In the view, ensure that you don’t include **created\_by** in the list of fields to edit, and override [**form\_valid()**](https://docs.djangoproject.com/en/2.1/ref/class-based-views/mixins-editing/#django.views.generic.edit.ModelFormMixin.form_valid) to add the user:

from django.views.generic.edit import CreateView

from myapp.models import Author

class AuthorCreate(CreateView):

model = Author

fields = ['name']

def form\_valid(self, form):

form.instance.created\_by = self.request.user

return super().form\_valid(form)

Compare the following two classes:*they are identical*

In these particular examples, we had to override the **form\_valid()** method so as to set some extra fields such as the **updated\_by** and **updated\_at**.

class PostUpdateView(UpdateView):

# compressed

success\_url = '/'

def form\_valid(self, form):

form.instance.updated\_by = self.request.user

form.instance.updated\_at = timezone.now()

return super(PostUpdateView, self).form\_valid(form)

In second example, I am already saving the object and returning a redirect response, calling super(PostUpdateView, self).form\_valid(form) is just repeating the code I have already written. It is fine to remove it.

<https://stackoverflow.com/questions/56029008/when-i-need-to-remove-superpostupdateview-self-form-validform-from-form>

class PostUpdateView(UpdateView):

# compressed

def form\_valid(self, form):

post = form.save(commit=False)

post.updated\_by = self.request.user

post.updated\_at = timezone.now()

post.save()

return redirect('board:view\_topic',

pk=post.topic.board.pk,

topic\_pk=post.topic.pk)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

def form\_valid(self, form):

form.instance.author = self.request.user

return super().form\_valid(form)