Работа с файлами

Теоретическая часть

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

Практическая часть

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

Загрузка файлов из браузера

Файлы в Django

- 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.

Файлы в модели I

- Обработчики загрузки определены в настройке FILE_UPLOAD_HANDLERS
- MemoryFileUploadHandler и TemporaryFileUploadHandler определяются обработку файлов по умолчанию в Django, загрузка небольших файлов в память и больших на диск.

Файлы в модели II

- Использовать внутри модели поле models.FileField();
- Можно указать парамтер upload_to --- папка, куда Django будет складывать файлы;
- Выбор места для хранения файлов задаётся в настройках DEFAULT_FILE_STORAGE;
- Можно для разных полей переопределить разные хранилища при помощи параметра storage.

Например

```
class CatFile( models.Model ):
    file_mime = models.CharField( max_length=72 )
    file_key = models.CharField( max_length=72 )
    file_content = models.FileField( null=True, upload_to="files" )

class Cat( models.Model):
    name = models.CharField( max_length=72 )
    age = models.IntegerField( default=0 )
    photo = models.FileField( null=True, storage=fs, upload_to="photos" )
    files = models.ManyToManyField( CatFile )
```

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);

Django 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).

- Зарегистрироваться на Mail.Ru Cloud Solutions;
- Подтвердить адрес электронной почты;
- Создать аккаунт для Cloud Storage (Облачное хранилище) и получить Access Key ID, Secret Key;
- Создать бакет.

Boto3

Используем boto3 библиотеку

```
pip3 install boto3            # установить библиотеку для работы с S3
pip3 install django-storages
```

```
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 аттрибутов пользователя;
- **Решение 2:** в Key можно указывать путь, например: Key='dir1/key.txt';
- В Body можно передать BufferedReader или содержимое файла (наш вариант).

Генерация URL

- Брать содержимое файла --- лишний траффик. URL на объект лёгкий;
- **Решение:** использовать generate_presigned_url, возвращающую валидный урл на файл.

```
url = s3_client.generate_presigned_url('get_object',
    Params={
        'Bucket': '<bucket name>',
        'Key': '<key>',
    },
    ExpiresIn=3600)
```

А можно по-другому? Можно!

- Можно настроить без вызова клиента s3;
- Нужно в settings.py переопределить DEFAULT_FILE_STORAGE и определить AWS_S3_ENDPOINT_URL, AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, AWS_STORAGE_BUCKET_NAME.

```
# Использование S3 хранилища
DEFAULT_FILE_STORAGE = 'storages.backends.s3boto3.S3Boto3Storage'
# Использование локального хранилища
DEFAULT_FILE_STORAGE = 'django.core.files.storage.FileSystemStorage'
```

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.).

```
pip3 install python-magic  # установить либу для определения mime типа

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

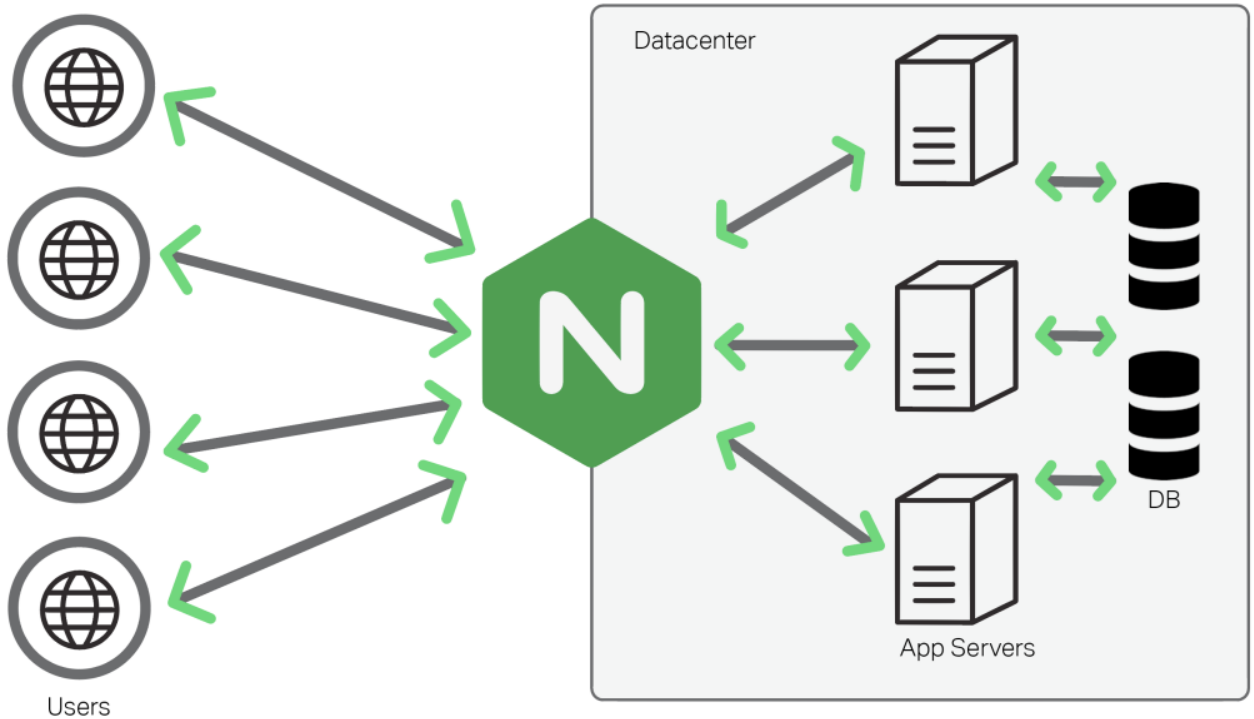
Проксирование, кеширование и контроль доступа к файлам в Nginx

Nginx

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

```
sudo apt install nginx                    # установить nginx
sudo vim /etc/nginx/conf.d/default.conf # редактируем конфиг
```

Nginx



Проксирование в Nginx

```
location / {
    ...
    proxy_pass http://127.0.0.1:8000/;
    ...
}
```

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.

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, или показать стриницу access denied.

Контроль доступа файлов Nginx

- Новый location с internal;
- Использование заголовка X-Accel-Redirect и X-Accel-Expires в ответах от сриптов backend-сервера.

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

Контроль доступа файлов 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";
```

Материалы

- Облачная платформа: Mail.Ru Cloud Soultions
- API django-storages: Amazon S3
- 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 localtion 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!