WEB

mentoriaT3ch

tecnologia, tendência & transformação.

Sobre o autor

Giovanni de Carvalho

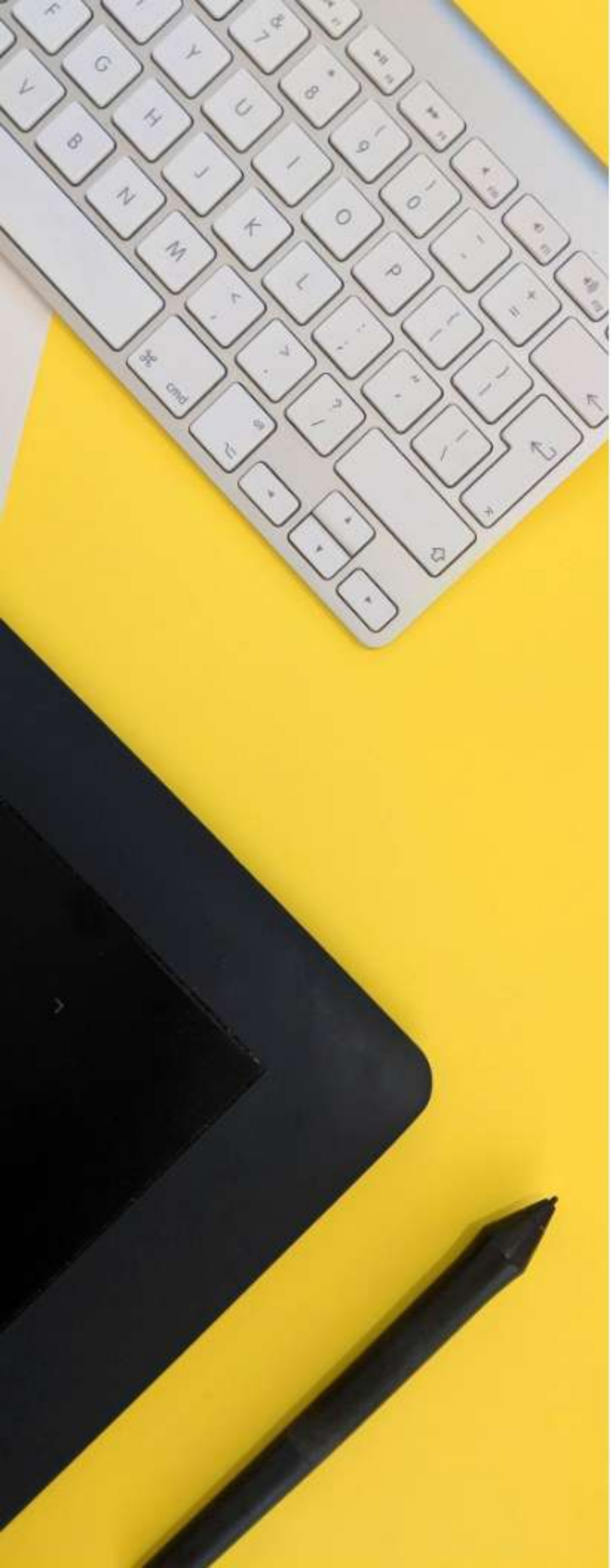
Minha carreira iniciou em 2016 quando comecei a cursar um curso de manutenção de computadores pelo Instituo do Banco do Brasil, ali já tinha decidido que ia seguir na área da tecnologia. Entre idas e vindas em 2017 comecei a trabalhar em uma empresa de Telecom no setor administrativo e consequentemente iniciei uma graduação de Administração de Empresas na Universidade Nove de Julho. Após 2 anos de curso, estava atuando em uma multinacional e sendo destaque em sala de aula com palestras e projetos que foram patrocinados por grandes empresas como B3 e Coca-Cola.

Bom depois de tudo isso você deve estar pensando, que mais um jovem pode querer? Bom eu ainda quero muito mais e um destes enormes desejos era retornar a área de tecnologia.

Conquistei uma bolsa de estudos no centro de inovação do Facebook, para estudar programação Web, não pensei duas vezes! Pedi demissão do meu emprego para agarrar um dos meus sonhos. De lá para cá fui aluno do Senai, realizei diversos cursos e estou no último ano de uma graduação de Análise e Desenvolvimento de Sistemas. Trabalhei em um das Startups mais reconhecidas do país e hoje trabalho no maior Banco da América Latina.

Tenho experiência em engenharia de software com integração e modelagem de sistemas, transformação digital e modernização de plataformas legadas atuando com tecnologias como: Html5, Css3, Javascript, Saas, Bootstrap, Materialize, Angular, React, Spring Boot, Django, NodeJs, MySQL, Db2, Oracle, Mainframe, Mvc, Rest, Docker. Além de experiência em: administração de equipes, administração de processos gerando KPI's relatórios e análise de dados para melhorias contínuas.

> “Feito é melhor do que perfeito” e é por isso que uma ideia pior do que a sua está fazendo sucesso
>
> - Mark Zuckerberg



Índice Geral

MÓDULO 1

visão geral, introdução

Bóra entender um pouco sobre Web

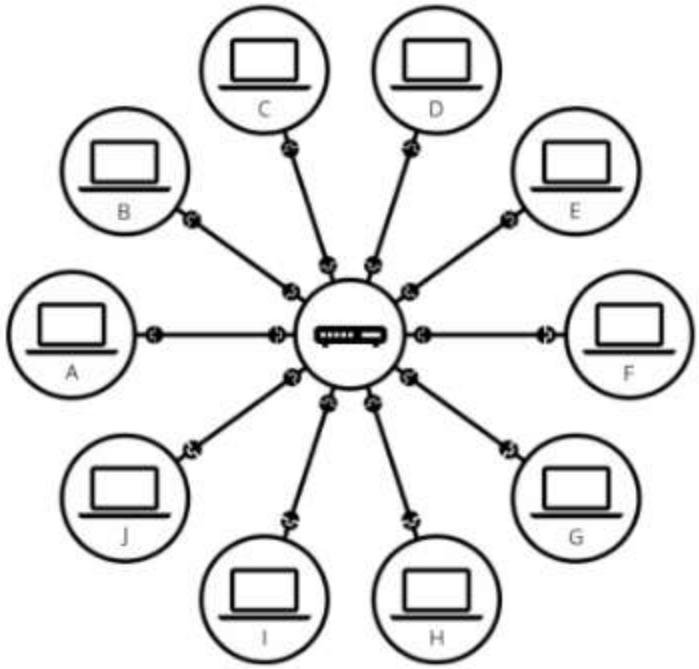
A Internet é a espinha dorsal da Web, a infraestrutura técnica que faz a Web possível. Mas basicamente, a Internet é uma gigantesca rede de computadores que se comunicam juntos.
Ela começou nos anos 60 como um projeto de pesquisa consolidado pelo exército norte americano, e tornou-se uma infraestrutura pública nos anos 80 com o suporte dado por diversas universidades públicas e companias privadas.

Uma rede simples

Quando dois computadores precisam se comunicar, você precisa conectá-los, seja fisicamente (normalmente com um Cabo de rede) ou de uma forma sem fio (por exemplo com sistemas WiFi ou Bluetooth). Todos os computadores modernos suportam alguma(s) dessas conexões.

Uma rede de redes

Por enquanto, tudo bem. Mas como conectar centenas, milhares, bilhões de computadores? Claro que um unico roteador não pode se adaptar para tanto, mas, se você ler com cuidado, nós dissemos que um roteador é um computador como qualquer outro, então o que nos impede de conectar dois roteadores juntos? Nada, então façamos isto.

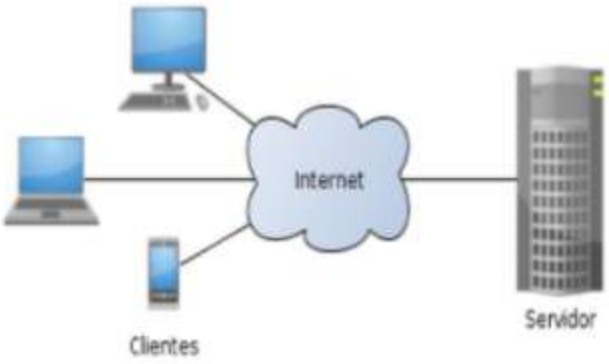


MÓDULO 2

cliente - servidor

Cliente - servidor

A arquitetura cliente servidor é uma arquitetura de aplicação distribuída, ou seja, na rede existem os fornecedores de recursos ou serviços a rede, que são chamados de servidores, e existem os requerentes dos recursos ou serviços, denominados clientes.



2.Modelo Cliente-Servidor

O cliente não compartilha nenhum de seus recursos com o servidor, mas no entanto ele solicita alguma função do servidor, sendo ele, o cliente, responsável por iniciar a comunicação com o servidor, enquanto o mesmo aguarda requisições de entrada.

3.Requisições e respostas

MÓDULO 3

métodos http

WEB

O que é o HTTP? Como funcionam requests e responses?

Quando você digita um endereço no navegador, antes dele quase sempre vem um "http://", mas você já parou pra pensar no que isso significa?

Como funcionam as aplicações web
Esse site aqui que você acessa nesse momento, o meu site. Os dados que aparecem na sua tela não estão em seu computador, certo?

Eles estão em outro computador, que chamamos de servidor.

Pra acessar essas informações, seu navegador precisa de alguma forma pedir os dados para o servidor.

O HTTP request

Com o IP em mãos, seu navegador abre uma conexão TCP com o servidor e envia dados pra ele.

Mas que tipo de dados?
Texto. Um monte de texto.

Mas esse texto precisa seguir algumas regras. Regras estabelecidas pelo Protocolo HTTP.

Esse texto estruturado baseado em uma série de regras é o que chamamos de Request, ou requisição em português.

Um Request tem 3 informações:
-Uma Request Line
-Um Request Header
-Um Request Body

Request Line

No Request line, temos: o método HTTP, a localização do recurso e a versão do protocolo HTTP que estamos usando.

GET /mentoriatech.com HTTP/1.1

Request header

No Header, ou cabeçalho da requisição, enviamos algumas informações referentes àquela requisição, como informações do host, política de cookies, encoding, tipo de resposta aceita e cache.

Host: mentoriatech.com
Connection: close
Accept: text/html

Request body

O Request Body, ou corpo da requisição, é onde geralmente enviamos dados que queremos gravar no servidor.

Não é muito utilizado em requisições do tipo GET, mas sim nas do tipo POST e PUT.

É no corpo da requisição onde você envia dados de um formulário de cadastro em seu site.

O HTTP Response

Depois que o servidor processa sua requisição, ele precisa devolver uma resposta. E isso é feito através de um Response, que nada mais é do que um monte de texto vindo do caminho inverso: do servidor para seu computador.

Esse monte de texto geralmente vai estar em um formato que seu navegador consiga entender: HTML, CSS, Javascript ou até outros formatos como JSON.

Além disso, a Response também vem com o que chamamos de Status Code, que informa o que aconteceu com a requisisição que você mandou.
Os Status Code tem um valor que varia de 100 a 500.

Status Code

100: as que começam com 100 são apenas informativas.
200: a partir de 200 significa que a requisição foi bem sucedida.
300: as com 300 querem te falar pra você fazer um redirecionamento, ou seja, uma segunda requisição. Pode ser porque o endereço está errado ou o recurso que você está tentando acessar está em outro endereço.
400: aqui significa que tem algum erro na requisição.

500: se deu erro 500 e alguma coisa, é erro no servidor.

MÓDULO 4

principais métodos http

WEB

Principais métodos http

POST

O verbo POST é mais frequentemente utilizado para criar novos recursos. Na criação bem-sucedida, retornar o status HTTP 201.

GET

O método HTTP GET é usado para ler ou recuperar uma representação de um recurso. Em caso de sucesso, retorna uma representação em JSON e um código de resposta HTTP de 200 (OK). Em caso de erro, ele geralmente retorna um 404 (NOT FOUND) ou 400 (BAD REQUEST).

PUT

PUT é mais utilizado para substituir (ou atualizar) recursos, executando a requisição para uma URI de recurso conhecido, com o corpo da requisição contendo a representação recém-atualizada do recurso original.

PATCH

PATCH é usado para modificar parcialmente os recursos. A requisição só precisa conter as alterações específicas para o recurso, não o recurso completo.
Se parece com PUT, mas o corpo contém um conjunto de instruções descrevendo como um recurso no servidor deve ser modificado para produzir uma nova versão.

DELETE

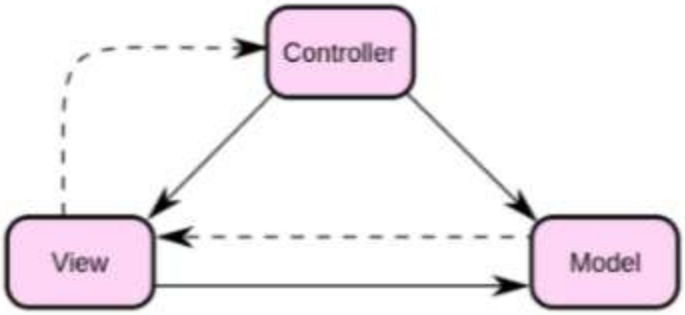
DELETE é bastante fácil de entender. Ele é usado para excluir um recurso identificado por um URI.
Na exclusão bem-sucedida, devolve o status HTTP 200 (OK) ou o status HTTP 204 (NO CONTENT) sem corpo de resposta.
Operações DELETE são idempotentes.

Endpoint	Método	Ação
/users	GET	Retorna a lista de usuários
/users	POST	Insere um novo usuário
/users/{id}	GET	Retorna o usuário com id = {id}
/users/{id}	PUT	Substitui os dados do usuário com id = {id}
/users/{id}	PATCH	Altera items dos dados do usuário com id = {id}
/users/{id}	DELETE	Remove o usuário com id = {id}

MÓDULO 5

padrão mvc

Padrão MVC



Tradicionalmente usado para interfaces gráficas de usuário, esta arquitetura tornou-se popular para projetar aplicações web e até mesmo para aplicações móveis, para desktop e para outros clientes. Linguagens de programação populares como Java, C#, Object Pascal/Delphi, Ruby, PHP, Java Script e outras possuem frameworks MVC populares que são atualmente usados no desenvolvimentos de aplicações web.

Model

Modelo é a ponte entre as camadas Visão (View) e Controle (Controller), consiste na parte lógica da aplicação, que gerencia o comportamento dos dados através de regras de negócios, lógica e funções. Esta fica apenas esperando a chamada das funções, que permite o acesso para os dados serem coletados, gravados e, exibidos.

É o coração da execução, responsável por tudo que a aplicação vai fazer a partir dos comandos da camada de controle em um ou mais elementos de dados, respondendo a perguntas sobre o sua condição e a instruções para mudá-las. O modelo sabe o que o aplicativo quer fazer e é a principal estrutura computacional da arquitetura, pois é ele quem modela o problema que está se tentando resolver. Modela os dados e o comportamento por trás do processo de negócios. Se preocupa apenas com o armazenamento, manipulação e geração de dados. É um encapsulamento de dados e de comportamento independente da apresentação.

View

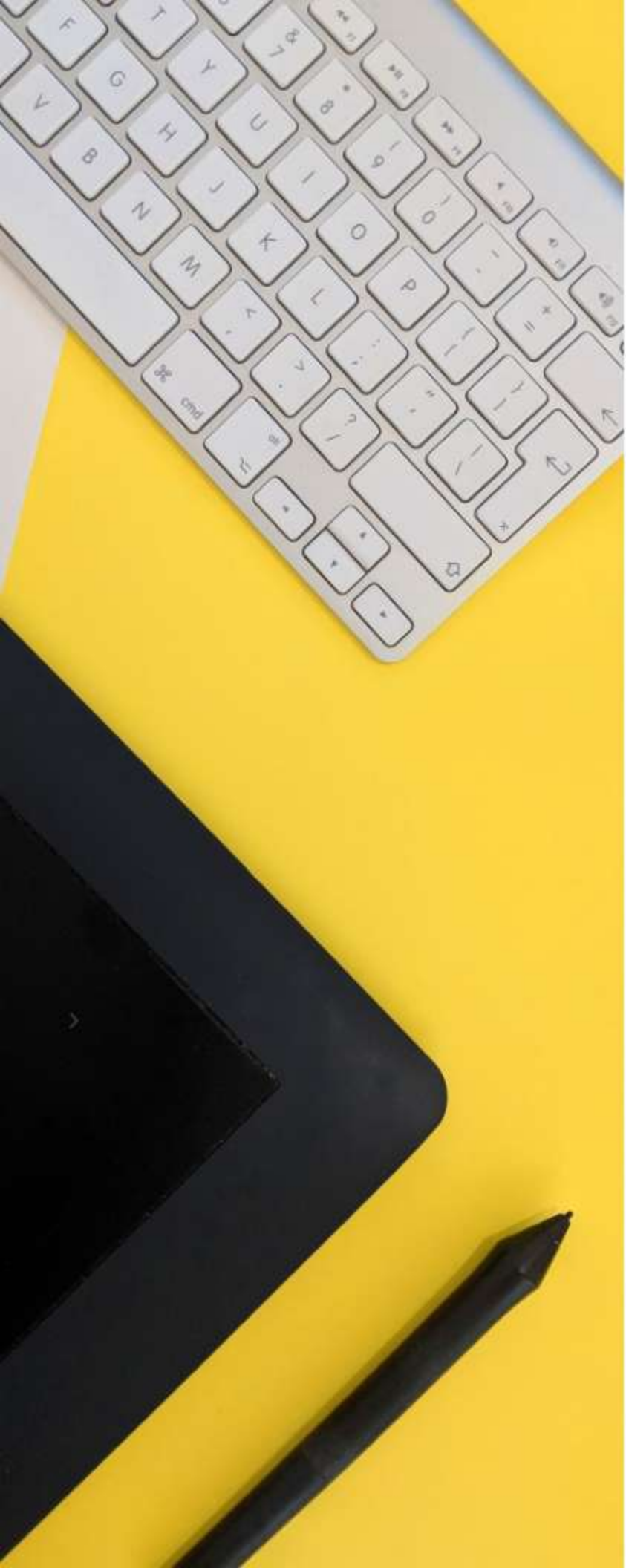
Visão pode ser qualquer saída de representação dos dados, como uma tabela ou um diagrama. É onde os dados solicitados do Modelo (Model) são exibidos. É possível ter várias visões do mesmo dado, como um gráfico de barras para gerenciamento e uma visão tabular para contadores. A Visão também provoca interações com o usuário, que interage com o Controle (Controller). O exemplo básico disso é um botão gerado por uma Visão, no qual um usuário clica e aciona uma ação no Controle.

Não se dedica em saber como o conhecimento foi retirado ou de onde ela foi obtida, apenas mostra a referência. Segundo Gamma et al (2006), "A abordagem MVC separa a View e Model por meio de um protocolo inserção/notificação (subscribe/notify). Uma View deve garantir que sua expressão reflita o estado do Model. Sempre que os dados do Model mudam, o Model altera as Views que dependem dele. Em resposta, cada View tem a oportunidade de modificar-se". Adiciona os elementos de exibição ao usuário : HTML, ASP, XML, Applets. É a camada de interface com o usuário. É utilizada para receber a entrada de dados e apresentar visualmente o resultado.

Controller

Controle é o componente final da tríade, faz a mediação da entrada e saída, comandando a visão e o modelo para serem alterados de forma apropriada conforme o usuário solicitou através do mouse e teclado. O foco do Controle é a ação do usuário, onde são manipulados os dados que o usuário insere ou atualiza, chamando em seguida o Modelo.

O Controle (Controller) envia essas ações para o Modelo (Model) e para a janela de visualização (View) onde serão realizadas as operações necessárias.



Obrigado

Queremos agradecer todo seu empenho e disposição em participar deste projeto que é a mentoria tech uma comunidade construída de alunos para alunos com grandes ambições e objetivos dentre eles temos:

- Fazer com que os alunos alcancem altos níveis em suas formações.
- Mentoria de alunos experientes para apoiar e moldar os alunos mais novos.
- Agregar valor na busca de seu primeiro emprego.
- Produzir e compartilhar conteúdo de qualidade feito por alunos com foco em alunos.
- Disseminar e promover acessibilidade e igualdade em nossos conteúdos ajudando assim toda cota de alunos.
- Realizar encontros com outras comunidades e eventos que gerem impactos positivos como networking, experiências, preposições técnicas entre outros.

Busque nossas redes sociais

+55 (11) 9 61985346

contatomentoriatech@gmail.com

@mentoriatech

www.mentoriatech.com.br

Bootcamp

ACESSE AGORA

Siga nossas mídias sociais
@mentoriatech