\* command for installing django : pip install Django

\*checking version : django-admin version

FRAMEWORK : collection of libraries

\*create project : django-admin startproject name

\*runserver : python manage.py runserver

\* strart application : python manage.py startapp appname

-------------------------------------

Architecture of django : MVT

MVT - Model Viev Templates

BACKEND:

----------

url - mapping to functions

view - function

model- database table

template - html layout

--------

databse:

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'DB\_name',

'HOST': '127.0.0.1',

'PORT': '3306',

'USER': 'DB\_user',

'PASSWORD': 'DB\_password',

}

}

-------------

workflow

--------

project url - application url - view - model - layout

=-------------

project(settings) - application url

============\

new project

application calculate

add

sub

---------

POST:

add csrf token when using post;

{%csrf\_token%}

------------

url href={%url 'name'%}

-----------------

import redirect for returning name in url

----------------------

KEY related error:

different key name : M

------------------------

After creating/updating models : python manage.py makemigrations - creating querry for executing migration file

execute file : python manage.py migrate

--------------

two string method:

def \_\_str\_\_(self):

return self.book\_name

print object

---------------------------

django shell : python manage.py shell

------------------

ORM Querries

(Objct Relational Mapper)

-----------

import model: from AddBook.models import AddBooks

create

-------

books=AddBooks(book\_name="Alchemist")

books.save()

print all : ref = modelname.objects.all()

For fetching a specific object: book=AddBooks.objects.get(book\_name="name")

book.author

book.price

fetch by id : book=AddBooks.objects.get(id=1)

UPDAT orm querry:

-----------

fetch : book=AddBooks.objects.get(book\_name="Das Kapital")

book.price=250

book.save()

delete:

-------

fetch : book=AddBooks.objects.get(id=1)

book.delete()

-----------

filter:

-------

books=AddBooks.objects.filter(category="Novel")

price>=300:books=AddBooks.objects.filter(price\_\_gte=100)

--lte(lessthan or equalto)

exclude: books=AddBooks.objects.exclude(category="Novel")

----

Display in ascending order: books=AddBooks.objects.all().order\_by('price')

-----

descendign order: books=AddBooks.objects.all().order\_by('-price')

---------

sort by price and find highest price : books=AddBooks.objects.all().order\_by('-price')[0]

--------

Max

-----

from django.db.models import Max

book=AddBooks.objects.aggregate(Max('price'))

Min

----

from django.db.models import Min

book=AddBooks.objects.aggregate(Min('price'))

---------

Case insensitive search: book=AddBooks.objects.filter(book\_name\_\_iexact="alchemist")

----------

contains:

book=AddBooks.objects.filter(book\_name\_\_contains="alc")

--------

HW:

project : CRM

app: employees

models : emp\_name,desig,salary,experience,email=unique

6 entry