SIMPLEISBETTERTHANCOMPLEX

By **Vitor Freitas**

I'm a passionate software developer and researcher from Brazil, currently living in Finland. I write about Python, Django and Web Development on a weekly basis. Read more.

Q&A

Ask Vitor #1: Getting form data to appear in URL and for use in the next view

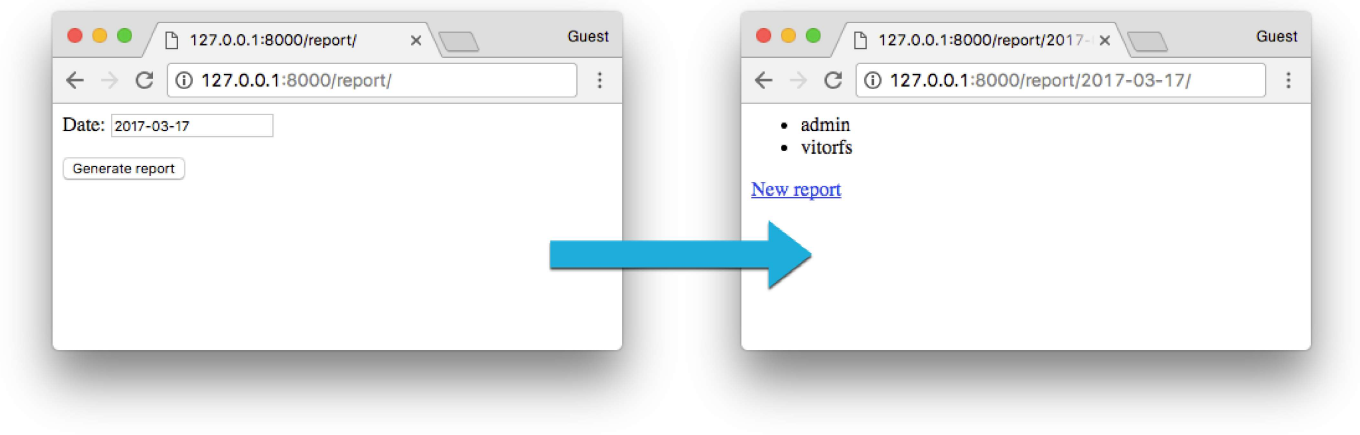
Mar 17, 2017 5 minutes read 7 comments 8,971 views

> *Devon Moore asks:*
>
> I want to have the user specify a date in a custom form. This date will append the current URL with the date value `path/YYYY-MM-DD/` I then need to capture the date and use it to filter data from the database to display that date's data.

> I'm using class based views and for the report I'm using generic view since this view is a custom report I'm building using multiple db models.

Answer

Here is what Devon wants to achieve:



There are a couple of ways to achieve the desired result. Using class-based views, we could perhaps do something like this:

forms.py

```
class ReportForm(forms.Form):
    date = forms.DateField()
```

urls.py

```
from django.conf.urls import url
from core import views  # import your views using the correct app name

urlpatterns = [
    url(r'report/$', views.Report.as_view(), name='report'),
    url(r'report/(?P<year>[0-9]{4})-(?P<month>[0-9]{2})-(?P<day>[0-9]{2})/$',
        views.ReportDetails.as_view(), name='report_details'),
]
```

Example of valid URL matching the `report_details` pattern: `/report/2017-03-17/`. If you want to change the URL to use slashes instead of dashes (`/report/2017/03/17/`), change the URL pattern to this:

```
    url(r'report/(?P<year>[0-9]{4})/(?P<month>[0-9]{2})/(?P<day>[0-9]{2})/$',
        views.ReportDetails.as_view(), name='report_details'),
```

views.py

```python
from django.contrib.auth.models import User
from django.views import View
from django.views.generic.edit import FormView
from django.utils.dateformat import format

class Report(FormView):
    template_name = 'report.html'
    form_class = ReportForm

    def form_valid(self, form):
        date = form.cleaned_data.get('date')
        year = date.year
        month = format(date, 'm')
        day = format(date, 'd')
        return redirect('report_details', year, month, day)

class ReportDetails(View):
    def get(self, request, year, month, day):
        users = User.objects.filter(date_joined__year__gte=int(year))
        # do your thing here, filter the data etc
        return render(request, 'report_details.html', {'users': users})
```

The view `Report` is responsible just for validating the user input and redirecting the user to the view that will actually process the report, which is `ReportDetails`.

The form is posted to `Report`. `Report` validates the input and if it is valid, it fires a redirect towards the `ReportDetails`. `ReportDetails` grab the date information from the URL, process the querysets and finally returns to the user, rendering the template.

report.html

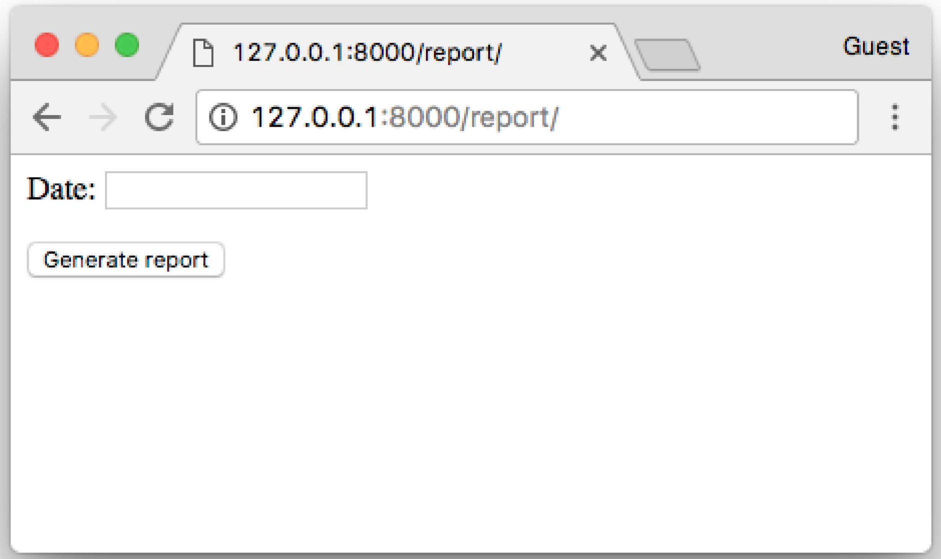
```html
{% extends 'base.html' %}

{% block content %}
  <form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Generate report</button>
  </form>
{% endblock %}
```

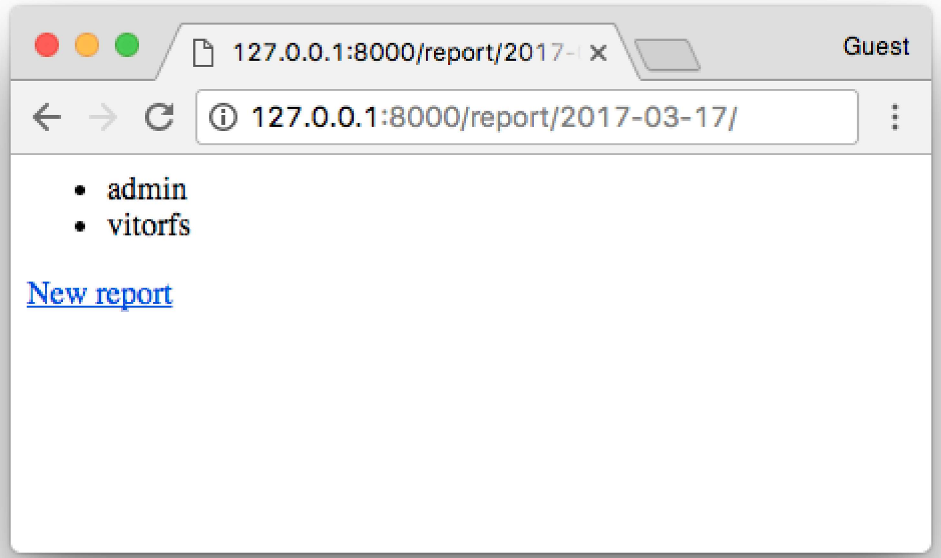
report_details.html

```html
{% extends 'base.html' %}
{% block content %}
  <ul>
  {% for user in users %}
    <li>{{ user.username }}</li>
  {% endfor %}
  </ul>
  <a href="{% url 'report' %}">New report</a>
{% endblock %}
```

The final result would be something like this:



Filtering with the proper URL:


127.0.0.1:8000/report/2017-
Guest
127.0.0.1:8000/report/2017-03-17/

- admin
- vitorfs

New report

Caveats

This implementation will only work if you only need the date to filter the report.

If you need to pass extra information to do the filtering of the querysets, I would recommend sending the form data directly to the `Report` view. And perhaps even using a GET request, because you are not modifying the data.

Something like this:

urls.py

```
urlpatterns = [
    url(r'report/$', views.Report.as_view(), name='report'),
]
```

views.py

```
class Report(View):
    def get(self, request):
        if 'date' in request.GET:  # only try to filter if `date` is in the querystring
            form = ReportForm(request.GET)
            if form.is_valid():
                # process the report
                date = form.cleaned_data.get('date')
                invoices = Invoices.objects.filter(date__year=date.year)
            return render(request, 'report_details.html', {'invoices': invoices})
    else:
        form = ReportForm()
    return render(request, 'report.html', {'form': form})
```

Then you would end up having a URL like this: `/report/?date=2017-03-17`. And if you had more information in the form, like the status, it would append in the URL: `/report/?date=2017-03-17&status=pending`.

The URL would still be friendly, and the implementation would be simpler.

Get the Code

The code is available on GitHub: github.com/sibtc/askvitor.

Related Posts

How to Implement Grouped Model Choice Field

Advanced Form Rendering with Django Crispy Forms

How to Use Bootstrap 4 Forms With Django

django forms urls cbv askvitor

Share this post

7 Comments Simple is Better Than Complex 1 Login

Recommend 5 Tweet Share Sort by Best

Join the discussion…

LOG IN WITH OR SIGN UP WITH DISQUS ?

Name

J • 2 years ago

Great post!
I would rather go with passing get args as you said. It's better for the future you when things get more complicated.

1 ^ | ˅ • Reply • Share ›

Devon Moore • 2 years ago

I've been stuck for days! Thank you so much for this!

1 ^ | ˅ • Reply • Share ›

Suman Astani • 4 months ago

The better approach is to manipulate url and pass extra query param to it. and access it in the request.GET

^ | ˅ • Reply • Share ›

Manish Kumar • 6 months ago

Is that same goes, with taking Valid url from users and return them with their title of the website

^ | ˅ • Reply • Share ›

Manish Kumar • 6 months ago

As usual... One of the best sources

^ | ˅ • Reply • Share ›

john benjamin zeus • 2 years ago

is it better to use class based generic views or function based views

^ | ˅ • Reply • Share ›

J ➦ john benjamin zeus • 2 years ago

There are certain things that don't fit properly with CBV. Or at least, are harder to implement. Particularly when you go out from crud operations

^ | ˅ • Reply • Share ›

ALSO ON **SIMPLE IS BETTER THAN COMPLEX**

Django Authentication Video Tutorial

11 comments • 9 months ago

Avatar **Vitor Freitas** — Next video coming up I will be adding bootstrap 4 to the forms :D

How to Implement Grouped Model Choice Field

4 comments • 7 months ago

Avatar **charettes** — Hey it's great to see this solution is being used in the wild :)One thing I might suggest is that you either

How to Use Bootstrap 4 Forms With Django

30 comments • a year ago

A Complete Beginner's Guide to Django - Part 3

166 comments • 2 years ago

Avatar real mah47 — Thank you ^-^

Avatar RAJAT MAAN — I am stuck with a error that is indentation and i am unable to correct it from 2 days so i thought to ask help from all …

Subscribe to our Mailing List

Receive updates from the Blog!

Your email address

SUBSCRIBE

Popular Posts

How to Extend Django User Model

How to Setup a SSL Certificate on Nginx for a Django Application

How to Deploy a Django Application to Digital Ocean