**Como é o MVC no Django**

O padrão MVC organiza o código para separar a apresentação dos dados (View) da lógica de interações e manipulação de dados (Model e Controller). Isso melhora a estrutura e facilita o desenvolvimento e a manutenção do sistema.

Model: Ele lida com a representação dos seus dados, serve como uma interface para os dados armazenados no próprio banco de dados e também permite que você interaja com seus dados sem precisar de todas as complexidades do banco de dados subjacente.

View: Como o próprio nome sugere, é a parte do sistema responsável por exibir o que o usuário vê no navegador de um aplicativo web ou na interface gráfica de um software de desktop.

Controller: Ele coordena o fluxo entre o modelo e a interface, buscando dados do banco e enviando para a visualização. Também captura informações do usuário e realiza atualizações conforme necessário. Em resumo, funciona como o "centro de comando" do sistema.

Mesmo com o Django seguindo o padrão MVC, ele prefere usar sua própria lógica na implementação. O framework lida com esses processos com sua própria lógica, usando o Modelo-Template-View. É por isso que o Django é mais conhecido como o framework MTV.

**MTV no Django e relação com cada item do MVC**

Model: Assim como a explicação do Modelo no padrão MVC, isso também assume a mesma posição que a interface ou relacionamento entre os dados e contém tudo relacionado ao acesso e validação de dados.

Template: Isso se relaciona com exibição no padrão MVC, pois é a camada de apresentação que manipula a lógica de apresentação no framework e basicamente controla o que deve ser exibido e domo deve ser exibido ao usuário.

View: Esta parte se relaciona com o Controlador no padrão MVC e lida com toda a lógica de negócios que retorna aos respectivos modelos. Ela serve como ponte entre o model e o template. No Django, a View inclui tanto a lógica de negócios quanto a de apresentação, ao contrário do padrão MVC, onde a View lida apenas com a apresentação. O Template no Django cuida exclusivamente da parte visual, enquanto no MVC tradicional, não há um componente separado para isso.