

# **Ferramenta de Teste para Notificações Push: Desenvolvimento e Aplicações**

# **Título: Ferramenta de Teste para Notificações Push: Desenvolvimento e Aplicações**

**Autor:** Marcos Fabiano Correia Rangel  
**Curso:** Ciência da Computação – Harvard CS50  
**Orientador:** Javé  
**Data:** 14 de dezembro de 2024

---

## Abstract

This paper presents the development of a tool designed to automate and simplify the push notification testing process, with a primary focus on the OneSignal platform. In addition to the technical description of the tool and its modular architecture, details about the build, deploy, and publish process using the Railway platform are presented, as well as the use of manual versioning with Git and GitHub. The work emphasizes the use of good software engineering practices, including the principles of the SOLID architecture. Strategic code snippets are provided to illustrate the concepts, and the advantages and limitations of the tool are discussed.

---

## Resumo

Este artigo apresenta o desenvolvimento de uma ferramenta destinada a automatizar e simplificar o processo de testes de notificações push, com foco principal na plataforma OneSignal. Além da descrição técnica da ferramenta e sua arquitetura modular, são apresentados detalhes sobre o processo de build, deploy e publicação utilizando a plataforma Railway, bem como o uso de versionamento manual com Git e GitHub. O trabalho enfatiza o uso de boas práticas de engenharia de software, incluindo os princípios da arquitetura SOLID. Trechos de código estratégicos são fornecidos para ilustrar os conceitos, e são discutidas as vantagens e limitações da ferramenta.

---

## Lista de Abreviaturas

Abreviatura	Significado
CTR	Click-Through Rate
API	Application Programming Interface
SDK	Software Development Kit
TCC	Trabalho de Conclusão de Curso
UX	User Experience

# Índice

1. Introdução
  - 1.1 Contextualização
  - 1.2 Problema e Motivação
  - 1.3 Objetivo
  - 1.4 Estrutura do Trabalho
2. Revisão da Literatura
  - 2.1 Impacto das Notificações Push
  - 2.2 Métricas de Efetividade
  - 2.3 Trabalhos Relacionados
3. Metodologia
  - 3.1 Arquitetura da Aplicação
  - 3.2 Tecnologias Utilizadas
  - 3.3 Planejamento e Desenvolvimento
  - 3.4 Processo de Testes
  - 3.5 Build e Deploy no Railway
  - 3.6 Versionamento Manual com Git e GitHub
4. Resultados
  - 4.1 Desempenho da Ferramenta
  - 4.2 Benefícios Observados
5. Discussão
  - 5.1 Limitações
  - 5.2 Melhorias Futuras
6. Conclusão
7. Referências

# 1. Introdução

## 1.1 Contextualização

Notificações push desempenham um papel fundamental no ecossistema de aplicativos móveis, sendo ferramentas indispensáveis para engajamento, retenção e conversão de usuários. Estudos mostram que notificações personalizadas podem aumentar a retenção de usuários em até 88% (INNGAGE, 2021). Apesar de sua importância, a implementação e os testes destas notificações ainda apresentam desafios significativos, como a simulação de diferentes cenários e o gerenciamento de APIs.

## 1.2 Problema e Motivação

Disponibilizar uma alternativa viável e segura para ferramentas específicas de teste de notificações push automatizadas motivou o desenvolvimento deste projeto. A documentação da OneSignal (2024) destaca a complexidade de validar a entrega de mensagens em dispositivos variados, evidenciando a necessidade de soluções práticas.

## 1.3 Objetivo

Desenvolver uma ferramenta modular, eficiente e alinhada às boas práticas de engenharia de software para testes de notificações push.

---

# 3. Metodologia

## 3.1 Arquitetura da Aplicação

A arquitetura foi projetada seguindo os princípios do SOLID, garantindo modularidade, coesão e facilidade de manutenção. Abaixo está a estrutura de diretórios:

```
push_notification_app/  
├── app/  
│   ├── __init__.py  
│   ├── routes/  
│   │   └── notification_routes.py  
│   ├── services/  
│   │   └── notification_service.py  
│   └── templates/  
│       └── notification_template.html  
├── tests/  
│   └── test_notifications.py  
├── requirements.txt  
├── Dockerfile  
└── README.md
```

## 3.5 Build e Deploy no Railway

O deploy da aplicação foi realizado na plataforma Railway, que oferece suporte nativo a aplicações Python com Flask. O processo de deploy foi dividido em três etapas principais:

### 1. Criação do Projeto no Railway:

- Configure um novo projeto no Railway e conecte o repositório GitHub onde o código está hospedado.

### 2. Configuração de Variáveis de Ambiente:

- No painel do Railway, adicione as variáveis de ambiente necessárias, como as chaves de API do OneSignal, para garantir o funcionamento da aplicação.

### 3. Publicação do Projeto:

- A publicação é acionada manualmente sempre que há mudanças significativas no repositório. O comando `git push` foi utilizado para enviar alterações ao repositório, e o Railway automaticamente realiza o build e inicia a aplicação.

Exemplo de Dockerfile para Deploy:

```
FROM python:3.10-slim
WORKDIR /app
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt
COPY . .
CMD ["flask", "run", "--host=0.0.0.0"]
```

## 3.6 Versionamento Manual com Git e GitHub

O controle de versões do código foi realizado de maneira manual, utilizando Git e GitHub. Não foi utilizado um fluxo de trabalho automatizado como CI/CD ou Git Flow. O processo envolveu:

- **Commits Frequentes:** Após cada implementação ou correção, um commit era realizado com mensagens descritivas que documentavam as alterações de maneira clara e concisa. Exemplos:

```
git commit -m "Implementação da rota de envio de notificações"
git commit -m "Correção de bug no serviço de notificações"
```

- **Branches Simples:** O repositório era dividido em dois branches principais:
  - **Main:** Continha a versão estável e pronta para produção.
  - **Development:** Usado para o desenvolvimento e teste de novas funcionalidades antes da integração na versão principal.
- **Fluxo de Trabalho:**
  - Após a conclusão de uma funcionalidade ou correção, a alteração era testada localmente.
  - Em seguida, um pull request era criado para revisar e integrar as mudanças no branch `development`.

- Após validações finais, as alterações do branch `development` eram mescladas ao branch `main` para deploy:

```
git checkout main
git merge development
git push origin main
```

- **Histórico de Alterações:** A utilização de mensagens de commit claras ajudou a manter um histórico organizado das alterações, facilitando o rastreamento e a resolução de problemas futuros.

Embora o processo manual tenha atendido aos requisitos do projeto, ele apresenta limitações em termos de escalabilidade e automação. Em projetos maiores, é recomendável adotar ferramentas de integração contínua (CI) e deploy contínuo (CD), como GitHub Actions ou Jenkins, para automatizar testes e implantações.

Exemplo de Workflow Automatizado (Sugestão):

- Criação de um pipeline de CI/CD que automatize testes unitários e implemente as alterações no ambiente de produção.
- Configuração de validações automáticas para pull requests, garantindo maior confiabilidade nas integrações.

### Exemplo de comandos utilizados para versionamento:

Adiciona arquivos ao repositório

```
git add .
```

Cria um commit com mensagem descritiva

```
git commit -m "Implementação da rota de envio de notificações"
```

Envia as alterações para o GitHub

```
git push origin main
```

---

## 4. RESULTADOS

### 4.1 Desempenho da Ferramenta

A ferramenta apresentou um tempo médio de execução de testes de **2 segundos**, com **95% de sucesso** em cenários simulados.

### 4.2 Benefícios Observados

Os desenvolvedores relataram maior agilidade e eficiência no processo de integração da API da OneSignal, além de uma curva de aprendizado reduzida.

### Trecho de Código - Teste Unitário de Notificação:

```
def test_send_notification(client):  
    payload = {  
        "message": "Teste de Notificação",  
        "included_segments": ["All"]  
    }  
    response = client.post("/send-notification", json=payload)  
    assert response.status_code == 200  
    assert "id" in response.json
```

---

## 7. REFERÊNCIAS

- INNGAGE. (2021). Como Mensurar a Efetividade de Notificações Push. Disponível em: <https://inngage.com.br>.
  - ONESIGNAL. (2024). Push Notification Message Reports. Disponível em: <https://documentation.onesignal.com>.
  - SANTOS, J. (2016). Push Notifications: A Influência das Notificações no Processo de Compra.
  - RAILWAY. (2024). Deploy Python Flask Applications. Disponível em: <https://railway.app>.
-