

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»

ЛАБОРАТОРНАЯ РАБОТА №2 «РЕАЛИЗАЦИЯ ПРОСТОГО САЙТА СРЕДСТВАМИ DJANGO»

Выполнил:

Студент III курса ИМРиП

Группы: D33101

Ф.И.О.: Чжан Цзяи

Проверил:

Говоров Антон Игоревич

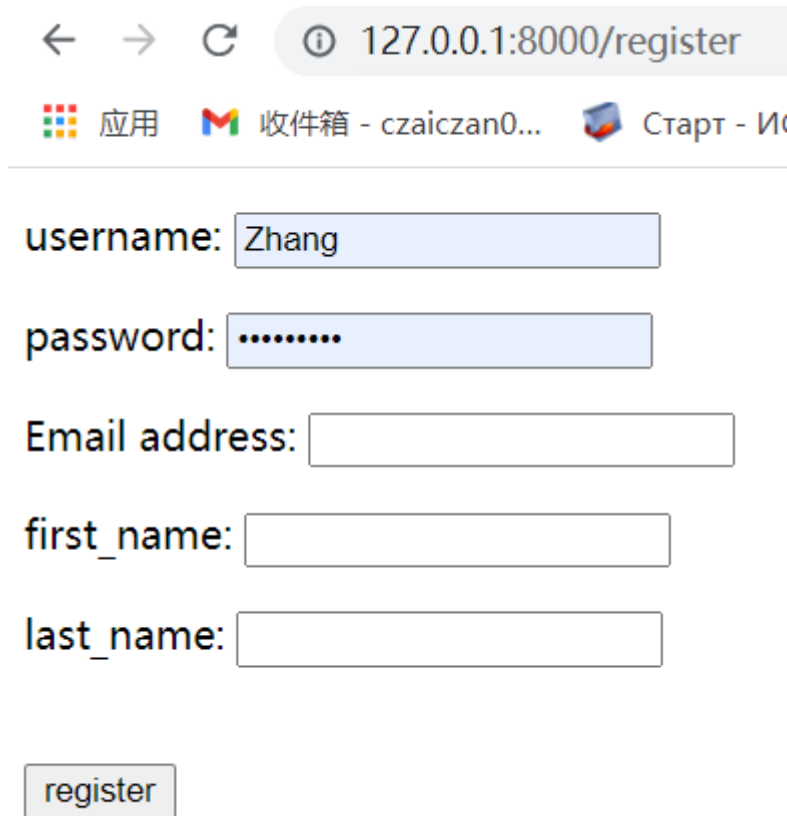
Санкт-Петербург
2021

1. Список отелей.

Необходимо учитывать название отеля, владельца отеля, адрес, описание, типы номеров, стоимость, вместимость, удобства.

Необходимо реализовать следующий функционал:

Регистрация новых пользователей.



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/register". Below the address bar, there are several tabs: "应用" (Applications), "收件箱 - czaiczano..." (Inbox - czaiczano...), and "Старт - И..." (Start - I...). The main content area of the browser displays a registration form with the following fields and labels:

- username:** A text input field containing the value "Zhang".
- password:** A text input field containing seven dots, indicating a masked password.
- Email address:** An empty text input field.
- first_name:** An empty text input field.
- last_name:** An empty text input field.
- register:** A button with the text "register".

Просмотр и резервирование номеров. Пользователь должен иметь возможность редактирования и удаления своих резервирований.



Hotel:

KHP

Owner: XiJinPing

Address: China

Description: Big hotel

Number room: [5](#)

Number room: [4](#)

Type: 2 человека

Price: 1000.0

Capacity: 100.0

Facilities: washroom

[Order](#) [Cancel](#)

Написание отзывов к номерам. При добавлении комментариев, должны сохраняться период проживания, текст комментария, рейтинг (1-10), информация о комментаторе.

Comment:

Point: 8/10

Date:

Oct. 1, 2021 - Oct. 1, 2077

User: Zhang

It's a big and good room

Add comment:

Comment:

- This field is required.

Point:

Begin time:

End time:

Администратор должен иметь возможность заселить пользователя в отель и выселить из отеля средствами Django-admin.

← → ↻ 127.0.0.1:8000/admin/

应用 收件箱 - czaiczao0... Старт - ИСУ ИТМ... Google 翻譯 Открытое образ... CS 65 CS 224 Личный кабинет...

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change
Users	+ Add	Change

HOTEL

Hotels	+ Add	Change
Rooms	+ Add	Change

Recent actions

My actions

- 1000.0, 100.0, washroom, 2 человека Room
- 10000.0, 1000.0, washroom TV, 4 человека Room
- 10000.0, 1000.0, washroom TV, 4 человека Room
- 1000.0, 100.0, washroom, 2 человека Room
- 10000.0, 1000.0, washroom TV, 4 человека Room
- 10000.0, 1000.0, washroom TV, 4 человека Room
- + 1000.0, 100.0, washroom, 2 человека Room
- + 10000.0, 1000.0, washroom TV, 4 человека Room
- + KHP, XiJinPing, China, Big hotel Hotel

В клиентской части должна формироваться таблица, отображающая постояльцев отеля за последний месяц.

<input type="checkbox"/>	Xiang	18181760569@163.com	1	2	✖
<input type="checkbox"/>	Zhang				✔

Код:

Models.py:

```
class Hotel(models.Model):
    name_hotel = models.CharField(max_length=50)
    owner_hotel = models.CharField(max_length=50)
    add_hotel = models.CharField(max_length=100)
    des_hotel = models.CharField(max_length=600)

    def __str__(self):
        return "{} , {} , {} , {}".format(self.name_hotel, self.owner_hotel, self.add_hotel, self.des_hotel)
```

```

class Room(models.Model):
    TYPES = [
        ('1', '1 человек'),
        ('2', '2 человека'),
        ('3', '3 человека'),
        ('4', '4 человека')
    ]
    type_room = models.CharField(max_length=1, choices=TYPES)
    price_room = models.FloatField()
    cap_room = models.FloatField()
    fac_room = models.CharField(max_length=200)
    hotel = models.ForeignKey(Hotel, on_delete=models.CASCADE)
    order_user = models.ForeignKey(User, on_delete=models.CASCADE, blank=True, null=True)

    def __str__(self):
        return "{}, {}, {}, {}".format(self.price_room, self.cap_room, self.fac_room, self.get_type_room_display())

```

```

class Comment(models.Model):
    POINT = (
        ('1', '1'),
        ('2', '2'),
        ('3', '3'),
        ('4', '4'),
        ('5', '5'),
        ('6', '6'),
        ('7', '7'),
        ('8', '8'),
        ('9', '9'),
        ('10', '10'),
    )
    text_comment = models.CharField(max_length=1600)
    point = models.CharField(max_length=2, choices=POINT)
    b_date = models.DateField()
    e_date = models.DateField()
    room = models.ForeignKey('Room', on_delete=models.CASCADE)
    user = models.ForeignKey(User, on_delete=models.CASCADE)

```

Views.py:

```

class HotelView(ListView):
    model = Hotel

    def get(self, request):
        hotel = Hotel.objects.get(pk=1)
        rooms = Room.objects.all()

        return render(request, 'hotel.html', {'hotel': hotel, 'rooms': rooms})

```

```

@login_required
def room_info(request, room_id):
    try:
        room = Room.objects.get(pk=room_id)
    except Room.DoesNotExist:
        pass

    try:
        comments = Comment.objects.filter(room=room_id).all()
    except Comment.DoesNotExist:
        pass

    if request.POST.get('user'):
        user_id = int(request.POST.get('user'))

    form = CommentForm(request.POST)

    if form.is_valid():
        new_form = form.save(commit=False)
        new_form.post = form
        new_form.room_id = room_id
        new_form.user_id = user_id
        new_form.save()

        return HttpResponseRedirect('/hotel/{}/'.format(room_id))

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

def register(request):
    registered = False

    if request.method == 'POST':
        create_user_form = UserForm(data=request.POST)

        if create_user_form.is_valid():
            user = create_user_form.save()
            user.set_password(user.password)
            user.save()
            registered = True

        else:
            print(create_user_form.errors)
    else:
        create_user_form = UserForm()

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

```

```

def user_login(request):
    if request.method == 'POST':

        username = request.POST.get('username')
        password = request.POST.get('password')

        user = authenticate(username=username, password=password)

        if user:
            if user.is_active:
                login(request, user)
                return HttpResponseRedirect('/hotel/')
            else:
                return HttpResponse("Incorrect username or password")

        else:
            return render(request, 'login.html', {})

```

```

def order_room(request, room_id):
    user = User.objects.get(id=request.user.id)
    try:
        room = Room.objects.get(pk=room_id)
    except Room.DoesNotExist:
        pass
    if room.order_user == user:
        message = 'You already order this room'
    elif room.order_user is not None and room.order_user != user:
        message = 'You can not order this room'
    else:
        message = 'You successfully order this room'
        room.order_user = User.objects.get(id=request.user.id)
        room.save()
    return render(request, 'order.html', {'message': message})

```



```
def cancel_room(request, room_id):
    user = User.objects.get(id=request.user.id)
    try:
        room = Room.objects.get(pk=room_id)
    except Room.DoesNotExist:
        pass
    if room.order_user != user:
        message = 'You did not order this room'
    elif room.order_user == user:
        message = 'You successfully cancel this room'
        room.order_user = None
        room.save()
    return render(request, 'cancel.html', {'message': message})
```

urls.py:

```
from django.urls import path
from .views import *
urlpatterns = [
    path('hotel/', HotelView.as_view()),
    path('hotel/<int:room_id>/', room_info),
    path('login', user_login),
    path('register', register),
    path('hotel/<int:room_id>/order', order_room),
    path('hotel/<int:room_id>/cancel', cancel_room),
```