Django

Django הוא framework שנוצר על מנת לתת מענה לצורך ליצור שרתי backend המנהלים את המידע של אתרי אינטרנט ושומרים אותו במסדי נתונים.
Django מאפשר קבלה ושליחה של מידע מאתר האינטרנט בצורה נוחה וידידותית. מעבר לכך, הופך את מלאכת ניהול מסד הנתונים לפשוטה ומוריד מהמתכנת את הצורך ליצור שאילתות SQL מסובכות למסד הנתונים.

מושגי יסוד בעבודה עם Django

Web Applications

כל שירות שהשרת נותן נקרא web application. לדוגמה, שירות קבלה / שליחה של מטרה במאגר המטרות.
את השמות של כל האפליקציות הפעילות בשרת שלנו Django מגדיר בקובץ settings.py ברשימה של INSTALLED_APPS:

```
INSTALLED_APPS= [
        # Default Django applications
        `targets`
]
```

ליצירת אפליקציה חדשה השתמשו בפקודה:

```
Python manage.py startapp [APPLICATION NAME]
```

URL

כל סוג בקשה לשרת מוגדר בעזרת URL שונה.
את כל הURLים הרלוונטים כתוב בקובץ url.py של כל אפליקציה, בצורה הבאה:

```
path(URL, handle_function)
```

בקובץ url.py של השרת כולו נשים include לURLים של האפליקציות באתר. לדוגמה:
קובץ url.py של אפליקציה:

```
urlpattern = [
        path("AddTarget/", views.add_target),
]
```

קובץ url.py של כל השרת:

```
urlpattern = [
        path("api/target/", include("target.urls")),
]
```

Models

מגדיר את הטבלאות במסד הנתונים.
נשמר בקובץ models.py.
כל model מוגדר בתור class ויורש מ models.Model
כל property של המחלקה הוא שדה בטבלה. יש סוגים שונים של שדות, למשל: IntegerField, charField וכו'

```
Class Target(models.Model):
        Name = models.CharField(max_length=100),
        ...
```

Serializers

כלי שמטרתו לוודא שהמידע נכנס ויוצא בפורמט הנכון.
נשמר בקובץ sterilizers.py
לכל model יש serializer
כל seriliazer מוגדר בתור class ויורש מserializers.ModelSerializer
הproperty fields הוא tuple שמגדיר את השדות של הטבלה.

```
Class TargetSerializer(serializers.ModelSerializer):
        Class Meta:
                Model = Target
                Fields = ('name', ...)
```

Views

הקובץ שמכיל את הפונקציות שמטפלות בבקשות.
נשמר בקובץ views.py
הקובץ url.py משתמש בviews על מנת לטפל בבקשות שנשלחות אליו.
כל פונקציה בviews מקבלת request כפרמטר.
בנוסף, כל פונקציה משתמשת בcsfr_exempt decorator@.

```
@csfr_exempt
def add_target(request):
        ....
```

עבודה עם Django

יצירת הטבלאות

```
python manage.py makemigrations
python manage.py migrate
```

הפעלת השרת

```
python manage.py runserver [PORT]
```

שמירה בDB

השמירה בDB כוללת 4 שלבים:

1. פרסור המידע שנשלח
2. מעבר שלו בserializer

```
target_serializer = TargetSerializer(data=target_data)
```

3. וידוא תקינות

```
if(target_serializer.is_valid()):
```

4. שמירה

```
target_serializer.save()
```

משיכה של המידע הרלוונטי ממסד הנתונים

```
Targets = Target.objects.filter(DB_FIELD=IDENTIFIER)
```

משיכה של כל המידע

```
Targets = Target.objects.all()
```

חשוב!
אחרי משיכה מהDB המידע שהתקבל צריך לעבור בserializer. לאחר מכן המידע יתאכסן בשדה data של הserializer.