Chapter-8

Session Management

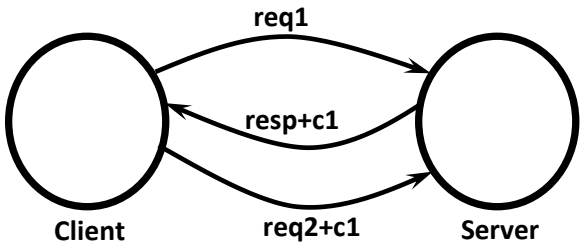
Session Management:

- Client and Server can communicate with some common language which is nothing but HTTP.
- The basic limitation of HTTP is, it is stateless protocol. i.e it is unable to remember client information for future purpose across multiple requests. Every request to the server is treated as new request.
- Hence some mechanism must be required at server side to remember client information across multiple requests.This mechanism is nothing but session management mechanism.
- The following are various session management mechanisms.

1) Cookies
2) Session API
3) URL Rewriting
4) Hidden Form Fields etc

Session Management By using Cookies:

Cookie is a very small amount of information created by Server and maintained by client.



Whenever client sends a request to the server,if server wants to remember client information for the future purpose then server will create cookie object with the required information. Server will send that Cookie object to the client as the part of response. Client will save that cookie in its local machine and send to the server with every consecutive request. By accessing cookies from the request server can remember client information.

How to Test our Browser Supports Cookies OR not:

We have to use the following 3 methods on the request object.

1) set_test_cookie()
2) test_cookie_worked()
3) delete_test_cookie()

views.py

```
1)  from django.shortcuts import render
2)  from django.http import HttpResponse
3)
4)  # Create your views here.
5)  def index(request):
6)      request.session.set_test_cookie()
7)      return HttpResponse('<h1>index Page</h1>')
8)
9)  def check_view(request):
10)     if request.session.test_cookie_worked():
11)         print('cookies are working properly')
12)         request.session.delete_test_cookie()
13)         return HttpResponse('<h1>Checking Page</h1>')
```

Note: Before executing this program compulsory we should perform migrate.

Application-1 Session Management by using Cookies (PageCount Application)

views.py

```
1)  from django.shortcuts import render
2)
3)  # Create your views here.
4)  def count_view(request):
5)      if 'count' in request.COOKIES:
6)          newcount=int(request.COOKIES['count'])+1
7)      else:
8)          newcount=1
9)      response=render(request,'testapp/count.html',{'count':newcount})
10)     response.set_cookie('count',newcount)
11)     return response
```

count.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)      <style >
7)      span{
```

```
8)       font-size: 200px;
9)       font-weight: 900;
10)    }
11)
12)    </style>
13)  </head>
14)  <body>
15)    <h1>Page Count is: <span> {{count}}<span></h1>
16)  </body>
17) </html>
```

Application-2 Session Management by using Cookies

```
1)  from django.shortcuts import render
2)  from testapp.forms import LoginForm
3)  import datetime
4)
5)  # Create your views here.
6)  def home_view(request):
7)      form=LoginForm()
8)      return render(request,'testapp/home.html',{'form':form})
9)
10) def date_time_view(request):
11)     # form=LoginForm(request.GET)
12)     name=request.GET['name']
13)     response=render(request,'testapp/datetime.html',{'name':name})
14)     response.set_cookie('name',name)
15)     return response
16)
17) def result_view(request):
18)     name=request.COOKIES['name']
19)     date_time=datetime.datetime.now()
20)     my_dict={'name':name,'date_time':date_time}
21)     return render(request,'testapp/result.html',my_dict)
```

home.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
```

```
8)    <h1>Welcome to DURGASOFT</h1>
9)    <form  action="/second">
10)    {{form.as_p}}
11)    {%csrf_token%}
12)    <input type="submit" name="" value="Enter Name">
13)   </form>
14)  </body>
15) </html>
```

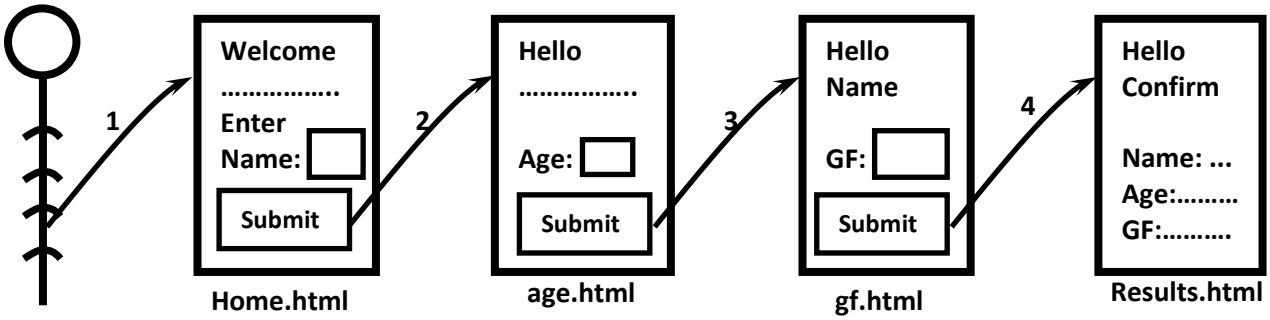
datetime.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)   <head>
4)    <meta charset="utf-8">
5)    <title></title>
6)   </head>
7)   <body>
8)    <h1>Hello {{name}}</h1> <hr>
9)    <a href="/result">Click Here to get Date and Time</a>
10)  </body>
11) </html>
```

result.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)   <head>
4)    <meta charset="utf-8">
5)    <title></title>
6)   </head>
7)   <body>
8)    <h1>Hello {{name}}</h1><hr>
9)    <h1>Current Date and Time:{{date_time}}</h1>
10)   <a href="/result">Click Here to get Updated Date and Time</a>
11)  </body>
12) </html>
```

Application-3 Session Management by using Cookies



views.py

```
1)  from django.shortcuts import render
2)
3)  # Create your views here.
4)  def name_view(request):
5)      return render(request,'testapp/name.html')
6)
7)  def age_view(request):
8)      name=request.GET['name']
9)      response=render(request,'testapp/age.html',{'name':name})
10)     response.set_cookie('name',name)
11)     return response
12)
13) def gf_view(request):
14)     age=request.GET['age']
15)     name=request.COOKIES['name']
16)     response=render(request,'testapp/gf.html',{'name':name})
17)     response.set_cookie('age',age)
18)     return response
19)
20) def results_view(request):
21)     name=request.COOKIES['name']
22)     age=request.COOKIES['age']
23)     gfname=request.GET['gfname']
24)     response=render(request,'testapp/results.html',{'name':name,'age':age,'
    gfname':gfname})
25)     response.set_cookie('gfname',gfname)
26)     return response
```


DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com

name.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Welcome to DURGASOFT</h1>
9)      <form action='/age'>
10)       Enter Name: <input type="text" name="name" value=""><br><br>
11)       <input type="submit" name="" value="Submit Name">
12)     </form>
13)   </body>
14) </html>
```

age.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Hello {{name}}..</h1><hr>
9)        <form action='/gf'>
10)         Enter Age: <input type="text" name="age" value=""><br><br>
11)         <input type="submit" name="" value="Submit Age">
12)       </form>
13)   </body>
14) </html>
```

gf.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Hello {{name}}..</h1><hr>
9)        <form action='/results'>
```

```
10)     Enter Girl Friend Name: <input type="text" name="gfname" value=""><br><br>
11)      <input type="submit" name="" value="Submit GFName">
12)     </form>
13)  </body>
14) </html>
```

results.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Hello {{name}} Thanks for providing info</h1>
9)      <h2>Please cross check your data and confirm</h2><hr>
10)    <ul>
11)      <li>Name:{{name}}</li>
12)      <li>Age:{{age}}</li>
13)      <li>Girl Friend Name:{{gfname}}</li>
14)    </ul>
15)  </body>
16) </html>
```

Limitations of Cookies:

1) By using cookies we can store very less amount of information. The size of the cookie is fixed.Hence if we want to store huge amount of information then cookie is not best choice.
2) Cookie can hold only string information. If we want to store non-string objects then we should not use cookies.
3) Cookie information is stored at client side and hence there is no security.
4) Everytime with every request, browser will send all cookies related to that application,which creates network traffic problems.
5) There is a limit on the max number of cookies supported by browser.

To overcome all these limitations we should go for sessions.

Temporary vs Permanent Cookies:

- If we are not setting any max_age for the cookie,then the cookies will be stored in browsers cache.Once we closed browser automatically the cookies will be expired.Such type of cookies are called temporary Cookies.
- We can set temporary Cookie as follows: `response.set_cookie(name,value)`

- If we are setting max_age for the cookie,then cookies will be stored in local file system permanently.Once the specified max_age expires then only cookies will be expired.Such type of cookies are called permanent or persistent cookies. We can set Permanent Cookies as follows

```
response.set_cookie(name,value,max_age=180)
response.set_cookie(name,value,180)
```

- The time unit for max_age is in seconds.

Application-4 Session Management by using Cookies (Shopping Cart Application)

views.py

```
1)  from django.shortcuts import render
2)  from testapp.forms import ItemAddForm
3)
4)  # Create your views here.
5)  def index(request):
6)      return render(request,'testapp/home.html')
7)  def additem(request):
8)      form=ItemAddForm()
9)      response=render(request,'testapp/additem.html',{'form':form})
10)     if request.method=='POST':
11)         form=ItemAddForm(request.POST)
12)         if form.is_valid():
13)             name=form.cleaned_data['itemname']
14)             quantity=form.cleaned_data['quantity']
15)             response.set_cookie(name,quantity,180)
16)             # return index(request)
17)     return response
18) def displayitem_view(request):
19)     return render(request,'testapp/showitems.html')
```

home.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <!-- Latest compiled and minified CSS -->
```


```
6)  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/cs
    s/bootstrap.min.css" integrity="sha384-
    BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" c
    rossorigin="anonymous">
7)    <title></title>
8)   </head>
9)   <body>
10)   <div class="container" align='center'>
11)    <div class="jumbotron">
12)      <h1>DURGASOFT ONLINE SHOPPING APP</h1>
13)      <a class="btn btn-primary btn-lg" href="/add"
    role="button">ADD ITEM</a>
14)      <a class="btn btn-primary btn-lg" href="/display"
     role="button">Display ITEMS</a>
15)    </div>
16)   </div>
17)  </body>
18) </html>
```

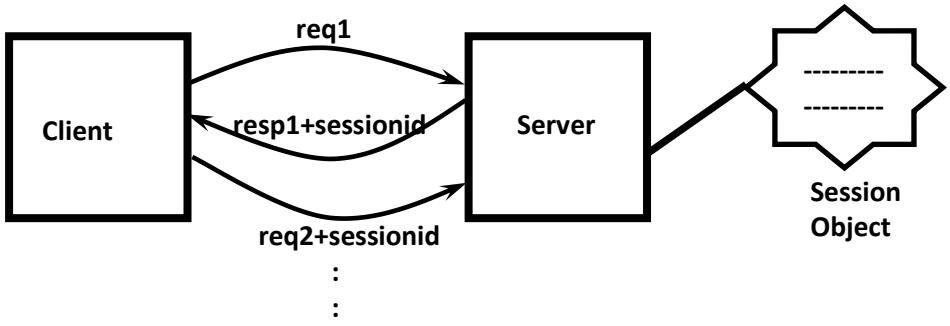
additem.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)   <head>
4)    <meta charset="utf-8">
5)    <title></title>
6)    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7
    /css/bootstrap.min.css" integrity="sha384-
    BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
    crossorigin="anonymous">
7)   </head>
8)   <body>
9)    <div class="container" align='center'>
10)    <h1>Add Item Form</h1>
11)    <form  method="post">
12)     {{form.as_p}}
13)     {%csrf_token%}
14)     <input type="submit" name="" value="Add Item">
15)    </form><br><br><br>
16)    <a class="btn btn-primary btn-lg" href="/display"
    role="button">Display ITEMS</a>
17)   </div>
18)  </body>
19) </html>
```

showitems.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Total Cookies Information:</h1>
9)      {%if request.COOKIES %}
10)    <table border=2>
11)      <thead>
12)        <th>Cookie Name</th>
13)        <th>Cookie Value</th>
14)      </thead>
15)      {% for key,value in request.COOKIES.items %}
16)      <tr>
17)        <td>{{key}}</td>
18)        <td>{{value}}</td>
19)      </tr>
20)      {% endfor %}
21)      </table>
22)     {%else%}
23)     <p>Cookie Information is not available</p>
24)    {%endif%}
25) </body>
26) </html>
```

Session Management By using Session API (Django Session Framework):



- Once client sends request to the server,if server wants to remember client information for the future purpose then server will create session object and store required information in that object. For every session object a unique identifier available which is nothing but sessionid.
- Server sends the corresponding session id to the client as the part of response. Client retrieves the session id from the response and save in the local file system. With every consecutive request client will that session id. By accessing that session id and corresponding session object server can remember client. This mechanism is nothing but session management by using session api.

Note: Session information will be stored in one of following possibilities

1. Inside a File
2. Inside a database
3. Inside Cache

The most straight forward approach is to use django.contrib.sessions application to store session information in a Django Model/database.

The Model Name is: django.contrib.sessions.models.Session

Note: To use this approach compulsory the following applicaiton should be configured inside INSTALLED_APPS list of settings.py file.

django.contrib.sessions

If it is not there then we can add, but we have to synchronize database

python manage.py syncdb

Note:

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

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

Useful Methods for Session Management:

1) request.session['key'] = value
   To Add Data to the Session.

2) value = request.session['key']
   To get Data from the Session

3) request.session.set_expiry(seconds)
   Sets the expiry Time for the Session.

4) request.session.get_expiry_age()
   Returns the expiry age in seconds(the number of seconds until this session expire)

5) request.session.get_expiry_date()
   Returns the data on which this session will expire

Note: Before using session object in our application, compulsory we have to migrate. Otherwise we will get the following error.
no such table: django_session

Application-1 Session Management by using Session API (PageCount Application)

views.py

```
1)  from django.shortcuts import render
2)
3)  # Create your views here.
4)  def page_count_view(request):
5)      count=request.session.get('count',0)
6)      newcount=count+1
7)      request.session['count']=newcount
8)      print(request.session.get_expiry_age())
9)      print(request.session.get_expiry_date())
10)     return render(request,'testapp/pagecount.html',{'count':newcount})
```

pagecount.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
```

```
5)     <title></title>
6)     <style >
7)     span{
8)       font-size: 300px;
9)     }
10)    </style>
11)  </head>
12)  <body>
13)    <h1>The Page Count:<span>{{count}}</span></h1>
14)  </body>
15) </html>
```

Application-2 Session Management by using Session API (Profile Application)

forms.py

```
1)  from django import forms
2)  class NameForm(forms.Form):
3)     name=forms.CharField()
4)
5)  class AgeForm(forms.Form):
6)     age=forms.IntegerField()
7)
8)  class GFForm(forms.Form):
9)     gf=forms.CharField()
```

views.py

```
1)  from django.shortcuts import render
2)  from testapp.forms import *
3)
4)  # Create your views here.
5)  def name_view(request):
6)     form=NameForm()
7)     return render(request,'testapp/name.html',{'form':form})
8)
9)  def age_view(request):
10)    name=request.GET['name']
11)    request.session['name']=name
12)    form=AgeForm()
13)    return render(request,'testapp/age.html',{'form':form})
14)
15) def gf_view(request):
```

```
16)    age=request.GET['age']
17)    request.session['age']=age
18)    form=GFForm()
19)    return render(request,'testapp/gf.html',{'form':form})
20)
21) def results_view(request):
22)    gf=request.GET['gf']
23)    request.session['gf']=gf
24)    return render(request,'testapp/results.html')
```

name.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Name Registration Form</h1><hr>
9)      <form  action="/age" >
10)       {{form}}
11)       {%csrf_token%}<br><br>
12)       <input type="submit" name="" value="Submit Name">
13)     </form>
14)   </body>
15) </html>
```

age.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Age Registration Form</h1><hr>
9)      <form  action="/gf" >
10)       {{form}}
11)       {%csrf_token%}<br><br>
12)       <input type="submit" name="" value="Submit Age">
13)     </form>
14)   </body>
15) </html>
```

gf.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      <h1>Girl Friend Registration Form</h1><hr>
9)      <form  action="/results" >
10)       {{form}}
11)       {%csrf_token%}<br><br>
12)       <input type="submit" name="" value="Submit GF Info">
13)     </form>
14)   </body>
15) </html>
```

results.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)    <head>
4)      <meta charset="utf-8">
5)      <title></title>
6)    </head>
7)    <body>
8)      {%if request.session %}
9)      <h1>Thanks for providing information..Plz confirm once</h1>
10)     <ul>
11)       {%for key,value in request.session.items %}
12)       <li> <h2>{{key|upper}}:    {{value|title}}</h2> </li>
13)       {%endfor%}
14)     </ul>
15)     {%else%}
16)     <p>No Information available</p>
17)     {%endif%}
18) </body>
19) </html>
```

Application-3 Session Management by using Session API (Shopping Cart Application)

settings.py

```
1) DATABASES = {
2)     'default': {
3)         'ENGINE': 'django.db.backends.mysql',
4)         'NAME': 'employeedb',
5)         'USER':'root',
6)         'PASSWORD':'root'
7)     }
8) }
```

forms.py

```
1) from django import forms
2) class AddItemForm(forms.Form):
3)     name=forms.CharField()
4)     quantity=forms.IntegerField()
```

views.py

```
1) from django.shortcuts import render
2) from testapp.forms import *
3)
4) # Create your views here.
5) def add_item_view(request):
6)     form=AddItemForm()
7)     if request.method=='POST':
8)         name=request.POST['name']
9)         quantity=request.POST['quantity']
10)        request.session[name]=quantity
11)    return render(request,'testapp/additem.html',{'form':form})
12)
13) def display_items_view(request):
14)    return render(request,'testapp/displayitems.html')
```

additem.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <meta charset="utf-8">
```

```
5)    <title></title>
6)    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7
    /css/bootstrap.min.css" integrity="sha384-
    BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" c
    rossorigin="anonymous">
7)   </head>
8)   <body>
9)    <div class="container" align='center'>
10)    <h1>Add Item Form</h1>
11)    <form method='POST'>
12)     {{form.as_p}}
13)     {%csrf_token%}
14)     <input type="submit" name="" value="Add Item">
15)    </form><br><br><br>
16)    <a class="btn btn-primary btn-lg"
    href="/display" role="button">Display ITEMS</a>
17)   </div>
18)  </body>
19) </html>
```

displayitems.html

```
1)  <!DOCTYPE html>
2)  <html lang="en" dir="ltr">
3)   <head>
4)    <meta charset="utf-8">
5)    <title></title>
6)   </head>
7)   <body>
8)    <h1>Your Shopping Cart Information:</h1>
9)    {%if request.session %}
10)   <table border=2>
11)    <thead>
12)     <th>Item Name</th>
13)     <th>Quantity</th>
14)    </thead>
15)    {% for key,value in request.session.items %}
16)    <tr>
17)     <td>{{key}}</td>
18)     <td>{{value}}</td>
19)    </tr>
20)    {% endfor %}
21)    </table>
22)   {%else%}
23)    <p>No Items in your shopping cart</p>
```

```
24)   {%endif%}
25) </body>
26) </html>
```

Note: The default max_age of the session is 14 days. But based on our requirement we can set our own expiry time.

```
request.session.set_expiry(time in seconds)
```

Important Methods related to Session:

1) set_expiry(time_in_seconds)

- If we are not performing any operation on the session specified amount of time (max inactive interval) then session will be expired automatically.

```
request.session.set_expiry(120)
```

- If we set 0 as argument, then the session will expire once we closed browser.

2) get_expiry_age()
3) get_expiry_date()

Note: We can observer that 0 value and 120 values are perfectly working in our application

```
1)  if request.method=='POST':
2)      name=request.POST['name']
3)      quantity=request.POST['quantity']
4)      request.session[name]=quantity
5)      request.session.set_expiry(0)
```

How to Delete Session Data:

```
del request.session[sessionkey]

for key in request.session.keys():
  del request.session[key]
```

In settings.py File

```
SESSION_SAVE_EVERY_REQUEST=True
```

OR

In views.py

```
request.session.modified = True
```

Browser Length Sessions and Persistent Sessions

- If the session information stored inside browsers cache such type of sessions are called browser length sessions.
- If the session information stored persistently inside file/database/cache, such type of sessions are called Persistent sessions.

Note: By default sessions are persistent sessions.