I am developing an ads web site using Django 3.0 and Python 3.8. I want to build multi step form wizard using django session. I have tried previously formtools.wizard but it failed to satisfy all of my requirements. Thus, I decided to write my own code. To do that, I used session to pass form inputs from one class view to another. The first form go through with no error. However, I got the following error message before second form was rendered:

```
Object of type Country is not JSON serializable
```

The view classes are as follow:

```
class PostWizardStepOne(View):

form_class = CommonForm

template_name = "towns/salehslist/ads_main_form.html"

wizard_data = {}

def get(self, request, *args, **kwargs):
   initial = {
           'wizard_data':request.session.get('wizard_data', None),
       }
   form = self.form_class(initial=initial)
   return render(request, self.template_name, {'form': form})

def post(self, request, *args, **kwargs):
   form = self.form_class(request.POST)
   print(request.POST)
   if form.is_valid():

      for k, v in form.cleaned_data.items():
         self.wizard_data[k] = v
      request.session['wizard_data'] = self.wizard_data
      request.session.modified = True


      print(self.wizard_data)
      return HttpResponseRedirect('PostWizardSecondStep')

   return render(request, self.template_name, {'form': form})
class PostWizardStepTow(View):

template_name = "towns/salehslist/forms/jobPostForm.html"

def get(self, request, *args, **kwargs):
   print(request.session['wizard_data'])

   return render(request, self.template_name, {})
```
Here are the urls:

```
path('post/', PostWizardStepOne.as_view(), name = 'PostWizardFirstStep'),
path('post/', PostWizardStepTow.as_view(), name = 'PostWizardSecondStep'),
```
Here are the forms:

```
class CommonForm(forms.ModelForm):


class Meta:
    model = Job

    fields = [

            'country',
            'province',
            'city',
            'category',
            'sub_category',
        ]

class JobForm(forms.ModelForm):

# to remove colons from the labels:
def __init__(self, *args, **kwargs):
    kwargs.setdefault('label_suffix', '')
    super(JobForm , self).__init__(*args, **kwargs)

class Meta:
    model = Job

    fields = [

          'employer',
          'title',
          'description',
          'Experience',
          'Education',
          'compensation',
          'employment_type',
        ]
```

```
class JobImagesForm(forms.Form):


# to remove colons from the labels:
def __init__(self, *args, **kwargs):
    kwargs.setdefault('label_suffix', '')
    super(JobImagesForm , self).__init__(*args, **kwargs)

    self.fields['image'].widget.attrs.update({ 'type':'file',
        'accept':'image/*',})

class Meta:

    model = JobImages

    fields = [
        'image',
    ]
```

Those are the models;

```
# country model
class Country(models.Model):
    name = models.CharField(max_length=64, unique=True)
    currency = models.CharField(max_length=16)

    def __str__(self):
        return "%s" % (self.name)

    class Meta:
            verbose_name_plural = "countries"


    class Job(models.Model):
    id = models.AutoField(primary_key=True)
    posted_by = models.ForeignKey(settings.AUTH_USER_MODEL,
    on_delete=models.CASCADE)
    employer = models.CharField(max_length=64)
    country = models.ForeignKey(Country, on_delete=models.CASCADE)
    province = models.ForeignKey(Province, on_delete=models.CASCADE)
    city = models.ForeignKey(City, on_delete=models.CASCADE)

    category = models.ForeignKey(Category, on_delete=models.CASCADE)
    sub_category = models.ForeignKey(SubCategory, on_delete=models.CASCADE)

    title = models.CharField(max_length=128)
    description = RichTextField(max_length=65536)
    Experience = RichTextField(max_length=65536)
    Education = RichTextField(max_length=65536)

    compensation = models.CharField(max_length=65536, blank=True)
    employment_type = models.ForeignKey(Employment_type, on_delete=models.CASCADE)

    date_created = models.DateTimeField(auto_now =False, auto_now_add=True)
    date_updated = models.DateTimeField(auto_now=True, auto_now_add=False)



    def __str__(self):
        return "%s %s %s" % (self.id, self.employer, self.title,)


    def username(self):
        return self.posted_by.first_name

    def categoryName(self):
        return self.category.name

    def subCategoryName(self):
        return self.sub_category.name

    def country_name(self):
        return self.country.name

    def province_name(self):
        return self.province.name

    def city_name(self):
        return self.city.name

    class Meta:
        verbose_name_plural = "jobs"
```
Here are the settings for backend:

```
INSTALLED_APPS = [ 'django.contrib.sessions', ]

MIDDLEWARE = ['django.contrib.sessions.middleware.SessionMiddleware',]
```

I am trying to save the 5 inputs of the first form into 1 variable called "wizard_data". and add to it the other inputs value from second and third forms, respectively, and finally save all of them into the data base and clear the session.

Note:

There are 4 steps as follow:

1- First step : Common form. always used (this form has 5 inputs) as follow:
 1- country, 2- province, 3-city, 4- category, 5- sub_category
2- Second step will use one of following forms based on category input selection in first step form:
 1- JobForm, or 2- ForSaleCarsTrucksForm, or 3- ForSaleOthersForm or 4- RealStatesForm, 5- CommunityForm, or 6- ServicesForm or 7- ResumesForm

There is extra condition to select between ForSaleCarsTrucksForm or ForSaleOthersForm. If user selected “for sale” category and then for subcategory “car” will be ForSaleCarsTrucksForm any other sub_category selection will be ForSaleOthersForm

3- Third step: will be also based on category selection in first step, will be one of following form for images:
 1- JobImagesForm, or 2- ForSaleCarsTrucksImagesForm, or 3- ForSaleOthersImagesForm or 4- RealStatesImagesForm, 5- CommunityImagesForm, or 6- ServicesImagesForm or 7- ResumesImagesForm

4- Fourth step will include the following:
 1- Revalidate the whole form and save it to the data base
 2- clear the session
 3- destroy all temporary uploaded images in file system

Note:

1- must set a life time for a session to 20 minutes, once expires clear it from the data base
2- if the user left the form unsubmitted and did not close the browser, the session wan’t expire and also the uploaded images will remain in the file system. Therefore, I need you to consider this case and take care of it while you are coding
3- You must use class-based view