Odd Semester (2022)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Individual Work)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |  |
| **Student Information**: **Surname** | | | | | **Given Names**    **Acheadeth** | | **Student ID Number**  **2301931384** | |  |
| 1. | | **Lay** |  | |  |
|  |  |  |
| **Course Code** | **: COMP6502** |  |  | | **Course Name** | | **: Introduction to Programming** | |  |
| **Class** | **: L1CC** |  |  | | **Name of Lecturer(s)** | | **: bagus kerthyayana manuaba** | |  |
|  |  |  |  | |  | |  | |  |
| **Major** | **: CS** |  |  | |  | |  | |  |
| **Title of Assignment**  (if any) | : | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |  |
| **Due Date** | **: 14-01-18** |  |  | | **Submission Date** | | **: 14-01-18** | |  |

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer’s instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

# Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

# Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student: (Name of Student)

1. Lay Acheadeth

**Name : Lay Acheadeth**

**ID : 2301931384**

1. **Description**

**The function of this program:**

Basically, this is a website. It’s a blog website which admin or owner can create as many blogs as they want without having to go through the html code but just go through the admin and fulfill the content of the blog and it will automatically display on the site. It’s a dynamic’s website, where you can post, edit, update, delete, basically take control 100% of your site without having to modify the html code which is kind of stress and require skill in this field.

In addition, each blog has the date of the posts, the producer, and article. Furthermore, user can also add some comment to the blog and admin can decide whether to moderate the comment or not. In case, the comment is insulting or something, we can reject it.

**II. Explanation of Each Function Inside the Class**

*app/models.py*

* **\_\_str\_\_(self) :**
  + **To convert python object into a string , returning a string object when print or \_\_STR\_\_ is invoked.**
  + **It return self.title, means the title here is now a string object not inside the class anymore.**

**. Class Meta**

**. It’s a class container that attach model built-in tools.** **It defines such things as available permissions, associated database table name, whether the model is abstract or not, singular and plural versions of the name ,ordering,etc.**

**In here, it does show the behavior that all the post should be display reversely by(-created-on function). It means the last post is at the top just like facebook and other app order.**

**. def** approve\_comments(self, request, queryset):

**. It’s a function to approve the data that has been sent to database to display out.(update the database). Here, it’s updated when we activate the comment.**

**III.Algorithm**

**1**.Open pycharm

2. Create a newproject(new virtual environment)

3. install Django using pip install django

4. startproject using django-admin startproject ‘name of the project’

5. enter project directory using cd ‘name of the project’

6. run python manage.py runserver to see if we get access to Django framework

7. after accessing start new app using python manage.py startapp ‘name of the app’

8. create urls. Py inside your app

9.Go to the view.py, create a function call home and type the following

def home(request):

return render(request,’home.html’)

10. type the following in the urls.py

path('', views.home,name='home-page'),

11. Create a template inside the app and inside the templates create another directory and inside that directory create html. File which name is home. Then, in the body part type ‘hello world’ between <p> tags.

12. Go to setting.py in the project and in install app, type ‘app(names).apps..(name from app.py class)’

13. then urls.py in project type the following

**from** django.contrib **import** admin  
**from** django.urls **import** path,include  
  
urlpatterns = [  
 *# admin path* path(**'admin/'**, admin.site.urls),  
 *#app path* path(**''**,include(**'app.urls'**)),  
]

14. Then type python manage.py runserver again and you will receive hello world on your homepage.

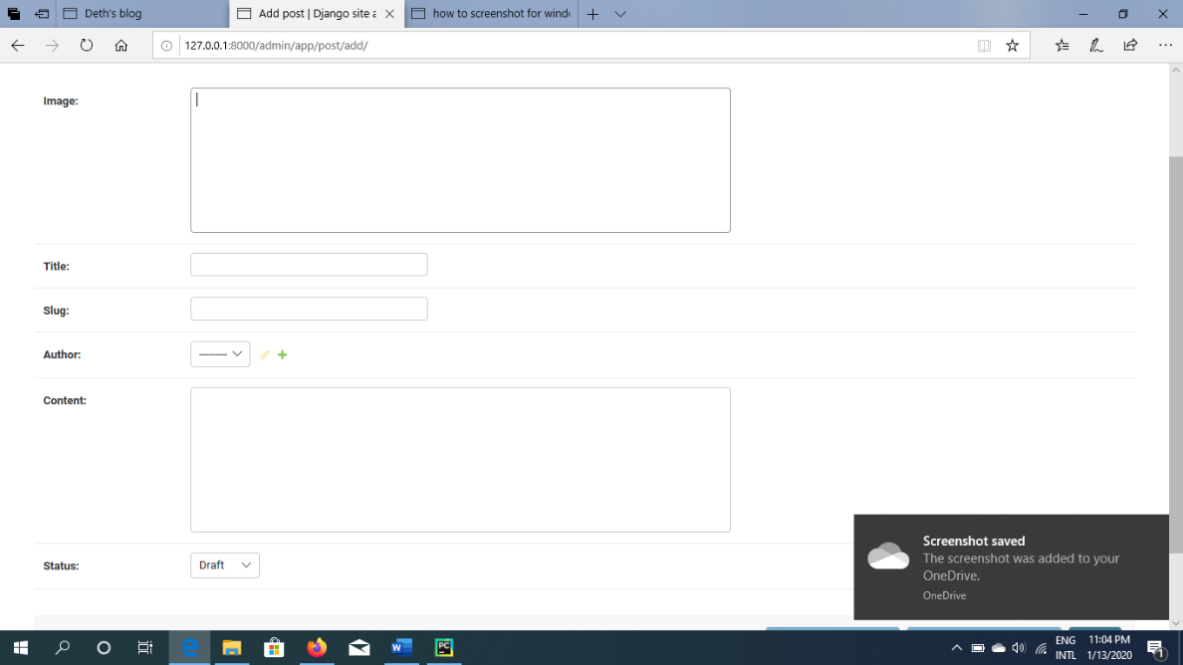
15. Creating a class in models.py( go to code to see those class)

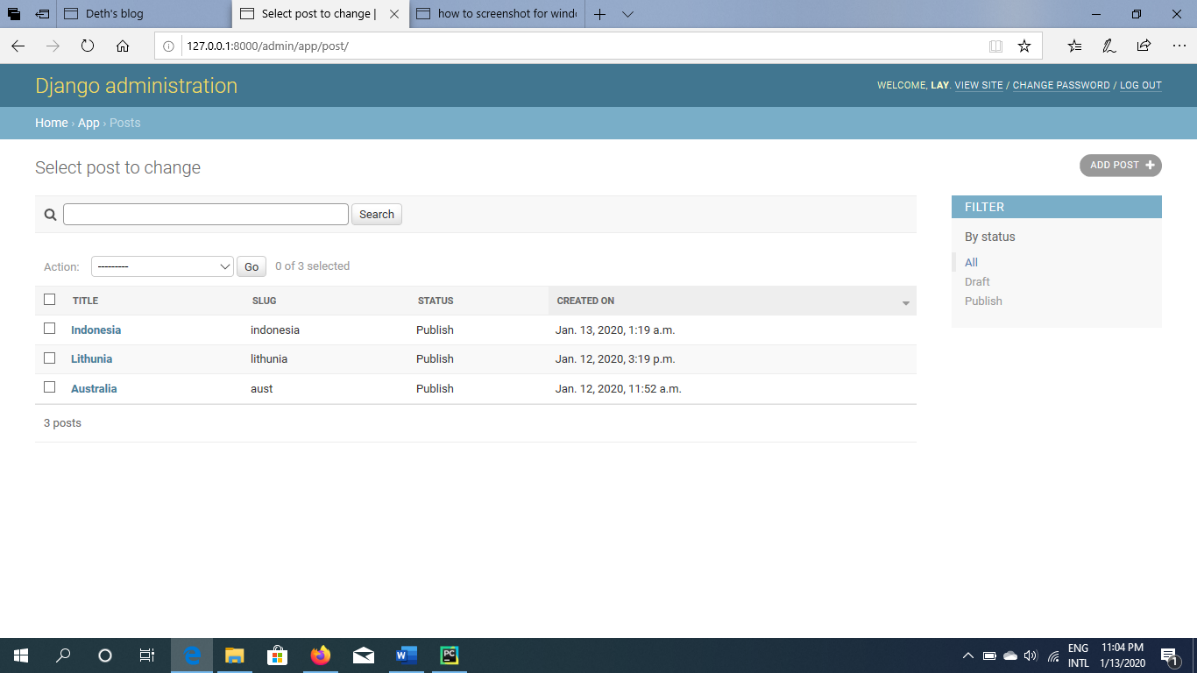
16. Creating a forms.py then type the following(go to the code)

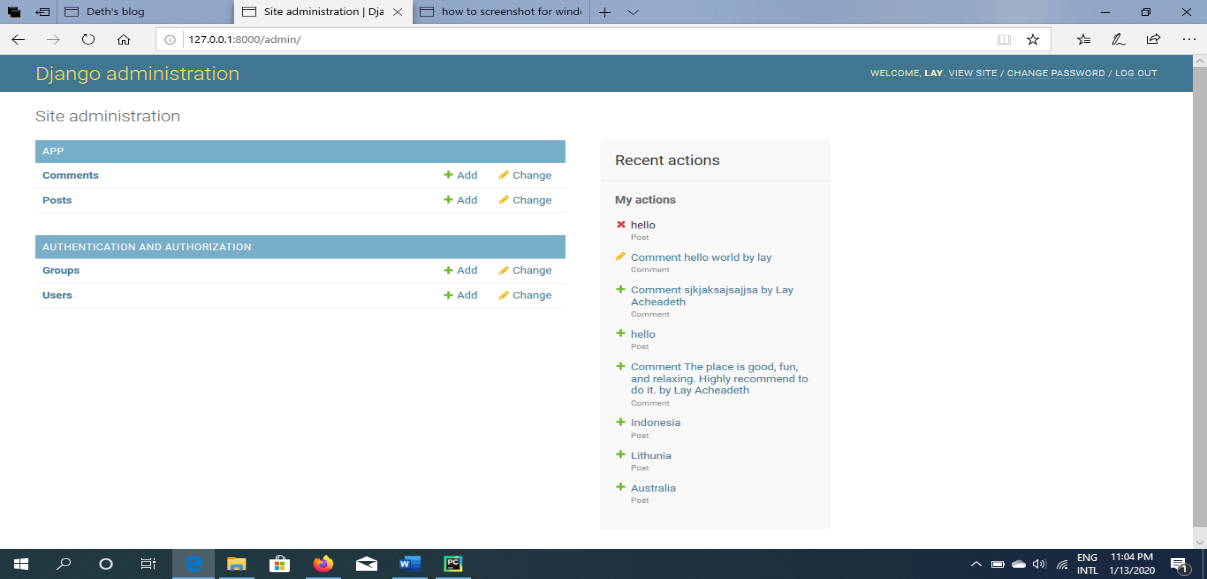
18. Then in the admin.py type(go to the code to see all)

19. In the views, type the code( go to the code to see)

20. Finally, create html code and accessing the database from models.py to the code and then you can later on try posting a blog in admin,site without having to code html again.







**Class Diagram**

|  |
| --- |
| **Deth’s blog** |
| **- Post(models.Model)**  **- Comment(models.Model)**  **- blog(models.Model)**  **- CommentAdmin(admin.ModelAdmin)**  **- PostAdmin(admin.ModelAdmin)** |
| **- post\_detail(request,slug)**  **- about\_us(request)**  **- approve\_comments(self, request, queryset)**  **- \_\_str(self)** |
|  |

**IV. Lessons that Have Been Learned**

***1.The use of class in Django(creating a database)***

**from** django.db **import** models  
**from** django.contrib.auth.models **import** User  
  
  
STATUS = (  
 (0,**"Draft"**),  
 (1,**"Publish"**)  
)  
  
**class** Post(models.Model):  
 image = models.TextField()  
 title = models.CharField(max\_length=200, unique=**True**)  
 slug = models.SlugField(max\_length=200, unique=**True**)  
 author = models.ForeignKey(User, on\_delete= models.CASCADE,related\_name=**'blog\_posts'**)  
 updated\_on = models.DateTimeField(auto\_now= **True**)  
 content = models.TextField()  
 created\_on = models.DateTimeField(auto\_now\_add=**True**)  
 status = models.IntegerField(choices=STATUS, default=0)  
  
 **class** Meta:  
 ordering = [**'-created\_on'**]  
  
 **def** \_\_str\_\_(self):  
 **return** self.title

I did watch youtube tutorial and realize class in Django is a database storage using sqlite platforms. The status enable us to choose between wanting to publish the post from database to site now or just draft it. models.Model is a built in function for Django where you can use in creating database field such as Textfield, Charfield, on\_delete, DateTimeField, and Integerfield. It’s like a library.

1. ***Display views using function in Django framework:***

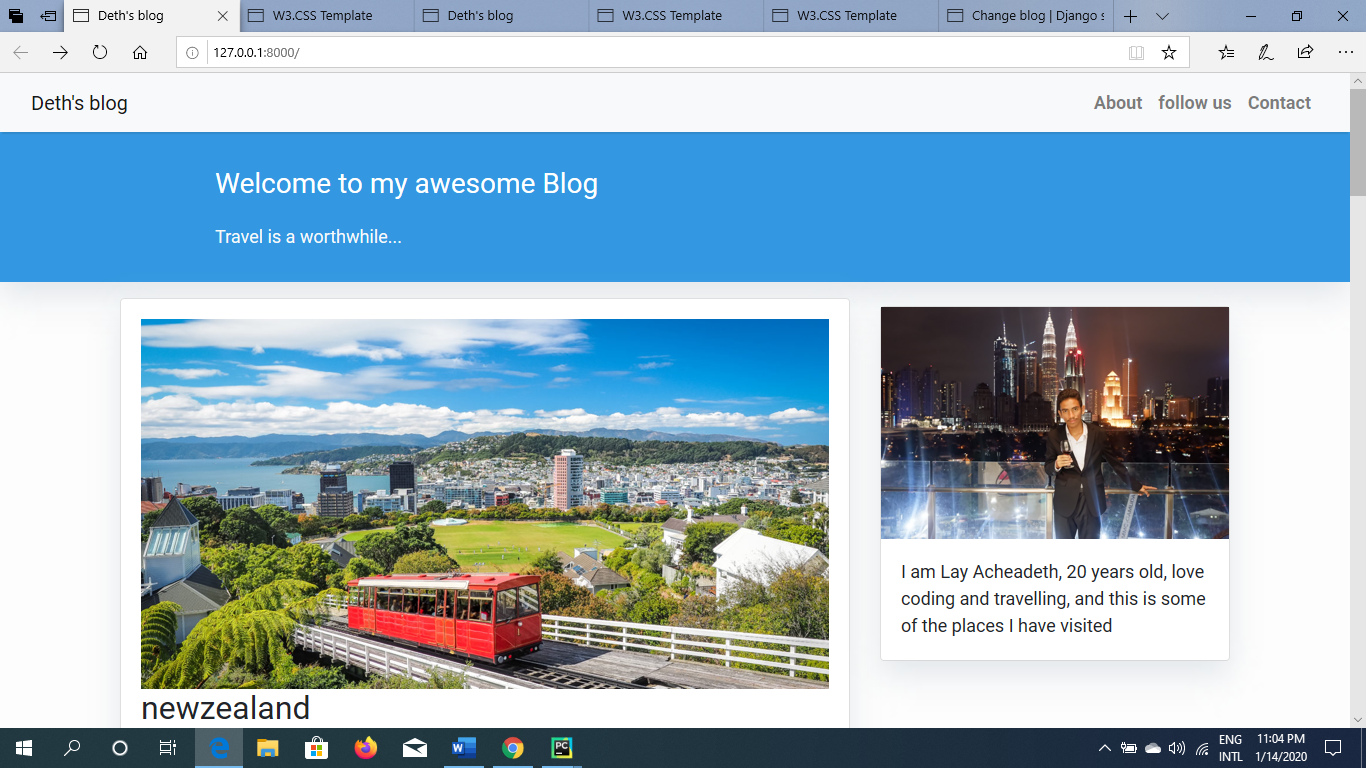
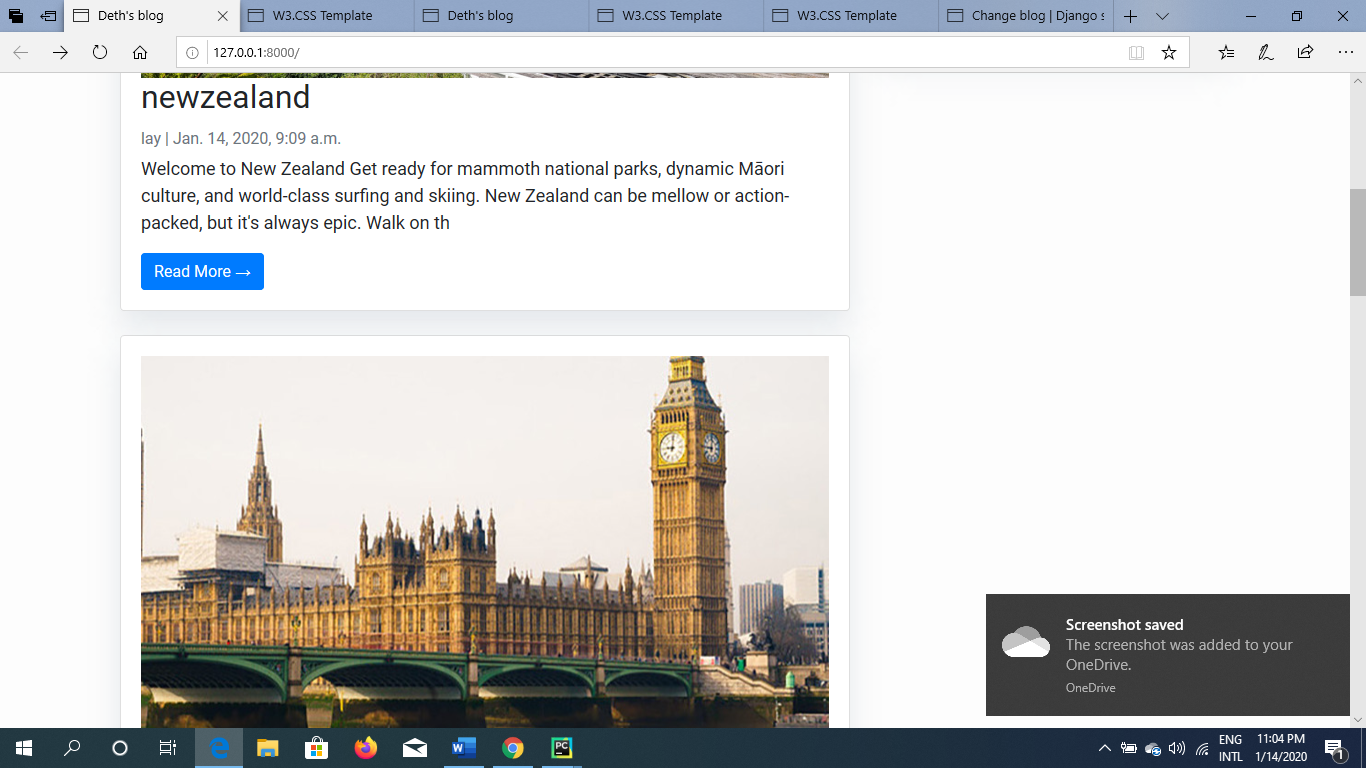
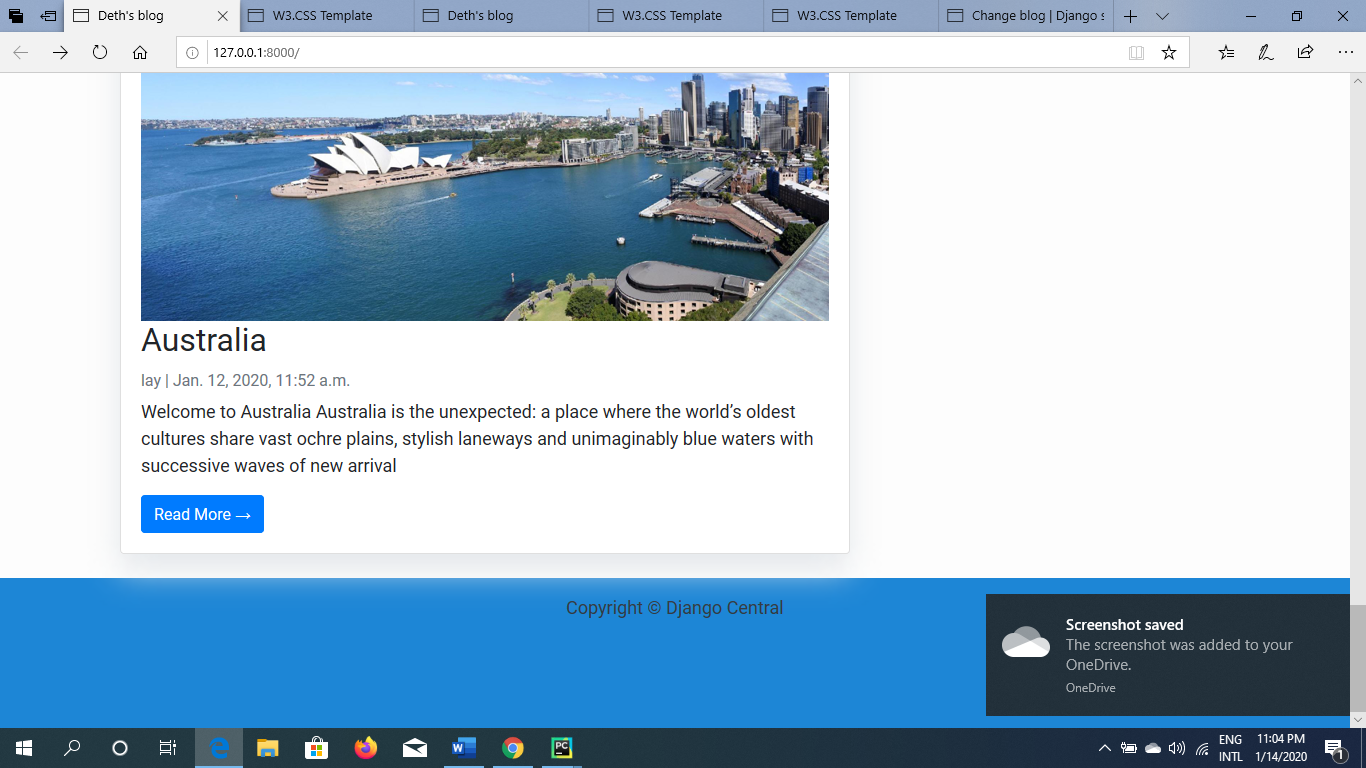
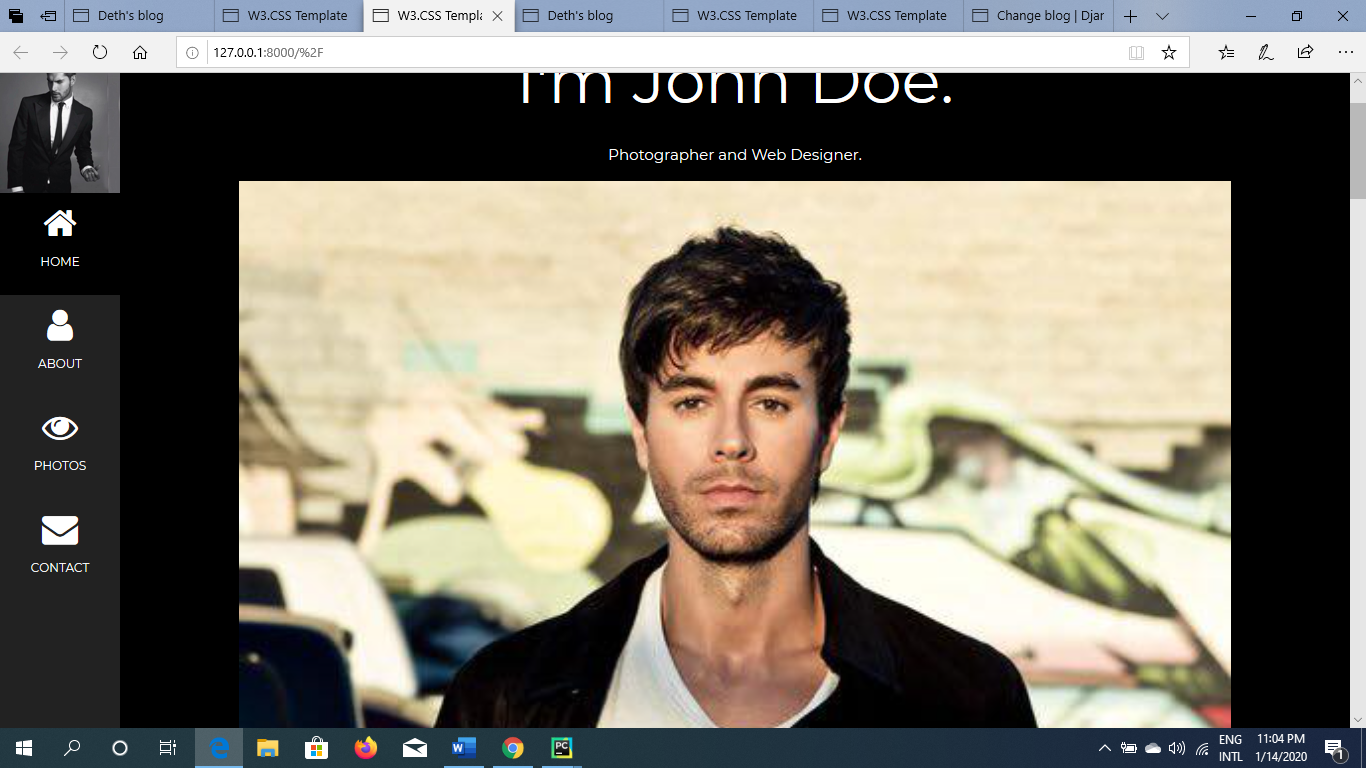
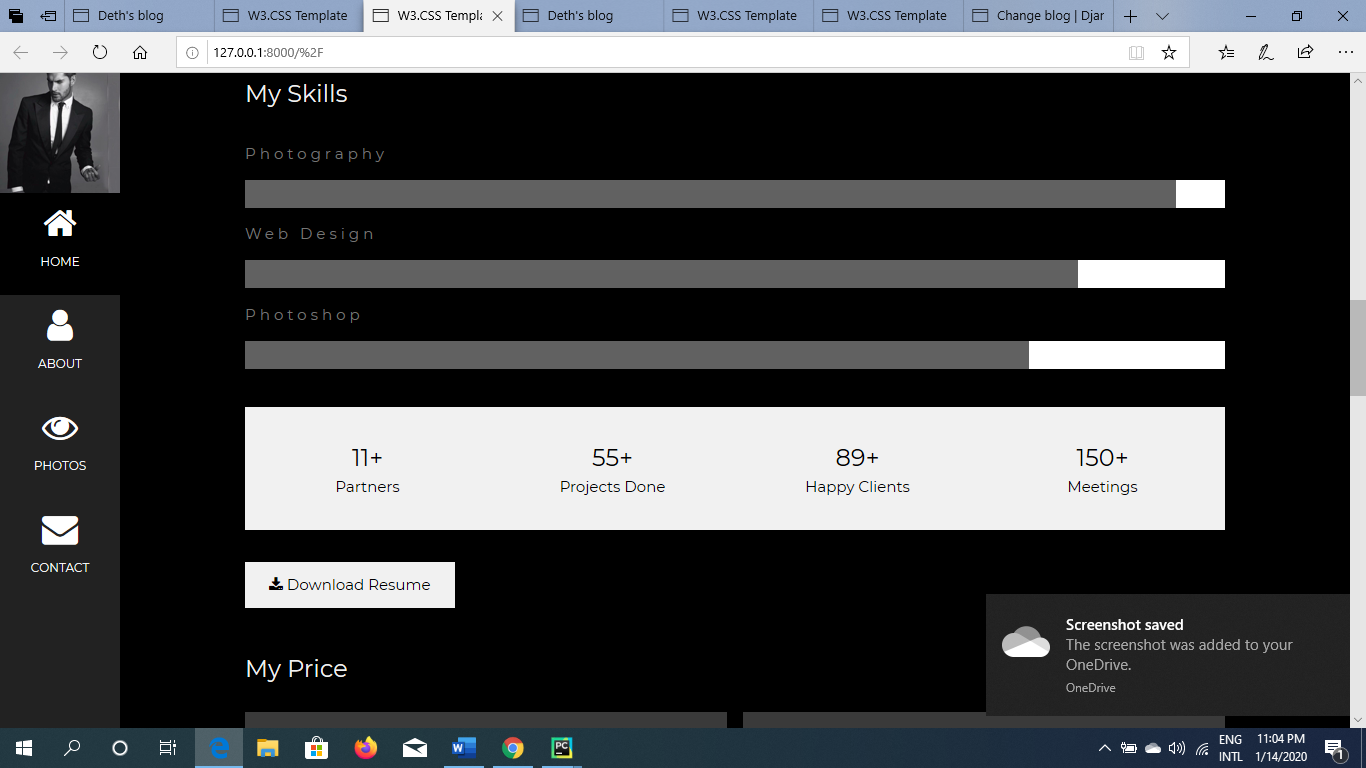
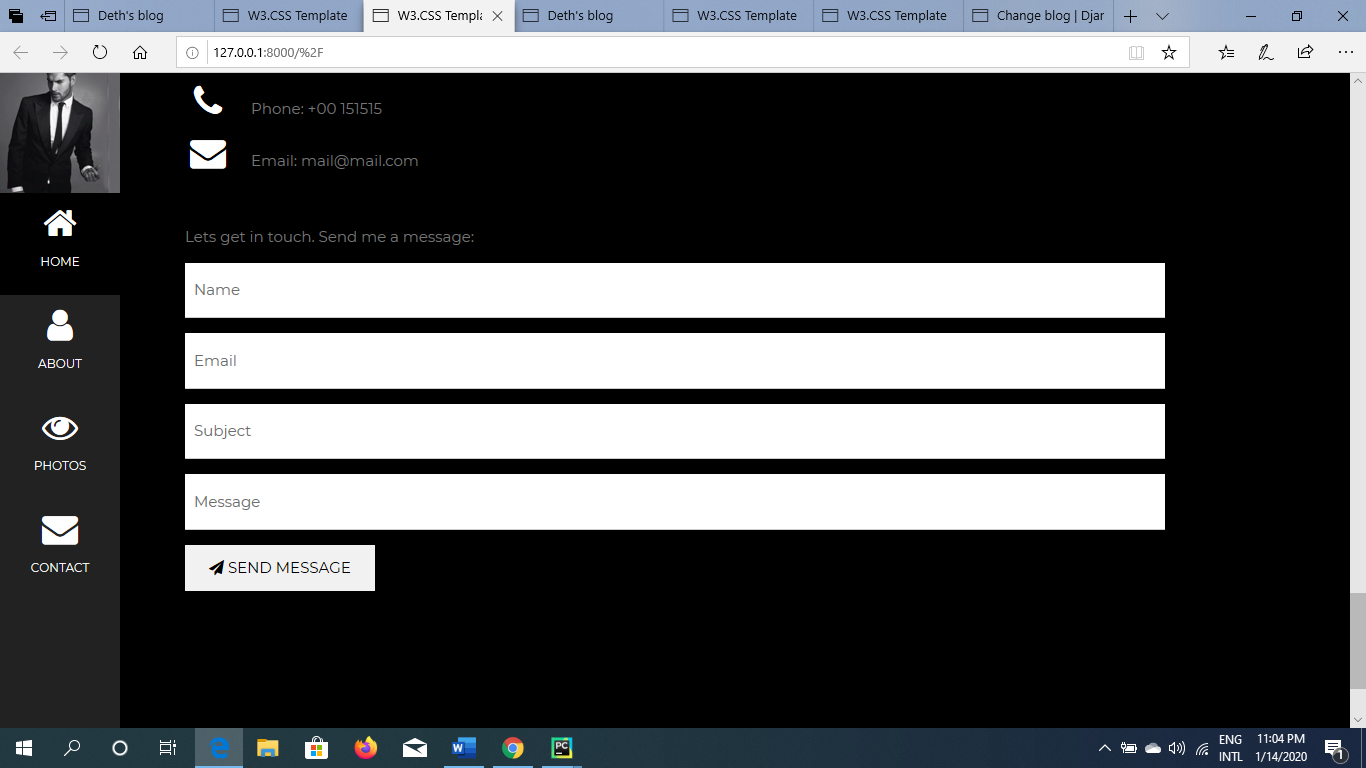
I learn to display the view of website by using function in django that access it url direct to the html file. With this framework, I have no constraint of managing all html at once, and I can even separate and organize the file really well and facilitate me to work better with html file.

Import generic here means importing the listviews style so you don’t have to do it manually(it’s built-in function).

Import get\_object\_or\_404 means it will display object, incase error it does not show the error but ‘404 error not found’

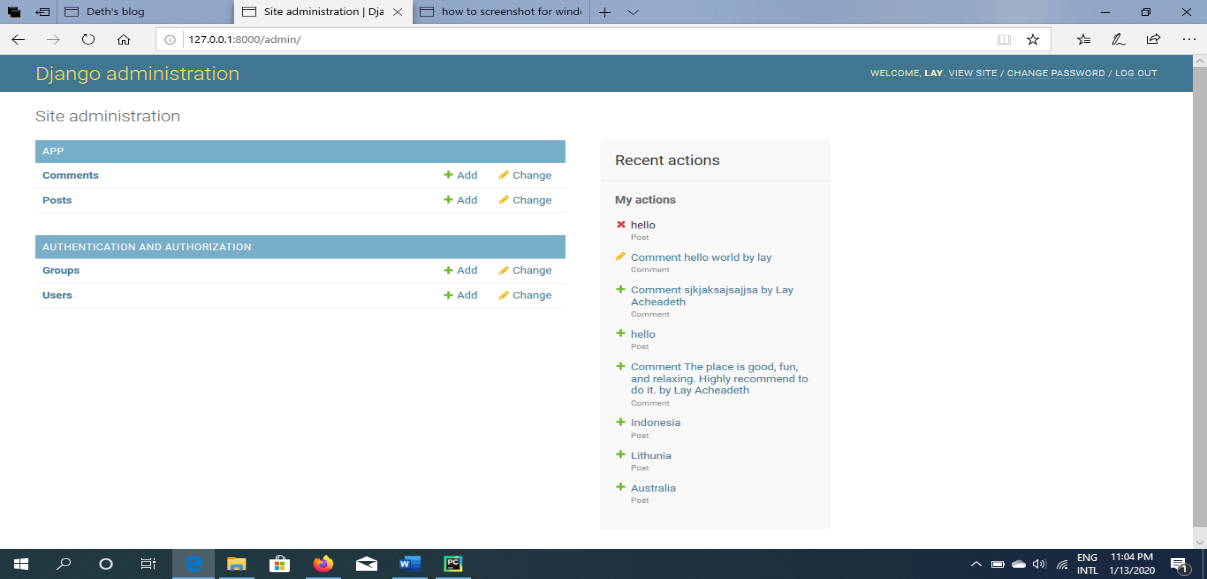
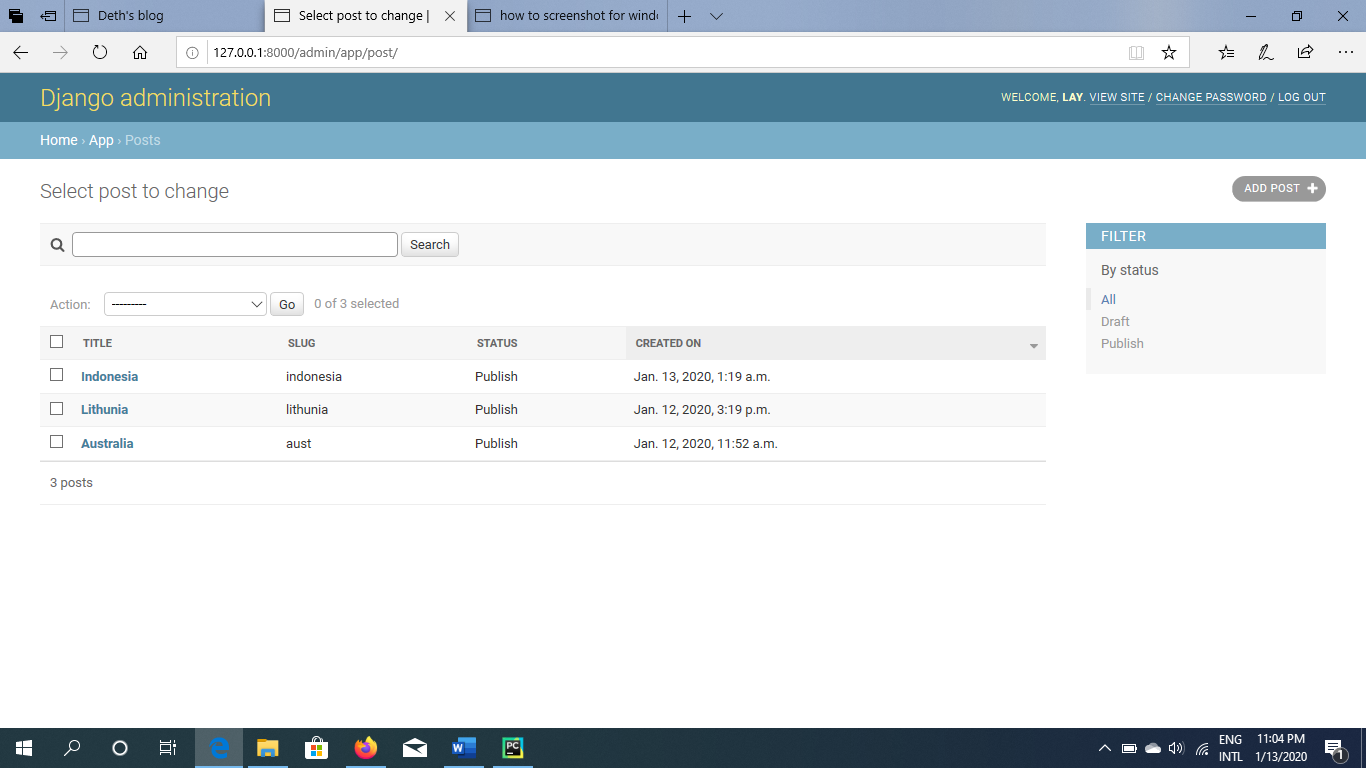
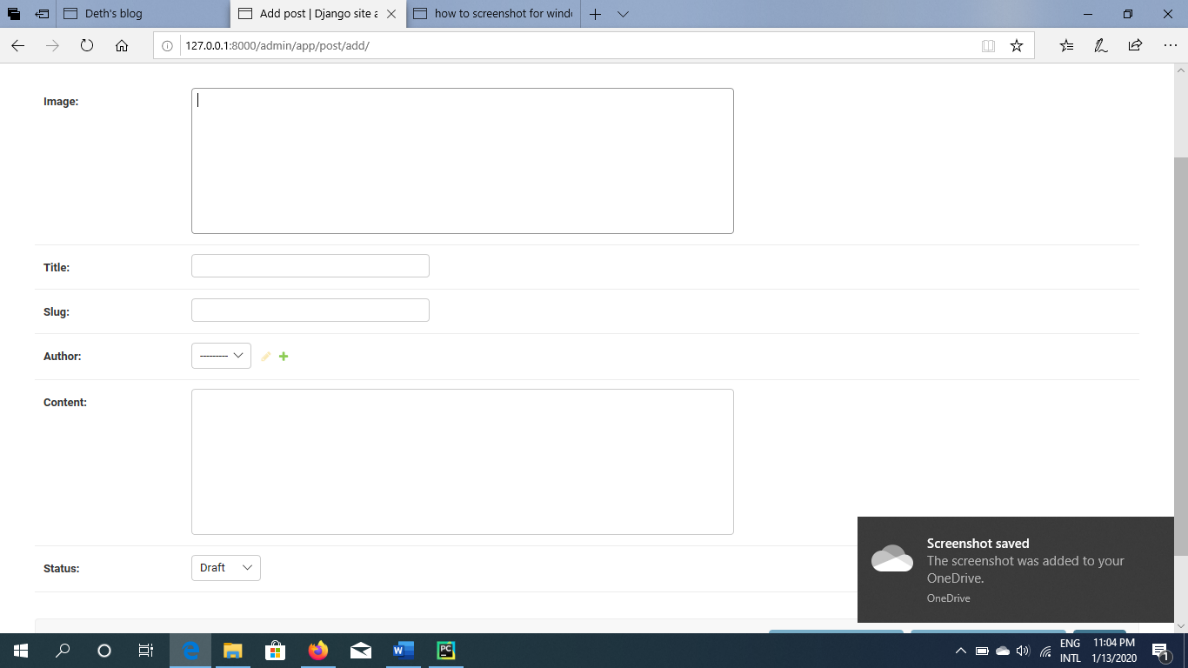
The other import is just import from the file in the same directory.

**from** django.views **import** generic  
**from** .models **import** Post,blog  
**from** .forms **import** CommentForm  
**from** django.shortcuts **import** render, get\_object\_or\_404  
  
  
  
**class** PostList(generic.ListView):  
 queryset = Post.objects.filter(status=1).order\_by(**'-created\_on'**)  
 template\_name = **'app/index.html'  
  
  
def** post\_detail(request, slug):  
 template\_name = **'app/post\_detail.html'** post = get\_object\_or\_404(Post, slug=slug)  
 comments = post.comments.filter(active=**True**)  
 new\_comment = **None** *# Comment posted* **if** request.method == **'POST'**:  
 comment\_form = CommentForm(data=request.POST)  
 **if** comment\_form.is\_valid():  
 *# Create Comment object but don't save to database yet* new\_comment = comment\_form.save(commit=**False**)  
 *# Assign the current post to the comment* new\_comment.post = post  
 *# Save the comment to the database* new\_comment.save()  
 **else**:  
 comment\_form = CommentForm()  
  
 **return** render(request, template\_name, {**'post'**: post,  
 **'comments'**: comments,  
 **'new\_comment'**: new\_comment,  
 **'comment\_form'**: comment\_form})  
**def** about\_us(request):  
 obj=blog.objects.get(id=1)  
 context={  
 **'object'**:obj,  
 **'description'**: **'hello I am LAY, 20 years old, and a student dream to be a full-stack-developer'**,  
 **'title'**:[**'deth'**,**'lay acheadeth'**,**'ace is the best in the world'**],  
 }  
 **return** render(request, **'app/about\_us.html'**,context)



1. ***The usage of database(letting admin to post and modify the blog):***

I can modify my blog without having to go to my html code, but using the database to display and it’s more dynamic. Using only one language and one framework. It does facilitate us a lot.



Here the code to this database

Admin.py( this part is to register all the database to admin so admin can modify it better)

**from** django.contrib **import** admin  
**from** .models **import** Post  
**class** PostAdmin(admin.ModelAdmin):  
 list\_display = (**'title'**, **'slug'**, **'status'**,**'created\_on'**)  
 list\_filter = (**"status"**,)  
 search\_fields = [**'title'**, **'content'**]  
 prepopulated\_fields = {**'slug'**: (**'title'**,)}  
admin.site.register(Post,PostAdmin)  
**from** django.contrib **import** admin  
**from** .models **import** Comment  
**class** CommentAdmin(admin.ModelAdmin):  
 list\_display = (**'name'**, **'body'**, **'post'**, **'created\_on'**, **'active'**)  
 list\_filter = (**'active'**, **'created\_on'**)  
 search\_fields = (**'name'**, **'email'**, **'body'**)  
 actions = [**'approve\_comments'**]  
  
 **def** approve\_comments(self, request, queryset):  
 queryset.update(active=**True**)  
admin.site.register(Comment)

**app/models.py**

**(This is where we create the database without the table and pushing to it yet. It’s a scheme for it.)**

**from** django.db **import** models  
**from** django.contrib.auth.models **import** User  
  
  
STATUS = (  
 (0,**"Draft"**),  
 (1,**"Publish"**)  
)  
  
**class** Post(models.Model):  
 image = models.TextField()  
 title = models.CharField(max\_length=200, unique=**True**)  
 slug = models.SlugField(max\_length=200, unique=**True**)  
 author = models.ForeignKey(User, on\_delete= models.CASCADE,related\_name=**'blog\_posts'**)  
 updated\_on = models.DateTimeField(auto\_now= **True**)  
 content = models.TextField()  
 created\_on = models.DateTimeField(auto\_now\_add=**True**)  
 status = models.IntegerField(choices=STATUS, default=0)  
  
 **class** Meta:  
 ordering = [**'-created\_on'**]  
  
 **def** \_\_str\_\_(self):  
 **return** self.title

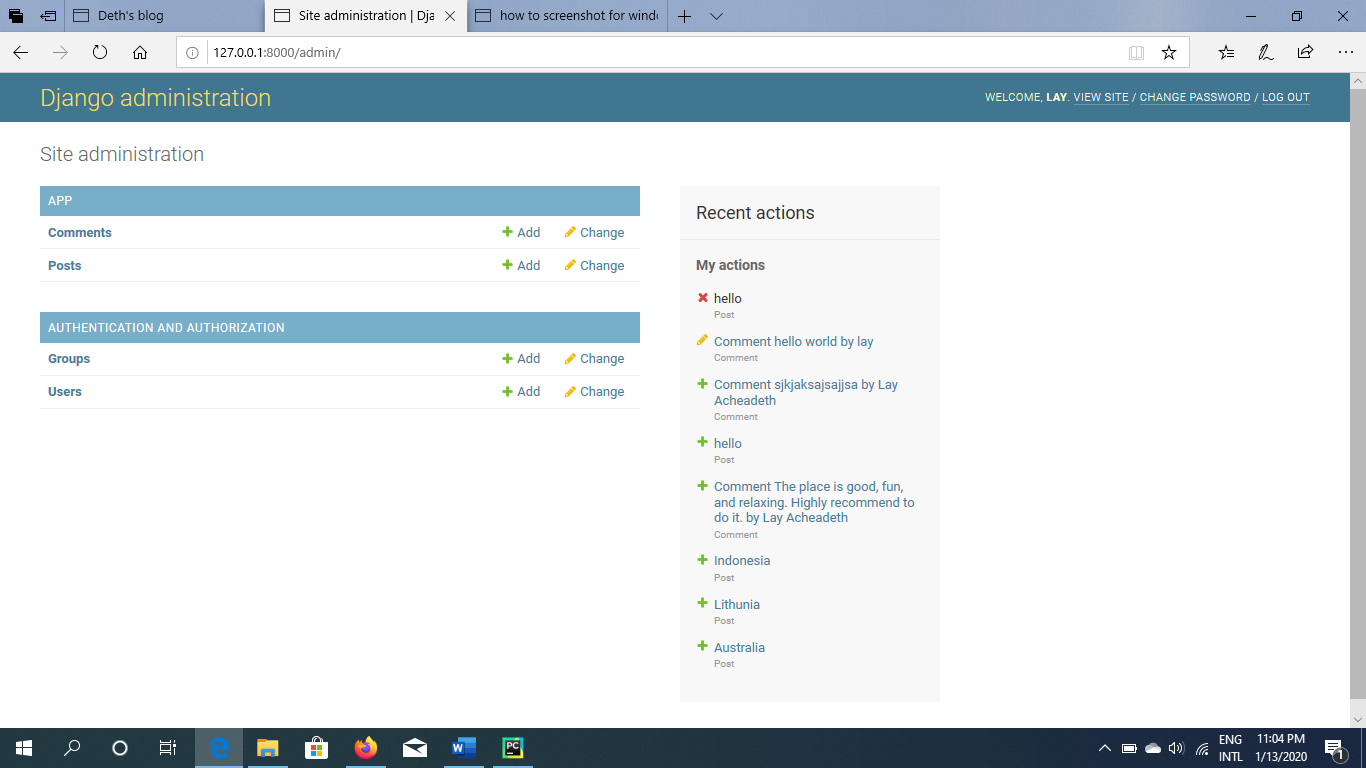
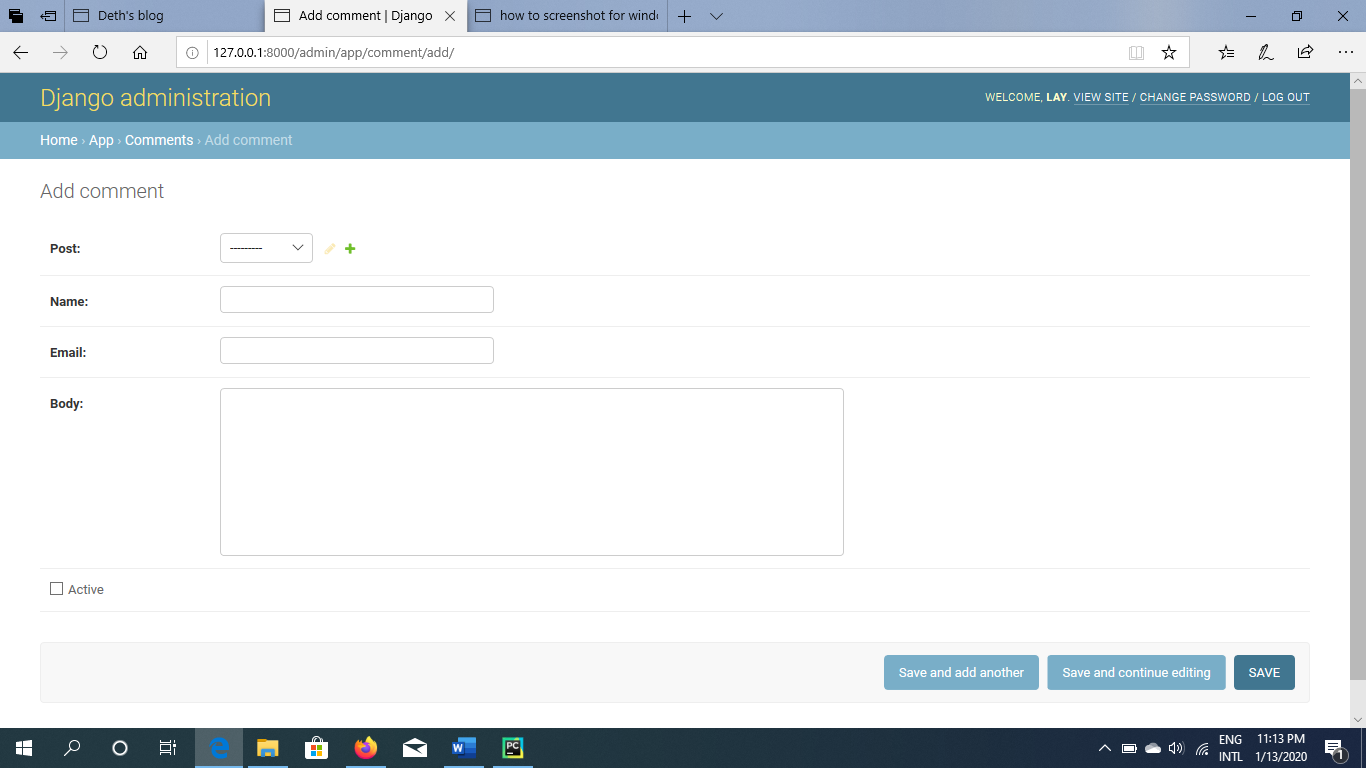
with a few command line which is:

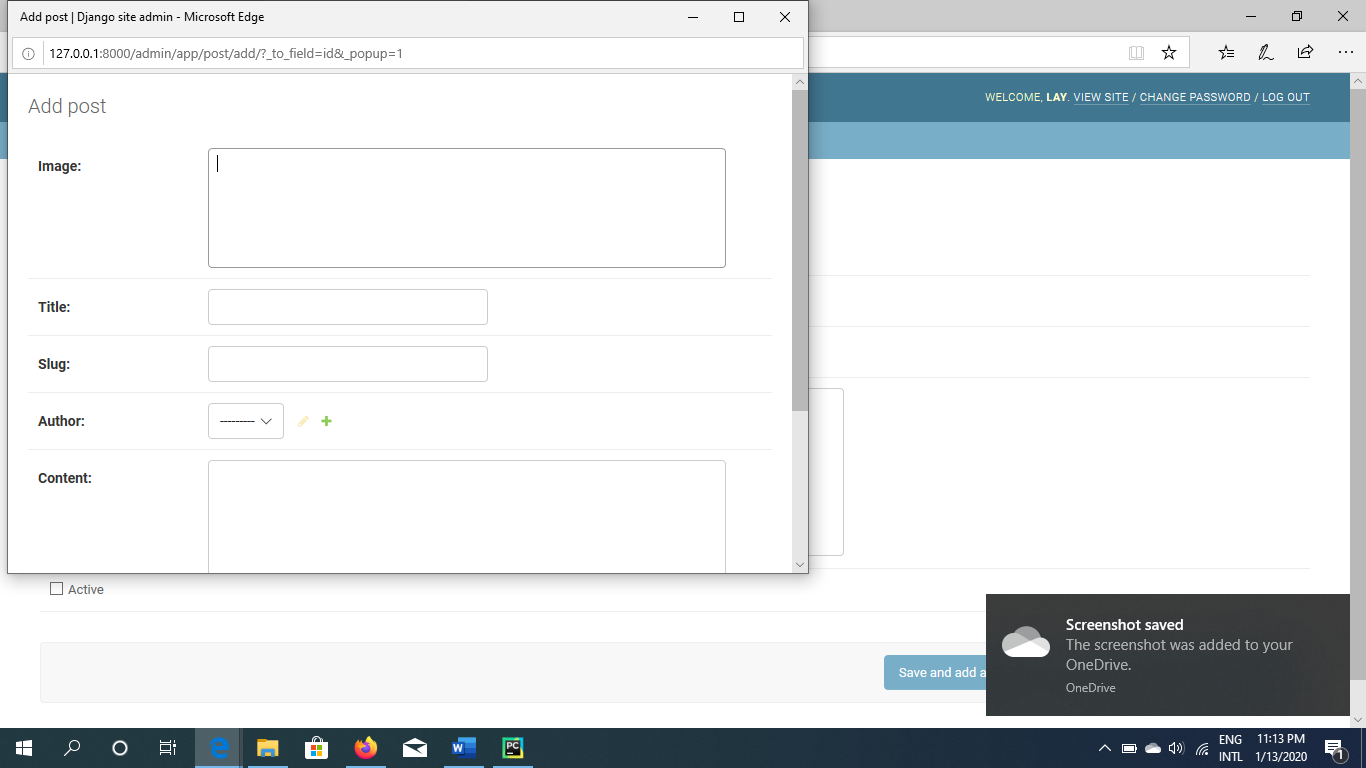
python manage.py makemigrations to create the table

python manage.py migrate to apply the object to the database

1. **Comment system within the blog.**

We enable our user to comment on our own post also and we have right to moderate which comments shall be published, which shall not.





Here is the code.

**from** .models **import** Comment  
**from** django **import** forms  
  
form.py( this is the form of comment form code)  
**class** CommentForm(forms.ModelForm):  
 **class** Meta:  
 model = Comment  
 fields = (**'name'**, **'email'**, **'body'**)

**models.py(this is the database of the comment form)**

**class** Comment(models.Model):  
 post = models.ForeignKey(Post,on\_delete=models.CASCADE,related\_name=**'comments'**)  
 name = models.CharField(max\_length=80)  
 email = models.EmailField()  
 body = models.TextField()  
 created\_on = models.DateTimeField(auto\_now\_add=**True**)  
 active = models.BooleanField(default=**False**)  
  
 **class** Meta:  
 ordering = [**'created\_on'**]  
  
 **def** \_\_str\_\_(self):  
 **return 'Comment {} by {}'**.format(self.body, self.name)

**admin.py(register database to admin and also to make approval approve by admin so as to display the comment)**

**class** CommentAdmin(admin.ModelAdmin):  
 list\_display = (**'name'**, **'body'**, **'post'**, **'created\_on'**, **'active'**)  
 list\_filter = (**'active'**, **'created\_on'**)  
 search\_fields = (**'name'**, **'email'**, **'body'**)  
 actions = [**'approve\_comments'**]  
  
 **def** approve\_comments(self, request, queryset):  
 queryset.update(active=**True**)  
admin.site.register(Comment)

1. Url path

**App/urls.py**

**from** django.urls **import** path  
**from** . **import** views  
urlpatterns = [  
 path(**''**, views.PostList.as\_view(), name=**'home'**),  
 path(**'<slug:slug>/'**, views.post\_detail, name=**'post\_detail'**),  
 path(**'/'**,views.about\_us,name=**'about\_us'**),  
]

project/urls.py

**from** django.contrib **import** admin  
**from** django.urls **import** path,include  
  
urlpatterns = [  
 path(**'admin/'**, admin.site.urls),  
 path(**''**,include(**'app.urls'**)),  
]

both of this code above is the url configuration. It’s different from the other url configuration for Django framework. This is the code that first build the url in the app and then transport to the project.

**V. Problem that Have Been Overcome**

First, this is a static website. Without any knowledge of coding, I can just google the templates and create as many blog as I want. However, I do try to make it more dynamic as much as I can by accessing to database and display those data. In this way, website will be much easier to take control. User wouldn’t have to face problem by having to go through html code. They just create their own account in our platform and open their database and publish their own blog as much as they want. It’s like a news’s website where all the reporter have their accounts and post their news on the site.

**Resources :**

- <https://djangocentral.com/creating-comments-system-with-django/> (creating comment system )

- <https://djangocentral.com/building-a-blog-application-with-django/(creating> blog)

- <https://stackoverflow.com> (website I used when I was trying to fix the errors and understanding the code better )

**VI. Source Code**

*app/model.py*

**from** django.db **import** models  
**from** django.contrib.auth.models **import** User  
  
  
STATUS = (  
 (0,**"Draft"**),  
 (1,**"Publish"**)  
)  
  
**class** Post(models.Model):  
 image = models.TextField()  
 title = models.CharField(max\_length=200, unique=**True**)  
 slug = models.SlugField(max\_length=200, unique=**True**)  
 author = models.ForeignKey(User, on\_delete= models.CASCADE,related\_name=**'blog\_posts'**)  
 updated\_on = models.DateTimeField(auto\_now= **True**)  
 content = models.TextField()  
 created\_on = models.DateTimeField(auto\_now\_add=**True**)  
 status = models.IntegerField(choices=STATUS, default=0)  
  
 **class** Meta:  
 ordering = [**'-created\_on'**]  
  
 **def** \_\_str\_\_(self):  
 **return** self.title

*app/urls.py*

**from** django.urls **import** path  
**from** . **import** views  
urlpatterns = [  
 path(**''**, views.PostList.as\_view(), name=**'home'**),  
 path(**'<slug:slug>/'**, views.PostDetail.as\_view(), name=**'post\_detail'**),  
]

*app/views.py*

**from** django.views **import** generic  
**from** .models **import** Post

**class** PostList(generic.ListView):  
 queryset = Post.objects.filter(status=1).order\_by(**'-created\_on'**)  
 template\_name = **'app/index.html'  
  
class** PostDetail(generic.DetailView):  
 model = Post  
 template\_name = **'app/post\_detail.html'**

*project/urls.py*

**from** django.contrib **import** admin  
**from** django.urls **import** path,include  
  
urlpatterns = [  
 path(**'admin/'**, admin.site.urls),  
 path(**''**,include(**'app.urls'**)),  
]

*project/settings.py*

INSTALLED\_APPS = [  
 **'app.apps.AppConfig'**,  
 **'django.contrib.admin'**,  
 **'django.contrib.auth'**,  
 **'django.contrib.contenttypes'**,  
 **'django.contrib.sessions'**,  
 **'django.contrib.messages'**,  
 **'django.contrib.staticfiles'**,  
 **'crispy\_forms'**,  
]

*app/admin.py*

**rom** django.contrib **import** admin  
**from** .models **import** Post  
**class** PostAdmin(admin.ModelAdmin):  
 list\_display = (**'title'**, **'slug'**, **'status'**,**'created\_on'**)  
 list\_filter = (**"status"**,)  
 search\_fields = [**'title'**, **'content'**]  
 prepopulated\_fields = {**'slug'**: (**'title'**,)}  
admin.site.register(Post,PostAdmin)

*app/base.html*

<!DOCTYPE **html**>  
<**html**>  
  
 <**head**>  
 <**title**>Deth's blog</**title**>  
 <**link href="https://fonts.googleapis.com/css?family=Roboto:400,700" rel="stylesheet"**>  
 <**meta name="google" content="notranslate"** />  
 <**meta name="viewport" content="width=device-width, initial-scale=1"** />  
 <**link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"  
 crossorigin="anonymous"** />  
  
 </**head**>  
  
 <**body**>  
 <**style**>  
 body {  
 font-family: "Roboto", sans-serif;  
 font-size: 17px;  
 background-color: #fdfdfd;  
 }  
 .shadow {  
 box-shadow: 0 4px 2px -2px rgba(0, 0, 0, 0.1);  
 }  
 .btn-danger {  
 color: #fff;  
 background-color: #f00000;  
 border-color: #dc281e;  
 }  
 .masthead {  
 background: #3398E1;  
 height: auto;  
 padding-bottom: 15px;  
 box-shadow: 0 16px 48px #E3E7EB;  
 padding-top: 10px;  
 }  
 #footer{  
 width:1350px;  
 height:150px;  
 background-color:#1D86D6;  
 }  
 </**style**>  
  
 *<!-- Navigation -->* <**nav class="navbar navbar-expand-lg navbar-light bg-light shadow" id="mainNav"**>  
 <**div class="container-fluid"**>  
 <**a class="navbar-brand" href="{% url 'home' %}"**>Deth's blog</**a**>  
 <**button class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse" data-target="#navbarResponsive"  
 aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle navigation"**>  
 <**span class="navbar-toggler-icon"**></**span**>  
 </**button**>  
 <**div class="collapse navbar-collapse" id="navbarResponsive"**>  
 <**ul class="navbar-nav ml-auto"**>  
 <**li class="nav-item text-black"**>  
 <**a class="nav-link text-black font-weight-bold" href="#"**>About</**a**>  
 </**li**>  
 <**li class="nav-item text-black"**>  
 <**a class="nav-link text-black font-weight-bold" href="#"**>  
 follow us  
 </**a**>  
 </**li**>  
 <**li class="nav-item text-black"**>  
 <**a class="nav-link text-black font-weight-bold" href="#"**>Contact</**a**>  
 </**li**>  
 </**ul**>  
 </**div**>  
 </**div**>  
 </**nav**>  
 {% block content %}  
 *<!-- Content Goes here -->* {% endblock content %}  
 *<!-- Footer -->* <**div id="footer" class="py-3 bg-lightshadow"**>  
  
 <**p class="m-0 text-dark text-center "**>Copyright **&copy;** Django Central</**p**>  
 </**div**>  
 </**body**>  
</**html**>

*app/index.html*

{% extends "app/base.html" %}  
{% block content %}  
<**style**>  
 body {  
 font-family: "Roboto", sans-serif;  
 font-size: 18px;  
 background-color: #fdfdfd;  
 }  
  
 .head\_text {  
 color: white;  
 }  
  
 .card {  
 box-shadow: 0 16px 48px #E3E7EB;  
 }  
 img{  
 max-width: 100%;  
 }  
</**style**>  
  
<**header class="masthead"**>  
 <**div class="overlay"**></**div**>  
 <**div class="container"**>  
 <**div class="row"**>  
 <**div class=" col-md-8 col-md-10 mx-auto"**>  
 <**div class="site-heading"**>  
 <**h3 class=" site-heading my-4 mt-3 text-white"**> Welcome to my awesome Blog </**h3**>  
 <**p class="text-light"**> Travel is a worthwhile...  
 </**p**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
</**header**>  
<**div class="container"**>  
 <**div class="row"**>  
 *<!-- Blog Entries Column -->* <**div class="col-md-8 mt-3 left"**>  
 {% for post in post\_list %}  
 <**div class="card mb-4"**>  
 <**div class="card-body"**>  
 <**img src="{{ post.image }}"**>  
 <**h2 class="card-title"**>{{ post.title }}</**h2**>  
 <**p class="card-text text-muted h6"**>{{ post.author }} | {{ post.created\_on}} </**p**>  
 <**p class="card-text"**>{{post.content|slice:":200" }}</**p**>  
 <**a href="{% url 'post\_detail' post.slug %}" class="btn btn-primary"**>Read More **&rarr;**</**a**>  
 </**div**>  
 </**div**>  
 {% endfor %}  
 </**div**>  
 {% block sidebar %} {% include 'app/sidebar.html' %} {% endblock sidebar %}  
 </**div**>  
</**div**>  
{% endblock %}

*app/post\_detail.html*

{% extends 'app/base.html' %} {% block content %}  
{% load crispy\_forms\_tags %}  
<**style**>  
 #comment{  
 margin-top:500px;  
 }  
</**style**>  
<**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-12 card mb-4 mt-3 left top"**>  
 <**div class="card-body"**>  
 <**img class="img-fluid" src="{{ object.image }}"**>  
 <**h1**>{% block title %} {{ object.title }} {% endblock title %}</**h1**>  
 <**p class=" text-muted"**>{{ post.author }} | {{ post.created\_on }}</**p**>  
 <**p class="card-text "**>{{ object.content | safe }}</**p**>  
 </**div**>  
 </**div**>  
<**div class="col-md-8 card mb-4 mt-3 "**>  
 <**div class="card-body"**>  
 *<!-- comments -->* <**h2**>{{ comments.count }} comments</**h2**>  
  
 {% for comment in comments %}  
 <**div class="comments" style="**padding: 10px;**"**>  
 <**p class="font-weight-bold"**>  
 {{ comment.name }}  
 <**span class=" text-muted font-weight-normal"**>  
 {{ comment.created\_on }}  
 </**span**>  
 </**p**>  
 {{ comment.body | linebreaks }}  
 </**div**>  
 {% endfor %}  
 </**div**>  
 </**div**>  
 <**div class="col-md-8 card mb-4 mt-3 "**>  
 <**div class="card-body"**>  
 {% if new\_comment %}  
 <**div class="alert alert-success" role="alert"**>  
 Your comment is awaiting moderation  
 </**div**>  
 {% else %}  
 <**h3**>Leave a comment</**h3**>  
 <**form method="post" style="**margin-top: 1.3em;**"**>  
 {{ comment\_form.as\_p }}  
 {% csrf\_token %}  
 <**button type="submit" class="btn btn-primary btn-lg"**>Submit</**button**>  
 </**form**>  
 {% endif %}  
 </**div**>  
 </**div**>  
 </**div**>  
</**div**>  
{% endblock content %}

*app/sidebar.html*

{% block sidebar %}  
  
<**style**>  
 .card{  
 box-shadow: 0 16px 48px #E3E7EB;  
 }  
  
</**style**>  
  
*<!-- Sidebar Widgets Column -->*<**div class="col-md-4 float-right "**>  
<**div class="card my-4"**>  
<**img src="https://scontent.fcgk5-1.fna.fbcdn.net/v/t1.15752-9/40621637\_1800720953368587\_7012541405104963584\_n.jpg?\_nc\_cat=111&\_nc\_oc=AQkeHRuUqr4Kft8Y7vs1AP7\_ZZqemlsK-XGpFjGCikPoCSbxhTypiY8rfaladykV3lU&\_nc\_ht=scontent.fcgk5-1.fna&oh=f5c64d8c6df8288d4eab0d8889cf6a8a&oe=5EA09923" style="**width:100%**"**>  
 <**div class="card-body"**>  
 <**p class="card-text"**>I am Lay Acheadeth, 20 years old, love coding and travelling, and this is some of the places I have visited </**p**>  
 </**div**>  
</**div**>  
</**div**>  
  
{% endblock sidebar %}