



Backend API Documentation

Biometric Auth System

Base URL (local, Docker, HTTPS):

`https://localhost:8000/api/v1`

Все запросы и ответы — `JSON`, кроме загрузки файлов (`multipart/form-data`).



Аутентификация (JWT)



Access Token

- Используется для всех защищённых запросов
- Передаётся в заголовке:

`Authorization: Bearer <access_token>`



Refresh Token

- Хранится в `httpOnly` cookie
 - Используется для обновления access token
-



Users / Auth

1 Регистрация пользователя

`POST /auth/register`

Описание

Создаёт пользователя, хеширует пароль и PIN.

Request (JSON)

```
{  
  "email": "user@example.com",  
  "password": "StrongPassword123",  
  "pin": "1234"  
}
```

Backend логика

- `password` → bcrypt
- `pin` → bcrypt
- `pin_attempts = 0`

Response 201

```
{  
  "id": 1,  
  "email": "user@example.com"  
}
```

2 Логин по PIN (fallback)

POST /auth/login/pin

Описание

Используется, если Face ID не прошёл.

Request (JSON)

```
{  
  "user_id": 1,  
  "pin": "1234"  
}
```

Логика безопасности

- max 5 попыток

- блокировка на 15 минут
- счётчик сбрасывается при успехе

Response 200 (успех)

```
{  
  "success": true,  
  "access_token": "<JWT>"  
}
```

Response 403 (заблокирован)

```
{  
  "detail": "PIN temporarily locked"  
}
```

Response 401 (неверный PIN)

```
{  
  "detail": "Invalid PIN"  
}
```

3 Refresh access token

POST /auth/refresh

Описание

Обновляет access token через refresh token cookie.

Response

```
{  
  "access_token": "<new JWT>"  
}
```



4 Face Enrollment (регистрация лица)

POST /biometrics/face/enroll

Описание

Сохраняет embedding лица пользователя.

Request

`multipart/form-data`

| Field | Type | Description |
|---------|-------|-----------------|
| user_id | int | ID пользователя |
| file | image | Фото лица |

Backend

- InsightFace
- embedding → AES encryption
- фото **НЕ сохраняется**

Response

```
{  
    "message": "Face enrolled",  
    "biometric_id": 1  
}
```

5 Face Verification (login)

POST /biometrics/face/verify

Описание

Проверяет лицо и выполняет liveness (поворот головы).

Request

`multipart/form-data`

| Field | Type |
|-------|------|
|-------|------|

| | |
|--------|-------|
| user_i | int |
| d | |
| file | image |

Проверки

- cosine similarity ≥ 0.6
- head rotation (yaw)

Response (успех)

```
{  
  "verified": true,  
  "similarity": 0.78,  
  "rotation_detected": true,  
  "liveness_pass": true,  
  "access_token": "<JWT>"  
}
```

Response (неуспех)

```
{  
  "verified": false,  
  "similarity": 0.42,  
  "liveness_pass": false  
}
```

👉 Frontend должен переключиться на PIN fallback



Security & Rate Limit

PIN Rate Limit

- pin_attempts

- `pin_locked_until`

| Попытки | Результат |
|---------|-----------|
|---------|-----------|

| | |
|----------|---------------|
| < 5 | проверка |
| ≥ 5 | блок 15 минут |

ФОТО и приватность

- Фото не сохраняются
 - Обрабатываются в памяти (`cv2.imread`)
 - GDPR-safe
-



HTTPS (local, Docker)

Backend работает по HTTPS:

<https://localhost:8000>

Self-signed certificate
(браузер покажет warning — это нормально)



Error Codes Summary

| Code | Meaning |
|------|---------|
|------|---------|

| | |
|-----|--------------------|
| 400 | Invalid input |
| 401 | Unauthorized |
| 403 | Locked / Forbidden |
| 404 | Not found |



Frontend Flow (важно)

Login Flow

Camera → Face Verify

↓ fail

PIN Login

Registration Flow

Register → Face Enroll → Done



Docker Notes

- Backend внутри Docker
- Frontend обращается к:

<https://localhost:8000>

- CORS должен разрешать frontend origin
-



MVP Scope (что реализовано)

- ✓ Face ID
- ✓ Liveness (rotation)
- ✓ PIN fallback
- ✓ Rate-limit
- ✓ JWT
- ✓ HTTPS
- ✓ Docker
- ✓ Privacy-safe (no raw images)

