Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»



**Отчет**

#### По Домашнему заданию

#### По курсу «Разработка интернет приложений»

«Разработка веб-приложенияоснове технологий Python, Django, JavaScript, MySQL»

Вариант 19

**ИСПОЛНИТЕЛЬ:**

Попов Илья

Группа ИУ5-51Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г.

Москва 2020

Задание

Разрабатываемое веб-приложение должно выполнять указанные ниже функции. Функции разбиты на группы. Для каждой функции указано количество рейтинговых баллов, которые получает студент в случае реализации данной функции.

Регистрация

1. При входе, если не залогинен, открывается регистрация (1 балл)
2. Есть рабочая кнопка выхода (1 балл)

Авторизация

1. Автоматически логинится, редирект на главную страницу (список сущностей). (1 балл)
2. Если логин / пароль неправильный, показана ошибка (1 балл)
3. Профиль пользователя использует стороннюю модель либо переопределена основная модель юзера (1 балл)

Список сущностей

1. Страница открывается и данные берутся из базы. (1 балл)
2. Отображаются картинки сущностей (1 балл)
3. Есть пагинация (1 балл)
4. Пагинация или бесконечный скролл реализован на AJAX. (1 балл)

Страница добавления сущностей

1. Реализована в виде модального окна для добавления сущностей. (1 балл)
2. Добавление сущности происходит по AJAX (1 балл)
3. Есть форма с необходимыми для сущности полями (1 балл)
4. Есть возможность загрузить картинку (1 балл)
5. В случае ошибок они отображаются (1 балл)
6. Валидация основных полей формы происходит с помощью JS (1 балл)
7. Сущность создается в базе данных и редирект на страницу с сущностью (1 балл)

Страница с сущностью

1. Отображаются данные из базы данных (1 балл)
2. При нажатии кнопки действия происходит добавление этого действия в базу, страница перезагружается, это новое действие добавляется в список действий. (1 балл)
3. Действие выполняется без перезагрузки (AJAX) (1 балл)
4. Список действий всех пользователей над этой сущностью (1 балл)

Страница личного кабинета пользователя

1. Возможность загрузить аватарку (1 балл)
2. Возможность изменить поля пользователя (1 балл)

Django-админка

1. Русификация. (1 балл)
2. Работа со всеми сущностями. (1 балл)
3. Фильтрация и поиск в админке основной сущности. (1 балл)

Deploy

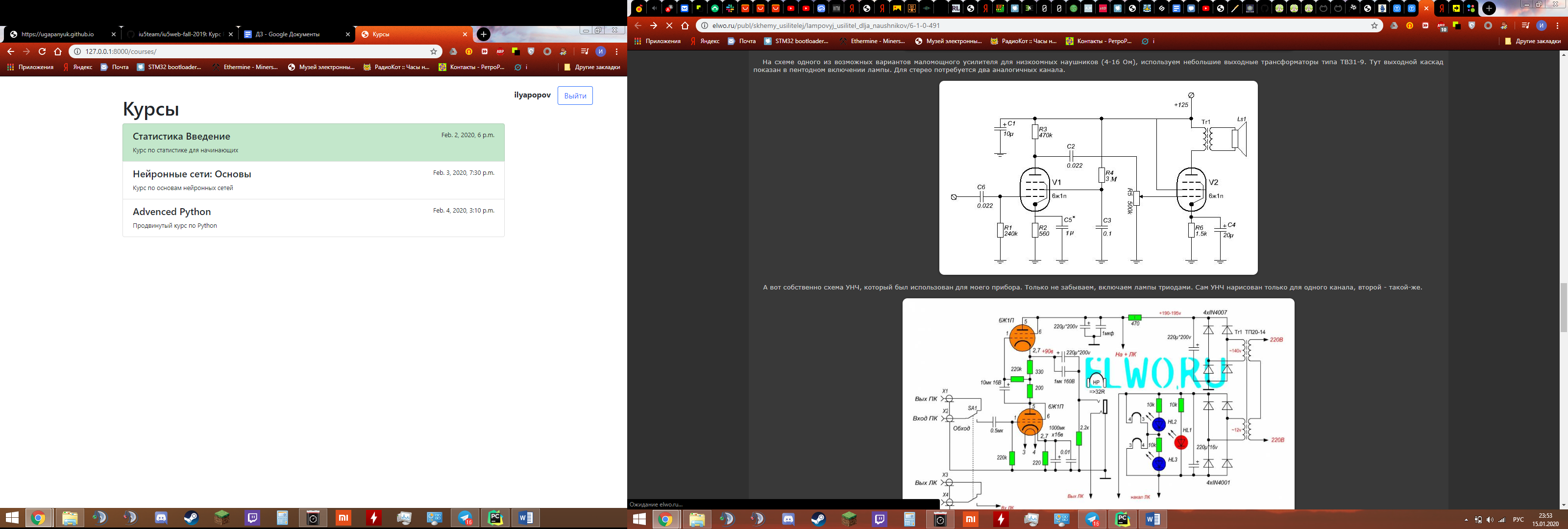
1. Настроен mysql (1 балл)
2. Настроен nginx (1 балл)
3. Настроен gunicorn (1 балл)
4. Проект находится в git-репозитории (1 балл)

Вариант 19

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Субъект** | **Отношение** | **Объект** |
| 19 | Пользователь | Участие | Курс |

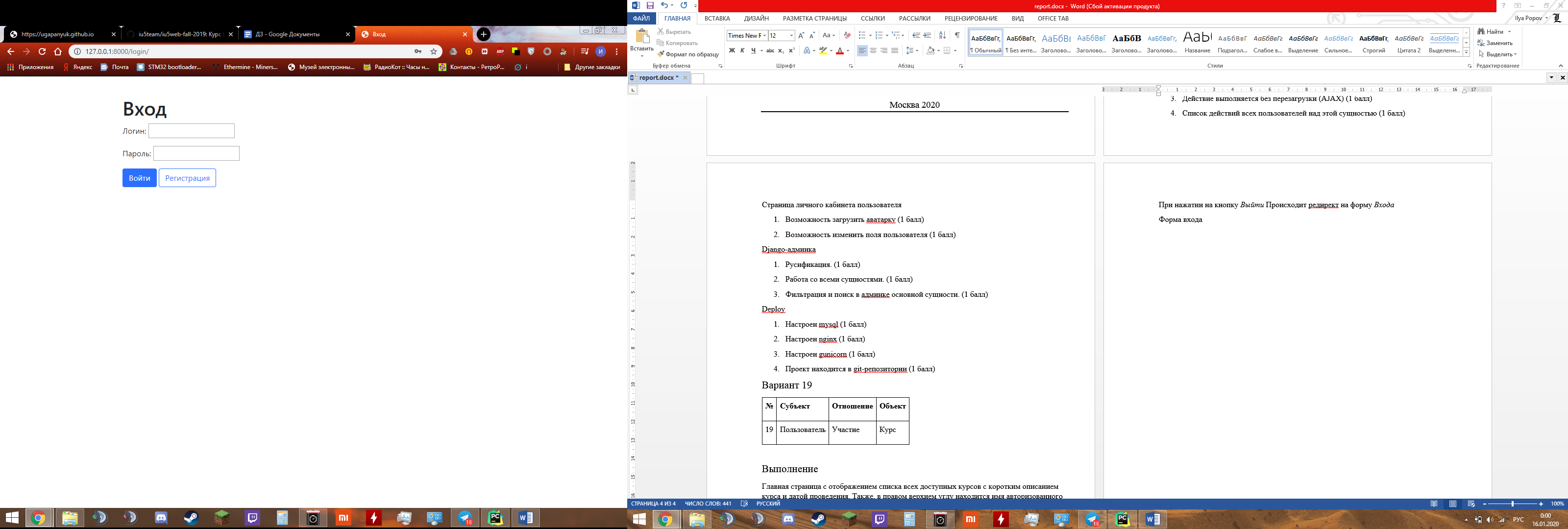
Выполнение

Главная страница с отображением списка всех доступных курсов с коротким описанием курса и датой проведения. Также, в правом верхнем углу находится имя авторизованного пользователя и кнопка выхода из аккаунта. Зелёным отмечены курсы, на которые записался пользователь. Курсы выделяются зелёным сразу после загрузки страницы, по средствам *ajax-запроса.* Таким же образом, при нажатии на курс происходит запись или «отписка» от курса (в зависимости от того, был пользователь записан на курс или нет) без перезагрузки страницы.

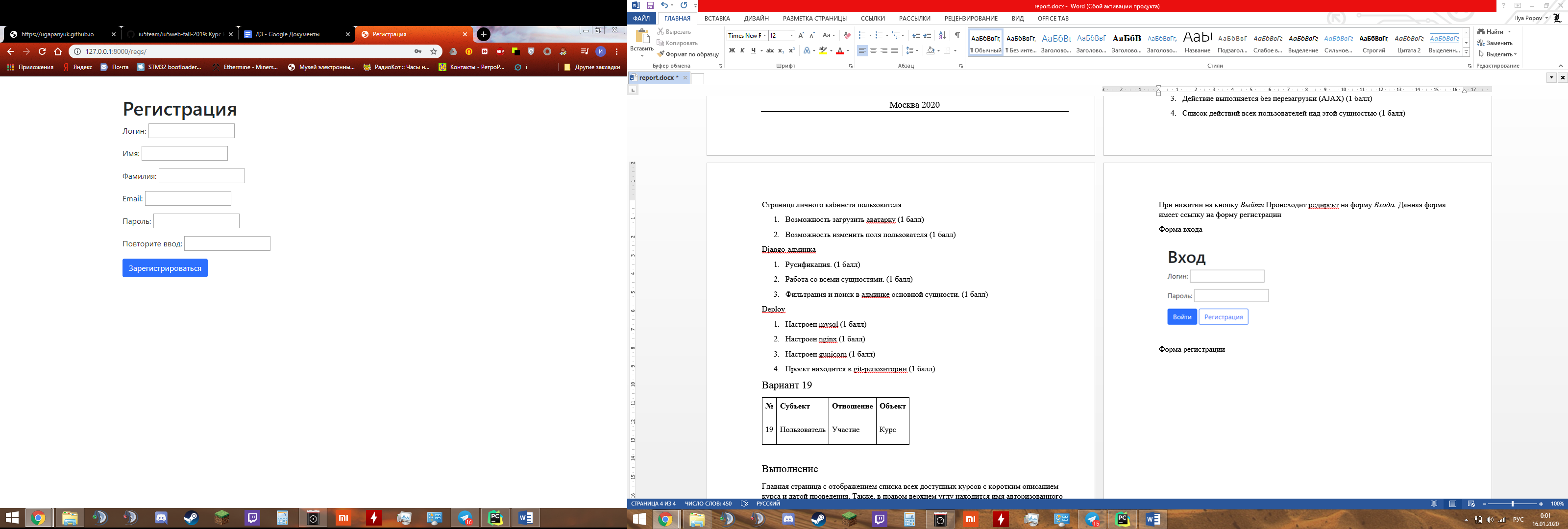


При нажатии на кнопку *Выйти* Происходит редирект на форму *Входа.* Данная форма имеет ссылку на форму *Регистрации*. После успешной регистрации происходит перенаправление на форму *Входа.* После успешной авторизации происходит перенаправление на основную страницу *Курсы*. При попытке зайти на основную страницу неавторизованного пользователя, происходит перенаправления на форму *Входа.*

Форма входа

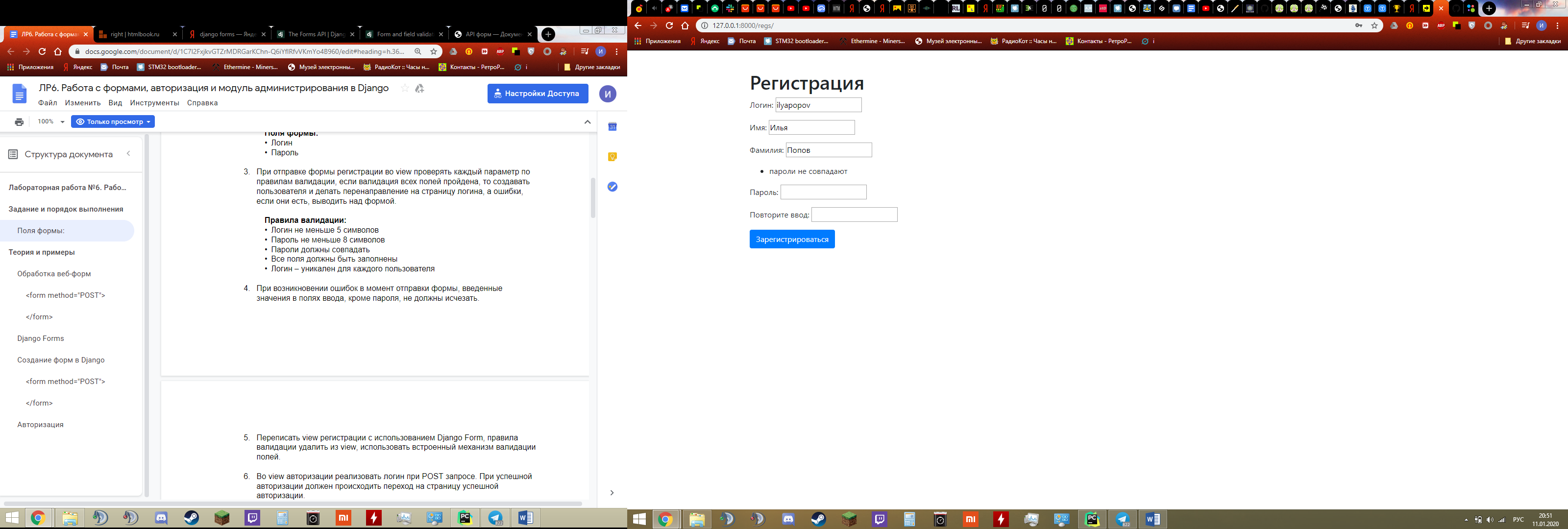


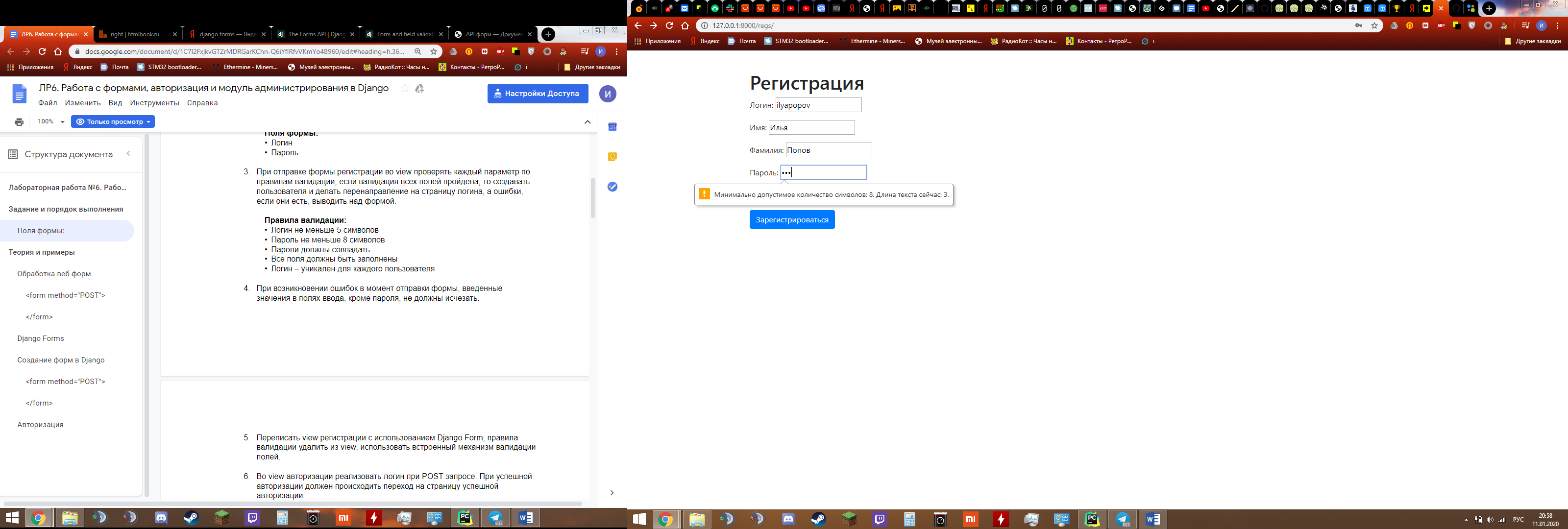
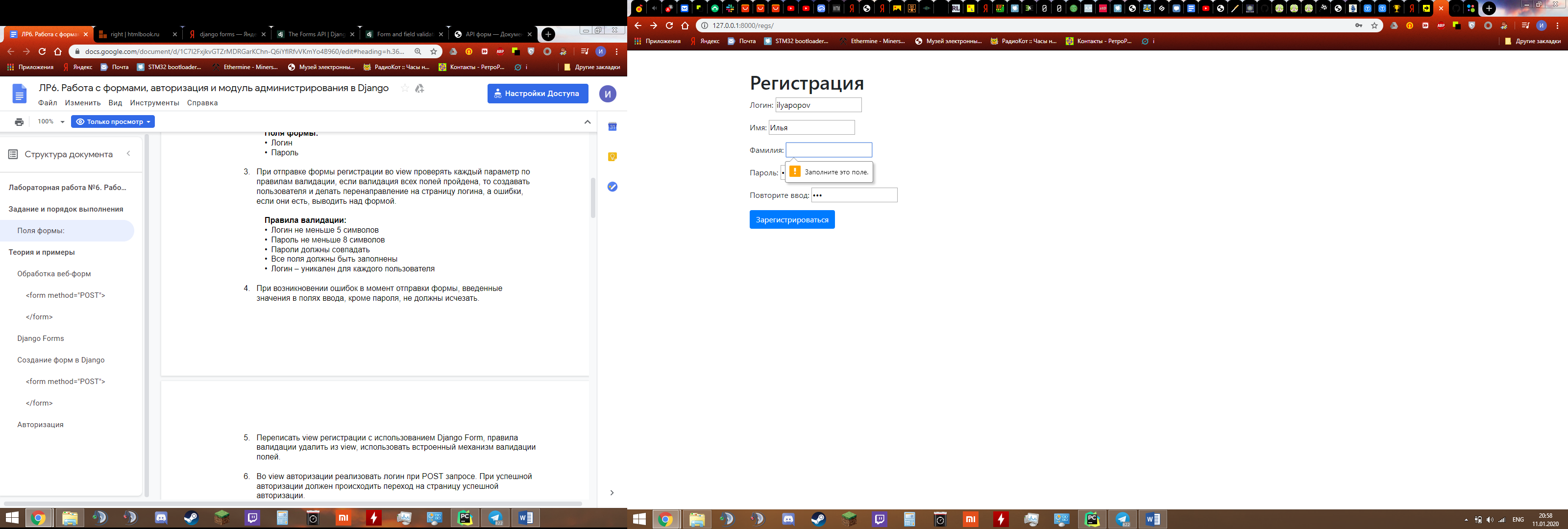
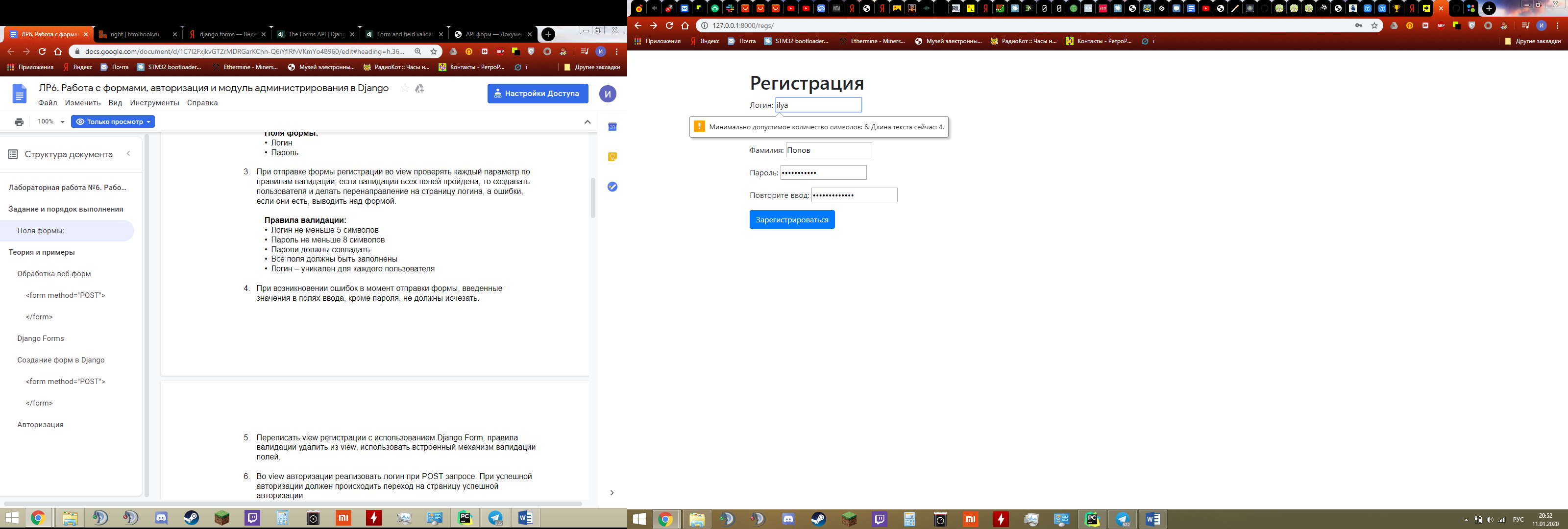
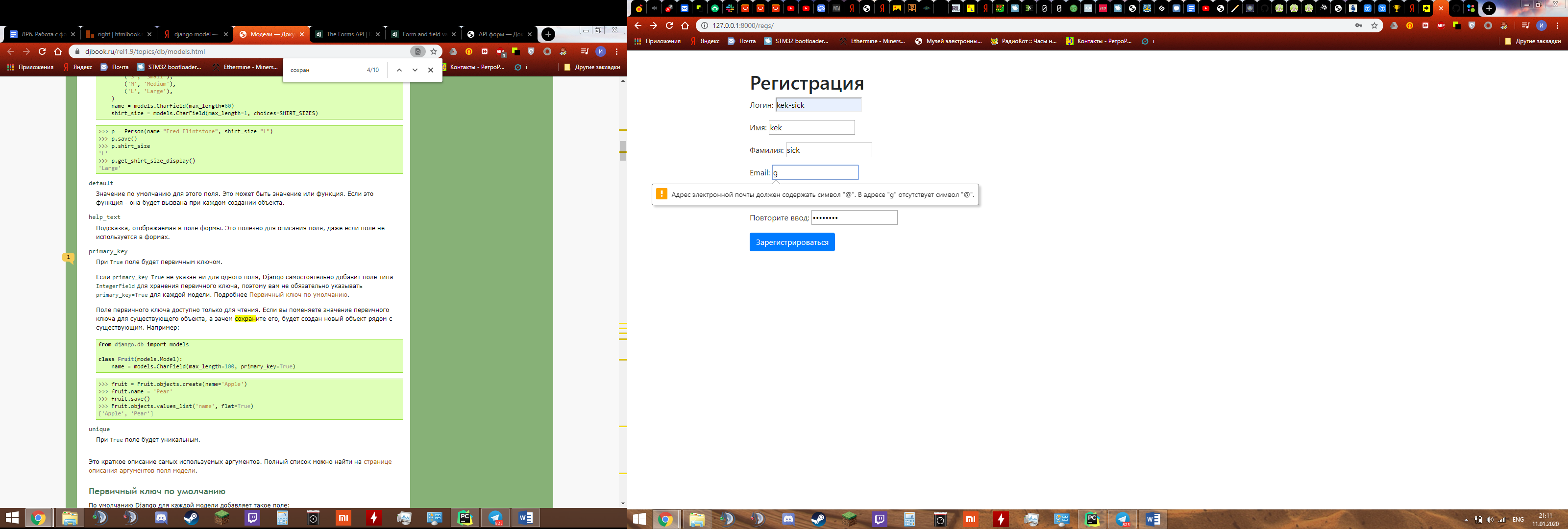
Форма регистрации



Формы имеют валидацию и показывают различные ошибки

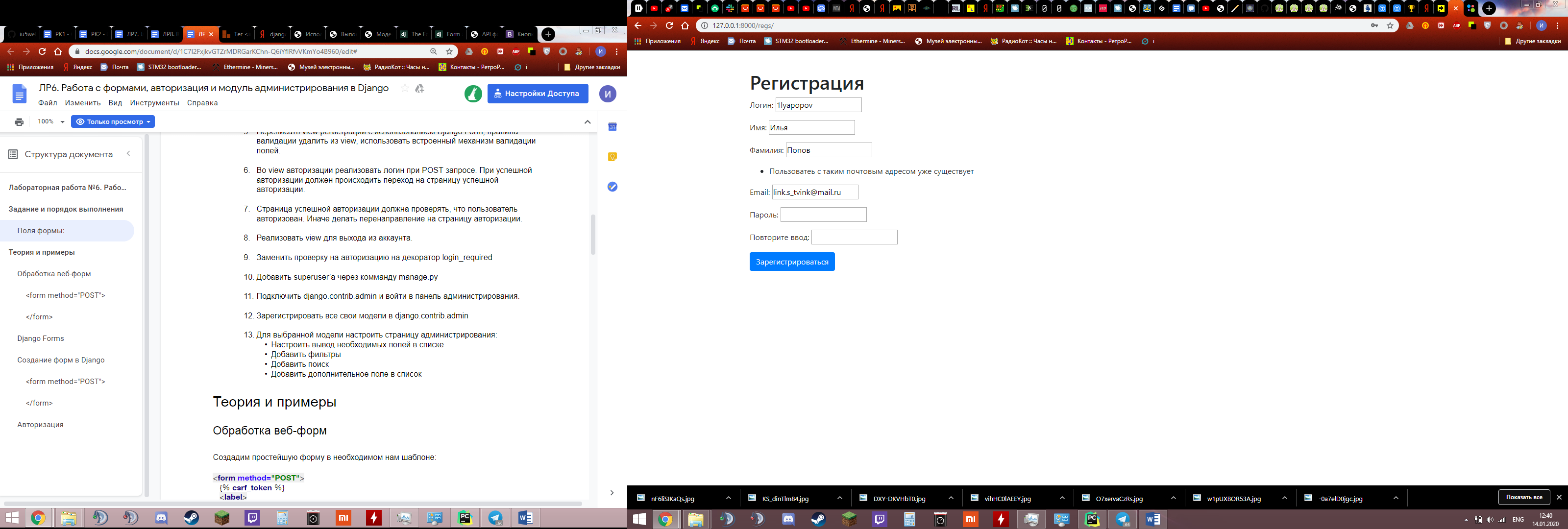
Вывод ошибки при несовпадающих паролях



Вывод ошибки при несоблюдении формата и минимальной длины слова в поле

Вывод ошибок при попытке регистрации пользователей с одинаковыми именами или адресами электронных почт





Исходный код

**urls.py**

from django.contrib import admin  
from django.urls import path  
from my\_app.views import RegistrationS, Login, Courses, SetParticipation  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path(r'regs/', RegistrationS.as\_view()),  
 path(r'courses/', Courses.as\_view()),  
 path(r'login/', Login.as\_view()),  
 path(r'courses/setcourse/', SetParticipation.as\_view()),  
]

**admin.py**

from django.contrib import admin  
from my\_app.models import Course, Participation  
  
  
class CourseAdmin(admin.ModelAdmin):  
 pass  
  
  
class ParticipationAdmin(admin.ModelAdmin):  
 pass  
  
  
admin.site.register(Course, CourseAdmin)  
admin.site.register(Participation, ParticipationAdmin)

**views.py**

from django.http import HttpResponse, HttpResponseRedirect, JsonResponse  
from django.views import View  
from django.shortcuts import render  
from django import forms  
from my\_app.models import Course, Participation  
from django.contrib.auth import authenticate, login, logout  
from django.contrib.auth.models import User  
  
  
class Courses(View):  
 def get(self, request):  
 user = 'nobody'  
 if request.user.is\_authenticated:  
 user = request.user.username  
 else:  
 return HttpResponseRedirect('/login/')  
 return render(request, 'courses.html', {  
 'list': Course.objects.all(),  
 'user': user,  
 })  
  
 def post(self, request):  
 if request.POST['data'] == 'logout':  
 logout(request)  
 return HttpResponseRedirect('/login/')  
  
 if request.POST['data'] == 'getcourses':  
 my\_courses = Participation.objects.filter(user\_id=request.user.id)  
 selected\_courses1 = {a.course\_id: a.course\_id for a in my\_courses}  
 return JsonResponse(selected\_courses1)  
  
  
class SetParticipation(View):  
 def post(self, request):  
 if request.POST['data'] == 'setcourse':  
 req = request.POST  
 course\_id = req['course']  
 user = request.user.id  
 if req['action'] == '1':  
 participation = Participation(user\_id=user, course\_id=course\_id)  
 participation.save()  
 else:  
 participation = Participation.objects.filter(user\_id=user, course\_id=course\_id)  
 participation.delete()  
 return HttpResponse('1')  
  
  
class RegistrationS(View):  
 def get(self, request):  
 form = RegistrationForm()  
 return render(request, 'registration\_form\_simple.html', {'form': form})  
  
 def post(self, request):  
 errors = []  
 if request.method == 'POST':  
 form = RegistrationForm(request.POST)  
 if form.data['password'] != form.data['password2']:  
 form.add\_error('password', 'пароли не совпадают')  
 elif User.objects.filter(username=form.data['username']).count() != 0:  
 form.add\_error('username', 'Пользоватеь с таким Логином уже существует')  
 elif User.objects.filter(email=form.data['email']).count() != 0:  
 form.add\_error('email', 'Пользоватеь с таким почтовым адресом уже существует')  
 elif form.is\_valid():  
 # успех... создание модели пользователя  
 new\_user = User.objects.create\_user(form.data['username'], form.data['email'], form.data['password'])  
 new\_user.first\_name = form.data['first\_name']  
 new\_user.last\_name = form.data['last\_name']  
 new\_user.save()  
 return HttpResponseRedirect('/login/')  
 else:  
 form = RegistrationForm()  
 return render(request, 'registration\_form\_simple.html', {'form': form})  
  
  
class RegistrationForm(forms.Form):  
 username = forms.CharField(min\_length=6, label='Логин')  
 first\_name = forms.CharField(label='Имя', min\_length=1)  
 last\_name = forms.CharField(label='Фамилия', min\_length=1)  
 email = forms.EmailField(label='Email')  
 password = forms.CharField(min\_length=8, widget=forms.PasswordInput, label='Пароль')  
 password2 = forms.CharField(min\_length=8, widget=forms.PasswordInput, label='Повторите ввод')  
  
  
class Login(View):  
 def get(self, request):  
  
 form = LoginForm()  
 return render(request, 'login\_form.html', {'form': form})  
  
 def post(self, request):  
 if request.method == 'POST':  
 form = LoginForm(request.POST)  
 form\_login = form.data['username']  
 form\_password = form.data['password']  
 user = authenticate(username=form\_login, password=form\_password)  
 if user:  
 login(request, user)  
 return HttpResponseRedirect('/courses/')  
 else:  
 form.add\_error('username', 'Неверный Логин или Пароль')  
 else:  
 form = LoginForm()  
 return render(request, 'login\_form.html', {'form': form})  
  
  
class LoginForm(forms.Form):  
 username = forms.CharField(min\_length=6, label='Логин')  
 password = forms.CharField(min\_length=8, widget=forms.PasswordInput, label='Пароль')

**models.py**

from django.db import models  
  
  
class User(models.Model):  
 first\_name = models.CharField(max\_length=60)  
 last\_name = models.CharField(max\_length=60)  
 login = models.CharField(max\_length=50, unique=True)  
 password = models.CharField(max\_length=100)  
 email = models.EmailField(null=True)  
  
  
class Course(models.Model):  
 title = models.CharField(max\_length=60)  
 description = models.CharField(max\_length=255)  
 date = models.DateTimeField()  
  
  
class Participation(models.Model):  
 user\_id = models.IntegerField()  
 course\_id = models.IntegerField()

**registration\_simple\_form.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">  
 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>  
 <meta charset="UTF-8">  
 <title>Регистрация</title>  
 <style>  
 .simpleForm{  
 padding-top: 40px;  
 padding-left: 250px;  
 padding-right: 250px;  
 }  
 label{  
 margin: 0;  
 }  
 </style>  
</head>  
<body>  
<form method="POST">  
 {% csrf\_token %}  
 <div class="simpleForm">  
 <h1>Регистрация</h1>  
 {{ form.as\_p }}  
 <button type="submit" class="btn btn-primary">Зарегистрироваться</button>  
 </div>  
</form>  
</body>  
</html>

**login\_form.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">  
 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>  
 <meta charset="UTF-8">  
 <title>Вход</title>  
 <style>  
 .simpleForm{  
 padding-top: 40px;  
 padding-left: 250px;  
 padding-right: 250px;  
 }  
 label{  
 margin: 0;  
 }  
 </style>  
</head>  
<body>  
<form method="POST">  
 {% csrf\_token %}  
 <div class="simpleForm">  
 <h1>Вход</h1>  
 {{ form.as\_p }}  
 <button type="submit" class="btn btn-primary">Войти</button>  
 <a class="btn btn-outline-primary" href="http://127.0.0.1:8000/regs/" role="button">Регистрация</a>  
 </div>  
</form>  
</body>  
</html>

**courses.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <script src="https://code.jquery.com/jquery-2.2.4.min.js"></script>  
 <script src="/static/my\_app/script/jquery.cookie.js"></script>  
 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">  
 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>  
 <meta charset="UTF-8">  
 <title>Курсы</title>  
 <script src="/static/my\_app/script/courses.js"></script>  
 <style>  
 .coursesList{  
 margin-top: 40px;  
 padding-left: 250px;  
 padding-right: 250px;  
 }  
 .user{  
 margin-right: 50px;  
 top: 20px;  
 right: 20px;  
 position: fixed;  
 }  
 .logout{  
 margin-left: 10px;  
 }  
 </style>  
</head>  
<body>  
<div class="user">  
 <form method="post" name="logout">  
 {% csrf\_token %}  
 <b>{{ user }}</b>  
 <input type="hidden" name="data" value="logout">  
 <button type="submit" class="btn btn-outline-primary logout">Выйти</button>  
 </form>  
</div>  
<div class="coursesList">  
 <h1>Курсы</h1>  
 <div class="list-group">  
 {% for element in list %}  
 <a id="{{ element.id }}" class="course list-group-item list-group-item-action">  
 <div class="d-flex w-100 justify-content-between">  
 <h5 class="mb-1">{{element.title}}</h5>  
 <small>{{ element.date }}</small>  
 </div>  
 <small>{{ element.description }}</small>  
 </a>  
 {% endfor %}  
 </div>  
</div>  
</body>  
</html>

**courses.js**

$(function () {  
 let csrftoken = $.cookie('csrftoken');  
  
 $(".course").click(function () {  
 let action;  
 if ($(this).hasClass('list-group-item-success')){  
 $(this).removeClass('list-group-item-success')  
 action = 0  
 }else {  
 $(this).addClass('list-group-item-success');  
 action = 1  
 }  
 $.ajax({  
 method: 'post',  
 url: 'setcourse/',  
 dataType: 'json',  
 data: {  
 data: 'setcourse',  
 action: action,  
 course: this.id,  
 csrfmiddlewaretoken: csrftoken  
 },  
 })  
 });  
   
 $.ajax({  
 method: 'post',  
 url: '',  
 dataType: 'json',  
 data: {  
 data: 'getcourses',  
 csrfmiddlewaretoken: csrftoken  
 },  
 }).complete(function (msg) {  
 for (key in msg['responseJSON']){  
 let id\_selector = '#' + key;  
 $(id\_selector).addClass('list-group-item-success')  
 }  
 })  
});