University of Windsor

School of Computer Science
https://cs.uwindsor.ca

Master of Applied Computing
COMP-8347 - Winter 2023
Internet Applications and Distributed Systems

Dr. Usama Mir – Usama.Mir@uwindsor.ca
Dr. Saja Al-Mamoori – sajak@uwindsor.ca

LAB 4 – Django Models and Queries

Marks: 2

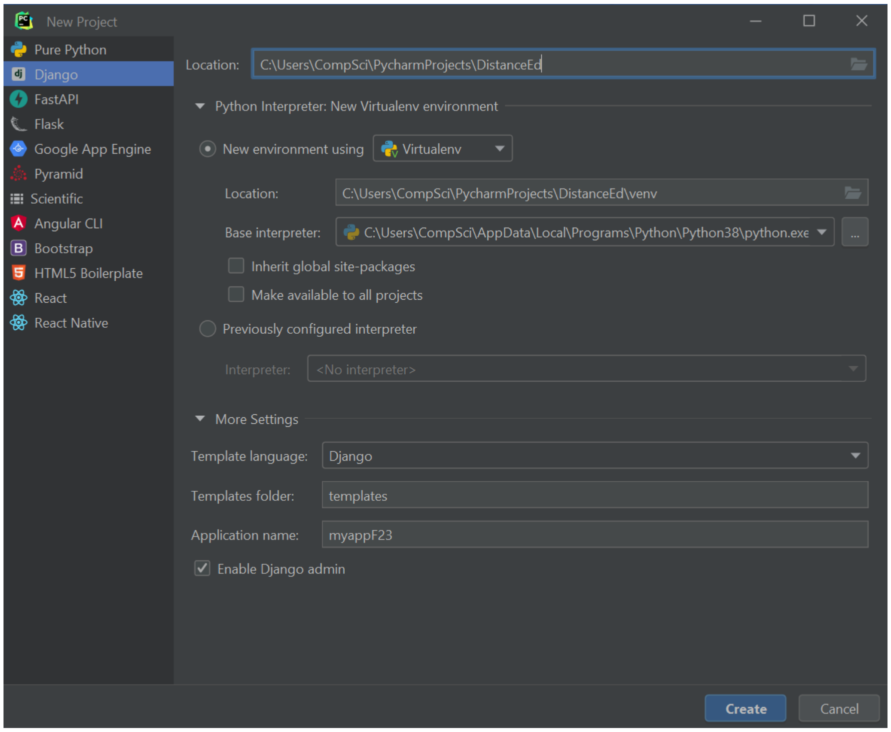
Submission: For part 2 submit models.py from myappF23. For part 3, submit an MS Word file with the queries' solutions. Only one group member should submit the files.

Part 1: Install necessary software, Python, Django, and PyCharm

Follow the setup instructions given in the installation guide for Python, Django, and PyCharm for Windows/MAC OS X. The guides are available on Brightspace under Resources.

Run Django from PyCharm: Start a new project in PyCharm as shown in the dialogue box below.

1. Open PyCharm and click on **File → New Project.** In left pane select "**Django**". In dialog box enter path and project name (**Distance-Ed**). Enter the **path to Python** as the Base interpreter. Click "**More Settings**" and enter Application name (***myappF23***). Click ***Create***.



Location: C:\Users\CompSci\PycharmProjects\DistanceEd

Python Interpreter: New Virtualenv environment

● New environment using Virtualenv

Location: C:\Users\CompSci\PycharmProjects\DistanceEd\venv

Base interpreter: C:\Users\CompSci\AppData\Local\Programs\Python\Python38\python.exe

□ Inherit global site-packages

□ Make available to all projects

○ Previously configured interpreter

Interpreter: <No interpreter>

More Settings

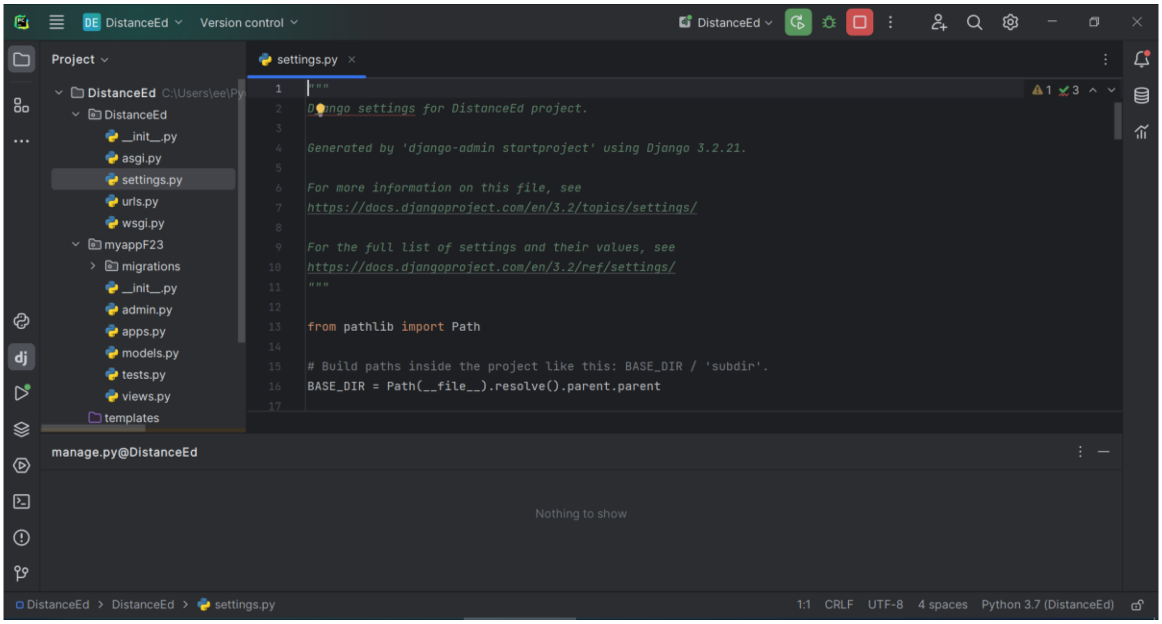
Template language: Django

Templates folder: templates

Application name: myappF23

☑ Enable Django admin

2. Once the project "Distance-Ed" is created, click on **Tools → Run manage.py**. After this, you will see a console as follows:



```
"""
Django settings for DistanceEd project.

Generated by 'django-admin startproject' using Django 3.2.21.

For more information on this file, see
https://docs.djangoproject.com/en/3.2/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.2/ref/settings/
"""

from pathlib import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent
```

manage.py@DistanceEd

Nothing to show

In the console shown at the bottom of the above figure, do the following:

a) Create the project database

- Type the command **`migrate`**
- Verify that the database is created by checking the **migration** folder.

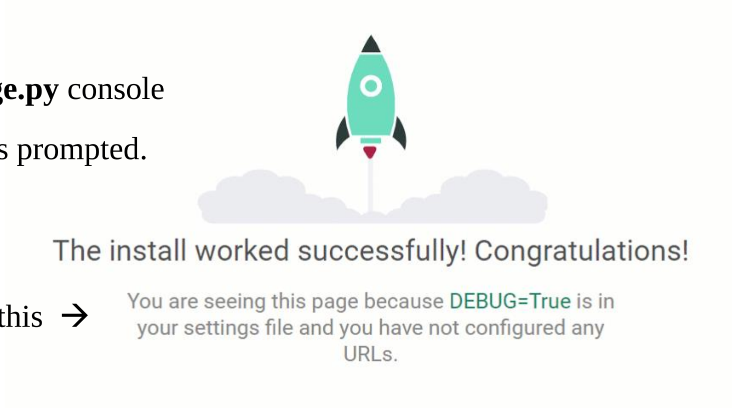
b) Create admin user

- Type **`createsuperuser`** in **manage.py** console
- Enter username, email, and password as prompted.

c) Start the server

- Click Run → **Run 'DistanceEd'**.

d) Go to 127.0.0.1:8000/ where you should see this →



The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

PART 2: Work with Django models

In this lab, you will start developing a Distance-Education **web app** in Django. First, you’ll build the following models/tables in the web app database. Then, you’ll connect these models through relationships, e.g., one-to-many, many-to-many, etc.

Edit *models.py* to create the necessary models for Distance-Education website. We will build this website step by step throughout the labs. Write meaningful comments using the # sign wherever necessary.

1. Add the following models.

```
from django.db import models

class Student(models.Model):
STUDENT_STATUS_CHOICES = [
        ('ER', 'Enrolled'),
        ('SP', 'Suspended'),
        ('GD', 'Graduated'),]

    first_name = models.CharField(max_length=100)
    last_name = models.CharField(max_length=100)
    email = …
    date_of_birth = models.DateField()
    status = models.CharField(max_length=10, choices=STUDENT_STATUS_CHOICES, 
default='enrolled')

    def __str__ …

class Category(models.Model):
    name = models.CharField(max_length=100)

    def __str__ …

class Course(models.Model):
    title = models.CharField(max_length=200)
    description = models.TextField()
    instructor = …
    categories = models.ForignKey(Category)
    students = models.ManyToManyField(Student, blank=True)
    start_date = models.DateField()
    end_date = models.DateField()
    price = models.DecimalField(max_digits=10, decimal_places=2)
    level = …

    def __str__ …
```

2. Update the models:
    - Create a new model/class named Instructor. The model should contain the fields, *first_name*, *last_name*, *bio* (of type TextField) in addition to 2 relationship fields to both ***Course (one-tom-many)*** and ***Student*** (many-to-many).
    - Create a field *instructor* in the Course model to connect the courses to their instructors using one-to-many relationship.
    - Create a field level in the Course model with choices equal to a list COURSE_LEVEL_CHOICES with the values BE for beginner, IN for intermediate and AD for advanced. Similar to the choices in the Student model.
    - Create a field *email* in Student model with property `unique=True`. Create a field *bio* in the Instructor model of type TextField.
    - Write a suitable str() method for each model.

3. Create database tables: (make sure *myappF23* is included under INSTALLED_APPS in *settings.py*). See what happens after each step.

    a. **Tools → Run*manage.py***
    b. In the console, type **makemigrations carapp**. Then, type **sqlmigrate carapp 0001** #Check latest files in *migrations* dir. Finally, type **migrate**
    c. Check the migrations folder and notice the changes stored in "**0001_initial**" file.

4. Register your models in the admin page. Update *admin.py* as follows:

```
from django.contrib import admin
from django.db import models
from .models import Category, Course, Student, Instructor

# Register your models here.
admin.site.register(Category)
admin.site.register(Course)
admin.site.register(Student)
admin.site.register(Instructor)
```

5. Start your server (**Run → Run 'myappF23'**) and navigate to admin site (127.0.0.1:8000/admin). Click Tools menu → run manage.py task → type create superuser. Follow the prompts to enter username, email and password.

6. Login to the admin page using *superuser* name and password. Enter the data for each model, Topic, Course, Student, and Instructor through the admin interface. Choose the data that is suitable and meaningful to the model to the corresponding model.

Part 3: Querying the database.

1. Open **Python Console.** In **Python console** import Django then models from *models.py*.

```
import django
from myappF23.models import Student, Course, Instructor, Category
```

2. Write queries to obtain the following information. Verify if your query generates the correct answer using the data entered in question 2.5. Can you use `annotate` for any of these queries? explain to the grader.

   1. Write a separate query to list all the students, instructors, courses, and categories in the db.
   2. List all courses in a specific category (e.g., 'Programming')
   3. List all courses taught by a specific instructor (e.g., 'John Smith)
   4. List all courses with a price greater than $100.
   5. List all students enrolled in a specific course (e.g., 'Web Dev Bootcamp').
   6. List all courses with start dates in the future.
   7. Retrieve a specific student by email (e.g., 'john@gmail.com'):
   8. Retrieve a specific instructor by name (e.g., 'Saja Al-Mamoori'):
   9. List all courses with a specific difficulty level (e.g., 'Intermediate').
   10. List all courses that have the word 'Python' in their title.
   11. List all students who were born after a certain date (e.g., after January 1, 2000).
   12. List all courses that started after a specific date (e.g., after January 1, 2023).