**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ И.РАЗЗАКОВА**

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

**КАФЕДРА ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

**Отчет**

**по предквалификационной практике**

**Тема: Автоматизированная система составления расписания занятий КГТУ**

**ВЫПОЛНИЛА:** СТУДЕНТ гр. ПИ(б)-1-14

Абубакир кызы Динай

**РУКОВОДИТЕЛЬ:** Семененко А. С.

**БИШКЕК 2018**

Оглавление

[Описание проблемы 2](#_Toc513456937)

[Use Cases 3](#_Toc513456938)

[Use Case “Ввод расчасовки кафедры” 3](#_Toc513456939)

[Use Case “Ввод личного времени преподавателя” 3](#_Toc513456940)

[Use Case “Составление расписания учебных занятий” 3](#_Toc513456941)

[Use Case “Получить расписание группы” 3](#_Toc513456942)

[Use Case “Получить расписание преподавателя” 4](#_Toc513456943)

[Use Case “Получить расписание кафедры” 4](#_Toc513456944)

[Use Case “Получить расписание аудитории” 4](#_Toc513456945)

[Функциональные требования 5](#_Toc513456946)

[Требования к данным 5](#_Toc513456947)

[Ограничения 6](#_Toc513456948)

[Проектирование базы данных 7](#_Toc513456949)

[Разработка проекта «Составитель расписания» 8](#_Toc513456950)

[Структура проекта 8](#_Toc513456951)

[Функционирование системы на примере приложения «Аудитории» 10](#_Toc513456952)

[auditorium/models.py 10](#_Toc513456953)

[auditorium/forms.py 11](#_Toc513456954)

[auditorium/views.py 12](#_Toc513456955)

[Интерфейс 13](#_Toc513456956)

[auditorium/templates/\_base.html 13](#_Toc513456957)

[auditorium/templates/auditoriumList.html 15](#_Toc513456958)

[auditorium/templates/auditorium.html 17](#_Toc513456959)

[auditorium/templates/auditorium\_form.html 19](#_Toc513456960)

[auditorium/templates/auditorium\_confirm\_delete.html 20](#_Toc513456961)

Описание проблемы

В настоящее время составление расписание включает следующие этапы:

сбор расчасовки с каждой кафедры университета КГТУ;

информирование диспетчерской о личном времени преподавателей;

затем на шаблонном листе диспетчеры вручную составляют расписание для групп, и так до тех пор пока не будет составлено расписание для всего университета. При составлении расписания стараются учитывать личное время преподавателей, количество пар в день, не допускать появление окон, накладок (несколько пар в одно время) у преподавателей, несколько учебных занятий в одной аудитории в одно и то же время и т.д;

Составленное на бумаге расписание переносится вручную в образовательный портал AVN.

Поскольку составление расписания является сложным и трудоемким процессом, который производится вручную, то оно подвержено ошибкам, связанными с человеческим фактором. Невозможно оценить качество расписания, так как оно составлено на бумаге и нет критериев для оценки качества составленного расписания, а также расписание приходится вручную заносить в AVN. Такие проблемы могут быть решены созданием программной системы для автоматизированного составления расписания учебных занятий.

Желаемое состояние:

Необходимо создать ПО, которое позволит минимизировать ошибки, связанные с человеческим фактором, снизить временные затраты на составление расписания, избавиться от бумажной работы, оценить и повысить качество расписания.

Список стейкхолдеров:

* Студенты;
* Преподаватели;
* Преподаватели-расчасовщики;
* Диспетчеры;
* Администраторы лабораторных аудиторий;
* Учебное управление;
* Отдел качества обучения.

Use Cases

Use Case “Ввод расчасовки кафедры”

1. Система запрашивает логин и пароль пользователя.
2. Пользователь вводит свой логин и пароль.
3. Система отображает список действий (ввести расчасовку кафедры, ввести личное время преподавателя).
4. Пользователь выбирает “Ввести расчасовку кафедры”.
5. Система отображает на экране форму для ввода расчасовки кафедры, за которой закреплен пользователь.
6. Пользователь вводит расчасовку кафедры (номер потока, курс, группа, количество студентов, ФИО преподавателя, дисциплина, вид занятия, количество часов за семестр, график учебного процесса - количество часов в неделю для всех недель семестра, аудитория).
7. Пользователь подтверждает ввод расчасовки кафедры.
8. Система отображает на экране сообщение об успешном вводе расчасовки кафедры.
9. Пользователь выходит из системы.

Use Case “Ввод личного времени преподавателя”

1. Система запрашивает логин и пароль пользователя.
2. Пользователь вводит свой логин и пароль.
3. Система отображает на экране форму для ввода личного времени пользователя.
4. Пользователь вводит личное время (время, когда он не может проводить учебное занятие).
5. Пользователь подтверждает ввод личного времени.
6. Система отображает на экране сообщение об успешном вводе личного времени.
7. Пользователь выходит из системы.

Use Case “Составление расписания учебных занятий”

1. Система запрашивает логин и пароль пользователя.
2. Пользователь вводит свой логин и пароль.
3. Система выводит на экран критерии качества расписания.
4. Пользователь выбирает критерии качества расписания, которые будут использоваться при составлении расписания учебных занятий.
5. Система выводит на экран форму для ввода расписания.
6. Пользователь вводит расписание (сочетание времени, аудитории, группы, преподавателя и предмета для всех учебных занятий университета).
7. Система проверяет расписание на соответствие заданным критериям.
8. Пользователь подтверждает ввод расписания.
9. Система отображает на экране сообщение об успешном вводе расписания.
10. Пользователь выходит из системы.

Use Case “Получить расписание группы”

1. Система отображает на экране список действий (“Получить расписание группы”, “Получить расписание преподавателя”, “Получить расписание кафедры”, “Получить расписание аудитории”).
2. Пользователь выбирает действие “Получить расписание группы”.
3. Система предлагает пользователю выбрать курс.
4. Пользователь выбирает курс.
5. Система предлагает пользователю выбрать группу.
6. Пользователь выбирает группу.
7. Система отображает на экране расписание выбранной пользователем группы.

Use Case “Получить расписание преподавателя”

1. Система отображает на экране список действий (“Получить расписание группы”, “Получить расписание преподавателя”, “Получить расписание кафедры”, “Получить расписание аудитории”).
2. Пользователь выбирает действие “Получить расписание преподавателя”.
3. Система предлагает пользователю выбрать преподавателя.
4. Пользователь выбирает преподавателя.
5. Система отображает на экране расписание выбранного пользователем преподавателя.

Use Case “Получить расписание кафедры”

1. Система отображает на экране список действий (“Получить расписание группы”, “Получить расписание преподавателя”, “Получить расписание кафедры”, “Получить расписание аудитории”).
2. Пользователь выбирает действие “Получить расписание кафедры”.
3. Система предлагает пользователю выбрать кафедру.
4. Пользователь выбирает кафедру.
5. Система отображает на экране расписание выбранной пользователем кафедры (расписание всех преподавателей кафедры).

Use Case “Получить расписание аудитории”

1. Система отображает на экране список действий (“Получить расписание группы”, “Получить расписание преподавателя”, “Получить расписание кафедры”, “Получить расписание аудитории”).
2. Пользователь выбирает действие “Получить расписание аудитории”.
3. Система предлагает пользователю выбрать корпус.
4. Пользователь выбирает корпус.
5. Система предлагает пользователю выбрать аудиторию.
6. Пользователь выбирает аудиторию.
7. Система отображает на экране расписание выбранной пользователем аудитории (расписание всех учебных занятий, проводимых в этой аудитории).

# Функциональные требования

SRS1.1. Система должна проверить логин и пароль, введенный пользователем.

SRS1.2. Система должна предоставить преподавателю-расчасовщику выбор задачи (ввести расчасовку кафедры, ввести личное время преподавателя).

SRS1.3. Система должна сохранить расчасовку кафедры.

SRS1.4. Система должна сохранить личное время преподавателя.

SRS1.5. Система должна предоставить пользователю выбор задачи (“Получить расписание группы”, “Получить расписание преподавателя”, “Получить расписание кафедры”, “Получить расписание аудитории”).

SRS1.6. Система должна предоставить пользователю выбор курса.

SRS1.7. Система должна предоставить пользователю выбор группы.

SRS1.8. Система должна предоставить пользователю выбор кафедры.

SRS1.9. Система должна предоставить пользователю выбор преподавателя.

SRS1.10. Система должна предоставить пользователю выбор корпуса.

SRS1.11. Система должна предоставить пользователю выбор аудитории.

SRS1.12. Система должна проверить введенное пользователем расписание на соответствие критериям качества.

# Требования к данным

SRS2.1. Система должна предоставить пользователю возможность ввода логина.

SRS2.2. Система должна предоставить пользователю возможность ввода пароля.

SRS2.3. Система должна оповестить пользователя в случаях, когда логин или пароль неверные.

SRS2.4. Система должна предоставить преподавателю-расчасовщику возможность ввода расчасовки кафедры:

* номер потока;
* курс;
* группа;
* количество студентов;
* ФИО преподавателя;
* дисциплина;
* вид занятия;
* количество часов за семестр, отведенных для данной дисциплины;
* график учебного процесса - количество часов в неделю для всех недель семестра;
* аудитория.

SRS2.5. Система должна оповестить пользователя об успешном сохранении расчасовки кафедры.

SRS2.6. Система должна предоставить преподавателю возможность ввода личного времени.

SRS2.7. Система должна оповестить пользователя об успешном сохранении личного времени преподавателя.

# Ограничения

SRS4.1. Система должна содержать следующие классы пользователей: обычный пользователь, преподаватель-расчасовщик, преподаватель, диспетчер.

# Проектирование базы данных

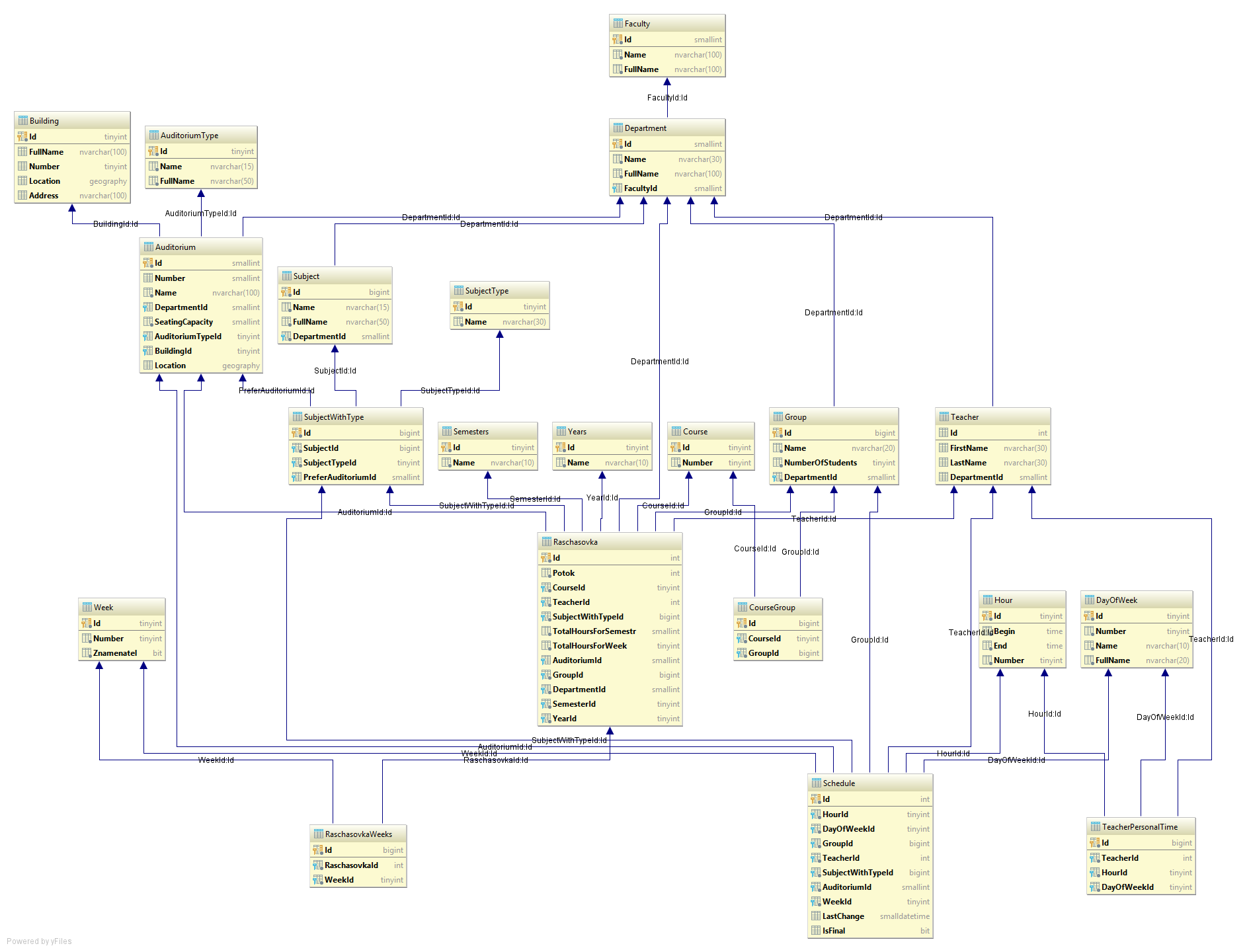
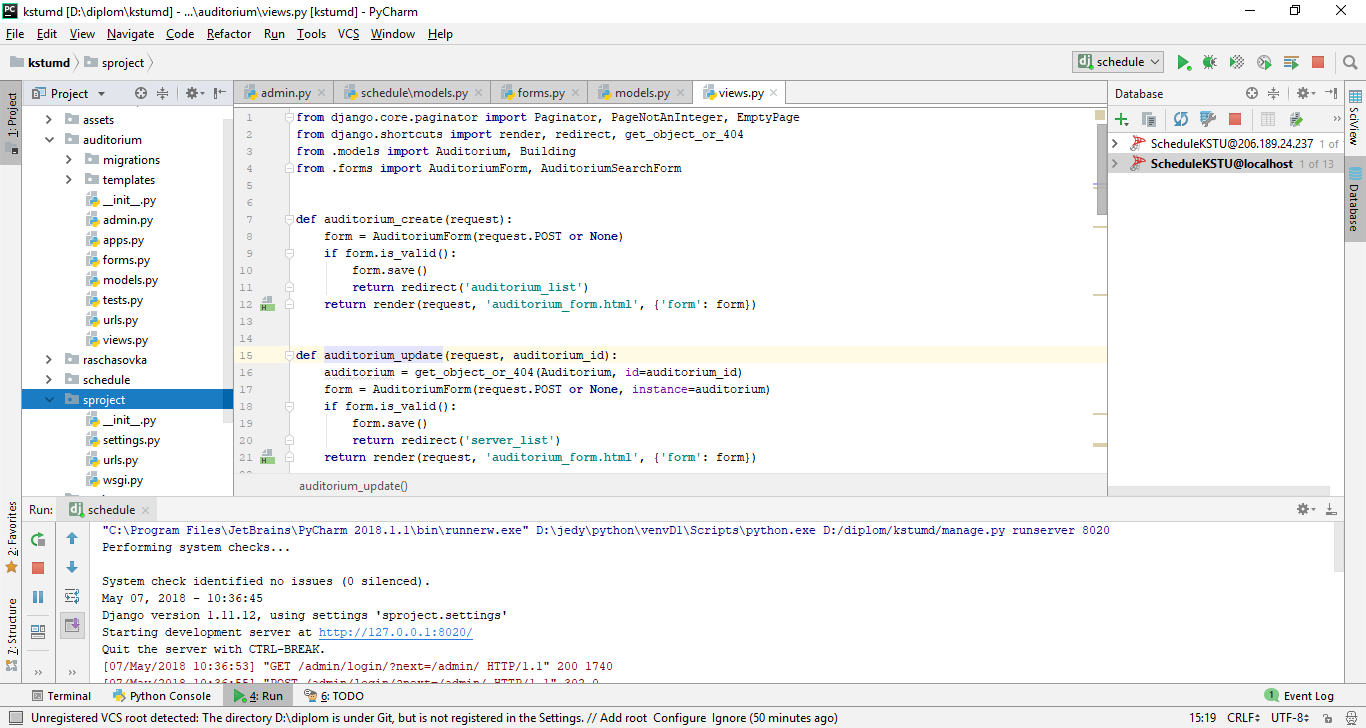


Рис.1. Схема таблиц базы данных

# Разработка проекта «Составитель расписания»

## Структура проекта



Настройки общие для всей системы

sproject/settings.py

INSTALLED\_APPS = [  
 **'auditorium'**,  
 **'django.contrib.admin'**,  
 **'django.contrib.auth'**,  
 **'django.contrib.contenttypes'**,  
 **'django.contrib.sessions'**,  
 **'django.contrib.messages'**,  
 **'django.contrib.staticfiles'**,  
]  
  
MIDDLEWARE = [  
 **'django.middleware.security.SecurityMiddleware'**,  
 **'django.contrib.sessions.middleware.SessionMiddleware'**,  
 **'django.middleware.common.CommonMiddleware'**,  
 **'django.middleware.csrf.CsrfViewMiddleware'**,  
 **'django.contrib.auth.middleware.AuthenticationMiddleware'**,  
 **'django.contrib.messages.middleware.MessageMiddleware'**,  
 **'django.middleware.clickjacking.XFrameOptionsMiddleware'**,  
]  
  
ROOT\_URLCONF = **'sproject.urls'**TEMPLATES = [  
 {  
 **'BACKEND'**: **'django.template.backends.django.DjangoTemplates'**,  
 **'DIRS'**: [],  
 **'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'**,  
 ],  
 },  
 },  
]  
  
*# Internationalization*LANGUAGE\_CODE = **'ru-ru'**TIME\_ZONE = **'Asia/Bishkek'***# Static files (CSS, JavaScript, Images)*STATIC\_URL = **'/static/'**STATICFILES\_DIRS = [  
 os.path.join(BASE\_DIR, **"assets"**),  
]  
MEDIA\_URL = **'/media/'**MEDIA\_ROOT = os.path.join(BASE\_DIR, **'media'**)  
  
LOGGING = {  
 **'version'**: 1,  
 **'disable\_existing\_loggers'**: **False**,  
 **'handlers'**: {  
 **'filedb'**: {  
 **'level'**: **'DEBUG'**,  
 **'class'**: **'logging.FileHandler'**,  
 **'filename'**: os.path.join(BASE\_DIR, **'debugdb.log'**),  
 },  
 **'console'**: {  
 **'level'**: **'DEBUG'**,  
 **'class'**: **'logging.StreamHandler'**,  
 },  
 },  
 **'loggers'**: {  
 **'django.db.backends'**: {  
 **'handlers'**: [**'filedb'**],  
 **'level'**: **'DEBUG'**,  
 **'propagate'**: **True**,  
 },  
 **'django'**: {  
 **'level'**: **'INFO'**,  
 **'handlers'**: [**'console'**],  
 },  
 },  
}

Общий роутинг:

Sproject/urls.py

**from** django.conf **import** settings  
**from** django.conf.urls.static **import** static  
**from** django.conf.urls **import** url, include  
**from** django.contrib **import** admin  
  
urlpatterns = [  
 url(**r'^auditorium/'**, include(**'auditorium.urls'**)),  
 url(**r'^admin/'**, admin.site.urls),  
]  
  
urlpatterns += static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)  
urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

## Функционирование системы на примере приложения «Аудитории»

### auditorium/models.py

**from** django.db **import** models  
  
**class** Auditorium(models.Model):  
 id = models.AutoField(db\_column=**'Id'**, primary\_key=**True**) *# Field name made lowercase.* number = models.SmallIntegerField(db\_column=**'Number'**, blank=**True**, null=**True**) *# Field name made lowercase.* name = models.CharField(db\_column=**'Name'**, max\_length=100, verbose\_name=**"Название"**) *# Field name made lowercase.* departmentid = models.ForeignKey(**'Department'**, models.DO\_NOTHING, db\_column=**'DepartmentId'**, blank=**True**, null=**True**, verbose\_name=**"Кафедра"**) *# Field name made lowercase.* seatingcapacity = models.SmallIntegerField(db\_column=**'SeatingCapacity'**, verbose\_name=**"Вместимость"**) *# Field name made lowercase.* auditoriumtypeid = models.ForeignKey(**'Auditoriumtype'**, models.DO\_NOTHING, db\_column=**'AuditoriumTypeId'**, blank=**True**, null=**True**, verbose\_name=**"Тип аудитории"**) *# Field name made lowercase.* buildingid = models.ForeignKey(**'Building'**, models.DO\_NOTHING, db\_column=**'BuildingId'**, blank=**True**, null=**True**, verbose\_name=**"Корпус"**) *# Field name made lowercase.* location = models.BinaryField(db\_column=**'Location'**, blank=**True**, null=**True**) *# Field name made lowercase.* **def** \_\_str\_\_(self):  
 **return** self.name  
  
 **class** Meta:  
 managed = **False** db\_table = **'Auditorium'** verbose\_name = **"Аудитория"** verbose\_name\_plural = **"Аудитории"  
  
  
class** Auditoriumtype(models.Model):  
 id = models.AutoField(db\_column=**'Id'**, primary\_key=**True**) *# Field name made lowercase.* name = models.CharField(db\_column=**'Name'**, max\_length=15, verbose\_name=**"Сокращенное навзание"**) *# Field name made lowercase.* fullname = models.CharField(db\_column=**'FullName'**, max\_length=50, verbose\_name=**"Полное название"**) *# Field name made lowercase.* **def** \_\_str\_\_(self):  
 **return** self.fullname  
  
 **class** Meta:  
 managed = **False** db\_table = **'AuditoriumType'** verbose\_name = **"Тип аудитории"** verbose\_name\_plural = **"Типы аудиторий"  
  
  
class** Building(models.Model):  
 id = models.AutoField(db\_column=**'Id'**, primary\_key=**True**) *# Field name made lowercase.* fullname = models.CharField(db\_column=**'FullName'**, max\_length=100, blank=**True**, null=**True**, verbose\_name=**"Название"**) *# Field name made lowercase.* number = models.SmallIntegerField(db\_column=**'Number'**, blank=**True**, null=**True**, verbose\_name=**"Номер корпуса"**) *# Field name made lowercase.* location = models.BinaryField(db\_column=**'Location'**, blank=**True**, null=**True**) *# Field name made lowercase.* address = models.CharField(db\_column=**'Address'**, max\_length=100, blank=**True**, null=**True**, verbose\_name=**"Адрес"**) *# Field name made lowercase.* **def** \_\_str\_\_(self):  
 **return** self.fullname  
  
 **class** Meta:  
 managed = **False** db\_table = **'Building'** verbose\_name = **"Корпус"** verbose\_name\_plural = **"Корпусы"  
  
  
class** Department(models.Model):  
 id = models.AutoField(db\_column=**'Id'**, primary\_key=**True**) *# Field name made lowercase.* name = models.CharField(db\_column=**'Name'**, max\_length=30, verbose\_name=**"Кратко"**) *# Field name made lowercase.* fullname = models.CharField(db\_column=**'FullName'**, max\_length=100, verbose\_name=**"Полное название"**) *# Field name made lowercase.* facultyid = models.ForeignKey(**'Faculty'**, models.DO\_NOTHING, db\_column=**'FacultyId'**, blank=**True**, null=**True**, verbose\_name=**"Факультет"**) *# Field name made lowercase.* **def** \_\_str\_\_(self):  
 **return** self.name  
 **class** Meta:  
 managed = **False** db\_table = **'Department'** verbose\_name = **"Кафедра"** verbose\_name\_plural = **"Кафедры"  
  
  
class** Faculty(models.Model):  
 id = models.AutoField(db\_column=**'Id'**, primary\_key=**True**) *# Field name made lowercase.* name = models.CharField(db\_column=**'Name'**, max\_length=100, verbose\_name=**"Кратко"**) *# Field name made lowercase.* fullname = models.CharField(db\_column=**'FullName'**, max\_length=100, verbose\_name=**"Полное название"**) *# Field name made lowercase.* **def** \_\_str\_\_(self):  
 **return** self.name + **"("** + self.fullname + **")"  
  
 class** Meta:  
 managed = **False** db\_table = **'Faculty'** verbose\_name = **"Факультет"** verbose\_name\_plural = **"Факультет"**

### auditorium/forms.py

**from** django.forms **import** ModelForm, Form, CharField, forms  
**from** django **import** forms  
**from** django.urls **import** reverse\_lazy **as** \_  
  
**from** .models **import** Auditorium  
  
  
**class** AuditoriumForm(ModelForm):  
 **class** Meta:  
 model = Auditorium  
 fields = [  
 **'name'**, **'auditoriumtypeid'**, **'seatingcapacity'**, **'departmentid'**, **'buildingid'** ]  
 widgets = {  
 **'name'**: forms.TextInput(attrs={**'class'**: **'form-control'**}),  
 **'auditoriumtypeid'**: forms.Select(attrs={**'class'**: **'form-control'**}),  
 **'seatingcapacity'**: forms.NumberInput(attrs={**'class'**: **'form-control'**}),  
 **'departmentid'**: forms.Select(attrs={**'class'**: **'form-control'**}),  
 }  
  
 **def** save(self, commit=**True**):  
 auditorium = super(AuditoriumForm, self).save(commit=**False**)  
  
 *# магия* commit = **True  
 if** commit:  
 auditorium.save()  
 **return** auditorium  
  
**class** AuditoriumSearchForm(Form):  
 search = CharField(required=**False**)  
 search.widget.attrs[**'class'**] =**"form-control"** search.widget.attrs[**'id'**] = **"audSearch"**

### auditorium/views.py

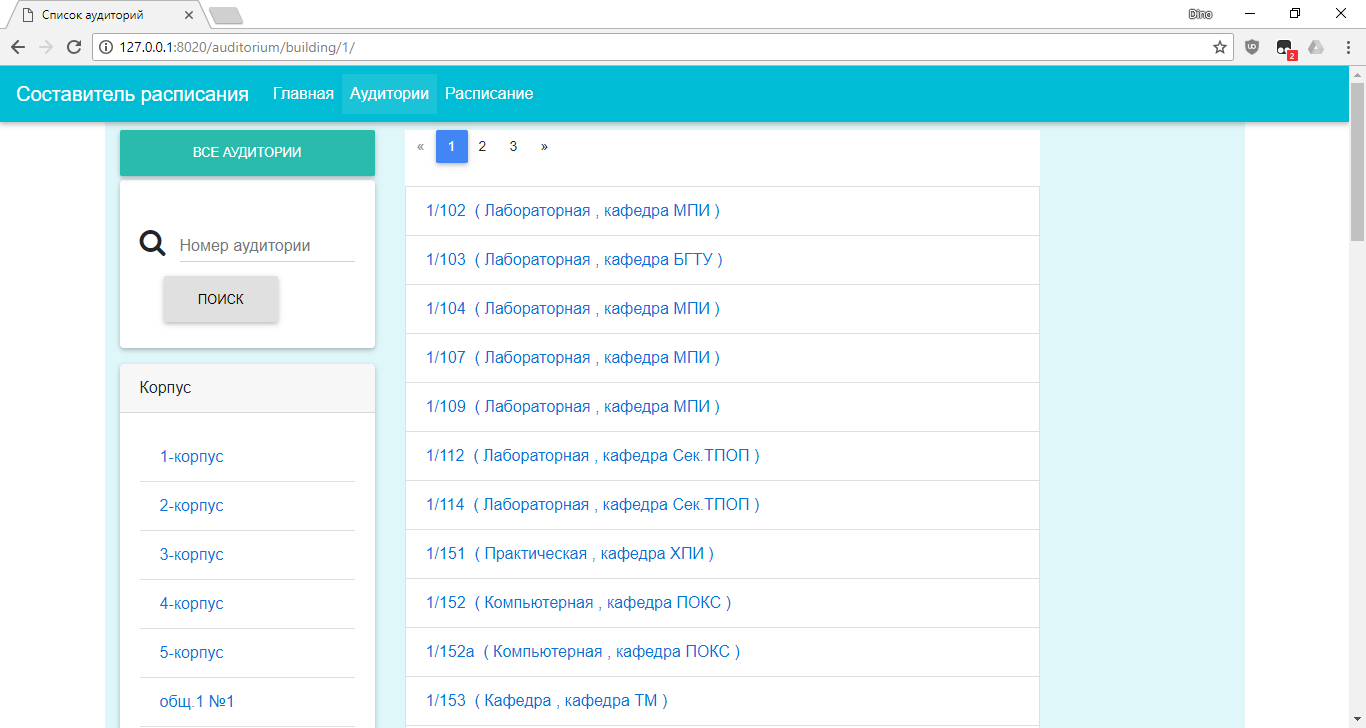
**from** django.core.paginator **import** Paginator, PageNotAnInteger, EmptyPage  
**from** django.shortcuts **import** render, redirect, get\_object\_or\_404  
**from** .models **import** Auditorium, Building  
**from** .forms **import** AuditoriumForm, AuditoriumSearchForm  
  
  
**def** auditorium\_create(request):  
 form = AuditoriumForm(request.POST **or None**)  
 **if** form.is\_valid():  
 form.save()  
 **return** redirect(**'auditorium\_list'**)  
 **return** render(request, **'auditorium\_form.html'**, {**'form'**: form})  
  
  
**def** auditorium\_update(request, auditorium\_id):  
 auditorium = get\_object\_or\_404(Auditorium, id=auditorium\_id)  
 form = AuditoriumForm(request.POST **or None**, instance=auditorium)  
 **if** form.is\_valid():  
 form.save()  
 **return** redirect(**'server\_list'**)  
 **return** render(request, **'auditorium\_form.html'**, {**'form'**: form})  
  
  
**def** auditorium\_delete(request, auditorium\_id):  
 auditorium = get\_object\_or\_404(Auditorium, id=auditorium\_id)  
 **if** request.method == **'POST'**:  
 auditorium.delete()  
 **return** redirect(**'server\_list'**)  
 **return** render(request, **'auditorium\_confirm\_delete.html'**, {**'auditorium'**: auditorium})  
  
  
*# url="auditorium", все аудитории  
# url="auditorium/building/<int>", аудитории корпуса***def** auditorium\_list(request, building\_id=**None**):  
  
 buildings = Building.objects.all() *# все корпуса(для навигации)* auditoriums = Auditorium.objects.all().order\_by(**'buildingid'**) *# все аудитории  
 # auditoriums = auditoriums.exclude(seatingcapacity\_\_lte=1) # исключить аудитории с вместимостью <= 1* **if** building\_id:  
 auditoriums = auditoriums.filter(buildingid=building\_id).order\_by( **'name'** ) *# аудитории одного корпуса* form = AuditoriumSearchForm(request.GET) *# создать форму для поиска аудитории* form.is\_valid()  
 **if** form.cleaned\_data.get(**'search'**):  
 auditoriums = auditoriums.filter(name\_\_icontains=form.cleaned\_data[**'search'**]) *# найденные аудитории* paginator = Paginator(auditoriums, 50) *# 50 аудиторий на страницу* page = request.GET.get(**'page'**)  
 **try**:  
 auditoriums = paginator.page(page)  
 **except** PageNotAnInteger:  
 *# If page is not an integer, deliver first page.* auditoriums = paginator.page(1)  
 **except** EmptyPage:  
 *# If page is out of range (e.g. 9999), deliver last page of results.* auditoriums = paginator.page(paginator.num\_pages)  
  
 context = {  
 **'buildings'**: buildings,  
 **'auditoriums'**: auditoriums,  
 **'form'**: form,  
 }  
 **return** render(request, **"auditoriumList.html"**, context)  
  
  
*# url="auditorium/<int>", конкретная аудитория***def** auditorium(request, auditorium\_id):  
 auditorium = Auditorium.objects.get(id=auditorium\_id) *# аудитория, соответствующая auditorium\_id* buildings = Building.objects.all() *# все корпуса(для навигации)* context = {  
 **'auditorium'**: auditorium,  
 **'buildings'**: buildings,  
 }  
 **return** render(request, **"auditorium.html"**, context)

### Интерфейс

#### auditorium/templates/\_base.html

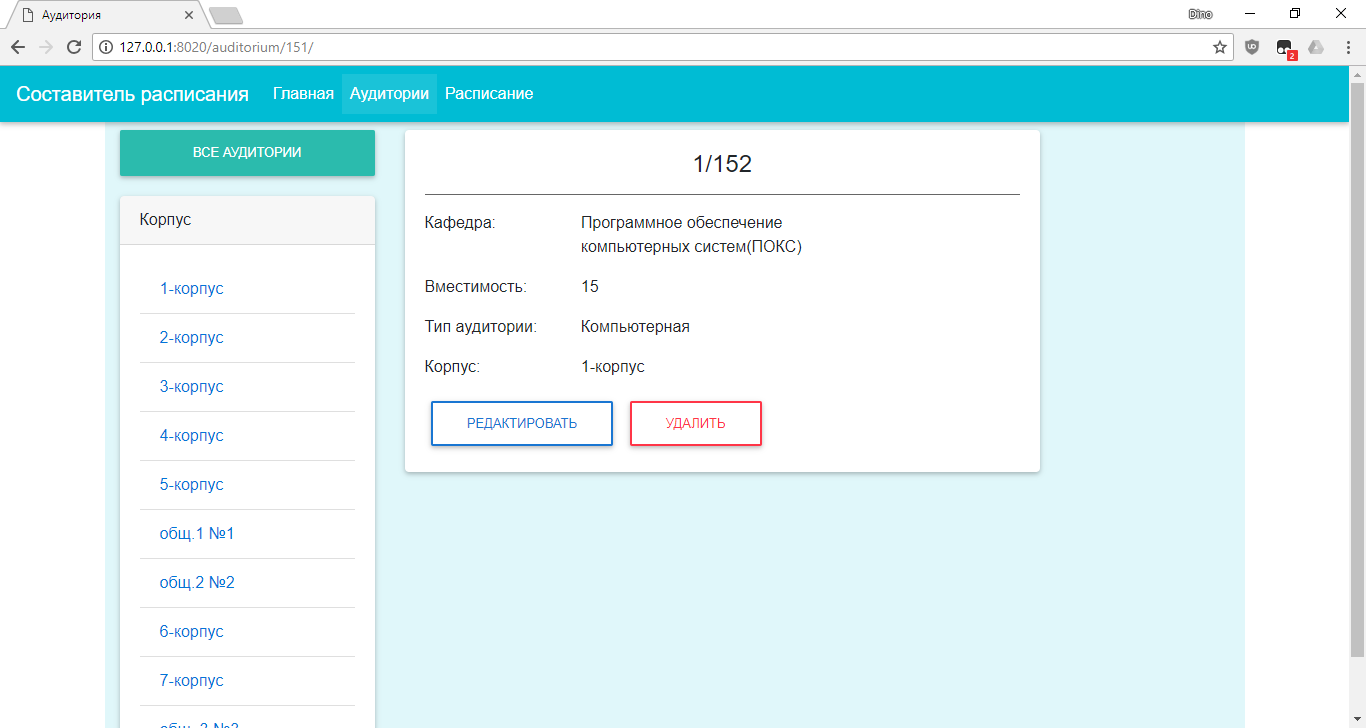
<!DOCTYPE **HTML**>  
<**html lang="ru"**>  
<**head**>  
 <**meta charset="utf-8"**>  
 <**meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"**>  
 <**meta http-equiv="x-ua-compatible" content="ie=edge"**>  
 <**title**>  
 {% **block title** %} Составитель расписания {% **endblock** %}  
 </**title**>  
 {% **load static** %}  
 *<!-- Font Awesome -->* <**link rel="stylesheet" href="**{% **static "/css/font-awesome.min.css"** %}**"**>  
 *<!-- Bootstrap core CSS -->* <**link href="**{% **static "/css/bootstrap.min.css"** %}**" rel="stylesheet"**>  
 *<!-- Material Design Bootstrap -->* <**link href="**{% **static "/css/mdb.min.css"** %}**" rel="stylesheet"**>  
 *<!-- Сustom styles -->* <**link href="**{% **static "/css/style.css"** %}**" rel="stylesheet"**>  
</**head**>  
<**body**>  
 {% **block navigation** %}  
 *<!--Main Navigation-->* <**nav class="navbar navbar-expand-lg navbar-dark cyan"**>  
 <**a class="navbar-brand" href="/"**><**strong**>Составитель расписания</**strong**></**a**>  
 <**button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation"**>  
 <**span class="navbar-toggler-icon"**></**span**>  
 </**button**>  
 <**div class="collapse navbar-collapse" id="navbarSupportedContent"**>  
 <**ul class="navbar-nav mr-auto"**>  
  
 *<!-- Навигация -->* <**li class="nav-item"**>  
 <**a class="nav-link" href="/"**>Главная <**span class="sr-only"**>(current)</**span**></**a**>  
 </**li**>  
 {% **block nav** %}  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Аудитории</**a**>  
 </**li**>  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Еще что то</**a**>  
 </**li**>  
 {% **endblock** %}  
 *<!-- Навигация -->* </**ul**>  
 </**div**>  
 </**nav**>  
 *<!--Main Navigation-->* {% **endblock** %}  
  
 *<!--Main Layout-->* <**main**>  
 {% **block content** %}  
 {% **endblock** %}  
 </**main**>  
 *<!--Main Layout-->  
 <!-- JQuery -->* <**script type="text/javascript" src="**{% **static "js/jquery-3.2.1.min.js"** %}**"**></**script**>  
  
 *<!-- Tooltips -->* <**script type="text/javascript" src="**{% **static "js/popper.min.js"** %}**"**></**script**>  
  
 *<!-- Bootstrap core JavaScript -->* <**script type="text/javascript" src="**{% **static "js/bootstrap.min.js"** %}**"**></**script**>  
  
 *<!-- MDB core JavaScript -->* <**script type="text/javascript" src="**{% **static "js/mdb.min.js"** %}**"**></**script**>  
</**body**>  
</**html**>

#### auditorium/templates/auditoriumList.html



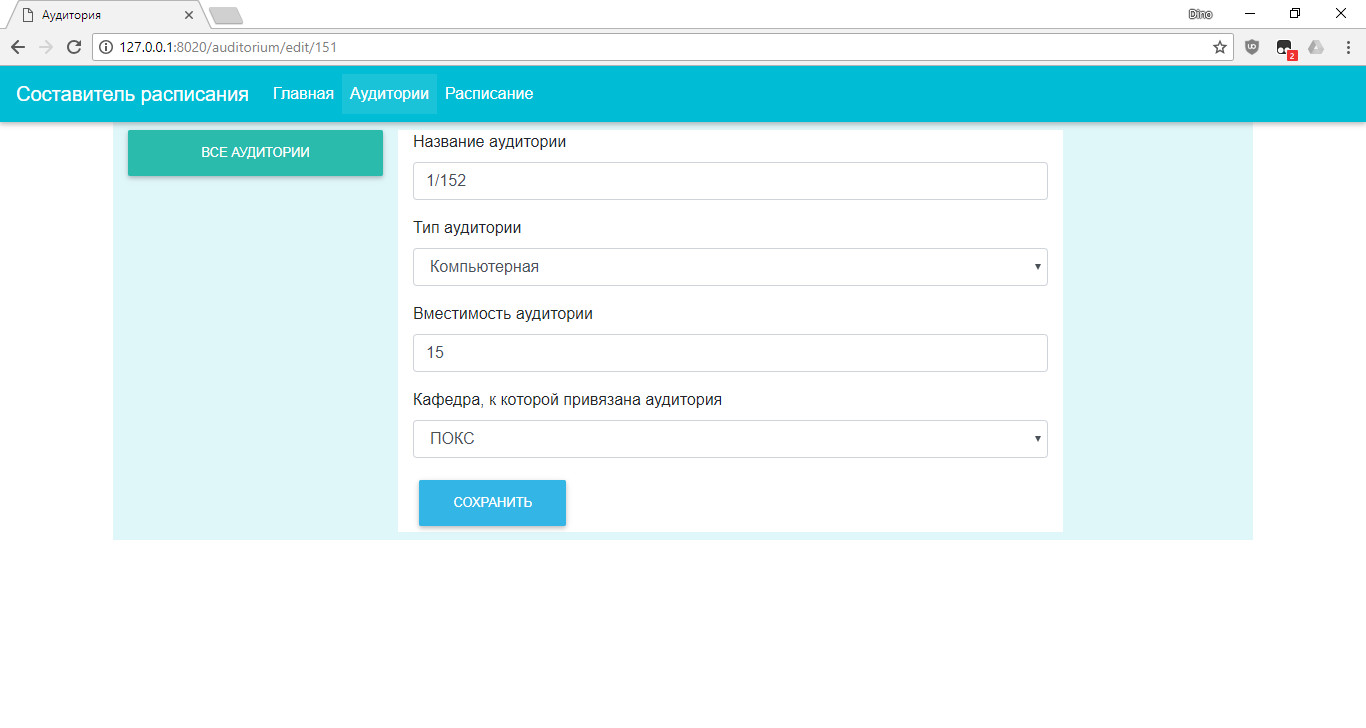
{% **extends "\_base.html"** %}  
  
{% **block title** %}Список аудиторий{% **endblock** %}  
  
*<!-- Навигация -->* {% **block nav** %}  
 <**li class="nav-item active"**>  
 <**a class="nav-link" href="**{% **url "auditorium\_list"** %}**"**>Аудитории </**a**>  
 </**li**>  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Еще что то</**a**>  
 </**li**>  
 {% **endblock** %}  
*<!-- Навигация -->*{% **block content** %}  
 <**div class="container"**>  
 <**div class="row cyan lighten-5 pb-2"**>  
 <**div class="col-3 mt-1 d-none d-md-block d-lg-block"**>  
 <**a href="**{% **url 'auditorium\_list'** %}**" class="btn btn-default btn-block mt-1 mb-1" role="button"**>  
 Все аудитории  
 </**a**>  
 *<!-- Поиск аудитории -->* <**div class="card"**>  
 <**div class="card-body"**>  
 <**form role="search" class="visible-xs"**>  
 <**div class="md-form"**>  
 <**i class="fa fa-search prefix"**></**i**>  
 {{ **form**.**search** }}  
 <**label for="audSearch"**>Номер аудитории</**label**>  
 <**button class="btn btn-light ml-4" type="submit"**> Поиск </**button**>  
 </**div**>  
 </**form**>  
 </**div**>  
 </**div**>  
 *<!-- Поиск аудитории -->  
  
 <!--Список корпусов-->* <**div class="card mt-3"**>  
 <**div class="card-header"**> Корпус </**div**>  
 <**div class="card-body"**>  
 <**ul class="list-group list-group-flush"**>  
 {% **for building in buildings** %}  
 <**li class="list-group-item"**>  
 <**a href="**{% **url 'aud\_build' building**.**id** %}**"**>  
 {{ **building**.**fullname** }}  
 </**a**>  
 </**li**>  
 {% **endfor** %}  
 </**ul**>  
 </**div**>  
 </**div**>  
 *<!--Список корпусов-->* </**div**>  
 <**div class="col-7 "**>  
 *<!-- Список аудиторий -->* <**div class="list-group white"**>  
 *<!-- Пагинация -->* <**nav**>  
 {% **if auditoriums**.**has\_other\_pages** %}  
 <**ul class="pagination pg-blue white"**>  
 {% **if auditoriums**.**has\_previous** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **auditoriums**.**previous\_page\_number** }}**"**>**&laquo;**</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item disabled"**>  
 <**a class="page-link" href="#" tabindex="-1"**>**&laquo;**</**a**>  
 </**li**>  
 {% **endif** %}  
 {% **for i in auditoriums**.**paginator**.**page\_range** %}  
 {% **if auditoriums**.**number** == **i** %}  
 <**li class="page-item active"**>  
 <**a class="page-link" href="?page=**{{ **i** }}**"**>{{ **i** }}</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **i** }}**"**>{{ **i** }}</**a**>  
 </**li**>  
 {% **endif** %}  
 {% **endfor** %}  
 {% **if auditoriums**.**has\_next** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **auditoriums**.**next\_page\_number** }}**"**>**&raquo;**</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item disabled"**><**span**>**&raquo;**</**span**></**li**>  
 {% **endif** %}  
 </**ul**>  
 {% **endif** %}  
 </**nav**>  
 *<!-- Пагинация -->* {% **for auditorium in auditoriums** %}  
 <**a href="**{% **url 'auditorium' auditorium**.**id** %}**" class="list-group-item waves-effect"**>  
 {{ **auditorium**.**name** }} **&nbsp;**(  
 {% **if auditorium**.**auditoriumtypeid** %}  
 {{ **auditorium**.**auditoriumtypeid** }}  
 {% **endif** %}  
 {% **if auditorium**.**departmentid and auditorium**.**departmentid**.**name** != **'---'** %}  
 , кафедра {{ **auditorium**.**departmentid** }}  
 {% **endif** %}  
 )  
 </**a**>  
 {% **endfor** %}  
  
 *<!-- Пагинация -->* <**nav**>  
 {% **if auditoriums**.**has\_other\_pages** %}  
 <**ul class="pagination pg-blue white"**>  
 {% **if auditoriums**.**has\_previous** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **auditoriums**.**previous\_page\_number** }}**"**>**&laquo;**</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item disabled"**>  
 <**a class="page-link" href="#" tabindex="-1"**>**&laquo;**</**a**>  
 </**li**>  
 {% **endif** %}  
 {% **for i in auditoriums**.**paginator**.**page\_range** %}  
 {% **if auditoriums**.**number** == **i** %}  
 <**li class="page-item active"**>  
 <**a class="page-link" href="?page=**{{ **i** }}**"**>{{ **i** }}</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **i** }}**"**>{{ **i** }}</**a**>  
 </**li**>  
 {% **endif** %}  
 {% **endfor** %}  
 {% **if auditoriums**.**has\_next** %}  
 <**li class="page-item"**>  
 <**a class="page-link" href="?page=**{{ **auditoriums**.**next\_page\_number** }}**"**>**&raquo;**</**a**>  
 </**li**>  
 {% **else** %}  
 <**li class="page-item disabled"**><**span**>**&raquo;**</**span**></**li**>  
 {% **endif** %}  
 </**ul**>  
 {% **endif** %}  
 </**nav**>  
 *<!-- Пагинация -->* </**div**>  
 *<!-- Список аудиторий -->* </**div**>  
 </**div**>  
 </**div**>  
{% **endblock** %}

#### auditorium/templates/auditorium.html



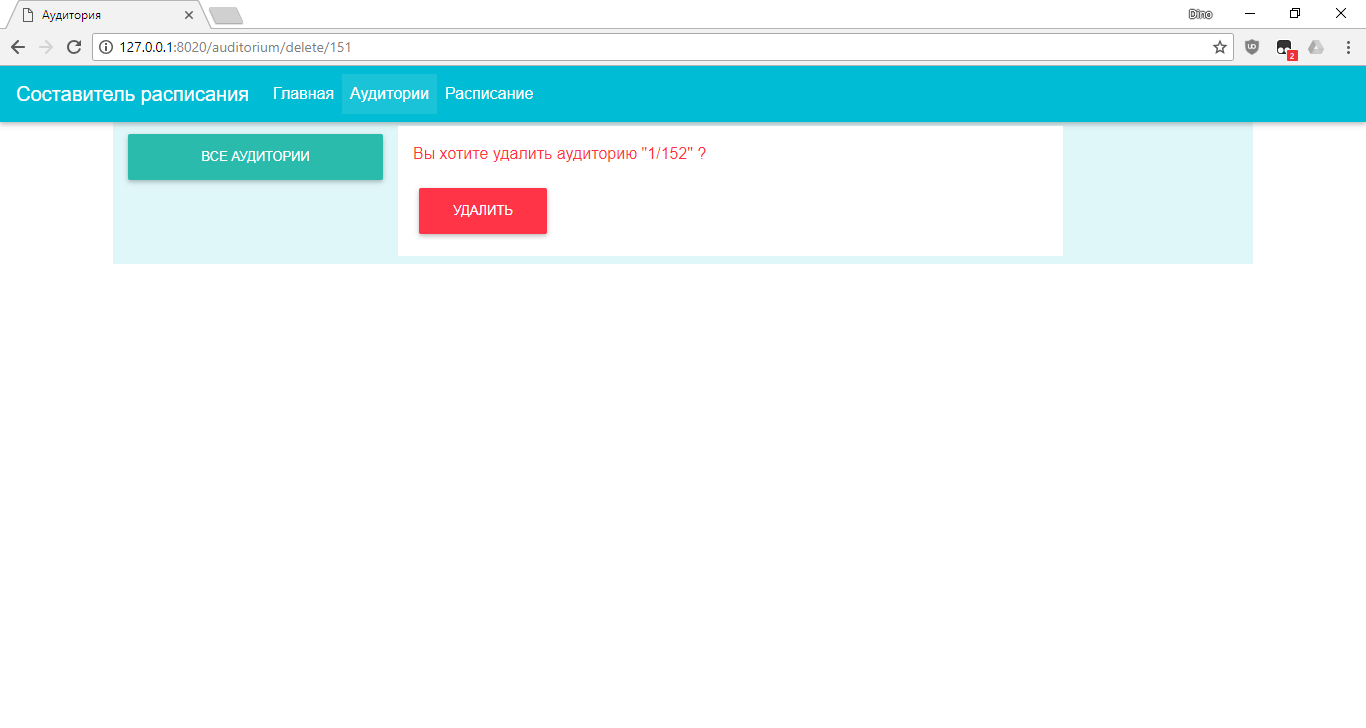
{% **extends "\_base.html"** %}  
  
{% **block title** %} Аудитория {% **endblock** %}  
  
*<!-- Навигация -->* {% **block nav** %}  
 <**li class="nav-item active"**>  
 <**a class="nav-link" href="**{% **url "auditorium\_list"** %}**"**>Аудитории </**a**>  
 </**li**>  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Еще что то</**a**>  
 </**li**>  
 {% **endblock** %}  
*<!-- Навигация -->*{% **block content** %}  
 <**div class="container pb-2"**>  
 <**div class="row cyan lighten-5 pb-2"**>  
 <**div class="col-3 mt-1"**>  
 <**a href="**{% **url 'auditorium\_list'** %}**" class="btn btn-default btn-block mt-1 mb-1" role="button"**>  
 Все аудитории  
 </**a**>  
 *<!--Список корпусов-->* <**div class="card mt-3"**>  
 <**div class="card-header"**> Корпус </**div**>  
 <**div class="card-body"**>  
 <**ul class="list-group list-group-flush"**>  
 {% **for building in buildings** %}  
 <**li class="list-group-item"**>  
 <**a href="**{% **url 'aud\_build' building**.**id** %}**" role="button"**>  
 {{ **building**.**fullname** }}  
 </**a**>  
 </**li**>  
 {% **endfor** %}  
 </**ul**>  
 </**div**>  
 </**div**>  
 *<!--Список корпусов-->* </**div**>  
 <**div class="col-7 mt-1"**>  
  
 *<!-- Информация об аудитории -->* <**div class="card white"**>  
 <**div class="card-body rounded-bottom"**>  
 <**h4 class="card-title text-center"**>{{ **auditorium**.**name** }}</**h4**>  
 <**hr class="hr-dark"**>  
 <**p**>  
 <**div class="row"**>  
 {% **if auditorium**.**departmentid** %}  
 <**div class="col-3"**>Кафедра:</**div**>  
 <**div class="col-6"**>{{ **auditorium**.**departmentid**.**fullname** }}({{ **auditorium**.**departmentid**.**name** }})</**div**>  
 {% **endif** %}  
 </**div**>  
 </**p**>  
 <**p**>  
 <**div class="row"**>  
 {% **if auditorium**.**seatingcapacity** != 0 %}  
 <**div class="col-3"**>Вместимость:</**div**>  
 <**div class="col-6"**>{{ **auditorium**.**seatingcapacity** }}</**div**>  
 {% **endif** %}  
 </**div**>  
 </**p**>  
 <**p**>  
 <**div class="row"**>  
 {% **if auditorium**.**auditoriumtypeid** %}  
 <**div class="col-3"**>Тип аудитории: </**div**>  
 <**div class="col-6"**>{{ **auditorium**.**auditoriumtypeid**.**fullname** }}</**div**>  
 {% **endif** %}  
 </**div**>  
 </**p**>  
 <**p**>  
 <**div class="row"**>  
 {% **if auditorium**.**buildingid** %}  
 <**div class="col-3"**>Корпус: </**div**>  
 <**div class="col-6"**>{{ **auditorium**.**buildingid** }}</**div**>  
 {% **endif** %}  
 </**div**>  
 </**p**>  
  
 <**a href="**{% **url "auditorium\_edit" auditorium**.**id** %}**" class="btn btn-outline-blue" role="button"**>  
 Редактировать  
 </**a**> <**a href="**{% **url "auditorium\_delete" auditorium**.**id** %}**" class="btn btn-outline-danger mt-1 mb-1" role="button"**>  
 Удалить  
 </**a**>  
  
 </**div**>  
 </**div**>  
 *<!-- Информация об аудитории -->* </**div**>  
 </**div**>  
 </**div**>  
{% **endblock** %}

#### auditorium/templates/auditorium\_form.html



{% **extends "\_base.html"** %}  
  
{% **block title** %} Аудитория {% **endblock** %}  
  
*<!-- Навигация -->* {% **block nav** %}  
 <**li class="nav-item active"**>  
 <**a class="nav-link" href="**{% **url "auditorium\_list"** %}**"**>Аудитории </**a**>  
 </**li**>  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Еще что то</**a**>  
 </**li**>  
 {% **endblock** %}  
*<!-- Навигация -->*{% **block content** %}  
 <**div class="container pb-2"**>  
 <**div class="row cyan lighten-5 pb-2"**>  
 <**div class="col-3 mt-1"**>  
 <**a href="**{% **url 'auditorium\_list'** %}**" class="btn btn-default btn-block mt-1 mb-1" role="button"**>  
 Все аудитории  
 </**a**>  
 </**div**>  
 <**div class="col-7 mt-1 white"**>  
  
 <**form**>  
  
 *<!-- Название аудитории -->* <**div class="form-group"**>  
 <**label for="inputName"**>Название аудитории</**label**>  
 {{ **form**.**name** }}  
 </**div**>  
  
 *<!-- Тип аудитории -->* <**div class="form-group"**>  
 <**label for="auditoriumTypeSelect"**>Тип аудитории</**label**>  
 {{ **form**.**auditoriumtypeid** }}  
 </**div**>  
  
 *<!-- Вместимость аудитории -->* <**div class="form-group"**>  
 <**label for="inputSeatingCapacity"**>Вместимость аудитории</**label**>  
 {{ **form**.**seatingcapacity** }}  
 </**div**>  
  
 *<!-- Кафедра, к которой привязана аудитория -->* <**div class="form-group"**>  
 <**label for="auditoriumDepartmentSelect"**>Кафедра, к которой привязана аудитория</**label**>  
 {{ **form**.**departmentid** }}  
 </**div**>  
  
 <**button type="submit" class="btn btn-info"**>Сохранить</**button**>  
  
 </**form**>  
  
 </**div**>  
 </**div**>  
 </**div**>  
{% **endblock** %}

#### auditorium/templates/auditorium\_confirm\_delete.html



{% **extends "\_base.html"** %}  
  
{% **block title** %} Аудитория {% **endblock** %}  
  
*<!-- Навигация -->* {% **block nav** %}  
 <**li class="nav-item active"**>  
 <**a class="nav-link" href="**{% **url "auditorium\_list"** %}**"**>Аудитории </**a**>  
 </**li**>  
 <**li class="nav-item"**>  
 <**a class="nav-link" href="#"**>Еще что то</**a**>  
 </**li**>  
 {% **endblock** %}  
*<!-- Навигация -->*{% **block content** %}  
 <**div class="container pb-2"**>  
 <**div class="row cyan lighten-5 pb-2"**>  
 <**div class="col-3 mt-1"**>  
 <**a href="**{% **url 'auditorium\_list'** %}**" class="btn btn-default btn-block mt-1 mb-1" role="button"**>  
 Все аудитории  
 </**a**>  
  
 </**div**>  
 <**div class="col-7 mt-1 white"**>  
  
 <**form method="post" class="mb-3 mt-3"**>  
 <**p class="text-danger"**>  
 Вы хотите удалить аудиторию "{{ **auditorium** }}" ?  
 </**p**>  
 <**input type="submit" value="Удалить" class="btn btn-danger"** />  
 </**form**>  
  
 </**div**>  
 </**div**>  
 </**div**>  
{% **endblock** %}