Django Web Framework

Django project vs Application-

What is Django Project?

A Django project is the top-level directory containing all the configurations, files, and apps needed for your Django web application to function. It's like the blueprint for your entire website.

Here you define crucial settings that govern the overall application's behavior, such as database connections, secret keys, security settings, and installed apps.

The project handles how URLs map to specific functions within your application. This is where you define how different parts of your website are accessed through URLs.

A project can hold one or more Django applications, each with a distinct functionality.

Example- Bank Project, E-commerce website, etc.

What is Django Application?

A Dango Application is responsible to perform a particular task in our entire web application.

A Django app represents a focused and self-contained module responsible for a particular aspect of your website's functionality. For instance, you might have separate apps for user authentication, blog posts, or online shopping carts.

Each Django app typically has its own models.py file defining data structures (models) for your app's data, views.py containing functions handling user requests (views), and templates (HTML files with Django template tags) for presenting data to the user.

The beauty of Django apps is their reusability. You can develop a well-designed app and use it in multiple projects without significant modifications.

Example- Withdraw app, loan app, shopping cart, registration, user profile, feedback etc.

How to create Django Project?

In the previous video, we downloaded and installed Django by following the steps.

Now, we are going to use terminal and VS code IDE to create our first Django project.

Steps to Create a Django Project-

1.Open VS Code

2.Open terminal or command prompt

3.In terminal at the folder where you want to save all your Django projects, navigate to that folder.

4.Use command-

Django-admin startproject name_of_the_project

First make folder where all your Django projects will be stored.
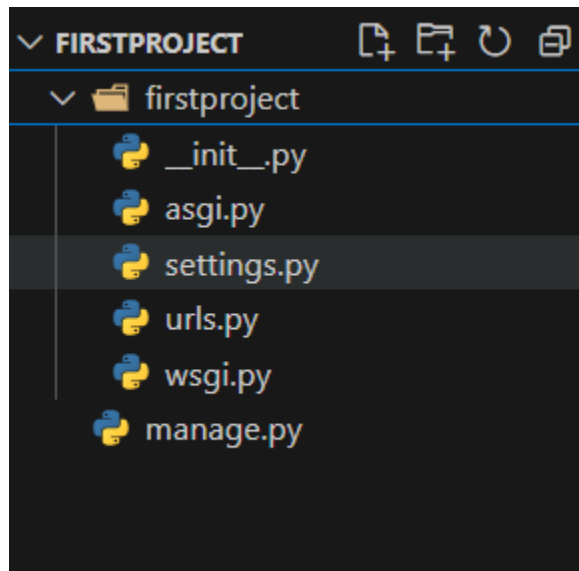
Use **cd command** to go inside a folder and **md command** to create a new folder/directory.

```
C:\Users\LENOVO>cd Documents

C:\Users\LENOVO\Documents>md djangocourse

C:\Users\LENOVO\Documents>cd djangocourse

C:\Users\LENOVO\Documents\djangocourse>
```

Now create your first project-

Django-admin startproject firstproject

The following project structure will be created-

Now we are going to discuss what are all these python files that are automatically created when we create a project.

__init__.py-

It is a blank python script. Because of this special file name, Django treat this folder as python package.

It is used to initialize the package when it is imported.

The __init__.py file can contain code that will be executed when the package is imported, as well as function definitions and variable assignments. It is a good place to put any code that you want to run when the package is first imported.

asgi.py (optional)-

This file configures your project to run using the Asynchronous Server Gateway Interface (ASGI). It's typically used for deploying Django applications in modern web servers like Asgi servers

wsgi.py-

This file configures your project to run using the Web Server Gateway Interface (WSGI). This is the traditional way to run Django applications on web servers.

We can use this file while deploying our application in production on online server.

manage.py-

This is a crucial script that serves as the command center for your project. You'll use manage.py for various tasks like running the development server, creating database tables (migrations), creating apps, and running administrative commands.

settings.py-

This is the heart of your project's configuration. It contains all the critical settings that govern the behavior of your Django application, including:

- Database connection details

- Secret keys for security purposes

- Installed applications (including your own apps)

- Middleware components that handle requests and responses

- Static and media file serving configurations

- Template loader settings

- Authentication and authorization settings

- And many more

urls.py-

This file defines the URL patterns for your entire project. It maps incoming URLs to specific views (functions) within your application, essentially acting as a roadmap for how users access different parts of your website.

How to Run Django Development Server?

Go to the root directory where manage.py is located. Now in terminal/cmd write command-

python manage.py runserver

After executing the command in cmd you will see something like this-

```
C:\Users\LENOVO\Documents\djangocourse\firstproject>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
 auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
May 12, 2024 - 16:26:01
Django version 5.0, using settings 'firstproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```
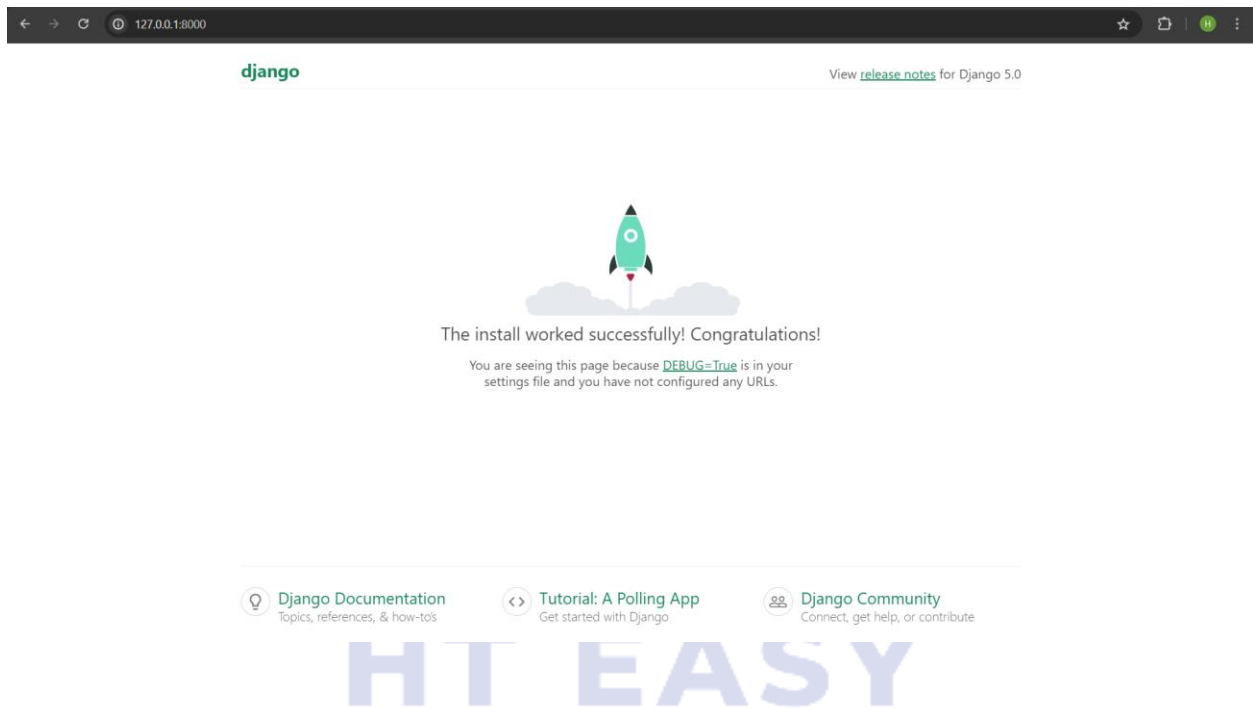
Copy the localhost url- http://127.0.0.1:8000/

Paste the URL on the web browser (Chrome, Microsoft Edge or Firefox)-

If the installation worked then you will see this page-

Django framework is responsible to provide development server. Even Django framework provides one inbuilt database sqlite3.

Note: Once we started Server a special database related file will be generated in our project folder structure.

db.sqlite3

Web Server-

- **Purpose:** A web server is a software program that runs on a computer and is responsible for:

    - **Receiving HTTP requests:** When you enter a website address (URL) in your web browser, it sends an HTTP request to the web server associated with that website.

    - **Processing requests:** The web server interprets the request, which typically specifies a web page or resource (like an image or file) on the website.

- o **Delivering responses:** The web server retrieves the requested content from the appropriate location (e.g., file system, database) and sends an HTTP response back to your web browser.

- o **Serving static content:** Web servers primarily serve static content like HTML pages, CSS stylesheets, JavaScript files, and images that make up the website.

Django Development Server

- **Purpose:** When you're developing a Django application, you typically use the built-in development server to run your project locally on your development machine. This server is designed for ease of use during development, offering features like:

  - o **Hot reloading:** As you modify your code and save changes, the development server automatically reloads the application, allowing you to see the effects of your changes instantly without manually restarting the server. This is a huge time-saver during development.

  - o **Simplicity:** Setting up the development server is straightforward. You typically just run python manage.py runserver from your project directory.

  - o **Debugging capabilities:** The development server provides some basic debugging features, helping you identify errors in your code.

Django development server is not suitable for production environments (public websites) due to:

- **Performance:** It's not optimized for handling high traffic volumes or complex applications.
- **Security:** It might not have the same level of security features as dedicated web servers.

In the next video we are going to create our first Django Application.

Join our growing community of tech enthusiasts! Subscribe to our YouTube channel, where we simplify programming languages in the easiest way possible. Gain a deeper understanding with our clear explanations and receive exclusive notes to enhance your learning journey. Don't miss out on valuable insights – hit that subscribe button now and embark on a programming adventure with us!

Subscribe to our channel: https://www.youtube.com/@HTEASYLEARNING