ADMIN:

python manage.py cratesuperuser

admin.site.register(Skill)

MODEL AND DB:

from django.db import models

- make a clss by intrehence from model.Model(use Model blocks in vscode) in model.py in our app and put fields we want and its type:

    intigergield, charfield, boolianfield, textfield

    and some parameters:

    intigerfield:validators=[MinValueValidator(18), MaxValueValidator(50)]

        ->from django.core.validators import MinValueValidator, MaxValueValidator

    .CharField(max_length=50, db_index=True)

    EmailField(max_length=254,null=True, blank=True, default="blabla")

        خالی بودن رشته خالی بودن فیلد در دیتابیس

- python manage.py makemigration: initalize migrates ->changing

- python mange..py migrate: execute and set my table in db.sqlite3-> changing

->class Meta:

        verbose_name = "Skill"

->+override __str__

-now I can make an object from my model : person1= Person(name="ali", age=10) |

for save it to db: person1.save() |or Person.objects.create(name="ali")

some important methodes:

    Person.objects.all()

    Person.objects.get(name="ali")

        **use it in try becaus if there are no one(except) return 404

    Person.objects.filter(name="ali) Person.objects.filer(age__gt=10)

                                                    .filter(age_lt=10, weight_gt=60)

    -> or filter:

    ->more abilitis:

        from django.db.models import Sum, Avg, Min, Max

        Developer.objects.aggregate(.....)

-most imprtant one: relations:

one to one: define a field in one of them(rather in one that has another):

    x= models.OneToOneField(Projects,on_delete=models.CASCADE) other model   all on_delet:

to access to related model fileds just use it's name: Skill.x.all() and x.skill.all()

**related_name:

one to many: define a field in کوچولوئه، منظورم از کوچولوئه اینه که تنها یک نفر داره اما اون یک نفر داره یه سرپرست داره اما اون سرپرست میتونه چند تا داشته باشه

developer= models.ForeignKey(Developer, on_delete=models.CASCADE)

to access: Skill.developer.name and Developer.skill-set.all

to add: skill1=Skill(name="english")

skill1.developer=developr1   automatic it adds to developer1 to

many to many: define a field in one of them(rather in one that has another):

projects=models.ManyToManyField(Projects)

to access: developer1.projects.all() and projects1.developer_set.all()

to add: developer1.projects.add(folan)   and projects1.developer_set.add(bahman)

*dont forget first save all in db then relate togather in this one.

FORM:

- we have 3 ways to create form:
- 1- manual form:
  - based on front, in html in form with POST methode we get atribute we want.
  - but not recommended.
- 2- django form:
  - first we create form.py in our app. then like the picture we make a class and inthrence from "forms.Form"
  - then we put our atrbiute we want.
  - in vews.py we import form.py. like picture, we have to modes:
    - get: just make an object of this form and send it to template.
    - post: make an object of form with "requesr.POST" constructor.
      - then if it was valid: we extract data and access it like dictionary
      - and we can use it like put in database.
    - at last I can send it to template like any variable I had.
  - in template:
    - in form tag, with POST methode(and don't forget csrf_token token)
    - then I put the_form like any variable I had.
    - at last put a button with "submit" type.

it's too good becaus error handeling and order.

3-form model: (put form data directly in model)

in next section.

+manage.py shell

+plug methodes: