НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Основы веб-программирования

Отчет по лабораторной работе №1
Вариант 3

Создание сайта на Django

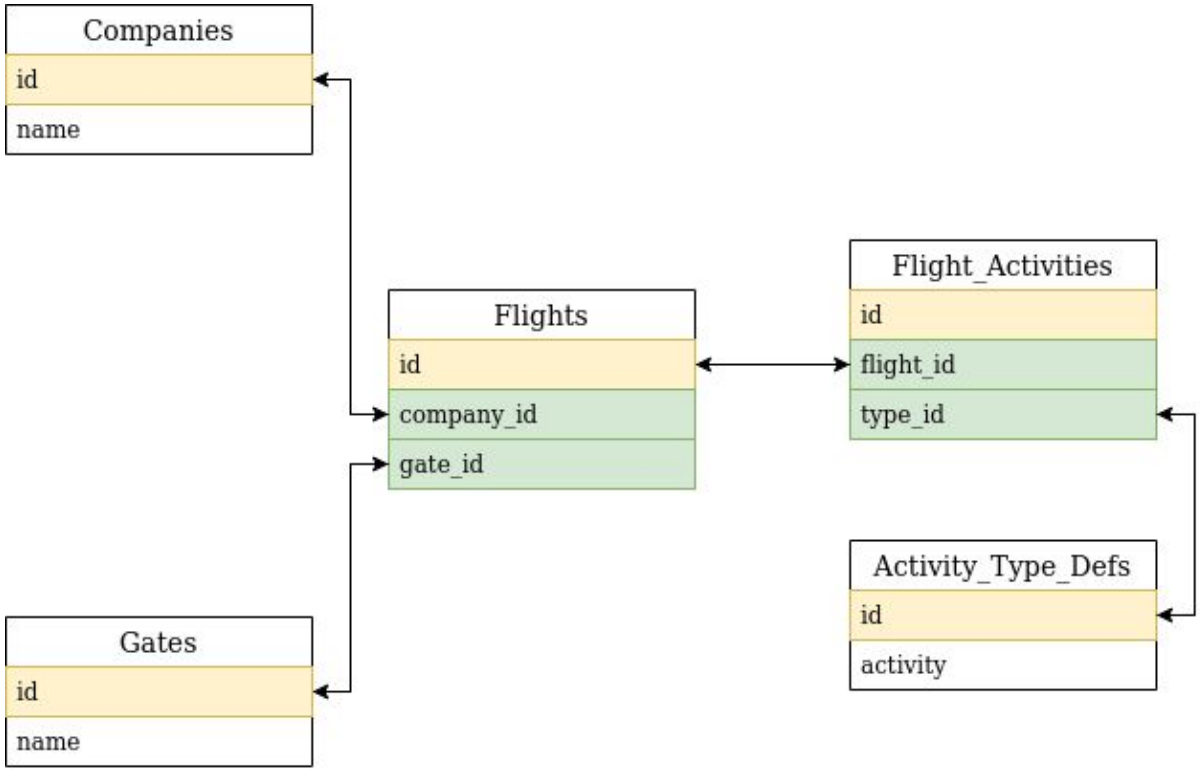
Выполнила:
Бережнова Марина
Группа: К3343

Проверил:
Говоров А. И.

2020 г.

Выполнение работы:

1. Для начала я решила отрисовать модель данных:



Максимальное распределение всех данных по сущностям несколько усложнит процесс разработки, зато всё будет изолировано друг от друга и понятно даже неподготовленному смотрящему.

Как можно понять, главной сущностью я выбрала таблицу Flights, в которой содержится основная информация по авиаперелетам.

2. models.py:

```python
from django.db import models

# Create your models here.
class Companies(models.Model):
    name = models.CharField(max_length=30)

    def __str__(self):
        return "{}".format(self.name)


class Gates(models.Model):
    name = models.CharField(max_length=30)

    def __str__(self):
        return "{}".format(self.name)


class Flights(models.Model):
    company = models.ForeignKey(Companies, on_delete=models.CASCADE)
    gate = models.ForeignKey(Gates, on_delete=models.CASCADE)

    def __str__(self):
        return "Company: {} | Gate: {}".format(self.company, self.gate)


class FlightActivities(models.Model):
    ACTIVITY = [
        ('0', 'arrival'),
        ('1', 'departure')
    ]

    flight = models.ForeignKey(Flights, on_delete=models.CASCADE)
    activity = models.CharField(choices=ACTIVITY, default='0', max_length=1)
    time = models.DateField()

    def __str__(self):
        return "{} | Arrival/departure: {} | Date {}".format(self.flight, self.get_activity_display(), self.time)
```

Сразу добавим в admin.py.

3. Теперь нужно создать суперпользователя, чтобы можно было добавлять данные, через админку:

   py manage.py createsuperuser

4. Проверяю, что всё создалось:

Всё ок.

5. Теперь создадим view для вывода всех данных по тому или иному полёту.

```python
from django.shortcuts import render
from .models import *
from django.http import Http404, HttpResponseRedirect
from django.views.generic.list import ListView

# Create your views here.

class FlightView(ListView):
    model = FlightActivities

    def get(self, request):
        context = {}

        try:
            context["flights"] = FlightActivities.objects.all()
        except Flight.DoesNotExist:
            raise Http404("Flight does not exist")

        return render(request, 'flight_list.html', context)


def show_flight(request, flight_id):
    try:
        flight = FlightActivities.objects.get(pk=flight_id)
    except Flight.DoesNotExist:
        raise Http404("Flight does not exist")

    return render(request, 'flight.html', {'flight': flight})
```

Я уже сделала для них простые шаблоны, теперь осталось добавить форму и связать это всё вместе. Для начала надо добавить модель для комментариев.

6. Она получилась вот такой:

```python
class FlightComments(models.Model):
    flight = models.ForeignKey(FlightActivities, on_delete=models.CASCADE)

    COMMENT_TYPE = [
        ('0', 'Gate changing'),
        ('1', 'Lateness'),
        ('2', 'Other')
    ]

    com_type = models.CharField(choices=COMMENT_TYPE, default='0', max_length=1)
    com_text = models.CharField(max_length=1024)
    author = models.EmailField(max_length=254)
```

7. Страница с комментариями по полёту:

8. Все полёты:

9. View для регистрации и авторизации:

```python
def register(request):
    registered = False
    if request.method == 'POST':
        user_form = UserForm(data=request.POST)

        if user_form.is_valid():
            user = user_form.save()
            user.set_password(user.password)
            user.save()

            registered = True

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

    return render(request,'registration.html', {'user_form': 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('/flights/')
            else:
                return HttpResponse("Your account was inactive.")
        else:
            print("Someone tried to login and failed.")
            print("They used username: {} and password: {}".format(username, password))
            return HttpResponse("Invalid login details given")
    else:
        return render(request, 'login.html', {})
```

Для остальных страничек поставила декоратор @login_required, чтобы пользователь изначально обязан был войти, прежде чем добавит комментарий.

Итоговый view для страницы полёта, на которой выводятся комментарии:

```
@login_required
def show_flight(request, flight_id):
    try:
        flight = FlightActivities.objects.get(pk=flight_id)
    except Flights.DoesNotExist:
        raise Http404("Flight does not exist")

    try:
        comments = FlightComments.objects.filter(flight=flight_id).all()
    except FlightComments.DoesNotExist:
        pass

    form = CommentForm(request.POST)

    if request.POST.get('author_id') != None:
        author_id = int(request.POST.get('author_id'))

    if form.is_valid():
        com_form = form.save(commit=False)
        com_form.post = form
        com_form.flight_id = flight_id
        com_form.author_id = author_id
        com_form.save()
        return HttpResponseRedirect('/flights/{}/'.format(flight_id))

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