**Curso N8N + AI**

**O que é o N8N?**

Primeira impressão:

- É uma plataforma "open source", ou seja, é possível baixar o código deles, verificar e se hospedar onde acharmos melhor

- Sendo uma ferramenta open source podemos usá-la pelo próprio site do N8N (n8n.io) ou "Self Hosted" que seria você baixar o código do N8N (baixar o programa/software) no seu PC ou em qualquer outro PC e usar de qualquer outro lugar em um servidor.

- Nesse curso aprenderemos a usar o N8N self hosted e para isso teremos 3 opções: VPS (Virtual Private Server - Quando você compra um servidor virtual em algum lugar. É basicamente um PC virtual na nuvem de alguém); Railway (é um tipo de VPS, mas é "autoregulado) e Localhost (é o próprio PC, seja o seu ou não)

Funcionalidades:

- O N8N é uma ferramenta de automação que usa "nós" (NODES) e "workflows". Dentro dessa plataforma é possível ter vários workflows, ou seja, fluxos de trabalho automatizados, formados por vários nós que seriam os pequenos pedaços de um fluxo.

- Vale lembrar que os nós existentes nos fluxos agem em sequência. Em outras palavras, os nós ou blocos seguem uma sequência ORDENADA de ações/demandas/ordens

- É uma ferramenta que une a característica do "Code" (literalmente: código, ou seja, programação em diversas linguagens) e do "No Code" (Literalmente: sem código, ou seja, é possível criar automações sem o uso de nenhuma linguagem de programação)

- Com relação ao aspecto No Code - que é o mais presente de certa forma - o N8N funciona com INTEGRAÇÕES, ou seja, a plataforma N8N tem a capacidade de integrar nos fluxos de trabalho outras ferramentas para realizar uma automação. Por exemplo, integrar em um único fluxo ferramentas de análise de dados, serviços de nuvem, ferramentas usadas para a construção de planilhas e dashboards para receber dados de uma nuvem e analisá-los numéricamente. Essas integrações são, justamente, os nós.

- Observação: Caso não exista uma ferramenta na lista de integrações do N8N, é possível usar API's

**Usando o N8N**

Áreas:

- Workflows: É a área inicial do N8N e onde estarão os fluxos de automação criados por nós;

- Templates: É a área onde existem projetos já prontos que podemos usar e modificar de acordo com nossas necessidades, ou seja, não é preciso começar um projeto do zero. Para ser mais preciso, ao clicar em templates seremos redirecionados para uma página dentro do site do N8N na qual existem vários fluxos já feitos;

- Credentials: É onde colocaremos nossas credenciais. O que é uma credencial? Como o N8N trabalha com diversas ferramentas usadas como nós e algumas não estão integradas em seu "banco de dados", é necessário ter acesso à elas por meio de uma API (é a conexão externa com outras plataformas) que por sua vez terá um acesso especial. Esse acesso especial é a credencial. Todas as vezes que for preciso fazer uso de uma ferramenta que precisa desse acesso especial, a credencial que estiver guardada nessa área será usada automaticamente;

- Variables: Não as usaremos, pois elas só estão disponíveis no plano Enterprise, e

- All Executions: Nessa área estarão todos os nossos fluxos já prontos e em execução

Criando workflows:

- Tudo começa no "Canva" que traduzindo seria a tela de pintura à óleo vazia que era usada pelos pintores. É onde criaremos, efetivamente, nossas automações.

- Para começar qualquer automação é preciso adicionar um "trigger" ou gatilho, os quais são:

* On app event: Nesse caso, estamos determinando que para iniciar o fluxo UM EVENTO DENTRO DE UM APP DEVE ACONTECER;
* On a schedule: Determinamos um momento específico para que o fluxo seja disparado;
* On webhook call;
* On form submission: É um formulário gerado pelo próprio N8N que é enviado para um lead (possível comprador/cliente) e quando esse lead confirmar que respondeu o formulário o fluxo dispara;
* Manually: Podemos disparar manualmente;
* When called by another workflow: Dispara quando outro workflow der esse comando, e
* Other ways: Nesse caso existem outro modos para disparar o fluxo.

- Uma vez adicionado o gatilho desejado criamos um "nó gatilho" (TRIGGER NODE); a partir daí estará a nossa disposição outros tipos de nós.

- Ao lado do gatilho veremos o símbolo de adição (+), clicando nele teremos acesso a lista de nós possíveis, qual seja:

* Action in an app: Literalmente, será executada uma ação em um app; e é aqui que poderemos ver todas as integrações que o N8N tem;
* Data transformation: Nessa opção vemos todas as possibilidades de manipulação de dados;
* Flow: Aqui nós iremos determinar os caminhos que serão seguidos pelo fluxo e as condições que guiarão o fluxo;
* Files: Diz respeito a todo tratamento relacionado a arquivos
* Core:
* Advanced AI: Faz a integração de vários tipos de IA's
* Add another trigger:

- Todos os nós funcionam da mesma forma, por isso sabendo 5 ou 6 já possível construir automações

- Vamos analisar mais a fundo como os nós funcionam, mas antes devemos destacar que os dados dentro dos nós são apresentados em formato TABLE; JSON e SCHEMA

- Para configurar ou analisar a configuração de um nó basta clicar no mesmo duas vezes no nó em questão. Dessa forma nós entramos dentro do nó.

- Dentro do nó temos 3 campos: Input; Parameters e Output, vejamos:

* Input: No input estão todos os dados que o nó está recebendo de nós anteriores;
* Parameters: É onde faremos todo o tratamento desses dados de acordo com as funcionalidades do nó que recebe esses dados, e
  + - -Condições: Os dados inclusos nesse campo são lidos em JAVA SCRIPT e é aqui que realizaremos a maior parte do tratamento dos dados;
    - -Propriedades: Nesse campo faremos configurações adicionais para deixar o tratamento mais objetivo.
* Output: Vemos o resultado do tratamento dado às informações recebidas

**\*\*\*\*\*\*\*O nó gatilho tem um layout/apresentação diferente!!!!!!\*\*\*\*\*\*\*\***

**Conceitos Básicos**

Introdução aos Assistentes IA

- Tratando dos assistentes IA, usaremos o assistente da OpenIA. E como isso funciona? Bom, é importante entender que o assistente é uma “entidade” que podemos criar dentro da plataforma da OpenIA e estabelecer parâmetros como:

\* O prompt que o assistente vai usar para funcionar;

\* O objetivo do assistente; qual problema ele vai resolver;

\*Qual o modelo do Chat GPT que o assistente vai usar e se o assistente vai possuir arquivos para a realização de buscas.

- Esse assistente tem o mesmo princípio dos chats que criamos dentro da OpenIA para uso próprio, por exemplo.

- Vale ressaltar que esse assistente será usado no formato de CHAT, ou seja, uma conversa como no WhatsApp, por exemplo. Partindo dessa perspectiva ficará mais fácil entender o conceito de “Thread”.

- Thread é, de forma bastante resumida e direta, uma conversa. É como se o assistente criasse um ambiente no qual essa conversa se desenrola com inputs do usuário (mensagem do usuário) e outputs do assistente (mensagem do assistente), e cada thread é única possuindo cada qual o seu ID. Conforme a conversa vai se desenrolando os dados dentro da thread vão aumentando e, em tese, não há limite para isso.

- A thread possui as seguintes características:

\* Ela própria é a memória do assistente. Conforme a conversa for se desenrolando o assistente vai armazenar os dados para trabalhar melhor;

\* Dentro de cada thread o assistente sempre buscará manter o contexto, e

\* Sempre vai manter o resultado de chamadas de funções solicitadas à ele.

- Uma vez criada a thread, haverá um outro processo que vai acontecer em paralelo a própria thread que é a RUN, ou seja, a ação de executar/ de rodar os inputs do usuário na própria OpenIA afim de que o input seja “tratado” e respondido de acordo. UMA VEZ QUE A OpenIA RESPONDE/EXECUTA/RODA O INPUT DO USUÁRIO, A RUN DEIXA DE EXISTIR E A RESPOSTA É ENVIADA PARA A THREAD PARA QUE O ASSISTENTE A ENTREGUE PARA NÓS.

- Características da Run:

\* Cria mensagens (é, literalmente, a mensagem que o assistente vai escrever como resposta à nossa pergunta;

\* Executa funções;

\* Executa Code Interpreter;

\* Executa Retrieval (é o ato da IA buscar em um arquivo que nós carregamos na conversa as informações necessárias para responder a alguma questão levantada pelo usuário);

\* EXISTE APENAS EM TEMPO DE EXECUÇÃO;

\* Possui “steps” (progresso, completo, chamando função, cancelado e erro) que são, basicamente, os status da run ou do “tratamento das informações”;

\* Também pode ter o modelo definido na hora de execução (diz respeito a versão do Chat GPT que vai tratar/executar/rodar os dados).

\*\*\*\*\*\*\*\*\*\*\*PROMPT = INSTRUÇÕES\*\*\*\*\*\*\*\*\*\*

O que são API’s?

- São as formas que nossos produtos se comunicam com outros serviços.

- Antes de tratarmos mais a fundo como e o que são API’s, precisamos entender que: APESAR DE TUDO ESTAR INTERLIGADO PELA INTERNET, AS COISAS NÃO ESTÃO LIGADAS NATURALMENTE OU NÃO FORAM FEITAS/PROGRAMADAS/PROJETADAS LIGADAS UMAS AS OUTRAS. O que isso quer dizer? Sabemos que tudo que se refere ao digital é fruto de uma programação, é fruto de uma construção a partir de uma linguagem de programação. Essa construção, ou melhor, esse “produto” que é programado o foi dentro de um ambiente digital alheio a internet, por exemplo: para que eu crie um site eu preciso ter um programa (VS Code) que me permita construir o site em todos os seus aspectos. Contudo, uma vez que eu termine o site ele ainda não estará ligado a internet; para isso eu preciso de alguma forma conectá-lo ao world wide web, ou seja, a internet.

- As API’s são algumas das formas de estabelecer comunicações entre sistemas no meio digital, mas o exemplo acima ilustra a necessidade de estabelecer comunicações e que as coisas não estão naturalmente conectadas, mas precisam ser, artificialmente, ligadas umas as outras.

- Certo, mas como as API’s funcionam? Bom, de um lado temos o nosso produto/software/bot e do outro nós temos o serviço ou função que desejamos acessar com nosso produto/software/bot; no entanto, como vimos antes, os sistemas não estão ligados ou não tem acesso uns aos outros naturalmente. É aí que a API vai entrar: ela vai fazer o papel não apenas de ponte, mas de uma prestadora de serviço buscando o serviço ou funcionalidade que desejamos incluir/integrar/implementar ao nosso produto/software/bot.

\*\*\*\*\*\*Uma **interface** é o meio pelo qual o usuário se comunica com um sistema para realizar tarefas. Por exemplo, quando você digita um texto no Microsoft Word, está interagindo com a tela disponibilizada pela aplicação. A interface é a camada de interação entre o usuário e o sistema, onde são apresentados elementos visuais e funcionais que permitem ao usuário executar ações e obter as informações desejadas\*\*\*\*\*\*\*\*

- API = Application Programming Interface (Interface para Programação de Aplicações)

- Características:

\* Geralmente as API’s tem o formato de uma URL (é preciso acessar a plataforma que possui a função ou serviço para gerar a URL da API) – ALGUNS VOCÊ PRECISA PAGAR

\* Possui MÉTODOS, entre eles:

\* GET: Para “Pegar” dados;

\* POST: Para “Criar” dados;

\* PUT: Para “Editar” dados

O que são Webhooks?

\*\*\*\*\*\*É um tipo de API\*\*\*\*\*\*\*

- Com o Webhook não dá para solicitar nenhum serviço ou realização de alguma função

- O Webhook vai receber do serviço notificações que nós iremos especificar para aí tomar alguma decisão que também iremos determinar qual será.

- Quase que paralelamente ao recebimento da notificação, o webhook encaminha para o serviço uma resposta referente ao recebimento, geralmente, dizendo que a notificação foi recebida, mas pode notificar qualquer tipo de situação como um erro por exemplo

\*\*\*\*\*\*\*\*\*NA MAIORIA DAS AUTOMAÇÕES O GATILHO SERÁ UM WEBHOOK\*\*\*\*\*\*

- De maneira geral, gatilhos que disparam com o recebimento de uma mensagem são webhooks. No N8N temos os nós específicos do WhatsApp e do Telegram, por exemplo; mas tem também um nó denominado “Webhook”

- Os nós gatilhos não tem a necessidade de configuração, mas o nó “webhook” precisa de algumas configurações, tais como:

\* Tipo de URL: Esse tipo de nó gatilho permite gerar ou uma URL teste ou Uma URL de “produção”

\* Autorização: Alguns webhooks precisam de “autenticação” que seria uma chave para bater com a notificação que será recebida do serviço desejado

\* HTTP Method: Assim como as API’s, o Webhook necessita da especificação do tipo de método (GET; POST; PUT; DELETE; HEAD; PATCH)

\* Path: É o “local” para o qual o serviço encaminhará a notificação para avisar nosso webhook e iniciar a automação

\* Respond: Literalmente, é a resposta do webhook. São elas: Imediatamente, quando o último nó terminar ou usando o nó “responder para o webhook”.

\* Response code: é um código numérico que significa algo. Existem vários códigos para dizer várias coisas.

- Ainda sobre as respostas do webhook para o serviço, devemos destacar isso como algo muito importante porque o serviço precisa saber o “status da operação”. Como existem vários serviços cada um age de um modo determinado diante da ausência de resposta; alguns ficam mandando a notificação até que o wenhook responda algo, alguns enviam a notificação até que haja uma resposta, mas com intervalos maiores, e outros enviam uma única notificação e o resto é conosco.