МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

дисциплина «Средства и технологии разработки информационных систем»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ

«Разграничение доступа»

ВАРИАНТ № 22

Выполнил:

студент группы ИТП-31, Пронуза М.Ю.

Принял:

преподаватель Карась О.В.

Гомель 2024

**1. Цель работы:** освоить методы разграничения доступа к функциональности веб приложений на Django.

**2. Ход работы и результаты.**

Задание: в проекте, выполненном в лабораторной работе №9 реализовать функции регистрации, авторизации и аутентификации. Реализовать две роли администратор и пользователь. Согласно индивидуальному заданию реализовать ограничения доступа к некоторым функциям приложения.

Вариант задания представлен на рисунке 1.

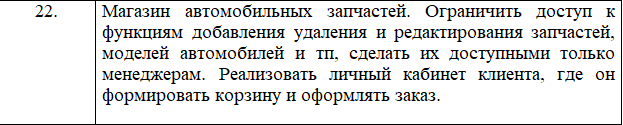
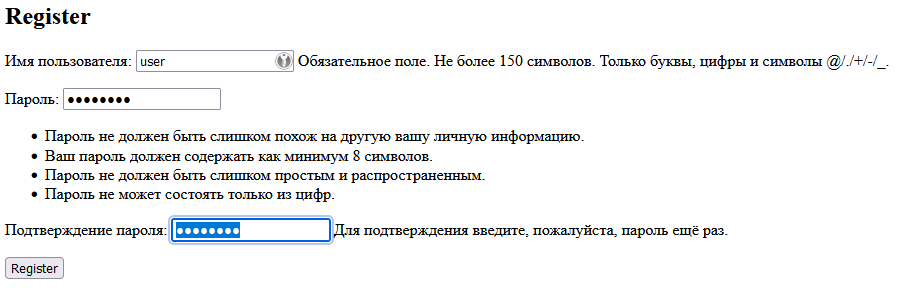


Рисунок 1 - вариант задания

Регистрация представлена на рисунке 2.

Рисунок 2 - регистрация

Вход представлен на рисунке 3.

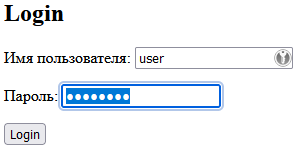
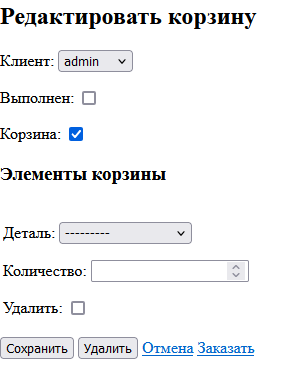


Рисунок 3 - вход

Редактирование корзины представлен на рисунке 4.

Рисунок 4 - редактирование корзины

Код *djangoapp/urls.py* представлен в листинге A.

Код *djangoapp/settings.py* представлен в листинге Б.

Код *magazine/urls.py* представлен в листинге В.

Код *magazine/views.py* представлен в листинге Г.

Код *templates/car\_list.html* представлен в листинге Д.

Код *magazine/models.py* представлен в листинге Е.

Код *templates/spare\_list.html* представлен в листинге Ё.

Код *magazine/forms.py* представлен в листинге Ж.

Код *templates/add\_car.html* представлен в листинге З.

Код *templates/add\_spare.html* представлен в листинге И.

Код *templates/order\_list.html* представлен в листинге Й.

Код *templates/order.html* представлен в листинге К.

Код *templates/completed\_orders\_sum.html* представлен в листинге Л.

Код *authdj/models.py* представлен в листинге М.

Код *authdj/urls.py* представлен в листинге Н.

Код *authdj/views.py* представлен в листинге О.

Код *templates/login.html* представлен в листинге П.

Код *templates/register.html* представлен в листинге Р.

**3. Выводы:** были освоены методы разграничения доступа к функциональности веб приложений на Django.

**Листинг А**

from django.contrib import admin

from django.urls import include, path

urlpatterns = [

path('admin/', admin.site.urls),

path('', include("magazine.urls"))

]

**Листинг Б**

STATIC\_URL = 'static/'

STATICFILES\_DIRS = [

os.path.join(BASE\_DIR, "static"),

]

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': ['templates'],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'magazine'

]

**Листинг В**

from django.urls import path

from . import views

urlpatterns = [

path("carlist/", views.car\_list, name="car\_list"),

path('sparelist/<int:car\_id>/', views.spare\_list, name='spare\_list'),

path('spare/<int:spare\_id>/', views.SpareManagementView.as\_view(), name='spare\_management'),

path('spare/', views.SpareManagementView.as\_view(), name='spare\_add'),

path('car/<int:car\_id>/', views.car\_management, name='car\_management'),

path('car/', views.car\_management, name='car\_add'),

path('orderlist/', views.order\_list, name='order\_list'),

path('order/<int:order\_id>/', views.order\_management, name='order\_management'),

path('order/', views.order\_management, name='order\_add'),

path('completed\_orders\_sum/', views.completed\_orders\_sum, name='completed\_orders\_sum'),

path('', views.HomeView.as\_view(), name='home')

]

**Листинг Г**

from django.shortcuts import render, redirect, get\_object\_or\_404

from django.db.models import F, Sum

from django.views.generic import TemplateView

from django.views.generic.edit import FormMixin

from django.urls import reverse\_lazy

from magazine.forms import SpareForm, CarForm, OrderForm, OrderItemFormSet, DateRangeForm

from magazine.models import Spare, Car, Order, OrderItem

class HomeView(TemplateView):

template\_name = 'home.html'

def spare\_list(request, car\_id):

car = Car.objects.get(pk=car\_id)

spare\_list = Spare.objects.filter(car=car)

return render(request, 'spare\_list.html', {'car': car,'spare\_list': spare\_list})

def car\_list(request):

car\_list\_ = Car.objects.all()

return render(request, 'car\_list.html', {'car\_list': car\_list\_})

class SpareManagementView(FormMixin, TemplateView):

model = Spare

template\_name = 'spare.html'

form\_class = SpareForm

success\_url = reverse\_lazy('car\_list')

def get\_object(self):

spare\_id = self.kwargs.get('spare\_id')

if spare\_id:

return get\_object\_or\_404(Spare, id=spare\_id)

return None

def get\_context\_data(self, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

context['spare'] = self.get\_object()

if context['spare']:

context['form'] = self.form\_class(instance=context['spare'])

else:

context['form'] = self.form\_class()

return context

def post(self, request, \*args, \*\*kwargs):

spare = self.get\_object()

if spare:

form = self.form\_class(request.POST, instance=spare)

else:

form = self.form\_class(request.POST)

if 'delete' in request.POST and spare:

spare.delete()

return redirect(self.success\_url)

if form.is\_valid():

form.save()

return redirect(self.success\_url)

return self.form\_invalid(form)

def car\_management(request, car\_id=None):

if car\_id:

car = get\_object\_or\_404(Car, id=car\_id)

if request.method == 'POST':

if 'delete' in request.POST:

car.delete()

return redirect('car\_list')

else:

form = CarForm(request.POST, instance=car)

if form.is\_valid():

form.save()

return redirect('car\_list')

else:

form = CarForm(instance=car)

else:

if request.method == 'POST':

form = CarForm(request.POST)

if form.is\_valid():

form.save()

return redirect('car\_list')

else:

form = CarForm()

return render(request, 'car.html', {'form': form, 'car': car if car\_id else None})

def order\_list(request):

order\_list\_ = Order.objects.all()

return render(request, 'order\_list.html', {'order\_list': order\_list\_, 'qqq': len(order\_list\_)})

def order\_management(request, order\_id=None):

order = get\_object\_or\_404(Order, id=order\_id) if order\_id else None

if request.method == 'POST':

if order\_id:

order\_form = OrderForm(request.POST, instance=order)

order\_item\_formset = OrderItemFormSet(request.POST, instance=order)

else:

order\_form = OrderForm(request.POST)

order\_item\_formset = OrderItemFormSet(request.POST)

if 'delete' in request.POST:

if order:

order.delete()

return redirect('order\_list')

if order\_form.is\_valid() and order\_item\_formset.is\_valid():

order = order\_form.save()

order\_item\_formset.instance = order

order\_item\_formset.save()

return redirect('order\_list')

else:

if order\_id:

order\_form = OrderForm(instance=order)

order\_item\_formset = OrderItemFormSet(instance=order)

else:

order\_form = OrderForm()

order\_item\_formset = OrderItemFormSet()

return render(request, 'order.html', {

'order\_form': order\_form,

'order\_item\_formset': order\_item\_formset,

'order': order

})

def completed\_orders\_sum(request):

total\_sum = None

form = DateRangeForm(request.POST or None)

if request.method == 'POST' and form.is\_valid():

start\_date = form.cleaned\_data['start\_date']

end\_date = form.cleaned\_data['end\_date']

total\_sum = OrderItem.objects.filter(

order\_\_completed=True,

order\_\_date\_\_range=(start\_date, end\_date)

).aggregate(total=Sum(F('count') \* F('spare\_\_price')))['total']

if total\_sum is None:

total\_sum = 0

return render(request, 'completed\_orders\_sum.html', {

'form': form,

'total\_sum': total\_sum

})

**Листинг Д**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Список машин</title>

</head>

<body>

<ul>

{% for i in car\_list %}

<li><a href="{% url 'spare\_list' car\_id=i.id %}">{{i.name}}</a> <a href="{% url 'car\_management' i.id %}">Редактировать</a>

</li>

{% endfor %}

</ul>

</body>

</html>

**Листинг Е**

from django.db import models

from django.contrib.auth.models import User

class Car(models.Model):

name = models.CharField('название', max\_length=100, unique=True)

def \_\_str\_\_(self):

return f"{self.name}"

class Spare(models.Model):

car = models.ForeignKey(Car, on\_delete=models.CASCADE, verbose\_name='машина')

name = models.CharField('название', max\_length=100)

price = models.PositiveIntegerField('цена')

def \_\_str\_\_(self):

return f"{self.name}"

class Order(models.Model):

user = models.ForeignKey(User, on\_delete=models.CASCADE, verbose\_name='клиент')

date = models.DateTimeField('дата заказа', auto\_now\_add=True)

completed = models.BooleanField('выполнен', default=False)

def \_\_str\_\_(self):

return f"{self.user}, {self.date}"

class OrderItem(models.Model):

order = models.ForeignKey(Order, on\_delete=models.CASCADE, verbose\_name='заказ')

spare = models.ForeignKey(Spare, on\_delete=models.CASCADE, verbose\_name='деталь')

count = models.PositiveIntegerField('количество')

class Meta:

constraints = [

models.UniqueConstraint(fields=['order', 'spare'], name='unique\_order\_spare')

]

def \_\_str\_\_(self):

return f"{self.order}, {self.spare}, {self.count}"

**Листинг Ё**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>{{car.name}}</title>

</head>

<body>

<h3>{{car\_name}}</h3>

{% for i in spare\_list %}

<p>{{i.name}}: {{i.price}} руб. <a href="{% url 'spare\_management' i.id %}">Редактировать</a></p>

{% endfor %}

</body>

</html>

**Листинг Ж**

from django import forms

from .models import Spare, Car, Order, OrderItem

class SpareForm(forms.ModelForm):

class Meta:

model = Spare

fields = '\_\_all\_\_'

class CarForm(forms.ModelForm):

class Meta:

model = Car

fields = '\_\_all\_\_'

class OrderForm(forms.ModelForm):

class Meta:

model = Order

fields = '\_\_all\_\_'

OrderItemFormSet = forms.inlineformset\_factory(

Order, OrderItem,

form=forms.ModelForm,

fields=['spare', 'count'],

extra=1,

can\_delete=True

)

class DateRangeForm(forms.Form):

start\_date = forms.DateField(widget=forms.TextInput(attrs={'type': 'date'}), label='Начальная дата')

end\_date = forms.DateField(widget=forms.TextInput(attrs={'type': 'date'}), label='Конечная дата')

**Листинг З**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Добавление машины</title>

</head>

<body>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Добавить машину</button>

</form>

</body>

</html>

**Листинг И**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Добавление детали</title>

</head>

<body>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Добавить деталь</button>

</form>

</body>

</html>

**Листинг Й**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Список заказов</title>

</head>

<body>

<h2>Список заказов</h2>

<ul>

{% for i in order\_list %}

<li><a href="{% url 'order\_management' order\_id=i.id %}">{{i.user}} - {{i.date}}</a></li>

{% endfor %}

</ul>

</body>

</html>

**Листинг К**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>{% if order %}Редактировать{% else %}Добавить{% endif %} заказ</title>

</head>

<body>

<h2>{% if order %}Редактировать{% else %}Добавить{% endif %} заказ</h2>

<form method="post">

{% csrf\_token %}

{{ order\_form.as\_p }}

<h3>Элементы заказа</h3>

{{ order\_item\_formset.management\_form }}

<table>

{% for form in order\_item\_formset %}

<tr>

<td>{{ form.as\_p }}</td>

</tr>

{% endfor %}

</table>

<button type="submit">{% if order %}Сохранить{% else %}Добавить{% endif %}</button>

{% if order %}

<button type="submit" name="delete">Удалить заказ</button>

{% endif %}

<a href="{% url 'order\_list' %}">Отмена</a>

</form>

</body>

</html>

**Листинг Л**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Сумма заказов</title>

</head>

<body>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Рассчитать</button>

</form>

{% if total\_sum is not None %}

<h2>Сумма выполненных заказов: {{ total\_sum }}</h2>

{% endif %}

</body>

</html>

**Листинг М**

from django.db import models

# Create your models here.

**Листинг Н**

from django.urls import path

from django.contrib.auth import views as auth\_views

from . import views

app\_name = 'authdj'

urlpatterns = [

path('register/', views.register, name='register'),

path('login/', auth\_views.LoginView.as\_view(template\_name='login.html'), name='login'),

path('logout/', views.logout\_view, name='logout'),

]

**Листинг О**

from django.shortcuts import render, redirect

from django.contrib.auth.forms import UserCreationForm

from django.contrib.auth import login, logout

def register(request):

if request.method == 'POST':

form = UserCreationForm(request.POST)

if form.is\_valid():

user = form.save()

login(request, user)

return redirect('home')

else:

form = UserCreationForm()

return render(request, 'register.html', {'form': form})

def logout\_view(request):

logout(request)

return redirect('home')

**Листинг П**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>login</title>

</head>

<body>

<h2>Login</h2>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Login</button>

</form>

</body>

</html>

**Листинг Р**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>register</title>

</head>

<body>

<h2>Register</h2>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Register</button>

</form>

</body>

</html>