Laboratório de Inteligência de Máquina
e Modelos de Computação

ERAD-SP 2018

Colocando Ideias em Produção

Ariel Schvartz

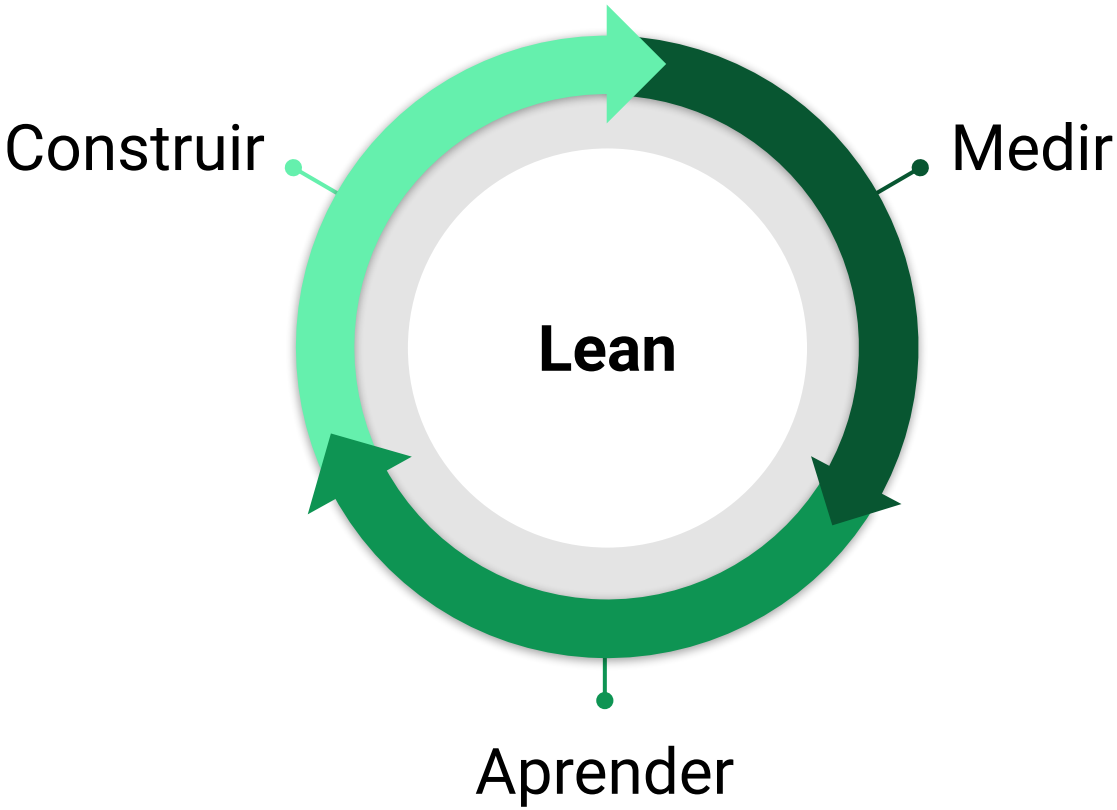
Ariel Schvartz

linkedin.com/in/arielschvartz

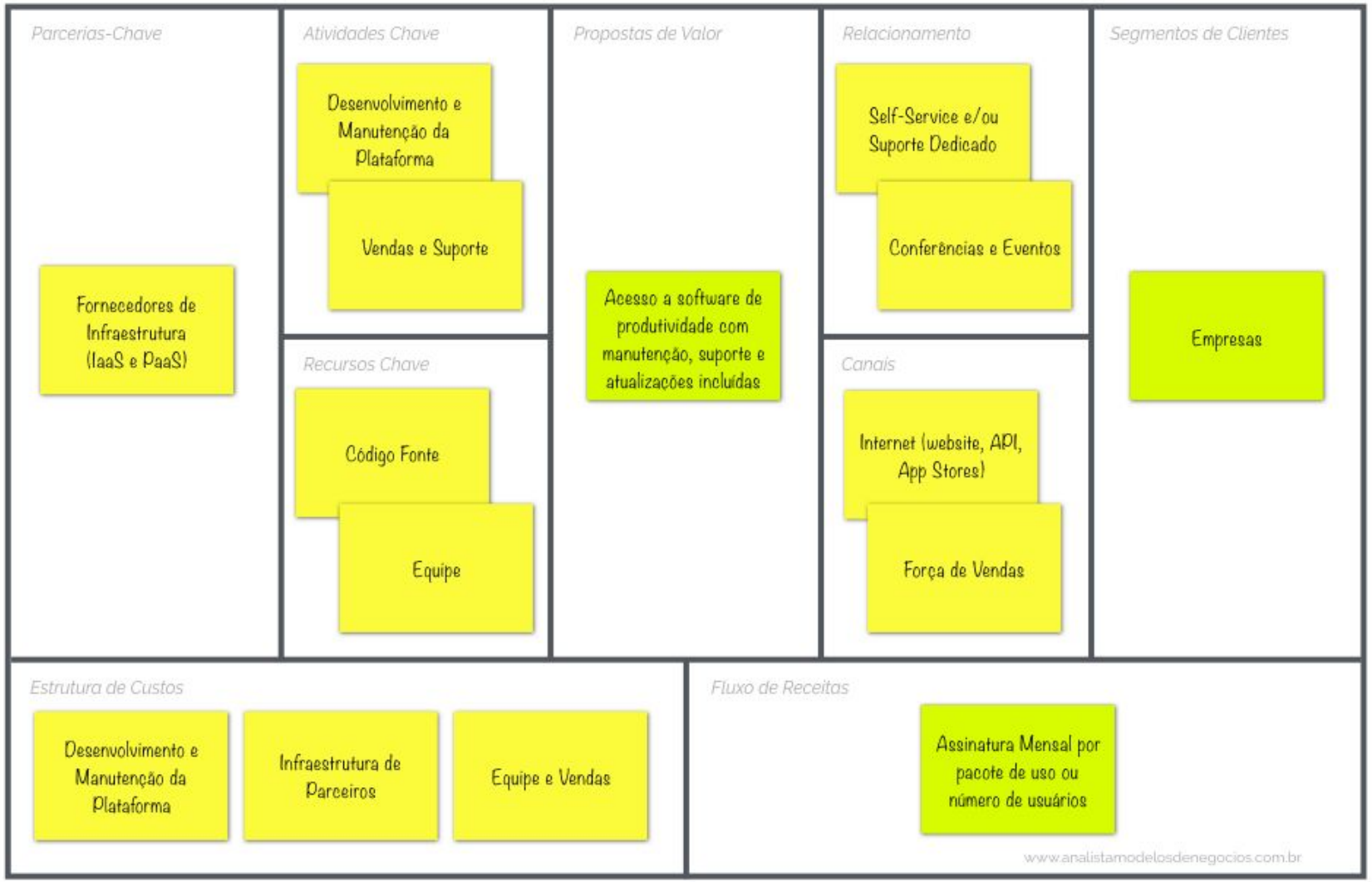
Agenda

1. **Design Thinking**
   Entendendo a ideia e mapeando a solução
2. **Design**
   Materializando a ideia e colocando ela no papel
3. **Definindo a Arquitetura**
   Modelo de dados, comunicação e etc
4. **Escolha de Linguagens e Frameworks**
   Vantagens, desvantagens e pontos de decisão
5. **Desenvolvimento de Código**
   Boas práticas, agilidade e eficiência.
6. **Testes**
   Ambientes de testes e testes automáticos
7. **CI / CD**
   Automatizando o processo de colocar em produção
8. **Escala**
   Como arquitetar e otimizar o servidor para aguentar carga
9. **Casos Reais**
   Lawgile
   Rupee
   Medicamentos Inteligentes
   BGC Brasil
10. **Serverless**
    Conceito e como usar o Framework

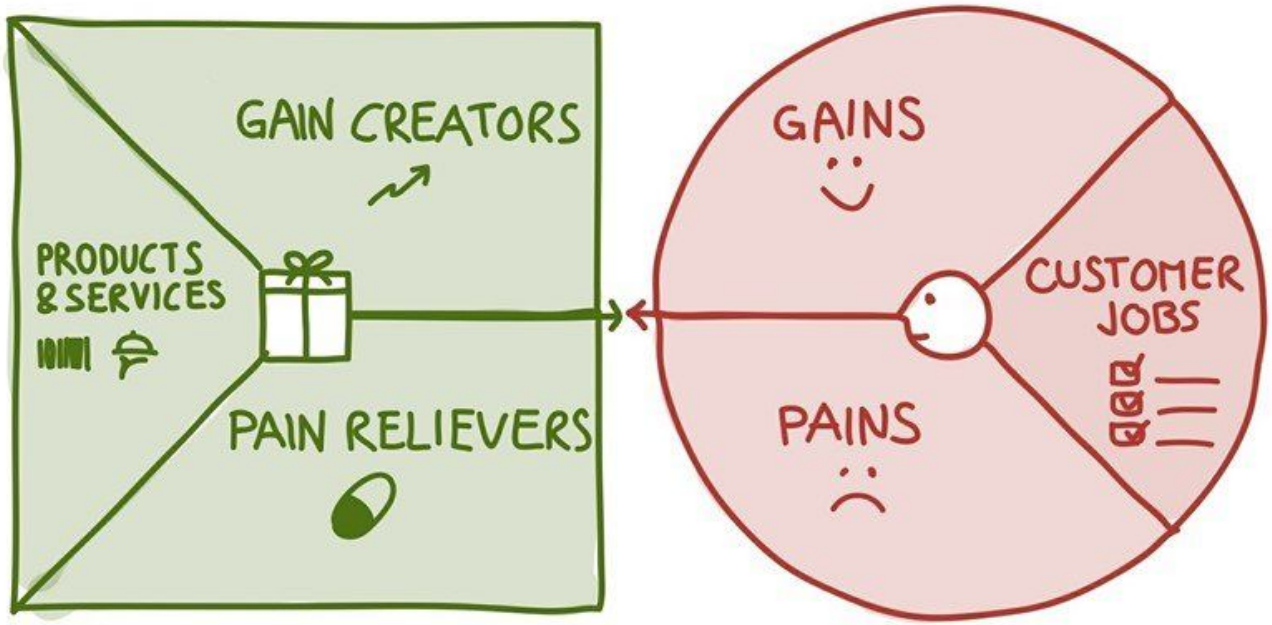
Lean Startup



Business Model Canvas

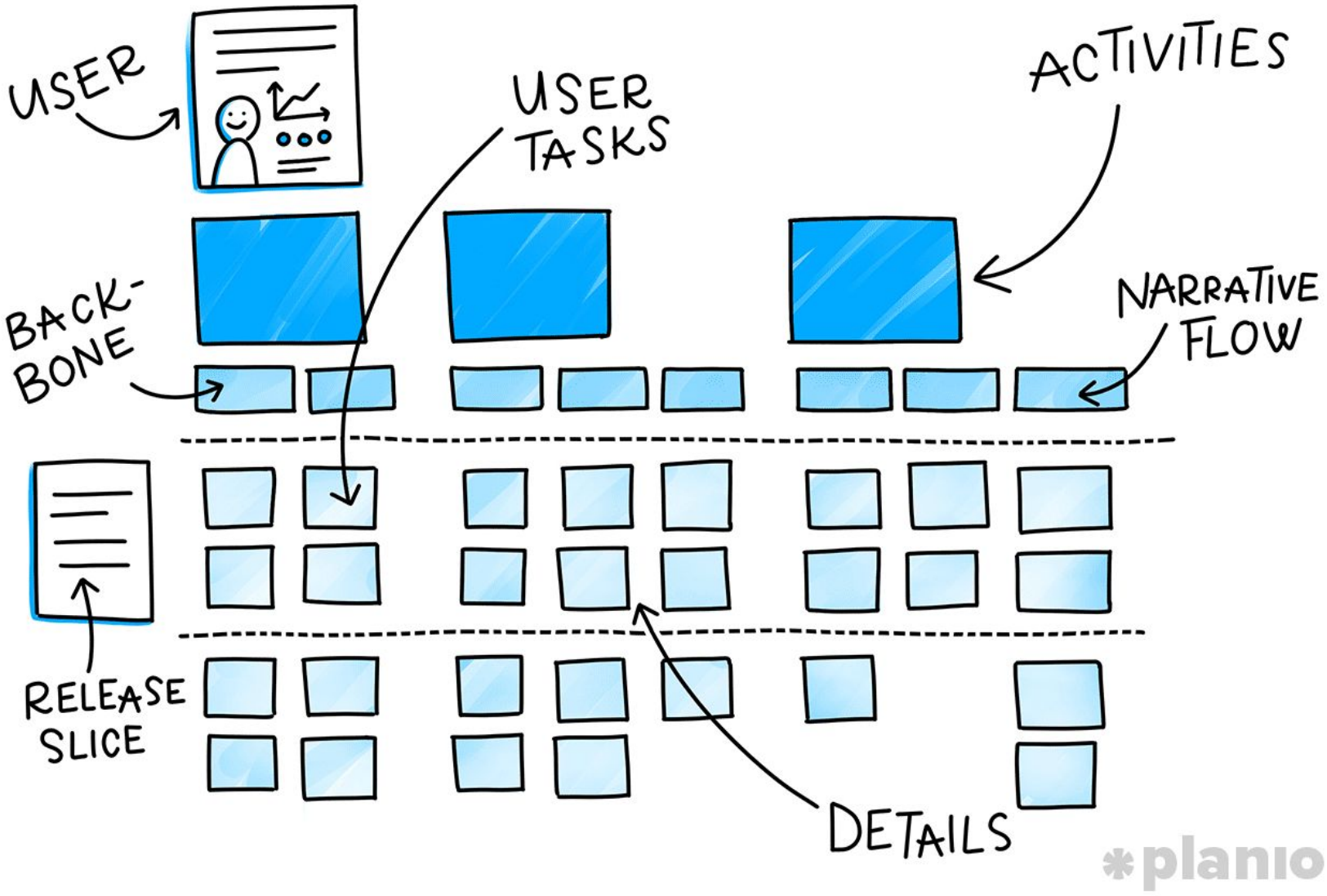


Value Proposition Canvas

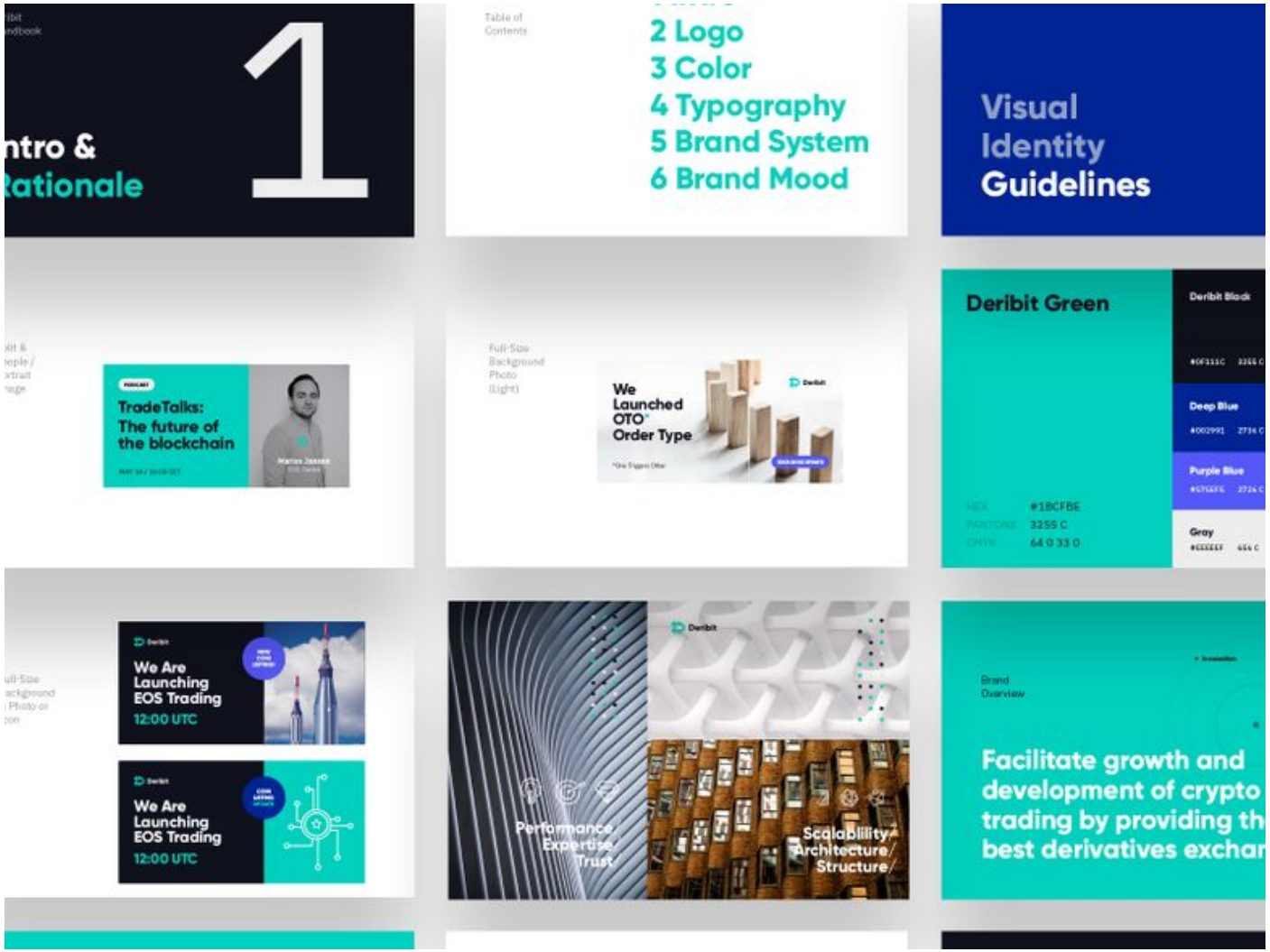


Design Studio

User Story Mapping

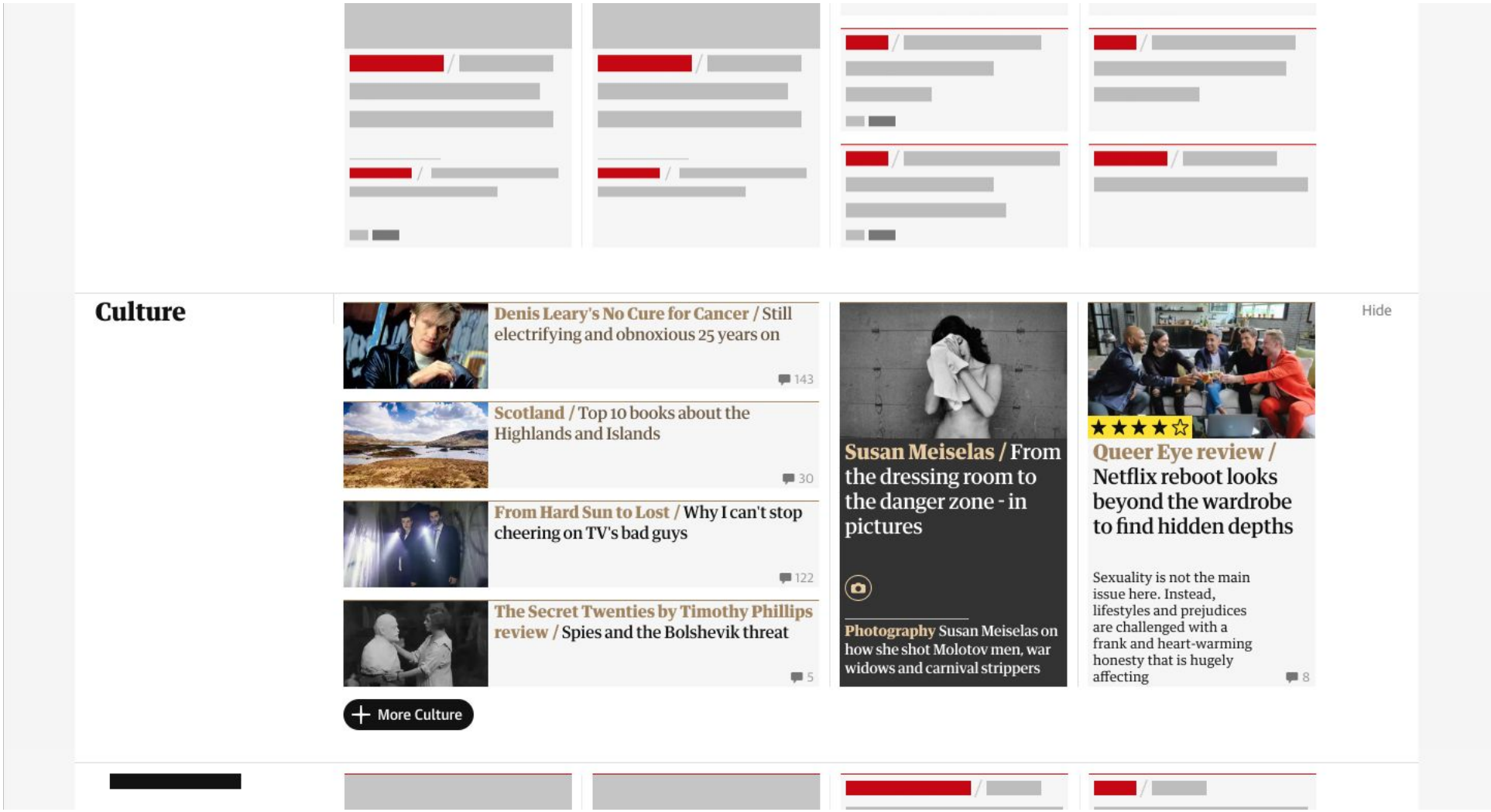


Identidade Visual

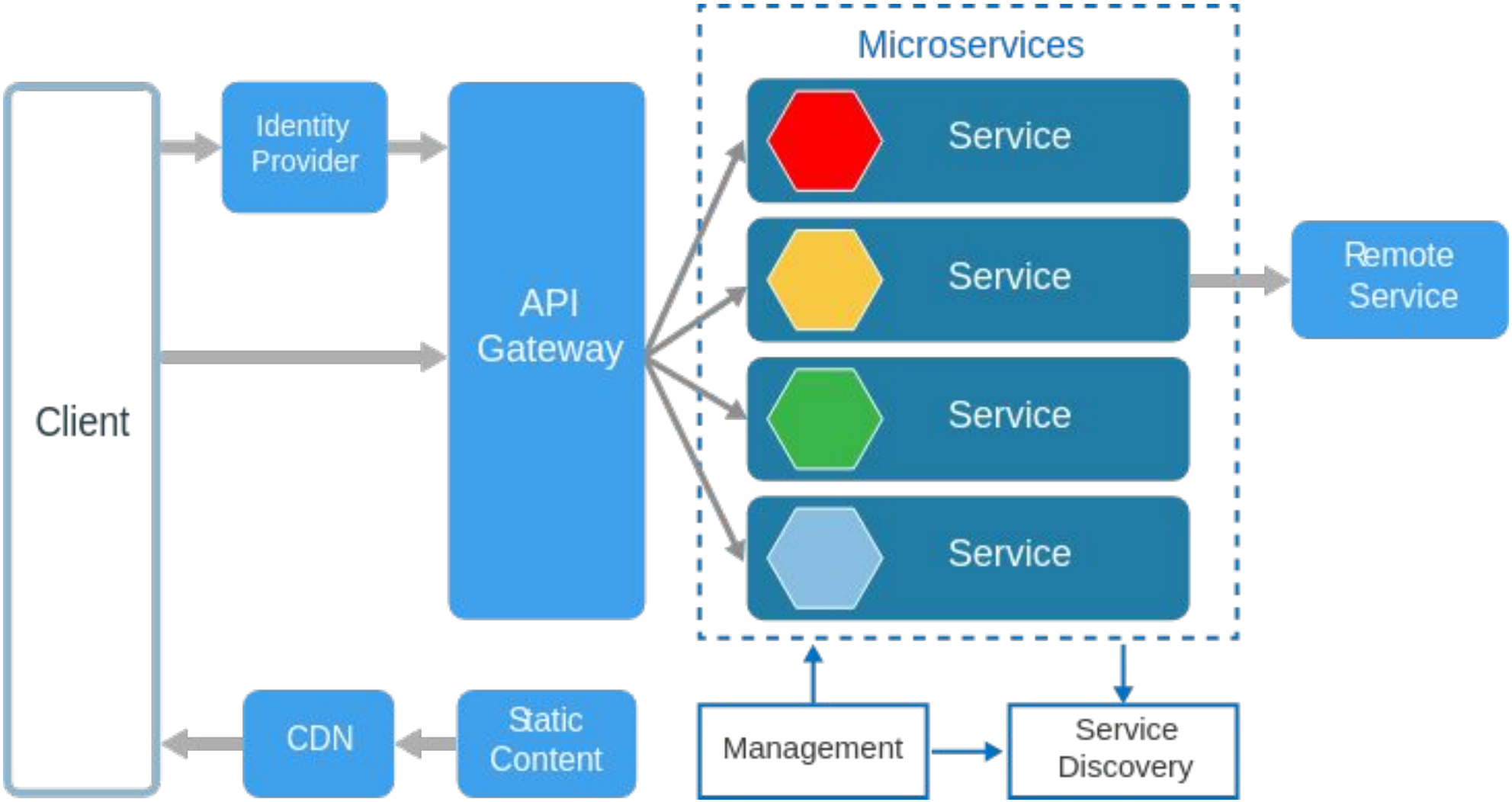


Wireframes

Layout


Culture
Denis Leary's No Cure for Cancer / Still electrifying and obnoxious 25 years on
143
Scotland / Top 10 books about the Highlands and Islands
30
From Hard Sun to Lost / Why I can't stop cheering on TV's bad guys
122
The Secret Twenties by Timothy Phillips review / Spies and the Bolshevik threat
5
More Culture
Susan Meiselas / From the dressing room to the danger zone - in pictures
Photography Susan Meiselas on how she shot Molotov men, war widows and carnival strippers
Hide
Queer Eye review / Netflix reboot looks beyond the wardrobe to find hidden depths
Sexuality is not the main issue here. Instead, lifestyles and prejudices are challenged with a frank and heart-warming honesty that is hugely affecting
8

Arquitetura do Sistema



Partes do Sistema

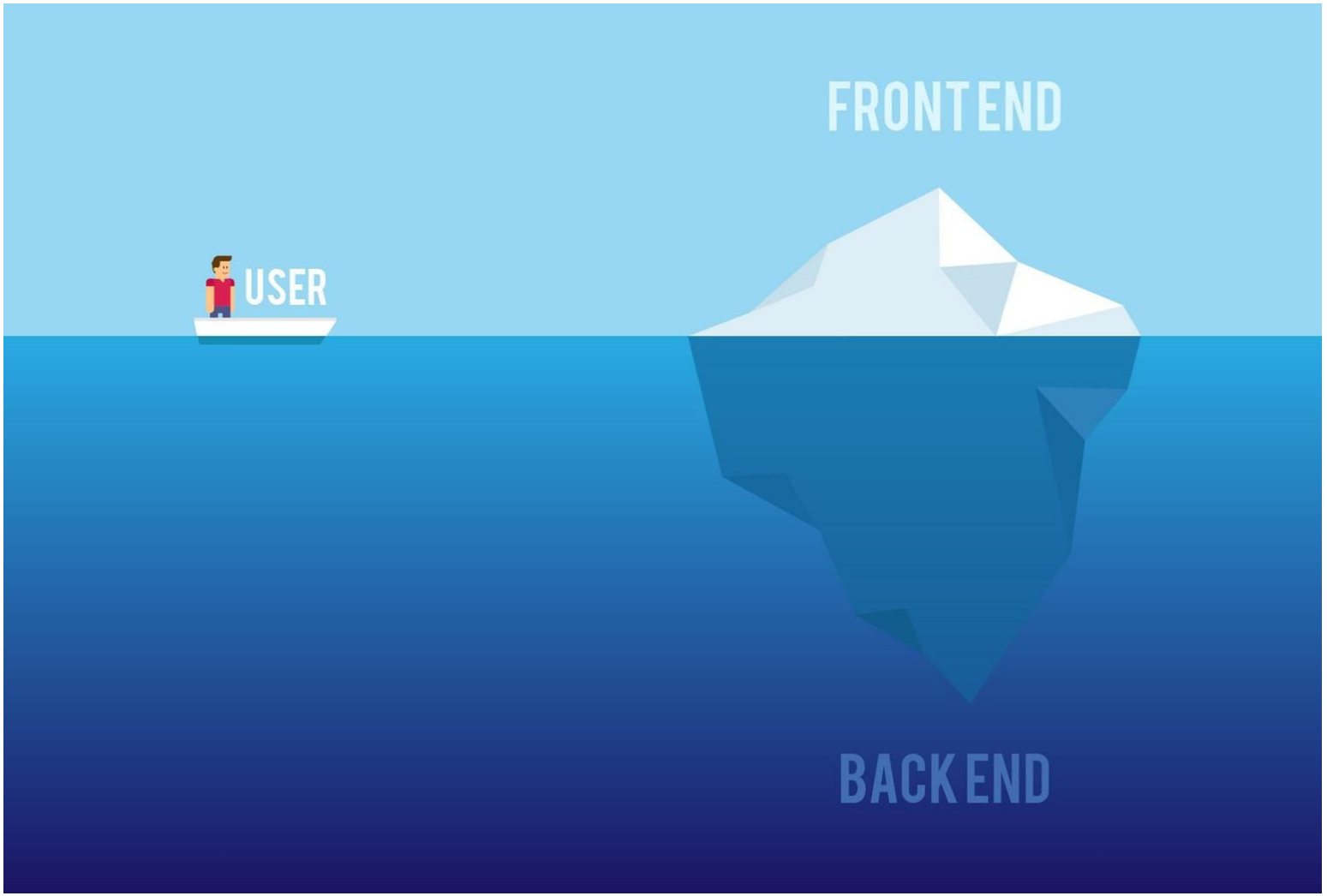
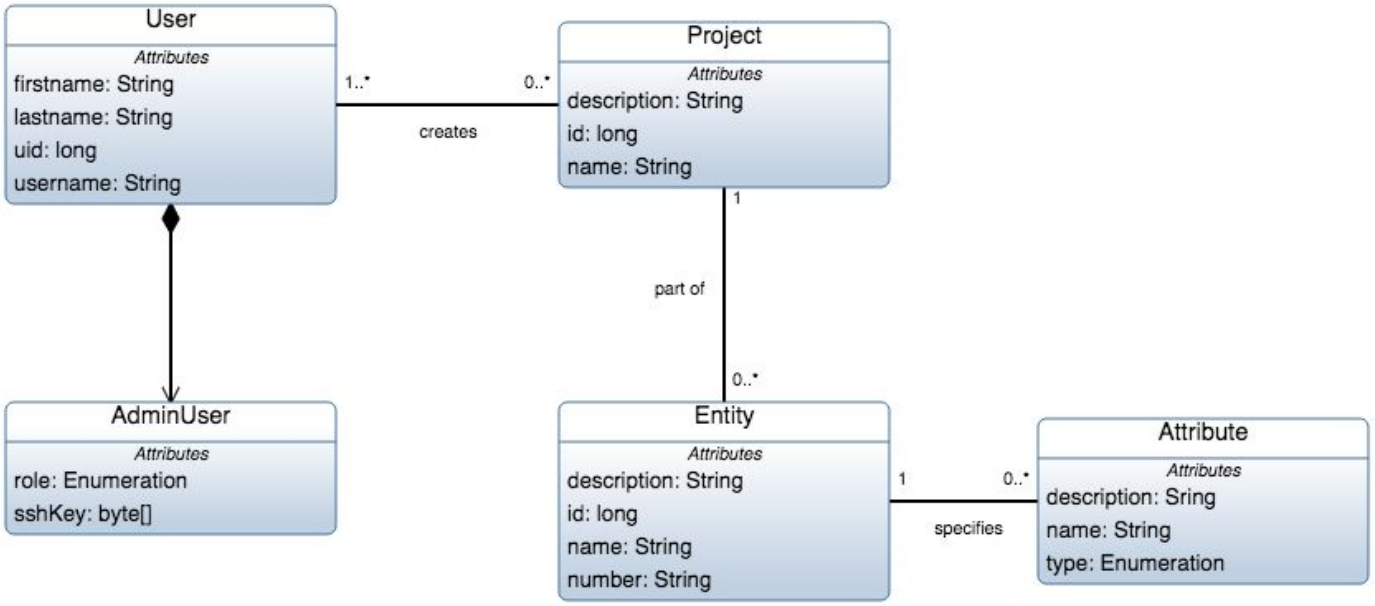


Diagrama de Classes

- Como vamos organizar o Banco de Dados?
- O que precisamos guardar?
- Quais entidades existem no sistema?
- Como as entidades se relacionam?



Arquitetura do Banco de Dados

- Centralizado
- Distribuído
- Replicado
- Blockchain

- Bancos Relacionais (SQL)
  - MySQL
  - PostgreSQL
  - SQLite

- Bancos não Relacionais (NoSQL)
  - Chave-valor
  - Documento
  - GraphDB (BD orientado a grafos)
  - Colunas

Arquitetura do Banco de Dados

SQL

- Modelagem pré-definida
- Consistência de Dados
- Bom para queries complexas de dados
- Costuma precisar de uma infra mais cara

NoSQL

- Dinâmico
  - Não precisa pré definir colunas
- Performance
  - Consultas simples
  - Escrita
  - Escala horizontal

Estrutura do Código

Monolito

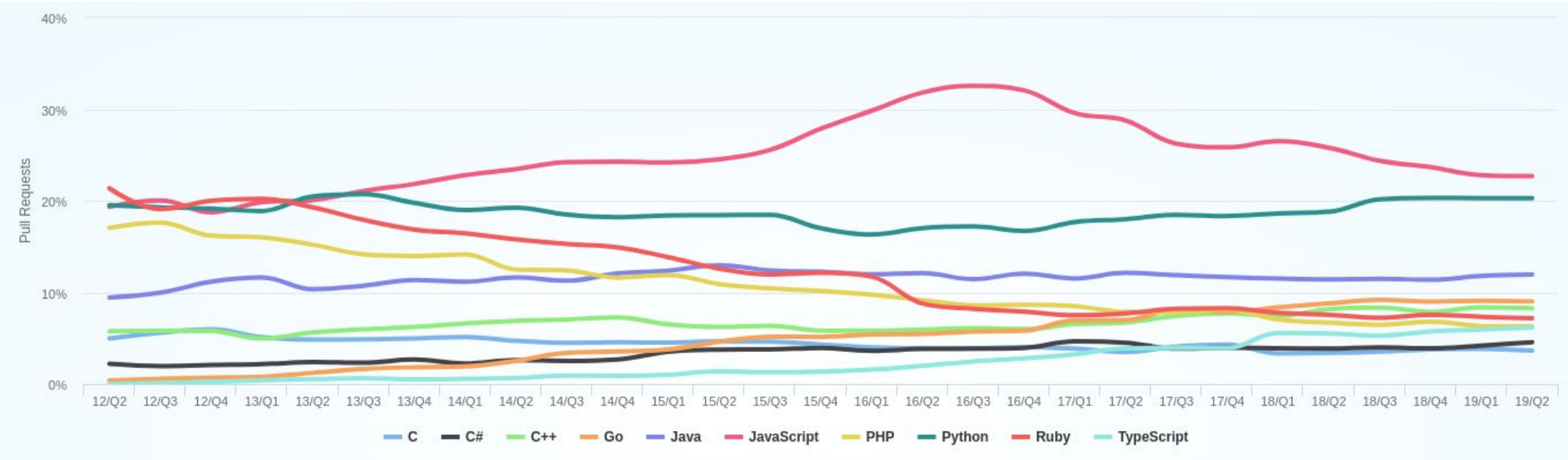
- Tudo num mesmo projeto
- Deploy único
- Gestão única
- Código mais amarrado e mais difícil de manter
- Mais ágil pro MVP

Micro serviços

- Comunicação por APIs
- Bancos de Dados independentes
  - Replicação do necessário
- Múltiplos deploys
- Multi linguagens
- Otimização por serviço

Qual linguagem escolher?

Qual linguagem escolher?



Qual linguagem / framework escolher?

- Quais as vantagens específicas da linguagem?
  - Velocidade de Processamento? Agilidade no Desenvolvimento?
- Qual o canal de interação do usuário?
  - Internet vs Tempo de Processamento
- Qual o tamanho da comunidade?
  - A comunidade costuma usar com o mesmo objetivo?
  - Tem bastante lib open source?
  - É facil achar gente no mercado de trabalho?
- Meu case me obriga a algum caso específico?

Qual linguagem / framework escolher?

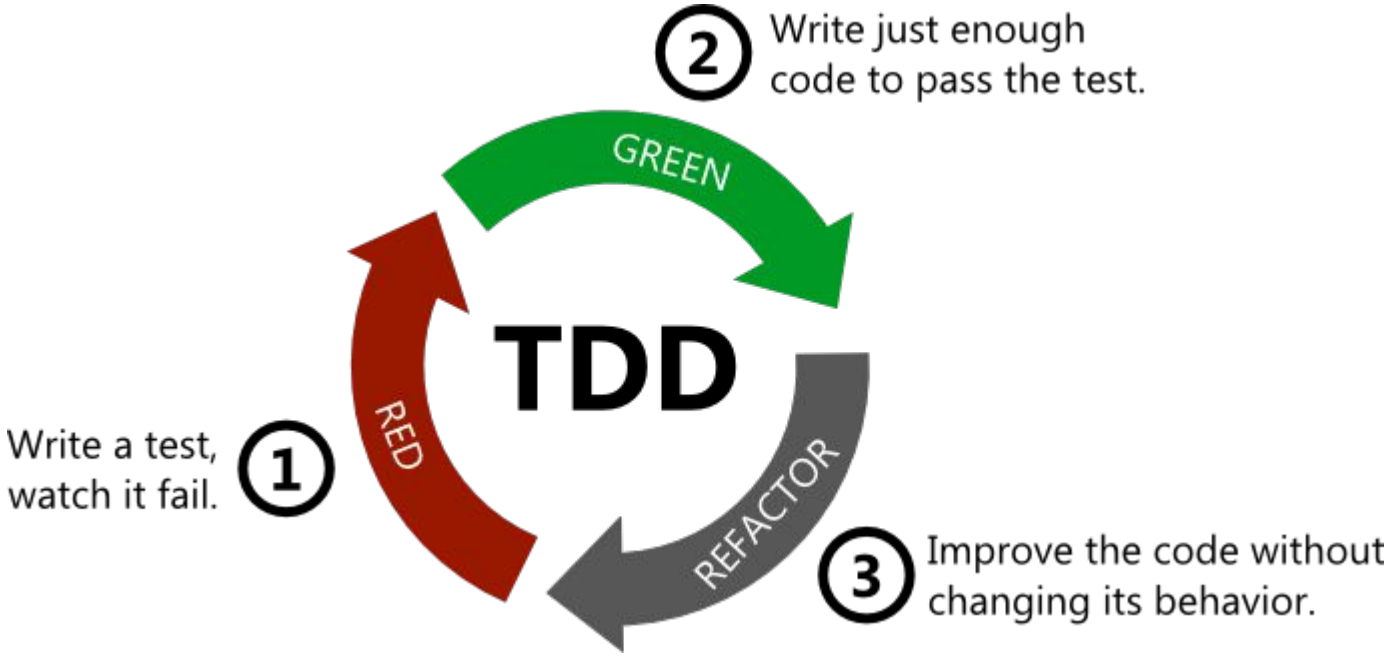
- AirBnB
  - Rails + React, React Native (Depois Java e Swift)
- Instagram
  - Django + React Native
- Nubank
  - Clojure, Python + React Native, Swift, Java
  - Micro serviços
- Uber
  - Python, Node (Depois Java, Go) + Node, Swift, Java
  - Micro serviços (SOA)

Desenvolvendo o código

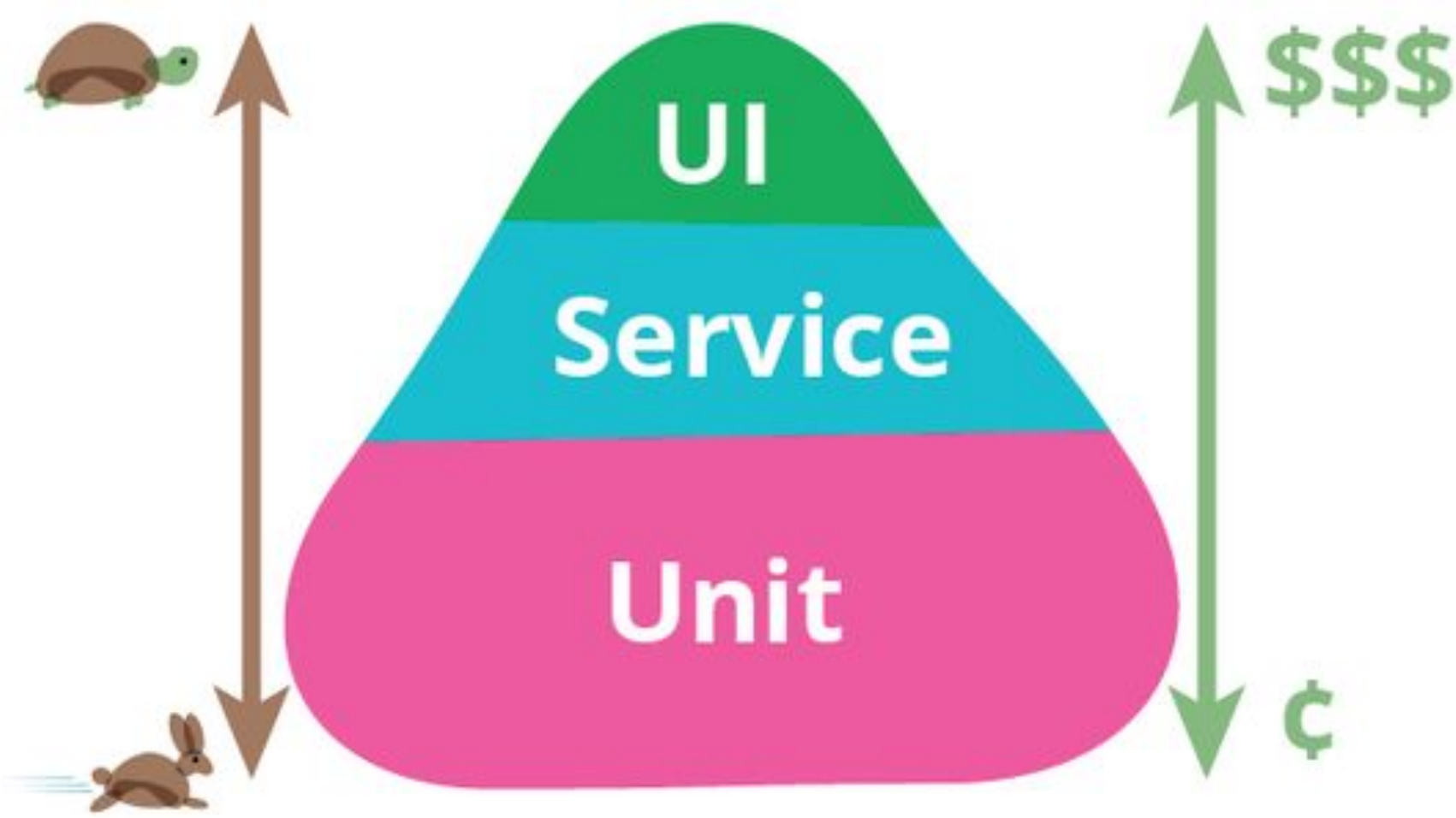
- Discutir a solução antes de começar
- Pair Programming
- Refactor Constante
- Boas práticas
  - DRY
  - Nomenclatura
- Manutenabilidade
  - COC (Convention Over Configuration)
- TDD

TDD

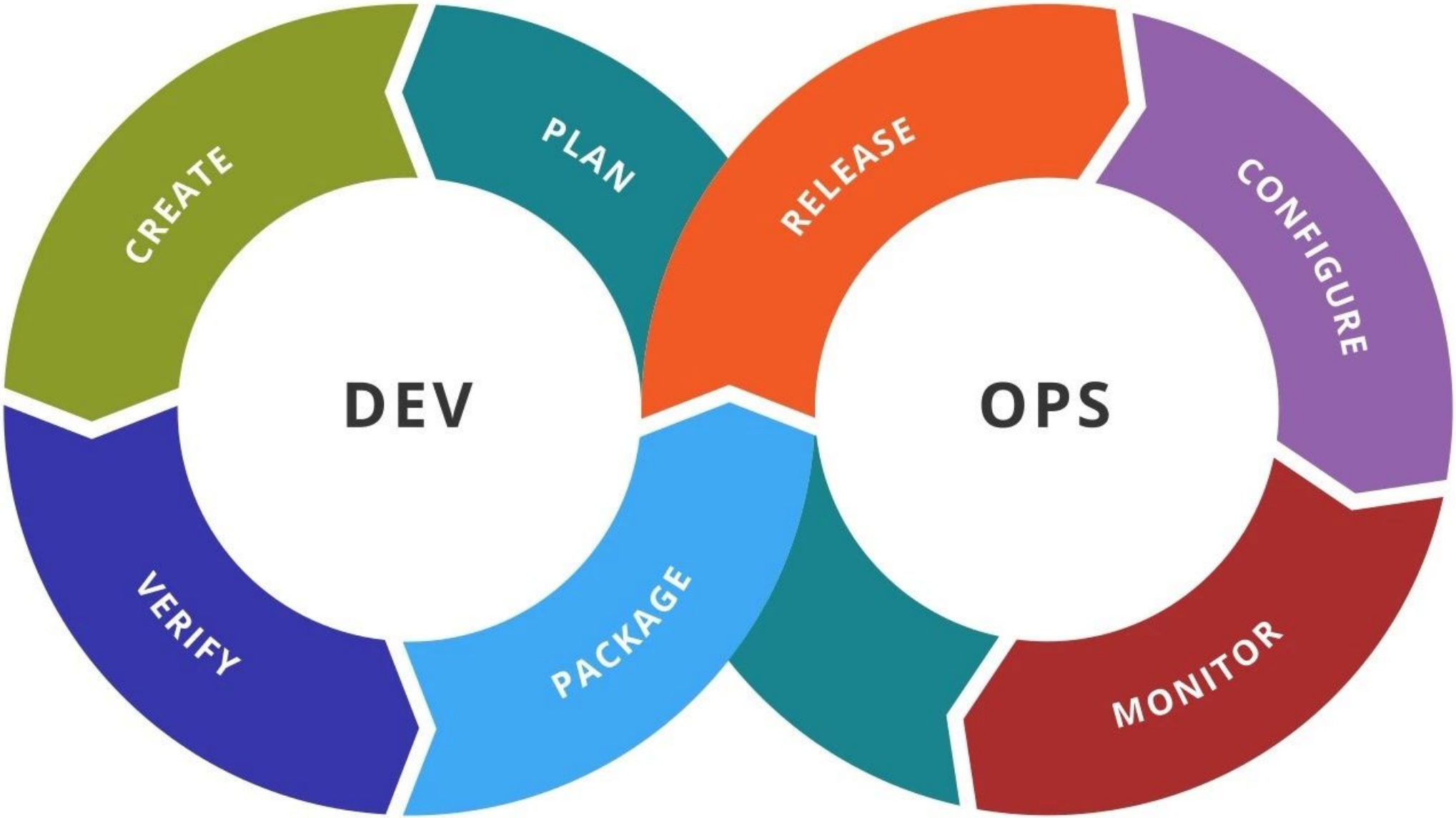
- Tende a ter menos bugs
- Ajuda a pensar em casos de uso não pensados antes
- Costuma economizar tempo



Testes Automáticos



CI / CD

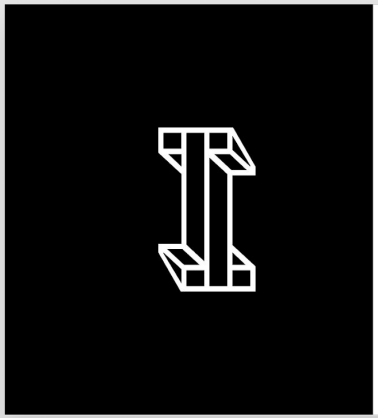

CREATE
PLAN
RELEASE
CONFIGURE
DEV
OPS
VERIFY
PACKAGE
MONITOR

Colocando em Produção

- Configurando Servidores
  - Maquina Virtual? Kubernetes? Serverless?
  - Bancos de Dados
    - Migrations & Seeds
    - Backups
- Interfaces Web
  - Sites Estáticos vs Server Side Rendering
- Infrastructure as Code
  - Terraform, Cloudformation

Escala do Servidor

- Métricas de Performance
  - Alarmes para problemas
- Logs bem estruturados
- Custos gerados
  - Como otimizar os custos?
- Downtime para mudanças
- Tempo de Resposta aceitável
- Distribuição de carga de acesso
- Estratégias de Deploy
  - Big Bang, A/B, Canary



INVENTOS
digitais

pyggbank

CULTURA INGLESA

code COMET

LIVIMA

WITSEED

cmov

BizCapital

Super Revendedores

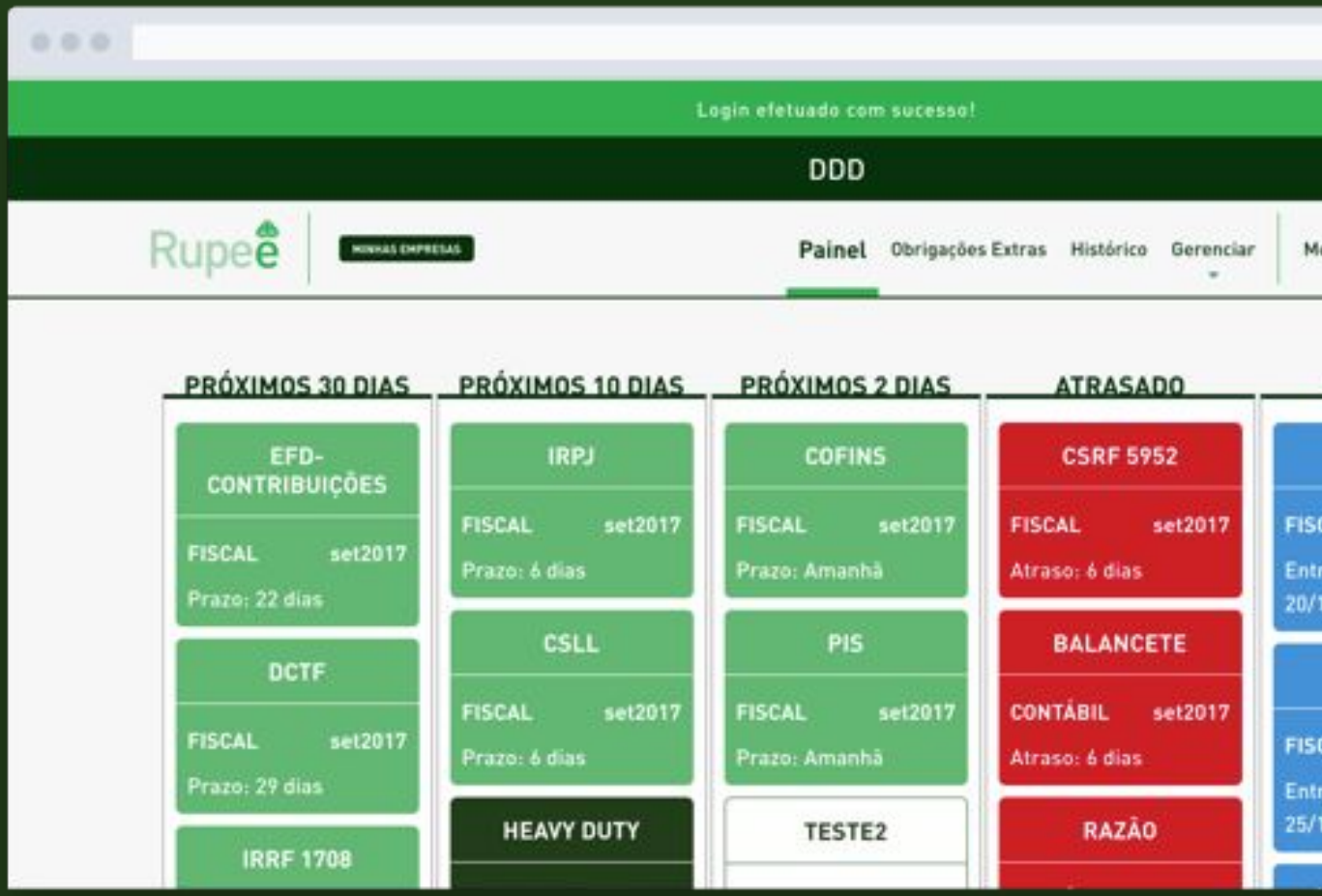
OLIVIA

Rupeê

Plataforma de automação e controle contábil, fiscal e tributário.

Atendendo a:

- Grandes conglomerados
- Multinacionais
- Empresas listadas em bolsa
- Grandes franquias
- Um dos maiores escritórios de contabilidade do Brasil



Rupee

- Projeto em Rails
- Velocidade de Desenvolvimento de Produto
- Banco de Dados SQL
  - Queries complexas
  - Cache
- Segurança da Informação
  - Controle de acessos
  - Links temporários

lawgile

Eficiência e transparência no mundo jurídico

Lawgile

- Projeto Serverless
- NodeJS
- React + React Native
- API GraphQL
- Processo bem elaborado de CI/CD
  - Muito integrado às validações
- Segurança da Informação


Medicamentos Inteligentes
Tecnologias para a saúde do futuro

Medicamentos Inteligentes

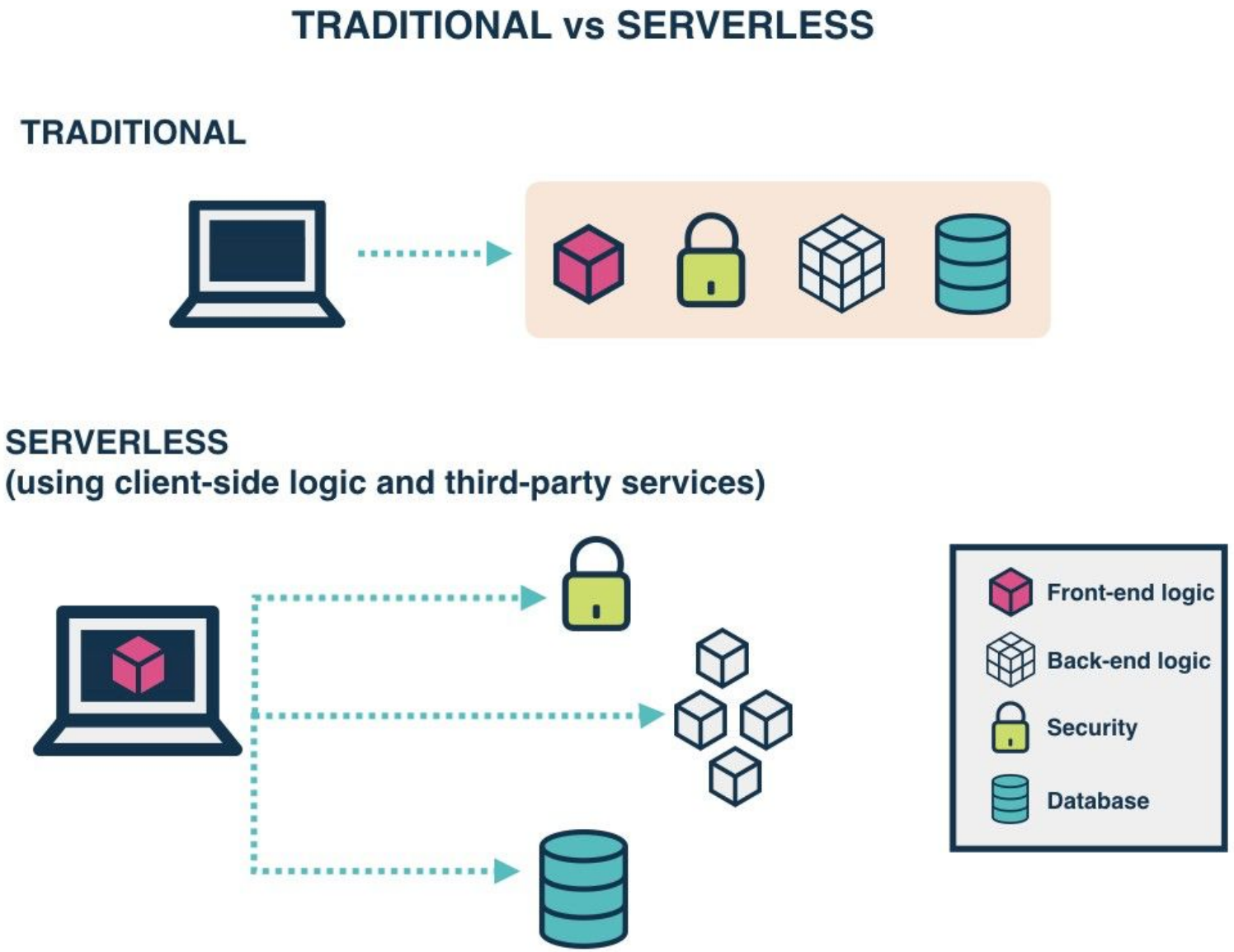
- Interface via Whatsapp
  - WhatsApp Business API
  - DialogFlow do Google
    - Arvore de decisão
- Backend Serverless
- Machine Learning

A bgc é a maior especialista em background checks do Brasil, com uma solução que alia tecnologia e expertise jurídica e entrega resultados rápidos e seguros para empresas de todos os tamanhos.

BGC

- Arquitetura Serverless
- Quebrado em micro serviços
  - BDs próprios
  - Streaming de dados
- Filas para controle de volume
- Máquinas de Estado
- Bancos SQL e NoSQL
- Otimização de custos

Serverless


TRADITIONAL vs SERVERLESS
TRADITIONAL
SERVERLESS
(using client-side logic and third-party services)
Front-end logic
Back-end logic
Security
Database

Serverless

Vantagens:

- Zero preocupação com manutenção de servidor
- Sem custo de tempo ocioso
- Sem problemas com escala
- Atualização e deploy rápido

Desvantagens:

- Dificil de testar e debuggar
- Processos longos
- Ficar preso a provedores de infra
- Poucas pessoas no mercado com experiência

Serverless

Principais fornecedores:

- Amazon AWS Lambda
  - Mais antigo e maduro
  - Introduzido em Nov 2014
- Google Cloud Functions
- Azure Functions
- IBM OpenWhisk

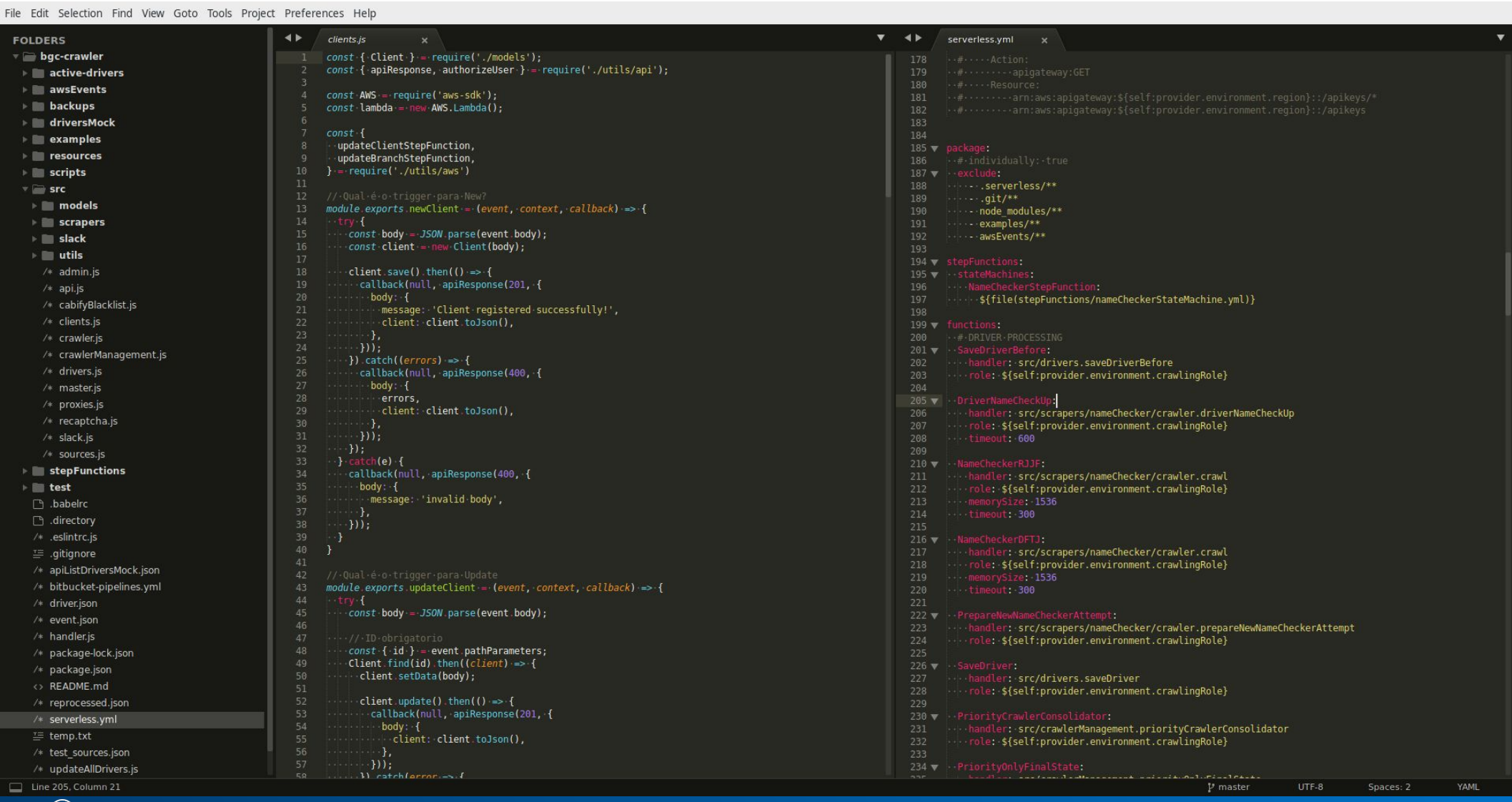
AWS:

- Lambda
- Step Functions
- DynamoDB on Demand
- RDS Aurora
- Kinesis Streaming
- CloudFormation

Serverless Framework

- Organização de código
- Plugins
- Linha de comando
  - Deploy
  - Logs
- Multi Provedor

Vamos ver na prática?



Obrigado

Contatos:

- Ariel Schvartz
- E-mail: ariel@ariels.com.br
- LinkedIn: linkedin.com/in/arielschvartz
- Telefone: +55 21 99631-8859

Estamos sempre contratando! =)

Só entrar em contato!