

# AUMENTE A SEGURANÇA DE SUAS APLICAÇÕES COM SOFTWARE LIVRE

Samuel Gonçalves

“*O espírito humano precisa  
prevalecer sobre a tecnologia.*”

*Albert Einstein*





# Samuel Gonçalves Pereira

- Consultor de Tecnologia nas áreas  
**Segurança Ofensiva e DevSecOps**
- Mais de 10 anos de experiência 💻
- "3k++" alunos ensinados no 🌎
- Músico, Contista, YouTuber e Podcaster
- Apaixonado por Software Livre ❤️



# Vamos nos conectar?

É só escanear ou acessar o link:  
<https://beacons.ai/sgoncalves>



**SORTEIO NO FIM DA PALESTRA!**

**Inscreva-se e participe!**

Precisamos falar de  
SEGURANÇA!



# Princípios da segurança

Também conhecidos por **CID**, são:

**Confiabilidade, Integridade e Disponibilidade.**

Se algum desses princípios falhar, teremos uma falha de segurança em nossa aplicação.

# Segurança do início ao fim!



É importante trazer a segurança para o projeto desde o início do processo. Esta abordagem é conhecida como **Secure Shift Left**.

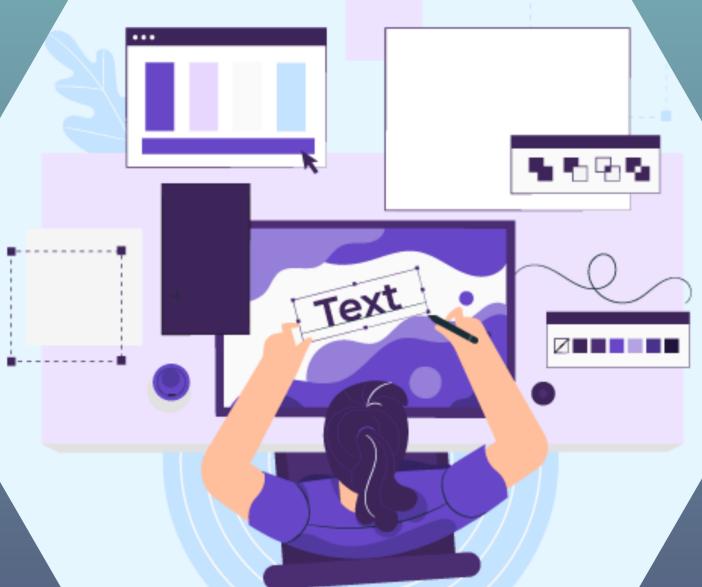
# 1º Fase: *Requisitos*

Nesta fase vamos nos preocupar com os princípios da segurança em cada um dos requisitos levantados, nos atentando ao processo de desenvolvimento seguro.



## 2º Fase: *Design*

Vamos utilizar o **Threat Modelling**, ou **Modelagem de Ameaças**. Levantaremos pontos como:



- Haverá interação com aplicação externa?
- Esta aplicação externa, possui alguma vulnerabilidade?
- Qual versão do banco de dados será utilizado?
- Qual tipo de autenticação utilizada pela API?

# 3º Fase: *Develop*

O foco é: evitar vulnerabilidades, para isso podemos usar recursos como a **OWASP Top 10**, a **OWASP API e Code Review Guide**.

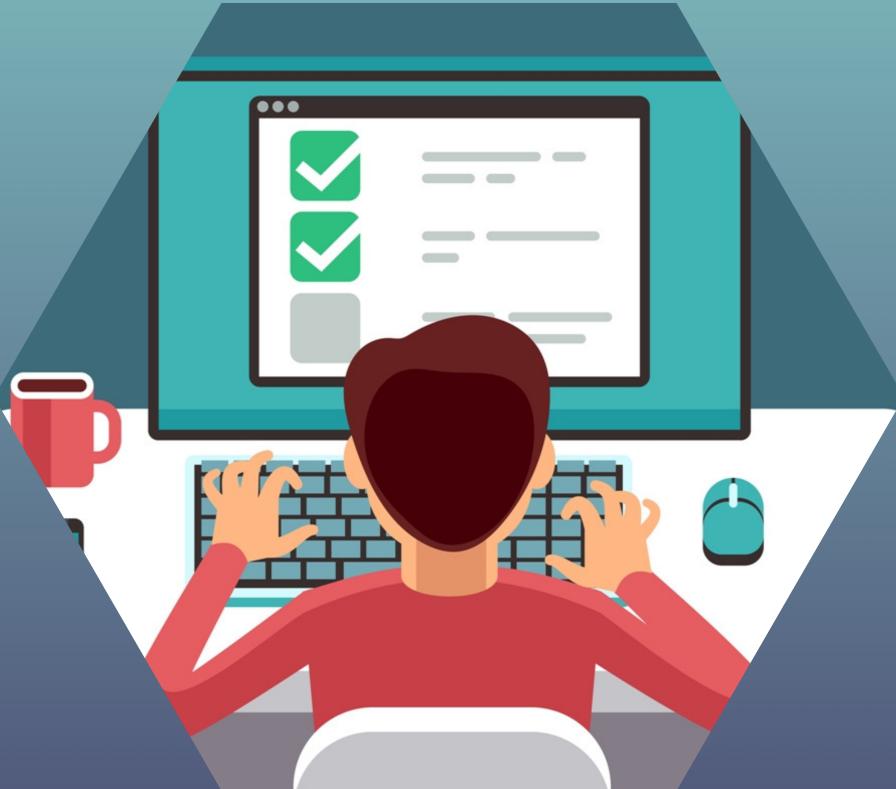
Vamos validar *inputs*, verificar de nossas APIs estão usando autenticação, se os métodos utilizados estão corretos...

O uso de scanners de código diretamente na IDE, como o SonarLint, é recomendado.



## 4° Fase: Test

Esta etapa contempla o **Code Review**, que visa encontrar vulnerabilidades no código fonte da aplicação, bem como realizar o **SAST** (Teste estático), com o mesmo objetivo. Existem várias ferramentas opensource para esta etapa, sempre baseadas em linguagem de programação.



# Ferramentas para SAST

Ferramenta	Linguagem	Link para acesso
Brakeman	Ruby	<a href="https://brakemanscanner.org/">https://brakemanscanner.org/</a>
SonarQube	27 linguagens	<a href="https://www.sonarqube.org/">https://www.sonarqube.org/</a>
Bandit	Python	<a href="https://pypi.org/project/bandit/">https://pypi.org/project/bandit/</a>
Horusec	Várias linguagens	<a href="https://horusec.io/">https://horusec.io/</a>
CPPCheck	C++	<a href="http://cppcheck.sourceforge.net/">http://cppcheck.sourceforge.net/</a>
Graudit	Groovy	<a href="https://github.com/wireghoul/graudit">https://github.com/wireghoul/graudit</a>
Dependency Check (SCA)	Dependências de Código	<a href="https://owasp.org/www-project-dependency-check/">https://owasp.org/www-project-dependency-check/</a>

## 5° Fase: *Deploy*

Antes do deploy é crucial fazer um **Pentest**, ou mesmo o **DAST (Teste Dinâmico)**. O objetivo é estressar a aplicação ao máximo para descobrir falhas que não foram vistas nas etapas anteriores.

Outro ponto importante é o **Hardening** das instâncias que executam a aplicação.



# Ferramentas para DAST

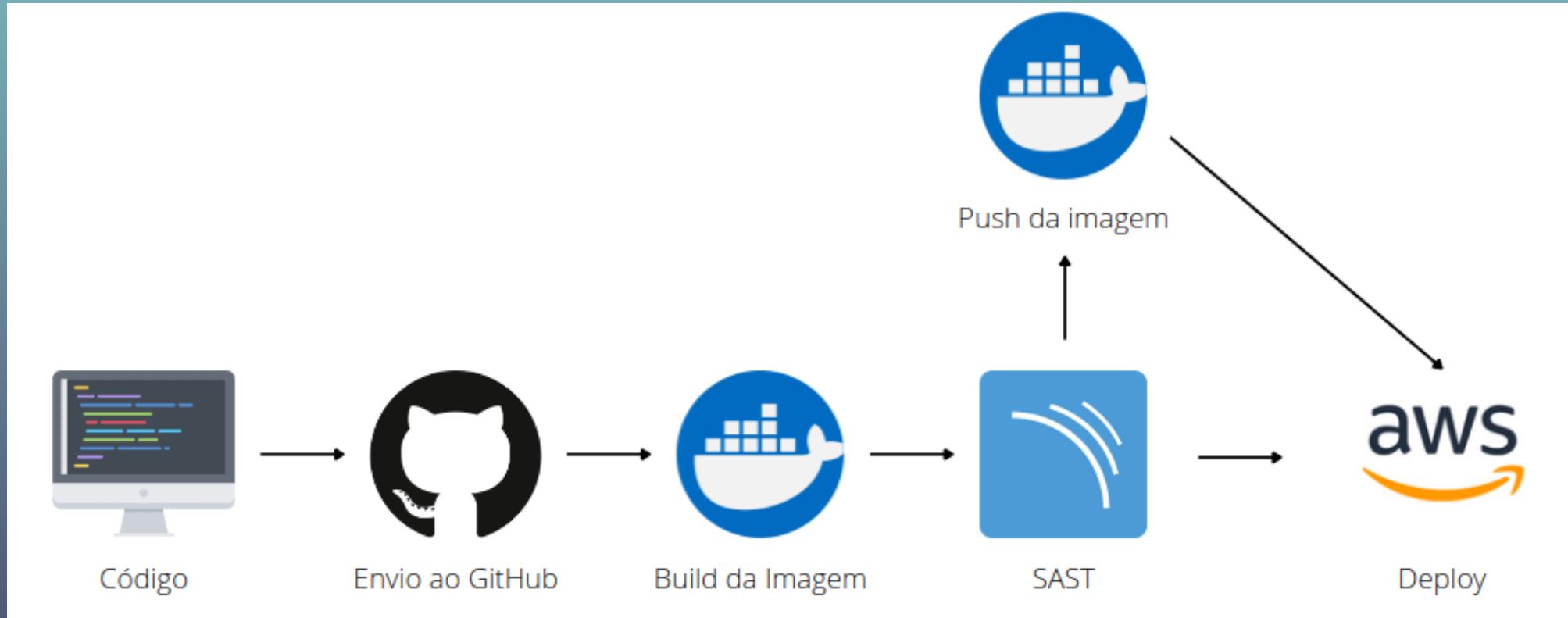
Ferramenta	Link para acesso
OWASP ZAP	<a href="https://owasp.org/www-project-zap/">https://owasp.org/www-project-zap/</a>
Arachni	<a href="https://www.arachni-scanner.com/">https://www.arachni-scanner.com/</a>
SQLmap	<a href="http://sqlmap.org/">http://sqlmap.org/</a>
Gauntlet	<a href="http://gauntlet.org/">http://gauntlet.org/</a>
BDD Security	<a href="https://github.com/iriusrisk/bdd-security">https://github.com/iriusrisk/bdd-security</a>
Nikto	<a href="https://github.com/sullo/nikto">https://github.com/sullo/nikto</a>
Golismero	<a href="https://github.com/golismero/golismero">https://github.com/golismero/golismero</a>

# Integração Contínua



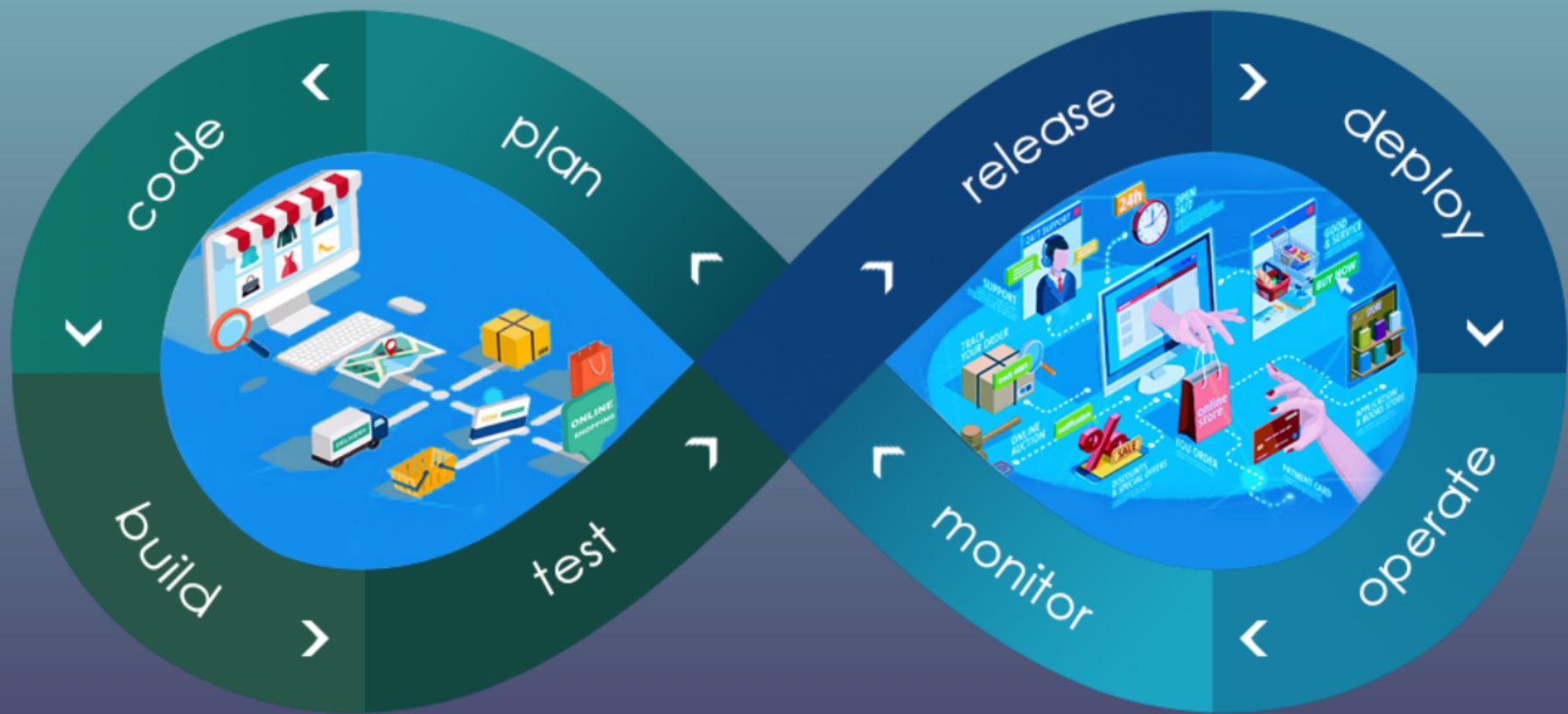
**CI** → Integração contínua está ligada ao processo de automação.

# Deploy Contínuo



**CD** → Deploy contínuo está ligado ao lançamento automático em ambiente produtivo.

# DevSecOps





# Dúvidas?

# Obrigado!

Vamos nos conectar?

- **Site:** [sgoncalves.tec.br](http://sgoncalves.tec.br)
- **E-mail:** [samuel@sgoncalves.tec.br](mailto:samuel@sgoncalves.tec.br)
- **Linkedin:** [linkedin.com/in/samuelgoncalvespereira/](https://linkedin.com/in/samuelgoncalvespereira/)
- **Telegram:** [t.me/Samuel\\_gp](https://t.me/Samuel_gp)
- **Todas as redes:** <https://beacons.ai/sgoncalves>

# Fontes Bibliográficas

<https://promovesolucoes.com/devsecops-seguranca-continua-lgpd/>

<https://www.redhat.com/pt-br/topics/devops/what-is-devsecops>

<https://www.ibm.com/br-pt/cloud/learn/devsecops>

<https://promovesolucoes.com/devsecops-seguranca-continua-lgpd/>

<https://4linux.com.br/cursos/treinamento/devsecops-seguranca-em-infraestrutura-e-desenvolvimento-agil/>

<https://blog.4linux.com.br/devsecops-implementacao-em-6-passos/>

<https://blog.gitguardian.com/security-tools-shift-left/>

<https://michelleamesquita.medium.com/entendendo-o-ciclo-de-vida-de-desenvolvimento-de-software-seguro-ssdlc-ccc173f583de>