SESSÕES

As sessões em Flask ficam armazenadas no servidor. No entanto, uma pequena parte da informação, chamada de "session cookie", é armazenada no navegador do cliente. Esse cookie contém apenas um identificador único (por exemplo, um token) que o servidor usa para associar a sessão armazenada no lado do servidor com o cliente específico.

Por padrão, Flask usa cookies assinados (usando a `SECRET_KEY`) para garantir a integridade dos dados da sessão, mas os dados sensíveis não são armazenados no cookie. Em vez disso, eles são mantidos no lado do servidor, seja na memória, em um banco de dados, ou em outro tipo de armazenamento persistente.

Isso garante que, mesmo que o cliente tente modificar o cookie, Flask detectará que ele foi adulterado e não permitirá o acesso não autorizado.

A linha de código `{{ formulario.hidden_tag() }}` em um template Jinja2 no Flask é usada para renderizar automaticamente todos os campos ocultos (`<input type="hidden">`) que fazem parte do formulário.

O que é o `hidden_tag()`?

- O método `hidden_tag()` do objeto `formulario` gera um conjunto de campos ocultos que Flask-WTF (ou WTForms) precisa para funcionar corretamente.
- Estes campos ocultos incluem, por exemplo, o token CSRF (Cross-Site Request Forgery), que é utilizado para proteger o formulário contra ataques CSRF. O token é enviado junto com o formulário, e o servidor verifica se o token recebido é válido.

CSRF (Cross-Site Request Forgery) é um tipo de ataque em aplicações web onde um atacante engana um usuário autenticado para que ele execute ações indesejadas em uma aplicação na qual está autenticado.

Como o CSRF Funciona:

1. **Autenticação**: O usuário faz login em um site legítimo, como um banco online, e o navegador armazena um cookie de sessão que autentica o usuário em futuras requisições.
2. **Engano**: O atacante cria um site malicioso e induz o usuário a visitá-lo, geralmente através de links enviados por e-mail, redes sociais, ou outras formas.
3. **Execução de Ação**: No site malicioso, o atacante insere um código que faz uma requisição ao site legítimo (onde o usuário está autenticado). Como o navegador do

usuário ainda possui o cookie de sessão válido, a requisição é considerada autenticada pelo site legítimo.

4. **Consequência**: O site legítimo processa a requisição, acreditando que foi feita pelo usuário intencionalmente. Isso pode resultar em mudanças de senha, transferências de dinheiro, ou outras ações indesejadas.

Proteção Contra CSRF:

Para proteger contra CSRF, as aplicações web implementam o uso de tokens CSRF. Esses tokens são gerados pelo servidor e inseridos nos formulários ou nas requisições HTTP. Quando o formulário é enviado, o token é enviado junto e o servidor verifica se ele é válido.

Em Flask, a proteção contra CSRF é geralmente gerenciada por Flask-WTF, que adiciona automaticamente o token CSRF nos formulários através do `hidden_tag()` mencionado anteriormente.