**Flask Notes :-> detailed ones**

----------------------------------------------------------Setup-----------------------------------------------------------------------

Install Postgres database

command:

psql -U postgres postgres

(Server) (database)

**Link to install all needed library using pip install name can be found in this link:-**

**http://flask.pocoo.org/extensions/**

Install Virtualenv to intall flask

password for anything is Fresh@31

Virtualenv create isolated python dev environment on your computer so that when we install some library we will not

afftect other installed libraries

Command:

virtualenv venv

pip install flask

then change directory to \Scripts:

and type -> activate

---------------------------------------------------------------------------

For heroku first signup at heroku

cred: thesumedh

pass Fresh@31

then download toolkit

then at cmd type command:

heroku login

then enter username and password

--------------------------------------------------------------------------------------------------------------------------------------

Making Templates:->

we write parent template once and inherit it in child template

----------\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_New Notes\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#Render template is used to return html pages

from flask import Flask,render\_template

app = Flask(\_\_name\_\_)

# '''#new instance of flask class saved it in app variable'''

@app.route("/") #'''here we map / to function index which will return page index.html when we type / '''

@app.route("/home")

def home():

return render\_template("home.html")

@app.route("/about")

def about():

return "About Page"

if \_\_name\_\_=='\_\_main\_\_':

app.run(debug=True) ) #'''app.run runs the app on local server in debug mode'''

#so that flask will know where to look for, for static files,templates

**Note:**

1) <link href="{{ url\_for('static', filename='css/main.css') }}" rel="stylesheet"> <!--here url\_for() is a flask function which goes to static folder and check for main.css -->.

2) The flask command is installed by Flask, not your application; it must be told where to find your application in order to use it. The FLASK\_APP environment variable is used to specify how to load the application.

3) Writhing Python code in Html is called jinja 2 it can be done using code block {%write code here %}

4)for printing that variable we use {{ write variable here}}

Eg:

from flask import Flask,render\_template

app = Flask(\_\_name\_\_)

#dummy data

posts = [

{

"author": "Sumedh D",

"title":"Post 1 ",

"content":"First port",

"date":"April 20 2018",

},

{

"author": "SD",

"title":"Post 2 ",

"content":"Second port",

"date":"April 25 2018"

}

]

@app.route("/")

@app.route("/home")

def home():

return render\_template("home.html",postss=posts)

#this postss variable will be accessed in html templates to get data from posts list

@app.route("/about")

def about():

return render\_template("about.html")

if \_\_name\_\_=='\_\_main\_\_':

app.run(debug=True)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_html file\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<!DOCTYPE html>

<html>

<head>

<title>Home Page</title>

</head>

<body>

{%for post in postss%}

<h1>{{post.title}}</h1>

<p>By {{post.author}} on {{post.date}}</p>

<p>{{post.content}}</p>

{%endfor%}

</body>

</html>

**Note:** We will create layout file for the template which is common to all the pages

Eg:

<!DOCTYPE html>

<html>

<head>

{% if title %}

<title>Flask Blog - {{title}}</title>

{% else %}

<title>About Page</title>

{% endif %}

</head>

<body>

**{% block content %} {% end block %}**

</body>

</html>

2) means on the part where block content is there will be modified.

3)for the child html file we use {% extends “layout.html”%} i.e parent file name.

4) then inside the block content (content is just a name it can be anything) we write new code

Eg : home.html

{% extends "layout.html" %}

{% block content %}

{%for post in postss%}

<h1>{{post.title}}</h1>

<p>By {{post.author}} on {{post.date}}</p>

<p>{{post.content}}</p>

{%endfor%}

{% endblock content %}

---------------------------**Adding bootstrap**------------------------------

5) we do copy and paste of some tags from getbootstrap.com

6)In flask Static file like css and javascript need to located in a static folder.

-------------------------------------------creatimg a signup form\_\_\_\_\_\_\_\_\_\_\_-----------------------------------------------

For creating form in python

**From flask\_wtf import FlaskForm**

Use it as:

**class RegistrationForm(FlaskForm):**

For Username field

**From wtforms import StringField**

To add validations in the field:

**From wtforms.validators import DataRequired, Length, Email, EqualTo**

Use it as:

**username = StringField('Username', validators=[DataRequired(), Length(min=2, max=20)])**

**email = StringField('Email',validators[DataRequired(),Email()])**

For adding password Field:

**From wtforms import StringField, PasswordField**

Use it as

**password =PasswordField('Password', validators=[DataRequired()])**

**confirm\_password =PasswordField('Confirm Password', validators=[DataRequired(),EqualTo('password')])**

For Submit Field:

**From wtforms import StringField, PasswordField, SubmitField**

Use it as:

**submit = SubmitField('Sign Up')**

To remember the user import BooleanField:

**From wtforms import StringField, PasswordField, SubmitField, BooleanField**

Use it as:

**remember = BooleanField('Remember me')**

**Note:-**

To avoid cross site requests , Forgery attacks we use Secret key:

Inside route file add line , below app variable

**app.config['SECRET\_KEY'] = '** **'53b73a29a3667dc1ea23f4a484ea' '**

To assign this Secret key to random value use following line

**Open cmd and start python interpreter**

**Then**

**-> import secrets**

**-> secrets.token\_hex(14)**

**'53b73a29a3667dc1ea23f4a484ea'**

**Note:**

1){{ form.hidden\_tag() }} <!--for csrf protection (cross site request forgery token)-->

2) <legend class="border-bottom mb-4">Join Today</legend>

**<!--mb-4 is margin bottom all bootstrap classes-->**

**<div class="form-group">**

3) {{ form.username.label(class="form-control-label") }}

**<!--form-control-label is a bootstrap class for decoration-->**

4) The **HTML** <**fieldset**> **tag** is used for grouping related form elements. By using the **fieldset tag** and the <legend> **tag**, you can make your forms much easier to understand for your users

<fieldset> will draw a border around input fields

**from flask import Flask,render\_template,url\_for,flash**

@app.route("/register", methods=['GET','POST'])

# here we specify methods as Get and post so that these operations will be allowed

def register():

form = RegistrationForm()

if form.validate\_on\_submit():# checking validity of content

flash(f'Account created for {form.username.data}!', 'success') # flash message will be displayed

return redirect(url\_for('home'))

return render\_template('register.html', title='Register', form=form)

**----------------------------------------To redirect the page -----------------------------------------------------------**

**from flask import Flask,render\_template,url\_for,flash, redirect**

use in code:

**return redirect(url\_for('home'))**

**To Display the message using layout.html:-**

{% with messages = get\_flashed\_messages(with\_categories=true) %}

{% if messages %}

{% for category, message in messages %}

<div class="alert alert-{{ category }}">

{{ message }}

</div>

{% endfor %}

{% endif %}

{% endwith %}

-------------------display individual error-----------------------------------------------------

<div class="form-group">

{{ form.username.label(class="form-control-label") }}

<!--form-control-label is a bootstrap class for decoration-->

{% if form.username.errors %}

{{ form.username(class="form-control form-control-lg is-invalid") }}

<div class="invalid-feedback">

{% for error in form.username.errors %}

<span>{{ error }}</span>

{% endfor %}

</div>

{% else %}

{{ form.username(class="form-control form-control-lg") }}

{% endif %}

</div>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Final Code files with notes**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_