**Regular expressions**

Regular expressions are extremely useful in extracting information from text such as code, log files, spreadsheets, or even documents. The first thing to recognize when using regular expressions is that everything is essentially a character, and we are writing patterns to match a specific sequence of characters (also known as a string). Most patterns use normal ASCII, which includes letters, digits, punctuation and other symbols on your keyboard like %#$@!, but unicode characters can also be used to match any type of international text.

**Some Regular expression notations**

**^:- Start**

**$:- Stop**

**.:- Any characters**

**\*:- Match one character 0+ times**

**+:- Match one character 1+ times**

**\s:- Whitespace**

**\S:- Non-whitespace**

**[abc]:- Match one character in the specified set**

**[^abc]:- Match one character not in the specified set**

**Python implementation**

**use re module**

examples

**1. Select all numbers from a string**

**import re**

**s = 'My name is Devi. I am 22 years and 5 months old.My email is devi@gmail.com'**

**result = re.findall("[0-9]+",s)**

**print(result)**

**Output**

**['22', '5']**

**re.findall("[0-9]",s)**

**Output**

**['2', '2', '5']**

**2. Extract only email id from a string**

**result = re.findall("\S+@\S+",s)**

**print(result)**

**output**

**['devi@gmail.com']**

**Study re module of python and prepare a study material.**

**What is a raw string in python? Explain with examples.**

**Python Virtual Environment**

A virtual environment is an isolated working copy of Python which allows you to work on a specific project without worry of affecting other projects. It enables multiple side-by-side installations of Python, one for each project. The main purpose of Python virtual environments is to create an isolated environment for Python projects. This means that each project can have its own dependencies, regardless of what dependencies every other project has.

**We can install virtual environment using pip (python package manager)**

**pip install virtualenvwrapper-win**

**Run the following command to verifiy if virtualenv is installed.**

**Virtualenv –version**

**Django**

Django is a python web framework to create web-based applications. It is free and opensource. It encourges rapid development of web based applications. Django is very secure. It follows MTV ( Model Template View) design pattern. Here

**Model is data access layer,**

**Template is presentation layer and**

**View is business logic layer**

**View is actually a bridge between model and template.**

**Django projects are divided into apps.**

**A django project can have one or more apps.**

**Django Installation**

**create a projects folder**

**cd projects**

**mkvirtualenv ve1**

**workon ve1**

**To install django use the following command**

**pip install django**

After installation **to start a django project** give the following command

**django-admin startproject djangoproject**

**A new folder with name djangoproject will be created.**

**cd djangoproject**

**In this folder** you can see another folder called djangoproject and a **python file manage.py**

**Inside the djangoproject folder** you can see the following files

**1. \_\_init\_\_.py**

**2. settings.py**

**3. urls.py**

**4. wsgi.py**

Django has it’s own webserver. To start this web server again go to the

folder and tpye the following command.

**python manage.py runserver**

**You will get the following screen**

**Performing system checks...**

**System check identified no issues (0 silenced).**

**You have 13 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.**

**Run 'python manage.py migrate' to apply them.**

**February 05, 2018 - 16:32:50**

**Django version 1.11.10, using settings 'djangoproject.settings'**

**Starting development server at http://127.0.0.1:8000/**

**4/13Quit the server with CONTROL-C.**

**[05/Feb/2018 16:36:17] "GET / HTTP/1.1" 200 1716**

**Not Found: /favicon.ico**

**[05/Feb/2018 16:36:17] "GET /favicon.ico HTTP/1.1" 404 1969**

**Now go to browser and give http://127.0.0.1:8000**

**and you will get a new page saying**

**It worked!**

**Congratulations on your first Django-powered page.**

To stop the server press ctrl C.

Now download XAMPP installer and install it.

After that install mysqlclient using the following command

**pip install mysqlclient**

After the installation open the **settings.py** file and **edit** it.

**Comment the DATABASES claues and insert the following**

**DATABASES = {**

**'default': { 'ENGINE': 'django.db.backends.mysql',**

**'NAME': 'django',**

**'USER': 'root',**

**'PASSWORD' :'mad123',**

**'HOST' : 'localhost',**

**'PORT' : ''**

**}**

**}**

**Here NAME is name of database.**

**Create a new database in mysql/mariadb using phpmyadmin and give**

**the name of that database here.**

While running the weserver we have got a message as follows

“You have 13 unapplied migration(s). Your project may not work

properly until you apply the migrations for app(s): admin, auth,

contenttypes, sessions.

**Run 'python manage.py migrate' to apply them.”**

Here migrations are Django’s way of propagating changes you make to

your models (adding a field, deleting a model, etc.) into your database

schema. They’re designed to be mostly automatic, but you’ll need to

know when to make migrations, when to run them, and the common

problems you might run into.

Now we can apply these migrations using the command

**python manage.py migrate**

Now **open the database again in phpmyadmin**. You can see that some

tables were created automatically.

| Tables\_in\_django

|

+----------------------------+

| auth\_group

|

| auth\_group\_permissions

| auth\_permission

| auth\_user

|

|

|

| auth\_user\_groups

|

| auth\_user\_user\_permissions |

| django\_admin\_log

|

| django\_content\_type

|

| django\_migrations

| django\_session

|

|

**Now run again our webserver**

**python manage.py runserver**

**and browse localhost:8000/admin**

You will get the admin panel of django. To login to the admin panel we have to create a superuser account first using the command,

**python manage.py createsuperuser -username=hadi –email=hadizcodez@gmail.com**

**system will ask for password.**

**Give an eight character password twice.**

**After that go to browser and browse localhost:8000/admin and login**

**using the new credentials.**

Now you can see the dashboard of django administration panel.

Now we will create a new app of our project.

To **create an app** do as follows

**python manage.py startapp sample**

sample is name of the app

Now you can see a new **folder** called **sample** with following files.

**admin.py**

**apps.py**

**\_\_init\_\_.py**

**migrations**

**models.py**

**tests.py**

**views.py**

Here **migartions** is a **folder** again.

Now will start **creating a new app** named sample with an index page

displaying **Welcome to Xanthron**.

**Steps**

1. **open settings.py** file and add the following in INSTALLED\_APPS section.

**‘sample’,**

and this section will be look like as follows.

**INSTALLED\_APPS = [**

**'sample',**

**'django.contrib.admin',**

**'django.contrib.auth',**

**'django.contrib.contenttypes',**

**'django.contrib.sessions',**

**'django.contrib.messages',**

**'django.contrib.staticfiles',**

**]**

2. **Open the file urls.py** in the djangoproject folder and add the following line in urlpatterns section and import a method include.

**Edit the first line from django.conf.urls import url as follows**

**django.conf.urls import url,include**

**and add the following line to urlpatterns sections**

**url(r'^sample/', include('sample.urls')),**

**It means, if you browse localhost:8000/sample go to the urls specified in the urls.py file in sample app.**

**We have to create a new file named urls.py in sample folder.**

**And add the following lines**

**from django.conf.urls import url**

**from . import views**

**urlpatterns = [**

**url(r'^$', views.index),**

**]**

Here we import views.py from the sample folder. And created the index page by saying in url that after the sample if there is nothing(meaning of ^$) execute the index function in the views of sample.

Now we have to create the index funtion in views.py of sample.

**from \_\_future\_\_ import unicode\_literals**

**from django.http import HttpResponse**

**from django.shortcuts import render**

**# Create your views here.**

**def index(request):**

**response = HttpResponse()**

**response.write("Welcome to Xanthron")**

**return(response)**

**Again run the webserver using python manage.py runserver and browse localhost:8000/sample.**

**The message “Welcome to Xanthron” will be displayed.**

We have created our first app in django without any database access. Now we will study how to use database in our django app. We have to create a model for this purpose.

You can see a **models.py** file inside the **sample folder**.

Model is the data layer of our app. In this we can define our database and tables and facility to query our database. A models.py can contain any number of models. Actually, a model is a class inherited from django.db.models.Model, and is used to define fields as class attributes. We can consider models as spreadsheets with rows and columns. Columns are fields and rows are records.

**Now we can see how to create a model of students in a class with following attributes.**

**Name of Student, Gender, Age.**

**Open the models.py** file to edit and enter the following.

**from \_\_future\_\_ import unicode\_literals**

**from django.db import models**

**# Create your models here.**

**class Student(model.Model):**

**stud\_name = models.CharField(max\_length=50)**

**stud\_gender = models.CharField(max\_length=1)**

**age = models.IntegerField()**

Here you can see that we have used some **key words for specifying the data** types of each **data field**.

**CharField() : VarChar**

**IntegerField() : Int**

**For various other data types we use :**

**TextField() : Text**

**EmailField() : email**

**URLField() : url**

**FileField() : for files**

**ImageField() : images**

**BooleanField() : boolean**

**DateTimeField : datatime**

**Attributes of fields: max\_length, null, blank, default and choices**

We have **created our model** description in models.py file. Now we have to create the script to do the action as described in models.py to our database. For this purpose we have to **create migration scripts**

**using the command**

**python manage.py makemigrations sample**

**To run this migation script use the command**

**python manage.py migrate**

**After that you check the folder migrations in sample folder. You can see the following files newly created**

**0001\_initial.py, 0001\_initial.pyc, \_\_init\_\_.py, \_\_init\_\_.pyc**

**and go to phpmyadmin and check the database for the new table sample\_student with following fields.**

+-------------+-------------+------+-----+---------+----------------+

| Field

| Type

| Null | Key | Default | Extra

|

+-------------+-------------+------+-----+---------+----------------+

| id

| int(11)

| NO | PRI | NULL | auto\_increment |

10/13| stud\_name | varchar(50) | NO | | NULL | |

| stud\_gender | varchar(1) | NO | | NULL | |

| age

| int(11)

| NO |

| NULL |

|

+-------------+-------------+------+-----+---------+----------------+

4 rows in set (0.00 sec)

Now we can register this model to the django’s buit-in app admin. For this purpose **open the admin.py** file and enter the following

**from \_\_future\_\_ import unicode\_literals**

**from django.contrib import admin**

**from .models import Student**

**admin.site.register(Student)**

Check your admin account using the url localhost:8000/admin and super user credentials. Click on Student under the Sample title and click on Add student link and start adding records without any extra coding.

After seeing this page make some more changes to tha admin.py and examin the changes in the admin display pages

**Create a new class StudentAdmin**

**from \_\_future\_\_ import unicode\_literals**

**from django.contrib import admin**

**from .models import Student**

**class StudentAdmin(admin.ModelAdmin):**

**list\_display = ['stud\_name','stud\_gender','age']**

**admin.site.register(Student,StudentAdmin)**

**save this file and check the difference in admin app.**

**Please report the differnces occured in admin app and send the last two screenshots(before and after the last change) of the display pages of admin app.**

Now you can query through this database easily.

Goto the command prompt and give the command

**python manage.py shell**

You will get the python shell with the following prompt

**Python 2.7.13 (default, Nov 23 2017, 15:37:09)**

**[GCC 6.3.0 20170406] on linux2**

Type "help", "copyright", "credits" or "license" for more information.

**(InteractiveConsole)**

>>>

**Here try the following commands and feel how easy to query through the database tables in python django.**

**>>>from sample.models import Student**

**>>> students = Student.objects.all()**

**>>> student = students[0]**

**>>> student.stud\_name**

**'Hadi'**

**>>> student.stud\_gender**

**'M'**

**>>> student.age**

**33**

**>>>**