### 1. SIMULADOR DE DADO

Objetivo: Seu script deve gerar um valor aleatório entre 1 e 6(ou uma faixa que você definir) e permitir que o usuário rode o script quantas vezes quiser.

* Habilidades praticas a aplicar:
* Tratamento de exceções
* Condicionais If/Else
* Input de dados
* Geração de valores
* Print

Detalhes e boas Práticas: Você deve desenvolver um projeto em Python que irá rodar inicialmente na linha de comando e que, ao ser executado, deverá pergunta o seguinte ao usuário: “Você gostaria de jogar o dado?” Ou alguma variação dessa pergunta. Depois de ter feito essa pergunta, o seu programa precisa avaliar a resposta que foi digitado pelo usuário.

Um passo importante aqui é que, quando digo avaliar quero dizer que você precisa receber o valor, tratar quando ele(a) disser que sim ou que não. Seu programa depois deverá fazer a ação necessária de acordo com a resposta que foi entrada pelo usuário. Seu script não deve quebrar ou para de funcionar caso o usuário entre algo que não seja esperado, como, por exemplo, um número. Trate as exceções ou erros para que seu script rode liso e sem problemas.

Caso a resposta a pergunta inicial tenha sido “sim” ou positiva de alguma forma, gere um valor aleatoriamente entre 1 e 6(você pode claro alterar essa faixa) e exiba o número no console para o usuário. Na sequência pergunte se ele(a) quer rodar o script novamente e trate essa situação para que continue rodando enquanto a resposta for positiva, fechando apenas quando for um “não”.

### ~~2. CHUTE O NÚMERO~~

~~Objetivo: Criar um script que gerá um valor aleatoriamente, guarda este valor, e pergunta repetidamente para o usuário chutar o valor gerado até que ele acerte.~~

~~Habilidades praticas a aplicar:~~

* ~~Random~~
* ~~Comparadores matemáticos~~
* ~~Controle de Fluxo~~
* ~~Entrada de dados~~
* ~~Saida de dados~~

~~Detalhes e boas Práticas: Você deve criar um projeto para rodar na linha de comando que ao iniciar, irá gerar, armazenar, porem não exibir um valor aleatório entre um valor mínimo e máximo que será definido por você ( 10-100, por exemplo). Com esse valor gerado e armazenado de alguma forma que você (o criador ou jogador do script) não possa ver, faça uma pergunta do tipo: “Chute um número” para quem estiver rodando o script e com isso o programa deve gravar a resposta que foi passada.~~

~~Depois disso você terá 3 caminhos possíveis: 1. Avisar que a pessoa chutou baixo, 2 dizer que chutou alto ou parabenizar dizer que acertou! Considerando os três possíveis caminhos que podem ser seguidos, você deve cuidar para que em todo o momento a entrada de dados seja tratado para exceções e que caso o usuário digite algo inesperado, que ele receba uma mensagem amigável o informando das possíveis opções que seu programa oferece. Isso deve continuar acontecendo indefinitivamente até que a pessoa acerte o número ou desista por não conseguir acertar hahaha(acontece).~~

### 3. DECIDA POR MIM

Objetivo: Crie um script que responda qualquer pergunta que for feita a ele. Recomendo ter uma base de possíveis respostas (10-20 ou mais). Ex: Será que devo sair de casa hoje? Seu script reponde: “Sim, vai lá!”

Detalhes e boas Práticas:

Habilidades praticas a aplicar:

* Listas
* Random
* Laços de Repetição
* Input de dados
* Saida de dados
* Geração de valores

### 4. JOGO DE AVENTURA

Objetivo: Crie um jogo que levará o usuário a vários possíveis finais de acordo com as respostas que forem passadas para ele.

Habilidades praticas a aplicar:

* Controle de fluxo elaborado
* If/else
* While
* Input de dados
* Saida de dados
* Listas

Detalhes e boas Práticas: Imagine que você vai criar uma história onde o protagonista é a pessoa rodando o seu script e cada pergunta respondida moverá a pessoa na sua história de acordo com as condições que você determinar. As possibilidades aqui são infinitas, você pode ter um jogo bem curto com poucos finais ou uma árvore gigantesca de finais. Receba o valor do usuário e trate a resposta para que respostas não válidas não quebrem o programa. Você deverá permitir que o usuário tente novamente quanto ele chegar em um dos possíveis finais. Encoraje que ele tente várias vezes para ver todos os possíveis finais.

### 5. GUI(PYSIMPLEGUI)

Objetivos: Seus projetos em Python tem uma chance muito maior de serem vistos e notados caso você crie uma interface gráfica(GUI) para todos seus projetos anteriores. Adoro trabalhar com a linha de comando, mas quando quero mostrar meus projetos a melhor forma que encontrei foi criando uma tela para eles. Hoje acredito que a maneira mais intuitiva, rápida e fácil que encontrei para fazer isso foi através da biblioteca PySimpleGUI. Ela permite criar uma interface gráfica com poucas linhas e inclusive criei um [tutorial completo de “Como criar Interfaces em Python usando PySimpleGUI”](https://www.youtube.com/watch?v=Et0fYeA2XxY).

Habilidades praticas a aplicar:

* Criação de Interface
* Design de Interface
* Interatividade

### **Entretenimento**

#### **1. Um web scraper que publica os 10 melhores blogs sem intervenção humana**



Um dos problemas que encontrei em alguns posts de recomendação de projetos para aprender a programar é esses projetos não são implementados de maneira a excitar o programador para continuar o desenvolvimento.

Por exemplo, vejo que muitas pessoas recomendam a construção de um web scraper.

Uma vez que você cria o web scraper e extrair os dados, o que você fará com eles?

Em vez de apenas extrair os dados, por que não criar um site com esses dados? Não precisa ser chique ou ter muitas visualizações.

Esse scraper pode extrair os dados para um banco de dados e selecionar as postagens mais populares. Daí, ele pode copiar o título, junto com algumas frases, e criar uma postagem que ele compartilhe online.

Este seria um projeto impressionante e simples que você pode realmente mostrar.

Aí sim, você pode mostrar que pode fazer mais do que apenas codificar um pequeno segmento de um sistema. Em vez disso, você pode pensar em um sistema inteiro.

Você precisa considerar como automatizar o processo, gerenciar o banco de dados, criar o site e selecionar as postagens. Isso também permite que você realmente tenha um produto final tangível.

Sem um produto final tangível, é muito fácil ficar desmotivado/a e simplesmente parar com apenas um web scraper.

Além disso, vai saber, talvez o seu site fique popular!

Habilidades: banco de dados, web scraper, automação, desenvolvimento web (para o blog) e programação geral.

#### **2. Um sistema de alerta de eventos usando APIs do Meetup e Eventbrite**

Você já quis ir a um stand-up ou show de alguma banda, mas percebeu que era na semana anterior?

Talvez tenha ocorrido um evento gratuito perto de você sobre data science ou big data e você tenha perdido, porque esqueceu de verificar.

Por que não criar seu próprio agregador usando as APIs do Meetup e Eventbrite, que avisarão quando as palavras-chave estiverem nas descrições ou nos títulos dos eventos?

Eu suponho que o Meetup e o Eventbrite já tenham opções semelhantes. Mas é sempre mais divertido tentar criar seu próprio sistema.

Você pode personalizar o sistema para funcionar da maneira que desejar e, talvez, até permitir que outras pessoas façam seus próprios alertas, se você tornar o projeto em um site.

O que gostamos nesse projeto é que você pode praticar o trabalho com duas APIs diferentes. Isso permitirá identificar e comparar o que você gosta e o que não gosta neles.

Dessa forma, se você precisar criar uma API, terá um melhor cenário do que funciona e não funciona.

Habilidades: APIs, banco de dados, automação, desenvolvimento web e programação geral.

#### 3. **9GAG copy cat**

Você nem sempre precisa tentar reinventar a roda ao criar seus próprios projetos.

Projetos simples, como um site que permite fazer login, publicar fotos, GIFs e rolar por um feed, oferecem a oportunidade de criar uma base de site primeiro.

Depois, você pode adicionar muitos recursos interessantes, como seguir, curtir e pesquisar.

A pesquisa, em particular, seria uma grande chance de aprender como os sistemas de recomendação e machine learning funcionam!

É sempre divertido tentar replicar sites populares. Na verdade, é realmente uma ótima maneira de aprender a programar, porque você precisa fazer a engenharia reversa de cada recurso.

A engenharia reversa é uma grande habilidade, pois como engenheiro/a de software você manterá constantemente o código de outras pessoas e precisará entender o que se passa na cabeça delas.

Habilidades: machine learning (para sistema de recomendação), banco de dados, automação, desenvolvimento web e programação geral.

#### **4.**Um aplicativo de recomendação de presente

Você já passou perrengue para conseguir encontrar o presente certo para um/a amigo/a?

E se você pudesse criar um site que ajudasse a prever o que comprar para um presente? Você pode permitir que o/a usuário/a final crie uma conta ou apenas receba uma recomendação de presente.

Novamente, isso permite a oportunidade de criar uma conta que requer autenticação, desenvolvimento de banco de dados, etc.

Além disso, outra coisa legal desse projeto é que você pode usar a API da Amazon para links de afiliados. Isso permitirá que você faça algumas coisas.

1. Aprenda como usar APIs e fique à vontade com a leitura da documentação da API;
2. Se você fizer bem, poderá obter uma comissão por cada produto que alguém comprar.

Este projeto também dá oportunidade de tentar criar um protótipo básico de machine learning.

Você pode criar um quiz que tente descobrir qual é o melhor presente e, com base nos cliques que as pessoas derem - ou não - no presente, orientar o protótipo a aprender com base na taxa de resposta.

Habilidades: APIs, banco de dados, programação geral e desenvolvimento de aplicativos.

#### **5. Um site para permuta e negociação**

Pense no [OfferUp](https://about.offerup.com/) (ou no Mercado Livre), mas ao invés de dinheiro, por que não criar um site que permita transações apenas?

Esse conceito forçará você a desenvolver vários recursos que precisam de reflexão. Você não poderá simplesmente botar a mão na massa neste projeto sem um plano.

Como as pessoas postarão, onde encontrarão itens postados recentemente e como elas irão procurar. Todos esses são recursos separados que você pode criar.

Além disso, você precisa pensar em como os/as usuários/as irão interagir e talvez até em como farão as negociações.

A ideia não precisa ser 100% prática para a vida real - ela só precisa ser prática no sentido de melhorar suas habilidades e te ajudar a aprender a programar.

Habilidades: banco de dados, desenvolvimento web, programação geral e desenvolvimento de aplicativos (se você optar por torná-lo um aplicativo).

#### **7. Sistema de gerenciamento de tarefas**

Os painéis de tarefas como o KanbanFlow são criados com vários recursos modulares que o tornam um ótimo projeto para brincar e aprender a programar.

Ok, vai dar um pouco de trabalho para começar, pois você precisará configurar uma UI que seja robusta e dinâmica também.

Na verdade, este projeto seria um trabalho para duas pessoas. Uma pessoa para trabalhar no front-end e outra pessoa para trabalhar no back-end.

Mas, não deixe que isso te desencoraje!

Isso é, na verdade, uma chance para você trabalhar em suas habilidades de [comunicação](https://gama.academy/blog/carreira/comunicacao-assertiva-cnv/) e trabalho em equipe.

Você precisará conversar sobre os projetos para garantir que ambos/as entendam completamente e saibam onde seus módulos irão se conectar.

Isso é sempre mais desafiador do que parece.

Leia também: [O que são soft skills e como isso pode acelerar minha carreira?](https://gama.academy/blog/carreira/o-que-sao-soft-skills/)

Habilidades: Comunicação, front-end, banco de dados, desenvolvimento web e programação geral.

#### **8. Um quadro de empregos**

Qualquer projeto que te force a permitir que as pessoas usuárias insiram como vários tipos de usuários, adiciona um aspecto interessante do design.

Como você garantirá que a maneira como os/as empregadores/as utilizam o site atenda às suas necessidades em relação à prospecção de empregos?

Como a maioria dos outros projetos, você não precisa se concentrar em tudo de uma só vez. Comece tentando criar a capacidade de criar um anúncio de emprego.

Então, você pode se concentrar nas pessoas que buscam emprego e em como elas respondem.

Habilidades: banco de dados, desenvolvimento web e programação geral.

#### 9. Um site que prevê lucros com base em conjuntos de dados padronizados

Existem muitos conjuntos de dados que são bastante padronizados para a maioria das empresas.

Isso inclui dados contábeis, geralmente baseados em centros de custo, contas, descrições de linhas e, finalmente, o custo real da transação.

O que é ótimo na padronização de qualquer conjunto de dados é que isso facilita a criação de análises sobre eles.

Por que não criar um painel padronizado que possa ajudar as empresas a prever gastos, despesas mensais e, possivelmente, ajudá-las a melhorar seus gastos?

Para este projeto, você provavelmente precisará gastar muito tempo aprendendo sobre como manter seus dados protegidos.

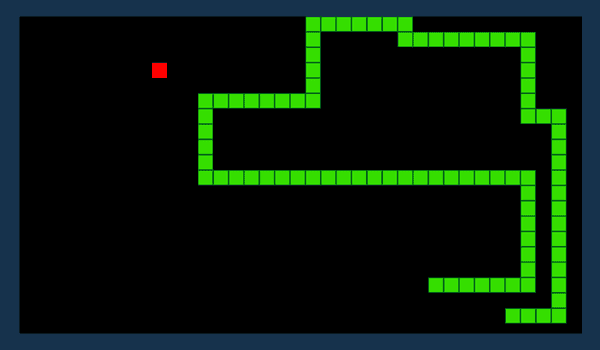
Obviamente, recomendo primeiro tentar criar os módulos que se concentram na recuperação, padronização e exibição dos dados - antes de se aprofundar na segurança.

É uma toca de coelho da qual você nunca pode escapar!

Habilidades: Previsão, lógica de negócios, banco de dados, desenvolvimento web e programação geral.

### **Ideias de jogos**

#### 10. Snake (ou "Cobrinha")



Se você teve um celular no início dos anos 2000, provavelmente jogou Snake.

É um jogo simples, ok, mas você sempre pode tentar tornar as coisas mais complexas!

Primeiro, comece apenas tentando fazer o jogo.

Isso exigirá que você descubra como desenvolver um jogo online. Esta serpente de néon do Sebastian Opperman é uma ótima forma de começar.

Mas, depois disso, talvez você possa adicionar novos recursos como itens ou poderes especiais.

Essa seria uma boa chance de se divertir. Este projeto não é tão técnico do ponto de vista de ter muitos/as usuários/as que se inscrevam e usem seu site.

No entanto, é um bom desafio para descobrir como fazer um jogo rodar online.

Habilidades: desenvolvimento Web, programação geral e UI;

## Projeto 1: Dashboard de Usuário



Aplicativo CRUD simples, ótimo para fundamentos. O que você vai aprender:

* Interagindo com um banco de dados – criando, lendo, editando, excluindo usuários.
* Criando e gerenciando usuários.
* Input Validation e como trabalhar com formulários.

## Projeto 2: Hackernews

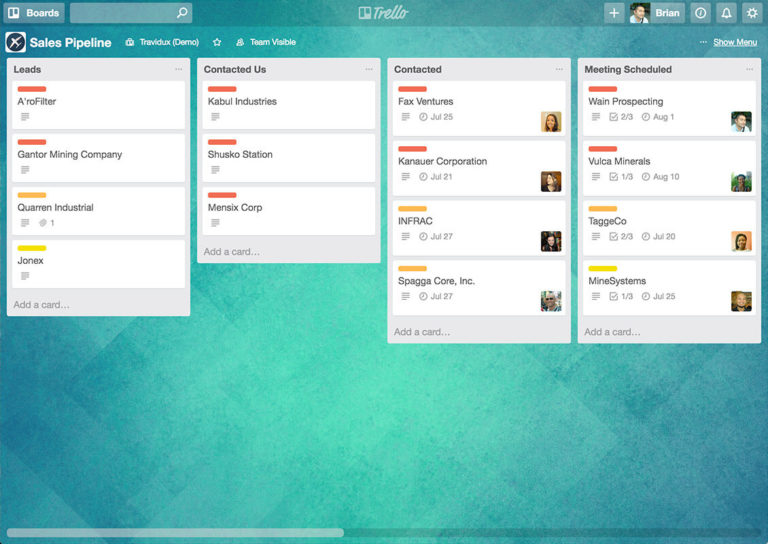


Todos criaram sua própria versão de notícias sobre hackers (você deve criar também).

O que você vai aprender:

* Interagindo com a API hackernews.
* Criando aplicativos de página única.
* Roteamento.
* Como implementar recursos como visualização de comentários, comentários únicos, perfis.

## Projeto 3: Trello



O que você aprenderá ao criar um clone do trello:

* Criando novos objetos (placas, listas, cartões).
* Manipulando inputs e validação.
* Arrastar e soltar.
* Roteamento.
* Como usar o armazenamento local, salvando dados no armazenamento local, lendo dados do armazenamento local.
* Como usar bancos de dados, salvando dados no banco de dados, lendo dados do banco de dados.

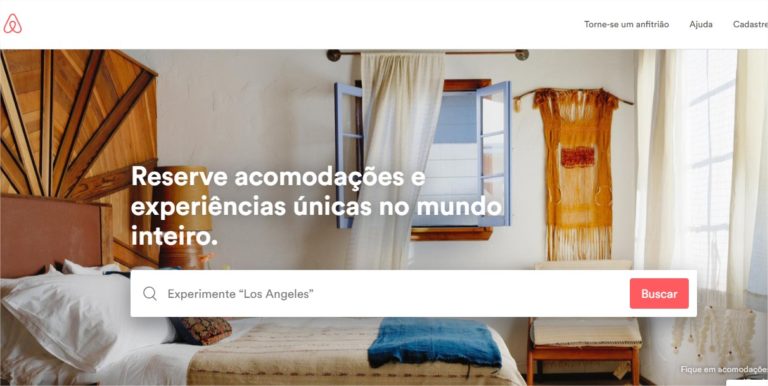
## Projeto 4: Cryptocurrency tracker (native mobile app)

Tem que ser um aplicativo nativo – Swift, Objective-C, React Native, [Java](https://programathor.com.br/jobs-java), Kotlin permitido.

O que você vai aprender:

* Como os aplicativos Native funcionam.
* Buscando dados de uma API.
* Como trabalhar com simuladores moblies.
* Como funcionam os layouts Native.

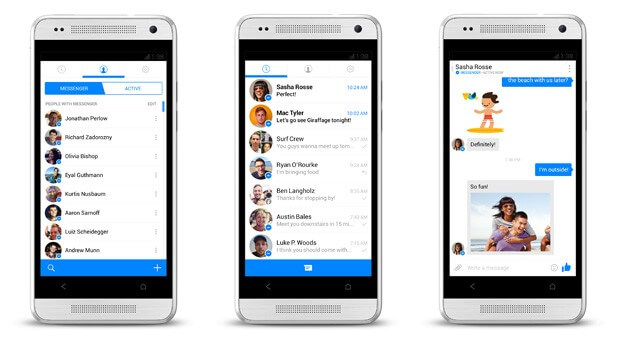
## Projeto 5: Airbnb



O que você aprenderá ao criar um clone do Airbnb:

* Cartografia/Maps (utilizando API google)
* Como aplicar GPS/Coordenadas.
* Search Bar.
* Usar Datepicker.

## Projeto 6: Messenger (aplicativo native)



Entender como aplicativos native e aplicativos da Web funcionam fará com que você se destaque facilmente no meio de tantos programadores.

O que você vai aprender:

* Web sockets (mensagens instantâneas).
* Como os aplicativos native funcionam.
* Como os layouts funcionam em native.
* Roteamento para aplicativos native.

Agora é com você! Escolha um ou alguns dos projetos acima e trabalhe duro, mesmo que isso leve mais de um mês para ser concluído. O tempo que levará não importa, mas sim o que você aprenderá com essas aplicações. Não se esqueça que isso te deixará preparado para testes em empresas e para colocar em destaque no seu portfólio.