**O que é o Blazor?**

* Blazor é um framework da Microsoft para a criação de aplicações web interativas utilizando C# em vez de JavaScript. Assim como Django no Python, o Blazor facilita a construção de aplicações completas, mas o faz usando a plataforma .NET e a linguagem C#.

**Funcionalidades do Blazor**

**Criação de UIs interativas com C#**

* Com Blazor, é possível criar interfaces de usuário avançadas utilizando C# graças ao suporte ao WebAssembly (WASM). O WebAssembly permite que os navegadores executem código binário compilado (como DLLs .NET), possibilitando a execução de código C# no cliente. Além disso, o Blazor permite a interação com JavaScript através de JavaScript Interop, caso seja necessário.

**Compartilhamento de lógica entre servidor e cliente**

* Uma das grandes vantagens do Blazor é a capacidade de compartilhar a lógica de negócios entre o cliente e o servidor, utilizando a mesma base de código na plataforma .NET. Isso simplifica o desenvolvimento e a manutenção de aplicações.

**Renderização de interface como HTML e CSS**

* O Blazor renderiza a interface do usuário utilizando HTML e CSS, garantindo compatibilidade com todos os navegadores modernos, incluindo navegadores móveis. Isso assegura que as aplicações Blazor possam ser executadas em diversas plataformas sem problemas.

**Modalidades de hospedagem**

* **Blazor WebAssembly (WASM):** Executa diretamente no navegador usando WebAssembly. Isso permite que o código C# seja baixado e executado no cliente.
* **Blazor Server:** Executa no servidor e usa SignalR para comunicação em tempo real com o cliente. A interface do usuário é atualizada via conexões SignalR, proporcionando uma experiência de aplicação de página única (SPA).

**Características dos componentes no Blazor**

**Baseado em componentes**

* Os aplicativos Blazor são compostos de componentes. Esses componentes são classes .NET que podem ser encapsuladas em assemblies .NET, facilitando a modularização e reutilização de código.

**Definição de lógica de renderização**

* Cada componente define sua lógica de renderização, permitindo criar interfaces de usuário dinâmicas e flexíveis.

**Manipulação de eventos do usuário**

* Os componentes podem manipular eventos do usuário, como cliques de botões, entradas de texto, etc.de maneira eficiente.

**Aninhamento e reutilização**

* Componentes podem ser aninhados dentro de outros componentes e reutilizados em diferentes partes da aplicação, promovendo a modularidade.

**Compartilhamento e distribuição**

* Componentes podem ser compartilhados e distribuídos como bibliotecas de classes Razor ou pacotes NuGet, facilitando a integração e a reutilização em múltiplos projetos.

**Ferramentas de desenvolvimento**

* **Blazor CLI**: Ferramenta de linha de comando para criar e gerenciar projetos Blazor.
* **Visual Studio e Visual Studio Code:** Suporte completo para desenvolvimento, depuração e publicação de aplicações Blazor.
* **Live Reload:** Funcionalidade que permite ver mudanças em tempo real sem a necessidade de recompilar a aplicação.