Python Flask Framework

A Step by Step Guide For the Beginners

By

Vasanth Nagarajan

CTO – Pinesphere Solutions

vasanth@pinesphere.com

ABOUT THIS BOOK :

This book is for total beginners who are interested to learn the python flask framework,this book does not contain any advanced concepts of the flask framework and

the advanced python programming concepts.This Book is totally focused on the beginners who are little bit familiar with the python programming and are interested to learn the python framework to develop the web applications .

About ME :

Myself Vasanth Nagarajan ,Working as an Software Engineer in KGiSL,My Area of Domain is Machine Learning & Artificial Intelligence ,Since I have much interested in python programming language ,i came up with an idea to write a book regarding the python and python framework ,to help the veterans who likes to start the python framework from the start.

If you found any mistakes or would you like to leave a feedback regarding this book , You are always welcome you can drop me a mail to the following mail id's will reply it as soon much as possible .

vasanth.n@kgisl.com
nvasanthnagarajan@gmail.com

Happy Programming

Prerequisites:

Before you start proceeding with this tutorial, I assuming that you have hands-on experience on HTML ,CSS,BootStrap, Python and Mysql . If you are not well aware of these concepts, then I suggest you to go through some short tutorials on those topics.

Flask:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Web Framework?

Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

WSGI

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

Werkzeug

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

Jinja2

jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support. Instead, Flask supports the extensions to add such functionality to the application.

WTForms

One of the essential aspects of a web application is to present a user interface for the user. HTML provides a <form> tag, which is used to design an interface. A Form's elements such as text input, radio, select etc. can be used appropriately.

Data entered by a user is submitted in the form of Http request message to the server side script by either GET or POST method.

- The Server side script has to recreate the form elements from http request data. So in effect, form elements have to be defined twice – once in HTML and again in the server side script.
- Another disadvantage of using HTML form is that it is difficult (if not impossible) to render the form elements dynamically. HTML itself provides no way to validate a user's input.

This is where WTForms, a flexible form, rendering and validation library comes handy. Flask-WTF extension provides a simple interface with this WTForms library.

Using Flask-WTF, we can define the form fields in our Python script and render them using an HTML template. It is also possible to apply validation to the WTF field.

LET'S GET STARTED

Flask

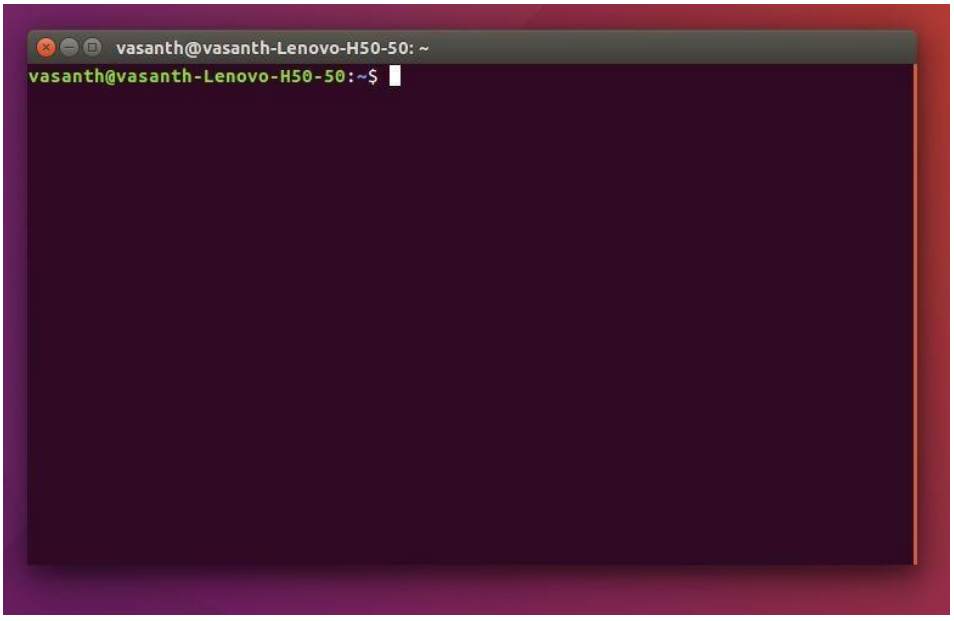
Note :

This Book Explains these basic concepts of Flask and Sql in Ubuntu Operating System (Linux),since am the linux user i have prepared this tutorial based on that,you no need to worry about the platform that you are using,the code works well in both the platform ,except the python packages and flask packages and the libraries of the wtforms and sql connectors installation alone differ rather than that the entire code works well.

If you are windows user please see the note commands that used in the book ,through the installation of the packages that used here ,follow the windows commands to install the specific packages ,if u still experiencing the errors ,please drop me a mail will help you out .

Installation Process :

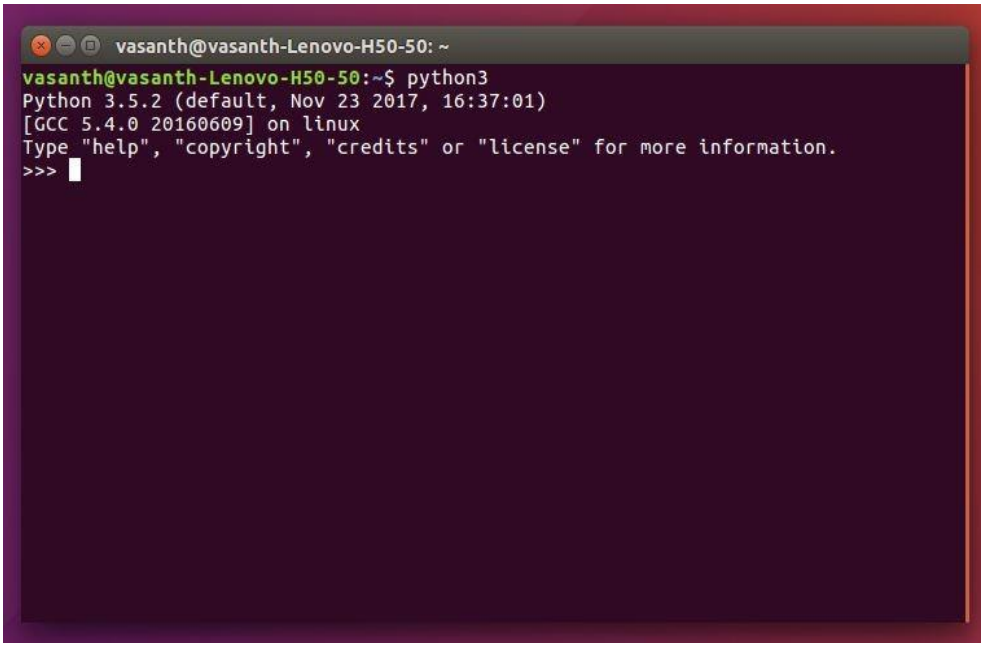
If your using Ubuntu 16 or any other linux latest platform the python has been already preinstalled in the operating system by the default .let's check the python version that present in the system just click the **ctrl + alt + T** in the keyboard or just type Terminal in the search bar in the ubuntu . a window will appear as shown below :



Now Type **Python3** in the Terminal as shown below :

vasanth@vasanth-Lenovo-H50-50:~$ python3

You will see a window as shown below prompting to enter command as shown below :



If this windows appear as shown above ,this means you have python3 already installed in the system.

If You are using **windows** python is not pre installed in your system so now go to the python download page for windows by clicking this link https://www.python.org/downloads/windows/ you will see a list of python files to download ,select the python version 3.7.0

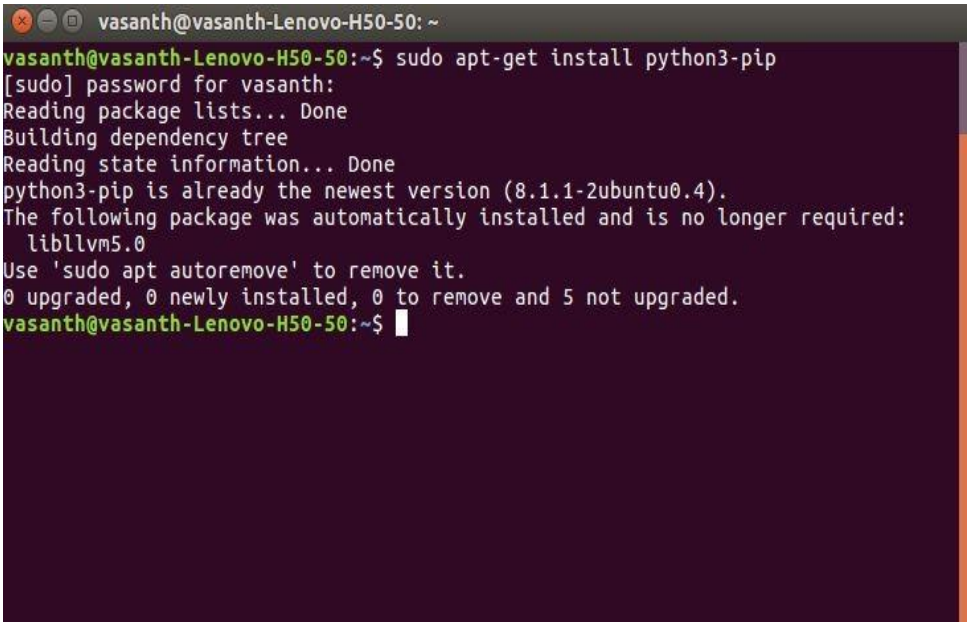
And install the python application in the desired path .(if you have any queries just drop me a mail)

Python Pip :

Pip is the python package manager and its used to install the packages that required for the development for the flask application and its also used to install the specific libraries to the python , to install the pip in ubuntu you have to type the following code in the terminal . just open the terminal and type the following code :

vasanth@vasanth-Lenovo-H50-50:~$ sudo apt-get install python3-pip

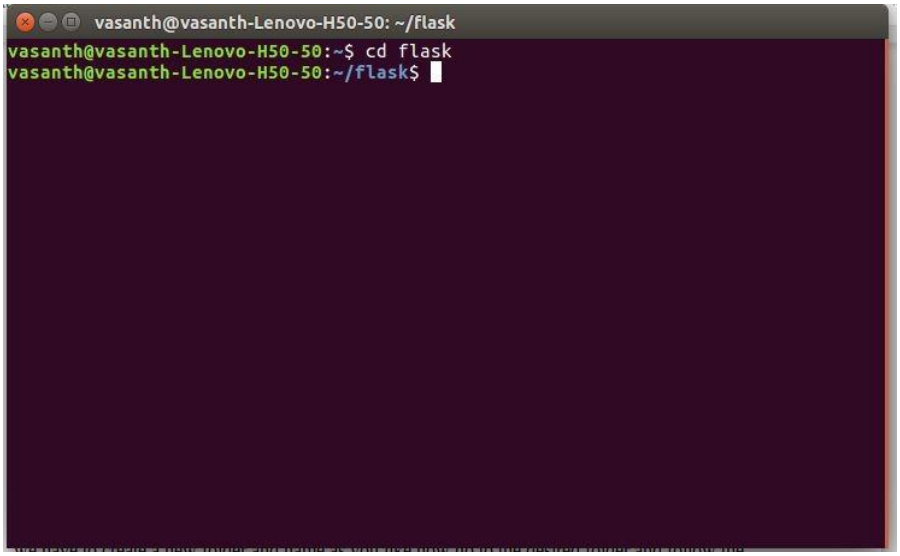
Now the necessary packages will be installed ,if the packages are already there u will get the below message :



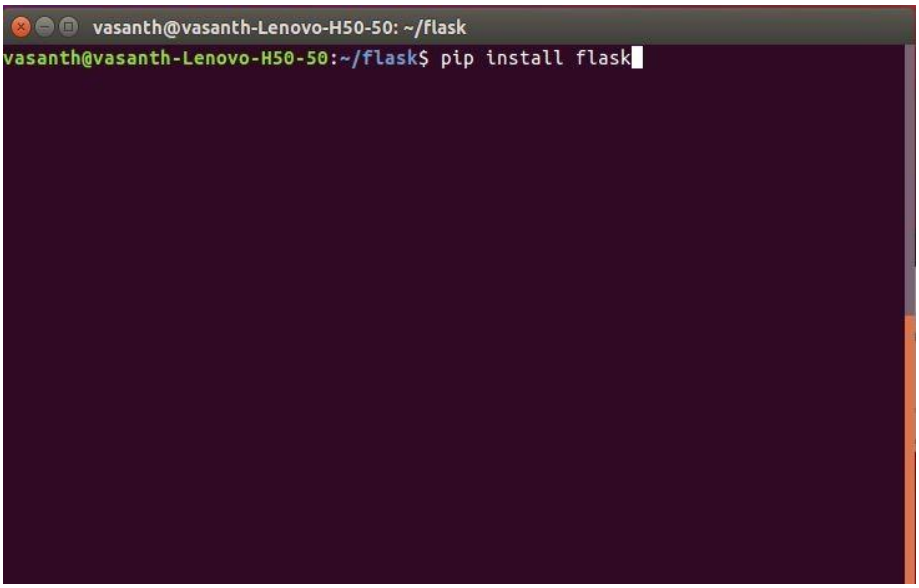
Installing Flask In the Machine :

Now We are going to install the Flask in our machine ,first we have to create a new folder and name as you like,now go to the desired folder and follow the instructions as shown below :

In my case the folder i had created in the home directory now i changed my working directory to my flask folder where am going to install flask and develop the application .

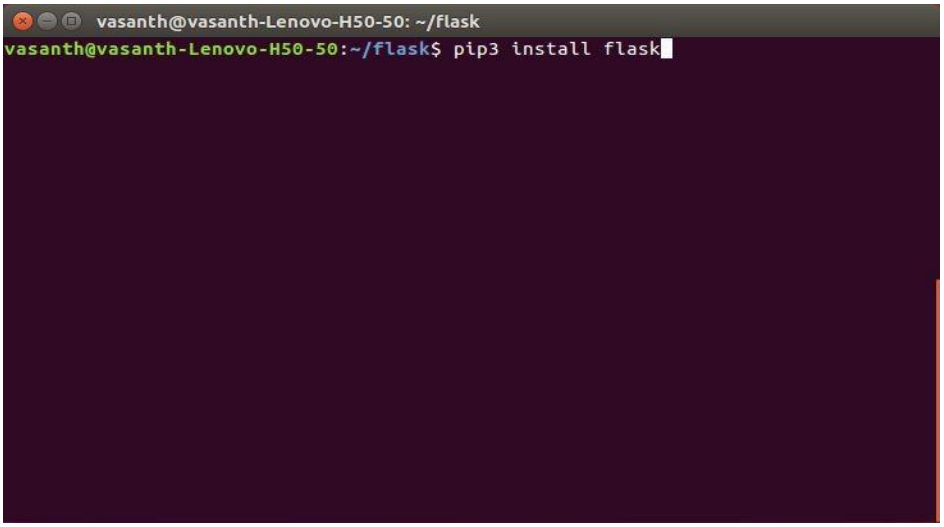


Now am installing the flask in the machine using the following command :



We can install flask using the pip python package manager . we have to install for python2 and python3

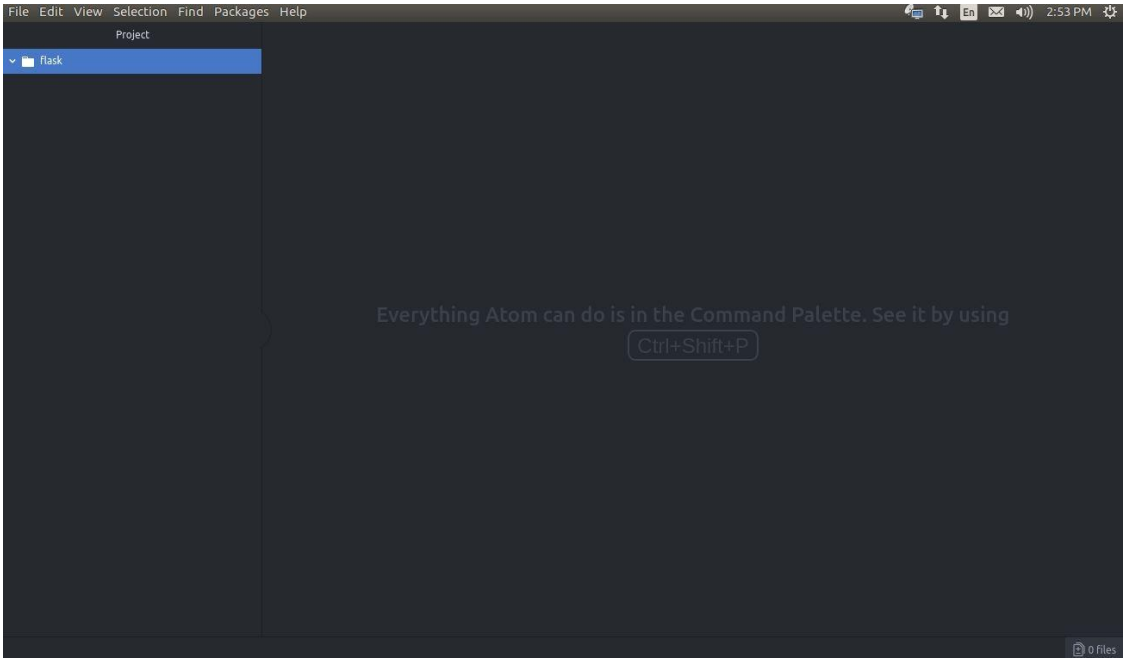
Now type the following command in the command prompt :



Now open the flask project folder using any python IDE,here i use the atom ide ,its free and open source, if you want to download the atom ide ,please click the following link and download and install the atom in your machine , the link is : https://atom.io/

After installed the atom now open the flask folder in the file menu where we going to create the flask application , the example is shown below :

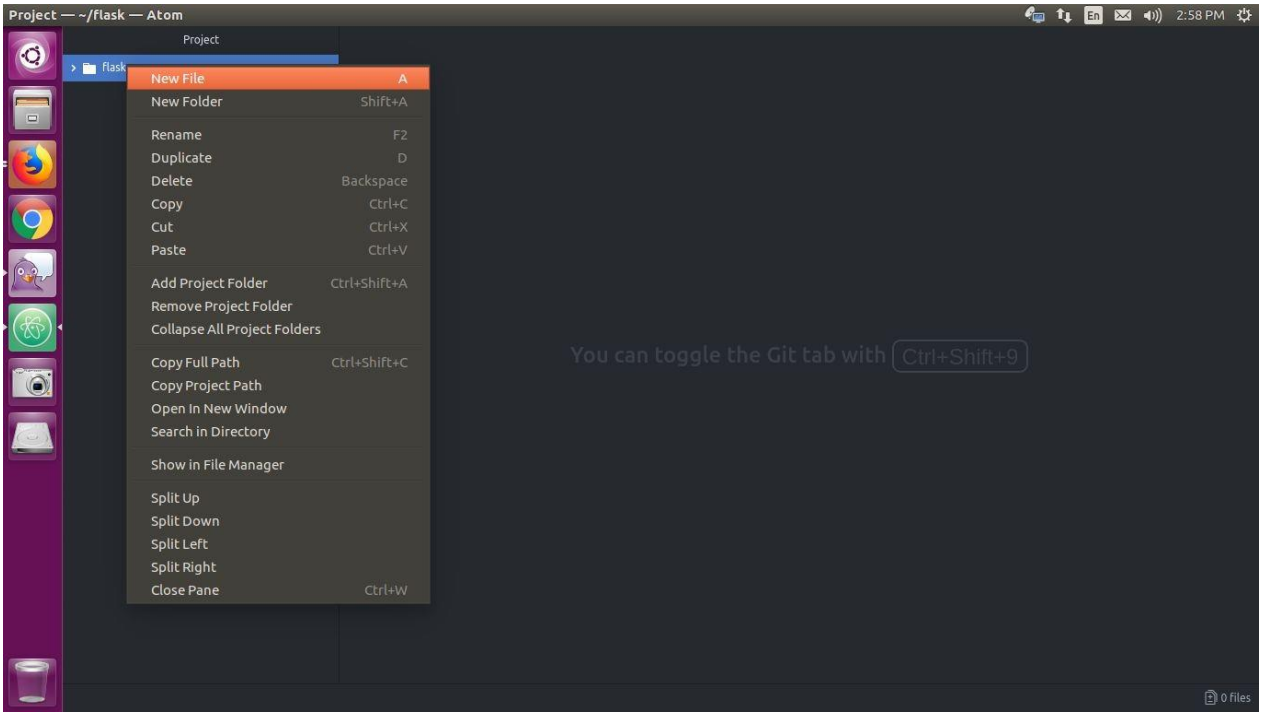
Use file------add project folder option in the file menu



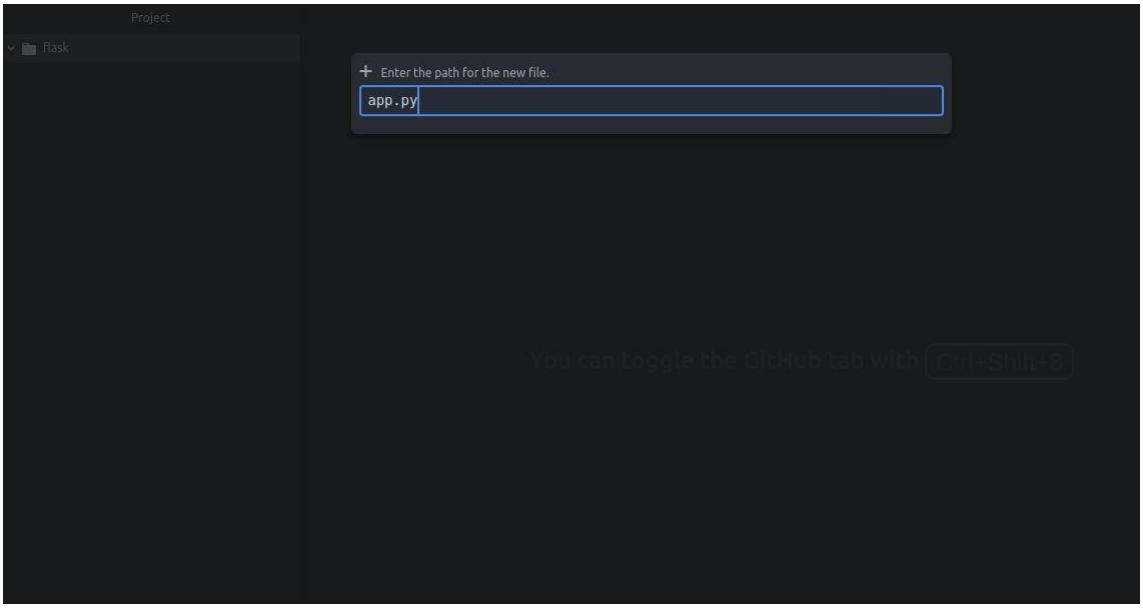
Creating First Flask Application :

Now we are going to create our first flask application ,follow the instructions as i said, if you have any queries regarding this you can mail me will reply you as soon as possible .

Now create a file in the ide inside the flask folder as show below :

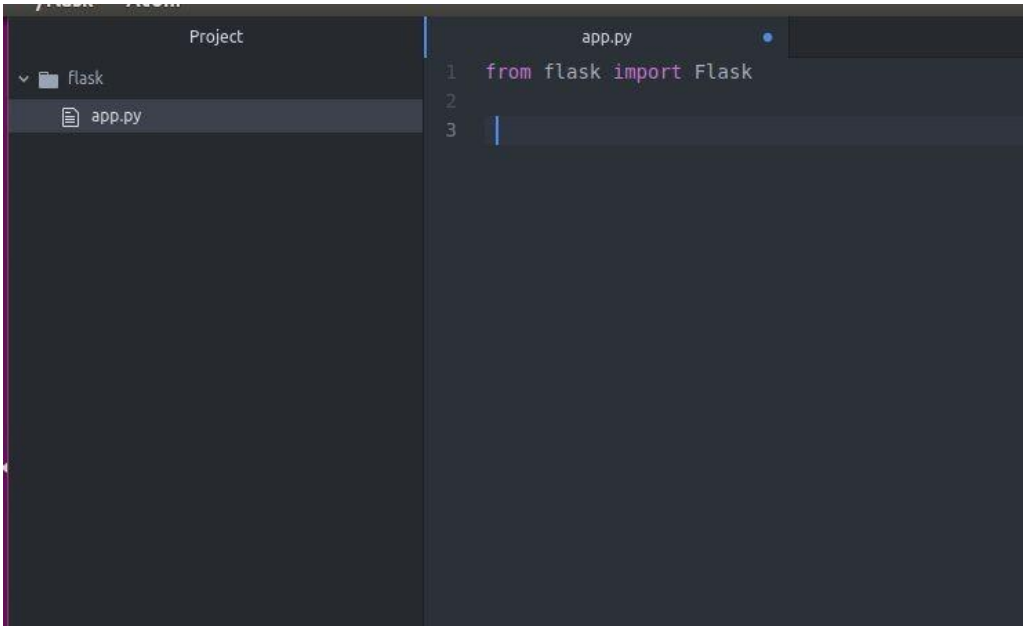


Now name the file as **app.py** NOTE : Python File Extensions will have .py as file format

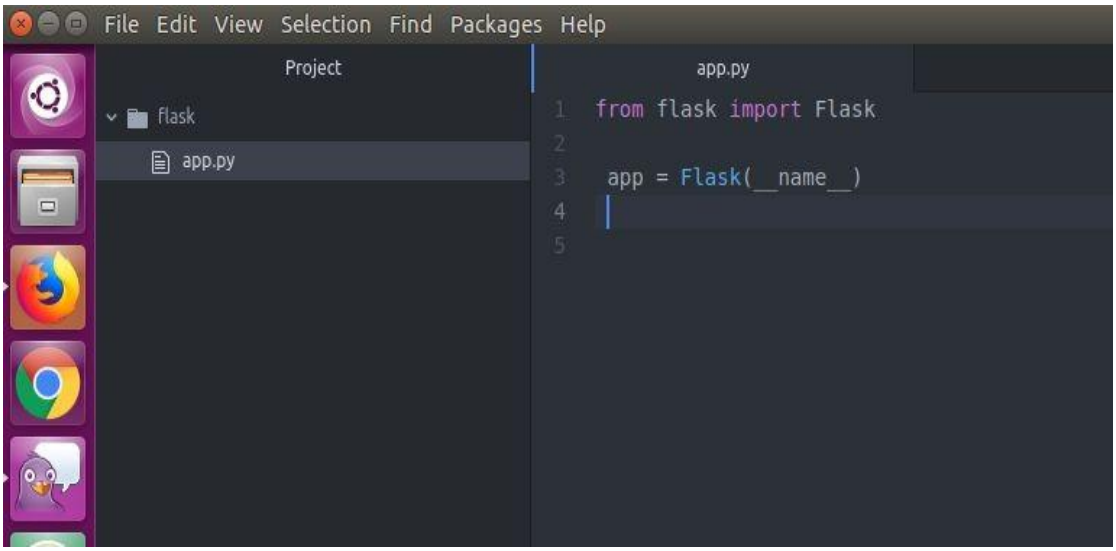


Now you have created a python file called **app.py** ,so this is the basic file or entry file for flask applications.

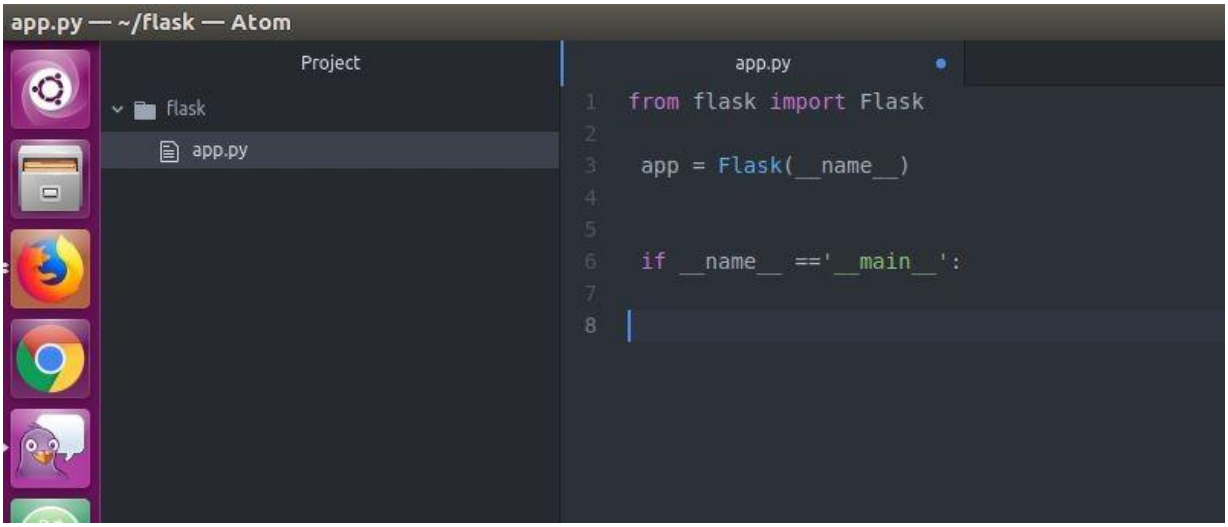
Now we have to import the flask to our app.py file for that you have to type the following command.



Now we want to create the instance of the flask class this is kind of place holder for the current module.



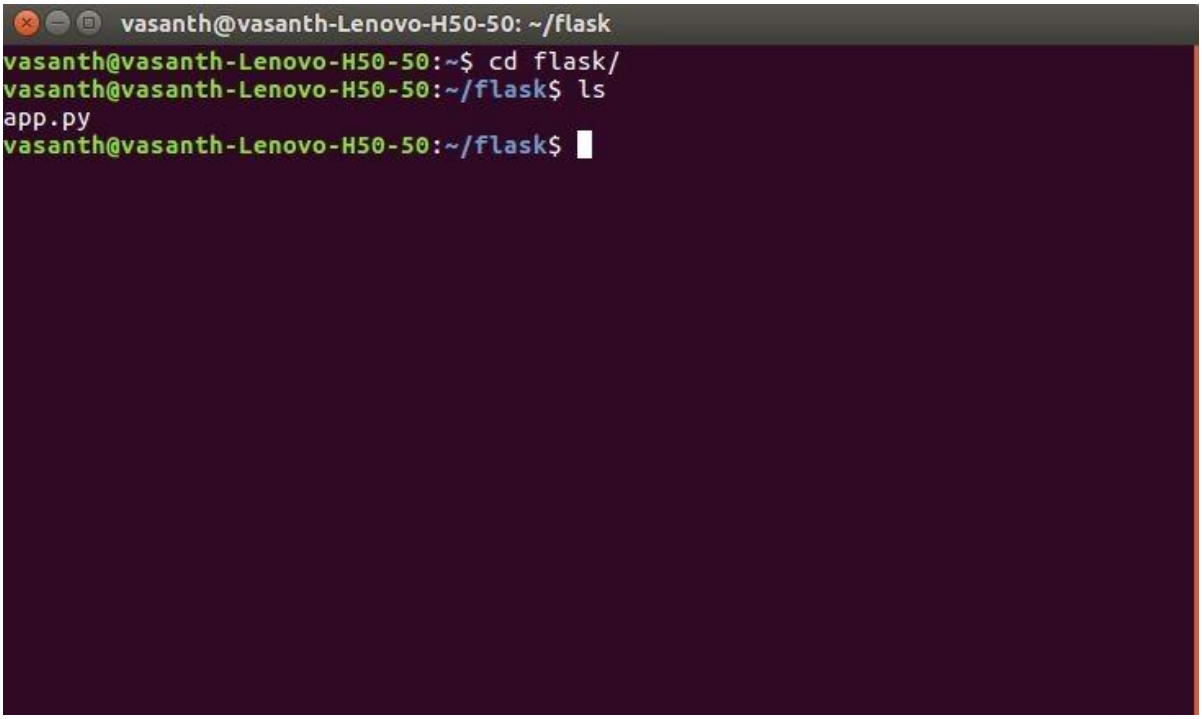
And down at the bottom we want to test to see if that value that double underscore name value is equal to double underscore main ,that's the script that's going to be executed.



Now type the code after the :

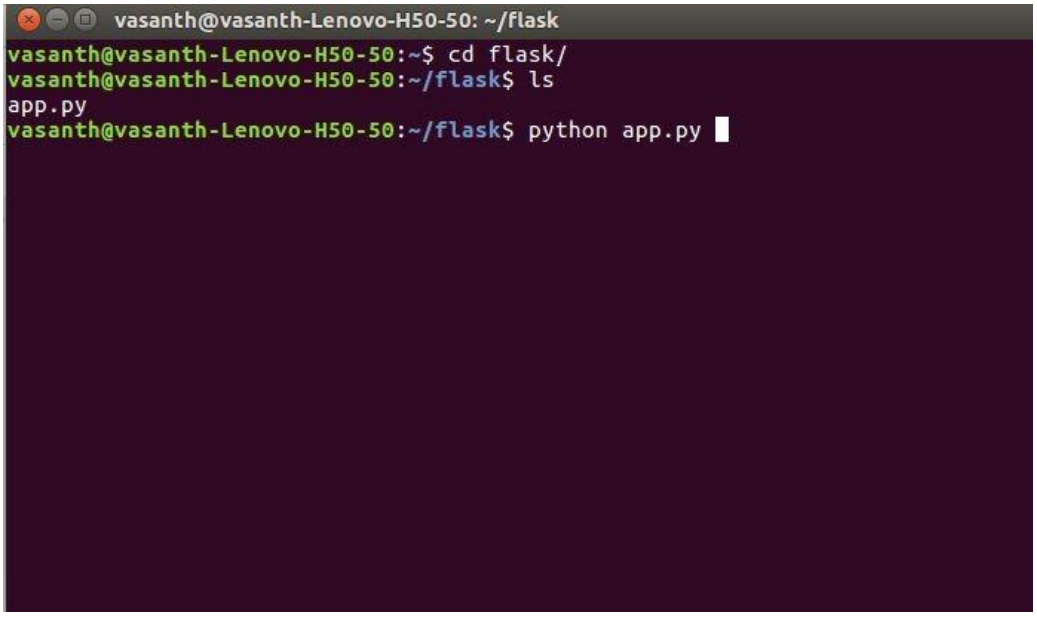
if __name__== '_main_':
app.run()

Now we are going to run our flask application,for that now we have to open a terminal using **ctrl+alt+T** now the terminal will open now navigate to our flask application folder where we have created the app.py file as shown below :



Now run our app.py file to check our flask application is working for that type the following command .

vasanth@vasanth-Lenovo-H50-50:~/flask$ python app.py



Now you can see the server is running without any errors as shown below :

Now you can type the above address in your web browser and press enter or you can type http://localhost:5000/ in the browser and hit enter now the browser will display the following content :

Now we need to create a route for our home page ,now go to our app.py file and type the following code :

```
@app.route('/')
```

And below that we have to create a function :

```
def index():
        return 'Hello World!!!'
```

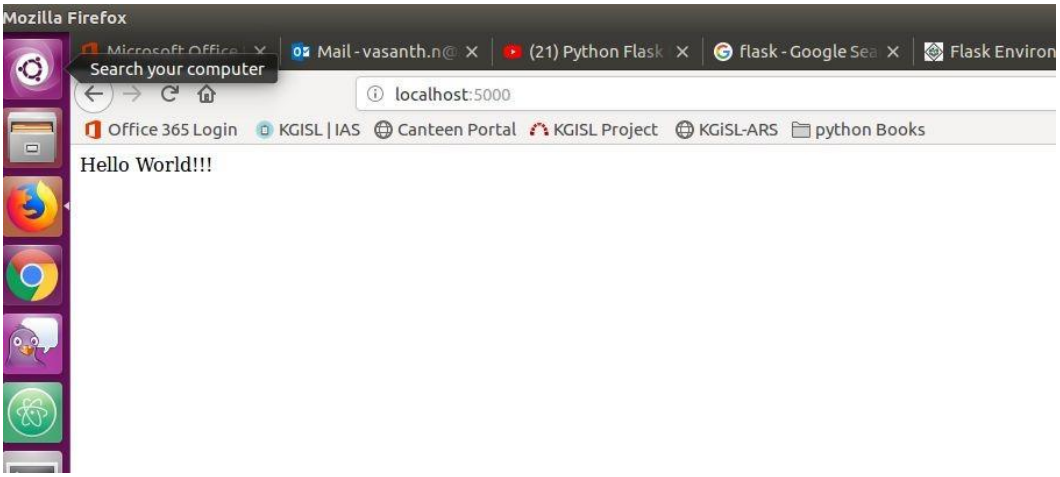
Now after typing the above code ,go to the terminal and stop the server using the following command ctrl+c .

Now start the server again by typing :

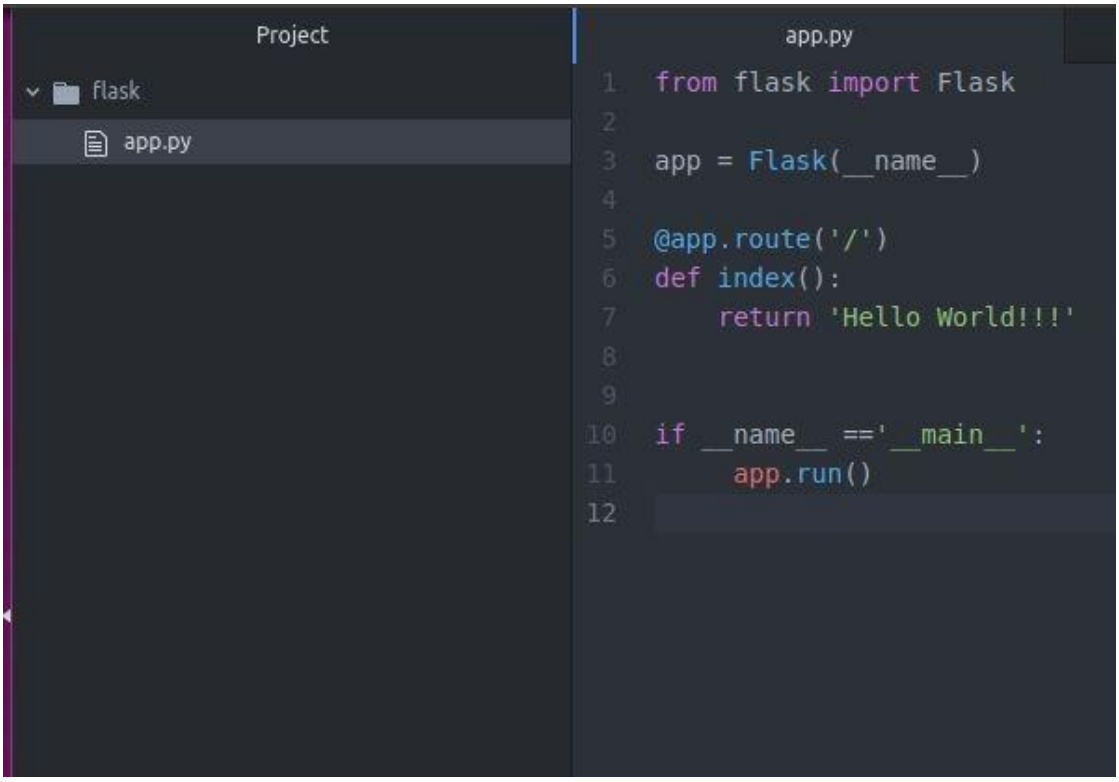
vasanth@vasanth-Lenovo-H50-50:~/flask$ python app.py

Now you can type the above address in your web browser and press enter or you can type http://localhost:5000/ in the browser and hit enter now the browser will display the following content :

The browser will display the **Hello World !!!** as shown below

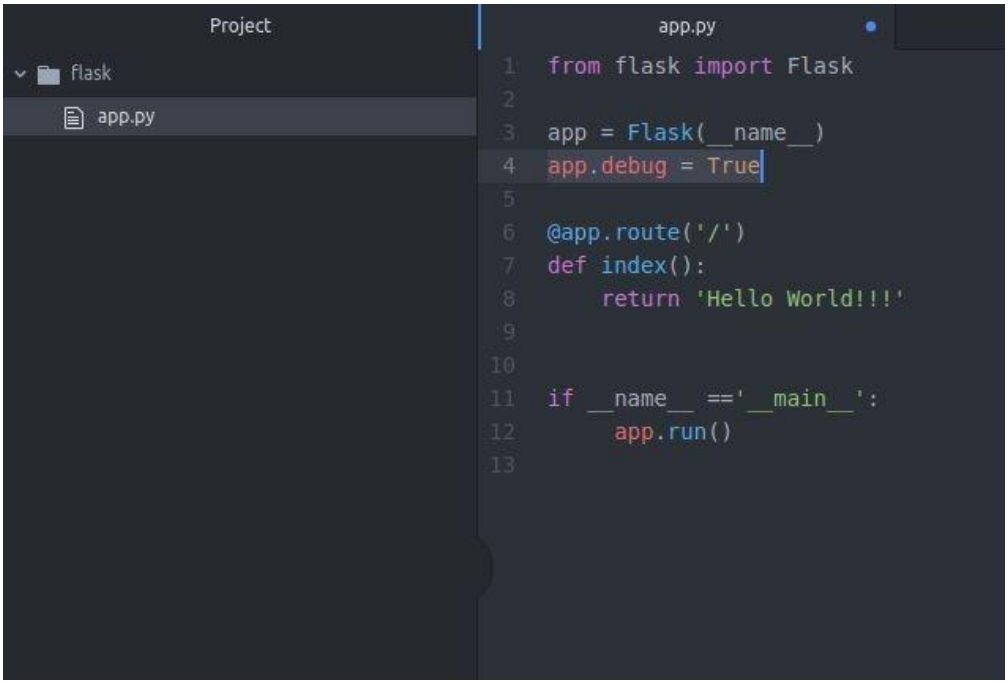


We have successfully installed the flask and we run the flask application as shown above , now the entire code will look like :



We will make some changes often in the app.py file,each and once we have to stop and start the server manually, this is hepatic for this we can make the server auto refresh ,when each time the changes has been made ,for this we have to type the following code in the app.py file.

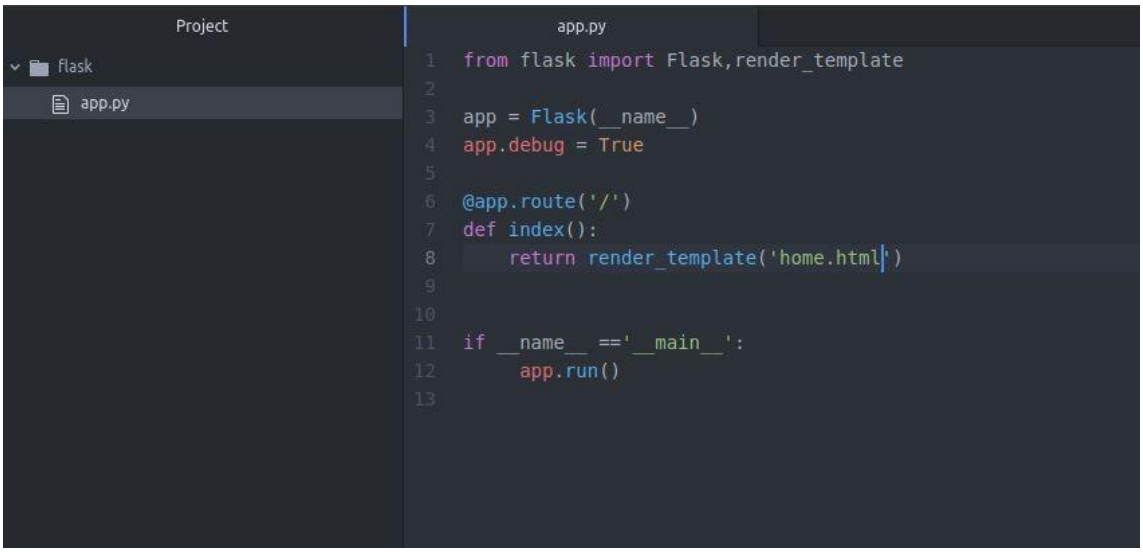
app.debug = True



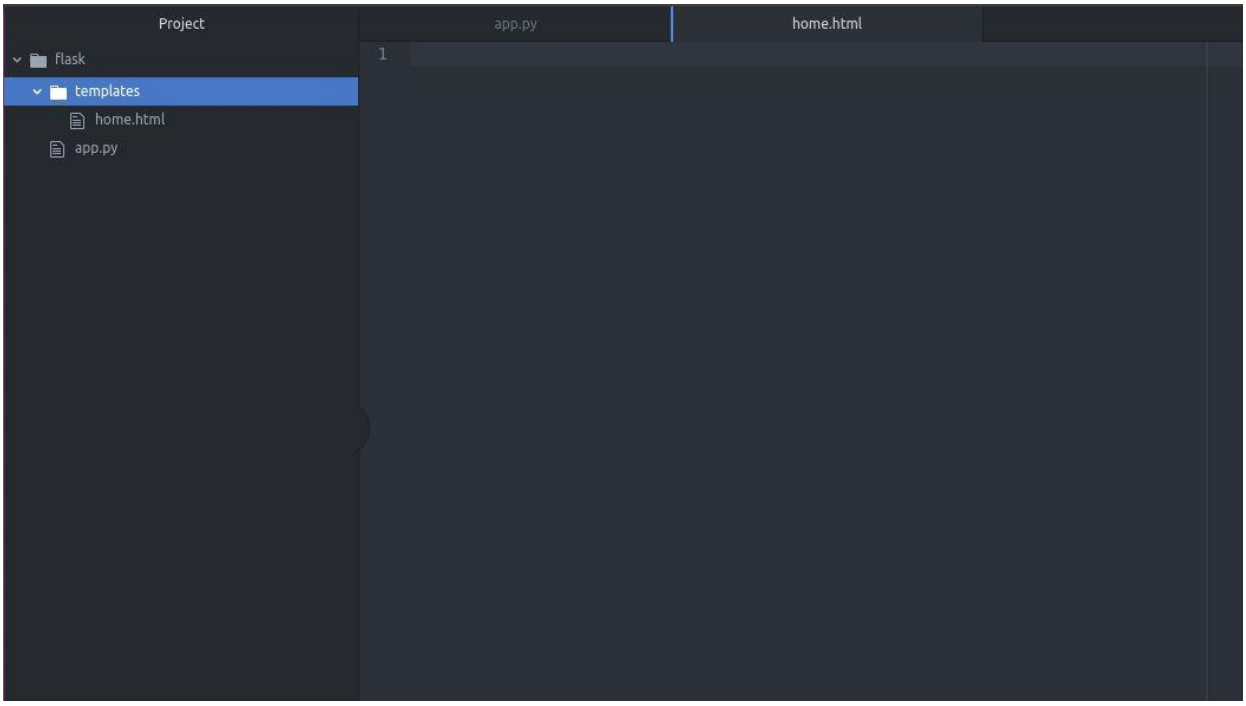
Now we have seen how to create a simple hello world !! program in the flask and how to create a route and navigate the application but basically we are not going to return string like this in the router,we are going to return a template ,so we can use the function called the return template

But that has to been imported first from the flask ,for that we have to add up the following code in the app.py file

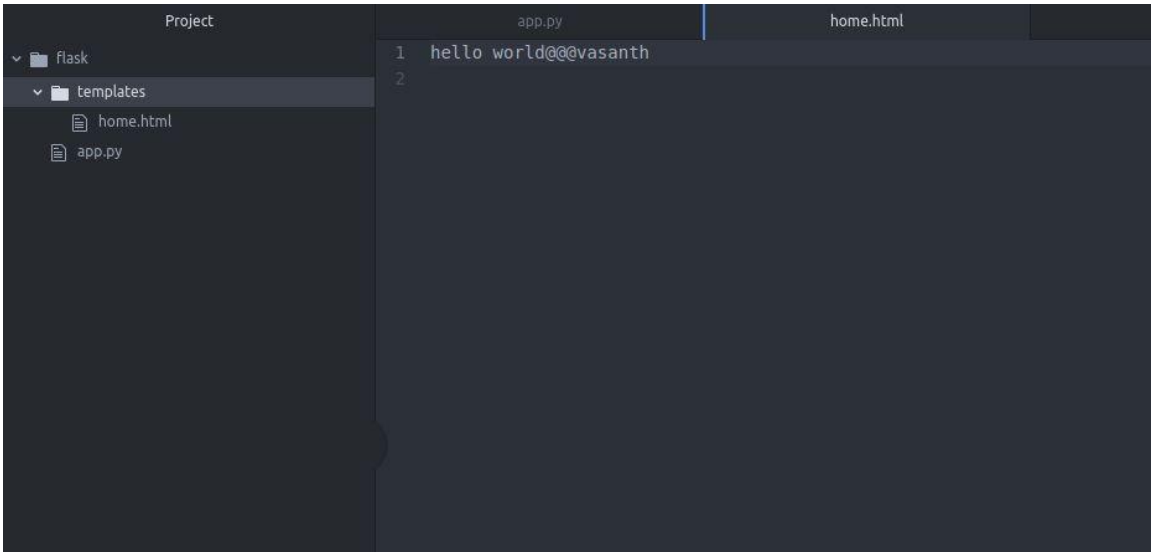
```
from flask import Flask,render_template

app = Flask(_name_)
app.debug = True

@app.route('/')
def index():
        return render_template('home.html')


if __name__=='_main_':
        app.run()
```



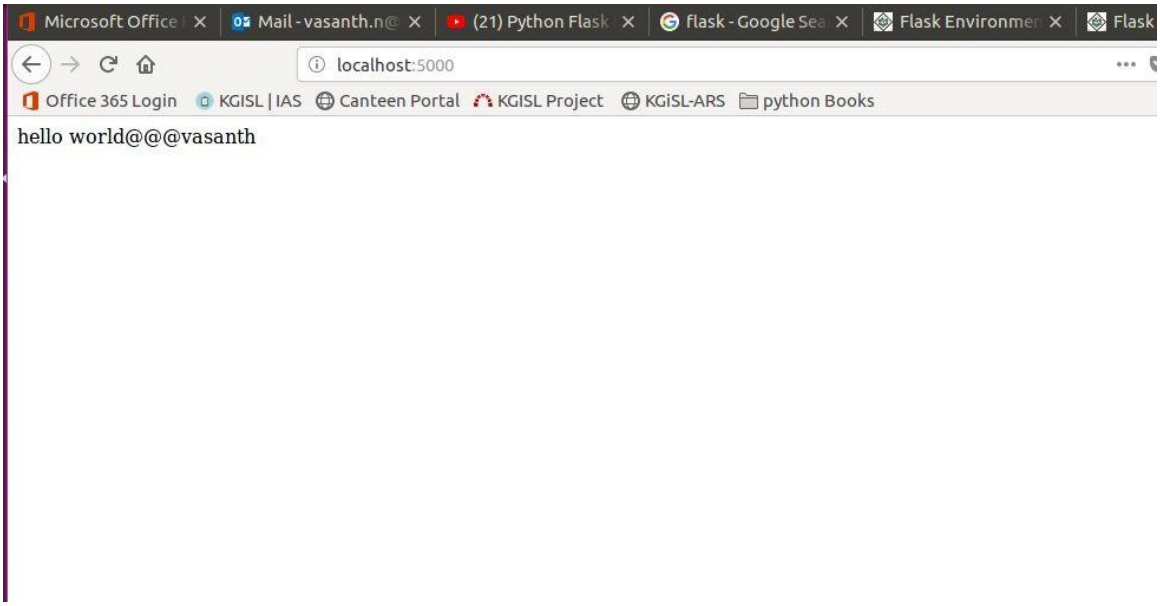
You have noticed that we used home.html in the above index function,so we have to create a template folder inside our flask app folder .



As shown above and inside the templates folder create a file called home.html as shown above. Inside the home.html file type as you like ,now save and refresh the browser the content now will display the new content that u have typed in the home.html

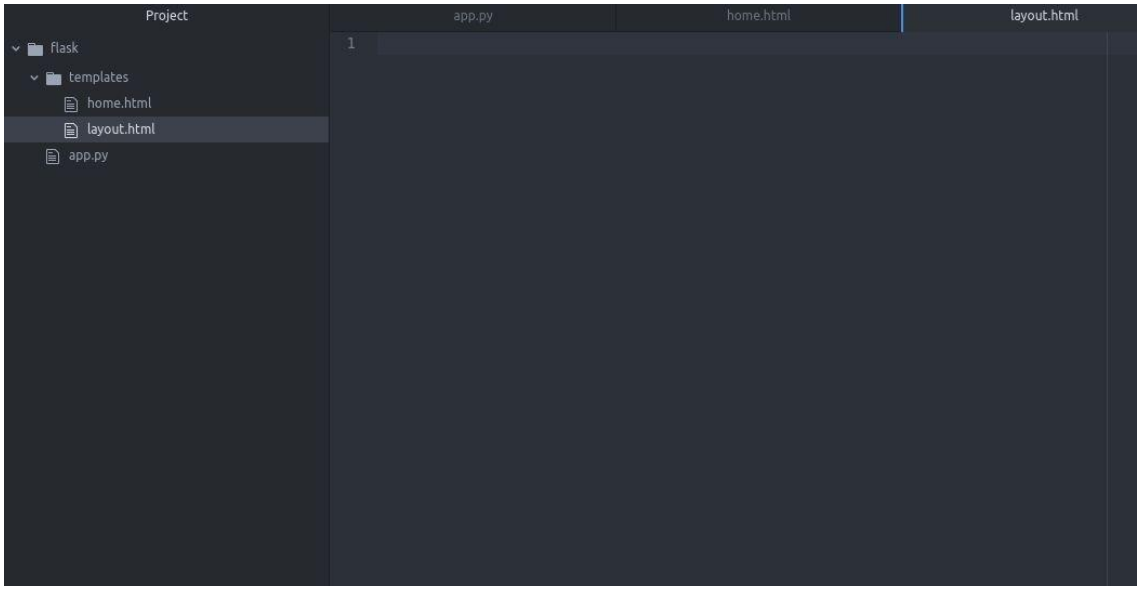


The browser will display the content as :



The next thing is that we are going to create a layout ,this will save a lot of time by repeating the same content in each file ,we no need to create those contents ,layout will take care of those content.

For this we have to create a new file **layout.html** inside the templates folder as shown :



Inside the layout file you can type all the html content ,inside the layout.html file we have some special syntax {% block body %}{% endblock %}

This syntax helps to use the python scripts inside the html file . The {% block body% } is called the start tag and {% endblock %} is called the end tag .the python scripts will go inside between these blocks .

And we can extends the layout.html file in any number of pages this will be explained below .

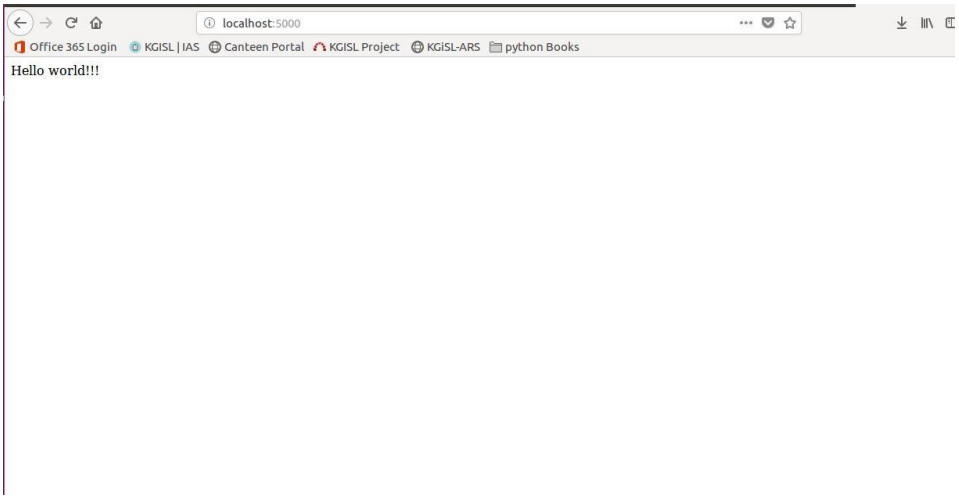
Layout.html

```
<html>
<head>
  <title>Vasanth Flask App</title>
</head>
<body>
{% block body %}{% endblock %}
</body>
</html>
```

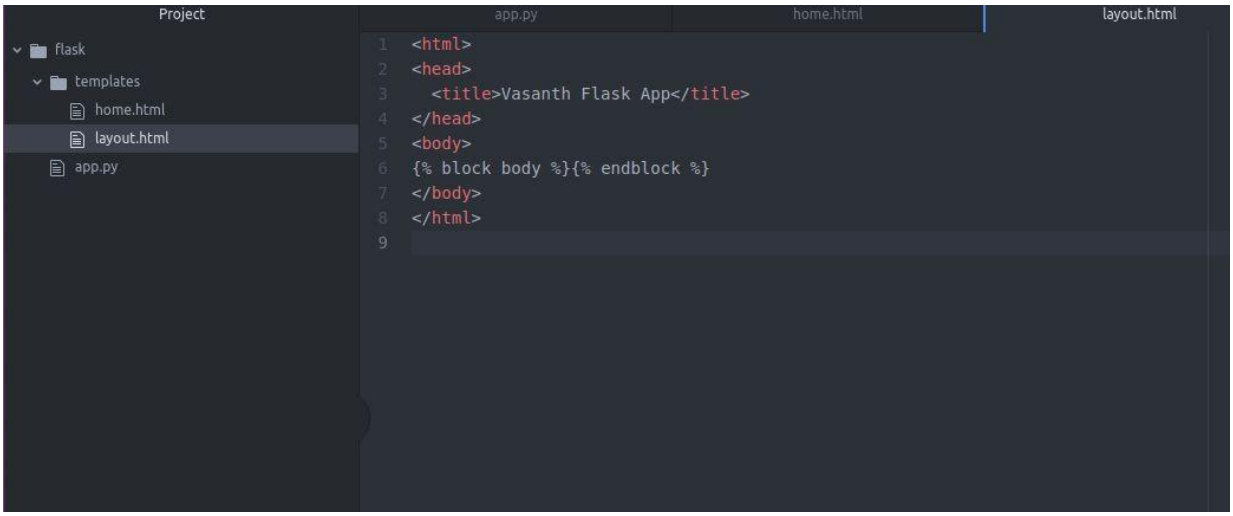
Home.html

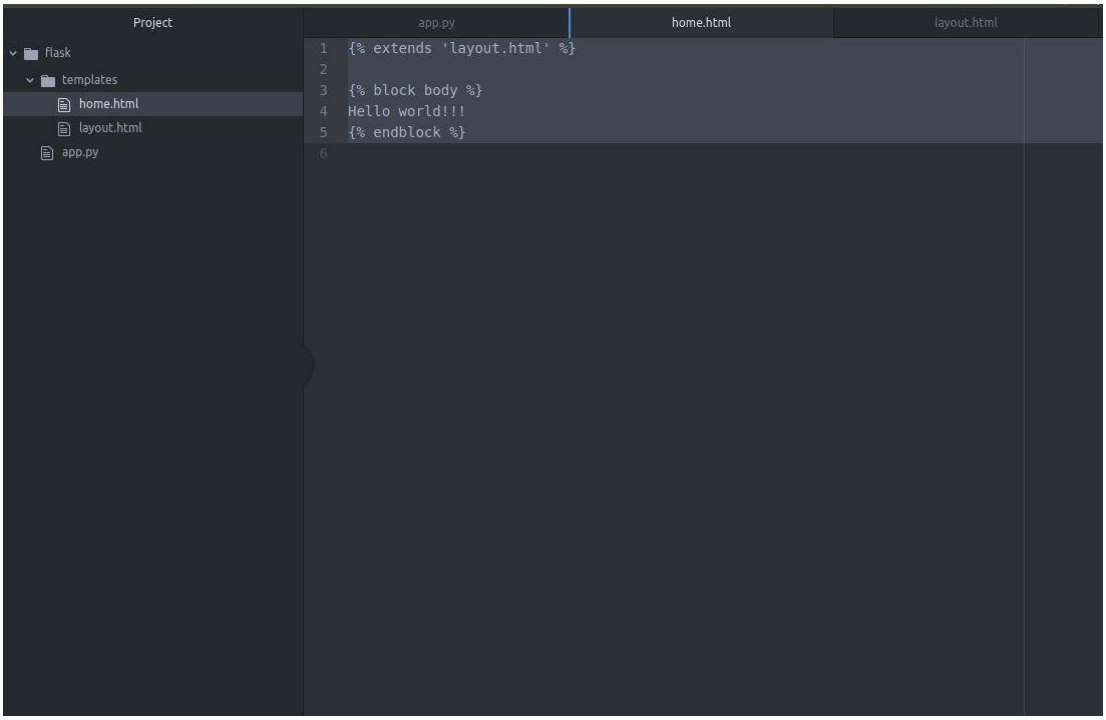
```
{% extends 'layout.html' %}

{% block body %}
Hello world!!!
{% endblock %}
```

Now save and refresh the browser you can see that the hello world!!! In the browser thus this how the templates works.



You can see that the title of the page will be render from the layouts.html and the entire code will look like :





Now we don’t want to repeat the same html content in every page we can create,instead of that we can use the extends template,this save a lots of time in the project development .

BOOTSTRAP:

Initially we are going to use the bootstrap for our application .

For this let's go to :http://blog.getbootstrap.com/2016/07/25/bootstrap-3-3-7-released/

However you can find the latest bootstrap am using the 3.3.7 version ,if you are familiar with bootstrap you can use any version as you like ,but in this tutorial am using the 3.3.7 version .

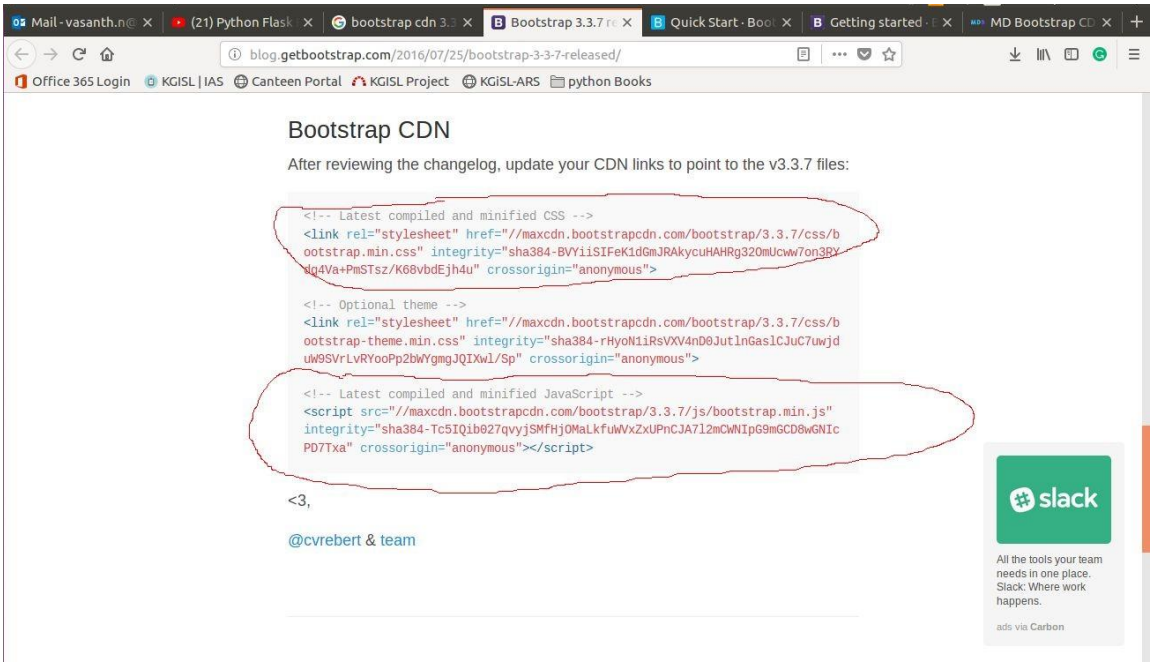
Now copy the following code as shown :

CSS:

```
<link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
```

JS :

```
<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"</script>
```



Now copy and paste the css file and js file inside our layout.html file

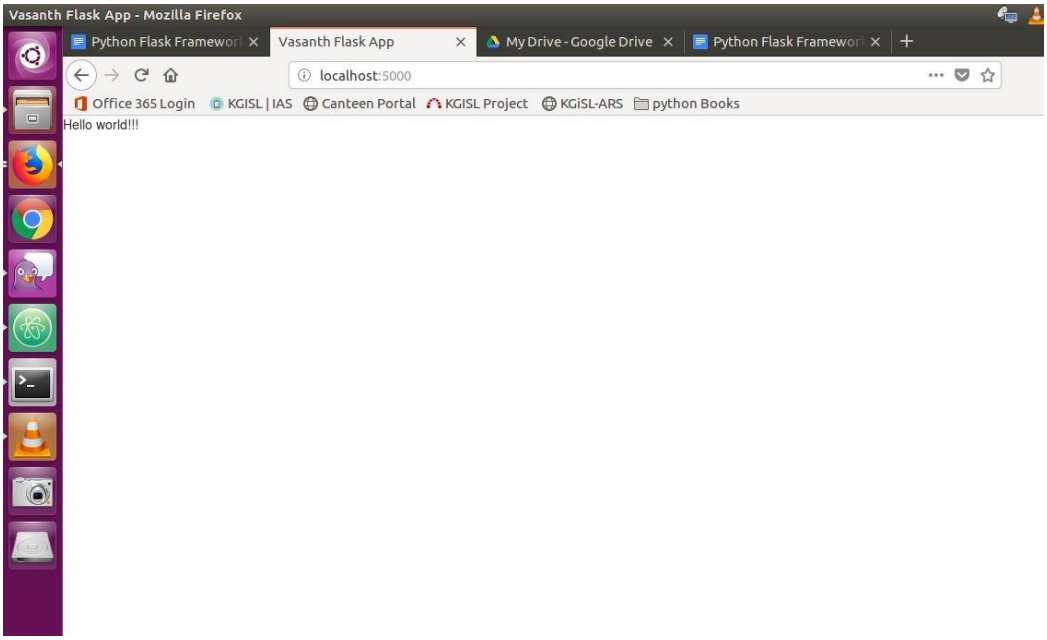
We have to paste the css file inside the head tag and the JS file in the end of the body tag as shown .

layout.html

```
<html>
<head>
  <title>Vasanth Flask App</title>
  <link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
{% block body %}{% endblock %}


<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
```

```
</body>
</html>
```

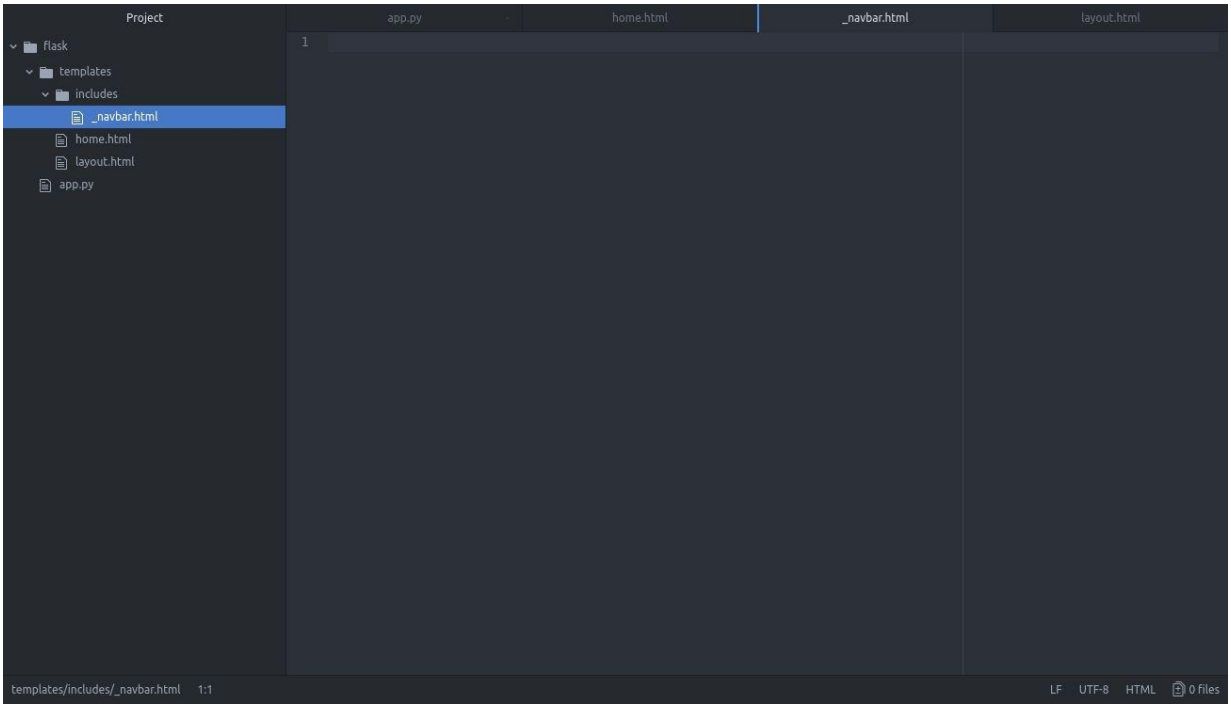
Now save and refresh the browser you can notice the bootstrap is enabled as you can see the change in fontsize.



Now we have to create a new folder called **includes** inside the template folder :

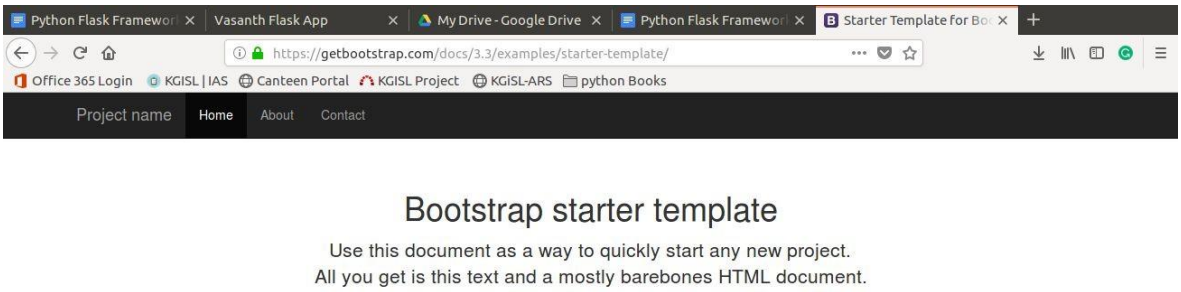
Create a new file inside the includes folder , the file name should be this

_navbar.html

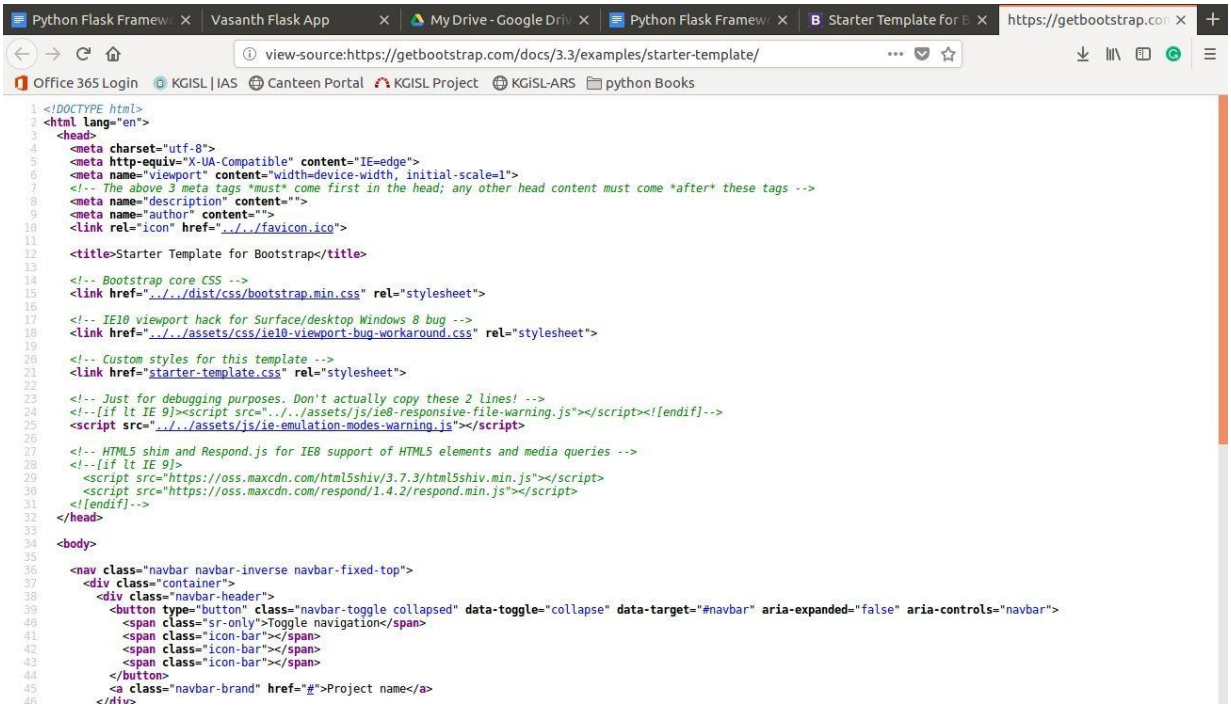


We have used _ because this file is partial file since we are going to use this _navbar.html only for creating the navbars .

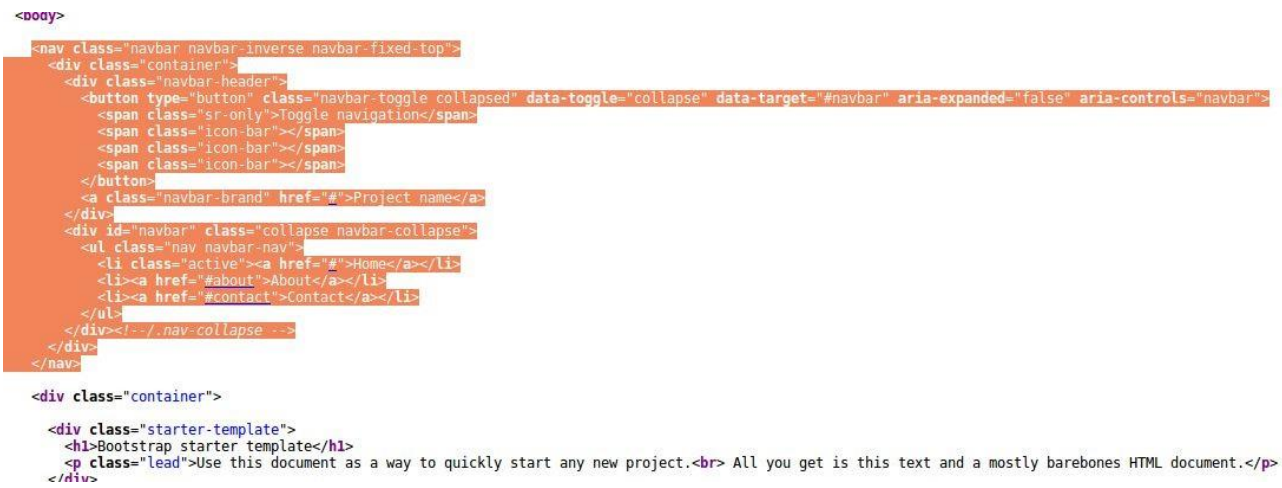
Now go to the following link https://getbootstrap.com/docs/3.3/examples/starter-template/



Now press **ctrl + u** in the keyboard now you will see the page that displayed as shown below :



In that contents copy the navbar contents as shown here :



The copied code should look like this :

```
<nav class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
      <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
data-target="#navbar" aria-expanded="false" aria-controls="navbar">
```

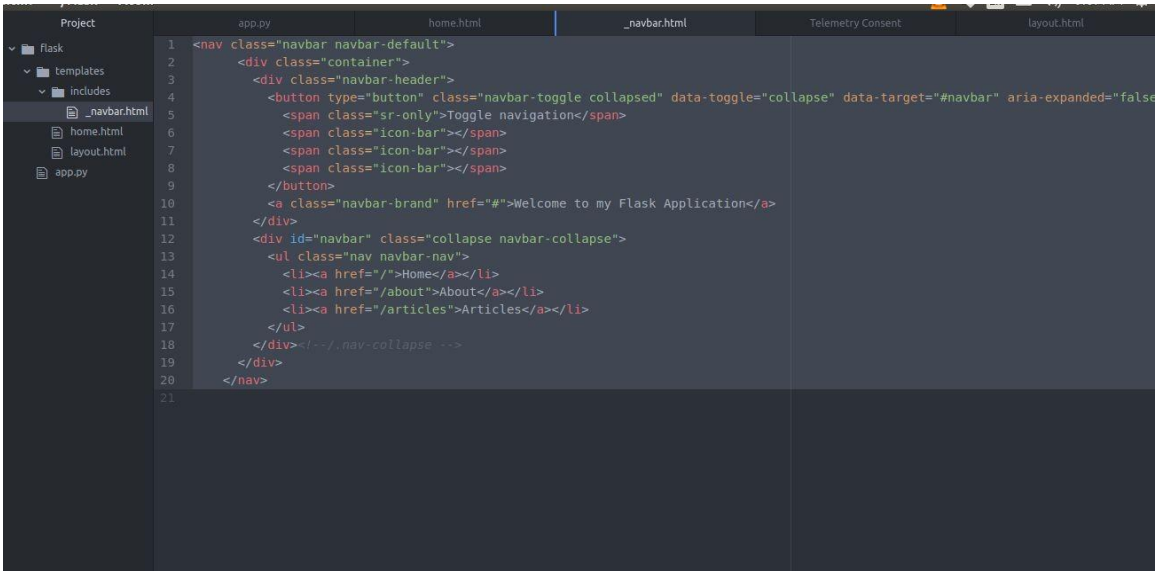
```
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Project name</a>
    </div>
    <div id="navbar" class="collapse navbar-collapse">
      <ul class="nav navbar-nav">
        <li class="active"><a href="#">Home</a></li>
        <li><a href="#about">About</a></li>
        <li><a href="#contact">Contact</a></li>
      </ul>
    </div><!--/.nav-collapse -->
  </div>
</nav>
```

Now we have to make some changes in the above code let's follow as i explained below :

```
<nav class="navbar navbar-default">
        <div class="container">
        <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
data-target="#navbar" aria-expanded="false" aria-controls="navbar">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        </button>
        <a class="navbar-brand" href="#">Welcome to my Flask Application</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
        <li><a href="/">Home</a></li>
        <li><a href="/about">About</a></li>
        <li><a href="/articles">Articles</a></li>
        </ul>
        </div><!--/.nav-collapse -->
```

```
</div>
</nav>
```

The above Highlighted area are the code has been modified , now your code should look like this



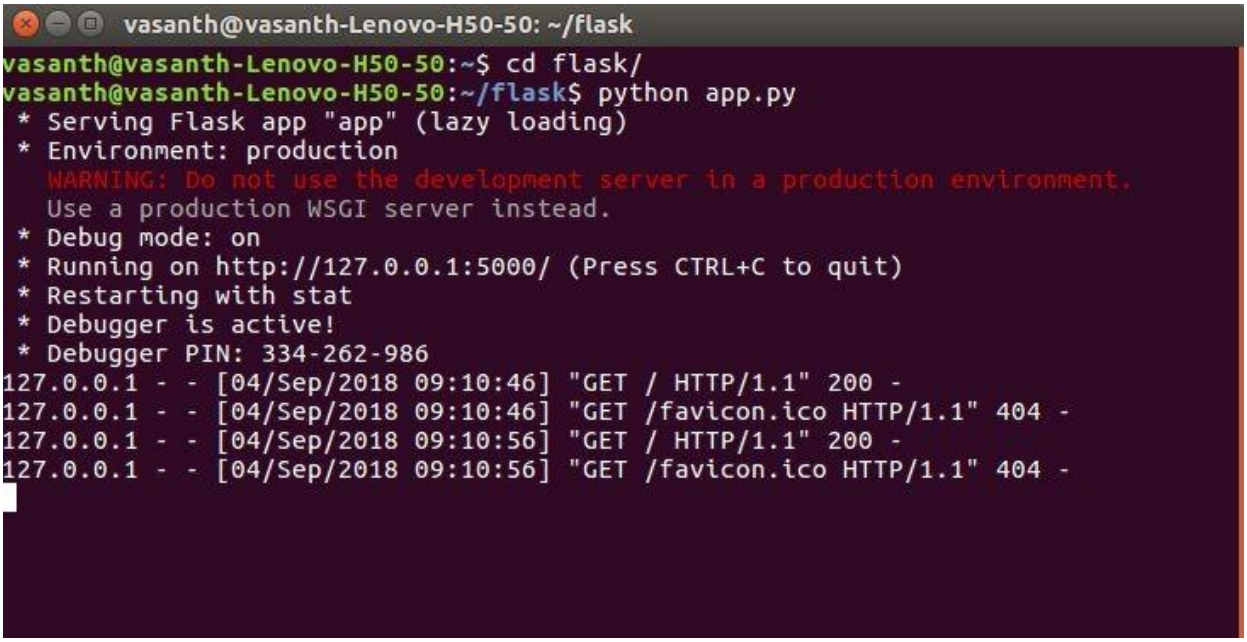
Now we have to include this in our layout files ,for this just go to **layout.html** file as shown below :

layout.html

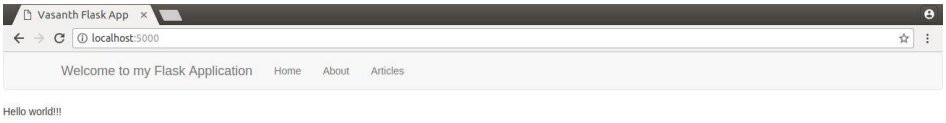
```
<html>
<head>
  <title>Vasanth Flask App</title>
  <link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
{% include 'includes/_navbar.html'%}


{% block body %}{% endblock %}
```

```
<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</body>
</html>
```

In the above code we have add the highlighted line ,this line is to include or includes folder where our **_navbar.html** file exist ,now save the file and run the flask server by running the **app.py** file



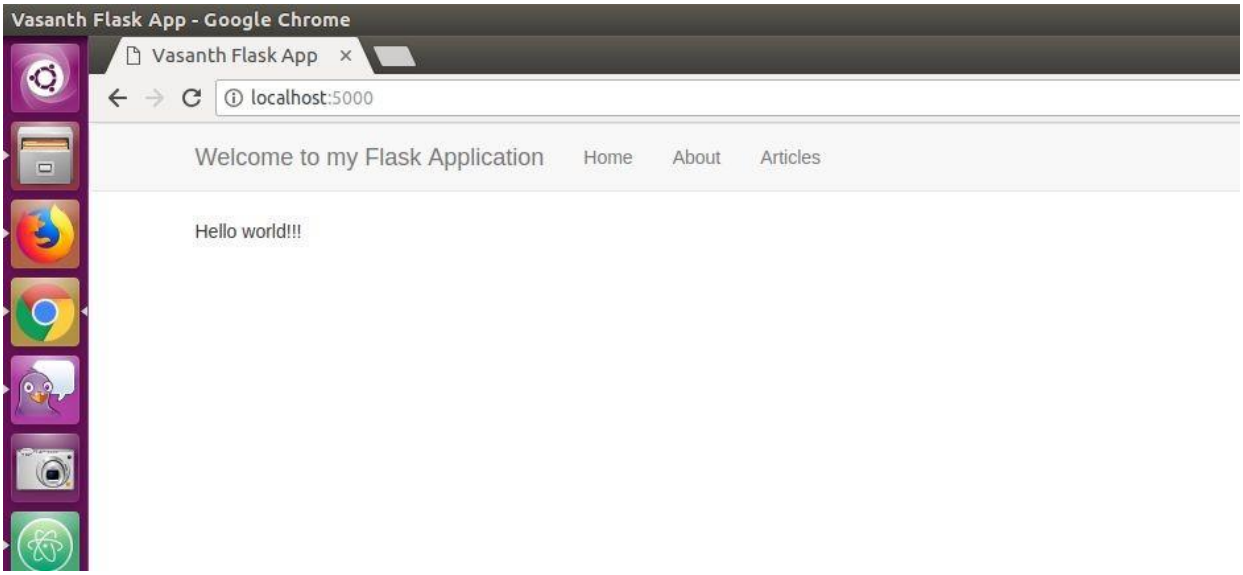
Now open the browser and type the address **localhost:5000 ,**now you can see the bootstrap is working fine as shown below :



Now i want to move all the content inside the website into a container class ,for that add the following line in the **layout.html file**

```
<html>
<head>
  <title>Vasanth Flask App</title>
  <link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
{% include 'includes/_navbar.html'%}

<div class="container">
  {% block body %}{% endblock %}
</div>


<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</body>
</html>
```

Now add those lines in the **layout.html** and save the file , now refresh the browser and see the world hello will be moved a bit towards the right,you can see the changes after refreshing the browser.

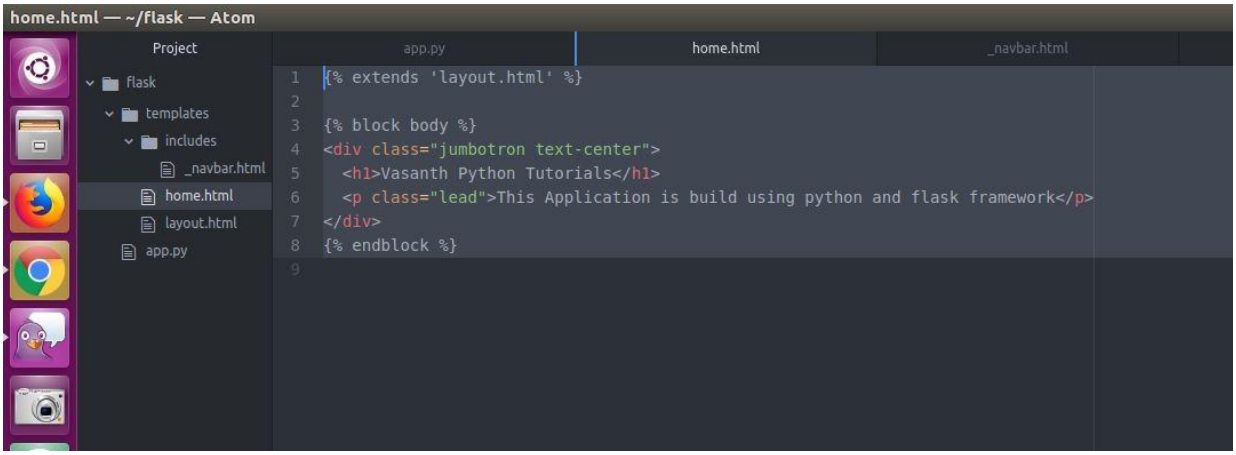


Now lets add some content in the home.html file ,now open the home.html file and add the following inside that as shown below .

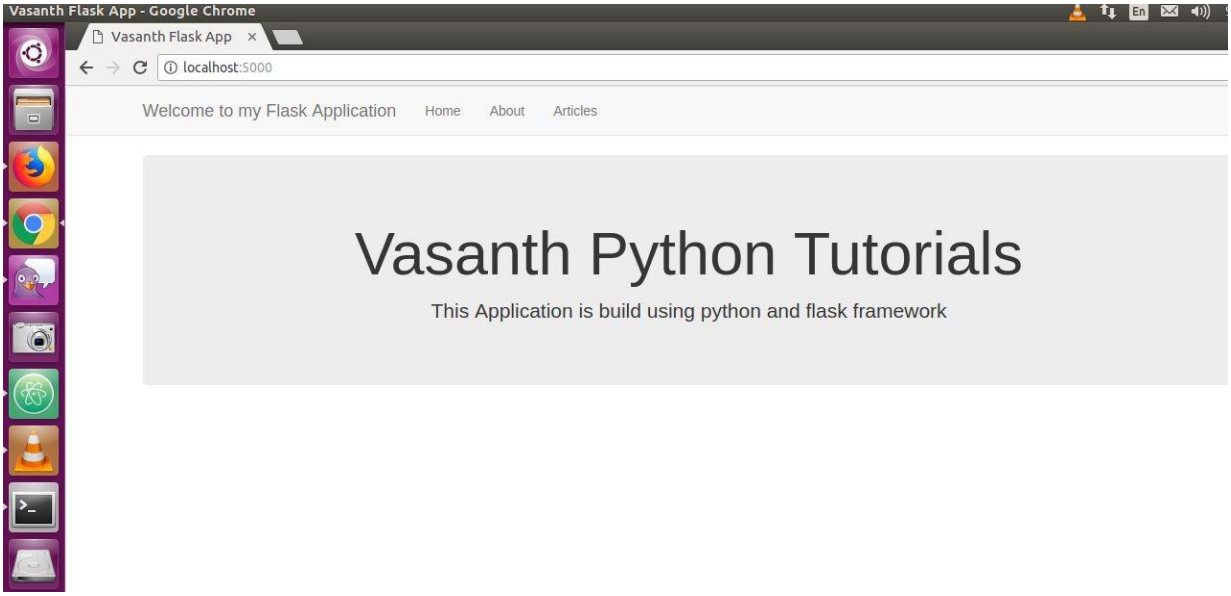
Home.html

```
{% extends 'layout.html' %}

{% block body %}
<div class="jumbotron text-center">
  <h1>Vasanth Python Tutorials</h1>
  <p class="lead">This Application is build using python and flask framework</p>
</div>
{% endblock %}
```

Your code should look like this :



Now save the file home.html and refresh the browser you will see the changes as shown :

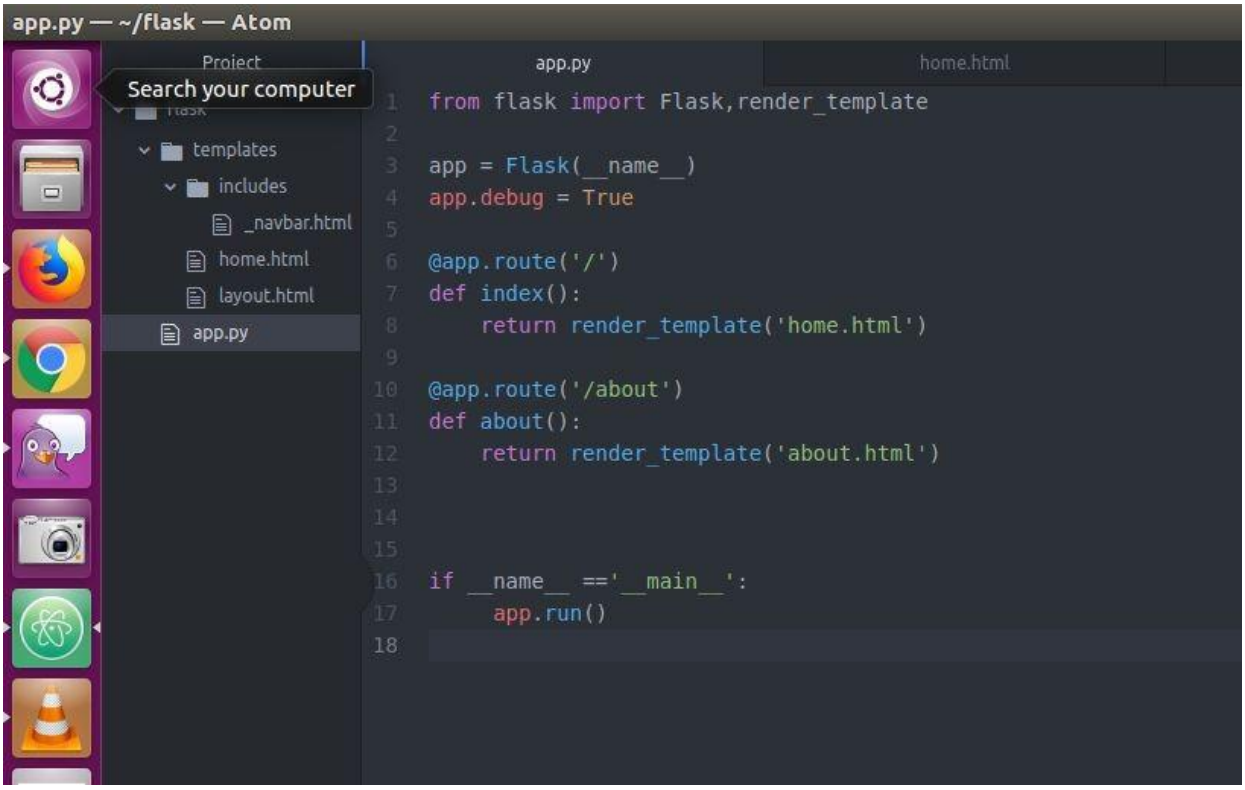


Now we have to create a navigation for the articles and about page for this ,add up the following code in the **app.py** file as shown below :

```
from flask import Flask,render_template

app = Flask(_name_)
app.debug = True

@app.route('/')
```

```
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')


if __name__=='_main_':
        app.run()
```

Your code will look like this :

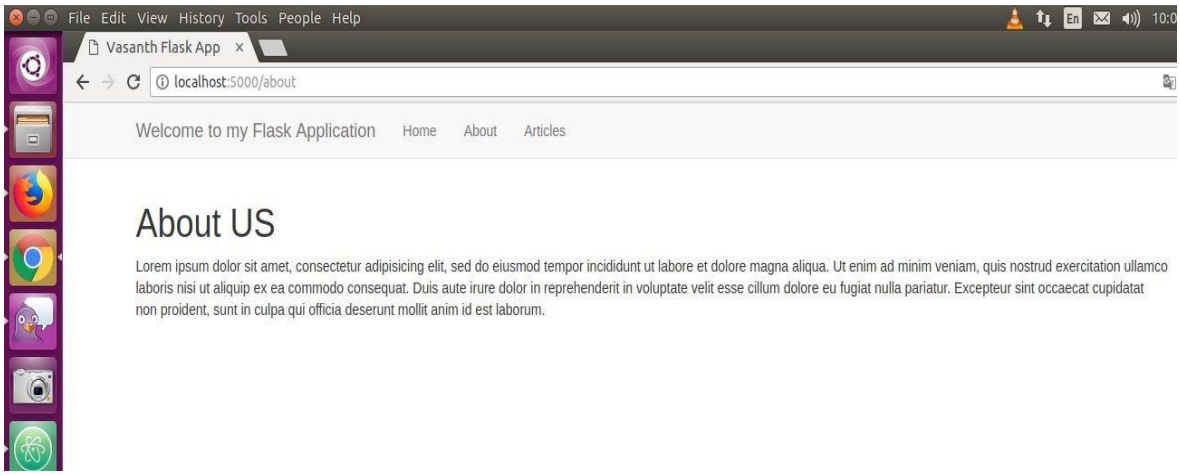


Now create a about.html in the templates folder as shown:

Now copy the code that you have it in the home.html file and make the following changes :

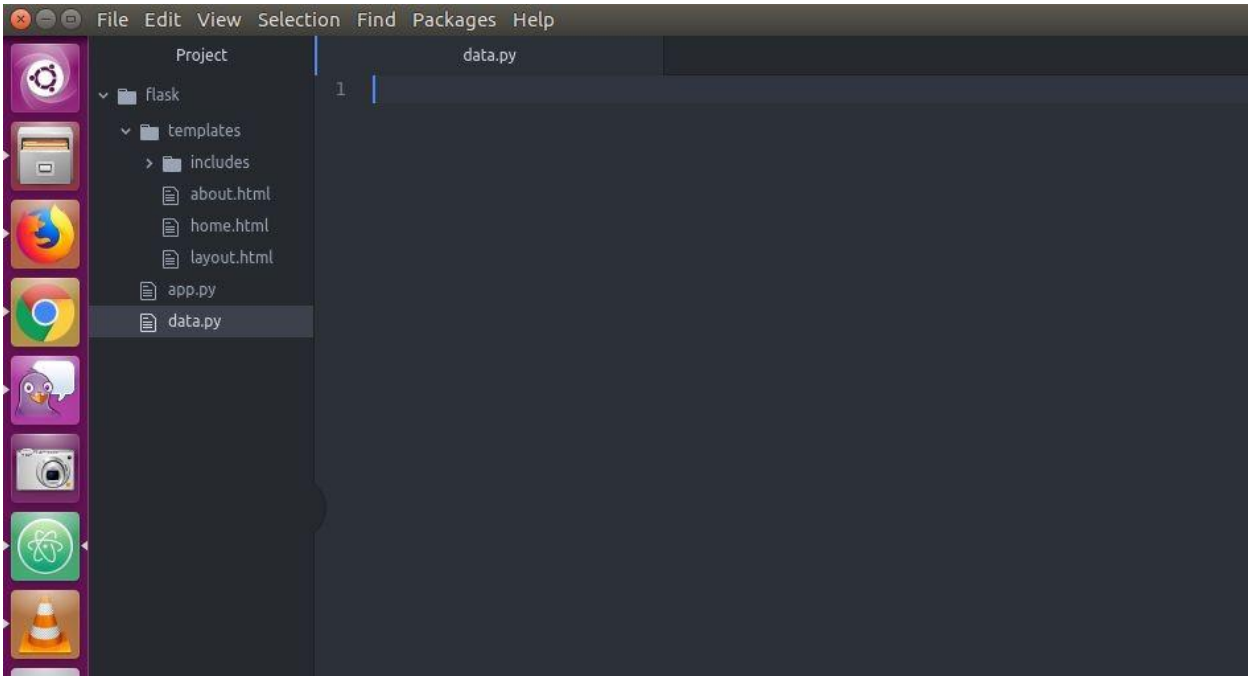
```
{% extends 'layout.html' %}

{% block body %}
<h1>About US </h1>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
{% endblock %}
```

Here i have added the dummy text for the paragraph ,now save the about.html file and refresh the browser and click about in the home page ,now your about page will look like this :



Actually we need the login and register button for our page but as of now we are not going to create that,and we are not going to use any database right now ,but we will create it later ,what i gonna do is that we are going to create a file for articles ,that will hold the data for the articles .

Now we wanna create a new file in the root directory as shown below and save the file as **data.py**



Now add the following code in the **data.py**Now we are going to define a function called articles as shown below :add the code in the **data.py**

```
def Articles():
        articles = [
        {
        'id': 1,
        'title':'Article one',
        'body':'Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.',
        'author':'vasanth',
        'create_date':'04-09-2018',
        },
        {
        'id': 2,
        'title':'Article two',
        'body':'Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit
```

in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.',

'author':'vasanth nagarajan',

'create_date':'05-09-2018',

},

{

'id': 3,

'title':'Article three',

'body':'Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.',

'author':'nagarajan vasanth',

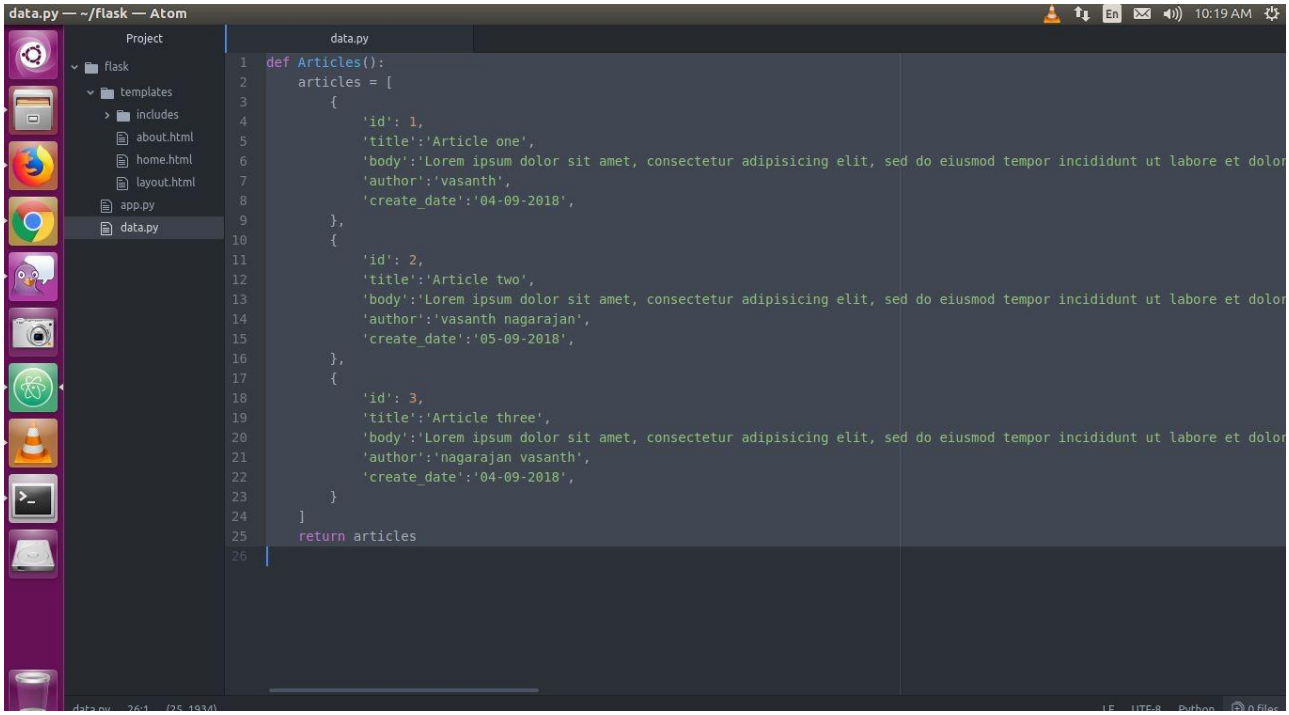
'create_date':'04-09-2018',

}

]

return articles

Your code should look like this :



Now go to the app.py file and import the data articles to our file ,this is done using the following code :

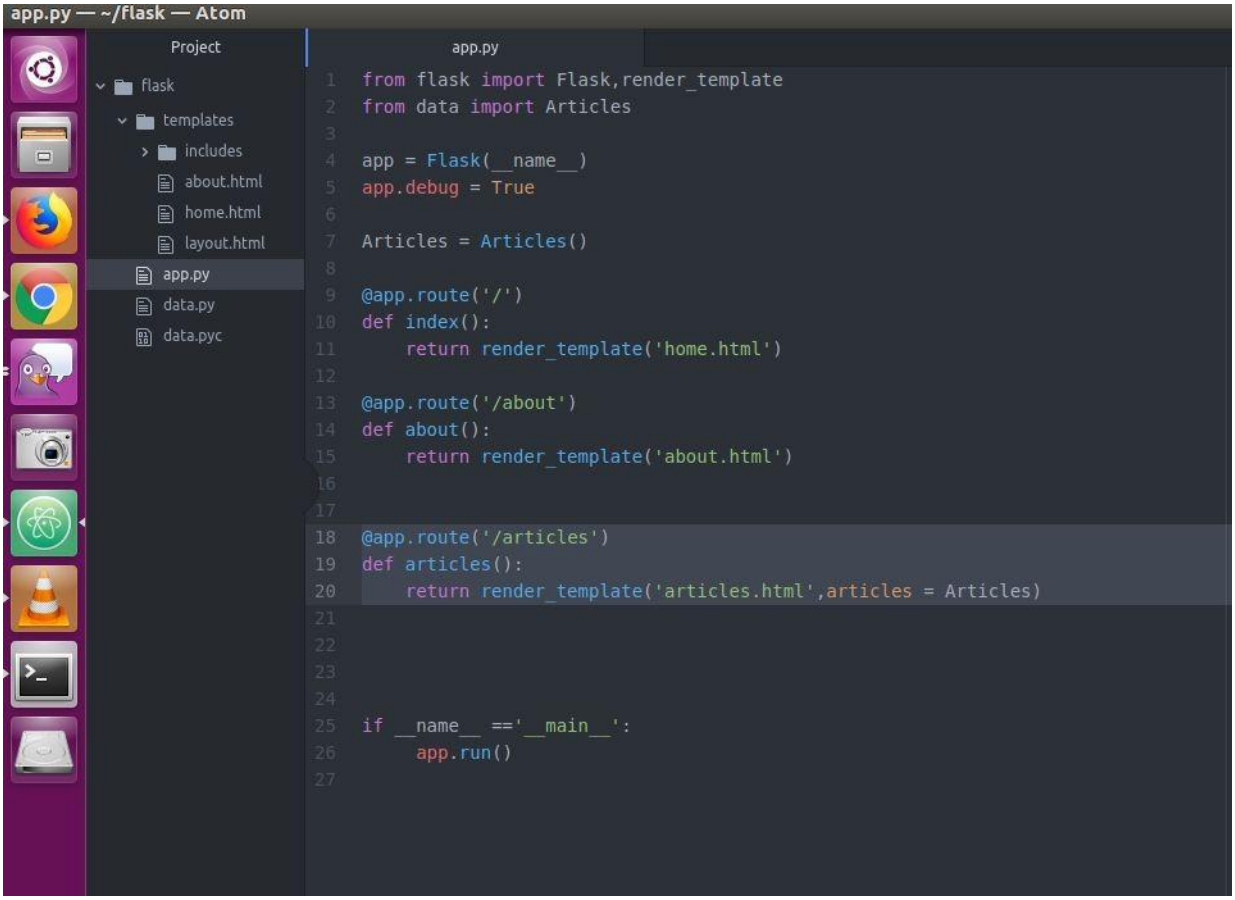
app.py

```
from flask import Flask,render_template
from data import Articles

app = Flask(_name_)
app.debug = True

Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')



if __name__==' main ':
        app.run()
```

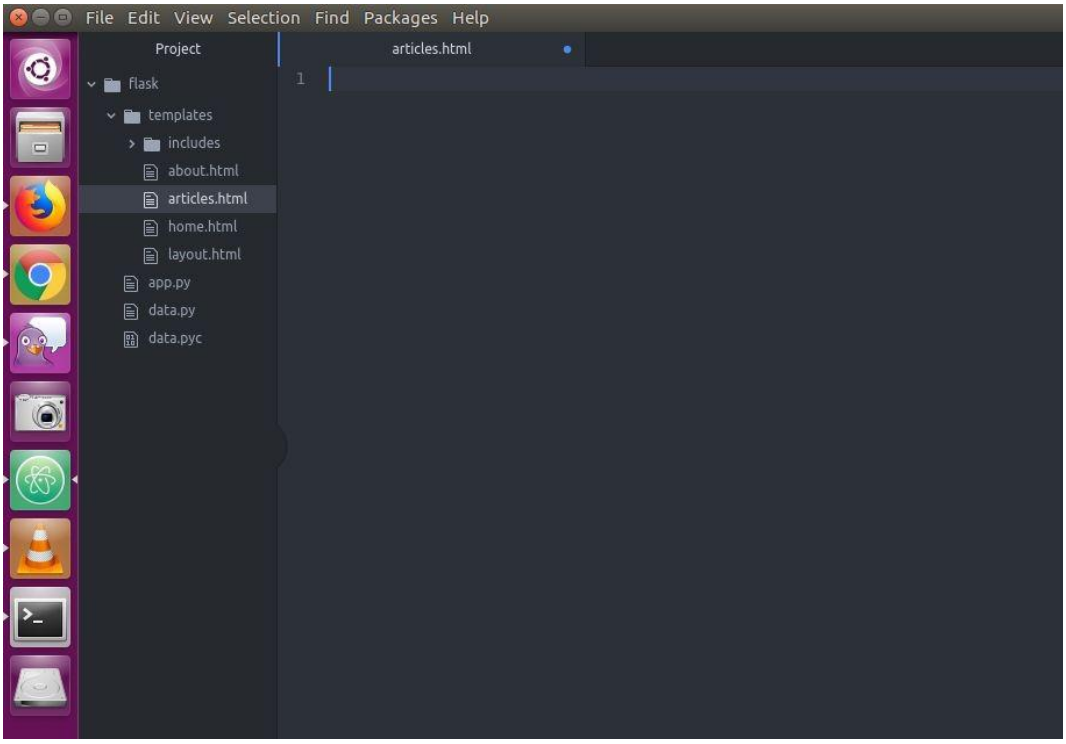
Now we have to create a route for the articles now add the following code in the app.py file :

```
@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)
```

Now your code look like this :



Now create a articles.html file in the template folder and write the following code and save it.



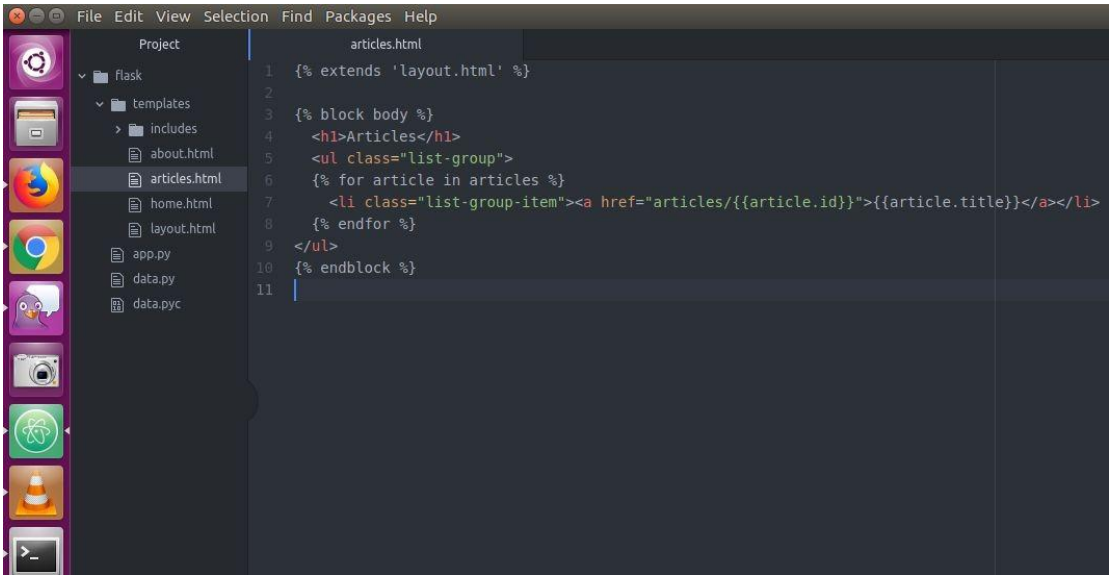
Articles.html

Now in this file we are going to use the for loop logic in this **articles.html** file :

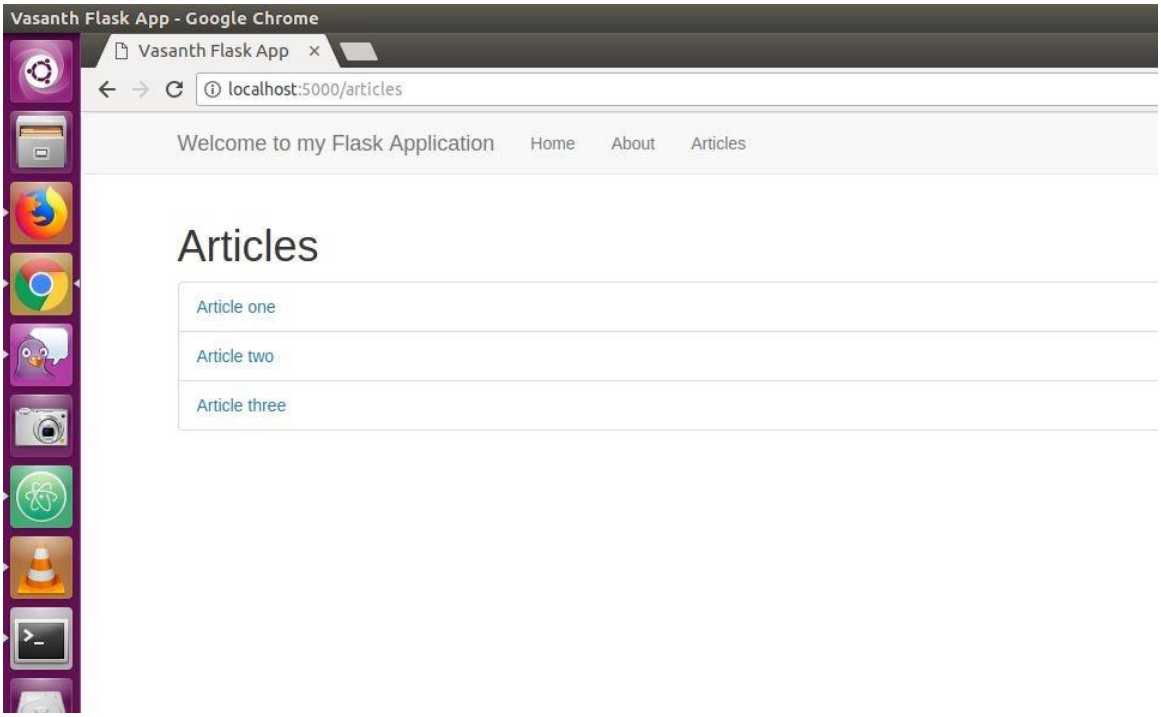
Articles.html

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Articles</h1>
  <ul class="list-group">
  {% for article in articles %}
        <li class="list-group-item"><a href="articles/{{article.id}}">{{article.title}}</a></li>
  {% endfor %}
</ul>
{% endblock %}
```

Your code will look like this :



Now save this file and refresh the browser and click the article option in th home page now the browser will display the article page as shown below :



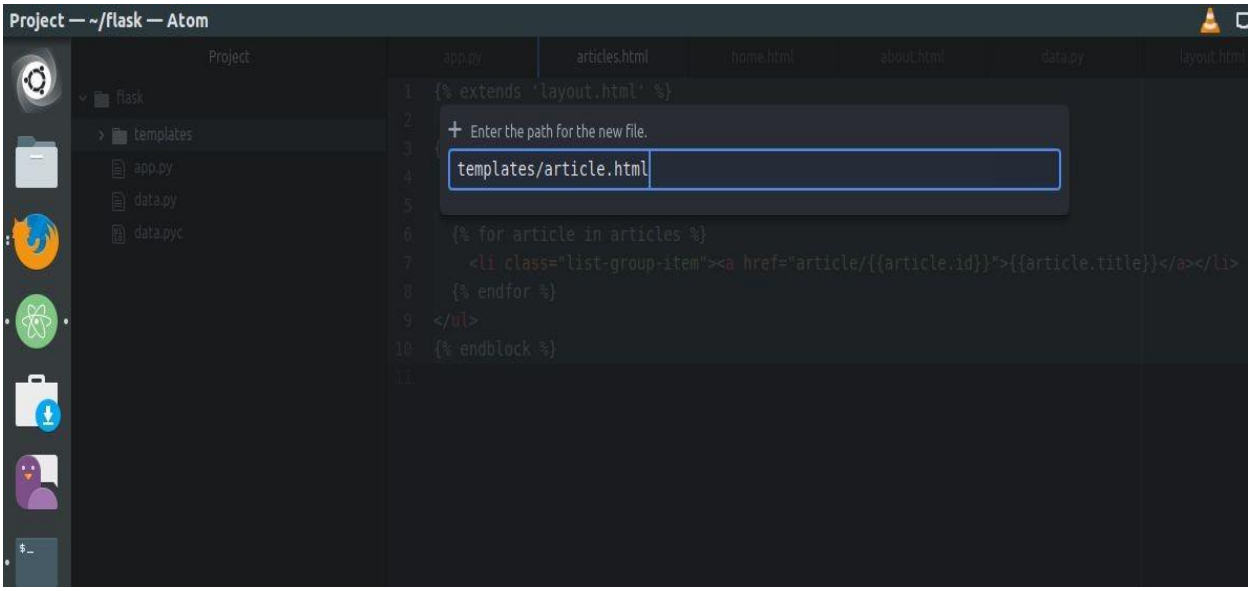
Now we wanna create a links for each article for this we have to create a new route for the articles ,now open app.py file add the following routes in the file .

```
from flask import Flask,render_template
from data import Articles

app = Flask(_name_)
app.debug = True

Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)




if __name__=='_main_':
        app.run()
```

NOTE : Now **Articles.html** change the following code :
Your **articles.html** file should look like this :

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Articles</h1>
  <ul class="list-group">
  {% for article in articles %}
        <li class="list-group-item"><a href="article/{{article.id}}">{{article.title}}</a></li>
  {% endfor %}
</ul>
{% endblock %}
```

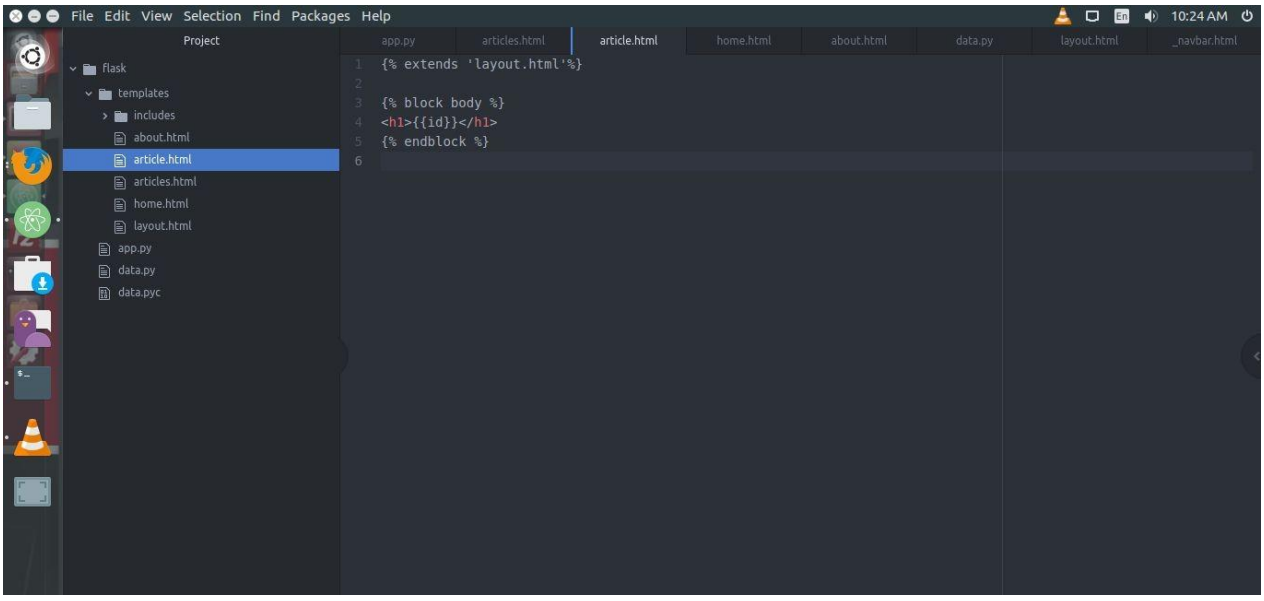
Now Create a **article.html** file in the template folder as shown:



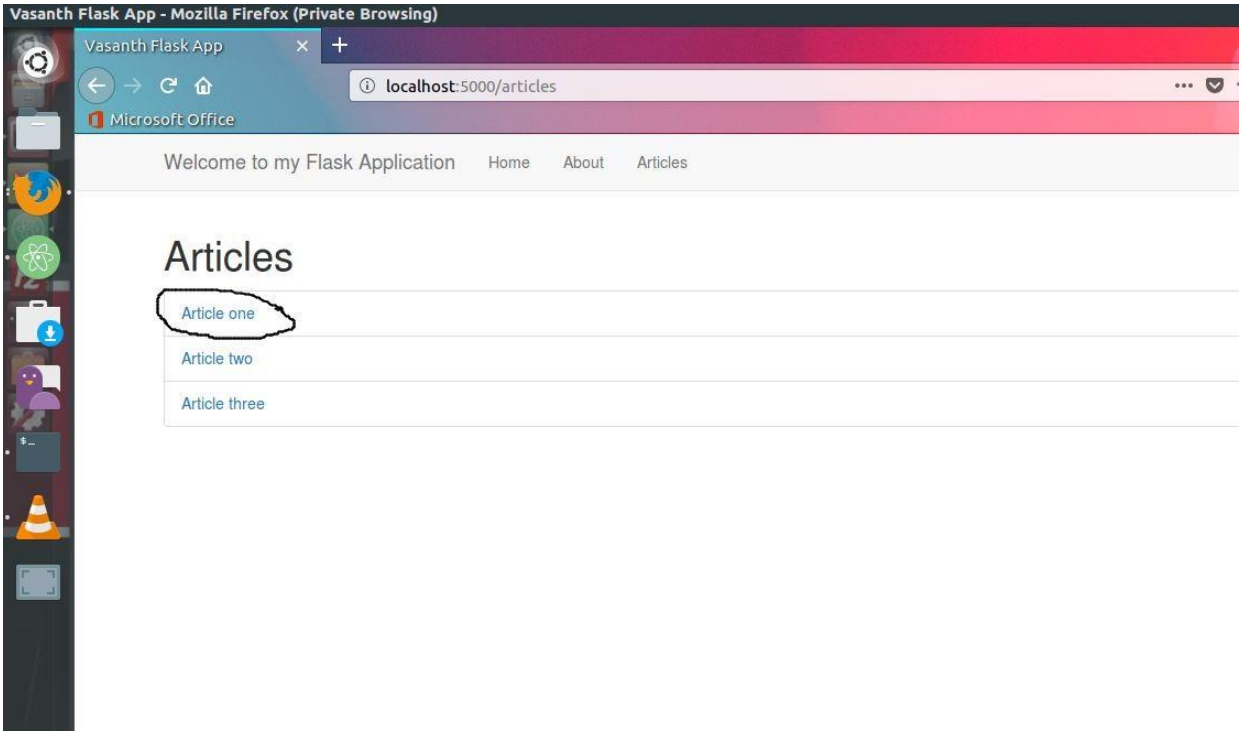
Now open the **article.html** file and add the following code in the file as shown below :

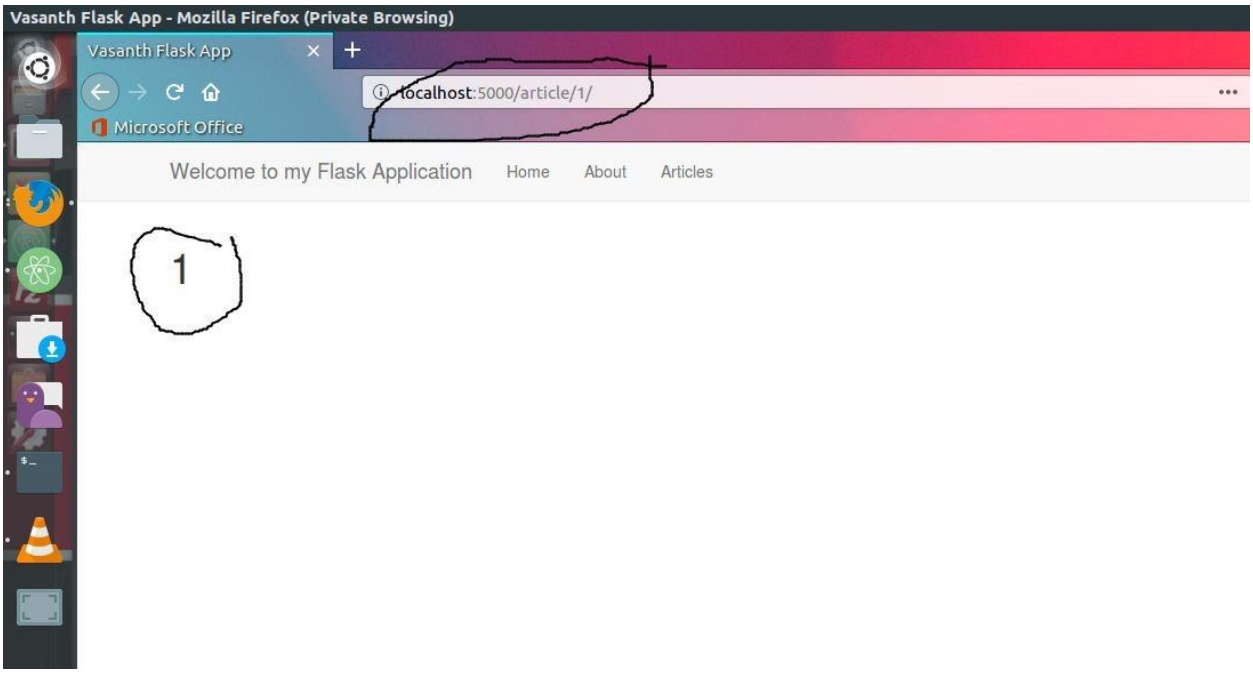
```
{% extends 'layout.html'%}

{% block body %}
<h1>{{id}}</h1>
{% endblock %}
```

Your article code should look like this :

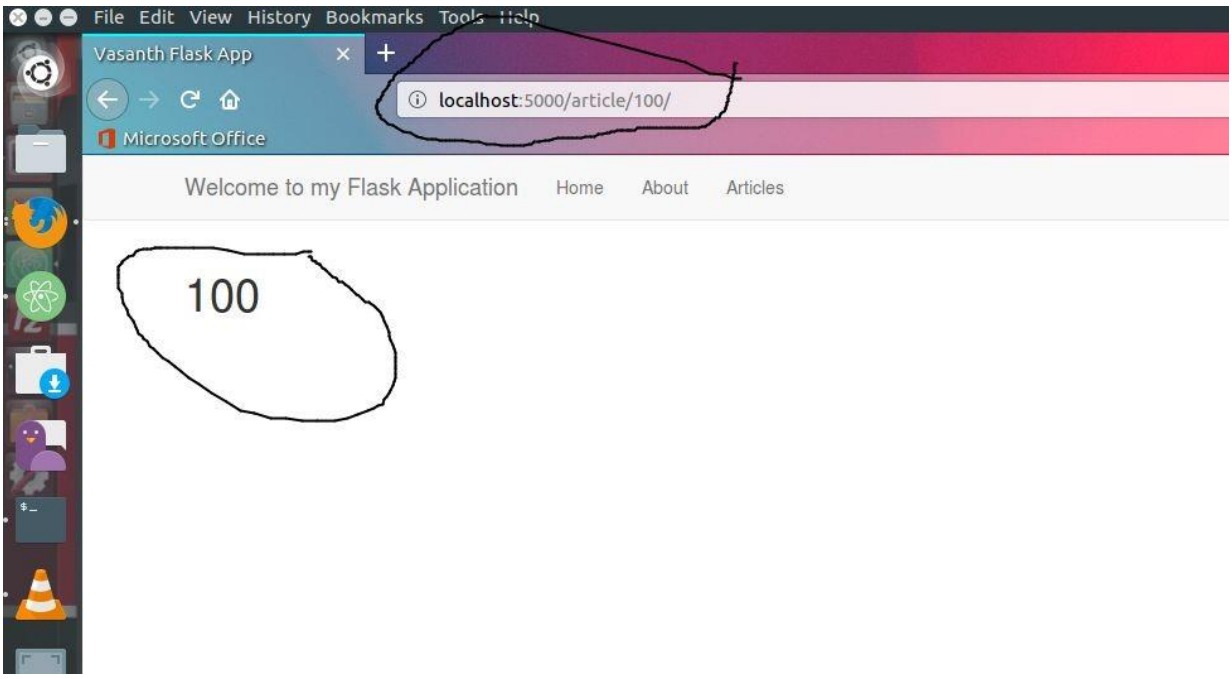


Now save the file and start the python flask server and look at http://localhost:5000 in your browser .





Now Type any number in the address bar it will display the number as id :



Setting Up With Database

Now we are going to install the mysql server and the client for our application for this installation ,follow the steps that given below ,

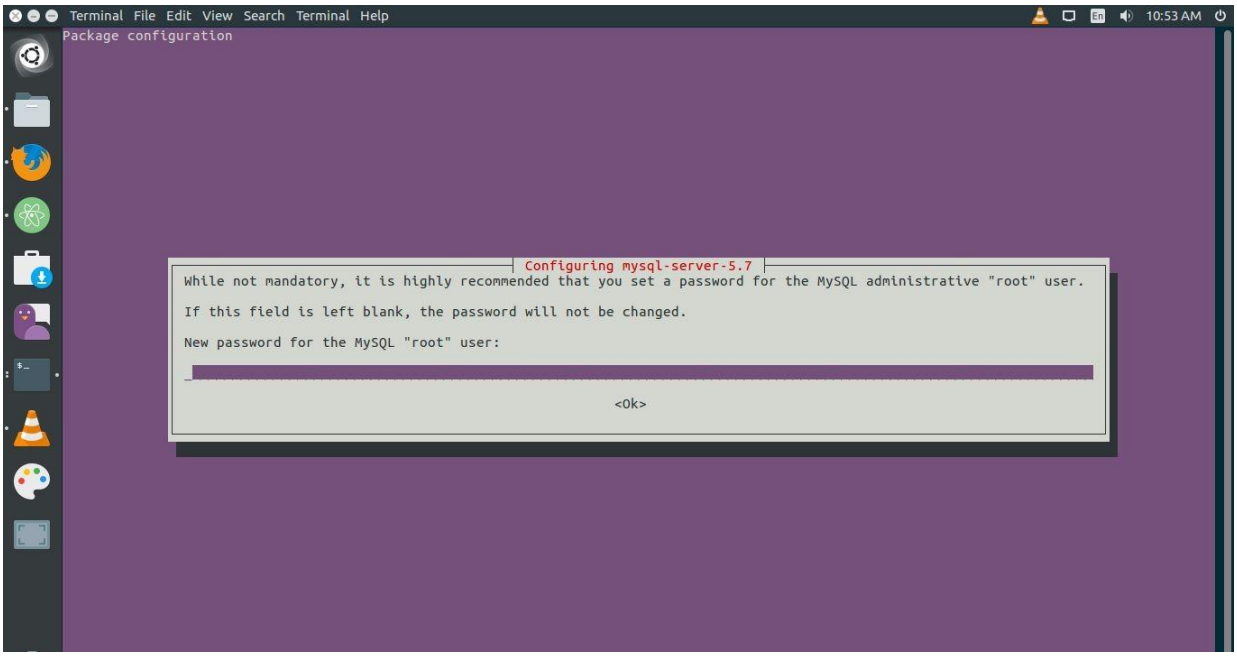
NOTE: MYSQL SERVER and MYSQL CLIENT are need to be installed

vasanth@vasanth-Lenovo-H50-50:~/flask$ sudo apt-get install mysql-server libmysqlclient-dev

Type the following command in the terminal , you have to type from the **sudo apt - get** command

.

While installing the MYSQL server , it will ask you to enter the root password .



For this enter **root** as password : **root**

Both the user name and the password will be the same.

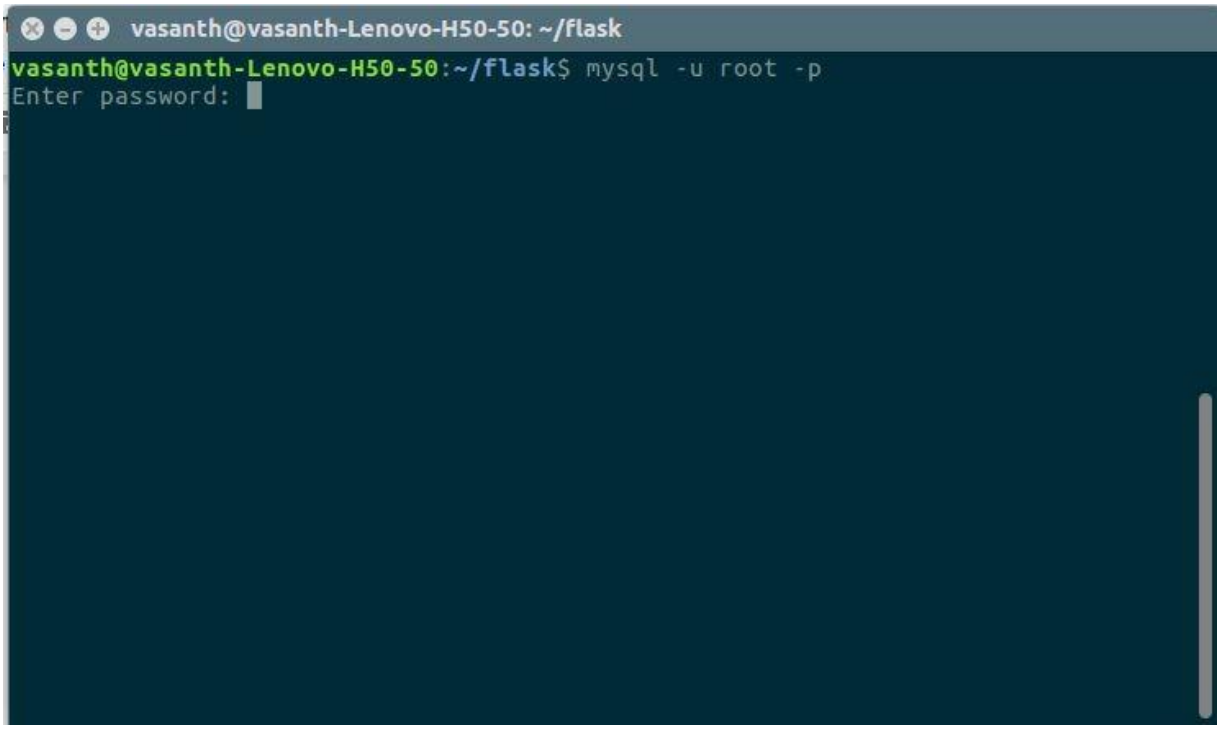
After Successfully Installed the Both the Mysql Server and client proceed as follows

Accessing the Database :

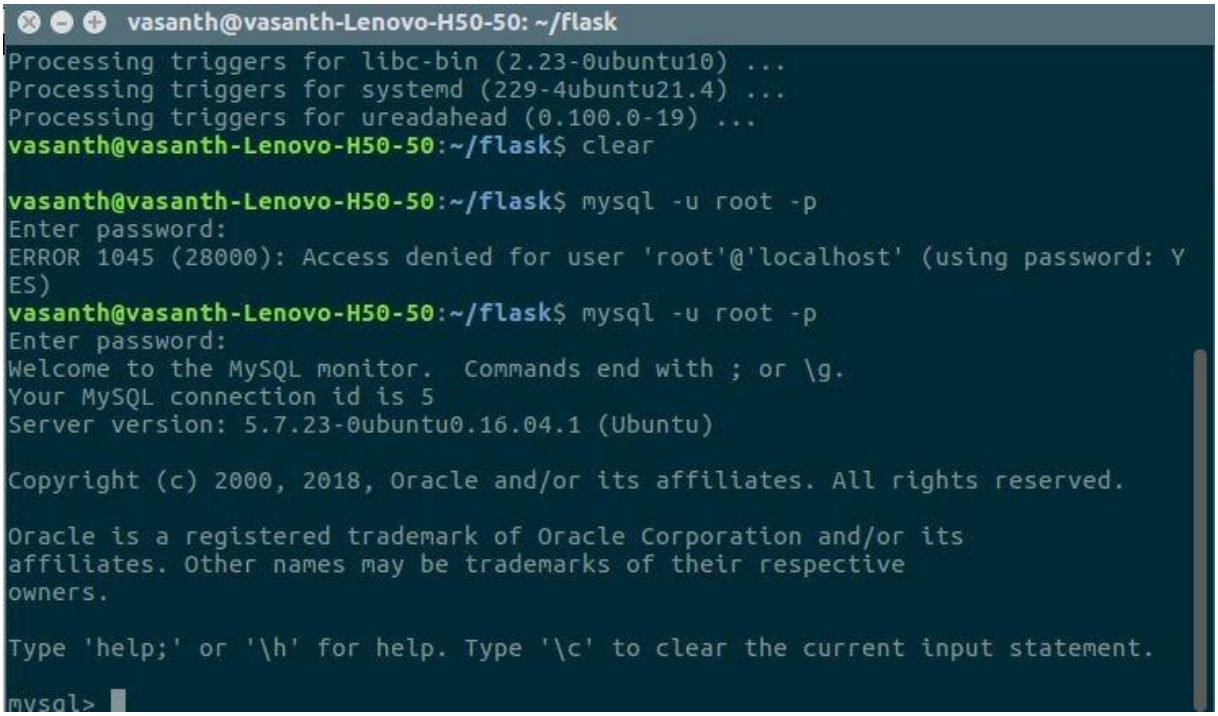
Now open the terminal and type the following code :

vasanth@vasanth-Lenovo-H50-50:~/flask$ mysql -u root -p

Now you will be asked to enter the password as shown below : type **root** as in the password file .



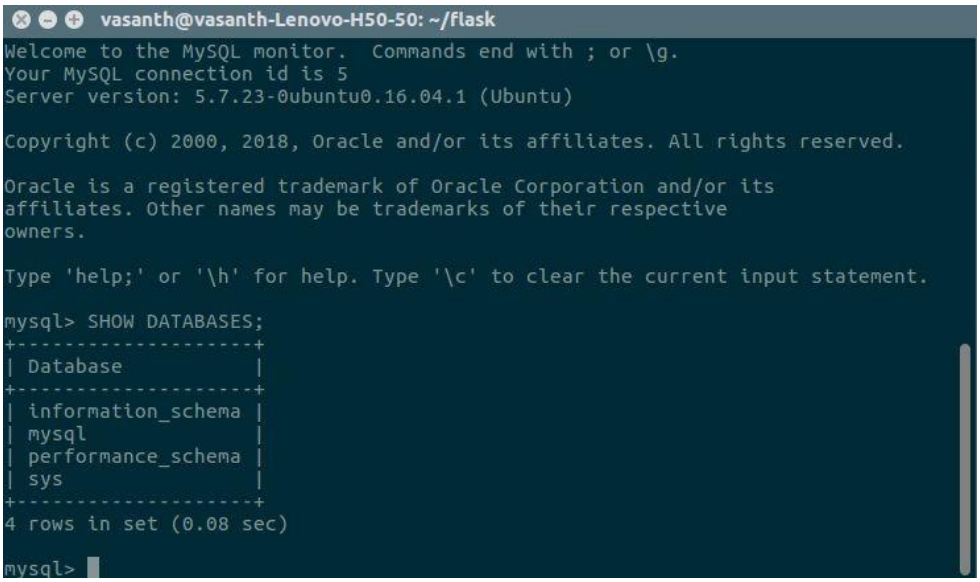
Now the Sql server will opened as shown below :



Now type the following command in the mysql terminal :

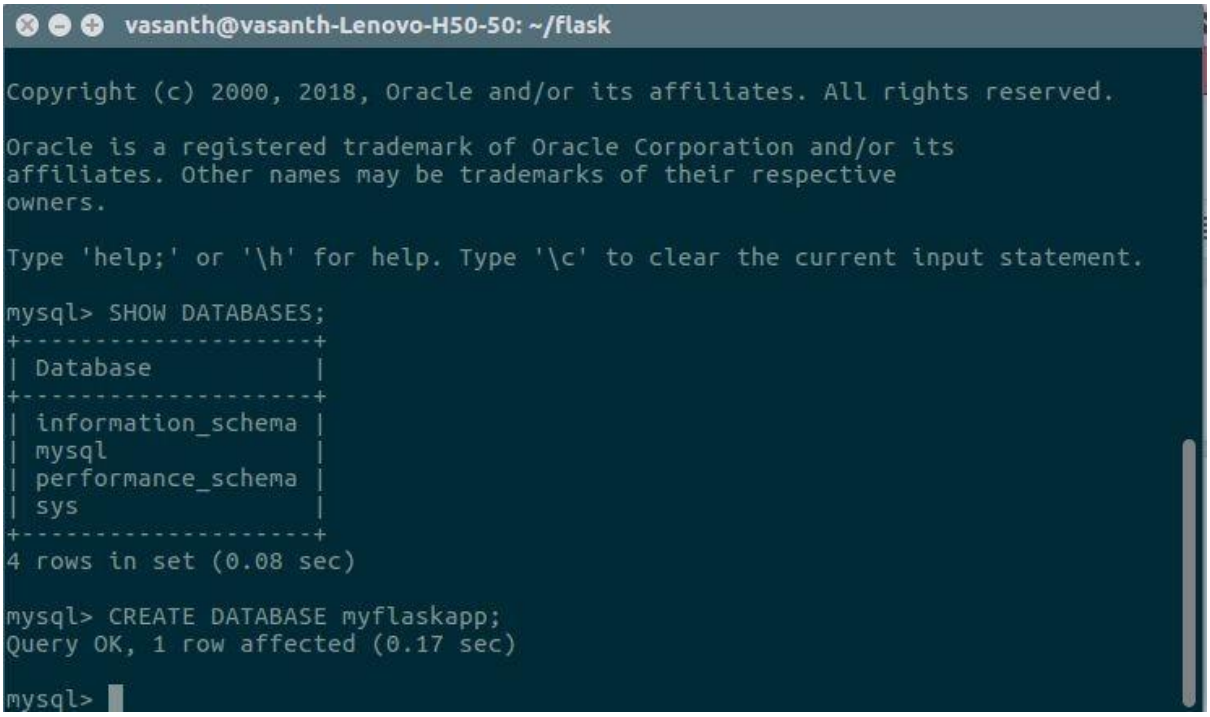
mysql> SHOW DATABASES;

Now list of databases will be displayed :



Now Create a database for our application for this type the following command in the mysql server

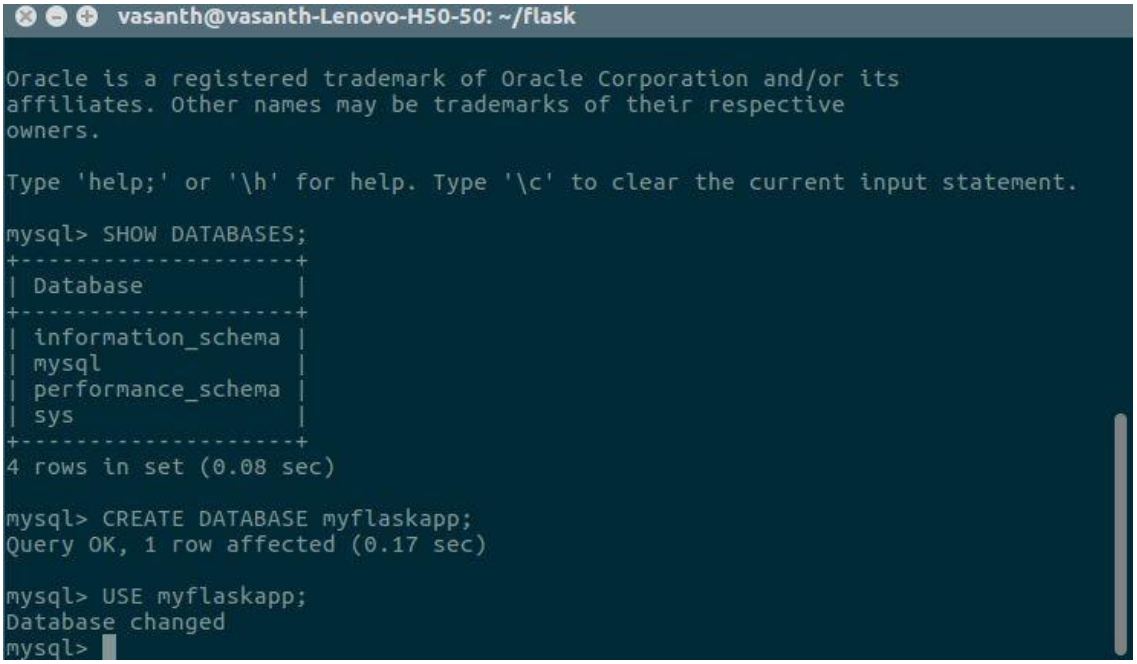
mysql> CREATE DATABASE myflaskapp;



Now Select our database to work with it,for that type the following command in the terminial .

mysql> USE myflaskapp;

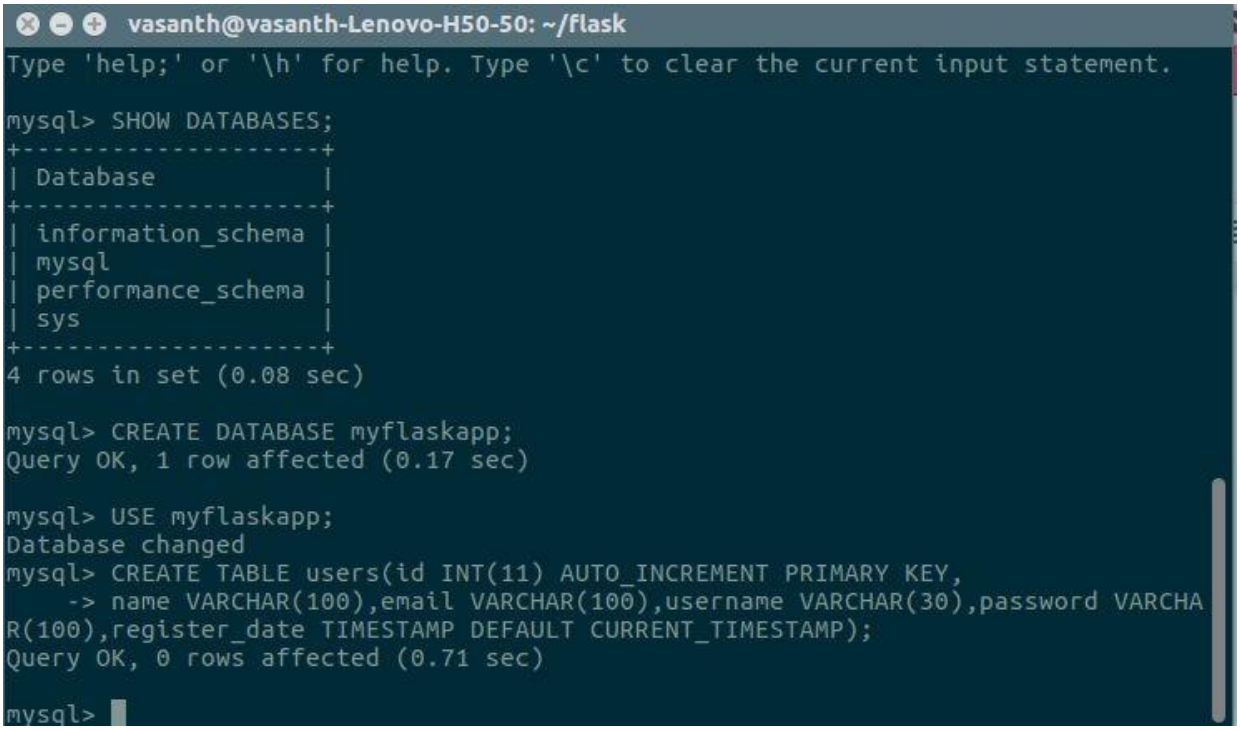
Now the database has been selected as shown below .



Now we have to create the user table ,use the following command and type in the sql server

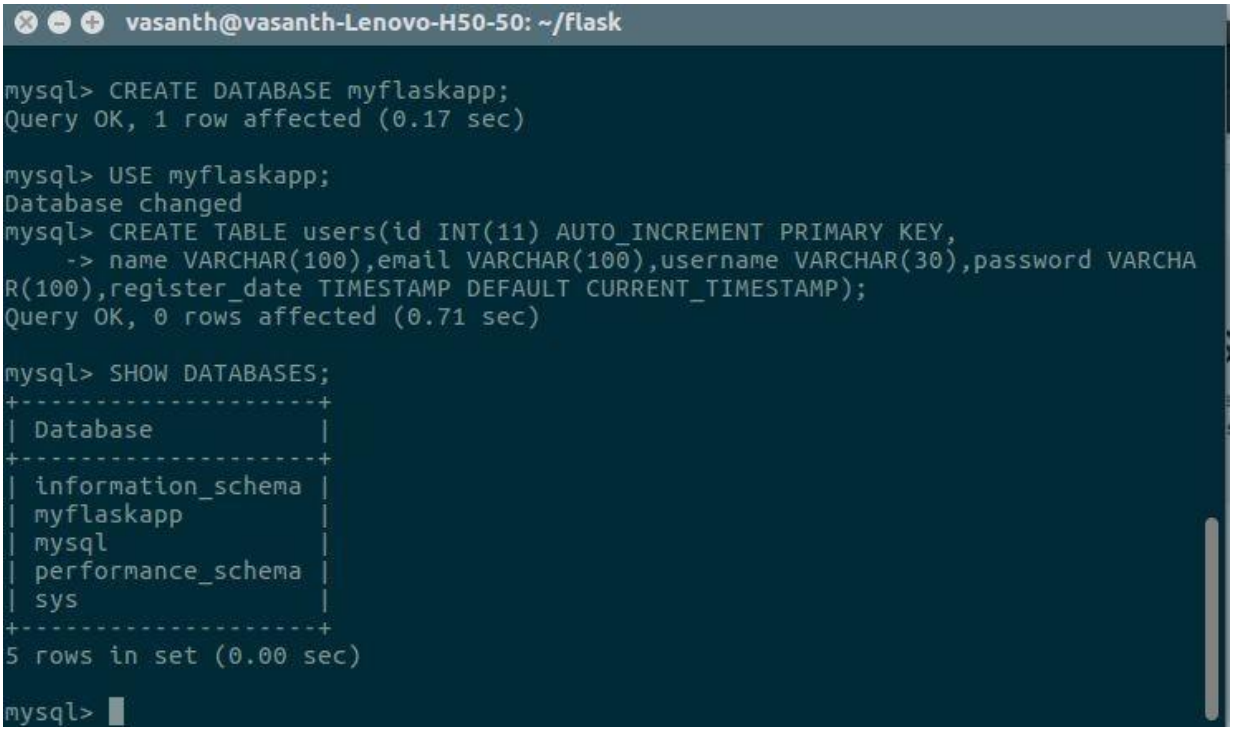
mysql> CREATE TABLE users(id INT(11) AUTO_INCREMENT PRIMARY KEY,
-> name VARCHAR(100),email VARCHAR(100),username VARCHAR(30),password VARCHAR(100),register_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP);

Now the table has been created successfully as shown below :



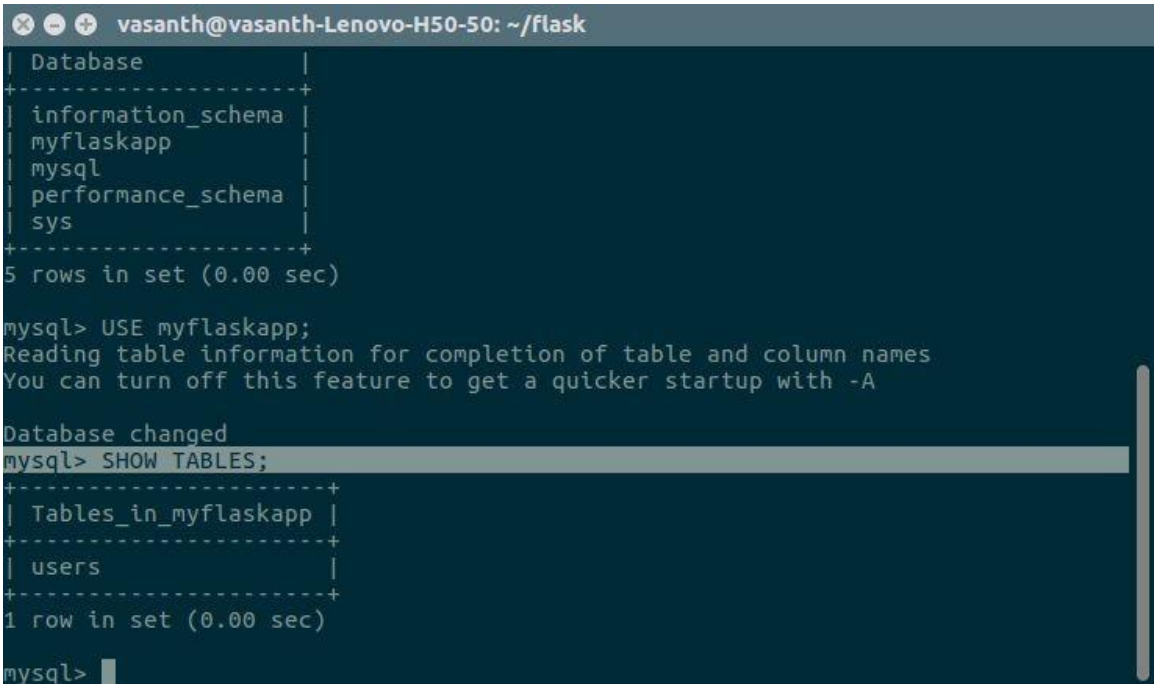
Now type the following command in the mysql server to list database :

mysql> SHOW DATABASES;



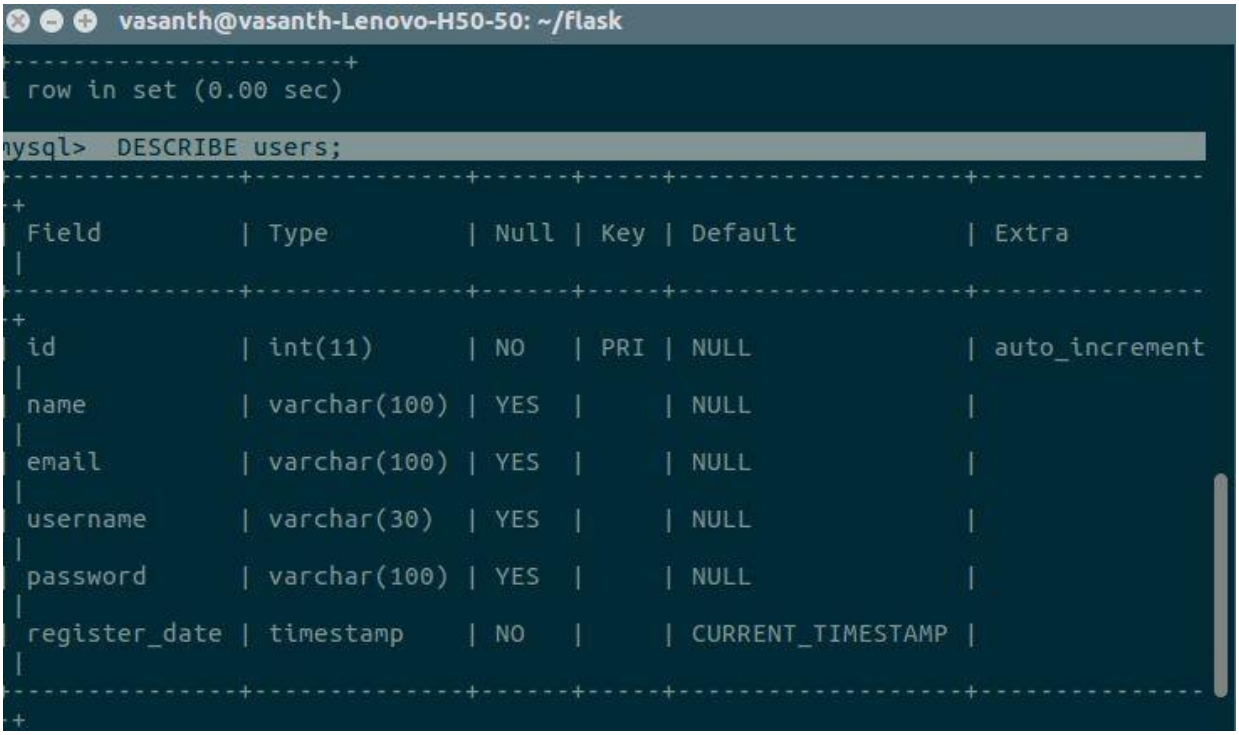
Now we can list our table by using the following command :

mysql> SHOW TABLES;



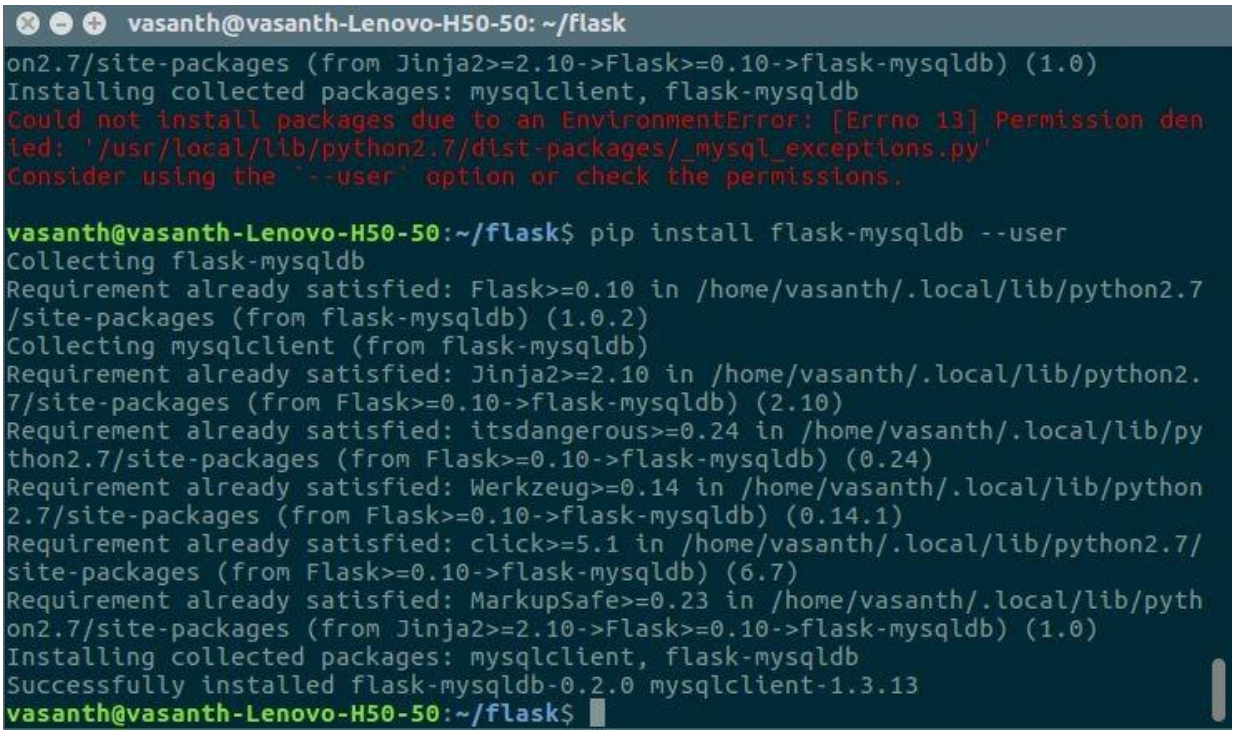
Now we can describe our users table by typing the following command in the mysql server :

mysql> DESCRIBE users;



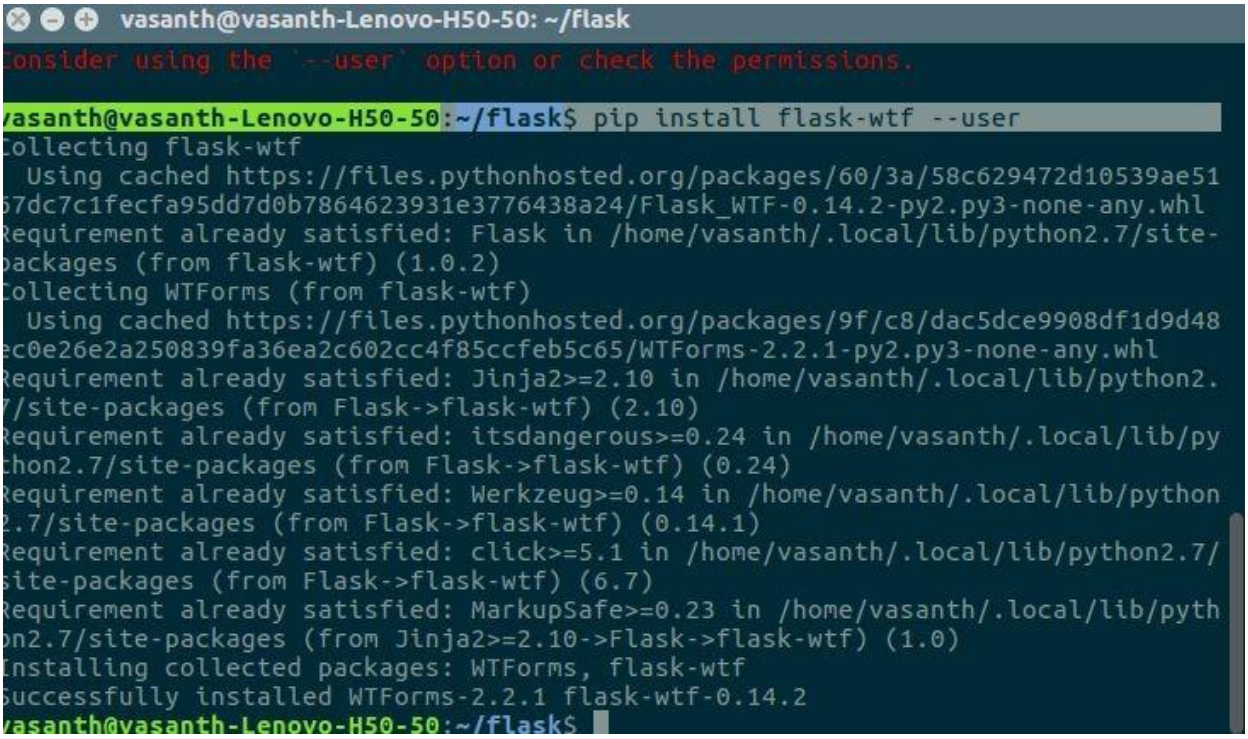
Now we have to install few things using pip ,so now we have to stop the python flask server by pressing ctrl c .

Now we want to install flask mysqldb that basically communicates with the flask and mysql ,to install this type the following command in the terminal .



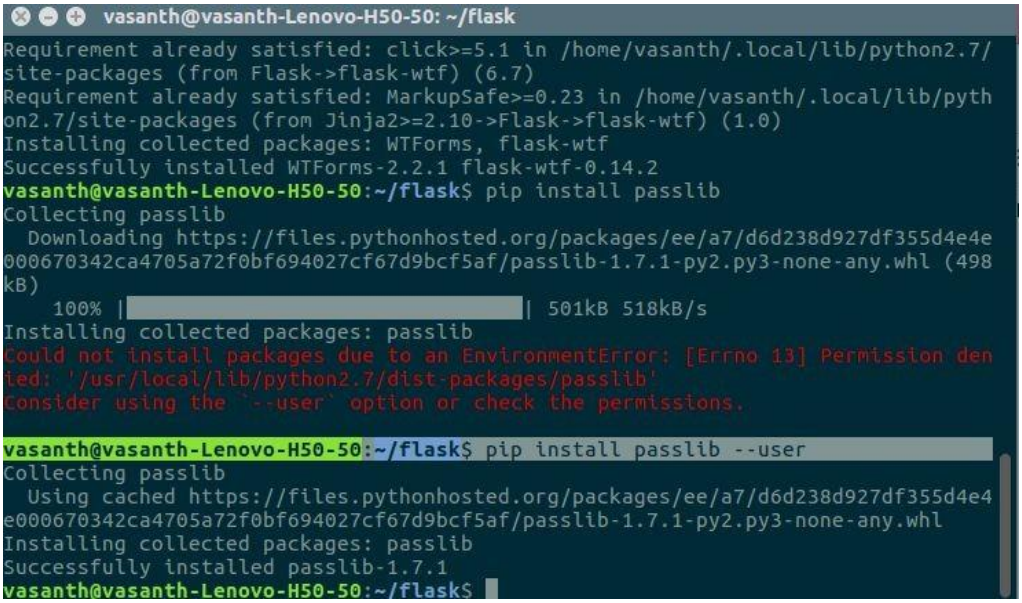
Now we have to install **WTFORMS ,**For that type the following command in the terminal ,

vasanth@vasanth-Lenovo-H50-50:~/flask$ pip install flask-wtf --user

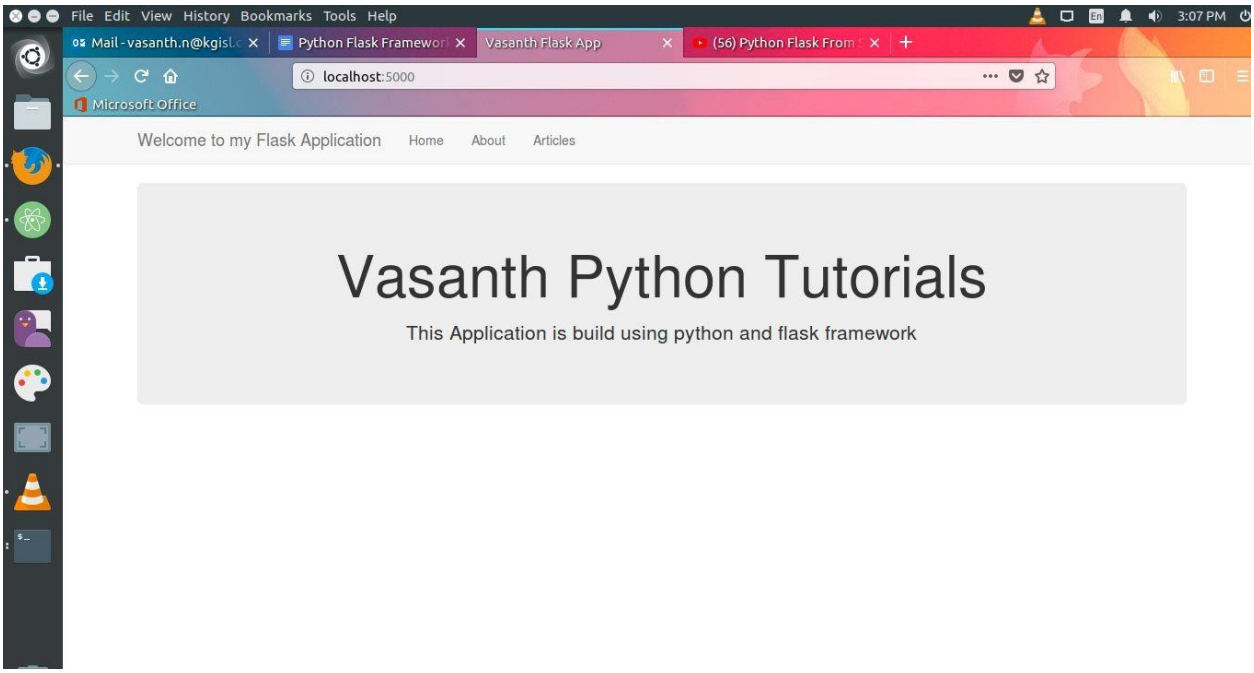


Now we are going to install passlib, which is going to encrypt the password that stored in the database .
Type the command in the terminal :

vasanth@vasanth-Lenovo-H50-50:~/flask$ pip install passlib --user



Now we can run the app,lets start the python flask server



Now we have to import some modules from flask .now go to app.py file and add the following in the app.py file .

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(__name__)
app.debug = True
mysql = MySQL(app)

Articles = Articles()

@app.route('/')
def index():
```

```
    return render_template('home.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/articles')
def articles():
    return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
    return render_template('article.html',id=id)
```

class RegisterForm(Form):
    name = StringField('Name',[validators.Length(min=1,max=50)])
    username = StringField('Username',[validators.Length(min=4,max=25)])
    email = StringField('Email',[validators.Length(min=4,max=25)])
    password = PasswordField('Password', [validators.DataRequired (),**validators.EqualTo('confirm',message ='passwords do not match')])**
    confirm = PasswordField('Confirm password')

Here we have created the Register form for our website , and now we have to create a route for the register form ,for that type the following code as given below in app.py file below the class register form :

```
@app.route('/register', methods=['GET','POST'])
def register():
    form = RegisterForm(request.form)
    if request.method == 'POST' and form.validate():

    return render_template('register.html',form=form)
```

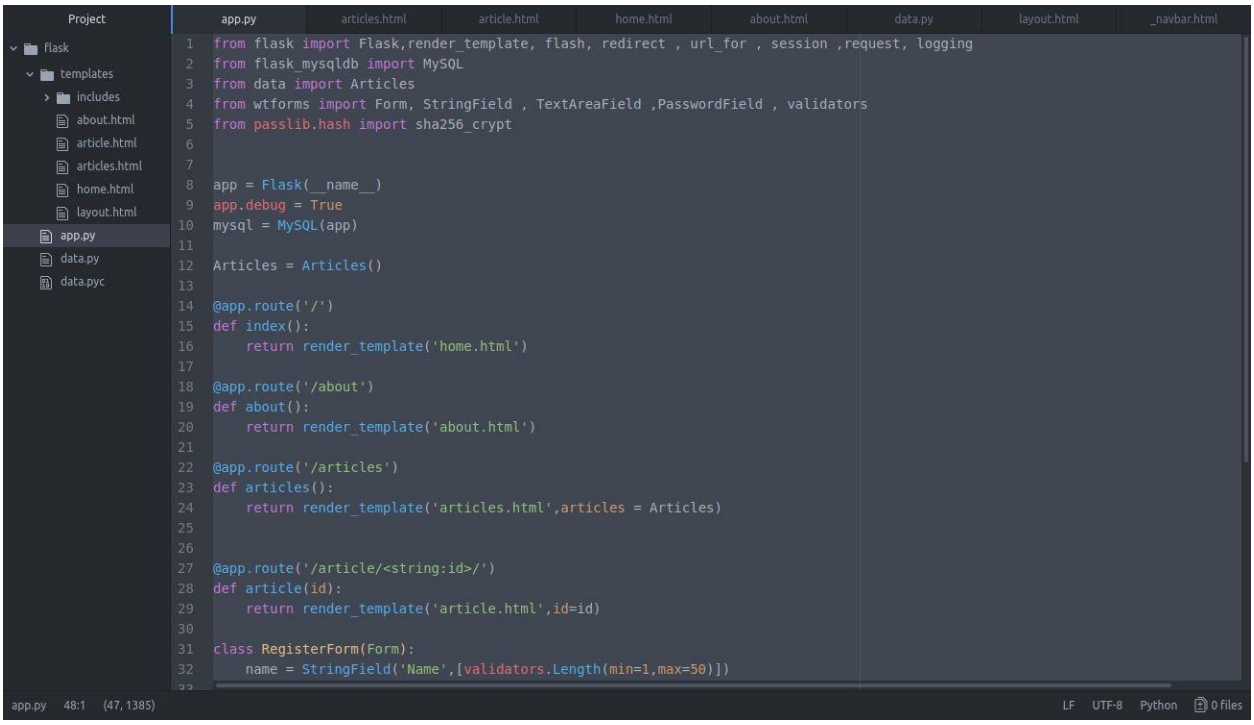
Now the entire code of app.py look like this :

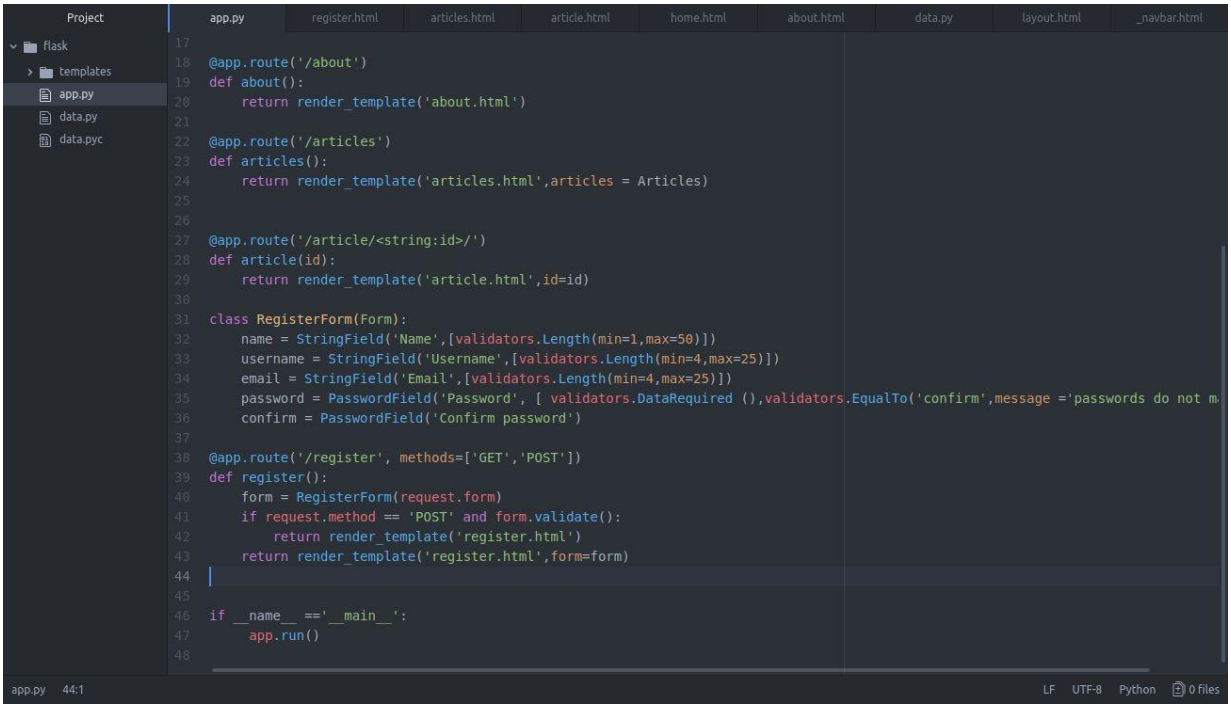
```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
```

```
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True
mysql = MySQL(app)

Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

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

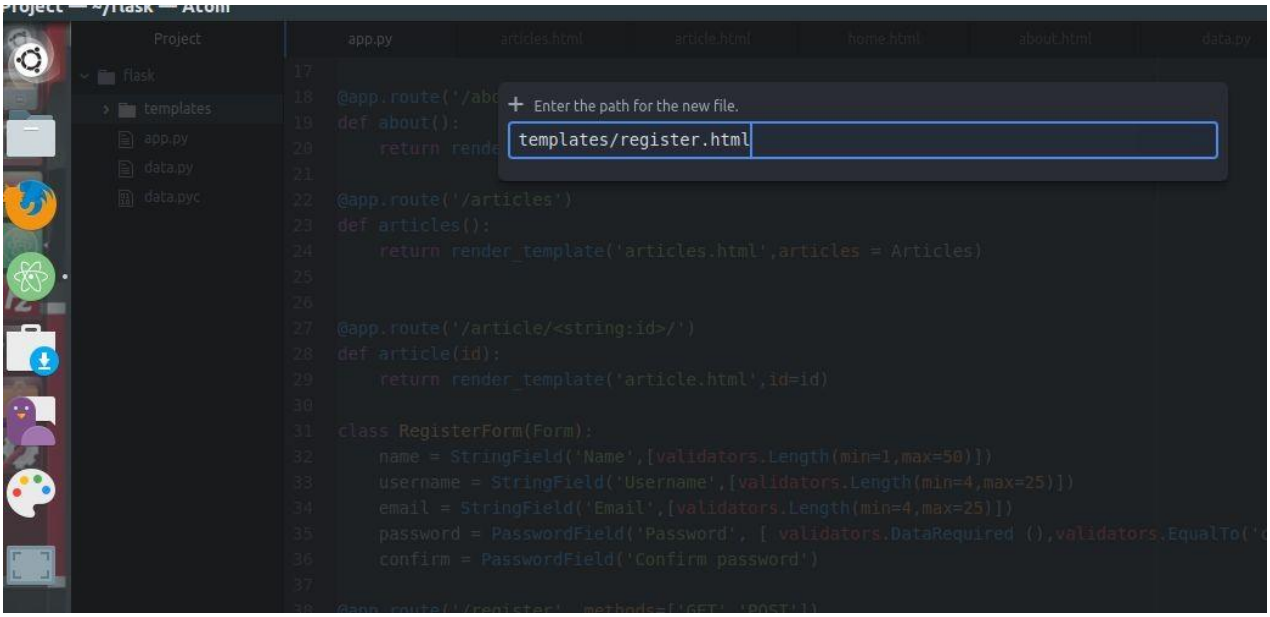
```
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        return render_template('register.html')
        return render_template('register.html',form=form


if __name__=='_main_':
        app.run()
```

The code looks like :





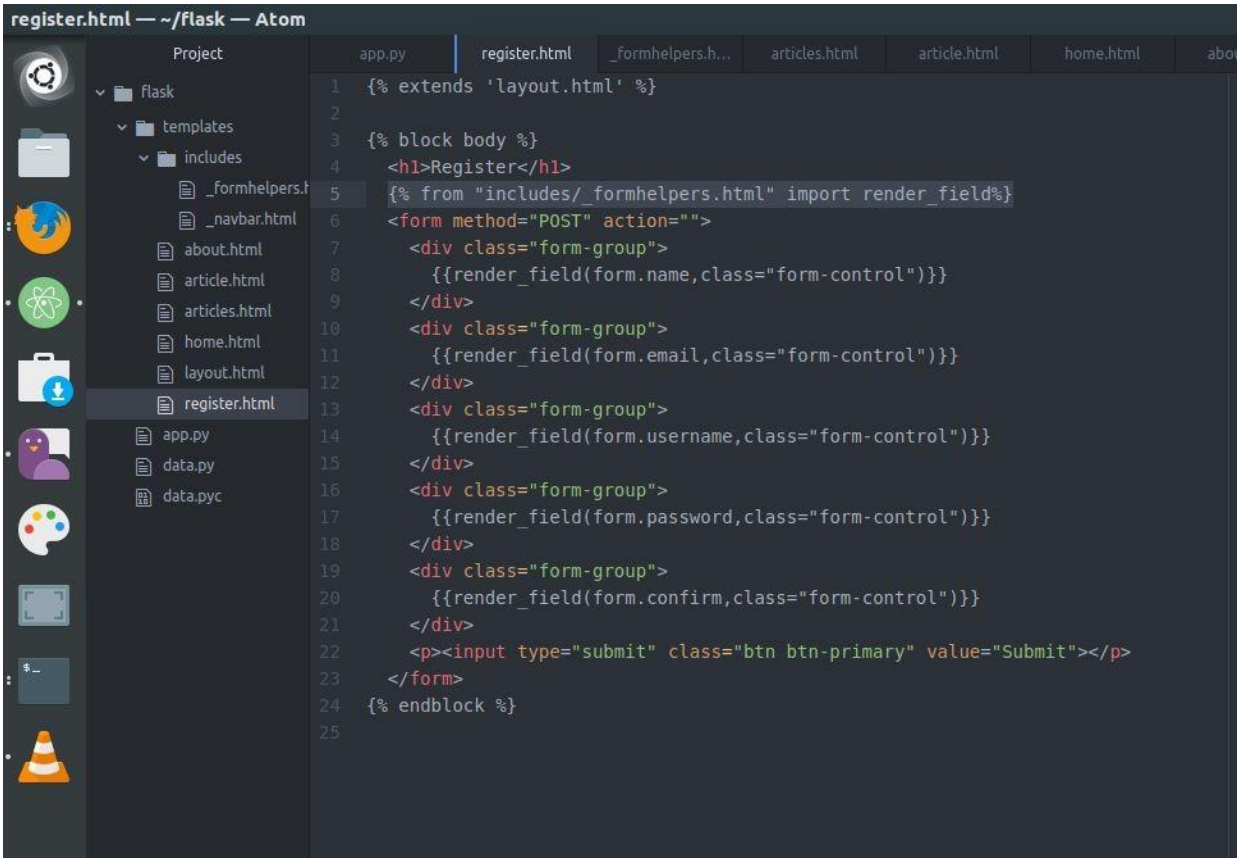
Now the next step is that we have to create a register.html page in the templates folder for that follow the steps as defined :



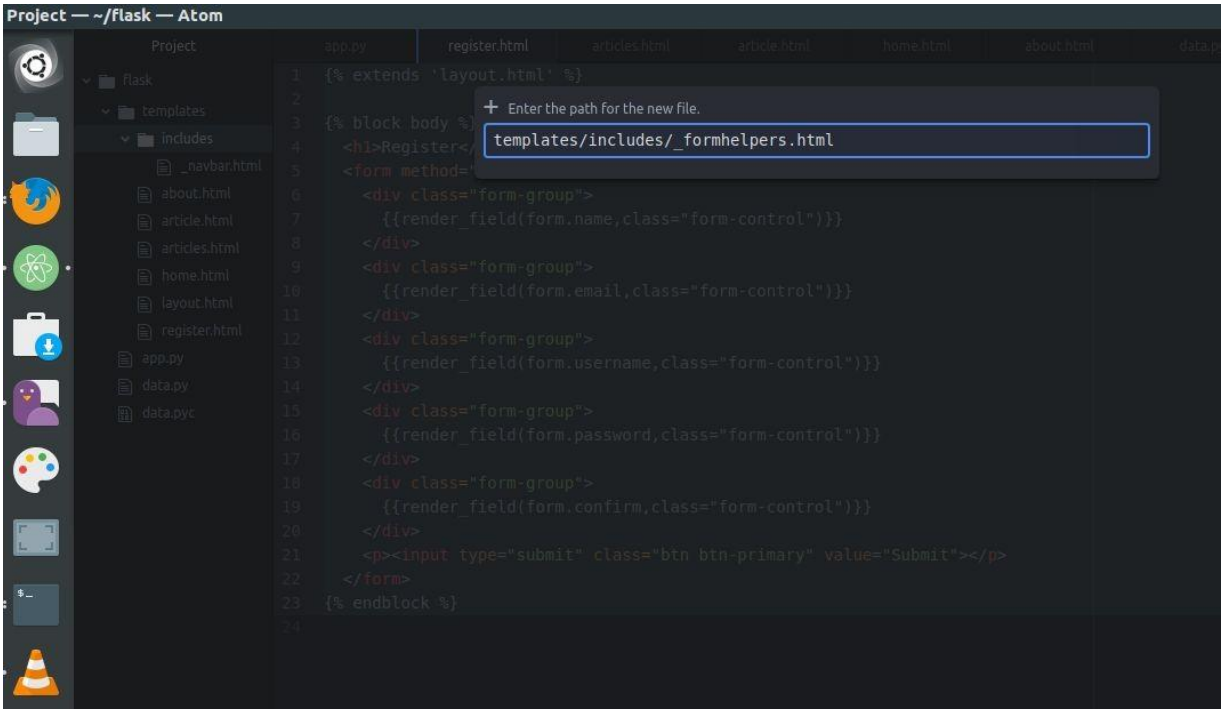
Now type the following code in the register.html file as shown below :

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Register</h1>
  {% from "includes/_formhelpers.html" import render_field%}
  <form method="POST" action="">
        <div class="form-group">
        {{render_field(form.name,class="form-control")}}
        </div>
        <div class="form-group">
        {{render_field(form.email,class="form-control")}}
        </div>
        <div class="form-group">
        {{render_field(form.username,class="form-control")}}
        </div>
        <div class="form-group">
        {{render_field(form.password,class="form-control")}}
        </div>
        <div class="form-group">
        {{render_field(form.confirm,class="form-control")}}
        </div>
        <p><input type="submit" class="btn btn-primary" value="Submit"></p>
  </form>
{% endblock %}
```

The Code should look like this :



Now go to the includes folder and create a file as **_formhelpers.html** as shown below :

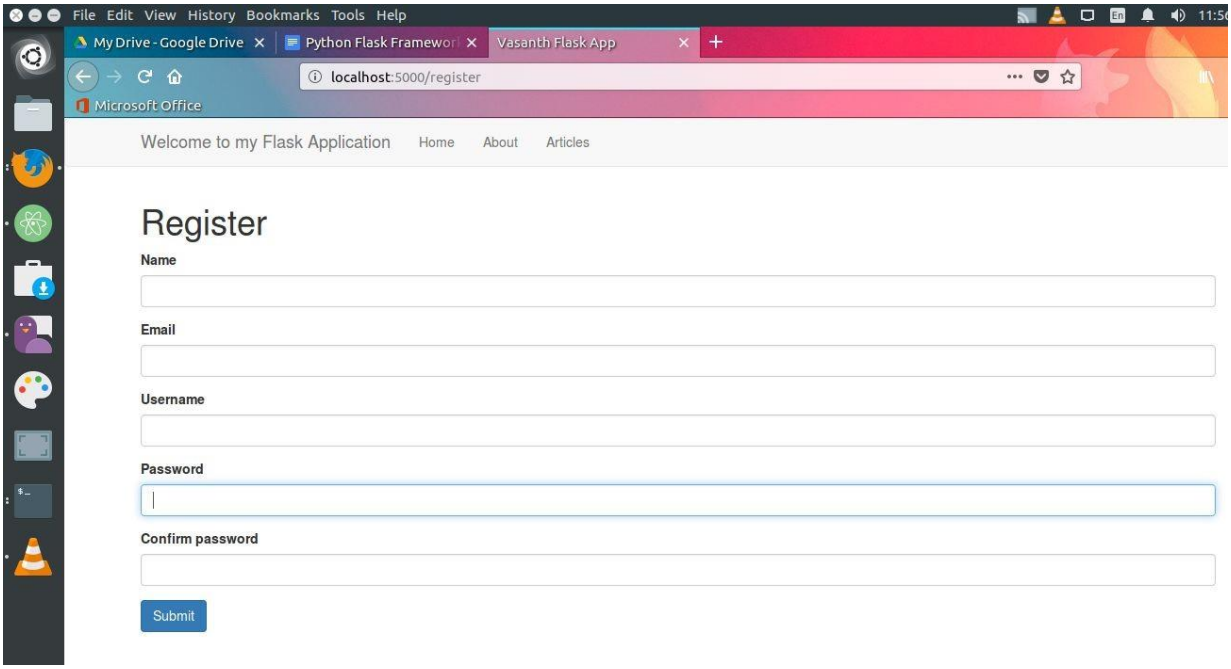


Now type the following code inside the _formhelpers.html

_formhelpers.html

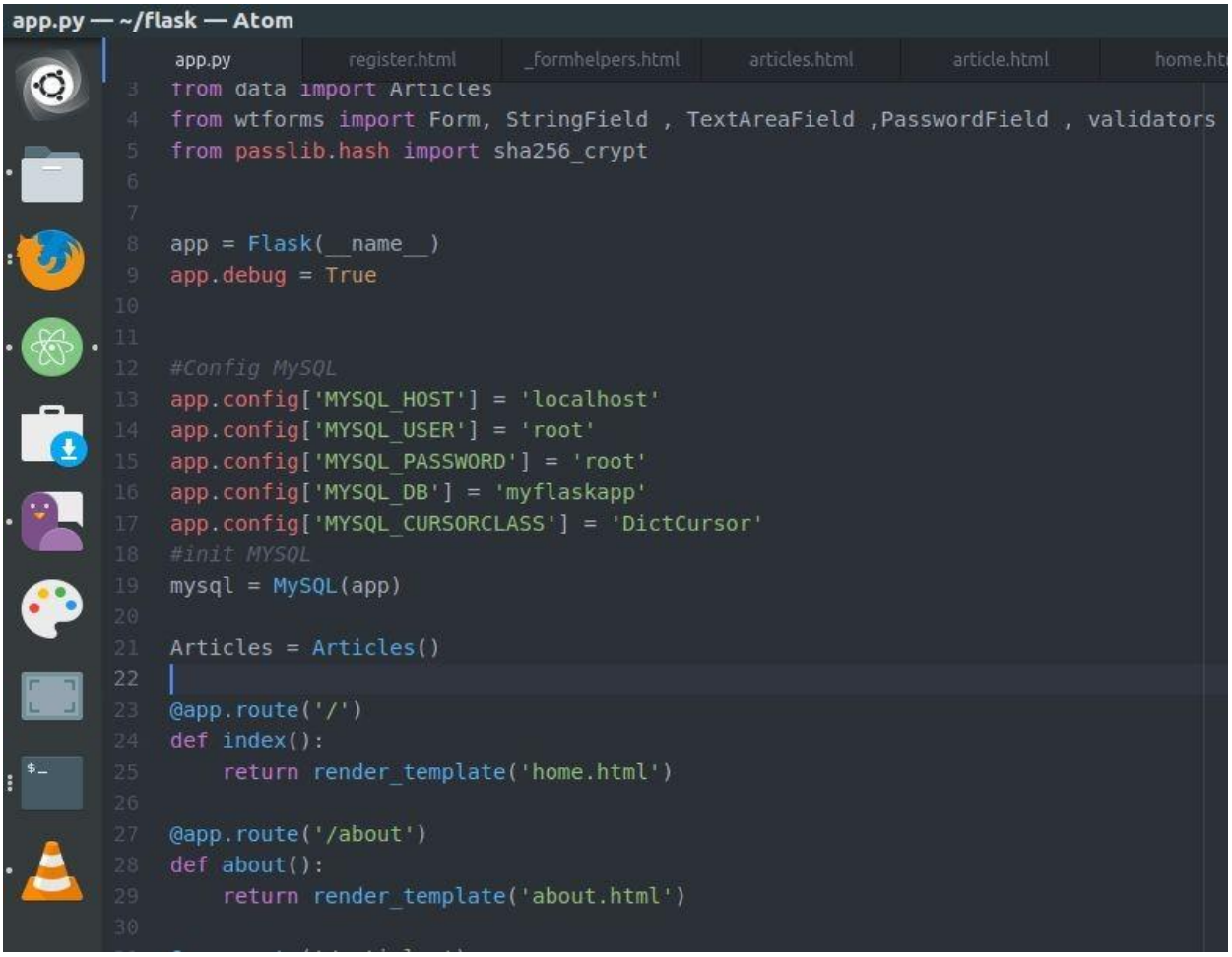
```
{% macro render_field(field) %}
  {{ field.label }}
  {{ field(**kwargs)|safe }}
  {% if field.errors %}
        {% for error in field.errors %}
        <span class="help-inline">{{ error }}</span>
        {% endfor %}
  {% endif %}
{% endmacro %}
```

And now save the file and run the python flask server and navigate to the register page and the register page will look like as :



Now we are going to connect the page with our mysql database for that we have to go to app.py file and add the configuration details in the **app.py** file as shown below :

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)

Articles = Articles()

@app.route('/')
```



Now we want to configure the register form function in the app.py file ,follow the instructions and make the changes in the app.py file

```
        @app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()
```

```
        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registerd and you can login", 'success')

        redirect(url_for('index'))

        return render_template('register.html')
        return render_template('register.html',form=form)
```

The Entire Code of **app.py** will look like as :

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request,
logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)
```

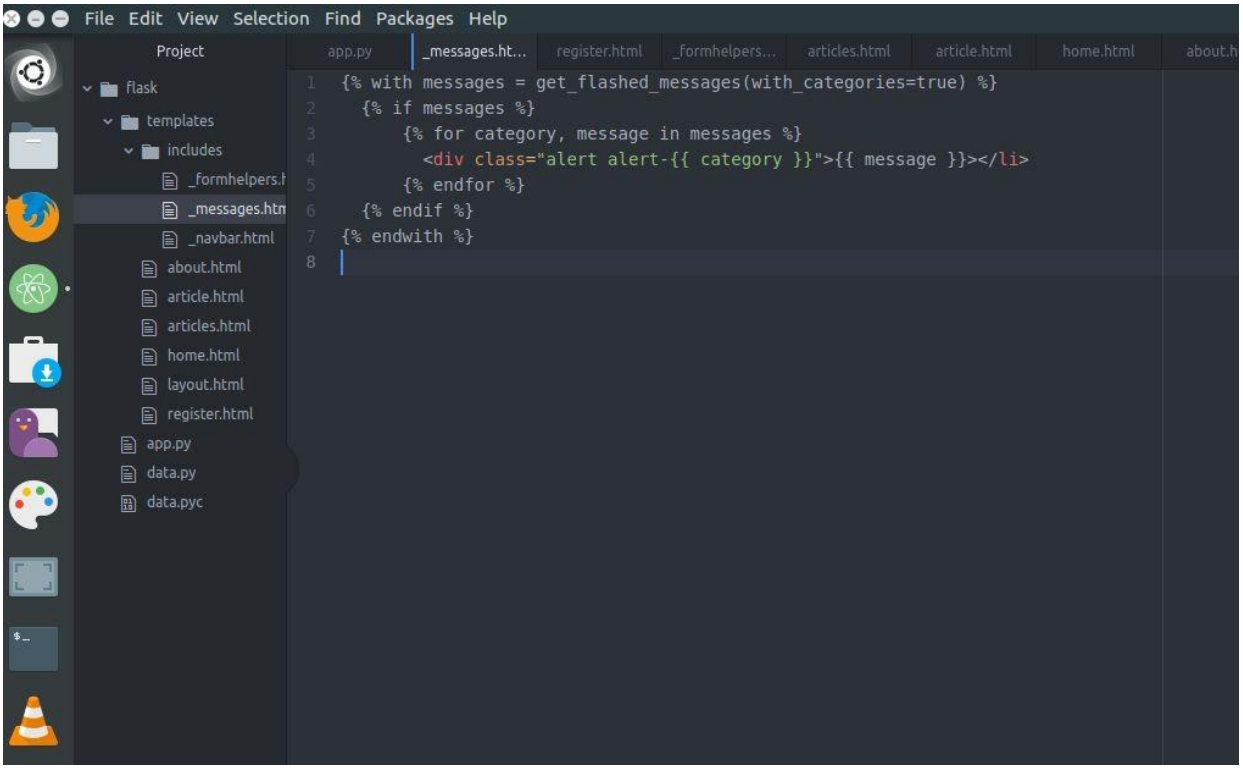
```
Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()
```

```
        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login", 'success')

        redirect(url_for('index'))
        return render_template('register.html',form=form)


if __name__=='_main_':
        app.run()
```

Now We have to create a file for messages for that we have create a new file in the includes folder and name the file as **_messages.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 %}
```

The file will look like :

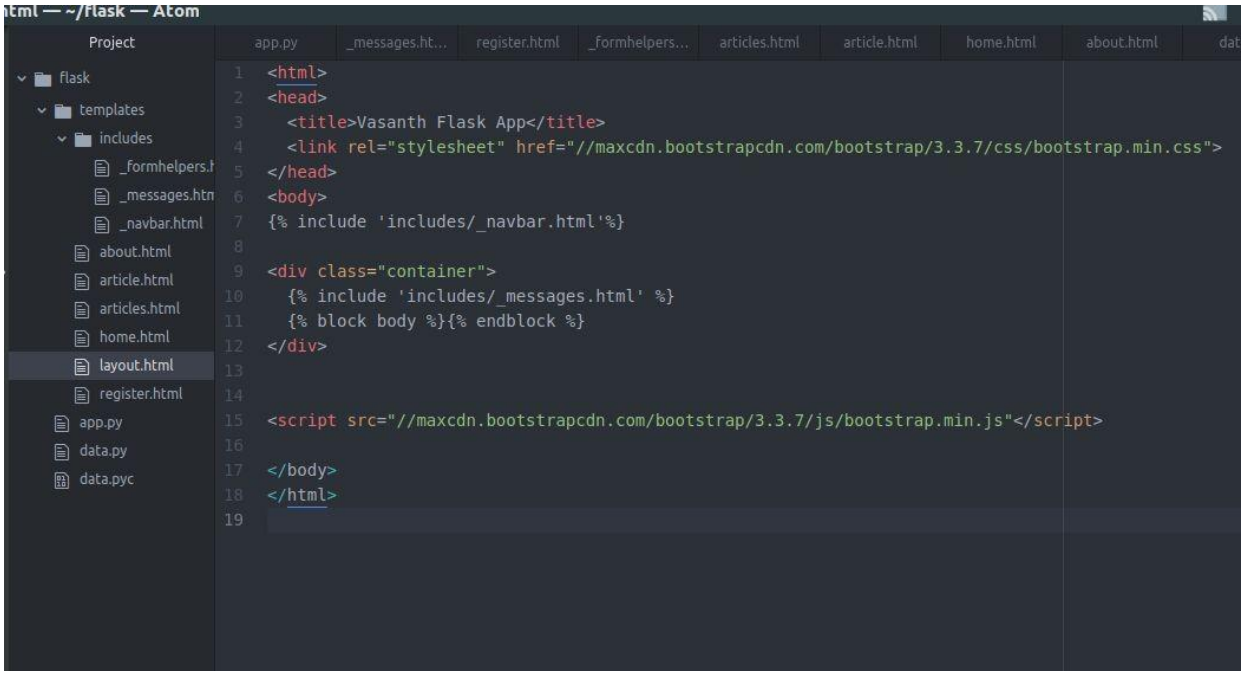


Now we have to add this in the layouts ,now go to layouts.html file and add up the following code in that .

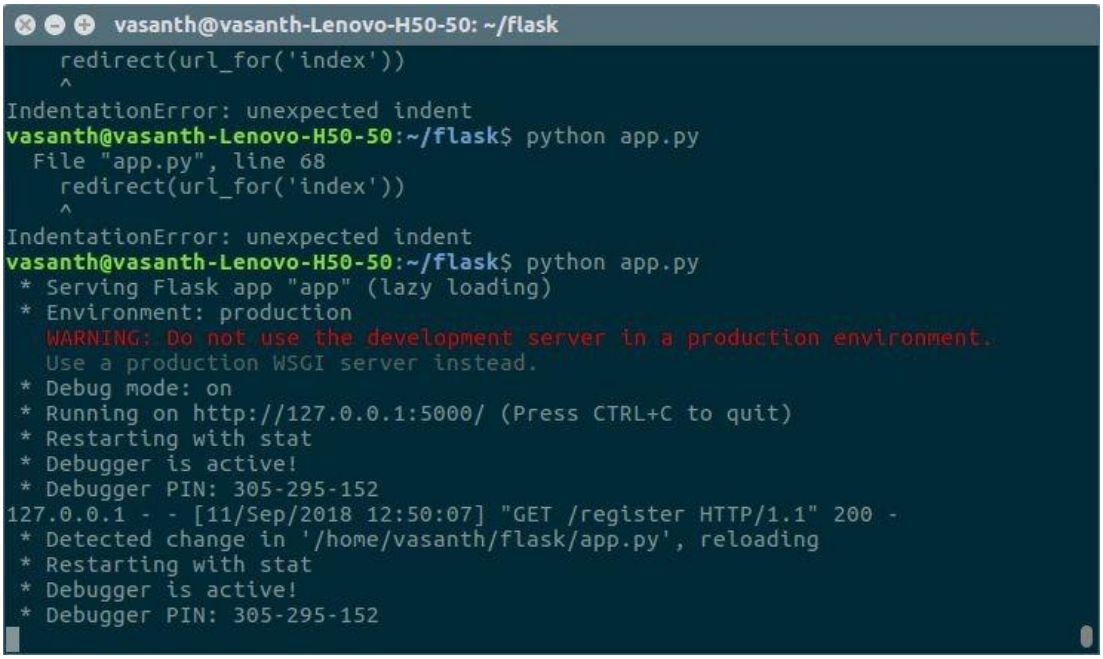
```
<html>
<head>
  <title>Vasanth Flask App</title>
  <link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
{% include 'includes/_navbar.html'%}

<div class="container">
  {% include 'includes/_messages.html' %}
  {% block body %}{% endblock %}
</div>


<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"</script>
```

</body>
</html>

Add the following changes and save the file ,now the file should look like this :



Now run the python server ,make sure there should be no errors .

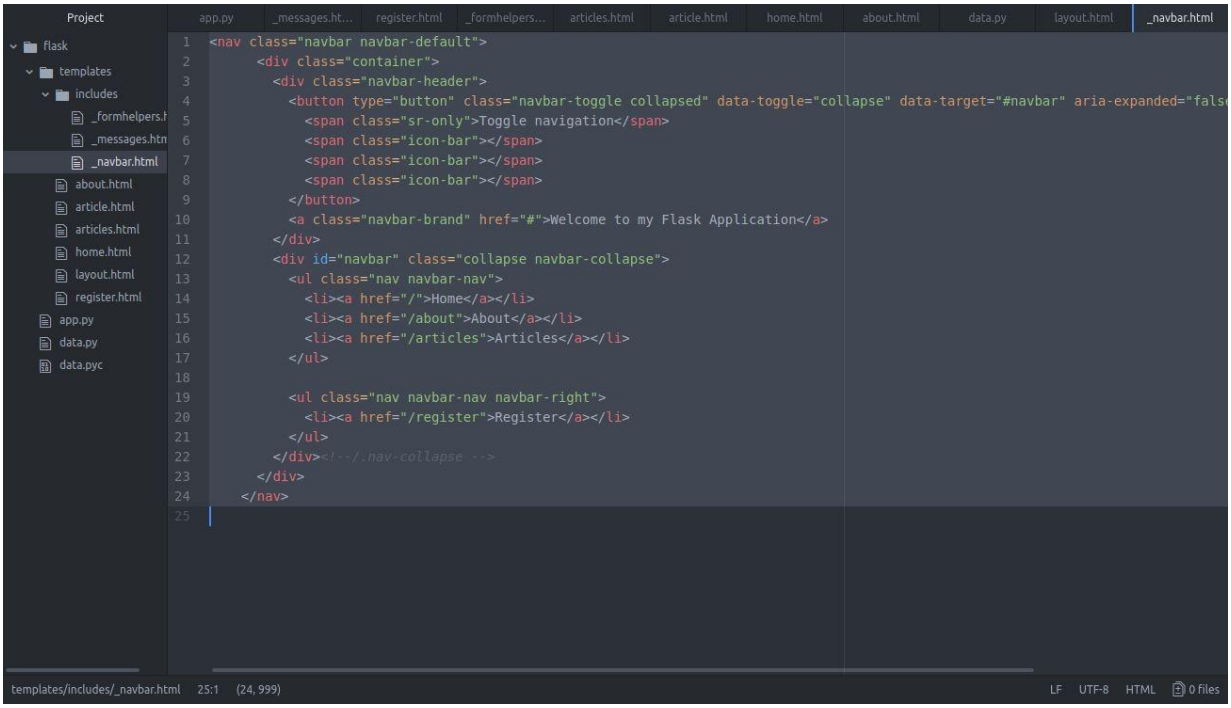


Now we need the register link in the home page for that just go the _navbar.html file and add the following code in this file as shown below :

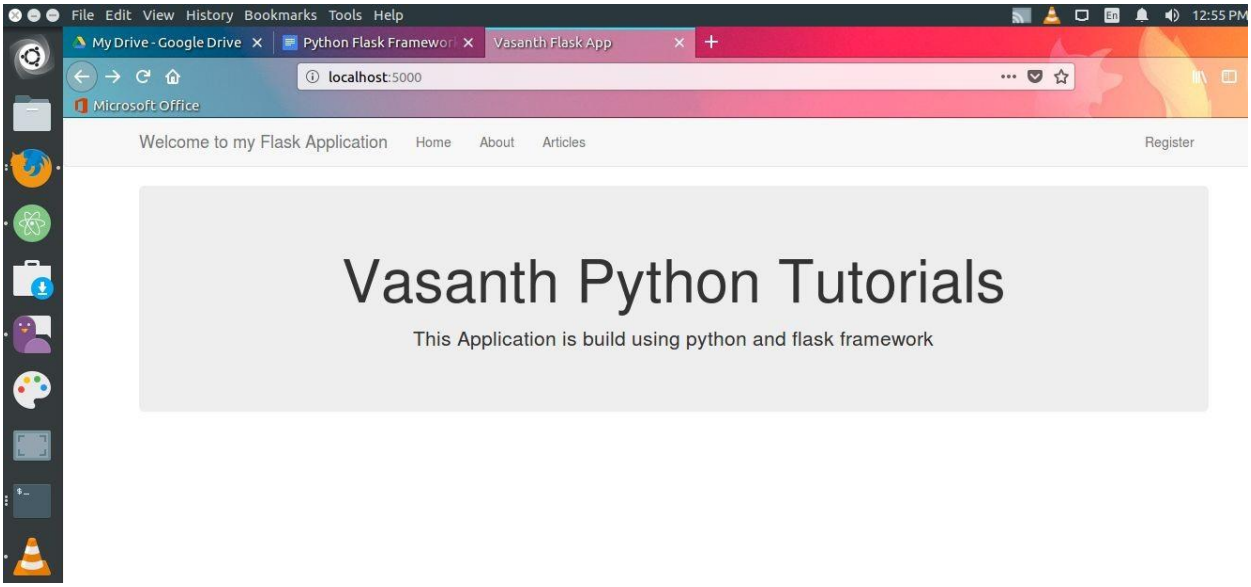
```
<nav class="navbar navbar-default">
        <div class="container">
        <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
data-target="#navbar" aria-expanded="false" aria-controls="navbar">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        </button>
        <a class="navbar-brand" href="#">Welcome to my Flask Application</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
        <li><a href="/">Home</a></li>
        <li><a href="/about">About</a></li>
        <li><a href="/articles">Articles</a></li>
        </ul>

        <ul class="nav navbar-nav navbar-right">
        <li><a href="/register">Register</a></li>
```

```
</ul>
</div><!--/.nav-collapse -->
</div>
</nav>
```

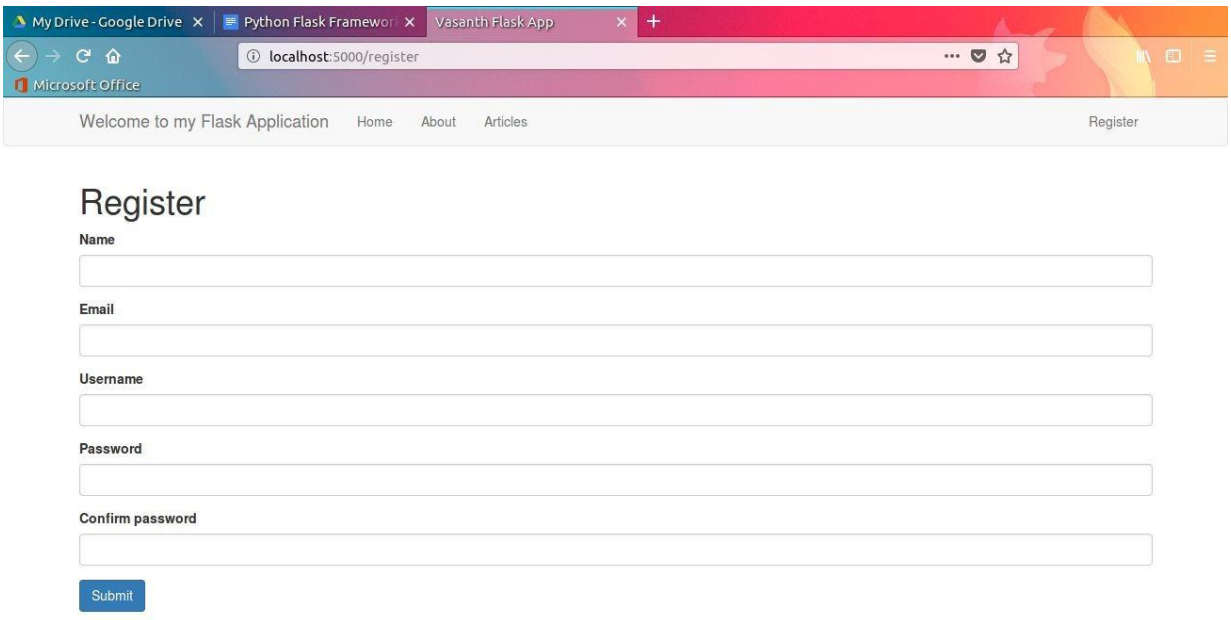
The code should look like this :



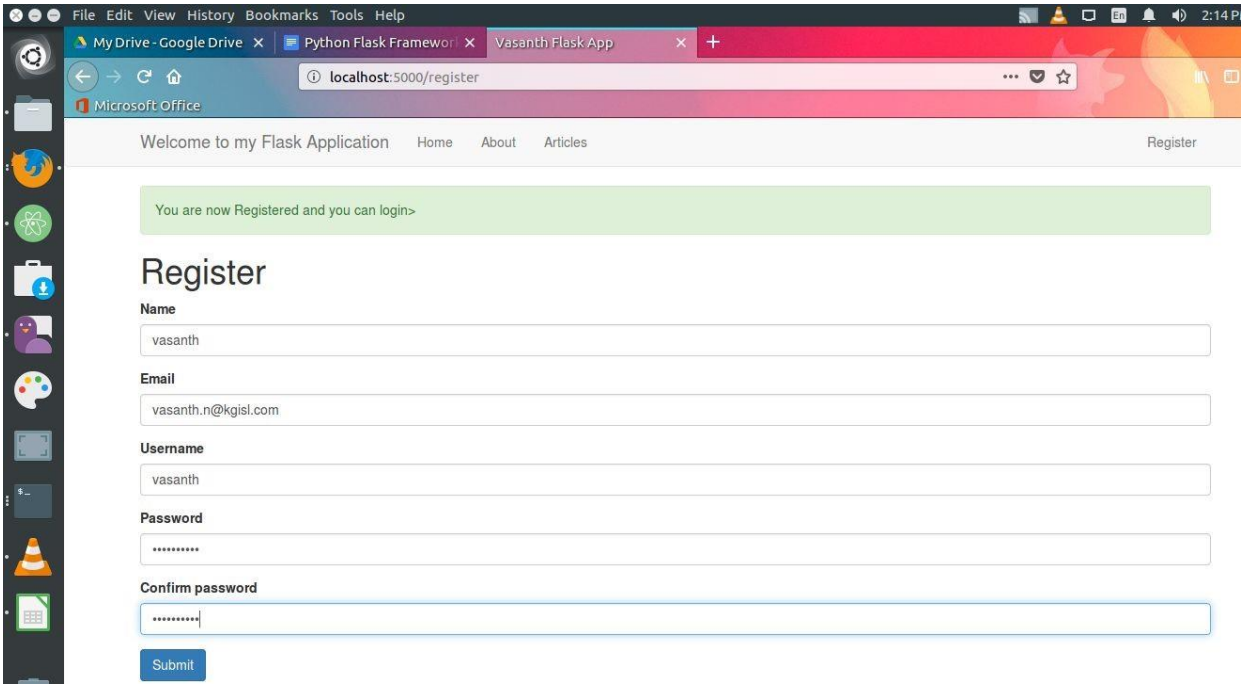
Now save and run the file ,ie. Run the python server and you will be seeing the register option that on the right top side of the website as shown below .



Now Click the registration you will see the following form :



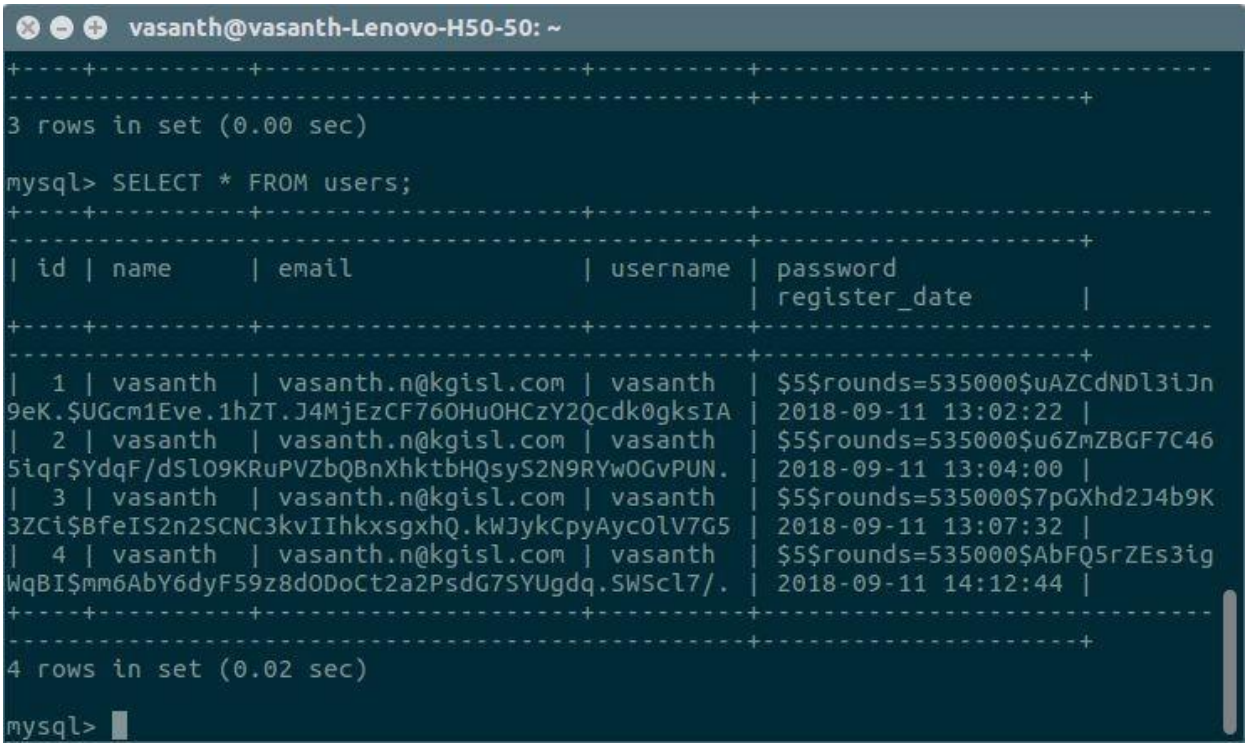
Now fill up the data and click submit ,



After filled up the data and once the submit button has been clicked the data will be saved in the database,now we can check our database by using the following command .

Type the following sql query in the mysql shell :

mysql> SELECT * FROM users;



You can see that our data are stored in the database and our password has been encrypted and stored in the database .

Now once we submitted the page has been to redirect to login form,for that we have change the following in the **app.py** file

```
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
```

```
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)
```

Now we have to create the login page for our website ,for this follow the below instruction to create a login page :

Go to **app.py** file and write the following code :(we are creating the route for the login)

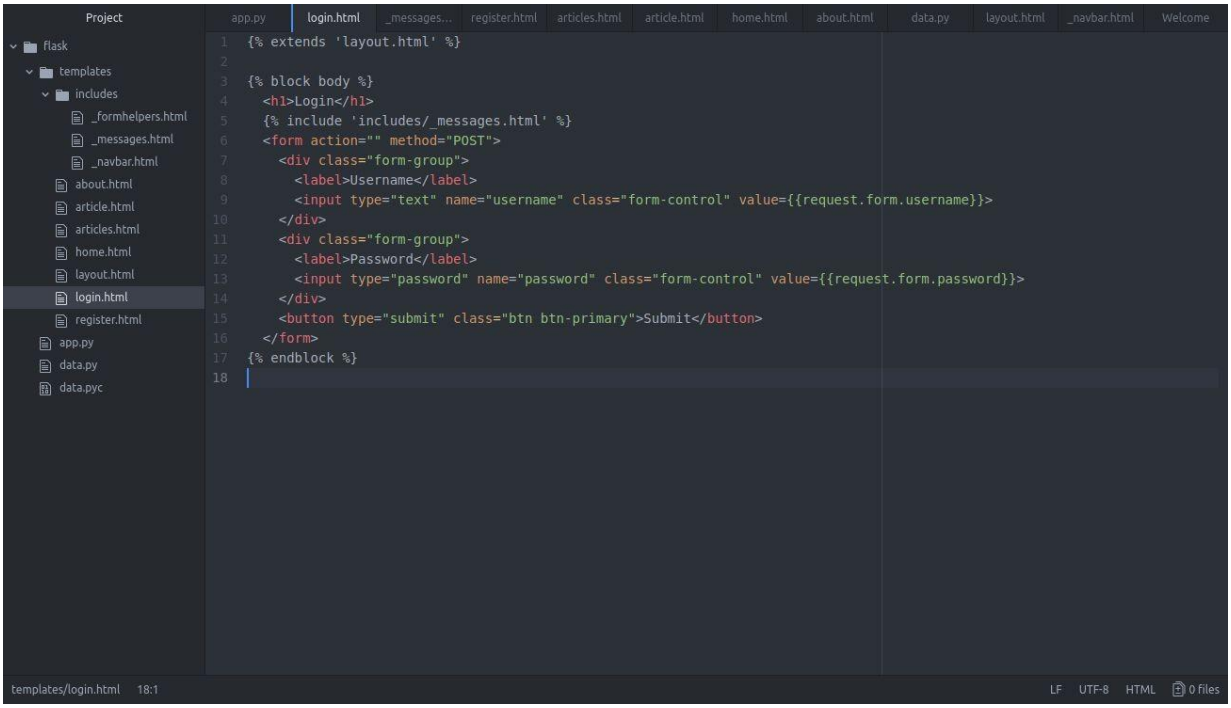
```
# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s"
,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']
```

```
        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                app.logger.info('PASSWORD MATCHED')
        else:
        app.logger.info('NO USER')

        return render_template('login.html')
```

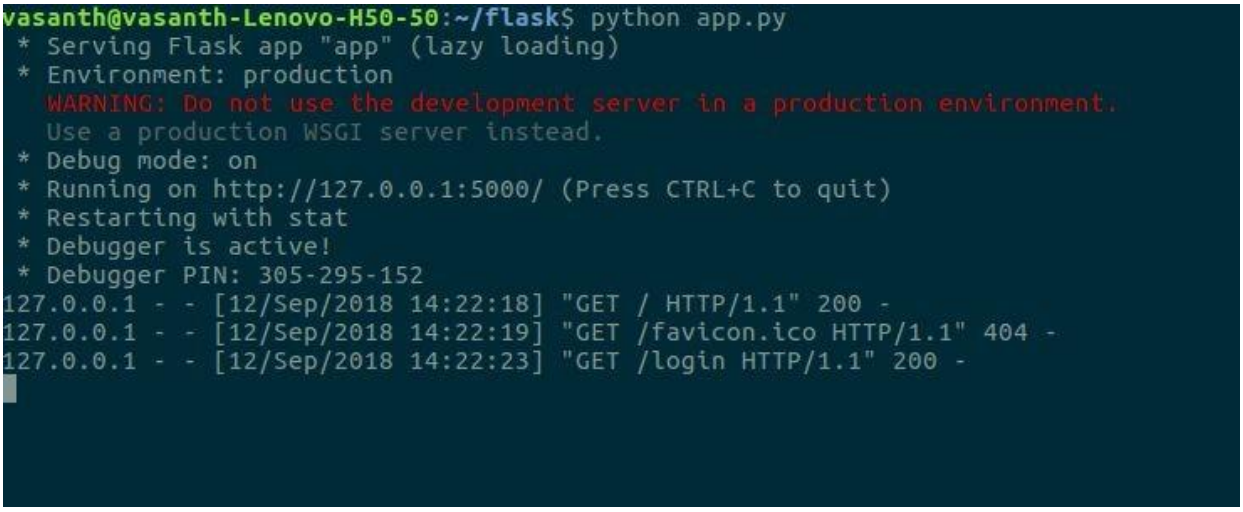
Now we have to create a login.html page for our application for that now go to template and create the **login.html** as shown below :

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Login</h1>
  <form action="" method="POST">
        <div class="form-group">
        <label>Username</label>
        <input type="text" name="username" class="form-control"
value={{request.form.username}}>
        </div>
        <div class="form-group">
        <label>Password</label>
        <input type="password" name="password" class="form-control"
value={{request.form.password}}>
        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
  </form>
{% endblock %}
```

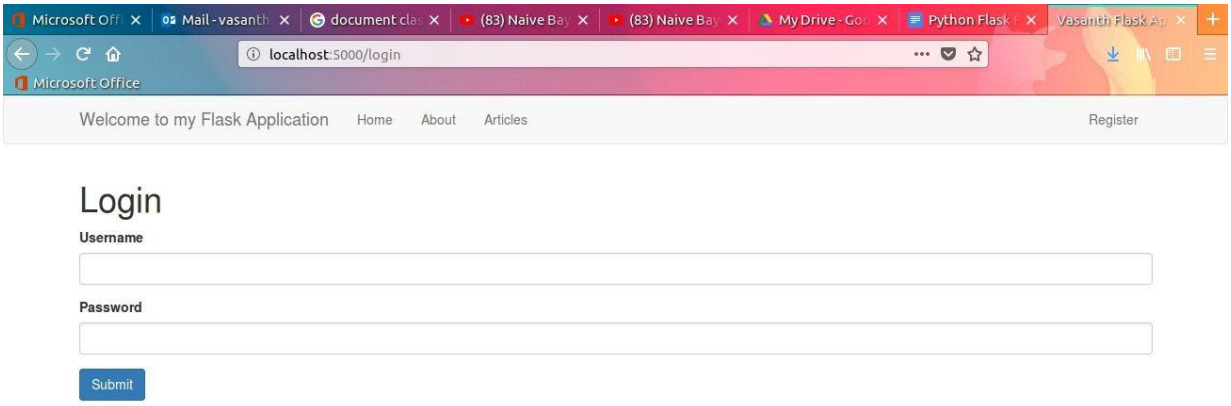
Your code should look like this :



Now save your file and run the python server ,now the server should run without any errors :



Now you can see the browser that our application running and type the login.html in the url ,now you can see the login page as shown below :



Now we are going to load the templates and create the session ,once the user login we have use the session to store the user name of the user whomever login .

Now go back to app.py and make the following changes :

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
```

```
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
```

```
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
```

```
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                app.logger.info('PASSWORD MATCHED')
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')
if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

We need to output the errors somewhere within the templates ,so now go back to the messages file ,in the includes for messages and make the following changes

_messages.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 %}


{% if error %}
  <div class="alert alert-danger">{{error}}</div>
{% endif %}
```

```
{% if msg %}
  <div class="alert alert-success">{{msg}}</div>
{% endif %}
```

After making the changes in the above file now save the file and now we have add the login link to the navbar ,so now go to navbar.html file and make the following changes .

_navbar.html

```
<nav class="navbar navbar-default">
        <div class="container">
        <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
data-target="#navbar" aria-expanded="false" aria-controls="navbar">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        </button>
        <a class="navbar-brand" href="#">Welcome to my Flask Application</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
        <li><a href="/">Home</a></li>
        <li><a href="/about">About</a></li>
        <li><a href="/articles">Articles</a></li>
        </ul>

        <ul class="nav navbar-nav navbar-right">
        <li><a href="/register">Register</a></li>
        <li><a href="/login">Login</a></li>
        </ul>
        </div><!--/.nav-collapse -->
        </div>
        </nav>
```

Now go to app.py and make the following changes :

app.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)
```

```
@app.route('/article/<string:id>/')
def article(id):
	return render_template('article.html',id=id)

class RegisterForm(Form):
	name = StringField('Name',[validators.Length(min=1,max=50)])
	username = StringField('Username',[validators.Length(min=4,max=25)])
	email = StringField('Email',[validators.Length(min=4,max=25)])
	password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
	confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
	form = RegisterForm(request.form)
	if request.method == 'POST' and form.validate():
	name = form.name.data
	email = form.email.data
	username = form.username.data
	password = sha256_crypt.encrypt(str(form.password.data))

	# Create crusor
	cur = mysql.connection.cursor()

	cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

	# commit to DB
	mysql.connection.commit()
	#close connection
	cur.close()

	flash("You are now Registered and you can login" , 'success')

	redirect(url_for('login'))
	return render_template('register.html',form=form)
```

```
# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)
```

```
        return render_template('login.html')
if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now we are going to create a dashboard link in the app.py for that make the following changes in the app.py

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
```

```
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
```

```
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
```

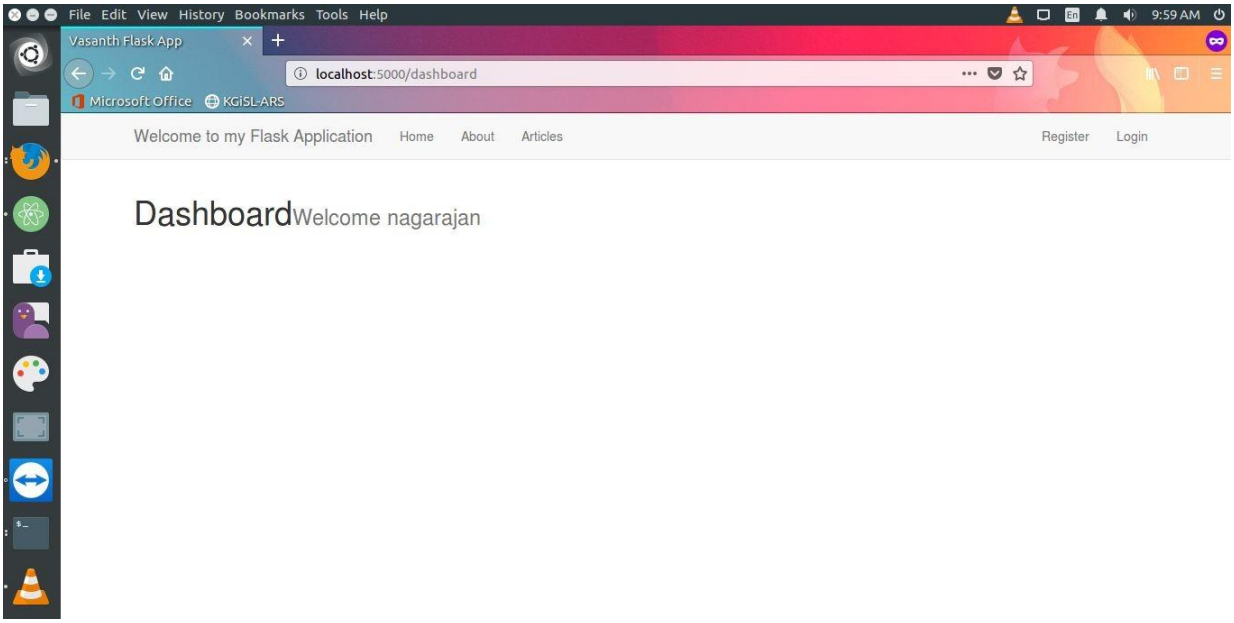
```
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

@app.route('/dashboard')
def dashboard():
        return render_template('dashboard.html')

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now we need to create the dashboard.html file in the templates for that go to the templates folder and create a new file called dashboard.html

Dashboard.html

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Dashboard<small>Welcome {{session.username}}</small></h1>
{% endblock %}
```

Now save the file and run the app.py server once you have successfully logged in ,you will get the user name in the dashboard as shown below .



Now we have create a logout option ,so now go to **app.py** and make the following changes in the app.py

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
```

```
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))
```

```
        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s"
,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']
```

```
        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#logout
@app.route('/logout')
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))

@app.route('/dashboard')
def dashboard():
        return render_template('dashboard.html')

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now make the above changes and save the file and now we have to create a link in the navbar for logout,for that now go to the navbar.html and make the following changes .

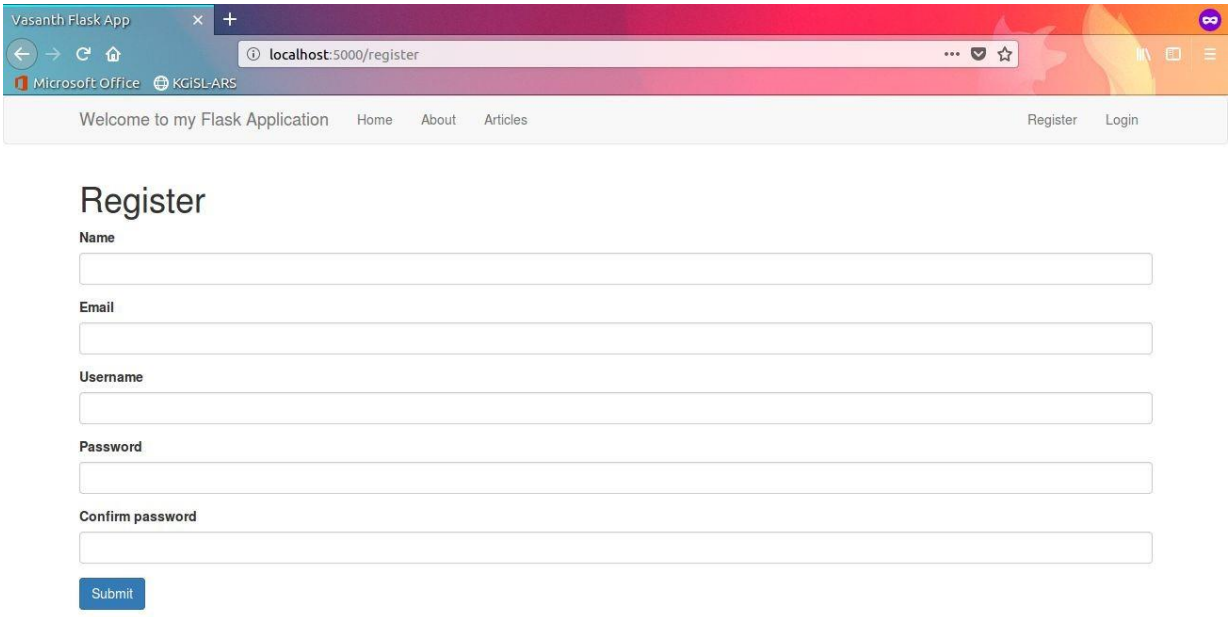
_navbar.html

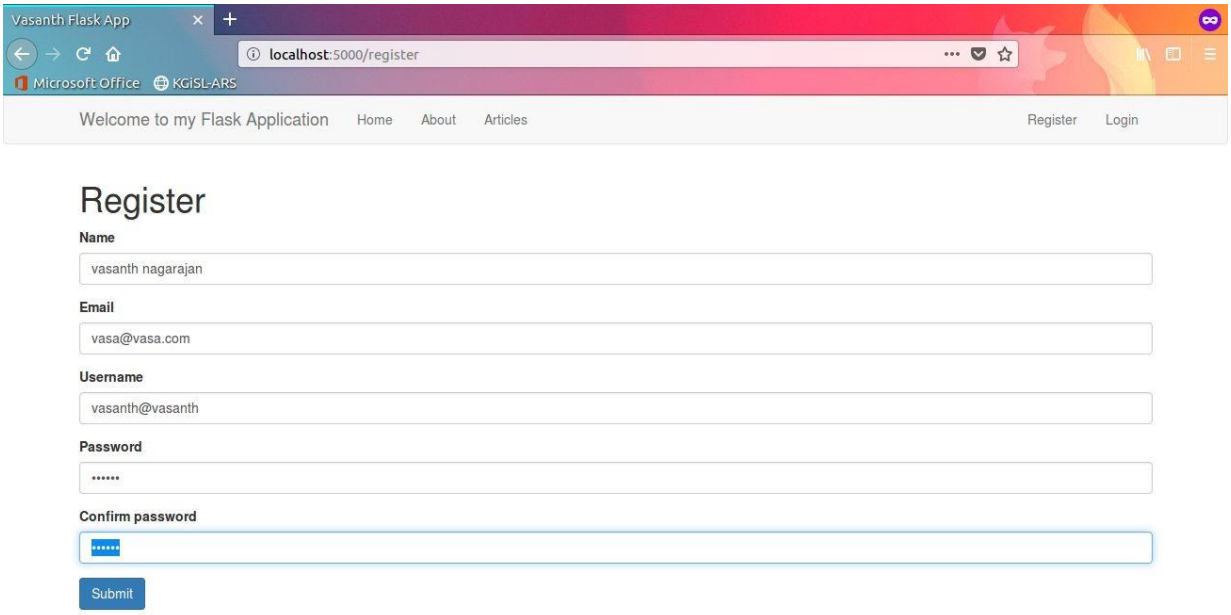
```
<nav class="navbar navbar-default">
        <div class="container">
        <div class="navbar-header">
        <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
data-target="#navbar" aria-expanded="false" aria-controls="navbar">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        </button>
        <a class="navbar-brand" href="#">Welcome to my Flask Application</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
        <li><a href="/">Home</a></li>
        <li><a href="/about">About</a></li>
        <li><a href="/articles">Articles</a></li>
        </ul>

        <ul class="nav navbar-nav navbar-right">
        {% if session.logged_in %}
        <li><a href="/login">Logout</a></li>
        {% else %}
        <li><a href="/register">Register</a></li>
        <li><a href="/login">Login</a></li>
        {% endif %}

        </ul>
        </div><!--/.nav-collapse -->
        </div>
        </nav>
```

Now save the file and run the app.py server and navigate to the browser and now see the application,its look like as shown below :

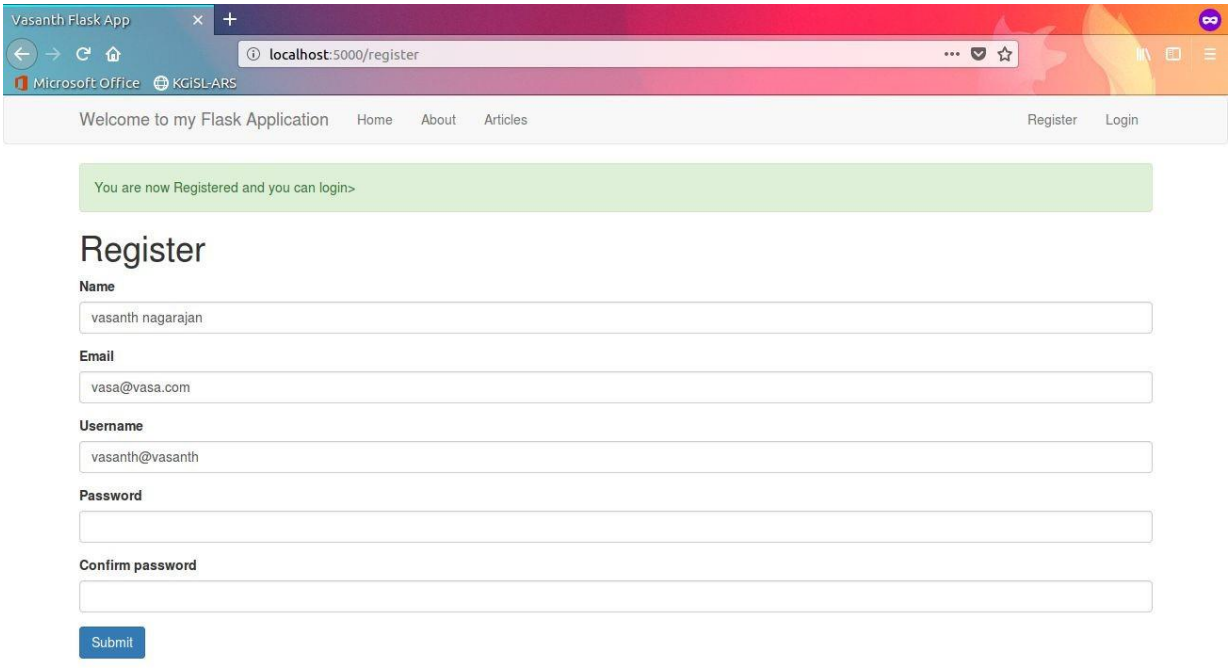
Home page of our application

Register page of our application :

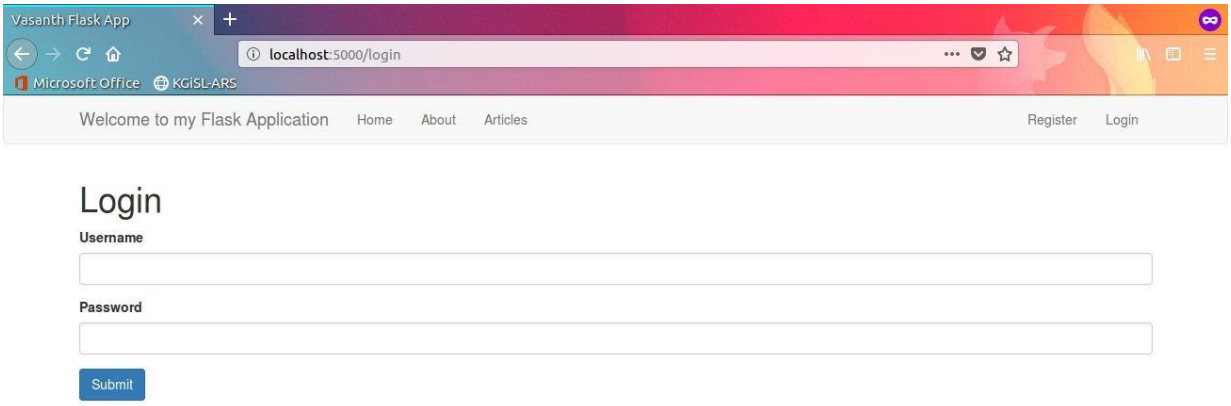




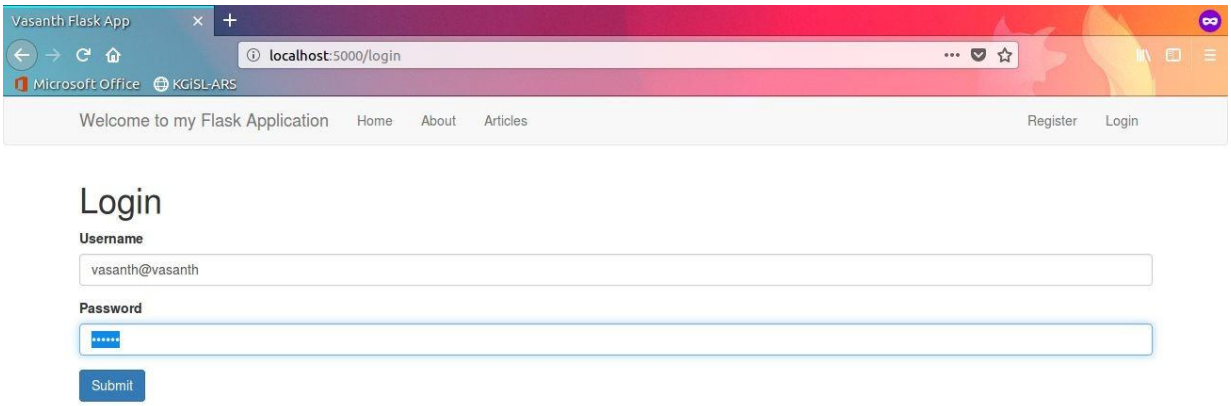
After entering the details and once clicked the submit button ,



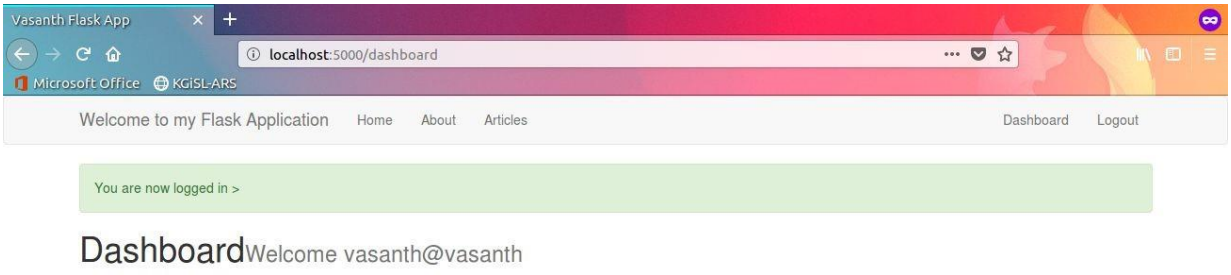
Now we have successfully registered ,and now we are going to login in to our application :



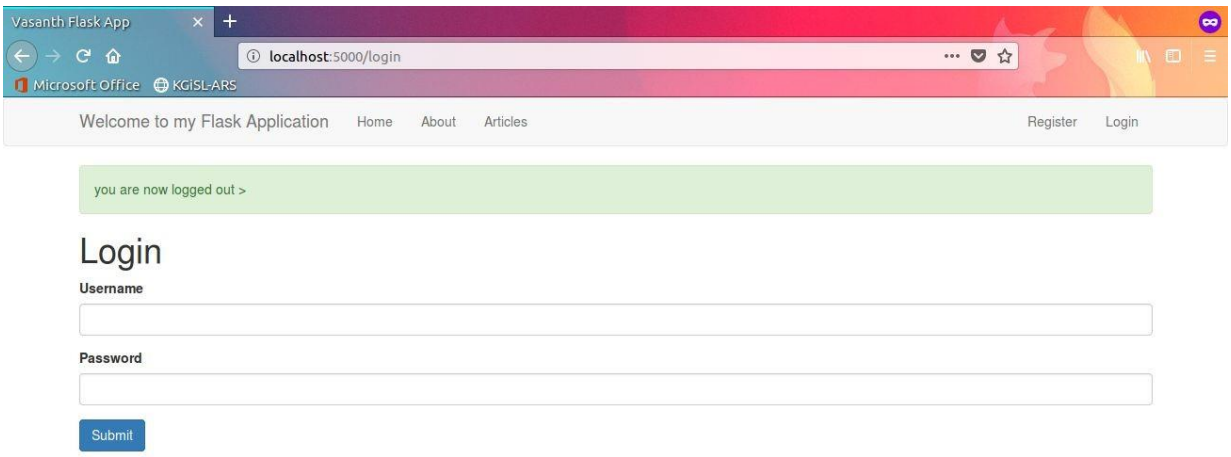
After entering the credentials ,



Once logged in :

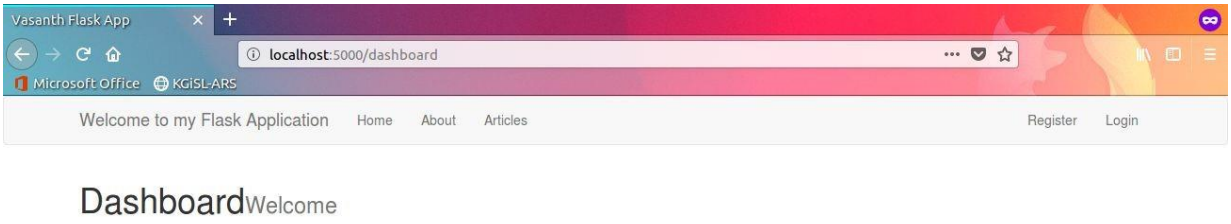


The dashboard displays the user name of the current logged in person and we can see the logout option at the right top of the page .



After clicked the logout button the above screen will be displayed .

But when we type the url directly in the browser the browser will navigate to the exact page that we requested,but this is not fair ,for example when we type the /dashboard in the url ,the browser will take u to the dashboard page automatically ,as shown below :



so we have to use the decorators,for that do the following steps in the program .
Now go to app.py and make the following changes :

from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps

app = Flask(_name_)
app.debug = True

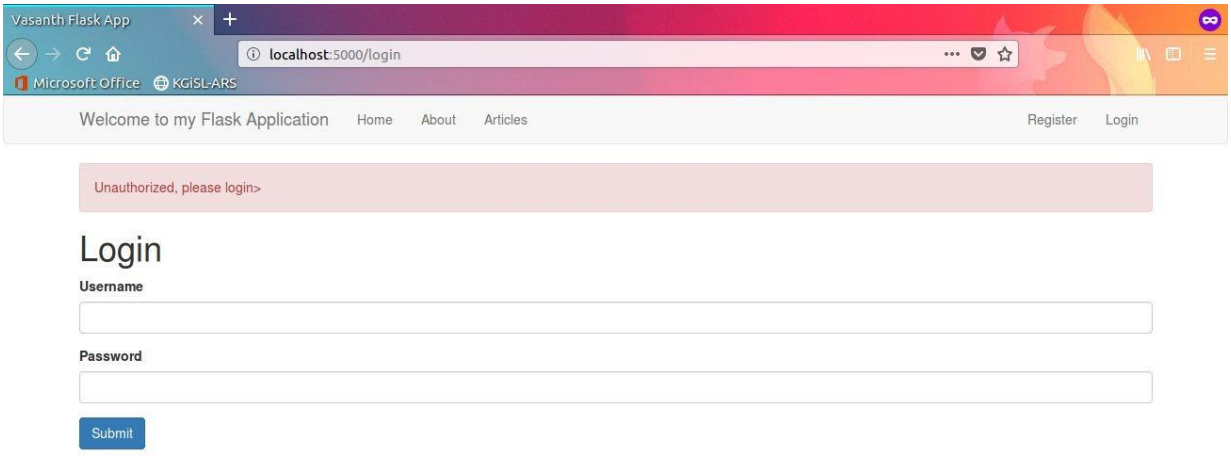
```
#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

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

```
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s"
,[username])
```

```
        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap
```

```
#logout
@app.route('/logout')
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():
        return render_template('dashboard.html')

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

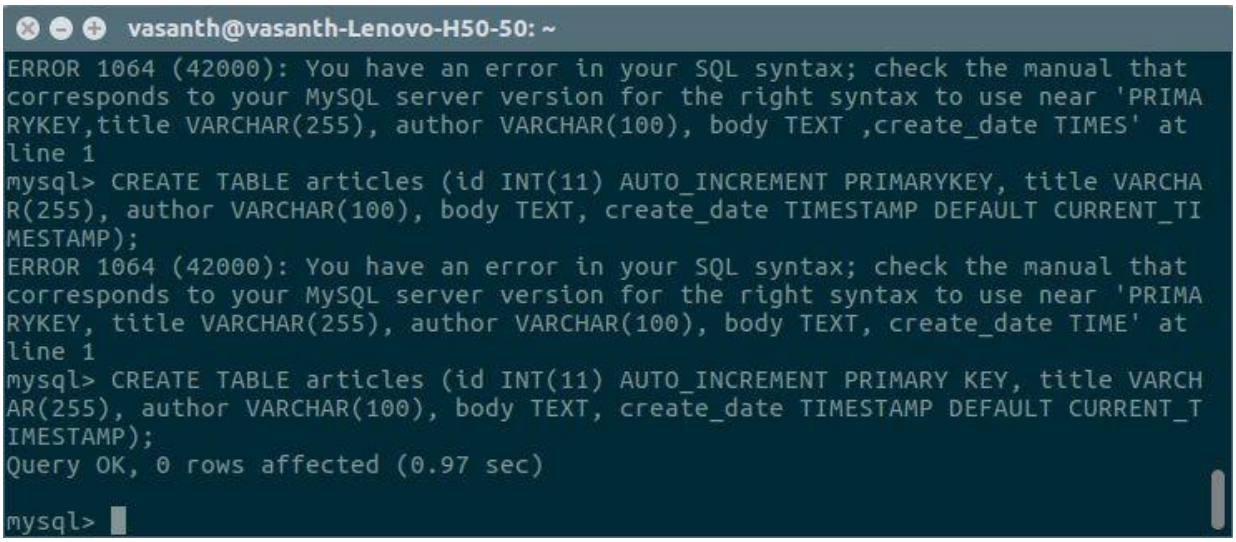
Now save and run the file , you will see the unauthorized message when you to try to navigate to dashboard without login



Now we are going to work on articles :
Still the data for the articles are coming from the file called data.py ,but we need those data to come from the database .

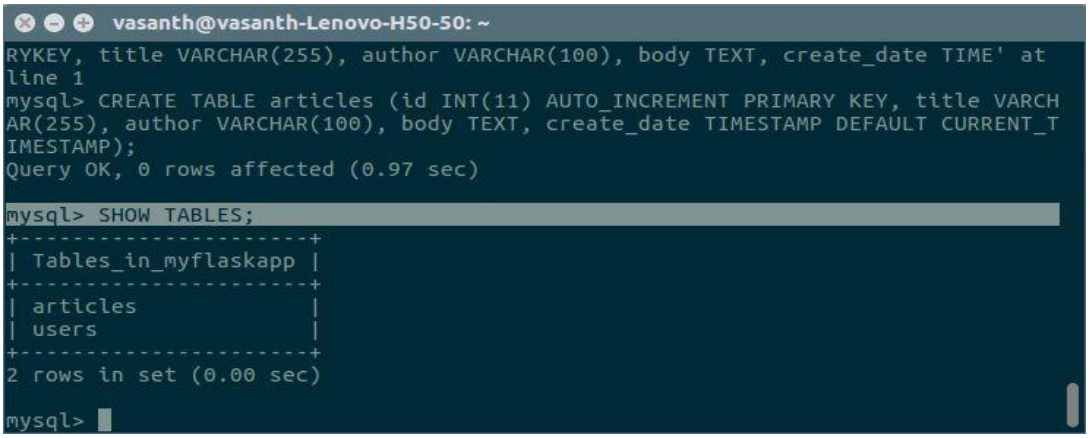
So now go to mysql console ,and now create a table for the articles :

mysql> CREATE TABLE articles (id INT(11) AUTO_INCREMENT PRIMARY KEY, title VARCHAR(255), author VARCHAR(100), body TEXT, create_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP);
Query OK, 0 rows affected (0.97 sec)



Now type the sql query for showing the tables :

mysql> SHOW TABLES;



Now we can add the article functionality ,for that first login in to our application ,the dashboard page will be displayed as shown below :

Now we have to add the article button in the dashboard ,now go to dashboard.html page

Dashboard.html

{% extends 'layout.html' %}

```
{% block body %}
  <h1>Dashboard<small>Welcome {{session.username}}</small></h1>
  <a class="btn btn-success" href="/add_article">Add Articles</a>
  <hr>
{% endblock %}
```

Now we have to add the route inside the app.py so now navigate to app.py and add the following commands in the **app.py**

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
```

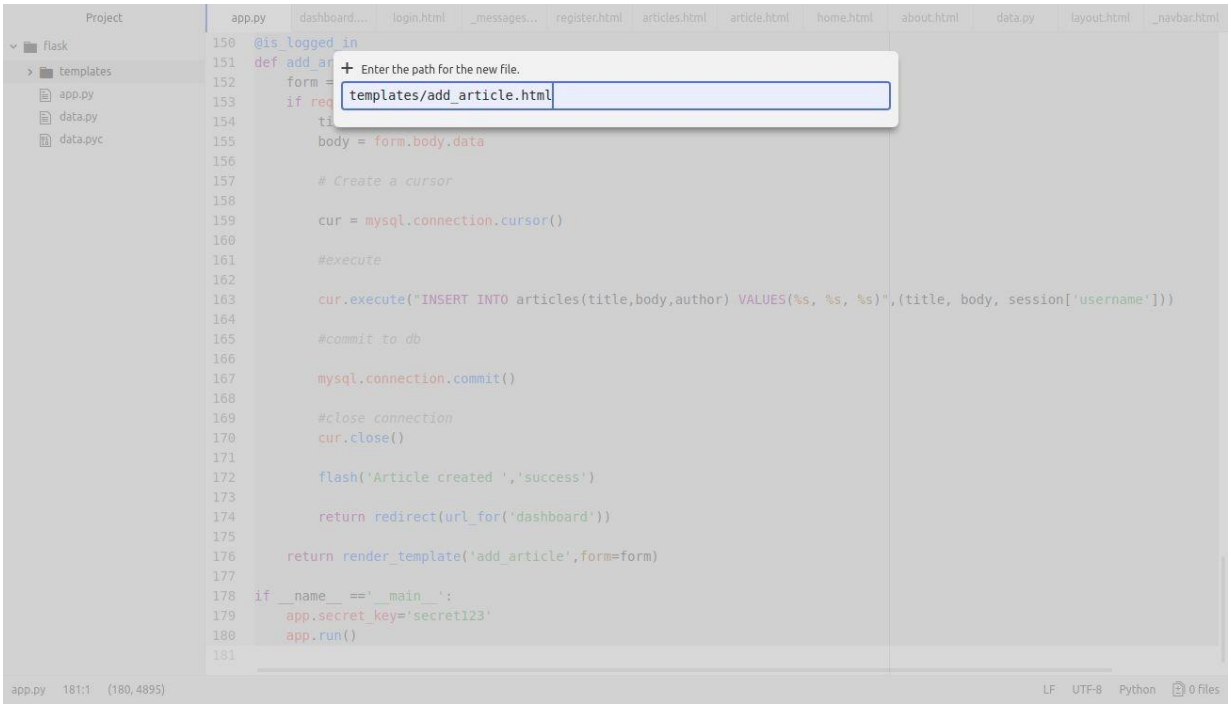
```
def about():
	return render_template('about.html')

@app.route('/articles')
def articles():
	return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
	return render_template('article.html',id=id)

class RegisterForm(Form):
	name = StringField('Name',[validators.Length(min=1,max=50)])
	username = StringField('Username',[validators.Length(min=4,max=25)])
	email = StringField('Email',[validators.Length(min=4,max=25)])
	password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
	confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
	form = RegisterForm(request.form)
	if request.method == 'POST' and form.validate():
	name = form.name.data
	email = form.email.data
	username = form.username.data
	password = sha256_crypt.encrypt(str(form.password.data))

	# Create crusor
	cur = mysql.connection.cursor()

	cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

	# commit to DB
	mysql.connection.commit()
	#close connection
	cur.close()
```

```
        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
```

```
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():
        return render_template('dashboard.html')

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
```

```
        body = TextAreaField('Body',[validators.Length(min=30,max=25)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now we have to create the actual form for that we have to go to templates and create a new file **add_article.html**

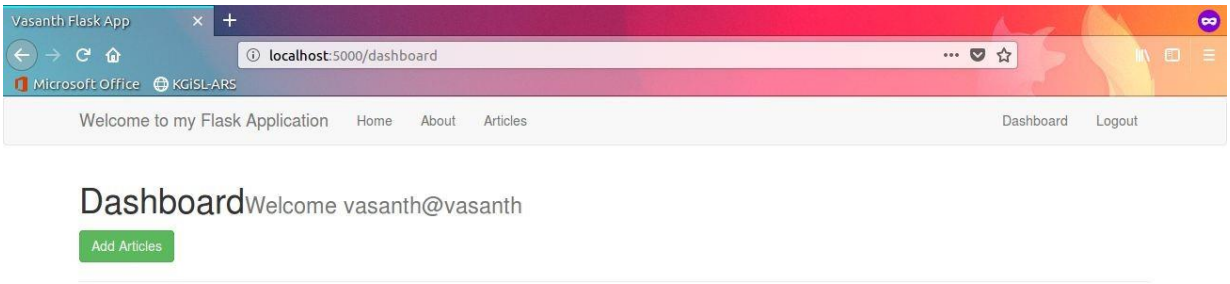


Add_article.html

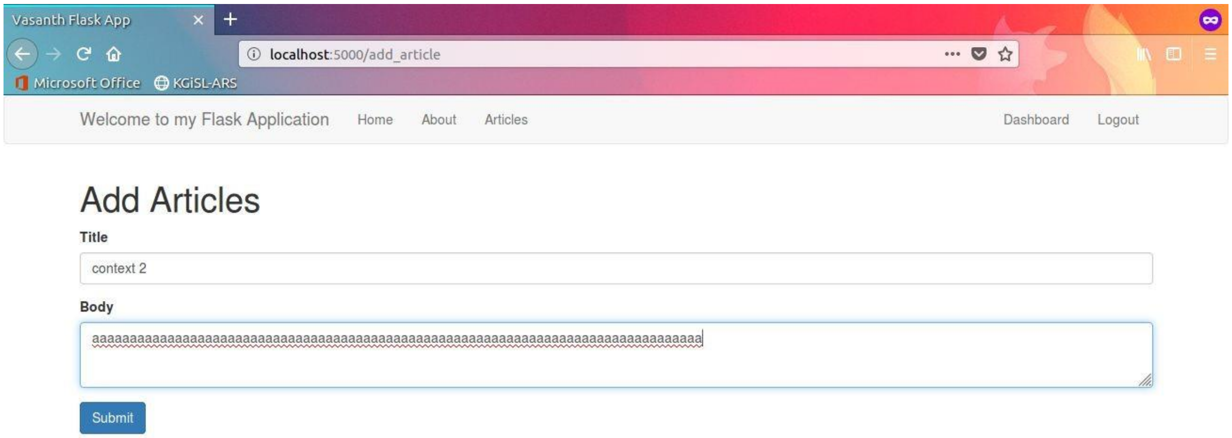
```
{% extends 'layout.html' %}

{% block body %}
        <h1> Add Articles</h1>
        {% from "includes/_formhelpers.html" import render_field %}
        <form method="POST" action="">
        <div class="form-group">
        {{ render_field(form.title,class_="form-control") }}
        </div>
        <div class="form-group">
        {{ render_field(form.body, class_="form-control") }}
        </div>
```

```
<p><input class="btn btn-primary" type="submit" value="Submit">
</form>
{% endblock %}
```

Now type the above code and save those file in the templates .now run the app.py and see the browser



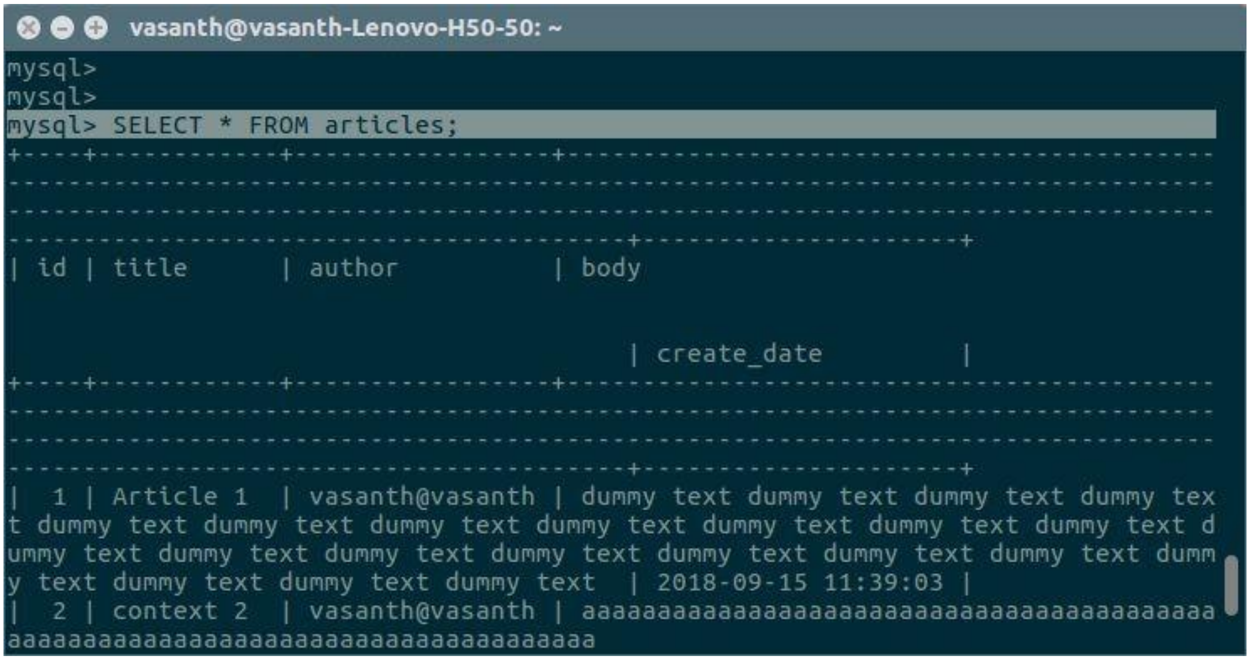
Now click on add article button and type the title and add some dummy text in the article page .



Now click submit ..

Now go to mysql and check the database using the following query :

mysql> SELECT * FROM articles;



Now we are going to add an editor in our page ,its cke editor

Now go to our layout.html file and add the following command in the layout.html file as shown below :

Layout.html:

```
<html>
<head>
  <title>Vasanth Flask App</title>
  <link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>
{% include 'includes/_navbar.html'%}

<div class="container">
  {% include 'includes/_messages.html' %}
  {% block body %}{% endblock %}
</div>
```

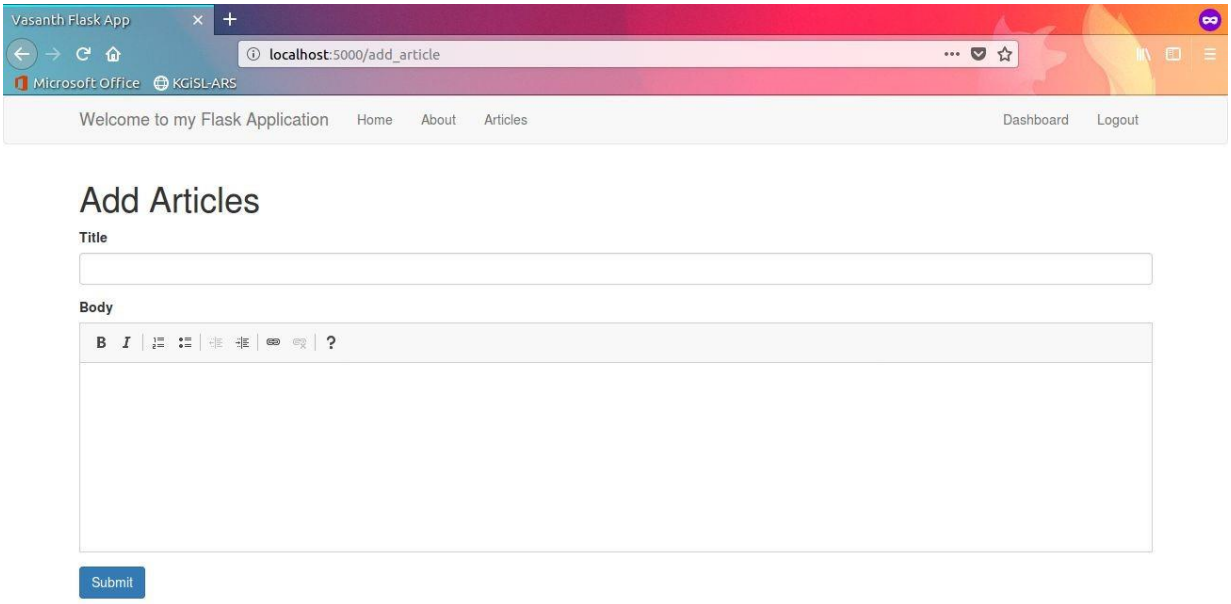
```
<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"</script>
<script src="//cdn.ckeditor.com/4.6.2/basic/ckeditor.js"></script>
<script type="text/jscript">
  CKEDITOR.replace('editor')
</body>
</html>
```

And now go to add article and make the following changes in the add article page as follows :

Add_article.html

```
{% extends 'layout.html' %}

{% block body %}
        <h1> Add Articles</h1>
        {% from "includes/_formhelpers.html" import render_field %}
        <form method="POST" action="">
        <div class="form-group">
        {{ render_field(form.title,class_="form-control") }}
        </div>
        <div class="form-group">
        {{ render_field(form.body, class_="form-control",id="editor") }}
        </div>
        <p><input class="btn btn-primary" type="submit" value="Submit">
        </form>
        {% endblock %}
```

Now save and run the python server ,you can see some formatting options for the editor as shown below .



Now in the dashboard we have to see the articles as we did that before when we clicked the articles as the list of articles contents will be displayed as same we have to do now ,but this time the articles should shown up in the homepage (dashboard).

Now go to app.py ,navigate to the dashboard route and make the following changes and save the **app.py** file

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
```

```
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():
        return render_template('articles.html',articles = Articles)


@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
```

```
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
```

```
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
```

```
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('dashboard.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
```

```
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

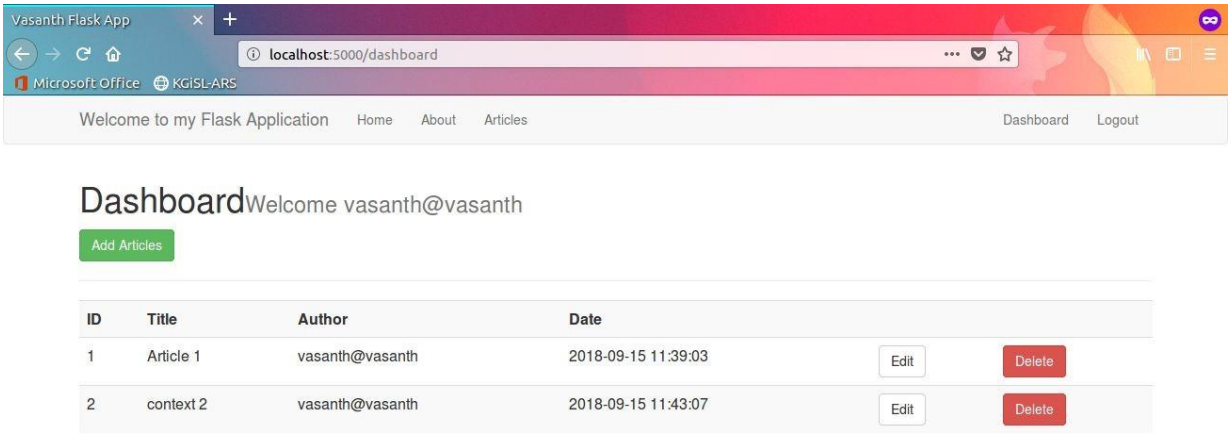
Now navigate to dashboard.html page that's in templates folder and make the following changes :

Dashboard.html

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Dashboard<small>Welcome {{session.username}}</small></h1>
  <a class="btn btn-success" href="/add_article">Add Articles</a>
  <hr>
```

```
 <table class="table table-striped">
        <tr>
        <th>ID</th>
        <th>Title</th>
        <th>Author</th>
        <th>Date</th>
        <th></th>
        <th></th>
        </tr>
        {% for article in articles %}
        <tr>
        <td>{{article.id}}</td>
        <td>{{article.title}}</td>
        <td>{{article.author}}</td>
        <td>{{article.create_date}}</td>
        <td><a href="edit_article/{{article.id}}" class="btn btn-default pullright">Edit</td>
        <td><a href="#" class="btn btn-danger">Delete</a></td>
        {% endfor %}
 </table>
{% endblock %}
```

Now save the dashboard.html file and run the app.py server and navigate to dashboard page in the browser , now your dashboard will display the articles content that are fetched from the database and you can see the edit and delete button as shown .



Now go to app.py and make the following changes in **articles route**

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
#from data import Articles
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
```

```
mysql = MySQL(app)


#Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('articles.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('articles.html',msg=msg)
        #close connection
        cur.close()



@app.route('/article/<string:id>/')
def article(id):
        return render_template('article.html',id=id)

class RegisterForm(Form):
```

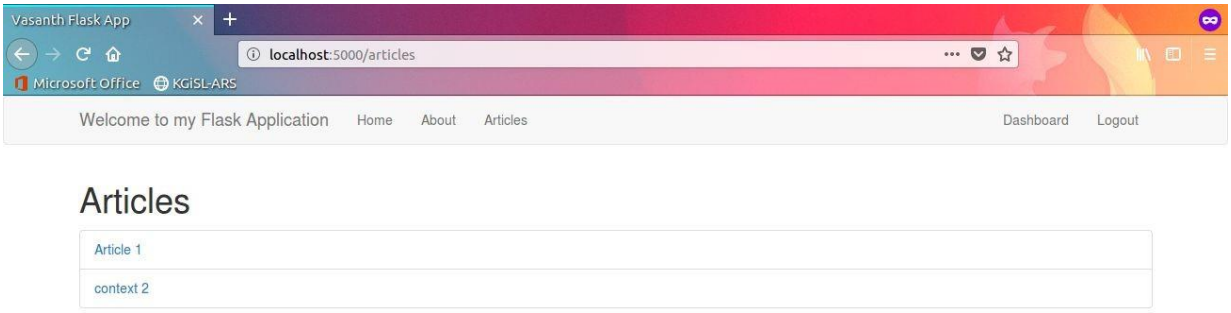
```
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']
```

```
        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
```

```
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('dashboard.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class
```

```
class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now save the app.py and run the python server,now you can see the content inside the articles that are fetched from the database as shown below :



Now we have make some other changes in the app.py lets do that ! ,

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
```

```
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


#Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('articles.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('articles.html',msg=msg)
        #close connection
        cur.close()



@app.route('/article/<string:id>/')
```

```
def article(id):
        #create cursor
        cur = mysql.connection.cursor()

        #get article
        result = cur.execute("SELECT * FROM articles WHERE id= %s",[id])

        article = cur.fetchone()

        return render_template('article.html',article=article)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()
```

```
        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()
```

```
        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")
```

```
        articles = cur.fetchall()

        if result > 0:
        return render_template('dashboard.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()
```

```
        #close connection
        cur.close()

        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

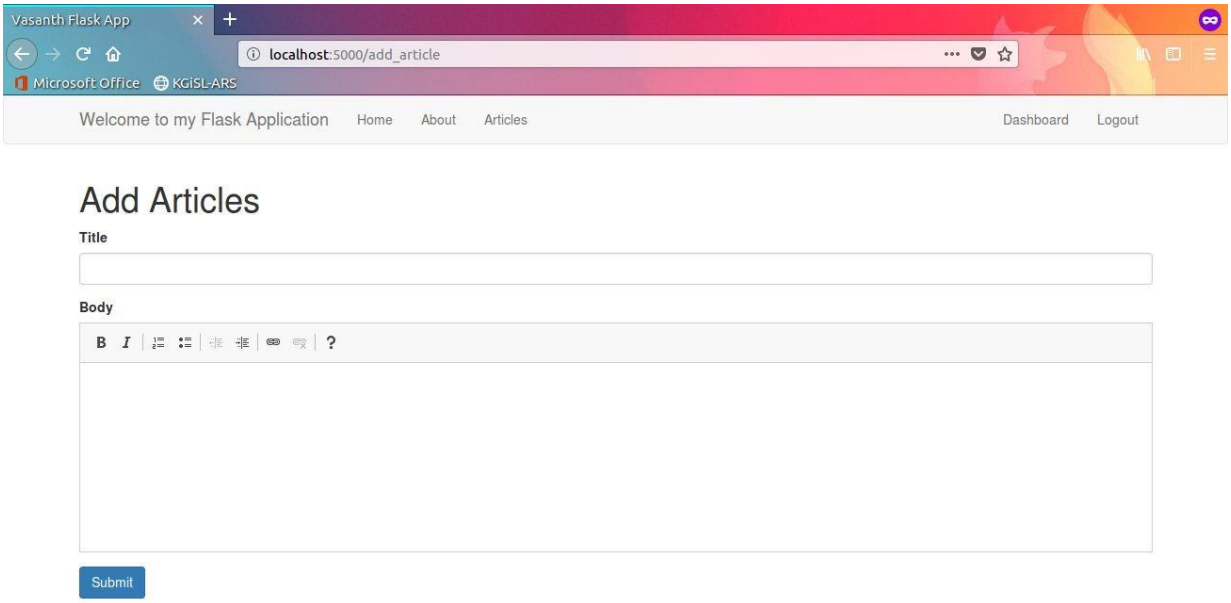
After making the changes in the app.py file ,now go to **article.html** file and add the following code :

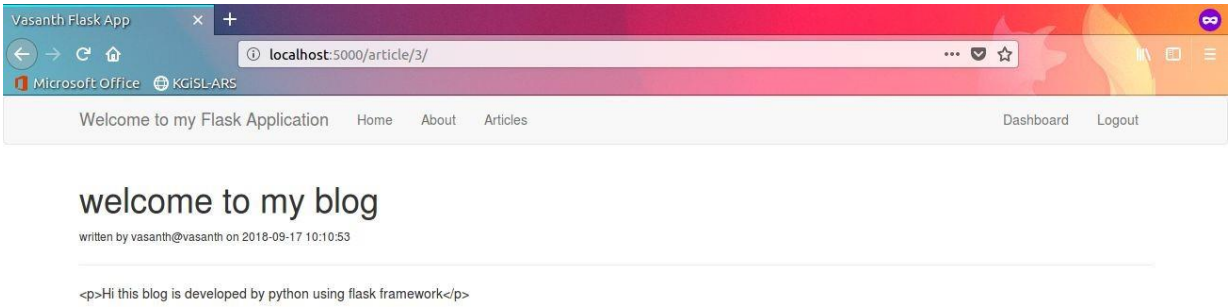
Article.html

```
{% extends 'layout.html'%}

{% block body %}
<h1>{{article.title}}</h1>
<small> Written by {{artilce.author}} on {{article.create_date}}</small>
<hr>
<div>
  {{article.body | safe}}
</div>
{% endblock %}
```

Now save the article.html file and reload the article tab in the browser now the contents are displayed as shown below :

Now to test it out ,go to dashboard and add a new article by clicking add article button as shown below :





Now we have to add the button in the dashboard ,now go to home.html

```
{% extends 'layout.html' %}

{% block body %}
<div class="jumbotron text-center">
  <h1>Vasanth Python Tutorials</h1>
  <p class="lead">This Application is build using python and flask framework</p>
  {% if session.logged_in == NULL %}
          <a href="/register" class="btn btn-primary btn-lg">Register</a>
          <a href="/login" class="btn btn-success btn-lg">Login</a>
  {% endif %}
</div>
{% endblock %}
```

Now save the file and run the app.py file ,

Logout the page if you are logged in ,now go to home page ,you will see the two button as shown below :

Now we have to work on the editing of the articles , now go to app.py file and make the following changes .

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
```

```
mysql = MySQL(app)


#Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('articles.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('articles.html',msg=msg)
        #close connection
        cur.close()



@app.route('/article/<string:id>/')
def article(id):
        #create cursor
        cur = mysql.connection.cursor()
```

```
    #get article
    result = cur.execute("SELECT * FROM articles WHERE id = %s",[id])

    article = cur.fetchone()

    return render_template('article.html',article=article)

class RegisterForm(Form):
    name = StringField('Name',[validators.Length(min=1,max=50)])
    username = StringField('Username',[validators.Length(min=4,max=25)])
    email = StringField('Email',[validators.Length(min=4,max=25)])
    password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
    confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
    form = RegisterForm(request.form)
    if request.method == 'POST' and form.validate():
    name = form.name.data
    email = form.email.data
    username = form.username.data
    password = sha256_crypt.encrypt(str(form.password.data))

    # Create crusor
    cur = mysql.connection.cursor()

    cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

    # commit to DB
    mysql.connection.commit()
    #close connection
    cur.close()

    flash("You are now Registered and you can login" , 'success')

    redirect(url_for('login'))
    return render_template('register.html',form=form)
```

```
# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)
```

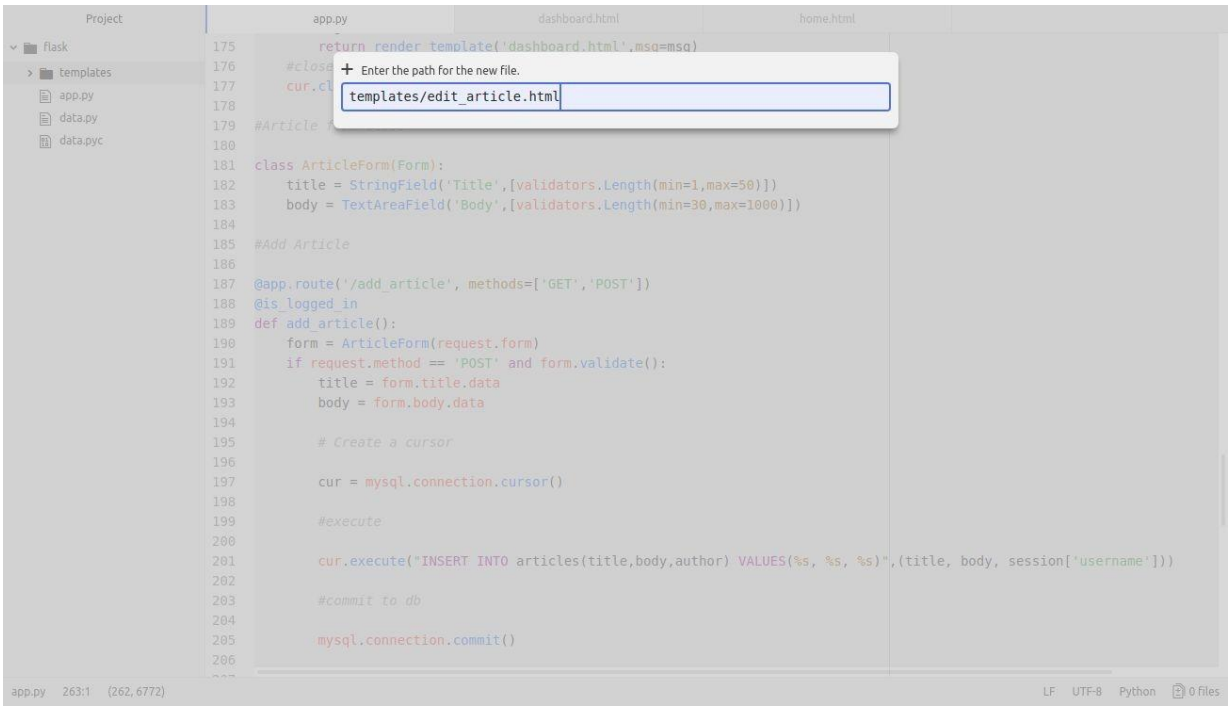
```
        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('dashboard.html',articles=articles)
```

```
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article created ','success')
```

```
        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

#Edit Article

@app.route('/edit_article/<string:id>', methods=['GET','POST'])
@is_logged_in
def edit_article(id):
        # Create cursor
        cur = mysql.connection.cursor()
        #get article by id
        result = cur.execute("SELECT * FROM articles WHERE id = %s", [id])

        article = cur.fetchone()

        #get form
        form = ArticleForm(request.form)

        #populate article form fields
        form.title.data = article['title']
        form.title.data = article['body']

        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("UPDATE articles SET name=%s, body=%s WHERE id = %s" ,
(name,body))

        #commit to db
```

```
        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article Updated ','success')

        return redirect(url_for('dashboard'))

        return render_template('edit_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

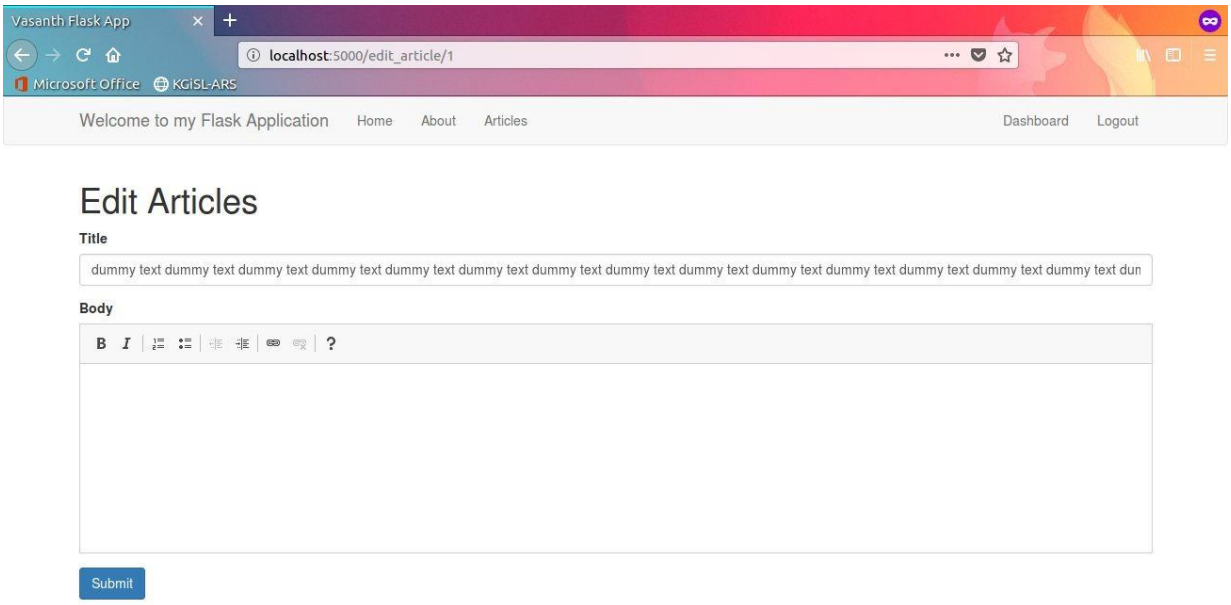
After making the above changes in the app.py file save the file and now go to templates and create a new file as edit_article.html as shown below :



Now type the following code in the edit_article.html as shown below :

```
{% extends 'layout.html' %}

{% block body %}
  <h1> Edit Articles</h1>
  {% from "includes/_formhelpers.html" import render_field %}
  <form method="POST" action="">
        <div class="form-group">
        {{ render_field(form.title,class_="form-control") }}
        </div>
        <div class="form-group">
        {{ render_field(form.body, class_="form-control",id="editor") }}
        </div>
        <p><input class="btn btn-primary" type="submit" value="Submit">
        </form>
  {% endblock %}
```

Now save the file and now execute the app.py file ,now you can edit the following content as shown below :



And now we have to make some changes in the **app.py** file as shown below :

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


#Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")
```

```
    articles = cur.fetchall()

    if result > 0:
    return render_template('articles.html',articles=articles)
    else:
    msg = 'No Articles Found'
    return render_template('articles.html',msg=msg)
    #close connection
    cur.close()



@app.route('/article/<string:id>/')
def article(id):
    #create cursor
    cur = mysql.connection.cursor()

    #get article
    result = cur.execute("SELECT * FROM articles WHERE id = %s",[id])

    article = cur.fetchone()

    return render_template('article.html',article=article)

class RegisterForm(Form):
    name = StringField('Name',[validators.Length(min=1,max=50)])
    username = StringField('Username',[validators.Length(min=4,max=25)])
    email = StringField('Email',[validators.Length(min=4,max=25)])
    password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
    confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
    form = RegisterForm(request.form)
    if request.method == 'POST' and form.validate():
    name = form.name.data
    email = form.email.data
```

```
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')

        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']
```

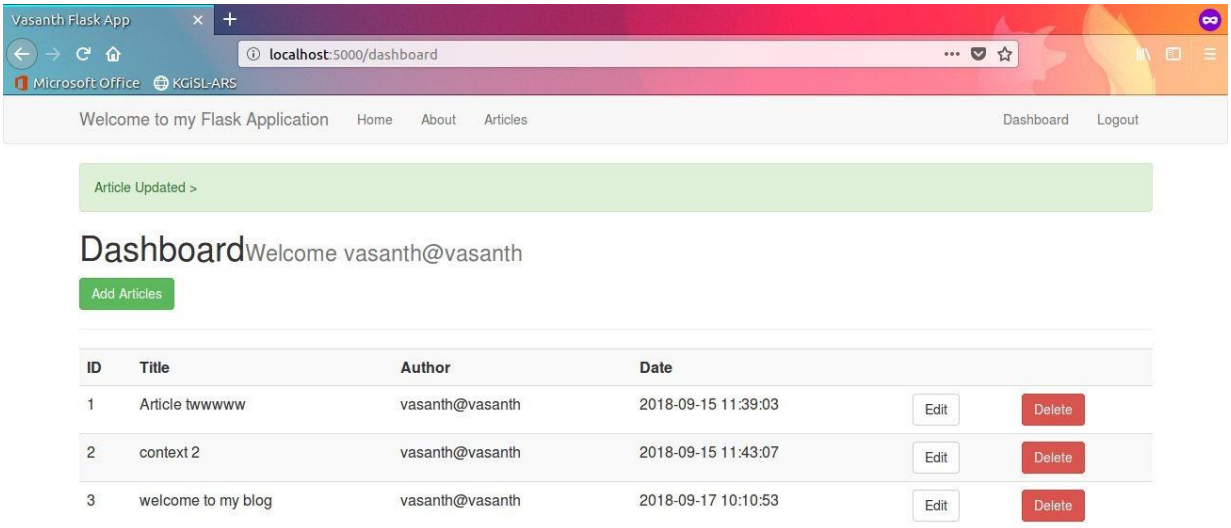
```
        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
        error = 'Username not found'
        return render_template('login.html',error=error)

        return render_template('login.html')

#check if user logged in

def is_logged_in(f):
        @wraps(f)
        def wrap(*args,**kwargs):
        if 'logged_in' in session:
        return f(*args, **kwargs)
        else:
        flash('Unauthorized, please login','danger')
        return redirect(url_for('login'))
        return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
```

```
        session.clear()
        flash('you are now logged out ','success')
        return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('dashboard.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data
```

```
        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

#Edit Article

@app.route('/edit_article/<string:id>', methods=['GET','POST'])
@is_logged_in
def edit_article(id):
        # Create cursor
        cur = mysql.connection.cursor()
        #get article by id
        result = cur.execute("SELECT * FROM articles WHERE id = %s", [id])

        article = cur.fetchone()

        #get form
        form = ArticleForm(request.form)

        #populate article form fields
```

```
        form.title.data = article['title']
        form.body. data = article['body']

        if request.method == 'POST' and form.validate():
        title = request.form['title']
        body = request.form['body']

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("UPDATE articles SET title=%s, body=%s WHERE id = %s" ,
(title,body,id))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article Updated ','success')

        return redirect(url_for('dashboard'))

        return render_template('edit_article.html',form=form)

if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now save the app.py file and refresh the browser and edit the articles and make some changes and click submit now we can see that we can edit the content of the article and we can make the changes that reflect in the database as shown



Now we want to create the functionality for the delete button now go to app.py and make the following changes in the file :

App.py

```
from flask import Flask,render_template, flash, redirect , url_for , session ,request, logging
from flask_mysqldb import MySQL
from wtforms import Form, StringField , TextAreaField ,PasswordField , validators
from passlib.hash import sha256_crypt
from functools import wraps


app = Flask(_name_)
app.debug = True


#Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'myflaskapp'
```

```
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
#init MYSQL
mysql = MySQL(app)


#Articles = Articles()

@app.route('/')
def index():
        return render_template('home.html')

@app.route('/about')
def about():
        return render_template('about.html')

@app.route('/articles')
def articles():

        #create cursor
        cur = mysql.connection.cursor()

        #get articles
        result = cur.execute("SELECT * FROM articles")

        articles = cur.fetchall()

        if result > 0:
        return render_template('articles.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('articles.html',msg=msg)
        #close connection
        cur.close()



@app.route('/article/<string:id>/')
def article(id):
        #create cursor
```

```
        cur = mysql.connection.cursor()

        #get article
        result = cur.execute("SELECT * FROM articles WHERE id = %s",[id])

        article = cur.fetchone()

        return render_template('article.html',article=article)

class RegisterForm(Form):
        name = StringField('Name',[validators.Length(min=1,max=50)])
        username = StringField('Username',[validators.Length(min=4,max=25)])
        email = StringField('Email',[validators.Length(min=4,max=25)])
        password = PasswordField('Password', [ validators.DataRequired
(),validators.EqualTo('confirm',message ='passwords do not match')])
        confirm = PasswordField('Confirm password')

@app.route('/register', methods=['GET','POST'])
def register():
        form = RegisterForm(request.form)
        if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        username = form.username.data
        password = sha256_crypt.encrypt(str(form.password.data))

        # Create crusor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(name,email,username,password)
VALUES(%s,%s,%s,%s)",(name,email,username,password))

        # commit to DB
        mysql.connection.commit()
        #close connection
        cur.close()

        flash("You are now Registered and you can login" , 'success')
```

```
        redirect(url_for('login'))
        return render_template('register.html',form=form)

# user login
@app.route('/login',methods =['GET','POST'])
def login():
        if request.method == 'POST':
        #Get Form Fields
        username = request.form['username']
        password_candidate = request.form['password']

        # Create cursor

        cur = mysql.connection.cursor()

        #Get user by username

        result = cur.execute("SELECT * FROM users WHERE username = %s" ,[username])

        if result > 0:
        # Get Stored hash
        data = cur.fetchone()
        password = data['password']

        # Compare Passwords
        if sha256_crypt.verify(password_candidate,password):
                #Passed
                session['logged_in'] = True
                session['username'] = username

                flash('You are now logged in ','success')
                return redirect(url_for('dashboard'))
        else:
                error = 'Username not found'
                return render_template('login.html',error=error)
                #close connection
        cur.close()

        else:
```

```
            error = 'Username not found'
            return render_template('login.html',error=error)

            return render_template('login.html')

#check if user logged in

def is_logged_in(f):
            @wraps(f)
            def wrap(*args,**kwargs):
            if 'logged_in' in session:
            return f(*args, **kwargs)
            else:
            flash('Unauthorized, please login','danger')
            return redirect(url_for('login'))
            return wrap



#logout
@app.route('/logout')
@is_logged_in
def logout():
            session.clear()
            flash('you are now logged out ','success')
            return redirect(url_for('login'))
# Dashboard
@app.route('/dashboard')
@is_logged_in
def dashboard():

            #create cursor
            cur = mysql.connection.cursor()

            #get articles
            result = cur.execute("SELECT * FROM articles")

            articles = cur.fetchall()
```

```
        if result > 0:
        return render_template('dashboard.html',articles=articles)
        else:
        msg = 'No Articles Found'
        return render_template('dashboard.html',msg=msg)
        #close connection
        cur.close()

#Article form class

class ArticleForm(Form):
        title = StringField('Title',[validators.Length(min=1,max=50)])
        body = TextAreaField('Body',[validators.Length(min=30,max=1000)])

#Add Article

@app.route('/add_article', methods=['GET','POST'])
@is_logged_in
def add_article():
        form = ArticleForm(request.form)
        if request.method == 'POST' and form.validate():
        title = form.title.data
        body = form.body.data

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("INSERT INTO articles(title,body,author) VALUES(%s, %s, %s)",(title,
body, session['username']))

        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()
```

```
        flash('Article created ','success')

        return redirect(url_for('dashboard'))

        return render_template('add_article.html',form=form)

#Edit Article

@app.route('/edit_article/<string:id>', methods=['GET','POST'])
@is_logged_in
def edit_article(id):
        # Create cursor
        cur = mysql.connection.cursor()
        #get article by id
        result = cur.execute("SELECT * FROM articles WHERE id = %s", [id])

        article = cur.fetchone()

        #get form
        form = ArticleForm(request.form)

        #populate article form fields
        form.title.data = article['title']
        form.body. data = article['body']

        if request.method == 'POST' and form.validate():
        title = request.form['title']
        body = request.form['body']

        # Create a cursor

        cur = mysql.connection.cursor()

        #execute

        cur.execute("UPDATE articles SET title=%s, body=%s WHERE id = %s" ,
(title,body,id))
```

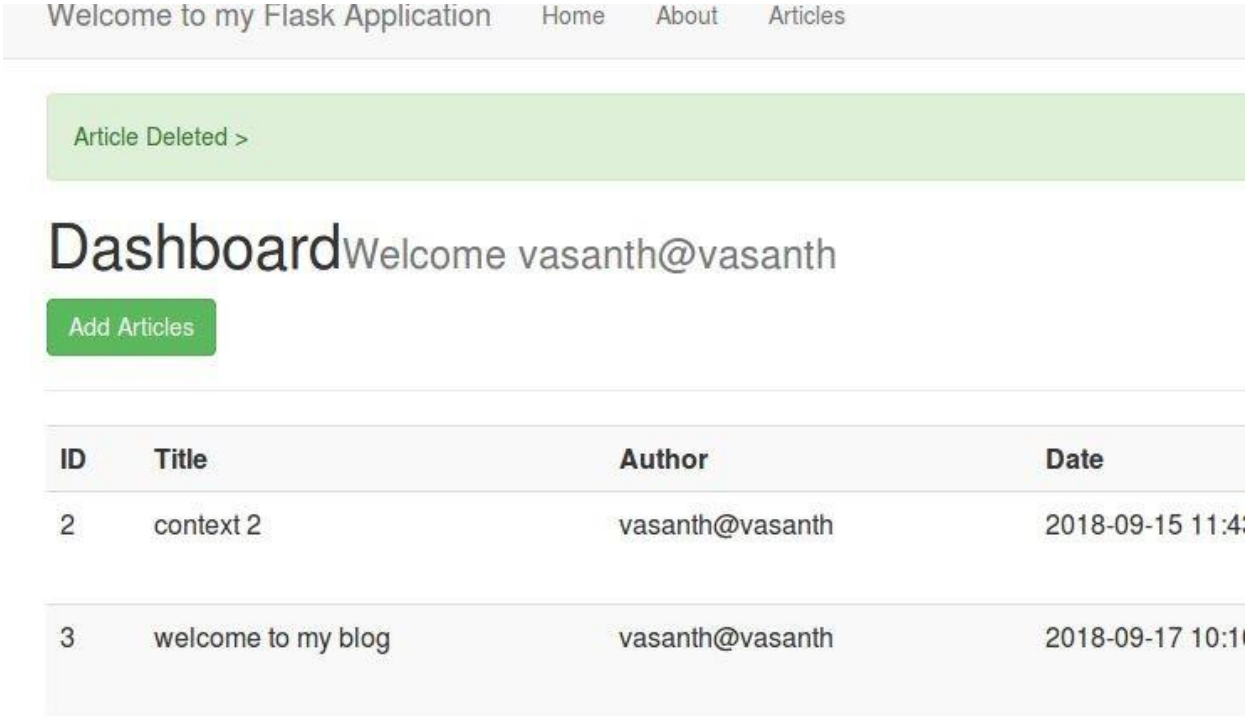
```
        #commit to db

        mysql.connection.commit()

        #close connection
        cur.close()

        flash('Article Updated ','success')

        return redirect(url_for('dashboard'))

        return render_template('edit_article.html',form=form)

#Delete article
@app.route('/delete_article/<string:id>', methods=['POST'])
@is_logged_in
def delete_article(id):
        # Create cursor
        cur = mysql.connection.cursor()

        #Execute
        cur.execute("DELETE FROM articles WHERE id = %s",[id])

        #Commit to DB

        mysql.connection.commit()
        #close connection

        cur.close()

        flash('Article Deleted ','success')

        return redirect(url_for('dashboard'))



if __name__=='_main_':
        app.secret_key='secret123'
        app.run()
```

Now go to dashboard.html and make the following changes :

Dashboard.html :

```
{% extends 'layout.html' %}

{% block body %}
  <h1>Dashboard<small>Welcome {{session.username}}</small></h1>
  <a class="btn btn-success" href="/add_article">Add Articles</a>
  <hr>
  <table class="table table-striped">
          <tr>
          <th>ID</th>
          <th>Title</th>
          <th>Author</th>
          <th>Date</th>
          <th></th>
          <th></th>
          </tr>
          {% for article in articles %}
          <tr>
          <td>{{article.id}}</td>
          <td>{{article.title}}</td>
          <td>{{article.author}}</td>
          <td>{{article.create_date}}</td>
          <td><a href="edit_article/{{article.id}}" class="btn btn-default pullright">Edit</td>
          <td>
          <form action="{{url_for('delete_article',id=article.id)}}" method="post">
          <input type="hidden" name="_method" value="DELETE">
          <input type="submit" value="Delete" class="btn btn-danger">
          </form>
          </td>
          </tr>
          {% endfor %}
  </table>
{% endblock %}
```

And now save the file and execute the app.py file server now you can delete the post that already present in the database ,thus we can add and delete the post in the articles .



Thus we have successfully created a web application using the python and flask .

If you need the source code for this mail me , will send you the entire source code for this :
Mail id :
vasanth.n@kgisl.com
nvasanthnagarajan@gmail.com

If you found any errors in this book,please free to inform me so that i can make those changes .

Thankyou

Vasanth Nagarajan


Thank
you!