Boot, Hibernate
  - Banco de Dados: PostgreSQL
  - Ferramentas de Integração Contínua: GitHub Actions para CI/CD
- 




## Fluxo de Desenvolvimento

- Integração Contínua (CI/CD): Implementamos CI/CD utilizando GitHub Actions para automatizar a compilação, testes e implantação do código.
  - Testes: Realizamos testes unitários e de integração para garantir a qualidade do código e a estabilidade do sistema.
- 



## Fluxo de Trabalho do CI/CD


- Passos do Pipeline:
  - Build com Maven: Compilação e empacotamento do código utilizando Maven.
  - Testes Unitários: Execução de testes unitários para validar componentes individuais do sistema.
  - Testes de Integração: Verificação de integridade e interoperabilidade entre os módulos do sistema.
  - Análise Estática: Análise estática de código para identificação de potenciais problemas e violações de boas práticas.
  - Implantação Automatizada: Implantação automática no ambiente de produção ou homologação, dependendo da branch.
- 



## Build com Maven:

- Esta etapa é responsável por compilar todo o código do projeto e empacotá-lo, garantindo que todas as dependências estejam corretamente resolvidas e que o projeto possa ser construído sem erros.

## Testes Unitários:


- Aqui, são executados testes automatizados que verificam se cada componente individual do sistema funciona conforme o esperado. Isso ajuda a garantir que partes específicas do código estejam corretas e que novas alterações não quebrem funcionalidades existentes.
- 



## **Upload de Relatórios de Cobertura (JaCoCo):**

- Após a execução dos testes unitários, o JaCoCo é usado para gerar relatórios detalhados sobre a cobertura do código pelos testes. Esses relatórios são então carregados como artefatos, permitindo uma análise posterior da cobertura de código.

## **Testes de Integração (Release):**

- Os testes de integração são realizados para validar a interação entre os diferentes componentes do sistema, garantindo que todos os módulos funcionem corretamente em conjunto. Essa etapa é crucial antes da implantação em ambientes de produção.
- 




## **Deploy da Aplicação (Release):**

Após a conclusão dos testes e análises, a aplicação é implantada automaticamente em um ambiente específico. Isso permite uma entrega contínua e automatizada de novas versões do software.

## **Análise de Código Estática (SonarQube):**

Utilizando o SonarQube, são realizadas análises estáticas detalhadas do código-fonte para identificar possíveis problemas de segurança, bugs e vulnerabilidades, além de garantir a conformidade com as melhores práticas de codificação.





## Configuração do CI/CD:


```
1  name: CI/CD
2
3  on:
4    push:
5      branches:
6        - "*"
7    pull_request:
8      branches:
9        - master
10     - release
11
12  jobs:
13    build:
14      runs-on: ubuntu-latest
15
16      steps:
17        - name: Checkout repository
18          uses: actions/checkout@v3
19
20        - name: Set up JDK
21          uses: actions/setup-java@v3
22          with:
23            distribution: 'adopt'
24            java-version: '17'
25
26        - name: Build with Maven
27          run: mvn clean install
28
```

```
29  test-unit:
30    runs-on: ubuntu-latest
31
32    steps:
33      - name: Checkout repository
34        uses: actions/checkout@v3
35
36      - name: Set up JDK
37        uses: actions/setup-java@v3
38        with:
39          distribution: 'adopt'
40          java-version: '17'
41
42      - name: Run unit tests
43        run: mvn test -B
44
45      - name: Generate JaCoCo report
46        run: mvn jacoco:report
47
48      - name: Publish JaCoCo report
49        uses: actions/upload-artifact@v2
50        with:
51          name: jacoco-report
52          path: target/site/jacoco/
53
54  upload-ftp:
55    needs: test-unit
56    runs-on: ubuntu-latest
57
58    steps:
59      - name: Checkout repository
60        uses: actions/checkout@v3
61
62      - name: Create JaCoCo report directory
63        run: mkdir -p target/site/jacoco/
64
65      - name: Upload JaCoCo report via FTP
66        uses: SamKirkland/FTP-Deploy-Action@4.1.0
67        with:
68          server: joaobsjunior.com.br
69          username: aluno-ftp
70          password: a1b2c3d4@
71          local-dir: target/site/jacoco/
72          server-dir: /public_html/Gerenciamento-de-Estoque-A3-master/
73
```

```
74  test-integration:
75    if: github.ref == 'refs/heads/release'
76    runs-on: ubuntu-latest
77
78    steps:
79      - name: Checkout repository
80        uses: actions/checkout@v3
81
82      - name: Set up JDK
83        uses: actions/setup-java@v3
84        with:
85          distribution: 'adopt'
86          java-version: '17'
87
88      - name: Run integration tests
89        run: mvn verify -B
90
91      - name: Generate JaCoCo report for integration tests
92        run: mvn jacoco:report
93
94      - name: Publish JaCoCo report for integration tests
95        uses: actions/upload-artifact@v2
96        with:
97          name: jacoco-report-integration
98          path: target/site/jacoco/
99
100  release:
101    if: github.ref == 'refs/heads/release'
102    needs: [upload-ftp, test-integration]
103    runs-on: ubuntu-latest
104
105    steps:
106      - name: Deploy Application
107        run: echo "Deploying application..."
108
109  sonarqube:
110    needs: test-unit
111    runs-on: ubuntu-latest
112
113    steps:
114      - name: Checkout repository
115        uses: actions/checkout@v3
116
117      - name: Set up JDK
118        uses: actions/setup-java@v3
119        with:
120          distribution: 'adopt'
121          java-version: '17'
122
123      - name: SonarQube Scan
124        env:
125          SONAR_TOKEN: ${ secrets.SONAR_TOKEN }
126        run: mvn sonar:sonar -Dsonar.projectKey=ProjetoA3 -Dsonar.organization=grbadas -Dsonar.host.url=https://sonarcloud.io -Dsonar.login=${ secrets.SONAR_TOKEN }
```



## Testes Implementados

- Tipos de Testes:
  - Testes Unitários: Verificam a funcionalidade de unidades isoladas de código para garantir comportamento esperado.
  - Testes de Integração: Validam a interação entre diferentes componentes do sistema.
  - Ferramentas Utilizadas: JUnit para testes unitários em Java, ferramentas específicas para testes de integração conforme necessário.
- 

## Exemplos de testes:

```
grbadas
@Test
void testCreateProduto() {
    Produto produto = new Produto();
    produto.setNome("Produto 1");
    produto.setQuantidade(5);

    produto = produtoRepositorio.save(produto);
    assertNotNull(produto.getId());
}

grbadas
@Test
void testReadProduto() {
    Produto produtoSalvo = produtoRepositorio.save(new Produto( nome: "Produto 1", quantidade: 5));

    Optional<Produto> produtoSalvoOPT = produtoRepositorio.findById(produtoSalvo.getId());
    assertTrue(produtoSalvoOPT.isPresent());
}

grbadas
```

```
grbadas
@Test
public void testaddProduto() {

    Produto produtoParaAdicionar = new Produto( nome: "Nome do produto", quantidade: 10);
    when(produtoRepositorio.save(produtoParaAdicionar)).thenReturn(produtoParaAdicionar);

    Produto produtoAdicionado = produtoService.addProduto(produtoParaAdicionar);

    assertNotNull(produtoAdicionado);
    assertEquals( expected: "Nome do produto", produtoAdicionado.getNome());
    assertEquals( expected: 10, produtoAdicionado.getQuantidade());
    verify(produtoRepositorio, times( wantedNumberOfInvocations: 1)).save(produtoParaAdicionar);
}

grbadas
@Test
public void testAddProdutoComNomeNulo() {
    Produto produto = new Produto( nome: null, quantidade: 10);

    assertThrows(IllegalArgumentException.class, () -> produtoService.addProduto(produto));
}
```

# Monitoramento e Relatórios

- Monitoramento Contínuo: Utilização de ferramentas de monitoramento para acompanhar a saúde do sistema após implantações.
- Relatórios de Cobertura: Relatórios de cobertura de testes (JaCoCo) para avaliar a eficácia dos testes implementados.

gerenciamento-de-estoque												
gerenciamento-de-estoque												
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
<a href="#">badas.dev.gerenciamento.de.estoque.model</a>	<div><div></div></div>	18%	<div><div></div></div>	0%	18	25	2	11	6	13	0	1
<a href="#">badas.dev.gerenciamento.de.estoque.controller</a>	<div><div></div></div>	23%		n/a	1	2	2	3	1	2	0	1
<a href="#">badas.dev.gerenciamento.de.estoque</a>	<div><div></div></div>	37%		n/a	1	2	2	3	1	2	0	1
<a href="#">badas.dev.gerenciamento.de.estoque.service</a>	<div><div></div></div>	100%	<div><div></div></div>	83%	1	6	0	7	0	3	0	1
Total	145 of 213	31%	25 of 30	16%	21	35	6	24	8	20	0	4

# Exemplo de Pipeline de CI/CD

← CI/CD

✓ Add files via upload #98

Re-run all jobs ⋮

Summary

Jobs

✓ build

✓ test-unit

🕒 test-integration

✓ upload-ftp

✓ sonarqube

🕒 release

Run details

🕒 Usage

📄 Workflow file

Triggered via push 9 minutes ago

GRBadas pushed · b49c5d3 master

Success

Total duration  
1m 39s

Artifacts  
1

actions.yml

on: push

✓ test-unit40s

🕒 test-integration0s

✓ build30s

✓ upload-ftp7s

✓ sonarqube43s

🕒 release0s

Annotations

11 warnings

⚠️ build

Node.js 16 actions are deprecated. Please update the following actions to use Node.js 20: actions/checkout@v3, actions/setup-java@v3. For more information see: <https://github.blog/changelog/2023-09-22-gi...>  
[Show more](#)

⚠️ build

The process `'/usr/bin/git'` failed with exit code 128



## Equipe:

GABRIEL REIS BADARÓ 12722116786

ELAINE SANTANA GONZAGA 12722131402

GUILHERME GÓES XAVIER GONÇALVES 12722131927

HERBERT LOPES SANTANA DA GUIA 1272211389

SABRINA FILGUEIRAS ALVES RAIOL 1272217396

