МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

Московский авиационный институт

«Национальный исследовательский университет»

Институт № 3 «Системы управления, информатика и электроэнергетика»

Кафедра 304: «Вычислительные машины, системы и сети»

Отчёт по лабораторной работе №16 «Установка Web-сервера NGINX (начало).»

по учебной дисциплине «Web-технологии»

Выполнил:

студент группы М3О-107БВ-24

Коркина К.В.

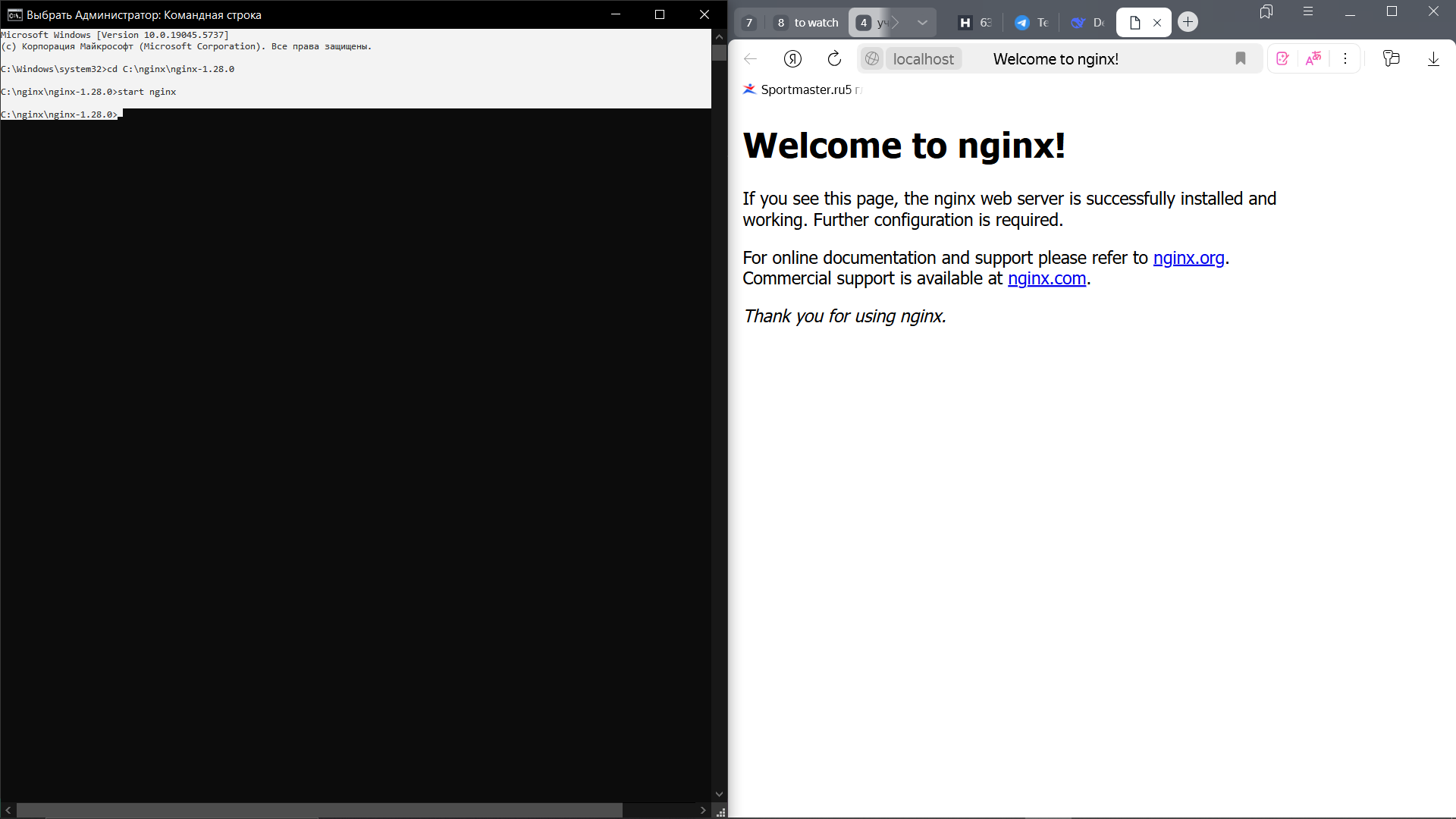
Приняла: Масленикова Т.Ю.

Титов Ю.П.

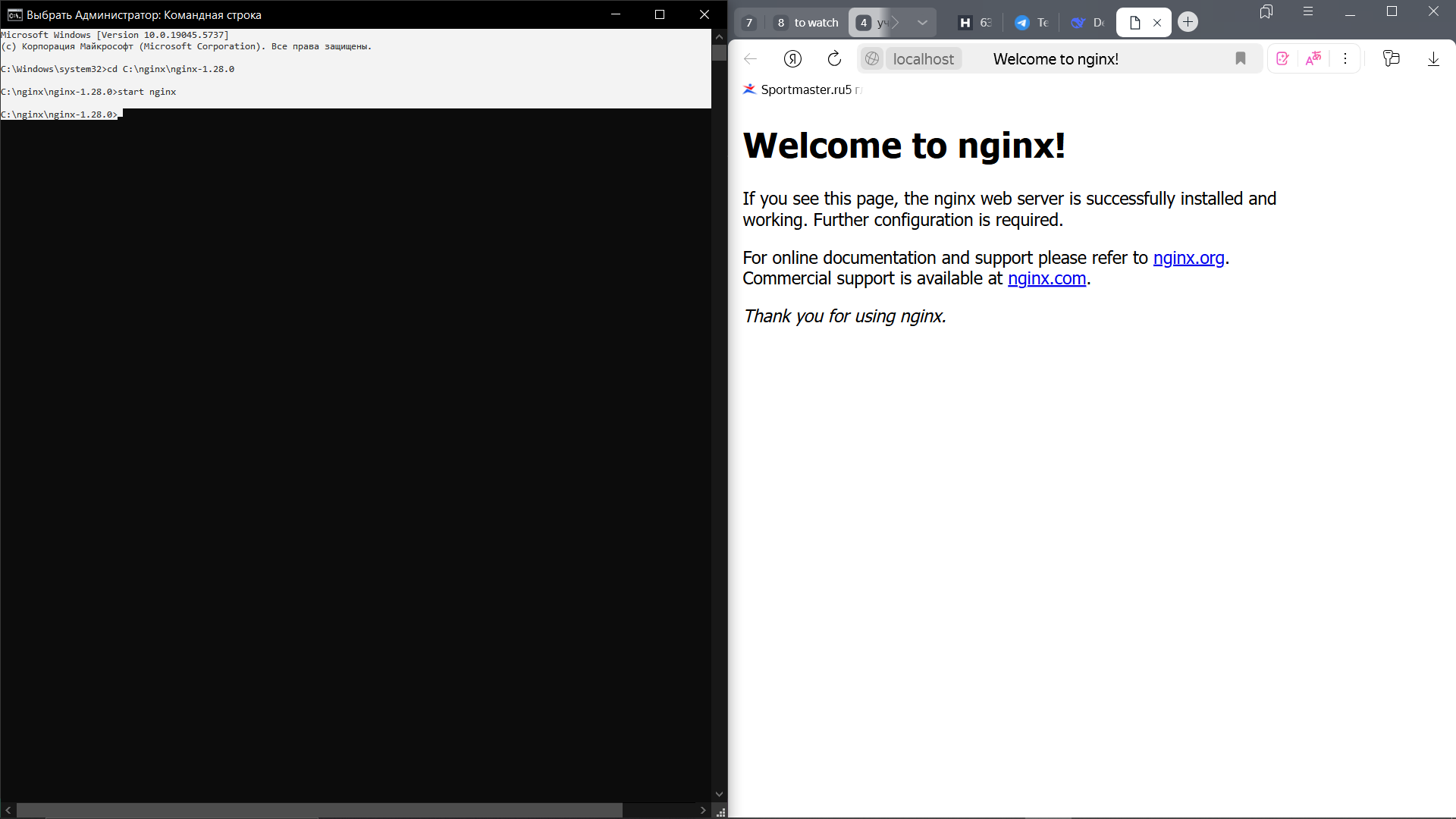
Москва 2025

1. Установить NGINX на компьютер, флеш-накопитель или виртуальную облачную машину. Рекомендуется использование ОС Linux (например Ubuntu), но возможна и установка ограниченной версии под ОС Windows. В отчете привести процесс установки (действия).

Рассматривается установка версии под ОС Windows (Stable version nginx-1.28.0). Архив распакован в папку (путь – C:\nginx). Командная строка от имени администратора имеет вид (открыть папку с распакованными из архива файлами и запущен nginx):



Для проверки в браузере открыто: <http://localhost>:



Так выглядит стандартная страница приветствия NGINX после успешной установки. По умолчанию NGINX показывает эту страницу, потому что в конфигурации (nginx.conf) указана папка html (Windows), где лежит этот файл, он ещё не заменён нашим сайтом.

1. Взять web-страницу, содержащую текст HTML, CSS, JS и картинки в различных форматах. Запустить отображение этой страницы на стандартном порту 80 с применением web-сервера NGINX.

**Index.html:**

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Сайтик</title>

<link rel="stylesheet" href="in.css">

</head>

<body>

<main>

<h2>Это джпгшный котик</h2>

<img src="/images/cat.jpg" alt="Котик">

<h2>Это пнгшный гриб</h2>

<img src="/images/logo.png" alt="Гриб">

<h2>Это свгшный магический шар</h2>

<img src="/images/icon.svg" alt="Магический шар">

<button id="myButton">Нажми меня</button>

</main>

<footer>

<p>© 2025 Котик, гриб и магический шар</p>

</footer>

<script src="script.js"></script>

</body>

</html>

**in.css:**

body {

font-family: Arial, sans-serif;

line-height: 1.6;

margin: 0;

padding: 20px;

background-color: #f0f0f0;

color: #333;

}

header {

color: #333;

padding: 20px;

text-align: center;

margin-bottom: 20px;

}

img {

max-width: 100%;

height: auto;

display: block;

margin: 10px 0;

border-radius: 5px;

max-width: 600px;

max-height: 400px;

}

button {

background-color: #FF99FF;

color: white;

border: none;

padding: 10px 20px;

font-size: 16px;

cursor: pointer;

border-radius: 5px;

display: block;

margin: 0 auto;

}

footer {

text-align: center;

margin-top: 20px;

padding: 10px;

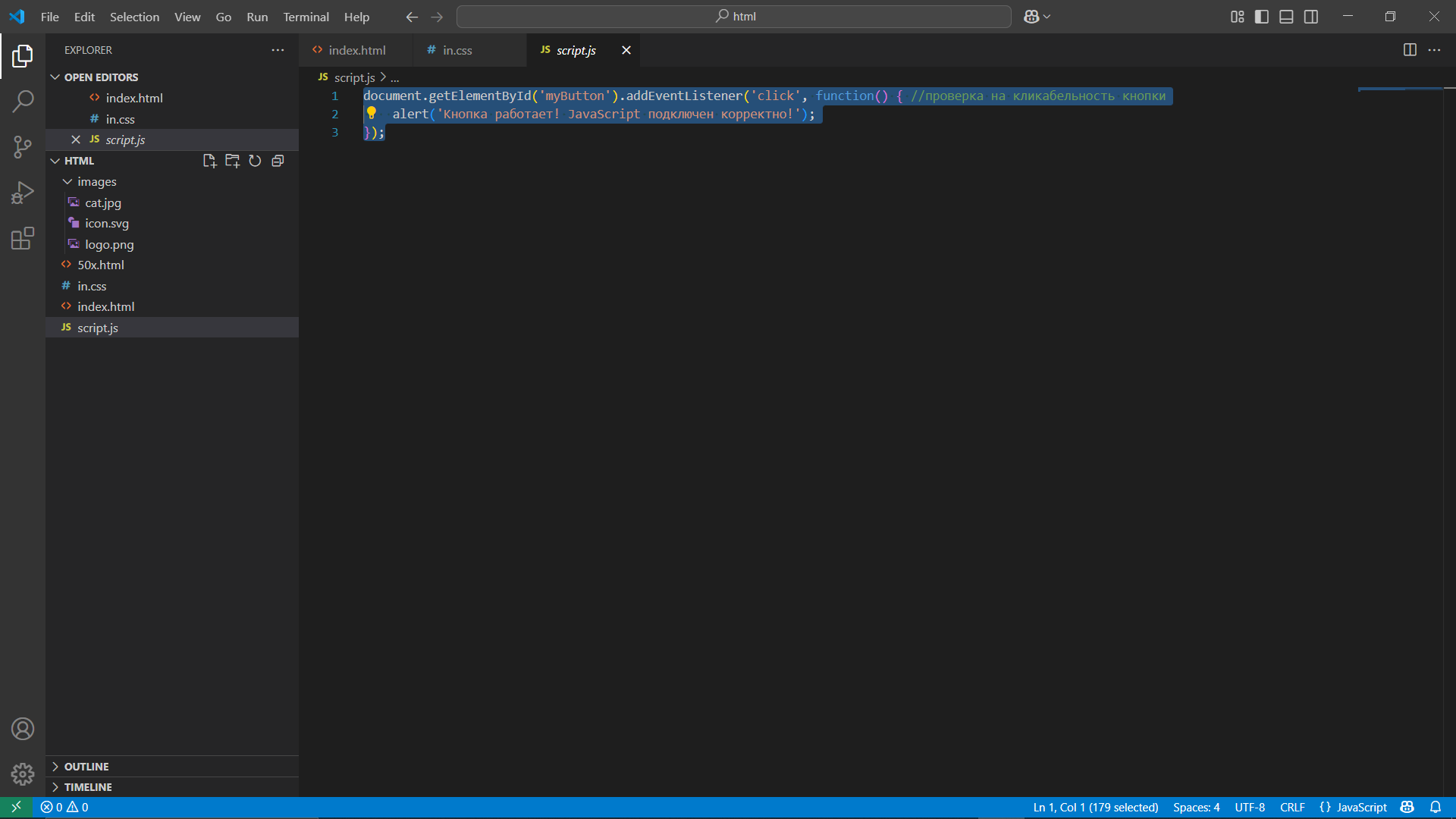
background-color: #ddd;

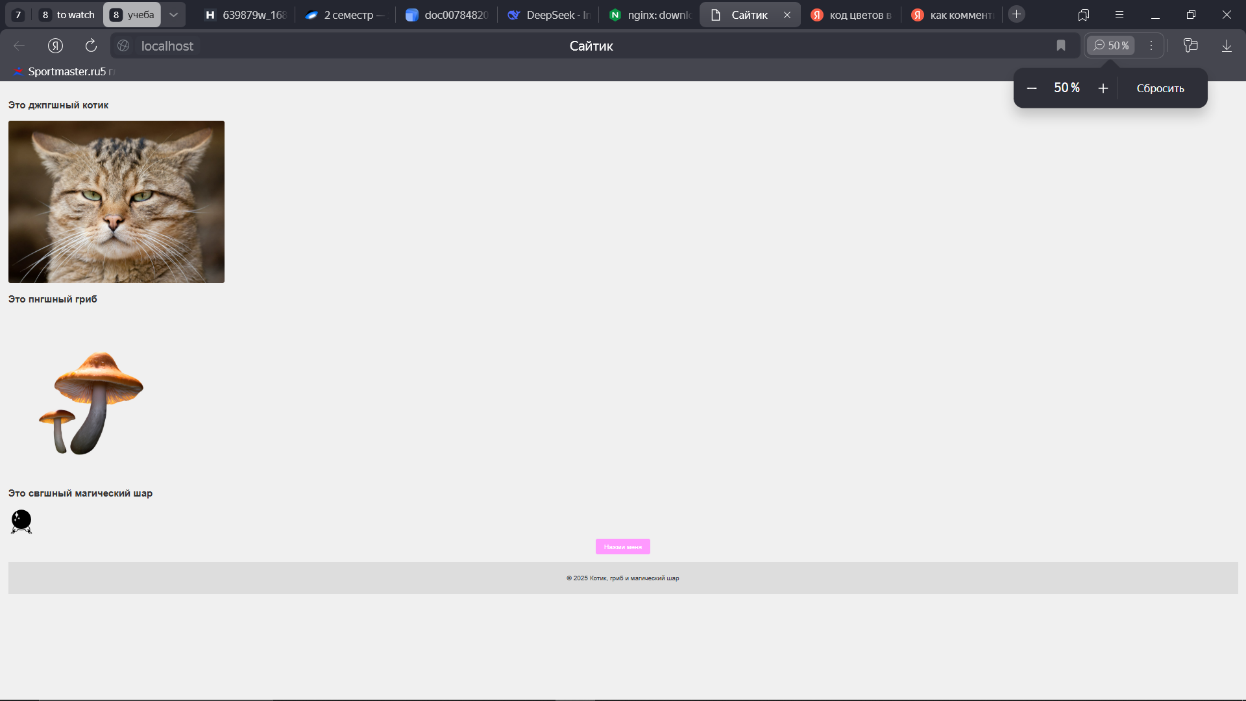
**Script.js:**

document.getElementById('myButton').addEventListener('click', function() { //проверка на кликабельность кнопки

alert('Кнопка работает! JavaScript подключен корректно!');

});

**Структура папки html:**



1. Добавить еще 3 страницы с переходами. Обновить файл конфигурации и перезапустить web-сервер. В файле конфигурации постараться использовать различные директивы.

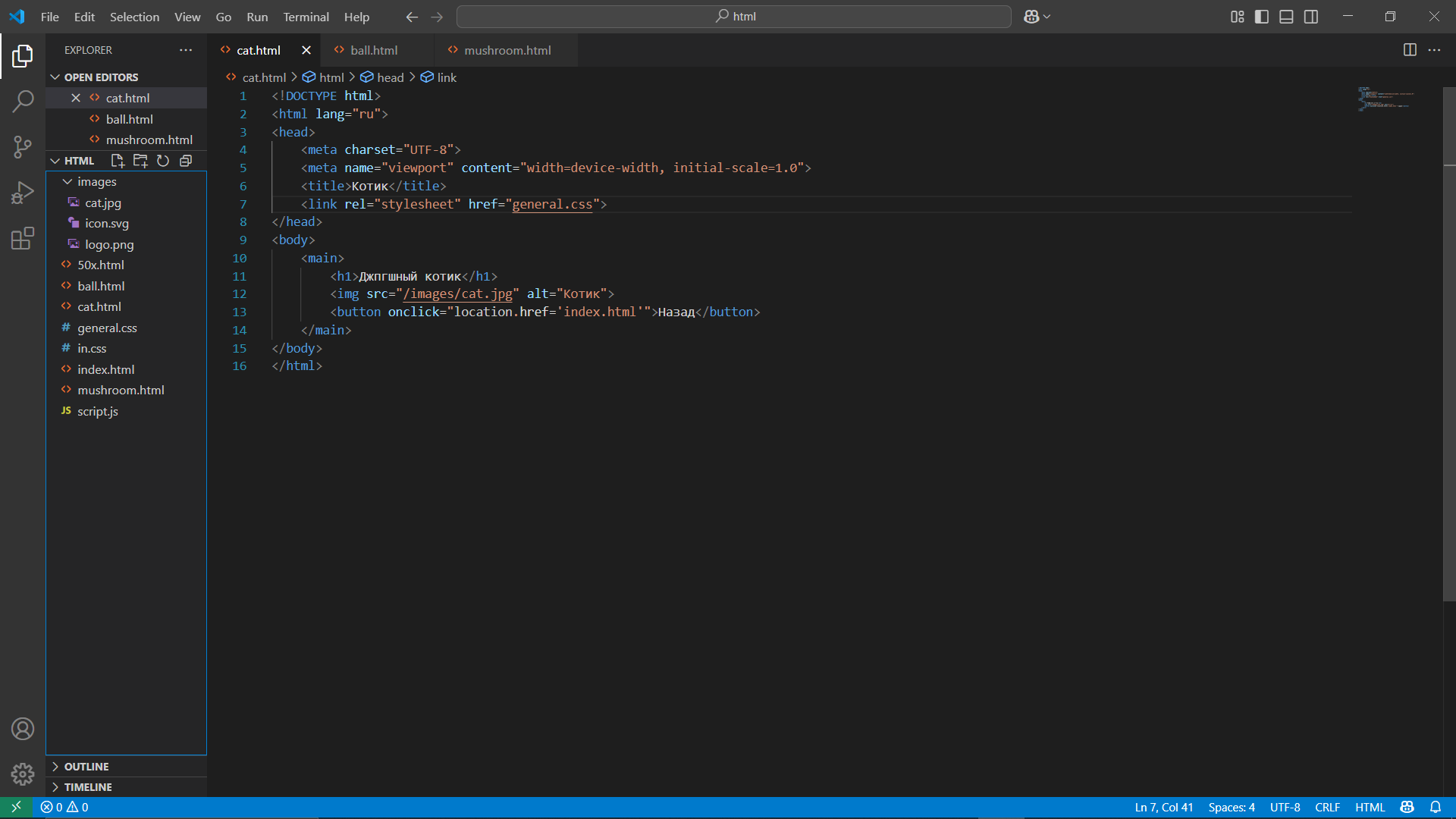
Обязательные дериктивы:

* worker\_process;
* worker\_connections;
* multi\_accept;
* charset;
* open\_file\_cache;
* Настройки буферов;
* Конфигурация timeout;
* add\_header;
* expires;
* gzip;
* limit\_conn\_zone;
* limit\_req\_zone.

Для новых 3-х страниц (только для них) добавить новый файл логирования ошибок. Для одной из страниц отключить логии (для примера).

Для одной из страниц настроить редирект с кодом ответа 301 или 302. В отчете привести основные файлы логов и новый файл логов для 3-х страниц.

**Новая структура папки html**:

**nginx.conf:**

worker\_processes 1;

events {

multi\_accept on;

worker\_connections 1024;

}

http {

charset utf-8;

open\_file\_cache max=1000 inactive=20s;

include mime.types;

default\_type application/octet-stream;

sendfile on;

keepalive\_timeout 65;

server {

listen 80;

server\_name localhost;

# Корневая location

location / {

root html;

index index.html index.htm;

}

**# Редирект /ball → /cat**

location = /ball {

return 301 /cat;

}

**# Блок для URL-пути /cat(логи)**

location /cat {

root C:/nginx/nginx-1.28.0/html;

try\_files $uri /cat.html =404;

access\_log C:/nginx/nginx-1.28.0/logs/cat\_access.log;

error\_log C:/nginx/nginx-1.28.0/logs/cat\_error.log;

}

**# Блок для URL-пути /mushroom(выключены логи)**

location /mushroom {

root C:/nginx/nginx-1.28.0/html;

try\_files $uri /mushroom.html =404;

access\_log off;

error\_log off;

}

# Общие настройки

client\_body\_buffer\_size 10K;

client\_header\_buffer\_size 1k;

client\_max\_body\_size 8m;

large\_client\_header\_buffers 2 1k;

client\_body\_timeout 12;

client\_header\_timeout 12;

send\_timeout 10;

add\_header X-Frame-Options "SAMEORIGIN";

add\_header X-XSS-Protection "1; mode=block";

expires 24h;

gzip on;

gzip\_types text/plain text/css application/json;

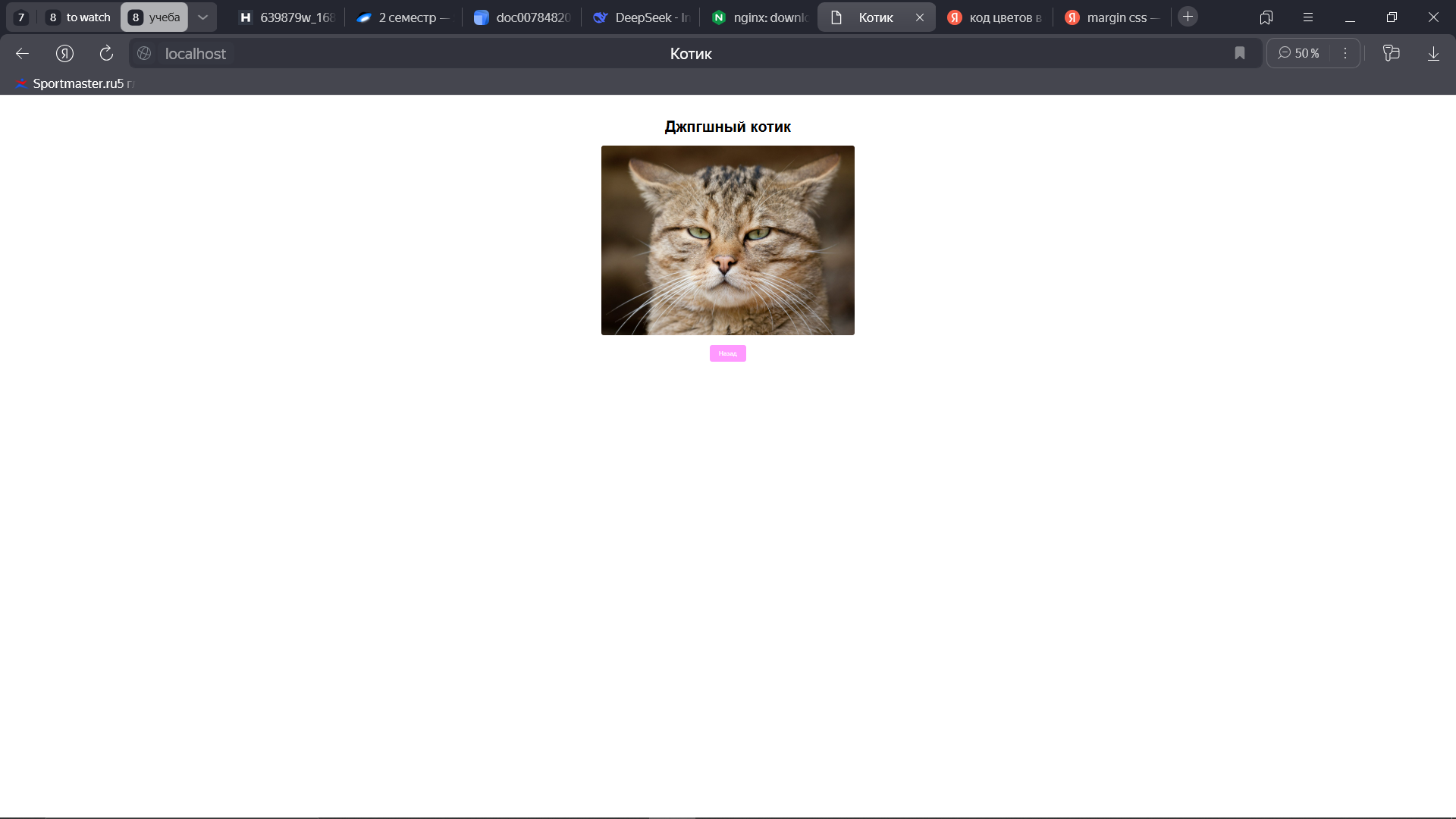
}

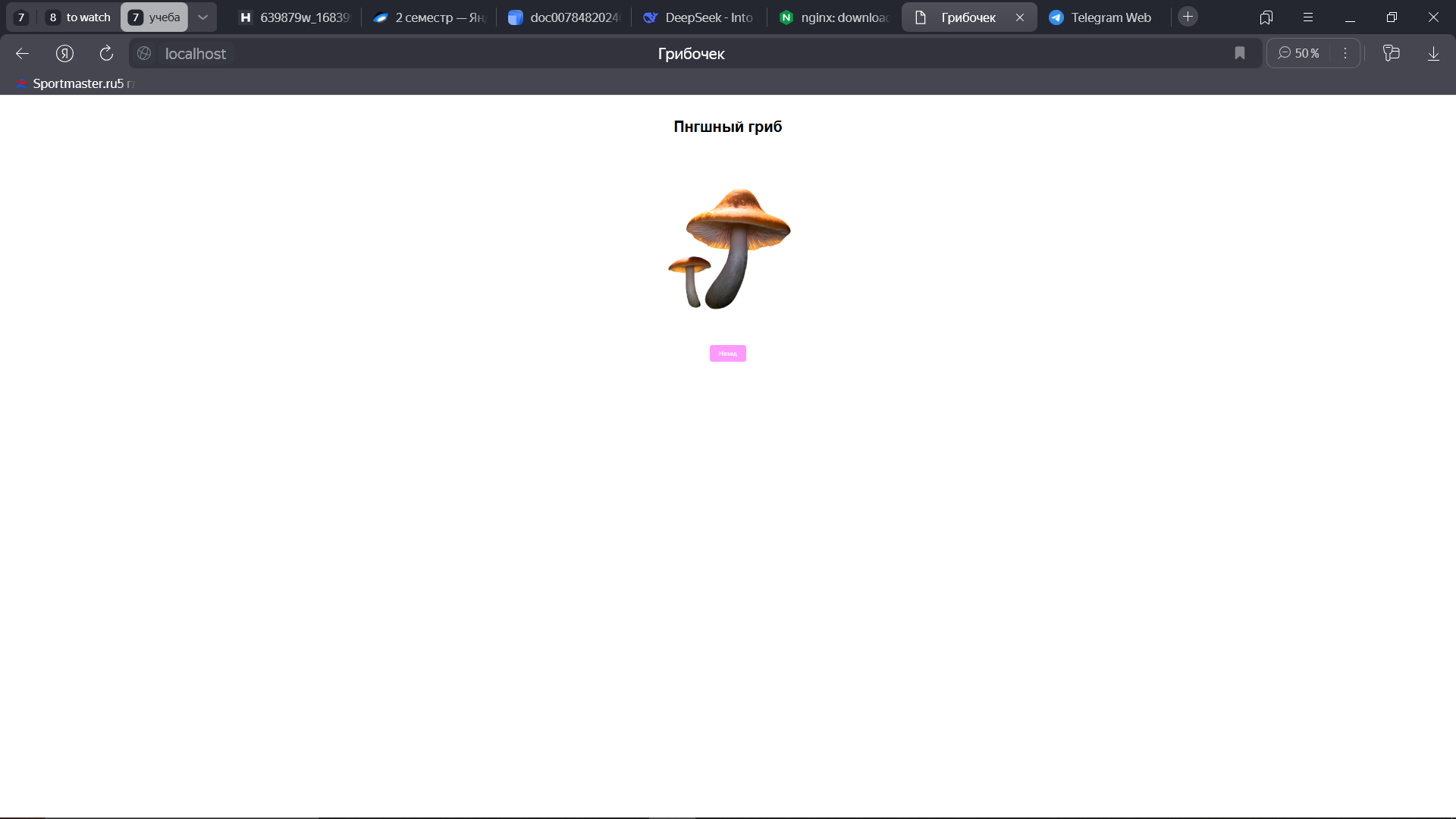
limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

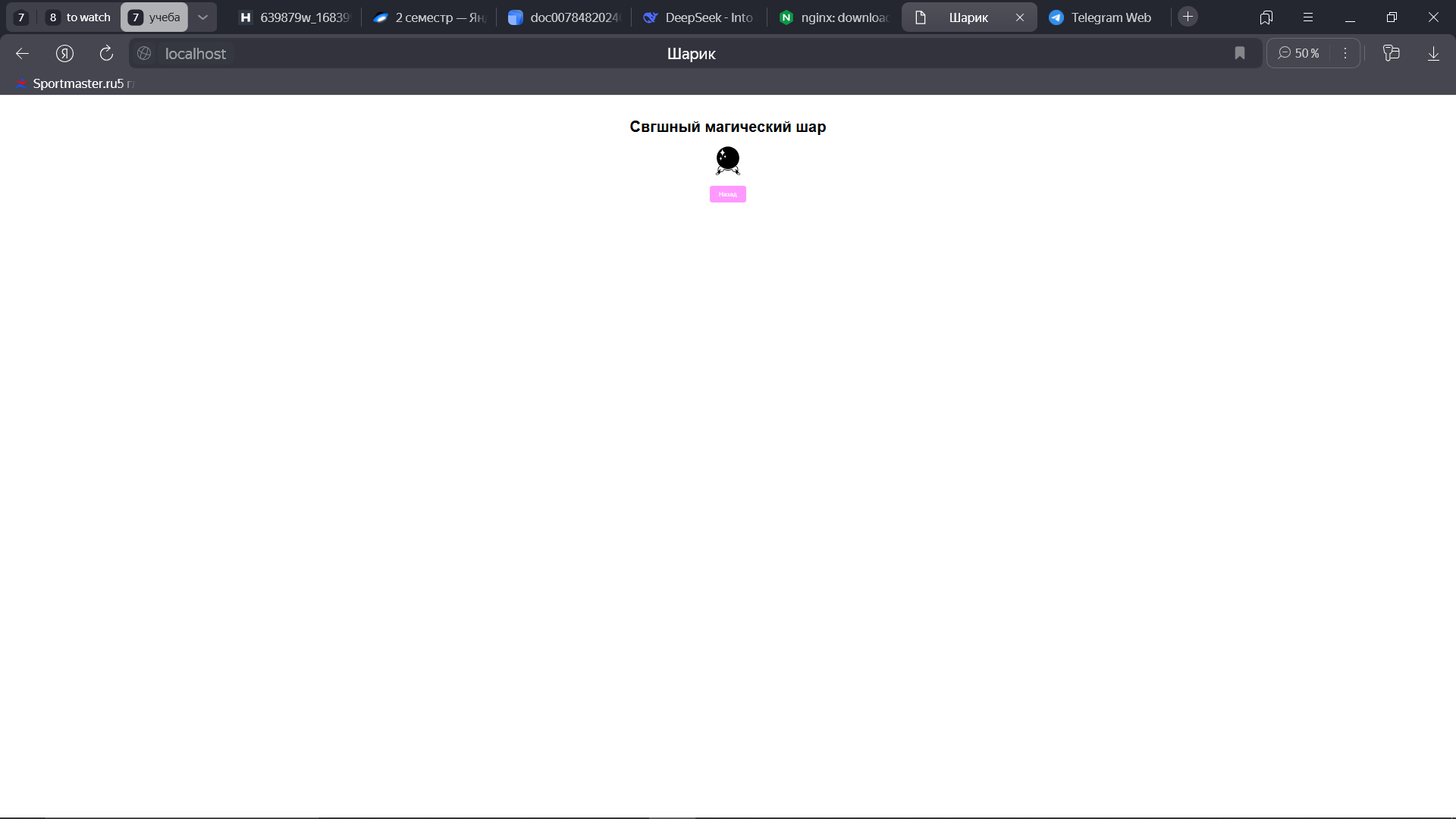
limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s;

}

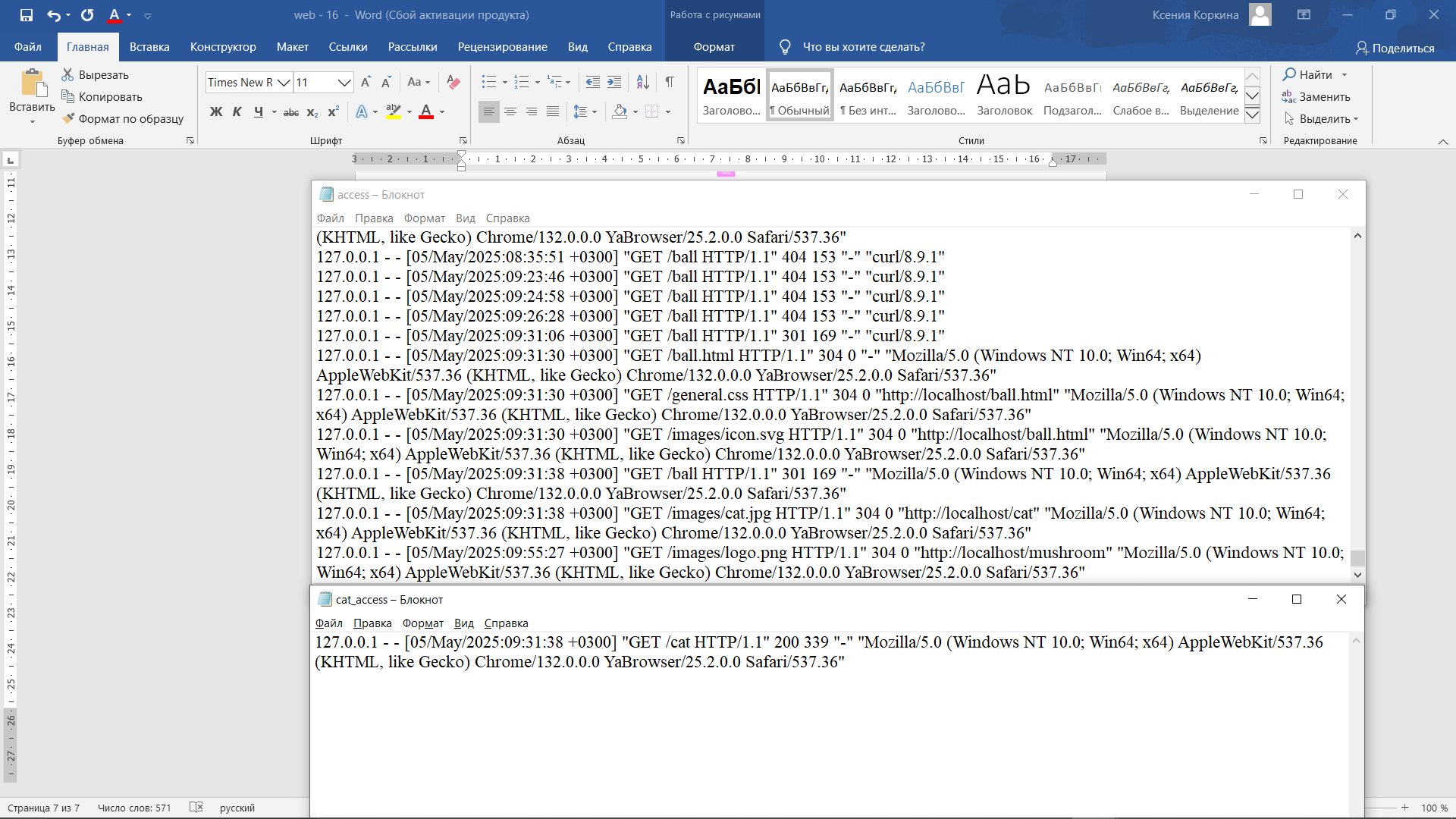
**3 Новые страницы с кнопкой перехода к исходной странице:**







**Логи:**



1. Настроить NGINX на запуск исполняемого файла на backend стороне. В отчете описать процесс настройки и запуска.

В блок server файла nginx.conf добавлена новый **location** для работы с бэкендом:

location /api/ {

proxy\_pass http://127.0.0.1:5000/;

proxy\_set\_header Host $host;

}

все запросы, начинающиеся с **/api/,** будут перенаправлены на бэкенд. **proxy\_pass** - перенаправляет запросы на локальный сервер на порту 5000, **proxy\_set\_header Host** - передаёт оригинальное имя хоста (из запроса клиента) в backend.

После изменения nginx.conf перезагрузить его: **nginx -s reload.**

Для дальнейшей работы требуется запустить любой backend-сервер на порту 5000.

Когда пользователь обращается по адресу http://.../api/..., NGINX перехватит этот запрос и перенаправит запрос на другой сервер, работающий локально на порту 5000 (<http://127.0.0.1:5000/>). Backend-сервер получает этот запрос, обрабатывает его и возвращает ответ. NGINX передаёт результат обратно пользователю.

1. Обеспечить потоковую передачу видео на сайт с помощью модуля mp4. Добавить для доступа к видео требование ввода логина и пароля (на стороне сервера).

В папке html создать новую папку videos b поместить туда видео;

Добавить в конфигурацию nginx блок для обработки видео:

# Потоковое видео с защитой паролем

location /videos/ {

root C:/nginx/nginx-1.28.0/html; # Путь до папки с видео

mp4; # Включение потоковой передачи

# Защита паролем

auth\_basic "Restricted Video Access";

auth\_basic\_user\_file C:/nginx/nginx-1.28.0/conf/.htpasswd;

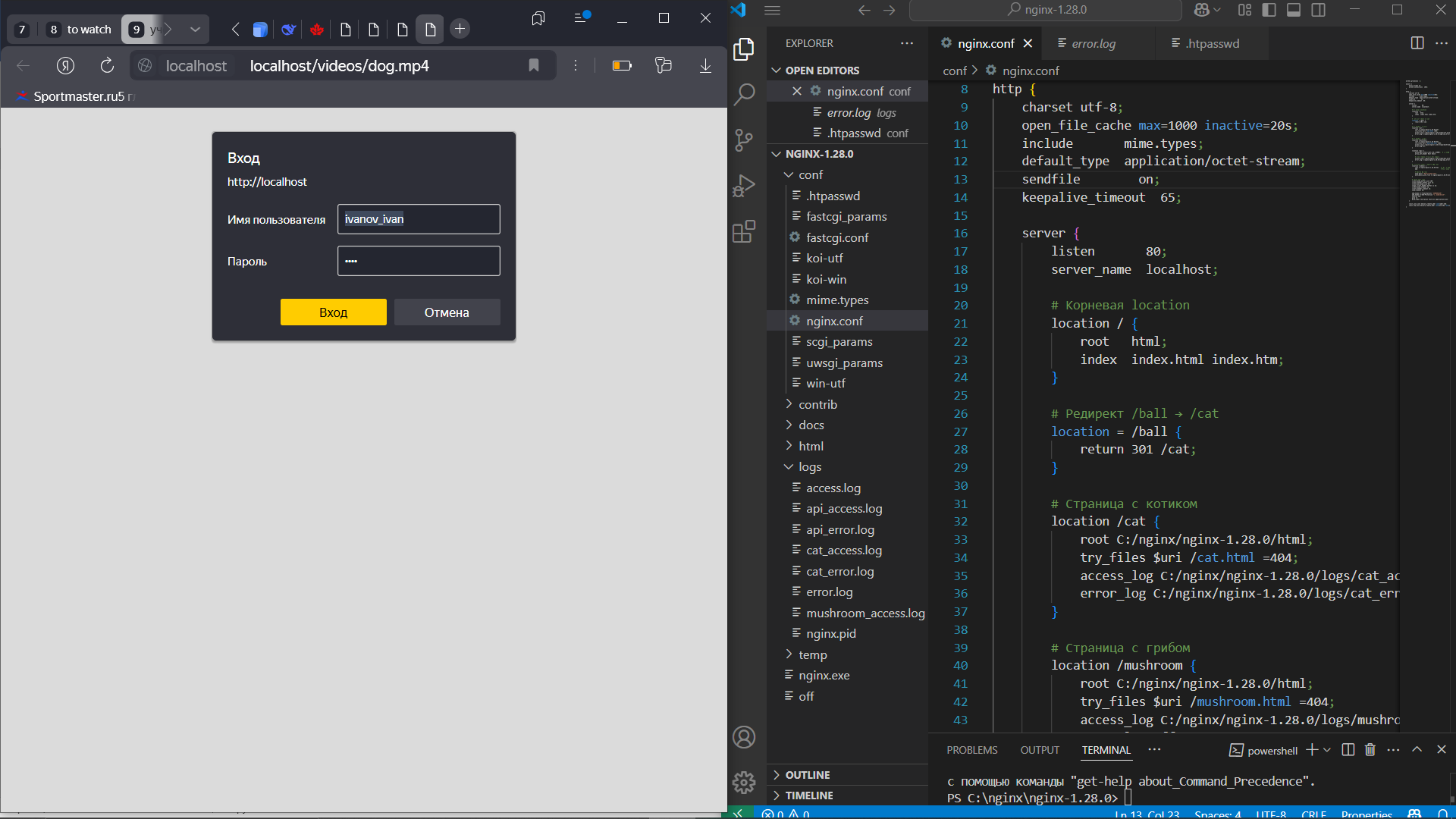
};

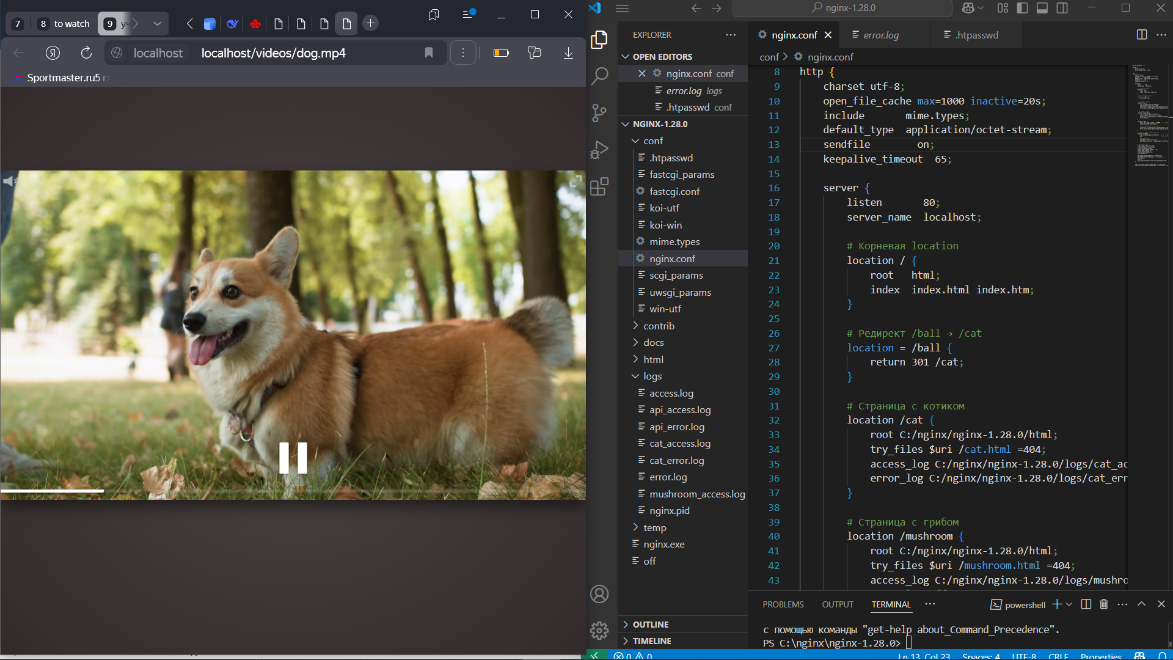
Проверить синтаксис и перезапустить nginx;

Создать файл .htpasswd, содержащий логин и пароль пользователя (использовался сайт для генерации файла: <https://hostingcanada.org/htpasswd-generator/>);

Снова перезапустить nginx;

Открыть видео в браузере: <http://localhost/videos/dog.mp4>





1. В отчете привести файл конфигурации.

# Указывает, сколько рабочих процессов будет запущено Nginx

worker\_processes 1;

# Блок настроек для обработки соединений

events {

multi\_accept on; # Разрешает одному worker'у принимать несколько соединений одновременно

worker\_connections 1024; # Максимальное количество соединений, которые может обработать один worker

}

# Основной блок HTTP-настроек

http {

charset utf-8;

open\_file\_cache max=1000 inactive=20s; # Кэширование информации об открытых файлах (макс 1000 записей, удалять неиспользуемые через 20 сек)

include mime.types; # Подключает файл с MIME-типами (соответствия расширений файлов их типам)

default\_type application/octet-stream; # MIME-тип по умолчанию, если не определён в mime.types (бинарный поток)

sendfile on;

keepalive\_timeout 65;

# Настройки сервера

server {

listen 80; # Порт, на котором сервер будет слушать запросы

server\_name localhost; # Доменное имя сервера (localhost для локального тестирования)

# Корневая location

location / {

root html; # Корневая директория, откуда берутся файлы (относительно папки nginx)

index index.html index.htm; # Файлы, которые будут искаться по умолчанию при запросе к директории

}

# Редирект /ball → /cat

location = /ball {

# Возвращает HTTP 301 (постоянный редирект) на /cat

return 301 /cat;

}

# Блок для URL-пути /cat

location /cat {

# Абсолютный путь к корневой директории для этого location

root C:/nginx/nginx-1.28.0/html;

# попытка найти нужный файл (сначала точный файл - $uri; если нет - отдаст - /cat.html)

try\_files $uri /cat.html =404;

# Лог успешных запросов и лог ошибок

access\_log C:/nginx/nginx-1.28.0/logs/cat\_access.log;

error\_log C:/nginx/nginx-1.28.0/logs/cat\_error.log;

}

# Блок для URL-пути /mushroom

location /mushroom {

root C:/nginx/nginx-1.28.0/html;

try\_files $uri /mushroom.html =404;

access\_log C:/nginx/nginx-1.28.0/logs/mushroom\_access.log;

error\_log off;

}

# Блок для API-запросов (все URL, начинающиеся с /api/)

location /api/ {

proxy\_pass http://127.0.0.1:5000/; # Перенаправление на backend

# Передаёт оригинальный заголовок Host из запроса клиента

proxy\_set\_header Host $host;

# Логирование

access\_log C:/nginx/nginx-1.28.0/logs/api\_access.log;

error\_log C:/nginx/nginx-1.28.0/logs/api\_error.log;

}

# Потоковое видео с паролем

location /videos/ {

root C:/nginx/nginx-1.28.0/html; # Путь до папки с видео

mp4; # Включение потоковой передачи

auth\_basic "Restricted Area"; #базовая HTTP-ауентификация

# Файл с логинами и паролями (в зашифрованном виде)

auth\_basic\_user\_file C:/nginx/nginx-1.28.0/conf/.htpasswd;

}

# Общие настройки

client\_body\_buffer\_size 10K; # Размер буфера для тела запроса

client\_header\_buffer\_size 1k; # Размер буфера для заголовков запроса

client\_max\_body\_size 8m; # Максимальный размер тела запроса (8 мегабайт)

large\_client\_header\_buffers 2 1k; # Буферы для больших заголовков

client\_body\_timeout 12; # Таймаут чтения тела запроса

client\_header\_timeout 12; # Таймаут чтения заголовков

send\_timeout 10; # Таймаут передачи ответа

# Заголовки безопасности

add\_header X-Frame-Options "SAMEORIGIN"; # Запрещает встраивание в iframe других сайтов

add\_header X-XSS-Protection "1; mode=block"; # Включает защиту от XSS-атак

expires 24h; # Время кэширования по умолчанию

gzip on; # Включает сжатие gzip

gzip\_types text/plain text/css application/json; # Типы файлов для сжатия

}

# Настройки ограничений (защита от DDoS)

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m; # Ограничение количества соединений с одного IP

limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s; # Ограничение частоты запросов (1 в секунду)

}

