

MB-03-TS01 Техническая спецификация — MoodBoss API — Direct Upload в Cloudflare R2

1. Основание и цель

Реализовать механизм безопасной загрузки файлов клиентами напрямую в **Cloudflare R2** по модели **Direct Upload (pre-signed PUT URL)** с:

- генерацией подписанных ссылок сервером,
- подтверждением загрузки (confirm endpoint),
- валидацией (size, mime, SHA256),
- логированием этапов,
- запуском асинхронной обработки,
- базовой lifecycle-логикой хранения,
- сохранением метаданных в БД.

Вся серверная логика реализуется в контейнере **calculations** (Django-приложение MoodBoss).

2. Архитектура решения

2.1 Общая схема

1. Клиент → POST /files/presign
2. Сервер (calculations) → генерирует pre-signed PUT URL для R2
3. Клиент → загружает файл напрямую в R2
4. Клиент → POST /files/confirm
5. Сервер:
 - делает HEAD к R2
 - проверяет size / mime / checksum
 - переводит статус
 - ставит задачу в очередь (если используется Celery)
6. Асинхронная обработка
7. Обновление статуса

3. Cloudflare R2 — требования

Используется S3-совместимый API R2:

- Endpoint:
https://<ACCOUNT_ID>.r2.cloudflarestorage.com
- Подпись: AWS Signature v4
- Метод загрузки: PUT
- Bucket: настраивается через переменные окружения

3.1 Переменные окружения (только через .env, не в Dockerfile)

В контейнере calculations должны использоваться:

R2_ACCOUNT_ID=
R2_ACCESS_KEY_ID=
R2_SECRET_ACCESS_KEY=
R2_BUCKET_NAME=
R2_ENDPOINT=https://<ACCOUNT_ID>.r2.cloudflarestorage.com
R2_PRESIGN_TTL=900
R2_MAX_FILE_SIZE=10485760
R2_ALLOWED_MIME=image/jpeg,image/png,application/pdf

Запрещено хранить ключи в коде или Dockerfile.

4. Функциональный объём

4.1 Endpoint генерации pre-signed URL

POST

/calculations/api/v2/files/presign

Вход:

```
{  
  "mime": "image/jpeg",  
  "size": 245678,  
  "checksum_sha256": "hex_string",  
  "profile_id": 426  
}
```

Логика:

1. Проверка авторизации (JWT).
2. Проверка допустимого mime.
3. Проверка $size \leq R2_MAX_FILE_SIZE$.
4. Генерация:
 - file_id (UUID)
 - object_key (например: uploads/{user_id}/{file_id})
5. Генерация pre-signed PUT URL.
6. Создание записи в БД со статусом presigned.

Ответ:

```
{  
  "file_id": "uuid",  
  "object_key": "uploads/123/uuid",  
  "upload_url": "https://...",  
  "expires_in": 900  
}
```


4.2 Confirm endpoint

POST

/calculations/api/v2/files/confirm

Вход:

```
{
  "file_id": "uuid",
  "object_key": "uploads/123/uuid",
  "size": 245678,
  "mime": "image/jpeg",
  "checksum_sha256": "hex_string"
}
```

Логика:

1. Проверка, что file_id принадлежит пользователю.
2. Проверка текущего статуса.
3. HEAD-запрос к R2:
 - проверка фактического размера.
4. Проверка mime.
5. Проверка checksum:
 - либо через метаданные объекта,
 - либо через серверную обработку (если требуется).
6. Обновление статуса → confirmed.
7. Постановка задачи в очередь (Celery).
8. Логирование этапов.

Ответ:

```
{
  "file_id": "uuid",
  "status": "confirmed"
}
```

5. Проверки

5.1 Размер

- превышение лимита → 400 file_size_exceeded

5.2 MIME

- неразрешённый тип → 400 invalid_mime_type

5.3 SHA256

- mismatch → 422 checksum_mismatch

6. Логирование (обязательно)

В логах должны фиксироваться события:

- presign_issued
- confirm_requested
- validation_passed
- validation_failed
- async_processing_enqueued
- processing_started
- processing_completed
- processing_failed

Каждая запись содержит:

- file_id
- user_id
- profile_id
- timestamp
- error_code (если есть)

7. Модель БД

Таблица, например: UploadedFileModel

Обязательные поля:

- id (UUID)
- user_id
- profile_id (nullable)
- object_key
- bucket
- checksum_sha256
- size
- mime
- status
- error_code
- error_message
- created_at
- confirmed_at
- processed_at

8. Статусы

- created
- presigned
- confirmed
- processing
- processed
- failed
- expired

9. Асинхронная обработка

После confirmed:

- задача ставится в Celery (если Celery используется в calculations),
- задача скачивает объект из R2,
- выполняет необходимую бизнес-обработку,
- обновляет статус.

10. Lifecycle-логика

В рамках контейнера calculations реализуется:

- очистка записей в статусе presigned, если confirm не был вызван в течение N часов;
- возможность удаления объекта из R2 при статусе failed (опционально);
- CRON / periodic task (Celery beat) для проверки просроченных записей.

11. Безопасность

- JWT-авторизация обязательна.
- Запрещено подтверждение чужого file_id.
- Pre-signed URL имеет TTL.
- object_key генерируется сервером, клиент не может его менять произвольно.

12. Критерии приёмки

Работа считается выполненной, если:

1. Pre-signed URL корректно загружает файл в Cloudflare R2.
2. Confirm endpoint валидирует:
 - размер,
 - mime,
 - checksum.
3. При успешной загрузке запускается асинхронная обработка.
4. Статусы корректно переходят по цепочке.
5. В логах видна полная трассировка по file_id.
6. Все ключи R2 берутся только из .env.
7. Решение не содержит жёсткой привязки к AWS — используется S3-совместимый интерфейс R2.

13. Подписи сторон

Исполнитель

ООО Вистади

Директор:  Дильдин В.С

Заказчик

ONERY OVERSEAS LIMITED

Директор: _____ Булициду А.