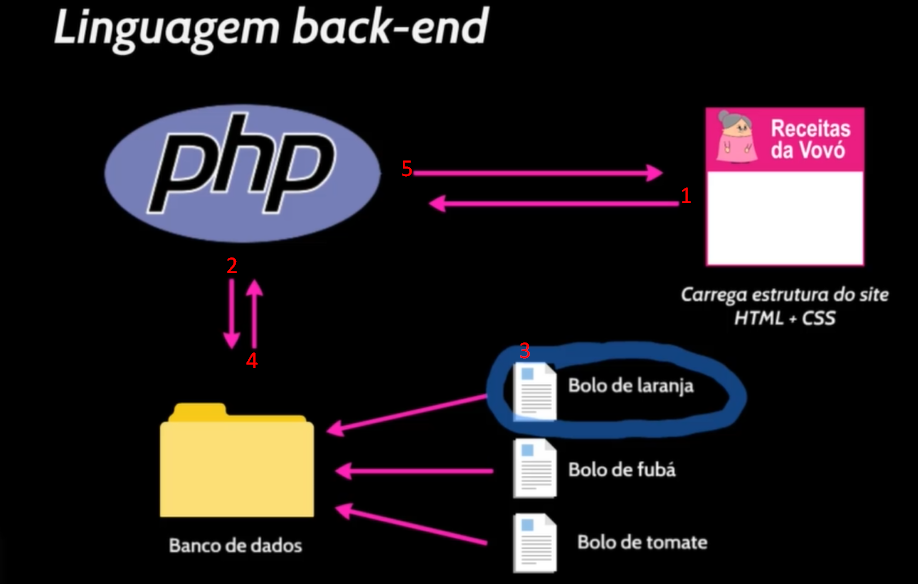
Aluno: Erivaldo Barros

Com base nos vídeos 1 e 2, explique como a web funciona a partir do ponto de vista do protocolo cliente servidor. Crie o começo de um projeto em node.js, usando o npm e informe os comandos utilizados.

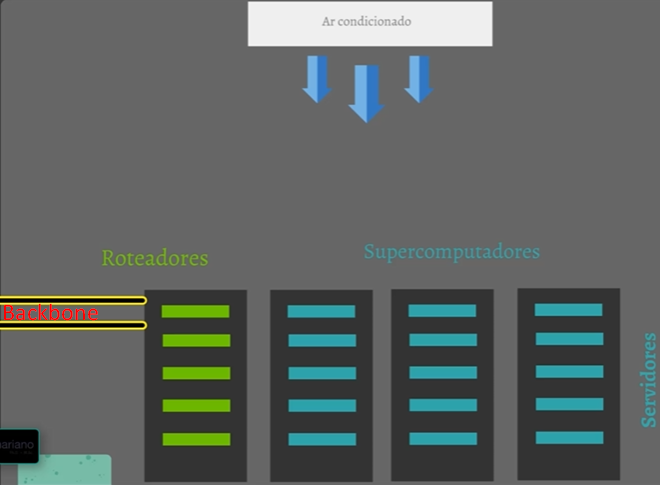
Os exemplos a seguir é uma forma "generalizada" de como a Internet funciona na prática.



Quando você acessa a **Internet** através de sua rede sem fio (*wireless*) ou cabeada o seu computador fará uma conexão com a Internet por meio de um roteador/modem que por sua vez fará com que a conexão de sua casa através da rede de cabeamentos que levará até o seu provedor que ficará responsável por conectar a sua máquina a um **Backbone** que é considerado uma "espinha dorsal da internet", são sistemas de cabeamentos que conecta computadores em uma grande rede mundial.

Então o backbone, por exemplo, fará a conexão com **Data Center** ou também conhecido como **CPD** (Centro de Processamento de Dados), que é um ambiente organizado com uma série de dispositivos eletrônicos, computadores, switches, roteadores etc... Um data center possui uma série de proteções físicas (invasores, incêndios, terremotos etc...) e lógicas (firewalls, ids, ips etc...) contra invasões ou falhas humanas, além de possuir múltiplas fontes de energia para manter o funcionamento caso ocorra alguma falha no fornecimento de energia. Com segurança de acessos restritos aos componentes e salas para reduzir problemas causados por humanos.

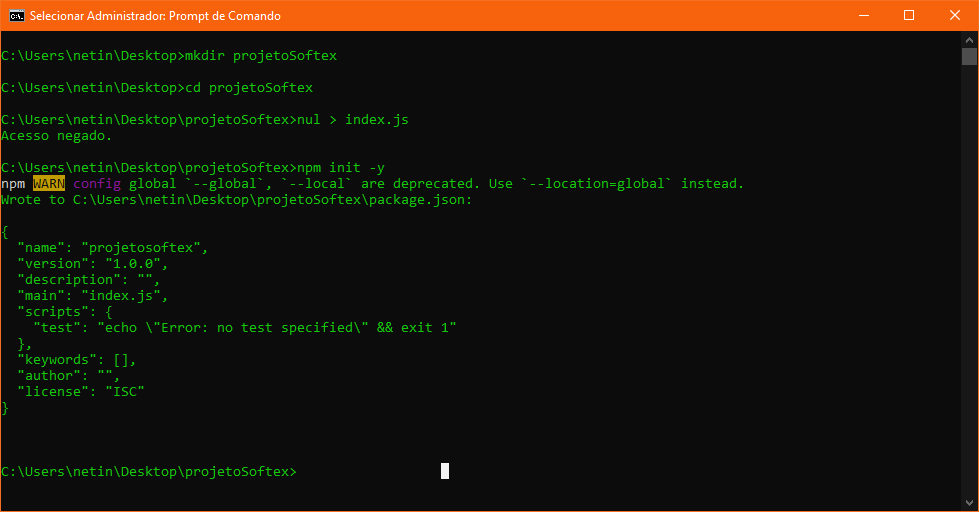
Ainda no data center, ele deve possuir **múltiplas conexões** com a internet através do backbone para evitar gargalos (travamentos) na rede, abaixo veremos uma estrutura de como uma sala de data center deve se comportar, com temperatura controlada por ar condicionados para reduzir desgastes de seus componentes, roteadores, servidores (supercomputadores) que fornecem algum serviço para usuários remotos, como um servidor web (HTTP), servidor de arquivos (FTP) e etc...



O Data Center possui conexões com outros Data Centers do país, que se conectarão a outros continentes por cabos submarinos e pelo mundo WWW.

**Introdução a criação de Projeto**

Primeiramente iremos montar a estrutura de diretórios do nosso projeto, utilizando o prompt de comandos:



O primeiro comando foi criado uma pasta **projetoSoftex** e logo em seguida entrei nela.

Depois foi criado o arquivo **index.js** e utilizado o comando **npm init -y**, para iniciar um projeto, o **-y** é apenas para dar *yes* “sim” em todas as configurações seguintes, para podermos alterar logo depois no **VSCODE**.

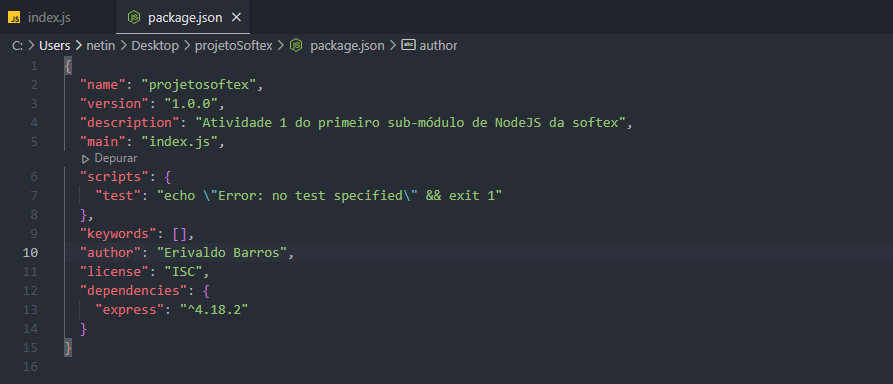
Aproveitando que estamos já dentro do diretório do index, basta dar code index.js para irmos direto para o VSCODE.

Antes de iniciar a programação no index, vamos instalar as dependências de rotas para o mini-projeto:

A biblioteca express faz com que o uso da comunicação com as rotas do programa na web seja mais fácil e rápido de ser configurado, para isso iremos baixa-la e importar no arquivo **index.js** e fazer um teste de rota:

* npm install express

No arquivo **package.json** poderá ser visto que o express já está catalogado em “***dependicies***” e aproveitei para por as devidas configurações nos respectivos campos de ***name*** e ***description***.



Para o uso básico do express irei utilizar os comandos:

const express = require('express');

const app = express();

app.get("/", (*req*,*res*) => {

*res*.send("Teste de rota GET!");

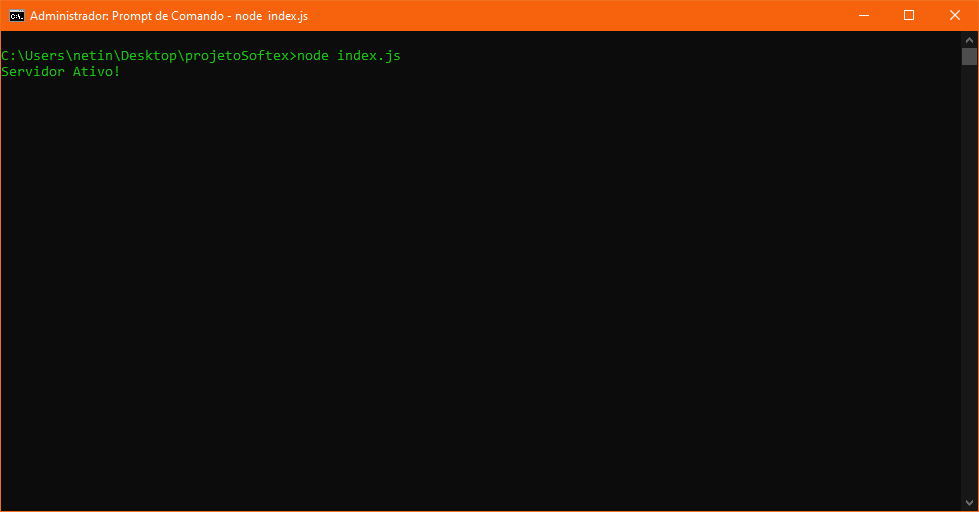
});

app.listen(3333, () => console.log('Servidor Ativo!'));

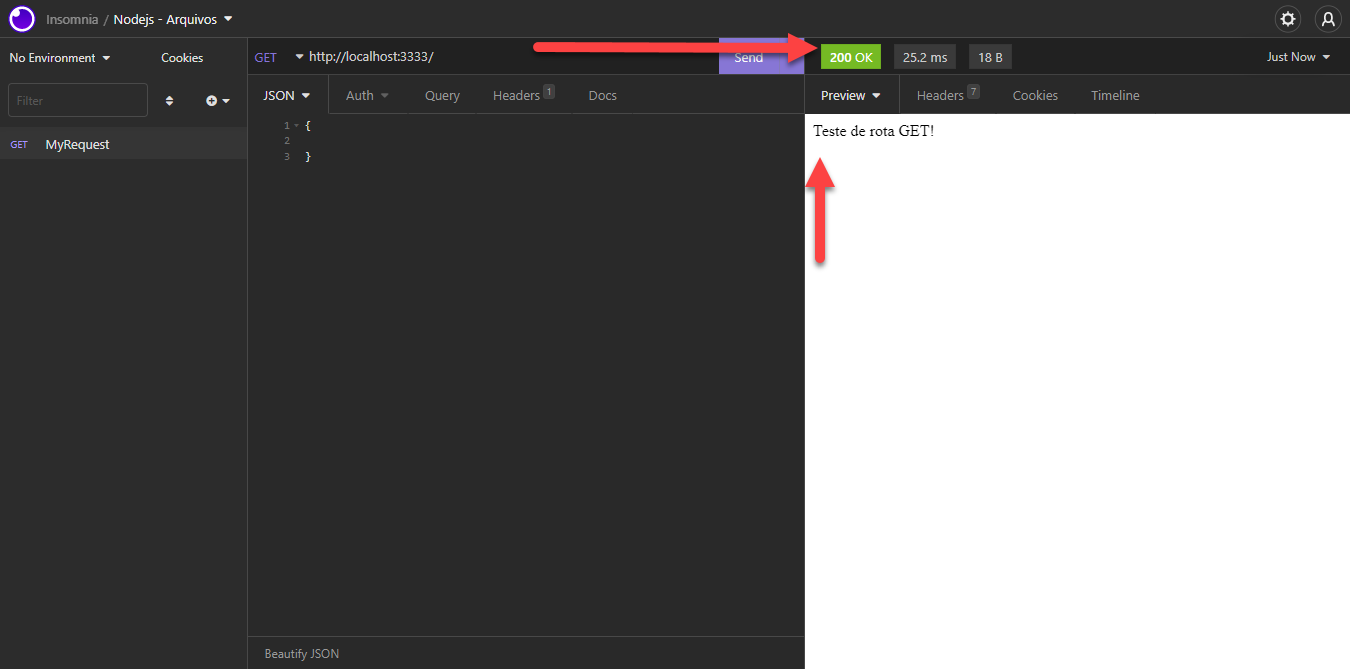
O **1**primeiro comando estou **importando o express**. No **2**segundo atribuindo o express à **constante app**, por boas práticas.

No **3**terceiro o “***app.get***” estou criando uma rota na raiz **“/”** com os parâmetros de requisição e resposta **(req,res)** e enviando **(res.send)** a mensagem (“Teste de rota GET!”), para leitura da informação no servidor.

Por fim o **4**“**app.listen**” para liberar a **porta 3333** para o acesso às rotas a serem criadas, podendo ser utilizada pelo endereço socket [**http://localhost:3333/**](http://localhost:3333/)**.**

Para testar se o servidor HTTP está ativo basta rodar no *prompt* o **node index.js**: 

**Validando a rota no INSOMNIA:**



Teste validado com sucesso! *Status code*: **200 OK**.