

Памятка

Тебе нужно будет распаковать все
и запустить в терминале

```
docker compose up --build
```

Создание форм из моделей. Демонстрация создания,
редактирования, удаления на сайте. Можно разные
таблицы.

/students/

lavbar Login Github Login

Name	Faculty	Enrollment	Graduation	Vacancies
JOHN DOE	Faculty of Cyber Security	Сен 08, 2020	Дек 08, 2023	<div>Vacancies:</div> <ul style="list-style-type: none">Data AnalystData Analyst

Страница 1 из 1

Create Student

/students/create/

Navbar Login Github Login

New Student

User:

Name:

Enrollment year:

Graduation year:

Faculty:

Resume:

Save

Back to List

Это страница создания

в файле `students/views.py`

```
def create_view(request):
    form = StudentForm(request.POST or None)
    if form.is_valid():
        form.save()
        return redirect('student-list-view')
    return render(request, 'students/create.html', {'form': form})
```

`/students/1/` (1 - это id студента)

Navbar Login Github Login

John Doe

User: admin

Name: John Doe

Enrollment year: 8 Сентябрь 2020

Graduation year: 8 Декабрь 2023

Faculty: Faculty of Cyber Security

Resume: My Resume - 2024-12-08 09:44:48.077247+00:00

Save

[Back to List](#)

Это страница обновления, на нее ты можешь перейти по кнопке **Detail**

```
def detail_view(request, pk):
    student = get_object_or_404(Student, pk=pk)
    form = StudentForm(request.POST or None, instance=student)
    if form.is_valid():
        form.save()

    return render(request, 'students/detail.html', {'student': student,
                                                    'form': form})
```

`/students/1/delete/`

Are you sure you want to delete "John Doe"?

[Cancel](#)

Это страница удаления, на нее ты можешь перейти по кнопке **Delete student**

```
def delete_view(request, pk):
    student = get_object_or_404(Student, pk=pk)
    if request.method == 'POST':
        student.delete()
        return redirect('student-list-view')
    return render(request, 'students/delete.html', {'student': student})
```

Файл requirements.txt

```
django-celery-beat==2.6.0
django==4.2
djangorestframework==3.14.0
django-filter==23.2
drf-yasg==1.21.5 # Если потребуется документация
django-simple-history==3.7.0
django-import-export==3.2.0
django-redis==5.4.0
pylint==3.2.3
django-allauth==0.63.3
social-auth-app-django==5.4.1
social-auth-core==4.5.4
pillow==11.1.0
django-debug-toolbar
```

Здесь прописаны все нужные библиотеки

Использование метода save() в модели

файл students/models.py

строка 60

```
def save(self, *args, **kwargs):
    # Проверка и автоматическое заполнение поля graduation_year
    if self.enrollment_year and not self.graduation_year:
        self.graduation_year = self.enrollment_year + timedelta(days=4 *
365)

    # Приведение имени к формату "Первая буква заглавная"
    if self.name:
        self.name = self.name.title()

    # Вызов метода full_clean() для валидации данных
    self.full_clean() # Проверит clean_<fieldname>(), если есть
    super().save(*args, **kwargs) # Вызов стандартного сохранения
```

Meta widgets в моделях

файл students/forms.py

строка 13

```
widgets = {
    'enrollment_year': forms.SelectDateWidget(years=range(2000,
2030)),
    'graduation_year': forms.SelectDateWidget(years=range(2000,
2030)),
}
```

Пример *clean_<fieldname>()*

файл students/forms.py

строка 18

```
def clean_enrollment_year(self):
    enrollment_year = self.cleaned_data.get('enrollment_year')
    if enrollment_year and enrollment_year.year > now().year:
        raise forms.ValidationError("Enrollment year cannot be in the
future.")
    return enrollment_year
```

```
def clean_graduation_year(self):
    enrollment_year = self.cleaned_data.get('enrollment_year')
    graduation_year = self.cleaned_data.get('graduation_year')
    if graduation_year and enrollment_year and graduation_year <
enrollment_year:
        raise forms.ValidationError("Graduation year cannot be earlier
than enrollment year.")
    return graduation_year
```

def save с commit=True

файл students/views.py

строка 39

```
form.save()
```

models.ManyToManyField с параметром through

файл students/models.py

строка 55

```
vacancies = models.ManyToManyField(Vacancy,
through="application.Application")
```

select_related(), prefetch_related()

файл students/views.py

строка 13

```
students = Student.objects.prefetch_related('vacancies',
'resume').select_related('faculty')
```

django-debug-toolbar

NavbarВыйти

Hello World! alexmintt!

Title	Description	Company	Salary
Data Analyst	Analyze data for insights. Remote	HealthCare Inc	60000,00
All (1)			

Скрыть »

История

/

Версии

Django 4.2

Время

CPU: 66.40ms (69.87ms)

Настройки

Заголовки

Запрос

list

SQL

3 queries in 0.73ms

Статические файлы

0 файлов используется

Шаблоны

vacancy_list.html

Кэш

0 обращений за 0.00ms

Сигналы

36 получателей 15 сигнала(ов)

Перехват
редиректов

Профилирование

Это вот эта панель справа