ARQUITETURA PARA AUTOMAÇÃO DE TESTES

PATTERNS E BOAS PRÁTICAS

#THEDEVCONF

SÃO PAULO 2020

SOBRE

BRUNO LUSA

Porto Alegre - RS

10 anos de TI

QA no Sicredi

/brunolusa

/brunolusa

AGENDA

- CENÁRIO DO CAOS
- APRESENTAÇÃO DA ARQUITETURA
- ESTRATÉGIA DE TESTE
- DESIGN PATTERN
- HANDS ON!
- EVITANDO O CAOS
- ENCERRAMENTO

CENÁRIO DO CAOS

CARACTERÍSTICAS DE PROJETOS DESORGANIZADOS

- Forte acoplamento;
- Projeto com muitas dependências;
- Testes dependentes entre si;
- Sem foco no produto;
- Sem arquitetura definida;
- Ausência de boas práticas;
- Alta curva de aprendizado;
- Monolitos;
- Não executam de forma fácil;
- Manutenibilidade complexa;
- Não escalam.

APRESENTAÇÃO DA ARQUITETURA

API Rest

- Aplicação Springboot em Java;
- Controller: /carros;
- Verbos: *GET* *POST* *PUT* *DELETE*.

Client Project

- *Biblioteca Java;*
- *Abstrai as chamadas para os endpoints da API;*
- *Pode ser utilizada por vários projetos.*

Integration Test Project

- *Projeto de teste automatizado;*
- *Utiliza a dependência client Project;*
- *Realiza os testes relacionados ao projeto;*
- *Utiliza algumas dependências comuns.*

Front-end Project

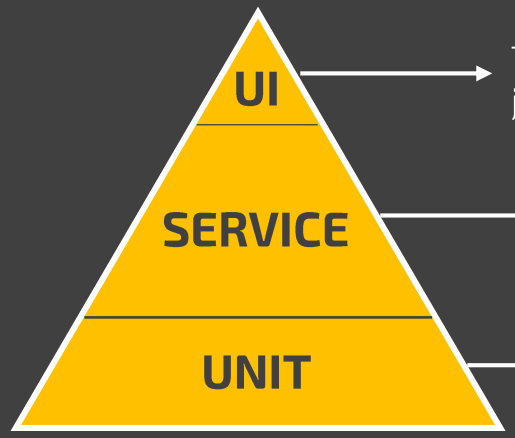
- *React JS Front-end;*
- *Utiliza a dependência Axios;*
- *Consome a aplicação API-Carros.*

E2E Project

- *Projeto de teste automatizado;*
- *Teste das principais funcionalidades de forma E2E;*
- *Utiliza as dependências Selenium e TesteNG.*

ESTRATÉGIA DE TESTE

PIRÂMIDE



Testes automatizados das principais jornadas de usuário de forma E2E.

Nesta camada concentra-se a maior parte dos testes, cenários positivos e fluxos de exceção.

Teste dos componentes e suas integrações adicionando validações de negócio.

TECNOLOGIAS

TESTE DE API

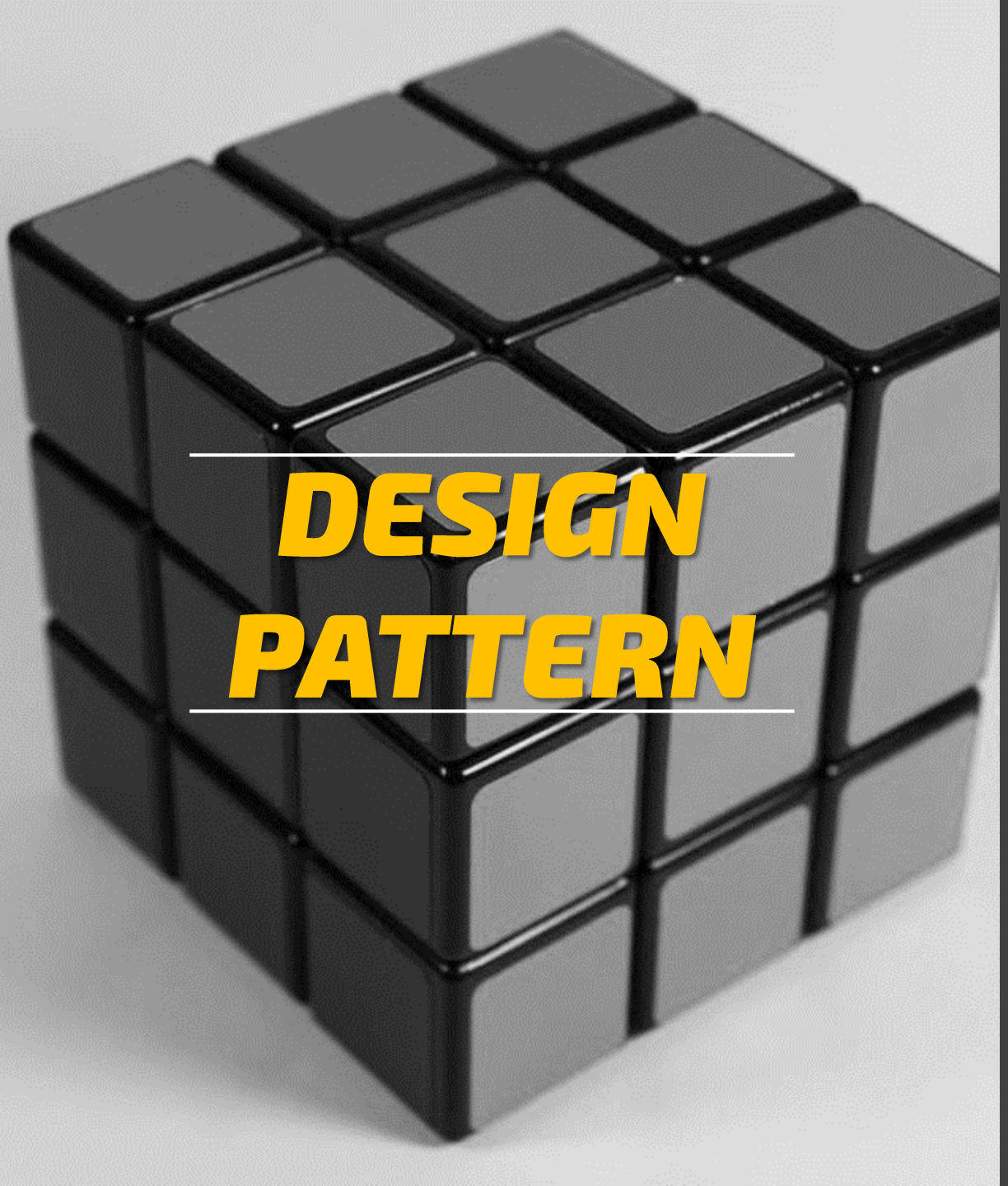
- ✓ Java
- ✓ Maven
- ✓ RestAssured
- ✓ TestNG
- ✓ ExtentReport
- ✓ GitLabCI

TESTE WEB

- ✓ Java
- ✓ Maven
- ✓ Selenium Web Driver
- ✓ Web Driver Manager
- ✓ TestNG
- ✓ ExtentReport
- ✓ GitLabCI

O QUALITY ENGINEER DEVE:

- Possibilitar a alteração do ambiente alvo dos testes via linha de comando ou properties;
- Ter conhecimento referente aos testes unitários desenvolvidos na aplicação;
- Viabilizar a execução em ambiente CI (Pipeline).



DESIGN PATTERN

CARACTERÍSTICAS

- Apresenta solução reproduzível de um problema;
- Geralmente problemas clássicos possuem um Pattern;
- Não traz a solução final;
- Induz à escrita com boas práticas;
- Ressalta os princípios da Orientação a Objetos.

BENEFÍCIOS

- Gera ganho de produtividade;
- Colabora para a organização do projeto;
- Contribui para a manutenibilidade do projeto;
- Facilita as discussões técnicas.

1 FACADE

CARACTERÍSTICAS

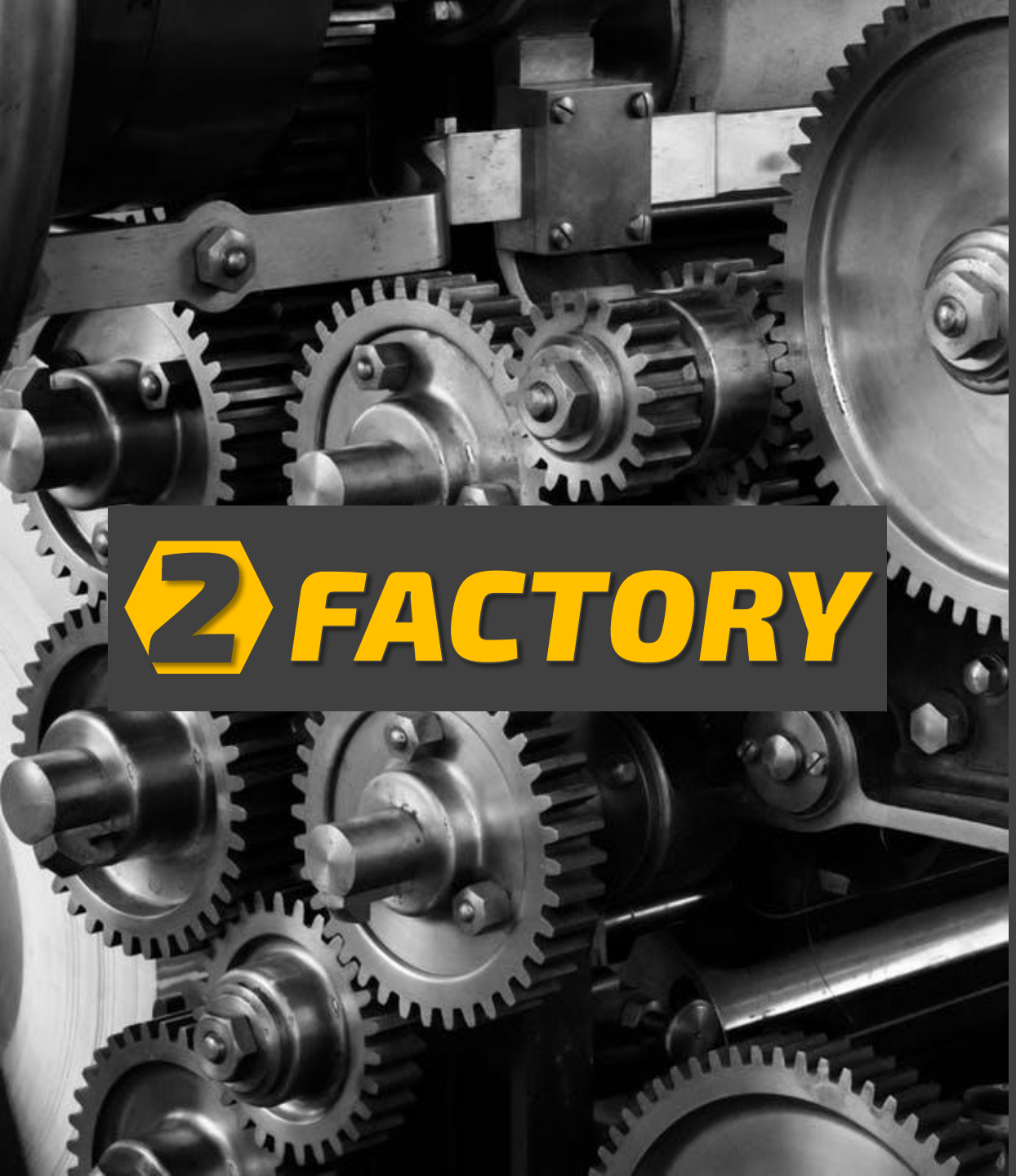
- É uma fachada que constrói os Objetos das classes PO;
- Consiste em criar uma classe para lidar com métodos complexos;
- Combina ações em páginas diferentes.

QUAL PROBLEMA SOLUCIONA?

- Reduz a complexidade de escrita de código contendo todos os objetos utilizados em seu construtor.

O QUE POSSIBILITA?

- Possibilita uma escrita mais legível dos passos, e unifica páginas criando uma jornada.



2 FACTORY

CARACTERÍSTICAS

- Consegue lidar com criação dinâmica de objetos;
- Consiste em criar uma classe com responsabilidade de construção de objetos;
- Entrega métodos que retornam objetos específicos.

QUAL PROBLEMA SOLUCIONA?

- Remove a responsabilidade de criação de objetos da classe de teste e contém sua lógica própria para lidar com diversas situações de dados.

O QUE POSSIBILITA?

- Possibilita construir objetos com características diversas em uma só classe, possuindo métodos com únicas responsabilidades.

3 SINGLETON

CARACTERÍSTICAS

- Verifica se a classe já foi instanciada uma vez, antes de retornar o Objeto;
- Consiste em criar uma classe com o construtor privado, e sendo referenciado de maneira estática;
- Entrega métodos que retornam objetos específicos.

QUAL PROBLEMA SOLUCIONA?

- Quando não queremos ter mais de uma instância de um Objeto no nosso contexto, esse Pattern traz a solução.

O QUE POSSIBILITA?

- Possibilita que apenas uma instância de um determinado Objeto exista a qualquer momento, dando a você esse controle de gestão.

4 PAGE OBJECT

CARACTERÍSTICAS

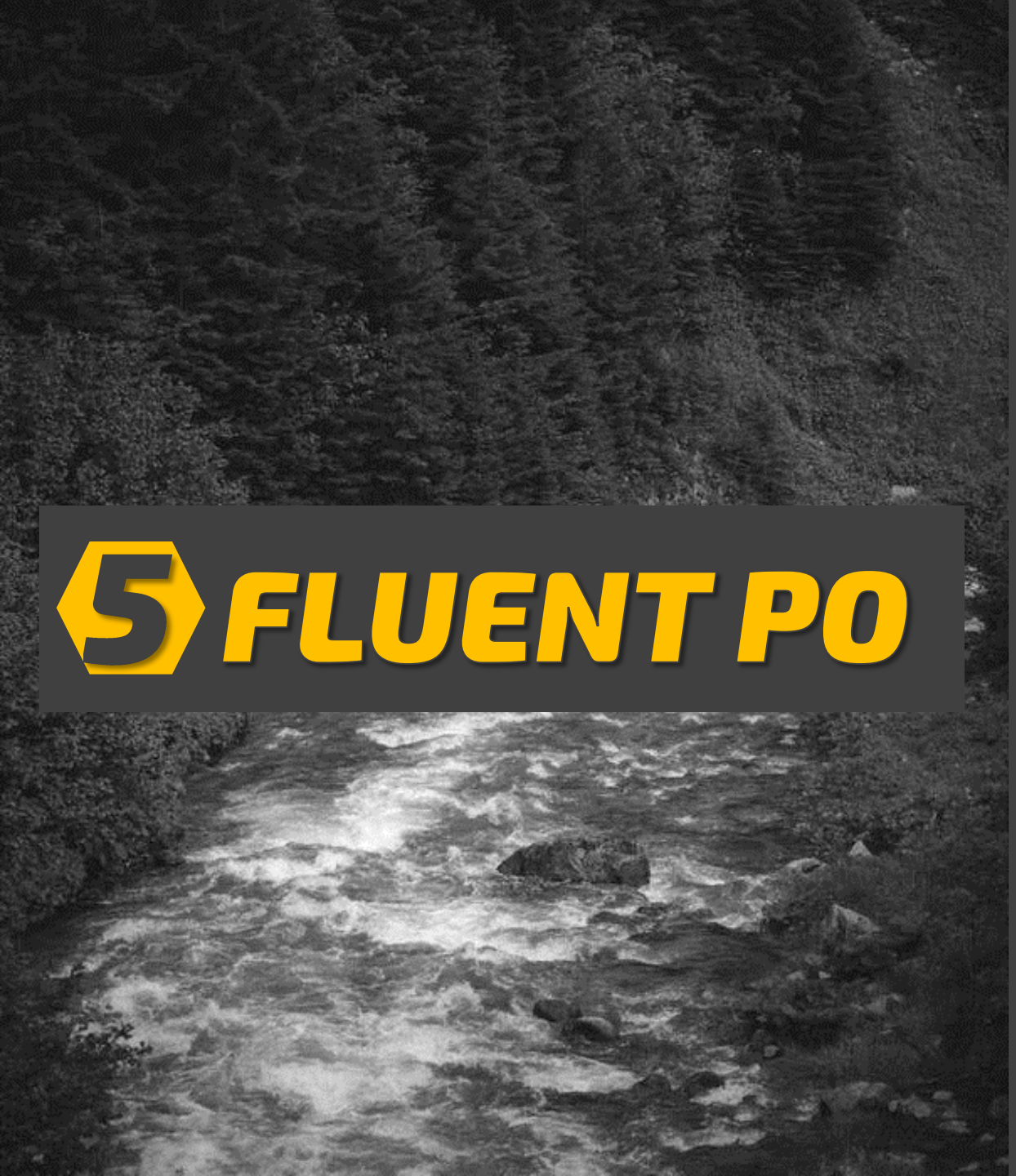
- A página vira uma classe orientada a objetos;
- Consiste em localizadores e métodos de interação com o teste.

QUAL PROBLEMA SOLUCIONA?

- Mapeamos cada página como uma classe e seus elementos como variáveis, e mantemos assim os localizadores e os casos de testes separados uns dos outros, possibilitando o reaproveitamento de código.

O QUE POSSIBILITA?

- Possibilita uma manutenção mais simples e reduz a complexidade das nossas classes de teste.



5 FLUENT PO

CARACTERÍSTICAS

- Implementa métodos para encadeamento da lógica de negócio nos testes;
- Consiste nos métodos que realizam ações, o retorno é ele mesmo;
- Possibilita o retorno de outra Classe de PO.

QUAL PROBLEMA SOLUCIONA?

- Simplifica o pattern Page Objects, que por sua vez soluciona a questão de mapeamento de páginas em testes Web.

O QUE POSSIBILITA?

- Possibilita uma escrita fluente dos seus métodos de ações, contidos na sua PageObject. Neste padrão, todos os métodos retornam ele mesmo.

6 BUILDER

CARACTERÍSTICAS

- Constrói objetos específicos de maneira mais simples;
- Utilizado em conjunto com o padrão Factory traz muitos benefícios.

QUAL PROBLEMA SOLUCIONA?

- Fornece uma solução para problemas relacionados a criação de objetos na programação orientada a objetos.

O QUE POSSIBILITA?

- Separa a construção de um objeto complexo da sua representação, fazendo com que possamos construir diferentes representações do mesmo objeto.

HANDS ON!

EVITANDO O CAOS

MÉTODOS PARA FUGIR DE PROJETOS DESORGANIZADOS

- Invista tempo projetando sua solução;
- Refatore;
- Não cometa o erro de sair automatizando sem saber o quê;
- Pense na sua automação como produto;
- Identifique os feedbacks que são importantes;
- Não esqueça das Boas Práticas;
- Cuide quando o projeto começar a crescer sem controle;
- Execute seus testes com periodicidade, isso lhe traz uma imagem do seu projeto;
- Alterá-lo não deve ser difícil.

ENCERRAMENTO

LEMBRE-SE:

- Os drivers da automação são as disciplinas de desenvolvimento de software;
- Busque elevar seu conhecimento técnico, utilize os desenvolvedores como seus gurus;
- Olhe para os seus projetos como produtos;
- Os projetos de automação devem ter arquitetura tão sofisticada quanto as do software.

Através dessas práticas nossos projetos irão mais do que simplesmente validar, irão possibilitar o monitoramento de negócio e a tão falada testabilidade.

ENCERRAMENTO

PROJETOS DISPONÍVEIS EM:

Aplicação alvo construída para esta apresentação:
https://github.com/brunolusa/api-carros

Projeto de teste automatizado Web apresentado:
https://github.com/brunolusa/carros-web-test

Front-end construído para esta apresentação:
https://github.com/brunolusa/react_api_carros

Projeto de teste automatizado API apresentado:
https://github.com/brunolusa/api-carros-integration-test

Client construído para esta apresentação:
https://github.com/brunolusa/api-carros-it-client

#THEDEVCONF

SÃO PAULO 2020