SUBJECT: FULLSTACK DEVELOPMENT (21CS62)

**LAB COMPONENT SOLUTIONS**

**Module-1: MVC based Web Designing**

## Laboratory Component:

1. **Installation of Python, Django and Visual Studio code editors can be demonstrated.**

Python download and installation Link: <https://www.python.org/downloads/>

Visual Studio Code download and installation link: <https://code.visualstudio.com/>

Django installation:

Open a command prompt and type following command: pip install django

# Creation of virtual environment, Django project and App should be demonstrated

Follow these steps

* 1. Install the [Python extension.](https://marketplace.visualstudio.com/items?itemName=ms-python.python)- Open VS Code IDE and click extensions there automatically u will be shown Python extension (Make sure you are connected to Internet)
  2. On your file system, create a project folder
  3. In that folder, use the following command (as appropriate to your computer) to create a virtual environment named env based on your current interpreter:

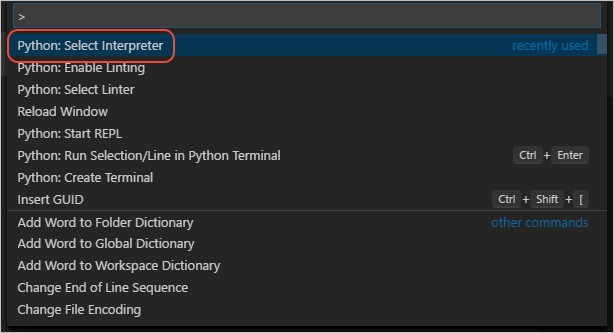
# Windows

python -m venv env

* 1. Open the project folder in VS Code by running code ., or by running VS Code

and using the **File** > **Open Folder** command.

* 1. In VS Code, open the Command Palette (**View** > **Command Palette** or (Ctrl+Shift+P)). Then select the **Python: Select Interpreter** command:



* 1. The command presents a list of available interpreters that VS Code can locate automatically (your list will vary; if you don't see the desired interpreter, see [Configuring Python environments](https://code.visualstudio.com/docs/python/environments)). From the list, select the virtual environment in your project folder that starts with ./env or .\env:
  2. Create a New Terminal : In Menu Terminal -> New Terminal option

## Creating project:

* + 1. Create a django project -

Type following command in the terminal opened: django-admin startproject p .

(dot following project name is important which refers to current directory)

This startproject command assumes (by use of . at the end) that the current folder is your project folder, and creates the following within it:

* manage.py: The Django command-line administrative utility for the project. You run administrative commands for the project using python manage.py

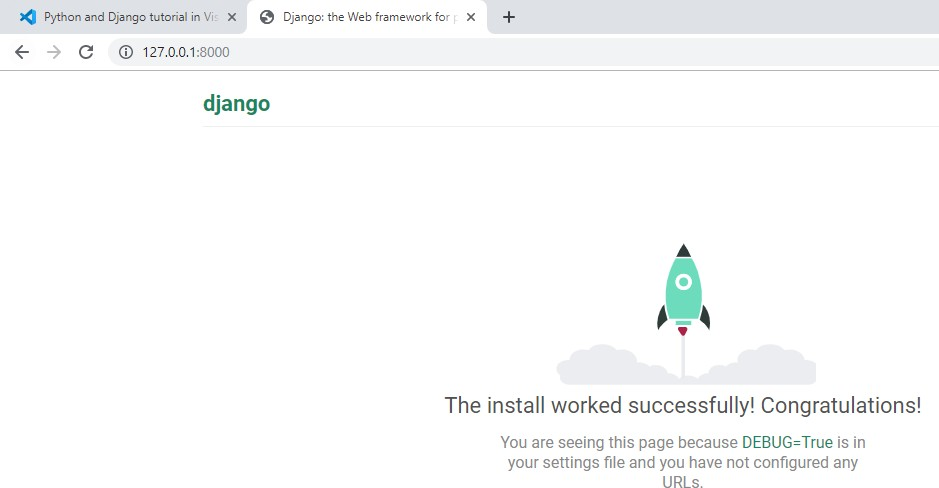
<command> [options].

* A subfolder named p which contains the following files:
  + init .py: an empty file that tells Python that this folder is a Python package.
  + wsgi.py: an entry point for WSGI-compatible web servers to serve your project. You typically leave this file as-is as it provides the hooks for production web servers.
  + settings.py: contains settings for Django project, which you modify in the course of developing a web app.
  + urls.py: contains a table of contents for the Django project, which you also modify in the course of development.
    1. To verify the Django project, make sure your virtual environment is activated, then start Django's development server using the

command python manage.py runserver. The server runs on the default port 8000, and you see output like the following output in the terminal window:

Verify server by typing:

python manage.py runserver



When you run the server the first time, it creates a default SQLite database in the file db.sqlite3, which is intended for development purposes but can be used in production for low-volume web apps. Also, Django's built-in web server is intended *only* for local development purposes. When you deploy to a web host, however, Django uses the host's web server instead. The wsgi.py module in the Django project takes care of hooking into the production servers.

If you want to use a different port than the default 8000, specify the port number on the command line, such as python manage.py runserver 5000.

* + 1. When you're done, close the browser window and stop the server in VS Code using Ctrl+C as indicated in the terminal output window.
    2. In the VS Code Terminal with your virtual environment activated, run the administrative utility's startapp command in your project folder (where manage.py resides):

python manage.py startapp lab1

* + 1. The command creates a folder called lab1 that contains a number of code

files and one subfolder. Of these, you frequently work with views.py (that contains the functions that define pages in your web app) and models.py (that contains classes defining your data objects). The migrations folder is used by Django's administrative utility to manage database versions. There are also the files apps.py (app configuration), admin.py (for creating an administrative interface), and tests.py (for unit tests).

# Develop a Django app that displays current date and time in server

In lab1 subfolder, make following changes to views.py:

from django.shortcuts import render from django.http import HttpResponse

# Create your views here. import datetime

def current\_datetime(request): now = datetime.datetime.now()

html = "<html><body><h1>It is now %s.</h1></body></html>" % now return HttpResponse(html)

In project named first, make following changes to urls.py

from django.contrib import admin from django.urls import path

from lab11.views import current\_datetime urlpatterns = [

path('cdt/', current\_datetime),

]

**Output:**



# Develop a Django app that displays date and time four hours ahead and four hours before as an offset of current date and time in server.

In lab11 subfolder, make following changes to views.py:

from django.shortcuts import render from django.http import HttpResponse

# Create your views here. import datetime

def current\_datetime(request): now = datetime.datetime.now()

html = "<html><body><h1>It is now %s.</h1></body></html>" % now return HttpResponse(html)

def four\_hours\_ahead(request):

dt = datetime.datetime.now() + datetime.timedelta(hours=4)

html = "<html><body><h1>After 4hour(s), it will be %s.</h1>"% (dt,) return HttpResponse(html)

def four\_hours\_before(request):

dt = datetime.datetime.now() + datetime.timedelta(hours=-4)

html = "<html><body><h1>Before 4 hour(s), it was %s.</h1>"% (dt,) return HttpResponse(html)

**In project named first, make following changes to urls.py**

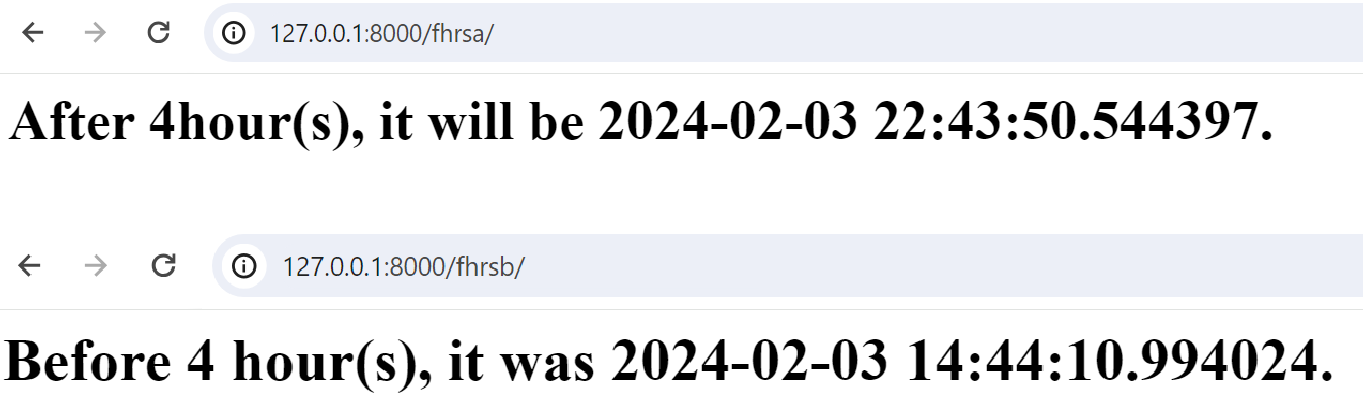
from django.contrib import admin from django.urls import path

from lab11.views import current\_datetime,four\_hours\_ahead,four\_hours\_before urlpatterns = [

path('cdt/', current\_datetime), path('fhrsa/',four\_hours\_ahead), path('fhrsb/',four\_hours\_before),

]

**Output:**



# Develop a Django app that displays date and time n hours in server.

# Input n=2 n=-4

# Develop a Django app that displays the message “Welcome to RNSIT”