

# WorkFlow **SUPER** **Full Stack**

Desenvolva seus projetos  
como um profissional

**ONE BIT CODE**  
PROGRAMAÇÃO ZEN!

# WorkFlow Super Full Stack

## Porque o WorkFlow é importante?

---

O desenvolvimento de software é um processo árduo e longo e em geral envolve muitas pessoas. A falta de um Workflow (fluxo de trabalho) leva ao stress da equipe, maior tempo e custos no desenvolvimento e possivelmente até ao abandono do projeto.

Ter um Workflow que funcione e se adapte a realidade de cada equipe e projeto é fundamental para a entrega do projeto proposto com qualidade (e saúde mental :)).

O Workflow Super Full Stack foi criado para se adaptar a projetos de diversas naturezas e equipes que vão de uma até dezenas de pessoas. Ele foi baseado nos métodos Ágeis e na vivência prática do desenvolvimento de software.

### **O que ele pode fazer por você**

Se você quer desenvolver um projeto de maneira sustentável, com entregas contínuas, baseado em testes, com maior qualidade e com a segurança de que você está indo na direção correta, o Workflow Super Full Stack certamente pode te ajudar nisto.

### **Porque ele foi criado**

Nos últimos anos eu havia desenvolvido muitos tipos de projetos (com várias tecnologias diferentes) sem um Workflow bem definido.

Como os projetos em geral eram pequenos ou médios, possuíam um tempo razoável de desenvolvimento, em equipes pequenas os problemas relativos à falta de um workflow fixo não causavam tanto transtorno (a verdade é que eu não tinha me dado conta da falta que isso me fazia :X).

---

Porém depois da criação do Bootcamp Super Full Stack (eu explico mais sobre ele nas últimas páginas) eu tive a necessidade de criar múltiplos projetos complexos em um tempo extremamente curto, com alta qualidade e ainda explicar o passo a passo para que os alunos pudessem reproduzir o projeto e aprender com ele.

Você deve imaginar que a minha necessidade de organização aumentou exponencialmente e foi exatamente isso que aconteceu. Para reagir a essa nova demanda eu experimentei várias organizações diferentes até conseguir sintetizar uma que funcionasse para mim.

Depois da sintetização da base desse Workflow eu fui melhorando, incrementando e continuo fazendo isso até hoje :)

## **Quem deve usar este Workflow**

Eu e outras dezenas de pessoas (eu ensino como usar esse workflow nas minhas consultorias a empresas e no Bootcamp) temos utilizado esse Workflow em projetos de várias naturezas, desde projetos de 2 semanas que envolvem apenas uma pessoa até projetos que duram meses e envolvem times de mais de dez pessoas.

Então caso você esteja iniciando um projeto (ou possa mudar o fluxo de um projeto que já está rolando) vale a pena testar na prática todos os pontos do Workflow Super Full Stack.

# WorkFlow Super Full Stack

## Aprendendo Parte a Parte

---

O Workflow é dividido em **3 fases progressivas** e cada uma dessas fases possui vários pequenos passos fundamentais para que você consiga entregar seu projeto com qualidade.

Eu vou explicar o porquê de cada fase e cada um dos passos que a compõe e no final eu vou deixar um resumo visual para que você possa usar no dia a dia.

### **O Workflow é um framework**

O Workflow funciona como um framework (assim como Ruby On Rails) por tanto ele foi criado para abordar os principais passos do desenvolvimento de um projeto mas nem todos os elementos dele precisam ser utilizados para que ele funcione.

Então eu vou deixar uma recomendação de uso para cada passo para que você consiga utilizar o que for mais importante para você :)

### **Esse processo deve ser feito por quem?**

O desenvolvimento dessa documentação deve incluir o máximo de pessoas da equipe (e quando for o caso o próprio cliente) por tanto evite segui-lo sozinho(a).

Quando os gerentes do projeto, os clientes, os designers, os programadores e etc se unem para criar a documentação, muitos pontos cegos são preenchidos e o projeto se desenvolve de maneira saudável.

### **Basta documentar e depois sair programando?**

Os projetos evoluem com o tempo e a documentação deve evoluir com eles, revisite a documentação e sempre que precisar implementar algo novo.

# Planejamento Inicial

FASE 1

# WorkFlow Super Full Stack

## Fase 1 - Planejamento Inicial

---

A maior parte dos programadores assim que recebe o pedido do desenvolvimento de um novo projeto (ou tem alguma ideia para um projeto pessoal) começa por qual parte? **Pelo código...**

Mas é claro que você deve imaginar que sair desenvolvendo sem se planejar é uma ideia bem pouco sustentável e vai levar ao abandono do projeto, então a primeira coisa que fazemos é compreender o que realmente precisamos fazer antes de sair codando.

### Passo 1 - Caos

No momento que você tem uma nova ideia ou quando recebe a demanda do cliente fica difícil organizar toda a implementação e as regras de negócio em um documento com formato fixo de documentação, então ao invés de fazer isso você vai:

1. Abrir o [Evernote](#)
2. Criar uma lista
3. Escrever todos os elementos que pareçam importantes até que suas ideias se esgotem:
  - a. Regras de negócio
  - b. Ideias para a implementação
  - c. E etc

### Porque esse processo é importante?

Quando você não tem muitas informações claras sobre o projeto mas tem muitas ideias sobre as regras de negócio, implementação, e etc é difícil enquadrar tudo isso em uma documentação formal mas ao mesmo tempo é importante não perder essas ideias iniciais (não bloqueie sua criatividade e não perca suas ideias).

## Quando usar?

Sempre que você estiver começando um projeto (ou feature mais complicada) onde você ainda não sabe muito sobre ele.

## **Passo 2 - Planejamento criativo**

Depois de passarmos pelo Caos é hora de começar a ordenar o que sabemos sobre o projeto, então neste passo nós vamos sintetizar algumas coisas no papel (virtual :)).

1. Crie uma pasta no [Google Drive](#) (ou onde você quiser deixar a organização do seu projeto)
2. Dentro desta pasta crie outra com o nome de “planejamento”
3. Agora na nova pasta crie um arquivo chamado “planejamento criativo”
4. Nesse documento você vai:
  - a. Sintetizar a ideia: Escrever um pequeno texto capaz de explicar o que é seu projeto (de uma maneira que um leigo entenda)
  - b. Sintetizar as funções do Projeto: Criar uma lista que explique cada uma das funções que o seu projeto vai ter

## Exemplo:

**Sintetização da ideia:** O Uber é um APP que conecta pessoas que precisam de um transporte privado com pessoas que desejam e possam oferecer o trabalho de motorista privado naquele momento e região.

## **Sintetização das funções:**

1. Cadastro de usuário
2. Cadastro de motorista
3. Busca de motoristas na região
4. Contratação de motoristas na região
5. Pagamento via cartão de crédito
6. Recebimento do extrato por email
7. ...

## Porque esse processo é importante?

Sintetizar no papel o que o seu projeto deve fazer e quais funções ele vai possuir vai te obrigar a considerar as necessidades reais do projeto (e visualizar o preço do desenvolvimento).

## Quando usar?

Sempre que for criar um novo projeto ou quando quiser repensar o projeto atual.

## **Passo 3 - Planejamento comercial**

Nós sabemos muito bem que desenvolver um novo projeto custa muito tempo e dinheiro, por tanto quanto antes uma má ideia for cancelada menos sofreremos no futuro.

Então nesse passo vamos analisar se realmente vale a pena e caso valha, como vamos rentabilizar o projeto (caso seja o caso).

\*Você deve ter percebido que desenvolvimento é muito mais que código, é visão empreendedora.

1. Na sua pasta Planejamento, crie um arquivo chamado “planejamento comercial”
2. Neste documento responda as seguintes perguntas:
  - a. Quais problemas meu APP resolve?
  - b. Porque ele se diferencia dos que já existem?
  - c. Como o projeto será rentabilizado?

Exemplo:

### **Quais problemas meu APP resolve?**

1. Conecta pessoas que precisam de transporte privado à pessoas que podem oferecer o serviço
2. Faz a troca de dinheiro entre os grupos de maneira confiável e digital
3. Garante um clima de segurança por gerar um histórico dos clientes e motoristas
4. Encontra o motorista mais próximo para atender mais rápido
5. ...

### **Porque ele se diferencia dos que já existem?**

1. Encontra um motorista mais rapidamente
2. Tem pagamento online
3. ...

### **Como o projeto será rentabilizado?**

A cada transação entre um cliente e um motorista nós receberemos 10% do valor.

### Porque esse processo é importante?

A maior parte dos projetos infelizmente são abandonados depois de alguns meses, isso porque os desenvolvedores não conseguem rentabilizá-lo ou porque o projeto simplesmente não se diferenciava dos outros. Esse planejamento tenta diminuir esse problema.

## Quando usar?

Sempre que você puder opinar sobre a viabilidade do projeto (principalmente em projetos pessoais).

## **Passo 4 - Planejamento técnico**

Nessa fase nós definimos as telas que serão necessárias (no caso de um site ou app mobile), modelo do banco de dados, endpoints da API (se for o caso) e quais tecnologias serão necessárias para desenvolver o nosso projeto.

Perceba que antes disso nós definimos como será o projeto e também se ele é viável, agora estamos preparados para definir as tecnologias.

1. Na sua pasta Planejamento crie um arquivo chamado “planejamento técnico”
2. Neste documento execute os seguintes passos:
  - a. Liste as páginas que serão necessárias no seu APP
  - b. Crie os mockups da sua aplicação (no caso de ser um site ou App mobile), eu recomendo o uso do [mockupflow](#)
  - c. Caso uma API seja necessária documente os endpoints que serão usados, eu recomendo o [swaggerhub](#)
  - d. Crie um mapa do banco de dados, eu recomendo o uso do [dbdesigner](#)
  - e. Defina o Stack do projeto (quais ferramentas serão usadas): Docker, Ruby On Rails, PostgreSQL e etc

## Porque esse processo é importante?

Criar uma relação das páginas necessárias e os mockups vão te ajudar a visualizar qual será o resultado do projeto (e também vai te forçar a definir bem o escopo do

seu projeto), documentar os endpoints da sua API (se for o caso) também vai te ajudar a visualizar o resultado do projeto e prever a arquitetura necessária, definir o banco de dados vai reforçar as regras de negócio e finalmente definir o Stack vai permitir que sua equipe prepare o ambiente para usar as tecnologias escolhidas.

\*Não deixe de criar os mockups, eles são fundamentais.

### Quando usar?

Todos os novos projetos precisam passar por essa fase.

## **Passo 5 - Planejamento de entrega**

Todo projeto tem como objetivo ficar online, mas do zero ao deploy pode demorar um longo período se você esperar terminar tudo para subir ele.

Para evitar esse gap tão grande faça entregas parciais (desenvolva a primeira parte útil e entregue, depois a segunda e assim por diante).

Esse documento tem como objetivo organizar a ordem de entrega das coisas.

1. Na sua pasta Planejamento crie um arquivo chamado “planejamento de entrega”
2. No documento defina as fases de entrega do seu projeto (você sempre deve entregar uma parte funcional e que entregue valor)

### Exemplo:

#### **Fase 1:**

1. Funções de pedido de motorista
2. Funções de cancelamento de pedido

3. Funções de pagamento por cartão de crédito
4. ...

**Fase 2:**

1. Funções de visualização dos motoristas no mapa
2. Histórico de corridas
3. Denúncia de corrida fraudulenta
4. ...

Porque esse processo é importante?

Se você está desenvolvendo um APP para um cliente é natural que ele queira testar o projeto durante o desenvolvimento, ter entregas parciais vai mantê-lo feliz (e alinhado com o que está acontecendo) e também vai detectar rapidamente divergências entre o que foi pedido e o que foi feito (erre rápido, corrija rápido).

Se você está desenvolvendo um APP pessoal, ter sempre uma parte online vai te permitir demonstrar a ideia a investidores (e possíveis clientes) e também vai te manter motivado por ver o projeto rodando.

Quando usar?

Sempre que você puder opinar sobre os passos da entrega.

# Preparação

FASE 2

# WorkFlow Super Full Stack

## Fase 2 - Preparação

---

Depois de fazer o planejamento inicial finalmente estamos prontos para começar a preparar o nosso projeto na prática.

Essa preparação tem como objetivo garantir que todas as ferramentas necessárias estejam prontas para hora que começarmos a codar o nosso projeto.

### Passo 1 - Ferramentas de gestão do trabalho

Organizar as ferramentas que servirão para gerenciar o trabalho é uma parte fundamental quando trabalhamos sozinhos ou com uma equipe, nesse passo você deve preparar suas ferramentas favoritas.

Dois tipos de ferramenta são interessantes nesse caso, então:

1. Crie um projeto no seu gerenciador de tarefas
2. Configure seu gerenciador de projetos (opcional)

Para gestão de tarefas eu recomendo:

1. [Meister Task](#)
2. [Trello](#)

Para gestão do projeto (e um pouco mais : ) :

1. [Basecamp](#)
2. [Active Collab](#)
3. [Jira](#)

## **Passo 2 - Preparação das ferramentas de integração**

Nessa parte você vai preparar as ferramentas que os desenvolvedores vão usar no dia a dia para integrar seus códigos e testar sua qualidade (prepare mesmo que você esteja sozinho no projeto).

Três ferramentas são importantes nesse caso, então:

1. Crie um repositório para o projeto
2. Crie um projeto no seu Continuous Integration
3. E finalmente integre seu repositório com um serviço de verificação de qualidade de código

### **Repositório:**

1. Gitlab
2. Github
3. Bitbucket

### **Continuous Integration:**

1. Gitlab (ele serve com CI também)
2. Codeship
3. Circle CI
4. Travis CI

### **Verificação de qualidade de código:**

1. Code Climate

## **Passo 3 - Preparação do ambiente (development)**

Os desenvolvedores envolvidos no projeto precisam preparar suas máquinas para trabalhar no projeto (instalar as bibliotecas necessárias).

Eu recomendo fortemente que as equipes usem o docker + docker-compose para manter um ambiente igual entre todos os desenvolvedores e acelerar o desenvolvimento.

Então nessa fase o docker + docker-compose devem ser instalados se for o caso.

## **Passo 4 - Setup Inicial do Projeto**

Finalmente vamos começar a mexer diretamente no nosso projeto, nesse primeiro contato com o código nós vamos preparar a base do nosso projeto para poder desenvolver com tranquilidade depois.

### **Gere o Projeto**

Caso você utilize um framework como Ruby On Rails use os comandos padrão para gerar o seu projeto com a configuração necessária.

### **Incialize o Git na pasta do projeto**

Incialize o git no seu projeto (git init) e passe a utilizá-lo para subir as alterações para o seu repositório (utilizando as melhores práticas se possível).

### **Instale as principais bibliotecas**

Nessa fase nós instalamos as bibliotecas principais que serão utilizadas pelo projeto, exemplos do mundo do Ruby On Rails: devise, CanCanCan, Materialize-Sass e etc

### **“Dockerize” seu projeto**

Se você estiver usando o docker como recomendado, nessa fase crie seu Dockerfile e docker-compose.yml para gerar um ambiente isolado para o seu projeto.

Pronto, agora estamos preparados para começar \o/

# Looping de Desenvolvimento

FASE 3

# WorkFlow Super Full Stack

## Fase 3 - Looping de Desenvolvimento

---

Agora que temos uma grande solidez na direção que estamos indo (documentação) e uma forma base para começar (projeto preparado), já podemos codar nossa solução.

Diferente das outras fases essa é um looping, ou seja assim que você chegar ao final (entrega) você voltará ao primeiro passo.

Cada volta no looping significa uma entrega parcial, então siga sua documentação de entrega.

### **Passo 1 - Alinhamento**

Verificação do que precisa ser entregue nessa volta do looping (sprint), revisão e ajustes na documentação.

### **Passo 2 - Planejamento de testes**

Nesse ponto você deve planejar (em um documento) quais testes precisam ser realizados para garantir que as features que serão implementadas funcionem como devem (esse passo evita muito retrabalho).

### **Passo 3 - Testes**

Caso você siga o [TDD](#) esse é o momento onde você vai implementar os testes que foram planejados no ponto anterior.

## **Passo 4 - Desenvolvimento**

Chegou a hora de implementar no código as features propostas para essa volta do looping. Caso você esteja trabalhando em equipe cada um dos desenvolvedores vai estar criando o seu próprio conjunto de features nesse momento.

## **Passo 5 - Homologação**

No servidor de staging (um ambiente semelhante ao de production) suba seu código para homologar junto ao cliente o software que foi desenvolvido nessa rodada e que será a entrega parcial.

Caso sejam encontrados problemas, volte para o passo pertinente.

\*Se for a primeira volta do looping e você não possuir um servidor de staging configurado para subir o código, configure.

## **Passo 6 - Entrega parcial**

Finalmente suba para o servidor de production a entrega parcial da volta do looping \o/

\*Se for a primeiro volta do looping e você não possuir um servidor de production configurado para subir o código, configure.

# WorkFlow Super Full Stack

## Resumo do Workflow

---

Um breve resumo do Workflow Super Full Stack para ficar mais prático visualizar e seguir.

### 1. Workflow Super Full Stack

#### a. Fase 1 - Planejamento Inicial

- i. Caos
- ii. Planejamento criativo
- iii. Planejamento comercial
- iv. Planejamento técnico
- v. Planejamento de entrega

#### b. Fase 2 - Preparação

- i. Ferramentas de gerenciamento dos trabalhos
- ii. Ferramentas de integração
- iii. Ambiente de development
- iv. Setup Inicial do Projeto

#### c. Fase 3 - Looping de Desenvolvimento

- i. Alinhamento
- ii. Planejamento de testes
- iii. Testes
- iv. Desenvolvimento
- v. Homologação
- vi. Entrega parcial

# WorkFlow Super Full Stack

## OneBitCode e Bootcamp Super Full Stack

---

### O que é o OneBitCode?

O [OneBitCode](#) inicialmente começou como um Blog sobre programação [focado em Ruby On Rails] mas rapidamente se tornou um veículo de ensino de programação através de posts, screencasts, talks ao vivo, podcasts e cursos.

Nosso trabalho continua e estamos evoluindo constantemente para levar cada vez mais conteúdos de qualidade e propagar a solução de problemas através da tecnologia no Brasil.

Se junte a nossa comunidade para fazermos a diferença juntos:

**Estamos no [Facebook](#), [Youtube](#), [Twitter](#) e [Slack](#)**

### O que é o Bootcamp Super Full Stack?

O Bootcamp Super Full Stack é uma **imersão** de aproximadamente **6 meses** no mundo do desenvolvimento de softwares [focado em Rails], 100% online e foi criado com o objetivo de levar aos alunos, através de aulas práticas e conceitos em vídeos e textos, a experiência e o conhecimento necessário para se destacar no mundo da programação e formar um profissional capaz de lidar com projetos reais.

O Bootcamp Super Full Stack também é uma comunidade onde programadores extremamente motivados se juntam para aprender a parte técnica, desenvolver soft skills e se preparar para construir soluções inovadoras.

Conheça mais sobre ele: [Bootcamp Super Full Stack](#)

# WorkFlow Super Full Stack

## Quem faz o OneBitCode

---



**Leonardo Scorza**

Super Full Stack focado em Rails, criador e escritor do One Bit Code. Entusiasta de todo tipo de tecnologia, em especial Ruby on Rails, ama ensinar e trocar conhecimentos.

[Facebook](#) | [Linkedin](#)



**Flávia Modesto**

Entusiasta da parte audiovisual, prefere o design e produção de vídeos à programação. Cuida dos bastidores e da base do One Bit Code. :)

[Facebook](#) | [Linkedin](#)

Obrigado por fazer parte da  
família One Bit Code!