OPINION BLOG CODE

DJANGO 1.8

MODELS.PY

**from** django.db **import** models  
**from** django.template.defaultfilters **import** slugify  
**from** django.contrib.auth.models **import** User  
**from** django.forms **import** ModelForm  
  
**from** tinymce.models **import** HTMLField  
  
**class** Category(models.Model):  
 name = models.CharField(max\_length=128, unique=**True**)  
 slug = models.SlugField(unique=**True**)  
  
 **def** save(self, \*args, \*\*kwargs):  
 self.slug = slugify(self.name)  
 super(Category, self).save(\*args, \*\*kwargs)  
  
 **def** \_\_unicode\_\_(self):  
 **return** self.name  
  
**class** Post(models.Model):  
 title = models.CharField(max\_length=100, unique=**True**)  
 slug = models.SlugField(max\_length=100, unique=**True**)  
 body = models.TextField()  
 posted = models.DateTimeField(db\_index=**True**, auto\_now\_add=**True**)  
 category = models.ForeignKey(Category)  
  
 **def** save(self, \*args, \*\*kwargs):  
 self.slug = slugify(self.title)  
 super(Post, self).save(\*args, \*\*kwargs)  
  
 **def** \_\_unicode\_\_(self):  
 **return '%s'** % self.title  
  
**class** Comment (models.Model):  
 comment\_body = models.TextField()  
 commented\_on = models.DateTimeField(db\_index=**True**,auto\_now=**True**)  
 commenter = models.ForeignKey(User)  
 post = models.ForeignKey(Post)  
  
 **def** \_\_unicode\_\_(self):  
 **return '%s'** % self.comment\_body  
  
**class** Tag (models.Model):  
 tag\_title = models.CharField(max\_length=100, unique=**True**)  
 slug = models.SlugField(max\_length=100, unique=**True**)  
 post = models.ManyToManyField(Post)  
  
 **def** save(self, \*args, \*\*kwargs):  
 self.slug = slugify(self.tag\_title)  
 super(Tag, self).save(\*args, \*\*kwargs)  
  
 **def** \_\_unicode\_\_(self):  
 **return '%s'** % self.tag\_title  
  
  
**class** UserProfile(models.Model):  
 *# This line is required.Links Userprofile to a User model instance.* user = models.OneToOneField(User)  
  
 *# The additional attributes we wish to include.* website = models.URLField(blank=**True**)  
 picture = models.ImageField(upload\_to=**'profile\_images'**, blank=**True**)  
  
 *# Override th \_\_unicode\_\_() method to return out something meaningful!* **def** \_\_unicode\_\_(self):  
 **return** self.user.username  
  
  
**class** Contact(models.Model):  
 name = models.CharField(max\_length=128)  
 email = models.EmailField(max\_length=75)  
 subject = models.CharField(max\_length=128)  
 message = models.TextField()  
  
 **def** \_\_unicode\_\_(self):  
 **return** self.subject  
  
  
LIKE\_CHOICES = (  
 (**'YS'**, **'YES I LIKE'**),  
 (**'NO'**, **'I DON\'T LIKE'**),  
 )  
  
**class** Feedback(models.Model):  
 name = models.CharField(max\_length=128)  
 email = models.EmailField(max\_length=75)  
 like\_dislike = models.CharField(max\_length=2, choices=LIKE\_CHOICES)  
 feedback\_msg = models.TextField()  
  
 **def** \_\_unicode\_\_(self):  
 **return** self.name

VIEWS.PY

**from** django.shortcuts **import** render, redirect  
**from** django.http **import** HttpResponse, HttpResponseRedirect  
**from** blog.models **import** Category, Post, Comment, Tag, Contact, Feedback  
**from** blog.forms **import** ContactForm, CategoryForm, FeedbackForm, CommentForm, PostForm, TagForm  
**from** django.contrib.auth.decorators **import** login\_required  
  
*# Create your views here.***def** index(request):  
 category\_list = Category.objects.order\_by(**'-name'**)[:5]  
 post\_list = Post.objects.order\_by(**'-posted'**)[:7]  
 context\_dict = {**'categories'**: category\_list, **'post\_list'**: post\_list}  
 response = render(request,**'blog/index.html'**, context\_dict)  
 **return** response  
  
  
**def** home(request):  
 context\_dict = {**'ahmed'** : **'Ahmed Khan'**}  
 response = render(request,**'blog/index.html'**, context\_dict)  
 **return** response  
  
  
**def** categories(request):  
 category\_list = Category.objects.order\_by(**'-name'**)  
 context\_dict = {**'categories'**: category\_list}  
 response = render(request,**'blog/categories.html'**, context\_dict)  
 **return** response  
  
  
**def** blog\_post(request, blog\_post\_slug):  
 post\_detail = Post.objects.get(slug=blog\_post\_slug)  
 current\_user = request.user  
 comments = Comment.objects.filter(post=post\_detail.id)  
 tags = Tag.objects.filter(post=post\_detail.id)  
 form = CommentForm(request.POST **or None**)  
 tagform = TagForm(request.POST **or None**)  
 **if** form.is\_valid():  
 comment = form.save(commit=**False**)  
 comment.post = post\_detail  
 comment.commenter = current\_user  
 comment.save()  
 **return** redirect(request.path)  
  
 **if** tagform.is\_valid():  
 post\_id = post\_detail.id  
 post\_for\_tag = Post.objects.get(id=post\_id)  
  
 tag\_title\_from\_form = tagform[**'tag\_title'**].value().strip().lower()  
 tag\_title\_from\_form = tag\_title\_from\_form.rstrip(**','**)  
 striped\_tags = tag\_title\_from\_form.split(**','**)  
 **for** t **in** striped\_tags:  
 print (t)  
 **try**:  
 nt = t.strip()  
 existing\_tag = Tag.objects.get(tag\_title\_\_iexact=nt)  
 post\_for\_tag.tag\_set.add(existing\_tag)  
 **except** Tag.DoesNotExist:  
 nt = t.strip()  
 new\_tag = Tag(tag\_title=nt)  
 new\_tag.save()  
 post\_for\_tag.tag\_set.add(new\_tag)  
 **continue  
 return** redirect(request.path)  
 context\_dict = {**'post'**: post\_detail, **'slug'**: blog\_post\_slug, **'comments'**:comments, **'tags'**:tags, **'form'**:form, **'tagform'**:tagform}  
 response = render(request,**'blog/blog\_post.html'**, context\_dict)  
 **return** response  
  
  
**def** add\_post(request):  
 **if** request.method == **'POST'**:  
 form1Post = PostForm(request.POST)  
 form2Tags = TagForm(request.POST)  
 **if** form1Post.is\_valid() **and** form2Tags.is\_valid():  
 post = form1Post.save()  
 *#post\_instance\_for\_tag = post.save()* tag\_title\_from\_form = form2Tags[**'tag\_title'**].value().strip().lower()  
 tag\_title\_from\_form = tag\_title\_from\_form.rstrip(**','**)  
 striped\_tags = tag\_title\_from\_form.split(**','**)  
 **for** t **in** striped\_tags:  
 received\_tag = t.strip()  
 tag, created = Tag.objects.get\_or\_create(tag\_title=received\_tag)  
 post.tag\_set.add(tag)  
 **return** index(request)  
 **else**:  
 print (form1Post.errors)  
 print (form2Tags.errors)  
 **else**:  
 form1Post = PostForm()  
 form2Tags = TagForm()  
  
 **return** render(request, **'blog/add\_post.html'**, {**'form1'**:form1Post,**'form2'**:form2Tags})  
  
  
**def** comment\_form(request):  
 **if** request.method == **'POST'**:  
 form = CommentForm(request.POST)  
  
 **if** form.is\_valid():  
 form.save(commit=**True**)  
 **return** index (request)  
 **else**:  
 print (form.errors)  
 **else**:  
 form = CommentForm()  
  
 **return** render(request, **'blog/comment\_form.html'**, {**'form'**:form})  
  
  
**def** category(request, category\_name\_slug):  
 category = Category.objects.get(slug=category\_name\_slug)  
 context\_dict = {}  
 context\_dict[**'category\_name'**] = category.name  
 posts = Post.objects.filter(category=category)  
 context\_dict[**'posts'**] = posts  
 context\_dict[**'category'**] = category  
 **return** render(request, **'blog/category.html'**, context\_dict)  
  
  
*#Add Category Veiw  
#@login\_required***def** add\_category(request):  
 **if** request.method == **'POST'**:  
 form = CategoryForm(request.POST)  
  
 **if** form.is\_valid():  
 form.save(commit=**True**)  
 **return** index(request)  
 **else**:  
 print (form.errors)  
 **else**:  
 form = CategoryForm()  
  
 **return** render(request, **'blog/add\_category.html'**, {**'form'**:form})  
  
  
  
**def** tag(request, tag\_name\_slug):  
 tag\_name = Tag.objects.filter(tag\_title=tag\_name\_slug)  
 tag\_id = Tag.objects.get(pk=tag\_name)  
 posts = Post.objects.all().filter(tag=tag\_id)  
 context\_dict = {}  
 context\_dict[**'tag\_name'**] = tag\_name\_slug  
 context\_dict[**'posts'**] = posts  
 **return** render(request, **'blog/posts\_by\_tags.html'**, context\_dict)  
  
  
**def** contact(request):  
 **if** request.method == **'POST'**:  
 form = ContactForm(request.POST)  
  
 **if** form.is\_valid():  
 form.save(commit=**True**)  
 **return** index (request)  
 **else**:  
 print (form.errors)  
 **else**:  
 form = ContactForm()  
  
 **return** render(request, **'blog/contact\_form.html'**, {**'form'**:form})  
  
  
**def** contact\_messages(request):  
 message\_list = Contact.objects.all()  
 context\_dict = {**'messages'**: message\_list}  
 **return** render(request,**'blog/contact\_messages.html'**,context\_dict,)  
  
  
**def** feedback\_form(request):  
 **if** request.method == **'POST'**:  
 form = FeedbackForm(request.POST)  
  
 **if** form.is\_valid():  
 form.save(commit=**True**)  
 **return** index (request)  
 **else**:  
 print (form.errors)  
 **else**:  
 form = FeedbackForm()  
  
 **return** render(request, **'blog/feedback\_form.html'**, {**'form'**:form})  
  
  
**def** feedback(request):  
 message\_list = Feedback.objects.all()  
 context\_dict = {**'messages'**: message\_list}  
 **return** render(request,**'blog/feedback\_rcvd.html'**,context\_dict,)

URLS.PY

**from** django.conf.urls **import** patterns, url  
**from** blog **import** views  
  
urlpatterns = patterns(**''**,  
 url(**r'^$'**, views.index, name=**'index'**),  
  
 url(**r'^contact/$'**, views.contact ,name=**'contact'**),  
 url(**r'^contact\_messages/$'**, views.contact\_messages,name=**'contact\_messages'**),  
  
 url(**r'^categories/$'**, views.categories, name=**'categories'**),  
 url(**r'^feedback\_form/$'**, views.feedback\_form, name=**'feedback\_form'**),  
 url(**r'^comment\_form/$'**, views.comment\_form, name=**'comment\_form'**),  
 url(**r'^feedback/$'**, views.feedback, name=**'feedback'**),  
 url(**r'^category/add\_category/$'**, views.add\_category, name=**'add\_category'**),  
 url(**r'^category/(?P<category\_name\_slug>[-\w]+)/$'**, views.category, name=**'category'**),  
 url(**r'^tag/(?P<tag\_name\_slug>[a-zA-Z0-9\_.-]+)/$'**, views.tag, name=**'tag'**),  
 url(**r'^add\_post/$'**, views.add\_post, name=**'add\_post'**),  
 url(**r'^(?P<blog\_post\_slug>[-\w]+)/$'**, views.blog\_post, name=**'blog\_post'**),  
)

FORMS.PY

**from** django **import** forms  
**from** django.contrib.auth.models **import** User  
**from** blog.models **import** Contact, UserProfile, Category, Post, Feedback, LIKE\_CHOICES, Comment, Tag  
**from** tinymce.widgets **import** TinyMCE  
  
  
**class** ContactForm(forms.ModelForm):  
 name = forms.CharField(max\_length=128, help\_text=**"Name"**)  
 email = forms.EmailField(max\_length=75, help\_text=**"Email"**)  
 subject = forms.CharField(max\_length=128, help\_text=**"Subject"**)  
 message = forms.CharField(widget=forms.Textarea,help\_text=**"Enter your message here"**)  
  
 **class** Meta:  
 model = Contact  
 fields = (**'name'**,**'email'**,**'subject'**,**'message'**)  
  
  
  
  
**class** CategoryForm(forms.ModelForm):  
 name = forms.CharField(max\_length=128, help\_text=**"Please enter category name."**)  
 slug = forms.CharField(widget=forms.HiddenInput(),required=**False**)  
  
 **class** Meta:  
 model = Category  
 fields = (**'name'**,)  
  
  
**class** PostForm(forms.ModelForm):  
 title = forms.CharField(max\_length=128)  
 body = forms.CharField(widget=TinyMCE(attrs={**'cols'**: 70, **'rows'**: 12}))  
 category = forms.ModelChoiceField(queryset = Category.objects.all())  
  
 **class** Meta:  
 model = Post  
 *#fields = ('title', 'body', 'category')* exclude = [**'slug'**]  
  
**class** TagForm1(forms.ModelForm):  
 tag\_title = forms.CharField(max\_length=100, help\_text=**"Tag Name"**)  
 slug = forms.CharField(widget=forms.HiddenInput(),required=**False**)  
 post = forms.MultipleChoiceField(required=**False**,  
 widget=forms.CheckboxSelectMultiple)  
  
 **class** Meta:  
 model = Tag  
 exclude = [**'slug'**]  
  
  
**class** TagForm(forms.Form):  
 tag\_title = forms.CharField(max\_length=100, help\_text=**"Use commas to separate tags."**)  
  
  
  
**class** CommentForm(forms.ModelForm):  
 **class** Meta:  
 model = Comment  
 *#fields = ('post','comment\_body','commented\_on','commenter',)* exclude =[**'post'**, **'commenter'**]  
  
  
**class** FeedbackForm(forms.ModelForm):  
 name = forms.CharField(max\_length=100, help\_text=**'Name'**)  
 email =forms.EmailField(help\_text=**'Your Email'**)  
 like\_dislike = forms.CharField(max\_length=2,  
 widget=forms.Select(choices=LIKE\_CHOICES))  
  
 feedback\_msg = forms.CharField(widget=forms.Textarea,help\_text=**"Enter your message here"**)  
  
 **class** Meta:  
 model = Feedback  
 fields = (**'name'**, **'email'**, **'like\_dislike'**, **'feedback\_msg'**,)

ADMIN.PY

**from** django.contrib **import** admin  
**from** blog.models **import** Category, Post, Comment, Tag, Contact, Feedback  
  
**class** CategoryAdmin(admin.ModelAdmin):  
 list\_display = (**'name'**,)  
 prepopulated\_fields = {**'slug'**:(**'name'**,)}  
  
  
**class** PostAdmin(admin.ModelAdmin):  
 list\_display = (**'title'**, **'body'**, **'posted'**, **'category'**)  
 prepopulated\_fields = {**'slug'**:(**'title'**,)}  
  
  
**class** TagAdmin(admin.ModelAdmin):  
 prepopulated\_fields = {**'slug'**:(**'tag\_title'**,)}  
  
**class** CommentAdmin(admin.ModelAdmin):  
 list\_display = (**'post'**,**'commented\_on'**,**'comment\_body'**,**'commenter'**)  
  
**class** ContactAdmin(admin.ModelAdmin):  
 list\_display = (**'name'**,**'email'**,**'subject'**,**'message'**)  
  
  
**class** FeedbackAdmin(admin.ModelAdmin):  
 list\_display = (**'name'**,**'email'**,**'like\_dislike'**,**'feedback\_msg'**)  
  
admin.site.register(Category, CategoryAdmin)  
admin.site.register(Post, PostAdmin)  
admin.site.register(Comment, CommentAdmin)  
admin.site.register(Tag, TagAdmin)  
admin.site.register(Contact, ContactAdmin)  
admin.site.register(Feedback, FeedbackAdmin)