**Московский государственный технический университет**

**им. Н.Э. Баумана**

**Факультет «Информатика и управление»**

Отчёт по лабораторной работе № 6

«Работа с формами, авторизация и модуль администрирования в Django»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-52 |  | преподаватель каф. ИУ5 |
| Гринин Олег |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

**1.** **Задание**

Основная цель данной лабораторной работы – научиться обрабатывать веб-формы на стороне приложения, освоить инструменты, которые предоставляет Django, по работе с формами. Также в этой лабораторной работе вы освоите инструменты Django по работе с авторизацией и реализуете простейшую авторизацию. Напоследок, вы познакомитесь с инструментом администрирования Django – как в несколько строчек кода сделать панель администратора сайта.

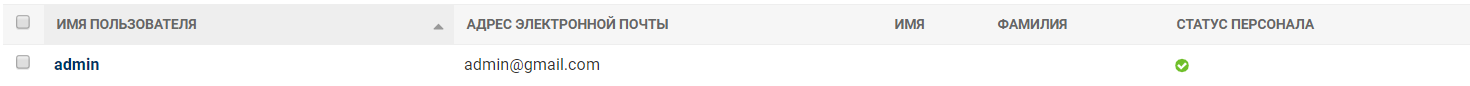
**2.** **Выполнение** **задания** **и** **исходный** **код** **проекта** Создайте view, которая возвращает форму для регистрации**:**

|  |  |
| --- | --- |
| **# Форма регистрации** | |
|  | **def register(request):** |
|  | **if request.method == 'POST':** |
|  | **user\_form = UserRegistrationForm(request.POST)** |
|  | **if user\_form.is\_valid():** |
|  | **user\_form.save()** |
|  | **return render(request, 'registration/register\_done.html')** |
|  | **else:** |
|  | **user\_form = UserRegistrationForm()** |
|  | **return render(request, 'registration/register.html', {'user\_form': user\_form})** |

|  |  |
| --- | --- |
|  | |
| **class UserRegistrationForm(forms.ModelForm):** | |
|  | **password = forms.CharField(label='Password', widget=forms.PasswordInput)** |
|  | **password2 = forms.CharField(label='Repeat password', widget=forms.PasswordInput)** |
|  |  |
|  | **class Meta:** |
|  | **model = User** |
|  | **fields = ('username', 'first\_name', 'email')** |
|  |  |
|  | **def clean\_password2(self):** |
|  | **cd = self.cleaned\_data** |
|  | **if cd['password'] != cd['password2']:** |
|  | **raise forms.ValidationError('Passwords don\'t match.')** |
|  | **return cd['password2']** |
|  |  |

Для view будет использоваться декоратор @login\_required для функций, а для классов будем использовать LoginRequiredMixin

Добавить superuser’a через команду manage.py



Настройка админ-панели

|  |  |
| --- | --- |
|  | **class PostAdmin(admin.ModelAdmin):** |
|  | **list\_display = ('title', 'slug', 'author', 'publish', 'status', 'author', 'image')** |
|  | **list\_filter = ('status', 'created', 'publish', 'author')** |
|  | **search\_fields = ('title', 'body')** |
|  | **prepopulated\_fields = {'slug': ('title',)}** |
|  | **date\_hierarchy = 'publish'** |
|  | **ordering = ['status', 'publish']** |
|  |  |
|  |  |
|  | **admin.site.register(Book, PostAdmin)** |
|  |  |
|  |  |
|  | **class CommentAdmin(admin.ModelAdmin):** |
|  | **list\_display = ('name', 'book', 'created', 'active', 'imagin')** |
|  | **list\_filter = ('active', 'created')** |
|  | **search\_fields = ('name', 'body')** |
|  | **raw\_id\_fields = ('book', 'name')** |
|  |  |
|  |  |
|  | **admin.site.register(Comment, CommentAdmin)** |
|  |  |
|  |  |
|  | **class ProfileAdmin(admin.ModelAdmin):** |
|  | **list\_display = ['user', 'photo']** |
|  |  |
|  |  |
|  | **admin.site.register(Profile, ProfileAdmin)** |