# Introduction of DJANGO:-

Django is a free & open-source web application framework written in Python. It is a MVT (Model View Template) based framework. It is maintained by the Django Software Foundation (DSF). By using Django, we can build web applications in very less time.

# History of DJANGO:-

Django was design and developed by Lawrence journal world in 2003 and publicly released under BSD license in July 2005. Currently, DSF (Django Software Foundation) maintains its development and release cycle. The framework was named after guitarist Django Reinhardt.

Django initially released on 21, July 2005 and current stable version is 5.2.5 which was released on 6th August, 2025.

# Versions of DJANGO:-

| Version | Date | Notes |
|---|---|---|
| 0.90 | 16 Nov 2005 | |
| 0.91 | 11 Jan 2006 | "new-admin" |
| 0.95 | 29 Jul 2006 | "magic removal" |
| 0.96 | 23 Mar 2007 | "newforms", testing tools |
| 1.0 | 3 Sep 2008 | API stability, decoupled admin, unicode |
| 1.1 | 29 Jul 2009 | Aggregates, transaction based tests |
| 1.2 | 17 May 2010 | Multiple db connections, CSRF, model validation |

| | | |
|---|---|---|
| 1.3 | 23 Mar 2011 | Class based views, staticfiles |
| 1.4 LTS | 23 Mar 2012 | Timezones, in browser testing, app templates. |
| 1.5 | 26 Feb 2013 | Python 3 Support, configurable user model |
| 1.6 | 6 Nov 2013 | Dedicated to Malcolm Tredinnick, db transaction management, connection pooling. |
| 1.7 | 2 Sep 2014 | Migrations, application loading and configuration. |
| 1.8 LTS | 1 Apr 2015 | Native support for multiple template engines. *Supported until at least April 2018* |
| 1.9 | 1 Dec 2015 | Automatic password validation. New styling for admin interface. |
| 1.10 | 1 Aug 2016 | Full text search for PostgreSQL. New-style middleware. |
| 1.11 LTS | 4 Apr 2017 | Last version to support Python 2.7. *Supported until at least April 2020* |
| 2.0 | 2 Dec 2017 | First Python 3-only release, Simplified URL routing syntax, Mobile friendly admin. |
| **2.1** | 1 Aug 2018 | Model "view" permission. |
| **2.2 LTS** | Apr 2019 | Supported until at least April 2022 |
| **3.0** | 2 Dec 2019 | ASGI Support |

| | | |
|---|---|---|
| **3.1** | 4 Aug 2020 | Asynchronous views and middleware |
| **3.2 LTS** | 6 Apr 2021 | *Extended Support until at least April 2024* |
| **4.0** | 7 Dec 2021 | *Extended Support until at least April 2023* |
| **4.1** | Aug 2022 | *Extended Support until at least December 2023* |
| **4.2 LTS** | Apr 2023 | *Extended Support until at least April 2026* |
| **5.0** | Dec 2023 | *Extended Support until at least April 2025* |
| **5.1** | 7 Aug 2024 | *Extended Support until at least December 2025* |
| **5.2 LTS** | 2 Apr 2025 | *Extended Support until at least April 2028* |

# Features of Django:-

- Rapid Development
- Secure
- Scalable
- Fully loaded
- Versatile
- Open Source
- Vast and Supported Community

# Django MVT Architecture :-

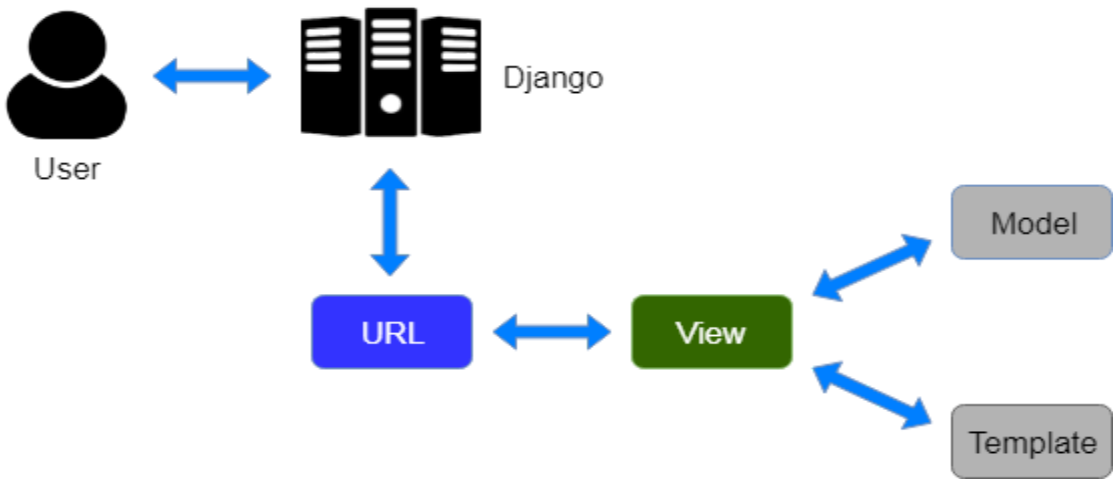
MVT (Model View Template) is a software design pattern which is a collection of three parts Model View and Template. The Model helps to handle database. It is a data access layer that handles the data.

The Template is a presentation layer that handles all the User Interface part. The View is used to execute the business logic and interact with model to carry data and renders template.

Although Django follows MVC pattern but maintains its own conventions so controlling is handle by the framework itself.

There is no separate controller and complete application is based on Model View and Template. That's why it is called MVT application.

See the following graph that shows the MVT based control flow.




Here, a user **requests** for a resource to the Django, Django works as a controller and check to the available resource in url.

If url maps, **view is called** that interact with model and template, it renders a template.

Django respond back to the user and send a template as **response**.

# Django Project Folder Directory Structure:-

A Django package contains the following packages and files. The outer directory is just a container for the application. We can rename it further.

- **manage.py:** It is a command-line utility which allows us to interact with project in various ways and also used to manage application that we will see later on in this tutorial.
- A directory (djangodemo) located inside, is the actual application package name. Its name is the Python package name which we'll need to use to import anything inside the application.
- **__init__.py:** It is an empty file that tells to the Python that this directory should be considered as a Python package.

- **settings.py:** This file is used to provide application settings such as database connection, static files linking etc.
- **urls.py:** This file contains the listed URLs of the application. In this file, we can mention the urls and corresponding actions to perform the task and display the view.
- **wsgi.py:** It is an entry-point for WSGI-compatible web servers to serve Django project.[Web Server Gateway Interface]