Relatório da Aula 4 da HastTag - Chat ao Vivo com Flet: Implicações para um Sistema de Gerenciamento de Vagas de Estágio em Django

Link da Aula: <https://www.youtube.com/watch?v=0eGOjMIpL7o>

Arquivos da aula: <https://drive.google.com/drive/folders/1kO1a2SnvOJjBUGcApPPZQovANJ4myzr0>

**Tema da Aula:** Construção de um Chat ao Vivo utilizando o framework Flet em Python.

**Principais Pontos Abordados:**

1. **Introdução ao Flet:** A aula inicia apresentando o Flet, um framework para desenvolvimento de aplicações web reativas em Python. Destaca-se a facilidade de uso e a sintaxe simplificada para criação de interfaces de usuário.
2. **Estrutura Básica de uma Aplicação Flet:** São apresentados os três pilares fundamentais para construir uma aplicação com Flet:
   * Importação da biblioteca flet: import flet as ft
   * Criação da função principal main() que conterá a lógica da aplicação.
   * Execução da aplicação utilizando ft.app().
3. **Widgets (Elementos Visuais):** A aula demonstra a utilização de alguns widgets básicos do Flet, como:
   * ft.Text(): Para exibição de textos estáticos.
   * ft.TextField(): Para campos de entrada de texto.
   * ft.ElevatedButton(): Para criação de botões com estilo elevado.
   * ft.Column(): Para organizar elementos verticalmente.
   * ft.Row(): Para organizar elementos horizontalmente.
   * ft.AlertDialog(): Para exibição de janelas pop-up modais.
4. **Gerenciando Eventos:** A aula explora como atribuir funcionalidades aos widgets através do tratamento de eventos, como por exemplo, o evento on\_click para o botão, que dispara uma função específica quando o usuário clica no botão.
5. **Atualização da Interface:** É apresentado o método page.update() para atualizar a interface do usuário em tempo real, refletindo as mudanças realizadas pela aplicação.
6. **Construindo o Chat:** A aula demonstra passo-a-passo a construção de um chat simples, incluindo:
   * Criação de um campo de texto para o usuário digitar a mensagem.
   * Criação de um botão para enviar a mensagem.
   * Exibição das mensagens em uma coluna, simulando um chat.
7. **Comunicação em Tempo Real com WebSockets:** A aula introduz o conceito de WebSockets como um meio de estabelecer uma conexão persistente entre cliente e servidor, permitindo a comunicação bidirecional em tempo real. No entanto, a implementação prática utilizando WebSockets não é demonstrada no código fornecido.

**Implicações para o Projeto de Gerenciamento de Vagas de Estágio em Django do Projeto Integrador GEO-EE:**

Apesar de a aula utilizar o Flet para construir um chat e o projeto em questão utilizar Django para um sistema de gerenciamento de vagas, existem diversos pontos relevantes que podem ser aproveitados:

1. **Conceitos Fundamentais de Frameworks Web:** A aula reforça conceitos importantes sobre o funcionamento de frameworks web, como a estrutura básica de uma aplicação, gerenciamento de eventos, atualização dinâmica da interface, que são aplicáveis a qualquer framework, inclusive Django.
2. **Compreensão de Front-End e Back-End:** Apesar de Flet ser um framework full-stack, a aula permite uma clara distinção entre a construção da interface do usuário (front-end) e a lógica por trás dela (back-end), o que é essencial para o desenvolvimento web em geral.
3. **Comunicação Assíncrona:** A introdução do conceito de WebSockets, mesmo que superficialmente, abre caminho para a exploração de funcionalidades em tempo real no projeto Django, como notificações instantâneas de novas vagas ou mensagens entre candidatos e empresas.

**Pontos de Divergência e Limitações:**

É crucial destacar que o código da aula em Flet **não pode ser diretamente aplicado** ao projeto Django devido às diferenças significativas entre os frameworks:

* **Paradigmas Diferentes:** Flet utiliza um modelo reativo e baseado em componentes, enquanto Django segue o padrão Model-View-Template (MVT).
* **Bibliotecas e Ferramentas:** Flet possui suas próprias ferramentas e bibliotecas para interface do usuário, gerenciamento de estado e comunicação, que não são compatíveis com Django.

**Conclusões:**

A aula sobre Flet, apesar de abordar um projeto distinto, fornece uma base em conceitos de desenvolvimento web e abre portas para a pesquisa e implementação de funcionalidades em tempo real no sistema de gerenciamento de vagas em Django. No entanto, a transposição direta do código não é viável e o aprendizado de Django, Django REST Framework, RQL e JWT continua sendo fundamental para o desenvolvimento do projeto.

**Recomendações:**

* **Explorar Django Channels:** Para implementar a funcionalidade de chat em tempo real ou serviços que necessitem qualquer tipo de tráfego de dados em tempo real no projeto Django, recomenda-se a pesquisa e o estudo de Django Channels, uma extensão do Django para gerenciar conexões persistentes com WebSockets.
* **Estudar Serializers em Django REST Framework:** A serialização de dados em Django REST Framework (uso de API) é crucial para a comunicação entre front-end e back-end. É importante entender como serializar os dados das mensagens do chat para que possam ser transmitidas e exibidas corretamente.
* **Implementar Autenticação JWT:** Garantir a segurança do chat com JWT é fundamental para proteger as mensagens e permitir que apenas usuários autenticados participem da conversa.