

Работа с файлами

Теоретическая часть

- Загрузка файлов из браузера
- MIME типы
- Хранение файлов в облаке S3
- Проксирование, кеширование и контроль доступа к файлам в nginx

Практическая часть

- Загрузка и сохранение файлов в S3
- Раздача загруженных файлов и ограничение доступа

Загрузка
файлов из
браузера

Файлы

- Файлы прописанные в коде и в шаблонах, назовем их STATIC файлы (то, что добавляет разработчик);
- Файлы, который используются в коде, но известны только в процессе работы кода, назовем их MEDIA файлы (то, что добавляет пользователь);
- Пользователь загружает файл на страничке, frontend берёт содержимое, кодирует при помощи base64 и отправляет на бекэнд. Наше дело --- раскодировать и сохранить.

MIME типы

MIME типы

MIME (Multipurpose Internet Mail Extension, Многоцелевые расширения почты Интернета) — спецификация для передачи по сети файлов различного типа: изображений, музыки, текстов, видео, архивов и др.

Основные MIME типы

- application (внутренний формат прикладной программы).
Примеры: json, xml, js, pdf.
- image (изображения). Примеры: jpeg, png, webp, gif.
- text (текст). Примеры: csv, markdown, plain, html, xml.
- multipart. Примеры: form-data, signed, encrypted.
- audio (аудио). Примеры: mpeg, aac, ogg.

Для чего нужно?

- Загружать/отображать файлы определенного типа (вкладки фото/видео/файлы в ВК, Телеграме и т.д.).

Установить либу для определения mime типа.

```
pip3 install python-magic
```

```
def check_in_memory_mime(in_memory_file):  
    mime = magic.from_buffer(in_memory_file.read(), mime=True)  
    return mime
```

Хранение
файлов на
локальном

хранилище и в
облаке S3

Облачное хранилище Mail.Ru S3

S3 (Simple Storage Service) — онлайновая веб-служба, предоставляющая возможность для хранения и получения любого объёма данных, в любое время из любой точки сети, так называемый файловый хостинг. S3 API — набор команд, которые «понимает» хранилище и выполняет в ответ некие действия (получение/запись файла).

Достоинства применения S3

- Высокая масштабируемость;
- Надёжность;
- Высокая скорость;
- SSL-соединение с хранилищем;
- Гарантирована на уровне 500 запросов в секунду;
- Недорогая инфраструктура хранения данных.

Виды хранилища Mail.Ru Cloud Solutions

- Icebox (для хранения редко используемых данных: архивов, резервных копий, журналов);
- Hotbox (для хранения часто используемых данных);

Flask и Mail.Ru Cloud Solutions S3

Boto это набор средств разработки (SDK) от Amazon Web Services (AWS) для языка Python, позволяющая разработчикам писать программы для сервисов S3 или EC2 (Elastic Compute Cloud).

- Прислать на почту a.kukhtichev@corp.mail.ru запрос для получения доступа с пометкой в теме [ТТ];
- Дождаться ответного письма с ссылкой-приглашением на [Mail.Ru Cloud Solutions](#);
- Подтвердить адрес электронной почты;

Как правильно назвать бакет?

Адрес: <https://mcs.mail.ru/>

- Т.к. аккаунт "коммунальный", необходимо соблюдать правила именования: `year-full_name-something`,
например `2018-anton-kukhtichev-bucket` ;

Boto3

Используем boto3 библиотеку

Установить библиотеку для работы с S3.

```
pip3 install boto3
```

```
import boto3
```

```
session = boto3.session.Session()
```

```
s3_client = session.client( ... )
```


Boto3

Аргументы `session.client(...)`:

- `service_name='s3'`
- `endpoint_url='http://hb.bizmrg.com'`
- `aws_access_key_id='<your access key id>';`
- `aws_secret_access_key='<your secret key>'`

Put/Get объект из S3

Успешно инициализировали s3_client.

```
s3_client.put_object( Bucket='<bucket name>', Key='<key>', Body=<content> )
```

- Одинаковые ключи по умолчанию перезаписываются;
- **Решение 1:** ключ --- хэш от названия файла и атрибутов пользователя;
- **Решение 2:** в Key можно указывать путь, например:
Key='dir1/key.txt';
- В Body можно передать BufferedReader или содержимое файла (наш вариант).

Проксирование,
кеширование и
контроль

доступа к
файлам в Nginx

Nginx

- Веб-сервер, работающий на Unix-подобных операционных системах;
- Nginx может работать при очень больших нагрузках;
- Игорь Сисоев начал разработку в 2002 году;
- Осенью 2004 года вышел первый публично доступный релиз.

установить nginx

```
sudo apt install nginx
```

редактируем конфиг

```
sudo vim /etc/nginx/conf.d/default.conf
```

Проксирование в Nginx

```
location / {  
    ...  
    proxy_pass http://127.0.0.1:5000/;  
    ...  
}
```

Кеширование в Nginx

- Не генерировать постоянно одни и те же скрипты;
- Приложение генерирует страницу один раз, и результат сохраняется в память;
- Периодически (time to live --- ttl) сохраненная версия будет удаляться и генерироваться новая;
- Ускорение сайта и экономию ресурсов;
- Ключи в оперативной памяти, кеш хранится на жёстком диске.

Кеширование в Nginx

```
http {  
    ...  
    proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=all:32m max_size=1g;  
    ...  
}
```

```
mkdir /var/cache/nginx  
chown nginx:nginx /var/cache/nginx/
```

Кеширование в Nginx

```
server {  
    ...  
    proxy_cache all;  
    # Кешировать указанные коды ответов 5 минут  
    proxy_cache_valid 200 301 302 304 10m;  
    # Кешировать ошибки 1 минуту  
    proxy_cache_valid 404 502 503 1m;  
}
```


Контроль доступа файлов Nginx

- Пользователь каким-то образом получил урл или название файла;
- Все запросы на скачивание файлов передаются скрипту, который решает, как поступить
 - отправить пользователю какой-либо файл;
 - показать страницу access denied.

Контроль доступа файлов Nginx.

Шаг 1

1) Новый location с internal;

```
location /s3/ {  
    ...  
    internal;  
    ...  
}
```

Контроль доступа файлов Nginx.

Шаг 2

- Создание метода, отвечающего за проверку доступа;
- Если пользователь заслуживает доступа к этому файлу, то проставить заголовок X-Accel-Redirect

Контроль доступа файлов Nginx.

Шаг 3

Сгенерировать подпись и ключ для доступа в s3 хранилище на nginx.

```
location /protected/ {
    set $s3_bucket 'upload_file';
    set $aws_access_key '<access key>';
    set $aws_secret_key '<secret key>';
    set $url_full      "$1";
    set $string_to_sign "$request_method\n\n\n\nx-amz-date:${now}\n/$bucket/$url_full";
    set_hmac_sha1      $aws_signature $aws_secret $string_to_sign;
    set_encode_base64  $aws_signature $aws_signature;

    proxy_http_version 1.1;
    proxy_set_header    Connection "";
    proxy_set_header    authorization "AWS $aws_access:$aws_signature";
    proxy_set_header    Host "https://${s3_bucket}.hb.bizmrg.com";
```

Контроль доступа файлов Nginx.

Шаг 3

На самом деле можно все подписи и ключи сгенерировать во вьюшке.

Материалы

- Облачная платформа: [Mail.Ru Cloud Souldtions](#)
- Nginx: [Официальный сайт](#)
- Как авторизоваться MRCS: [Создание подписей для запросов REST и их аутентификация](#)

Домашнее задание

- Реализовать метод API для загрузки файла (использовать base64) (3 балла)
- Использовать для хранения файла облачное S3 хранилище (3 балла)
- Создать location в Nginx для раздачи загруженных файлов (3 балла)
- Реализовать обработчик в приложении для проверки прав доступа к файлу (2 балла)

Срок сдачи: следующее занятие.

Спасибо за внимание!

