

Министерство науки и высшего образования Российской Федерации
Пермский национальный исследовательский политехнический университет
Кафедра ИТАС

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5
по дисциплине «Теоретические основы автоматизированного
управления»
Тема: **Разработка прототипа программного обеспечения
автоматизированной системы**
Вариант №14

Выполнил студент гр. РИС-19-16
Миннахметов Эльдар Юлдашевич

Проверил доцент кафедры ИТАС
Полевщиков Иван Сергеевич

Пермь, 2021 год

1 Задание к работе

1. Разработать прототип приложения (с применением GUI) в соответствии с функциональными требованиями из лабораторной работы №2.

Приложение должно взаимодействовать с БД из лабораторной работы №4.

Необходимо самостоятельно выбрать стек технологий (языки и среды программирования, СУБД и т.д.) для реализации приложения.

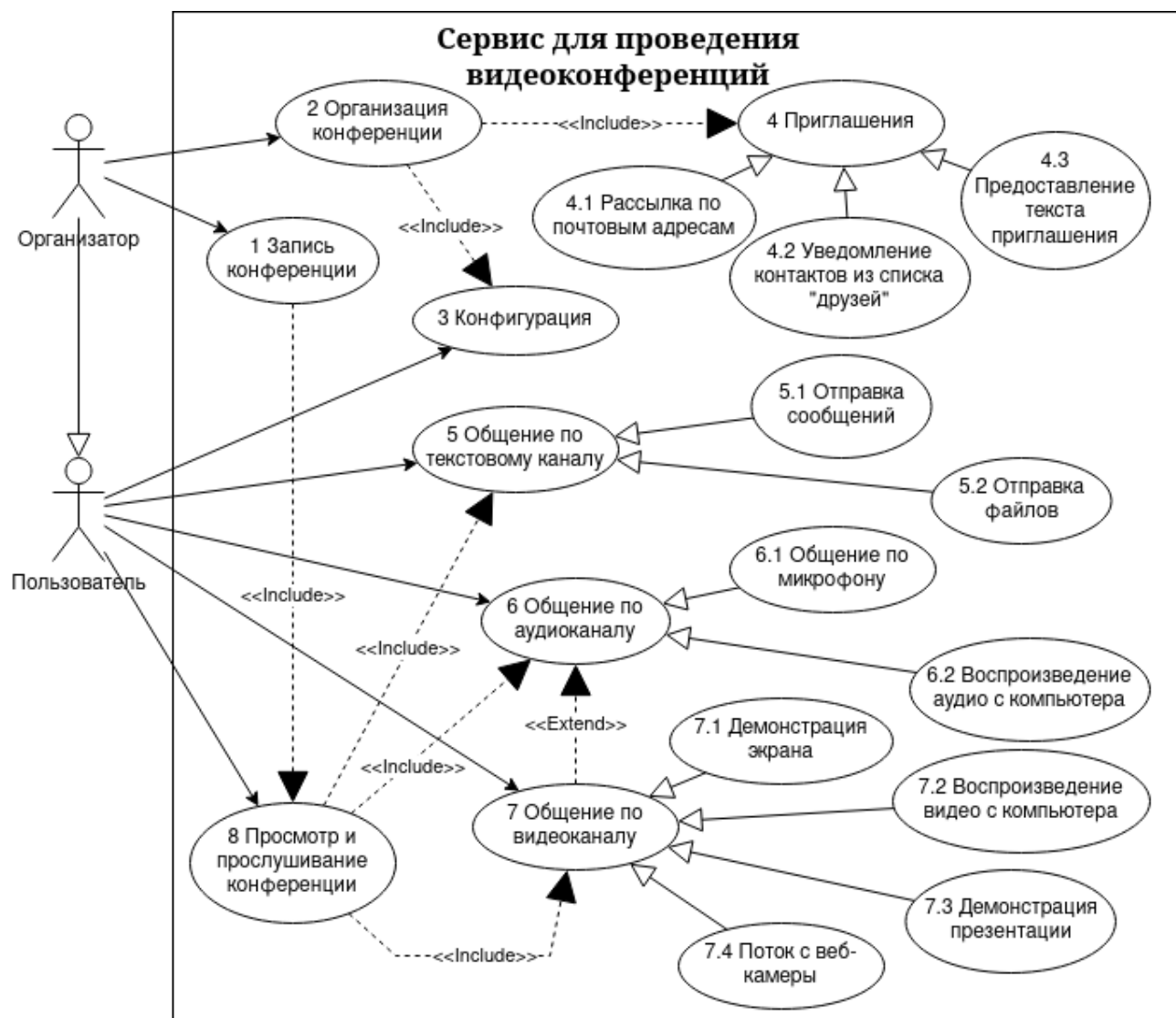
2. Продемонстрировать в отчете типовые сценарии работы приложения в соответствии с функциональными требованиями.

Примечание: в случае сложности реализации всех функциональных требований согласовать с преподавателем (по электронной почте) возможность реализации некоторых требований в упрощенной форме.

Программная система в соответствии с вариантом:

Сервис видеоконференций

2 Диаграмма вариантов использования UML



3 Описание вариантов использования

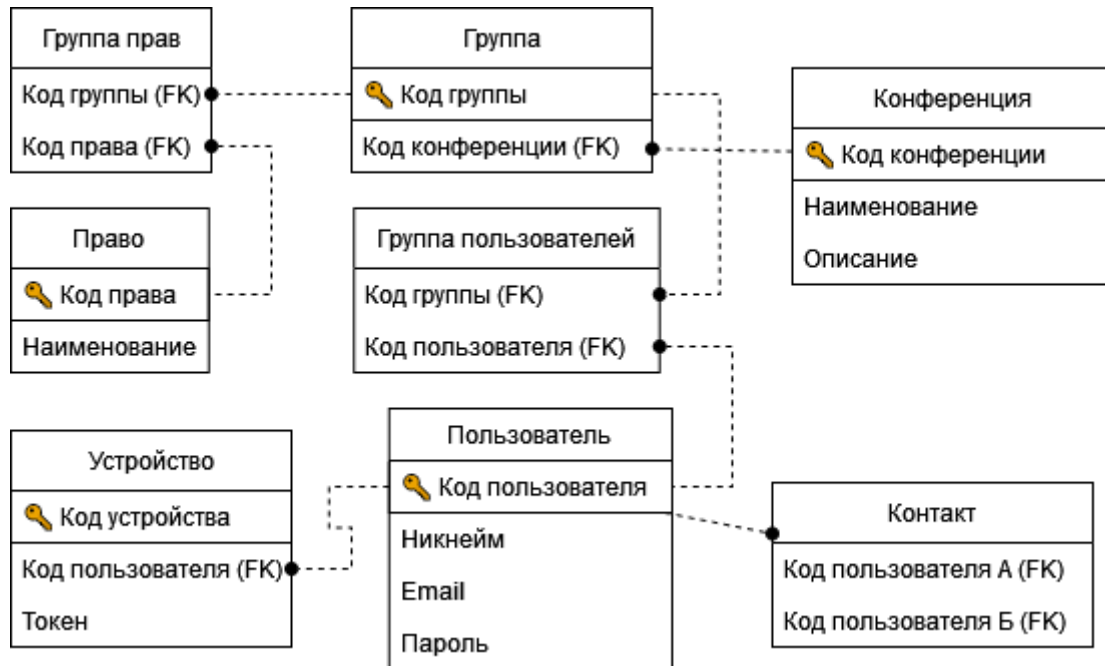
Вариант использования	Актёр, связанный с вариантом использования отношением ассоциации	Описание варианта использования
1 Запись конференции	Организатор	Организатор записывает видеоконференцию, получая текстовый, аудио- и видео- потоки, слитые воедино.
2 Организация конференции	Организатор	Создание конференции с учетом настроек в «3 Конфигурация» и рассылка приглашений в «4 Приглашения»
3 Конфигурация	Организатор, Пользователь	Организатор устанавливает права для пользователей и настройки записи. Пользователь считывает свои права, на основе которых может или не может пользоваться каналами общения
4 Приглашения	Организатор	Способ оповещения о начале конференции

4.1 Рассылка по почтовым адресам	Организатор	Организатор рассылает приглашения по почтам пользователей
4.2 Уведомление контактов из списка «друзей»	Организатор	Организатор рассылает приглашения пользователям в списке своих контактов
4.3 Предоставление текста приглашения	Организатор	Организатор получает текст приглашения, в котором содержится ссылка на вход в конференцию, а также указаны время начала и продолжительность и права для подключающихся пользователей
5 Общение по текстовому каналу	Пользователь	Пользователь по заданному праву пользуется текстовым каналом с помощью реализаций использования
5.1 Отправка сообщений	Пользователь	Пользователь пишет сообщения и отправляет их по текстовому каналу конференции
5.2 Отправка файлов	Пользователь	Пользователь выбирает файлы и отправляет их по текстовому каналу конференции
6 Общение по аудиоканалу	Пользователь	Пользователь по заданному праву пользуется аудиоканалом с помощью реализаций использования
6.1 Общение по микрофону	Пользователь	Пользователь пишет сообщения и отправляет их по текстовому каналу конференции
6.2 Воспроизведение аудио с компьютера	Пользователь	Пользователь по заданному праву пользуется имеет возможность включать аудиозаписи с компьютера и управлять их воспроизведением
7 Общение по видеоканалу	Пользователь	Пользователь по заданному праву пользуется видеоканалом с помощью реализаций использования
7.1 Демонстрация экрана	Пользователь	Пользователь по заданному праву пользуется имеет возможность включать демонстрацию экрана
7.2 Воспроизведение видео с компьютера	Пользователь	Пользователь по заданному праву пользуется имеет возможность включать видеозаписи с компьютера и управлять их воспроизведением
7.3 Демонстрация презентации	Пользователь	Пользователь по заданному праву пользуется имеет возможность включать презентации с компьютера и управлять ей
7.4 Поток с веб-камеры	Пользователь	Пользователь по заданному праву может включать веб-камеру и передавать её видеопоток на общий
8 Просмотр и прослушивание конференции	Пользователь	Пользователь просматривает видеопоток, прослушивает аудиопоток и просматривает текстовый поток.

4 Код программы

Программа написана на языке Python в среде разработки PyCharm с использованием фреймворка Django. Код программы представлен в приложении А.

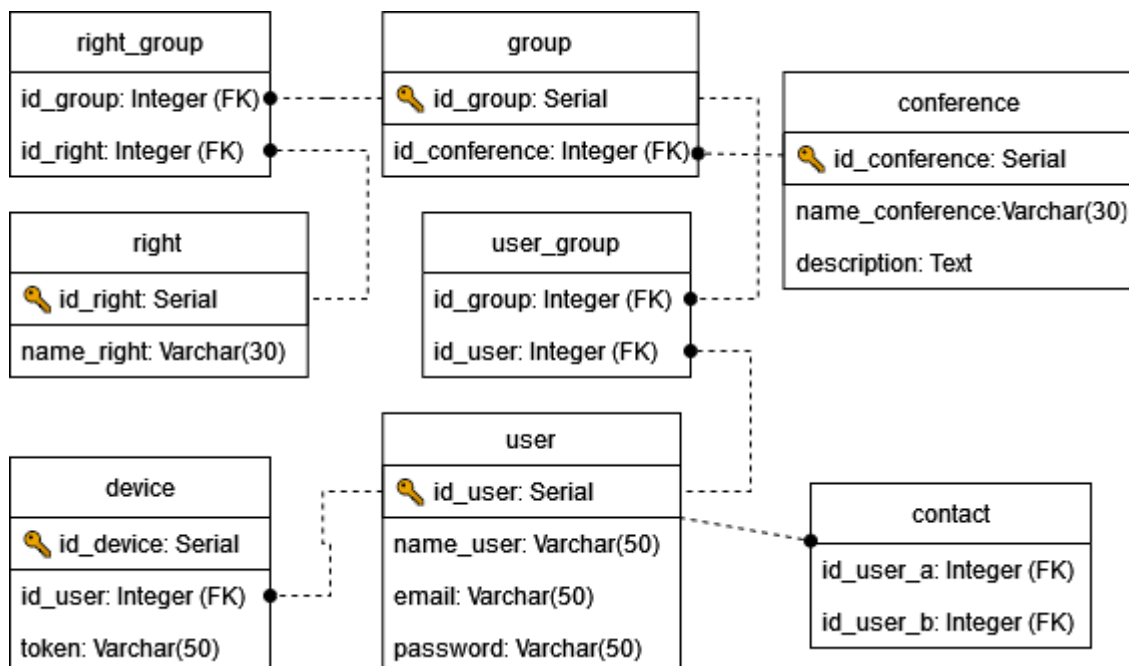
5 Логическая модель данных



6 Физическая модель данных

Выбранная СУБД – PostgreSQL.

Схема данных:



Поскольку проект является прототипом сервиса видеоконференций – он был упрощен. В нём будут иметься всего 4 таблицы: Пользователи, Конференции, Гости, Сообщения. Видео- и аудиоканала в сервисе не будет, будет только текстовый канал. Грубо говоря, будет реализован текстовый чат.

Физическое представление некоторых таблиц:

Таблица «Пользователь»:

main_user

columns 3

id integer (auto increment)

login varchar(20)

password varchar(40)

Таблица «Конференция»:

main_conference

columns 2

id integer (auto increment)

name varchar(20)

Таблица «Гость»:

main_guest

columns 3

id integer (auto increment)

conference_id_id bigint

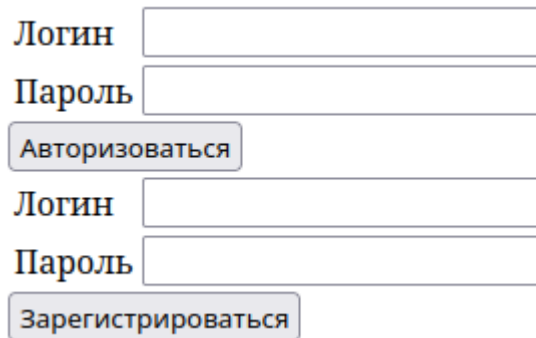
user_id_id bigint

Таблица «Сообщение»:

main_message	
columns 5	
id	integer (auto increment)
text	text
date	datetime
conference_id_id	bigint
user_id_id	bigint

7 Результаты выполнения программы

Страница авторизации и регистрации:



Логин

Пароль

Авторизоваться

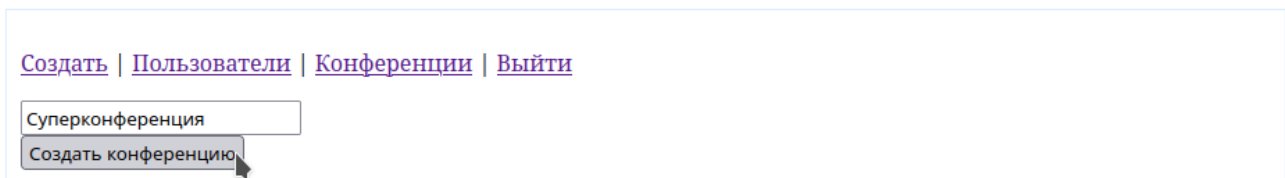
Логин

Пароль

Зарегистрироваться

Здесь пользователь может авторизоваться или, если его нет в системе, зарегистрироваться.

Страница создания конференции:

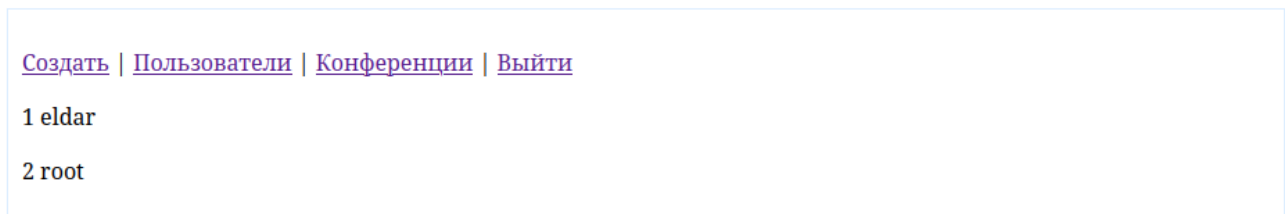


[Создать](#) | [Пользователи](#) | [Конференции](#) | [Выйти](#)

Создать конференцию

После создания конференции, текущий пользователь автоматически становится её гостем.

Список зарегистрированных пользователей:

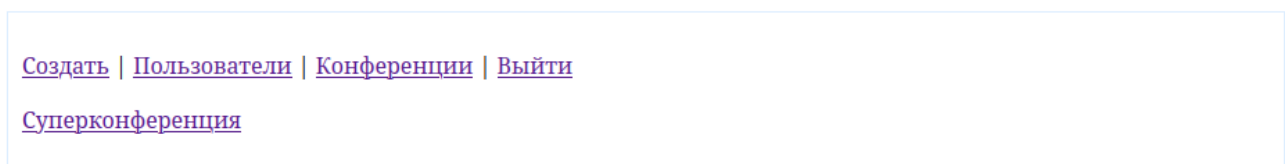


[Создать](#) | [Пользователи](#) | [Конференции](#) | [Выйти](#)

1 eldar

2 root

Список конференций, в котором текущий пользователь является гостем:



[Создать](#) | [Пользователи](#) | [Конференции](#) | [Выйти](#)

[Суперконференция](#)

Страница конференции:

[Создать](#) | [Пользователи](#) | [Конференции](#) | [Выйти](#)

Суперконференция

1 eldar

2 root

[Добавить пользователя](#)

eldar: Привет! 13 декабря 2021 г. 10:14

root: Привет! Как дела? 13 декабря 2021 г. 10:18

eldar: Всё отлично! Бизнес идёт в гору! 13 декабря 2021 г. 10:19

Отправить

Вверху представлено название конференции. Ниже идёт список пользователей, приглашенных в конференцию в качестве гостей (создатель конференции – всегда первый её гость). А после, идет история сообщений в конференции. Каждый гость имеет право оставлять сообщение в конференции.

Приложение А

Листинг файла models.py:

```
from django.db import models

class User(models.Model):
    login = models.CharField(max_length=20)
    password = models.CharField(max_length=40)

class Conference(models.Model):
    name = models.CharField(max_length=20)

class Guest(models.Model):
    conference_id = models.ForeignKey(Conference, on_delete=models.CASCADE)
    user_id = models.ForeignKey(User, on_delete=models.CASCADE)

class Message(models.Model):
    conference_id = models.ForeignKey(Conference, on_delete=models.CASCADE)
    user_id = models.ForeignKey(User, on_delete=models.CASCADE)
    text = models.TextField()
    date = models.DateTimeField()
```

Листинг файла views.py:

```
from .logic import *
from .models import *
from django.shortcuts import render
from django.http import HttpResponseRedirect
```

```
def index(request):
    if request.method == 'GET':
        id = request.session.get('id', '')
        if id != '':
            pass
        data = {
            'id': id,
        }
        return render(request, "index.html", data)
    return HttpResponseRedirect("/")

def sign_in(request):
    if request.method == 'POST':
        login = request.POST.get("login", "")
        password = request.POST.get("password", "")
        id = does_user_exist(login, password)
        if id:
            request.session['id'] = id
    return HttpResponseRedirect("/")

def sign_up(request):
```

```
if request.method == 'POST':
    login = request.POST.get("login", "")
    password = request.POST.get("password", "")
    if not does_user_exist(login, password):
        request.session['id'] = create_user(login, password)
    return HttpResponseRedirect("/")
```

```
def sign_out(request):
    if request.method == 'GET':
        request.session.flush()
    return HttpResponseRedirect("/")
```

```
def users(request):
    if request.method == 'GET':
        data = {
            'id': request.session.get('id', ''),
            'users': User.objects.all()
        }
    return render(request, "users.html", data)
return HttpResponseRedirect("/")
```

```
def conference(request):
    if request.method == 'GET':
        conference_id = request.GET.get('id', "")
        data = {
            'id': request.session.get('id', ''),
            'conference': get_conference(conference_id),
            'guests': get_guests(conference_id),
            'messages': get_messages(conference_id)
        }
    return render(request, "conference.html", data)
return HttpResponseRedirect("/")
```

```
def conferences(request):
    if request.method == 'GET':
        user_id = request.session.get('id', "")
        conference_s = get_conferences(user_id)
        data = {
            'id': request.session.get('id', ''),
            'conferences': conference_s
        }
    return render(request, "conferences.html", data)
return HttpResponseRedirect("/")
```

```
def create_conference(request):
    if request.method == 'POST':
        user_id = request.session.get('id', "")
        name = request.POST.get("name", "")
        conference_id = add_conference(name, user_id)
        return HttpResponseRedirect("/conference?id=" + str(conference_id))
    return HttpResponseRedirect("/")
```

```
def add_guest_list(request):
```

```

if request.method == 'GET':
    conference_id = request.GET.get("conference_id", "")
    data = {
        'id': request.session.get('id', ''),
        'conference_id': conference_id,
        'users': User.objects.all()
    }
    return render(request, "add_guest.html", data)
return HttpResponseRedirect("/")

def add_guest(request):
    if request.method == 'GET':
        conference_id = request.GET.get("conference_id", "")
        user_id = request.GET.get("user_id", "")
        add_user(conference_id, user_id)
        return HttpResponseRedirect("/conference?id=" + conference_id)
    return HttpResponseRedirect("/")

def add_message(request):
    if request.method == 'POST':
        conference_id = request.POST.get("conference_id", "")
        user_id = request.POST.get("user_id", "")
        message = request.POST.get("message", "")
        create_message(conference_id, user_id, message)
        return HttpResponseRedirect("/conference?id=" + conference_id)
    return HttpResponseRedirect("/")

```

Листинг файла logic.py:

```

import datetime
from .models import *

def does_user_exist(login, password):
    users = list(User.objects.filter(
        login=login,
        password=password
    ))
    return False if not len(users) else users[0].id

def create_user(login, password):
    user = User.objects.create(
        login=login,
        password=password
    )
    user.save()
    return user.id

def add_user(conference_id, user_id):
    Guest.objects.create(
        conference_id=get_conference(conference_id),
        user_id=get_user(user_id)
    ).save()

```

```

def create_message(conference_id, user_id, message):
    Message.objects.create(
        conference_id=get_conference(conference_id),
        user_id=get_user(user_id),
        text=message,
        date=datetime.datetime.now()
    ).save()

def add_conference(name, user_id):
    conference = Conference.objects.create(
        name=name
    )
    conference.save()
    add_user(conference.id, user_id)
    return conference.id

def get_user(user_id):
    return User.objects.get(id=int(user_id))

def get_conference(conference_id):
    return Conference.objects.get(id=int(conference_id))

def get_guests(conference_id):
    return list(Guest.objects.filter(conference_id=int(conference_id)))

def get_conferences(user_id):
    return map(
        lambda guest: guest.conference_id,
        list(Guest.objects.filter(user_id=get_user(user_id)))
    )

def get_messages(conference_id):
    return list(Message.objects.filter(conference_id=conference_id))

```

Листинг файла base.html:

```

<!DOCTYPE html>
{% load static %}
<html>
<head>
    <meta charset="utf-8" />
    <link rel="stylesheet" src="{% static 'css/styles.css' %}">
    <title>Сервис видеоконференций | {% block title %}{% endblock title %}</title>
</head>
<body>

<div class="content" style="width:900px;border:1px solid #def;margin:0 auto;padding:10px;">
{% if id == "" %}

    <form action="/sign_in" method="POST">
        {% csrf_token %}
        <table>
            <tr><td>Логин </td><td><input type="text" name="login"></td></tr>

```

```

        <tr><td>Пароль </td><td><input type="text" name="password"></td></tr>
    </table>
    <input type="submit" value="Авторизоваться">
</form>

<form action="/sign_up" method="POST">
    {% csrf_token %}
    <table>
        <tr><td>Логин </td><td><input type="text" name="login"></td></tr>
        <tr><td>Пароль </td><td><input type="text" name="password"></td></tr>
    </table>
    <input type="submit" value="Зарегистрироваться">
</form>

{% else %}

<p>
    <a href="/">Создать</a> |
    <a href="/users">Пользователи</a> |
    <a href="/conferences">Конференции</a> |
    <a href="/sign_out">Выйти</a>
</p>

<div>{% block content %}{% endblock content %}</div>

{% endif %}
</div>

</body>
</html>

```

Листинг файла index.html:

```

{% extends "base.html" %}
{% block title %}Index{% endblock title %}
{% block content %}
    <form method="post" action="/create_conference">
        {% csrf_token %}
        <input type="text" name="name"><br>
        <input type="submit" value="Создать конференцию">
    </form>
{% endblock content %}

```

Листинг файла users.html:

```

{% extends "base.html" %}
{% block title %}Пользователи{% endblock title %}
{% block content %}
    {% for user in users %}
        <p> {{ user.id }} {{ user.login }}
    {% endfor %}
{% endblock content %}

```

Листинг файла conferences.html:

```

{% extends "base.html" %}
{% block title %}Конференции{% endblock title %}
{% block content %}
    {% for conference in conferences %}
        <p><a href="/conference?id={{ conference.id }}">{{ conference.name }}</a>
    {% endfor %}

```

```
{% endfor %}
{% endblock content %}
```

Листинг файла conference.html:

```
{% extends "base.html" %}
{% block title %}Конференция{% endblock title %}
{% block content %}
    {{ conference.name }}
    <hr>
    {% for guest in guests %}
        <p> {{ guest.user_id.id }} {{ guest.user_id.login }}
    {% endfor %}
    <p> <a href="/add_guest_list?conference_id={{ conference.id }}">Добавить пользователя</a>
    <hr>
    {% for message in messages %}
        <p><span style="color:#9e2fce;">{{ message.user_id.login }}:</span> {{ message.text }} <span
style="color:#aaa;">{{ message.date }}</span>
    {% endfor %}
    <hr>
    <form method="post" action="/add_message">
        {% csrf_token %}
        <input type="hidden" name="conference_id" value="{{ conference.id }}">
        <input type="hidden" name="user_id" value="{{ id }}">
        <input type="text" name="message"><br>
        <input type="submit" value="Отправить">
    </form>
{% endblock content %}
```

Листинг файла add_guest.html:

```
{% extends "base.html" %}
{% block title %}Пользователи{% endblock title %}
{% block content %}
    {% for user in users %}
        <p><a href="/add_guest?conference_id={{ conference_id }}&user_id={{ user.id }}">{{ user.login }}</a>
    {% endfor %}
{% endblock content %}
```