Conteúdo dos cursos

Módulo 1

- **Motor:** NodeJS
- **Linguagem de programação:** JavaScript
- **Foco do módulo:** Lógica de programação
- **Conteúdo:**
  - Variáveis (`let`, `const`, `var`)
    - `let`: mutável, não pode referenciar antes da declaração
    - `const`: imutável, não pode referenciar antes da declaração
    - `var`: mutável, pode referenciar antes da declaração. Não é mais tão utilizada, mas serve para variáveis globais.
  - Tipos de dados (`string`, `number`; apresenta `boolean`)
  - Operadores matemáticos (`+`, `-`, `*`, `/`, `%`)
  - Estruturas de controle (`if`, `else`, `else if`; usa `boolean`)
  - Funções (`function` e `arrow function`)
  - Listas
  - Laços (`for index`, `while`, `for in`, `for of`, `forEach`)
  - Objetos
  - `Promises` - `async` / `await`
  - `JSON`
    - `JSON.parse()`
- **IDE:**
  - **Primeira semana:** GitPod com NodeJS (interface do VSCode online)
    - Fazer um estudo se o GitPod tem um workspace mais clean de NodeJS. Atualmente o que tem lá é com TS e tá bem cheio de coisa.
  - **Segunda semana:** VSCode Desktop
- **VSC:**
  - **1ª semana:** Git direto no site do GitHub
    - Se for com o GitPod, será integrado.
    - Se for VSCode offline, podemos pensar se vale arrastar e soltar arquivo.
  - **2ª semana em diante:** Git no VSCode
- **Preparando o setup:** pré-aula/welcome class da Blue

Módulo 2

- **Motor:** NodeJS
- **Framework:** Express - Renderização de HTML
- **Linguagem de programação:** JavaScript
- **Linguagens de Frontend:** HTML e CSS

- **Banco de Dados:** SQL (Prisma - JavaScript)
- **Cloud:** Heroku
- **VSC:** Continua no Git no VSCode, apresenta Git CLI.
    - **Passar conteúdo extra:** GitHub Desktop (recomendação de vídeo, etc)

Backend JS

Backend 3

- **Motor:** NodeJS
- **Framework:** Express - Renderização de JSON
- **Linguagem de programação:** JavaScript
- **Banco de Dados:** MongoDB (Mongoose)
- **Cloud:** Heroku + Mongo Cloud Atlas
- **ORM:** Prisma - JavaScript
- **Testes:** unitários e integrados - Jest

Backend 4

- **Motor:** NodeJS
- **Framework:** NestJS (com Express por baixo)
- **Linguagem de programação:** TypeScript
- **Banco de Dados:** SQL - PostgreSQL
- **ORM:** Prisma - TypeScript
- **Cloud:** AWS (EC2 - Backend; ElephantSQL - Banco de Dados)
- **Testes:** unitários e integrados - Jest

Frontend

Frontend 3

- **Motor:** NodeJS
- **Framework:** ReactJS
- **Linguagem de programação:** JavaScript
- **Integração com API:** JSON - `GET` (Read All & Read By Id)
- **Cloud:** Heroku

Frontend 4

- **Motor:** NodeJS
- **Framework:** NextJS
- **Linguagem de programação:** TypeScript
- **Integração com API:** JSON - `GET`, `POST`, `PUT`, `DELETE` (Read All & Read By Id, Create, Update, Delete)
- **Cloud:** AWS - EC2 + RDS

Fullstack

Full stack 3

- **Motor:** NodeJS
- **Framework:** Express (Backend) + ReactJS (Frontend)
- **Linguagem de programação:** JavaScript
- **Banco de Dados:** NoSQL - MongoDB
- **ORM:** Mongoose - JavaScript
- **Integração com API:** JSON - `GET` (Read All & Read By Id)
- **Cloud:** Heroku + MongoDB Cloud Atlas

Full stack 4

- **Motor:** NodeJS
- **Framework:** Nest (Backend) + Angular (Frontend)
- **Linguagem de programação:** TypeScript
- **Banco de Dados:** SQL - PostgreSQL
- **ORM:** Prisma - TypeScript
- **Integração com API:** JSON - `GET`, `POST`, `PUT`, `DELETE` (Read All & Read By Id, Create, Update, Delete)
- **Cloud:** AWS (S3 e CloudFront - Frontend; EC2 - Backend; ElephantSQL - Banco de Dados)