* Django’s MVT (Model-View-Template) architecture and how it handles request-response cycles.

**MVT Components**

1. **Model**:
   * **Role**: Represents the data structure. It’s responsible for database interaction, data validation, and defining the data schema.
   * **Examples**: Classes in models.py that define fields and behaviors of the data stored.

* python

from django.db import models

class Car(models.Model):

make = models.CharField(max\_length=50)

model = models.CharField(max\_length=50)

year = models.IntegerField()

1. **View**:
   * **Role**: Handles the business logic and interacts with the model to fetch the data. It processes the user request and returns a response.
   * **Examples**: Functions or classes in views.py that query models, process data, and render templates.

* python

from django.shortcuts import render

from .models import Car

def car\_list(request):

cars = Car.objects.all()

return render(request, 'car\_list.html', {'cars': cars})

1. **Template**:
   * **Role**: Determines how the data is presented to the user. It’s responsible for rendering the HTML and any other dynamic content.
   * **Examples**: HTML files in the templates directory that use Django's templating language to display data.

* html

<!-- car\_list.html -->

<!DOCTYPE html>

<html>

<head>

<title>Car List</title>

</head>

<body>

<h1>Available Cars</h1>

<ul>

{% for car in cars %}

<li>{{ car.make }} {{ car.model }} ({{ car.year }})</li>

{% endfor %}

</ul>

</body>

</html>

**Request-Response Cycle**

Here's how Django handles the request-response cycle using the MVT architecture:

1. **Client Request**: The client (typically a web browser) sends a request to the Django server.
2. **URL Routing**: The request is received and matched against URL patterns defined in urls.py.

* python

from django.urls import path

from . import views

urlpatterns = [

path('cars/', views.car\_list, name='car\_list'),

]

1. **View Processing**: Once the URL is matched, Django calls the appropriate view function. The view function interacts with the model to fetch data.

* python

def car\_list(request):

cars = Car.objects.all()

return render(request, 'car\_list.html', {'cars': cars})

1. **Template Rendering**: The view then passes the data to a template. The template renders the data into an HTML page.

* html

<!DOCTYPE html>

<html>

<head>

<title>Car List</title>

</head>

<body>

<h1>Available Cars</h1>

<ul>

{% for car in cars %}

<li>{{ car.make }} {{ car.model }} ({{ car.year }})</li>

{% endfor %}

</ul>

</body>

</html>

1. **Response Sent**: The rendered HTML page is sent back to the client's browser as the HTTP response.