Django is a free and open source web application framework.

A framework is a collection of components which helps us build web sites faster.

It is a back end framework.

Django has automatic admin interface which supports CRUD.

pip install django==<version> to install django in the machine.

django-admin startproject <project\_name> --> inorder to create a project in django.

django-admin startapp <app\_name> --> creates an app in django.

python manage.py startapp <app\_name> to create the app in django.

python manage.py runserver --> starts the server for django project.

python manage.py makemigrations app\_name --> migrates the given the app

python manage.py migrate --> applies migrate and all the datbases are createed.

python manage.py sqlmigrate food 0001 --> creates the models for the app food.

python manage.py showmigrations food --> shows the migrations for the app food.

python manage.py shell --> opens the interactive shell for the python.

python manage.py createsuperUser -->creates te user for admin panel.

chunks or microservices kind of structures in django are known as apps. a project may contain multiple apps.

view in django is a process user requests...

view is written as a python function.

view.py --> is used for for return something like serviceImplementation class in spring boot.

urls.py --> binds the enpoint which is like a controller in spring boot.

Large websites needs large amount of data to be stored.

Data for these websites is stored in the database. Models allows us to create the tables/database schema.

Models are the bkue print which can be used to create a database.

Models are classes in python. (models.py)

Inorder to get the databases created automatically then you have to add the app in the installed app in the settings.py only then when you run the

migrate command the databases get created successfully.

Data Storage in Django.

Database abstraction API in django helps in DDL and DML operations on Database.

---------------------------DATABASE OPERATIONS------------------------------------

<ModelName>.objects.all() --> lists all the data of that Table or objects of that Table.

a.id / a.pk --> gives id / primary key for the object.

We have to register the models in the admin.py file inorder to get it to displayed in the django admin site.

We retreive the object from the Database in django using QuerySet.

QuerySet = Model + Manager + Method == Model.Manager.method()

Item + objects + all() == Item.objects.all()

Now returning the QuerySet using django formats the data in some weird format now Templates come into play inorder to return the

django allows us to return the HTML to the HTTP Response.

Now we will create templates and the HTML code will be in that file and we can return that HTML file. and we can also pass the database\_entries

in the render and send it to the data.

Django comes with Django Template Engine default template engine.

Every template should have some context in order to render the template.

Templates are used to combine both dynamic data and static html we need these template inorder to render it.

Django Template Language is the syntax we use in templates or html pages in the django. (Jinga 2, DTL)

{{}} -> define a variable

{%%} -> control flow tags or flow statements

{% python syntax %}

{% endname %}

For using the some of the html files in other files we have to use block {% block block\_name %} {% endblock %} and we have to

use that in the template we want by using {% extends template\_name %}

Namespacing in django is used to differentiate the apps and urls.

In static folder we place all the CSS for the templates required.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* FORMS IN DJANGO \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

To use forms in django we have to create a seperate forms.py so that we create a meta classs.

Adding authentication to django we have to add the seperate app.

Inorder to add some fields to existing django forms we have to extend the form.

Login and Logout views not have views we have to add views for those only backend mechanism is handled by django.

The django gives user login status by inbuilt status.

Restricting the access to some views based on authentication then we have to add a decorator saying login\_required it will restrict the

login to only to logged in users.

--------------------------------------------DJANGO SIGNALS------------------------------------------------------------------

Lets suppose we want to create a profile page whenever the user gets registered into aplication now we want to automate this task .

Now django signals comes into picture which consists of signal dispatcher whose main task is to notify the applications whenever some changes occurs in the particular application.

In Django instead of using the views we can also write class based views in Django we can remove the functions in the views.py and write the classes instead of functions there.