Django Admin Panel

Django Admin panel

Faisal Memon



Django Admin Panel

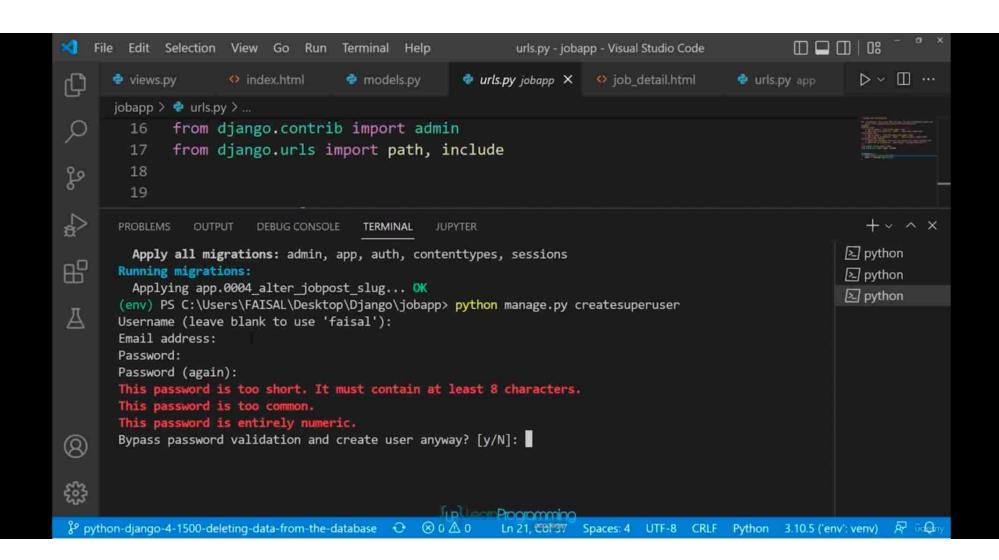
- Terminal may not be the best way to manage data.
- You need to provide them an interface which can be use to manage content on the website.
- ➤ Django provides an module called "The Django Admin" interface which can be used by site admins to manage content across the website.
- You can access it via <site-domain>/admin.

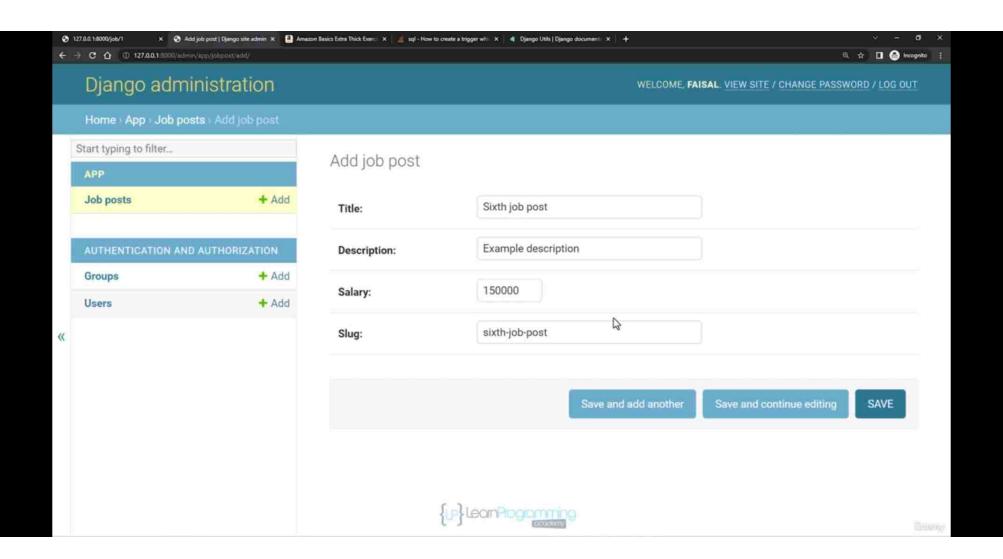


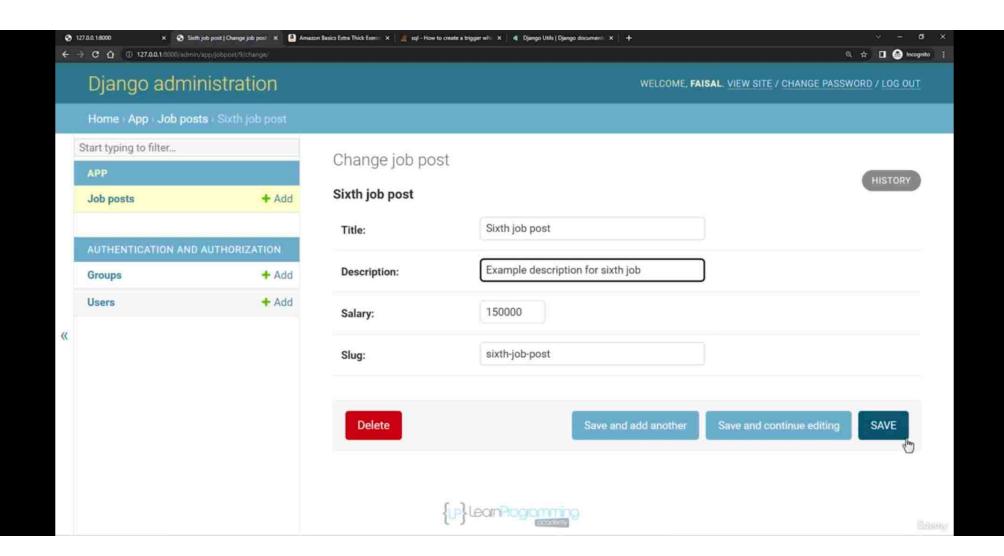
Django Admin Panel

- > Django was initially written for a news website.
- Django solves the problem of creating a unified interface for site administrators to edit content.
- The admin isn't intended to be used by site visitors. It's for site managers.
- > The inbuilt Django admin interface saves a lot of development efforts.









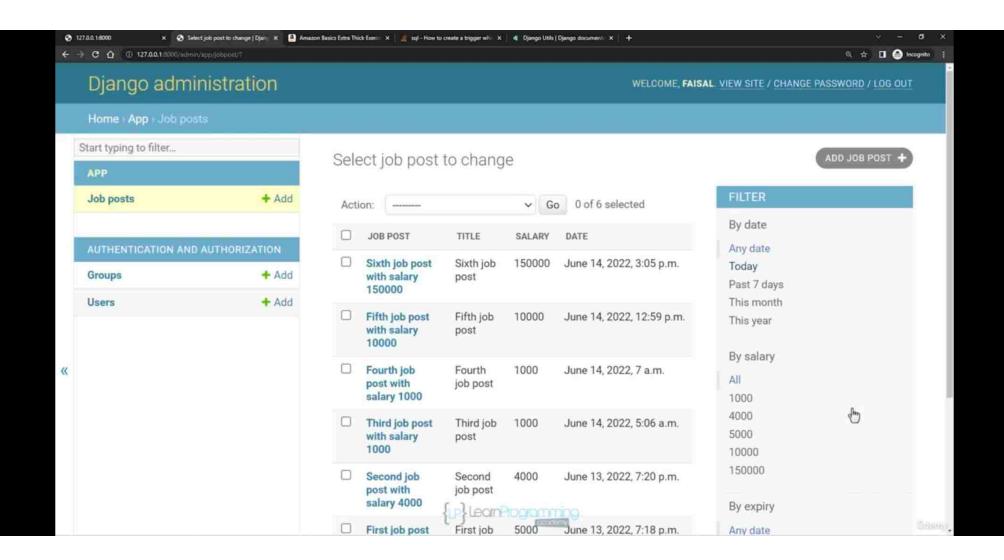
Customizing List Page

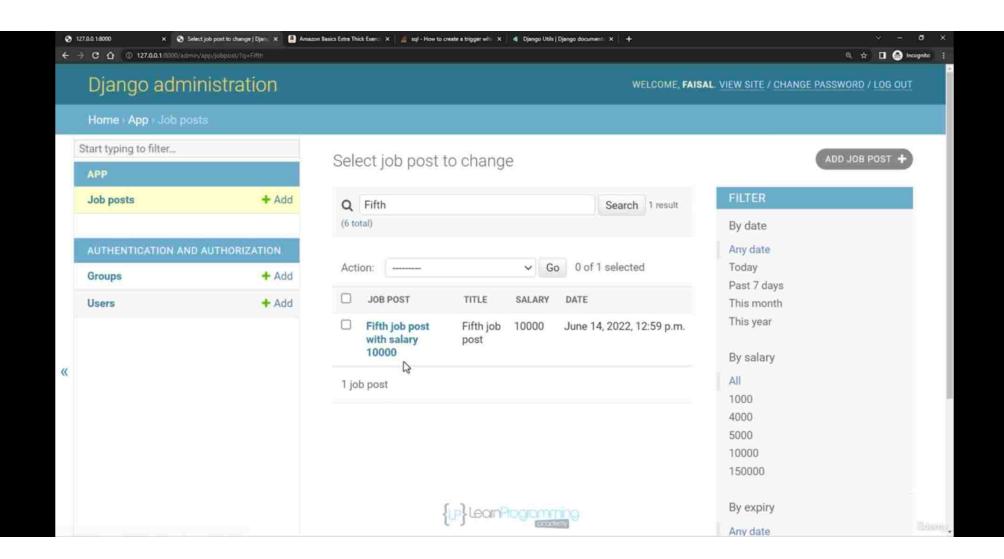
- > Change the way records are being displayed.
- > Better control over filters.
- Customize the options that are displayed in actions menu.

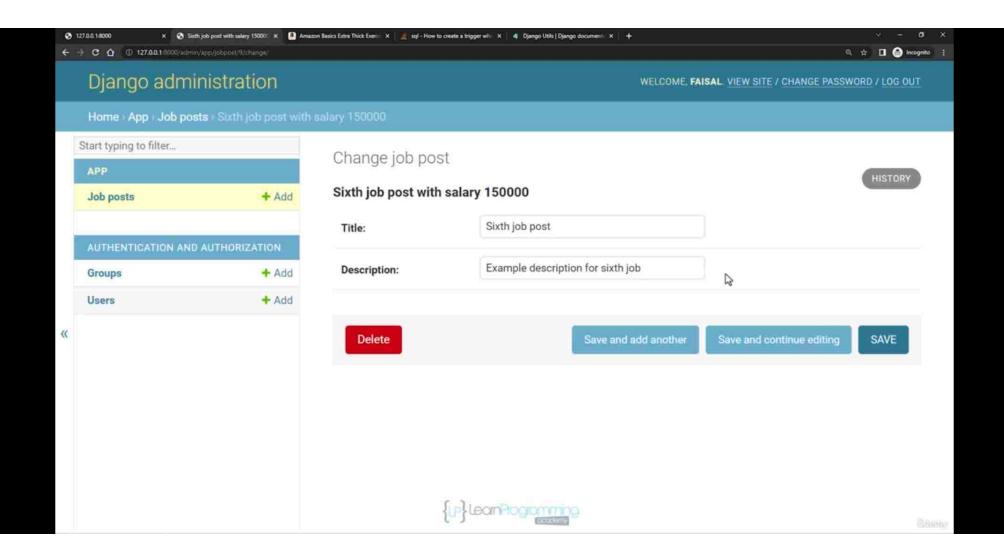
Customizing Detail Page

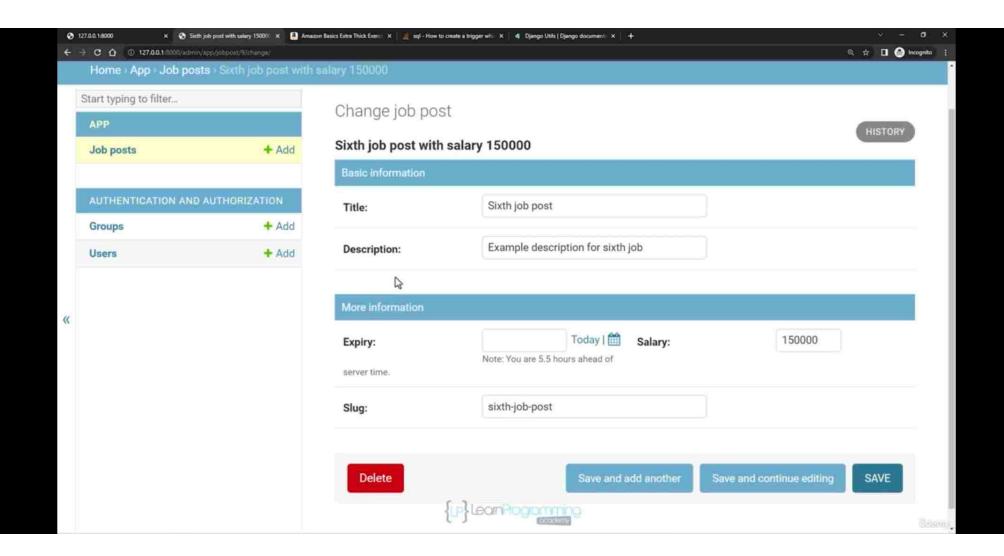
- > Customize and have control on what is being displayed.
- When working with multiple models, admin interface allows you to show related records along inline right in the form.

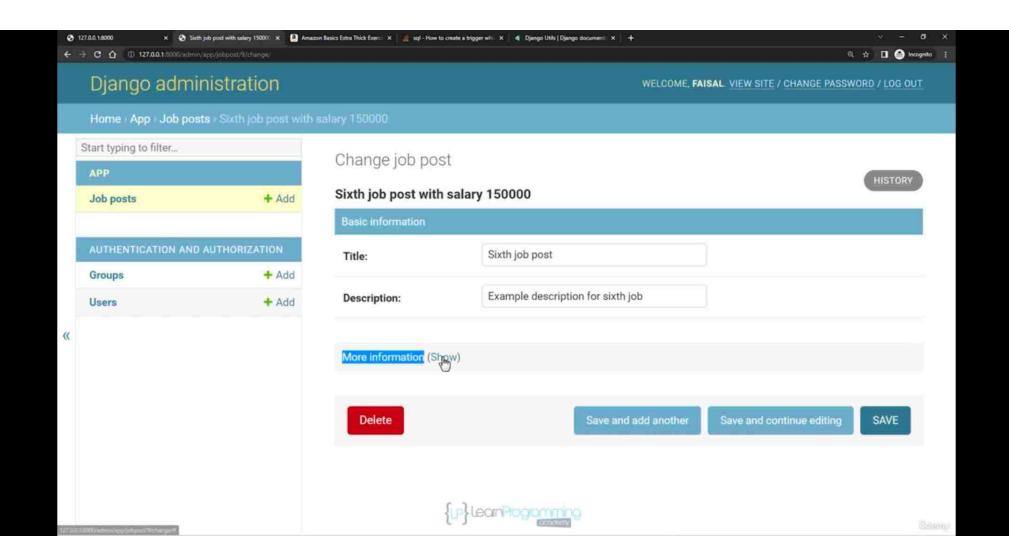












Enabling models in Admin panel

```
from app.models import JobPost
# Register your models here.
admin.site.register(JobPost)
```

Create superuser

python manage.py createsuperuser



admin.py

```
from django.contrib import admin
from app.models import JobPost

# Register your models here.
admin.site.register(JobPost)
```

admin.py

```
from django.contrib import admin
from app.models import JobPost

class JobAdmin(admin.ModelAdmin):
    pass

# Register your models here.
admin.site.register(JobPost, JobAdmin)
```



models.py

```
class JobAdmin(admin.ModelAdmin):
    list_display = ('title', 'salary', 'date')
    list_filter = ('date', 'salary', 'expiry')
    search_fields= ['title', 'description']
    search_help_text = "Write in your query and hit enter"
```



Control what is displayed using fields option

```
fields = ("title", "description")
```

Group similar fields

```
fields = (("title", "description"), "expiry")
```



Using fieldsets

```
fieldsets = (
          (<section name here>, {
                'fields': ('field 1, 'field 2)
          }),
          )
```

