



SwipeLab

## API Contract

This document describes the API calls from backend, with detailed requests and responses.

---

### Contents

API Contract .....	1
Authentication Endpoints .....	2
1.1 Registration .....	3
1.2 Login .....	5
1.3 Logout (Token Revocation) .....	6
1.4 Refresh Token .....	6
1.5 Current User .....	7
1.6 Forgot Password .....	7
1.7 Reset Password .....	7
1.8 Email Verification .....	8
1.8.1 Send Verification To Email .....	8
1.8.2 Verify Email (Confirm) .....	8
Classification Endpoints .....	9
2.1 Get Next Batch .....	10
2.2 Submit Classification .....	10
2.3 Progress .....	11
2.4 User Statistics .....	11

Authentication Endpoints

Auth API Summary (SwipeLab)

	Purpose	Endpoint
1.1.1	Register	POST /auth/register
1.1.2	Register (Google)	POST /auth/register/google
1.2.1	Login	POST /auth/login
1.2.2	Login (Google)	POST /auth/login/google
1.3	Logout	POST /auth/logout
1.4	Refresh token	POST /auth/refresh
1.5	Current user	GET /auth/me
1.6	Forgot password	POST /auth/password/forgot
1.7	Reset password	POST /auth/password/reset
1.8	Email verification	POST /auth/email/*

## 1.1 Registration

### 1.1.1 Via Username & Password:

POST /api/v1/auth/register

Request:

```
{
  "username": "john_doe",
  "email": "user@example.com",
  "password": "P@ssw0rd!"
}
```

Response:

```
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresIn": 2592000,
  "user": {
    "id": 1,
    "email": "user@example.com",
    "username": "john_doe",
    "role": "USER"
  }
}
```

Commented [פ1]: Auto-login after successful registration (very common UX).

### 1.1.2 Via Google:

POST /api/v1/auth/register/google

Request:

```
{
  "googleToken": "eyJhbGciOiJSUzI1NiIsImtpZCI6..."
}
```

Response:

```
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresIn": 2592000,
  "user": {
    "id": 1,
    "email": "user@example.com",
    "username": "john_doe",
    "role": "USER"
  }
}
```

Commented [פ2]: Auto-login after successful registration (very common UX).

#### Error Responses:

- **Username already exists**

```
{
  "errorCode": "USERNAME_ALREADY_EXISTS",
  "message": "Username is already taken"
}
```

- **Email already exists**

```
{
  "errorCode": "EMAIL_ALREADY_EXISTS",
  "message": "Email is already registered"
}
```

- **Email not allowed**

```
{
  "errorCode": "EMAIL_NOT_ALLOWED",
  "message": "Email is not allowed"
}
```

- **Weak password**

```
{
  "errorCode": "WEAK_PASSWORD",
  "message": "Password does not meet security requirements"
}
```

- **Invalid input**

```
{
  "errorCode": "VALIDATION_ERROR",
  "message": "Invalid registration data"
}
```

## 1.2 Login

*Via Username & Password:*

```
POST /api/v1/auth/login
Request:
{
  "username": "john_doe",
  "password": "P@ssw0rd!"
}

Response:
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresIn": 2592000,
  "user": {
    "id": 1,
    "email": "user@example.com",
    "username": "john_doe",
    "role": "USER"
  }
}
```

- *Via Google:*

```
POST /api/v1/auth/login/google
Request:
{
  "googleToken": "eyJhbGciOiJSUzI1NiIsImtpZCI6..."
}

Response:
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresIn": 2592000,
  "user": {
    "id": 1,
    "email": "user@example.com",
    "username": "john_doe",
    "role": "USER"
  }
}
```

### 1.3 Logout (Token Revocation)

POST /api/v1/auth/logout

Request:

```
{
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

Response:

```
{
  "message": "Logged out successfully"
}
```

### 1.4 Refresh Token

POST /api/v1/auth/refresh

Request:

```
{
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

Response:

```
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expiresIn": 2592000
}
```

**Commented [ש3]:** Access token is **NOT** sent here (it may be expired)

#### Error Responses:

- **Invalid token**

```
{
  "errorCode": "INVALID_REFRESH_TOKEN",
  "message": "Refresh token is invalid or expired"
}
```

- **Token revoked**

```
{
  "errorCode": "REFRESH_TOKEN_REVOKED",
  "message": "Refresh token has been revoked"
}
```

## 1.5 Current User

GET /api/v1/auth/me

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "id": 1,
  "email": "user@example.com",
  "username": "john_doe",
  "role": "USER"
}
```

## 1.6 Forgot Password

POST /api/v1/auth/password/forgot

Request:

```
{
  "email": "user@example.com"
}
```

Response:

```
{
  "message": "If the email exists, a reset link has been sent"
}
```

## 1.7 Reset Password

POST /api/v1/auth/password/reset

Request:

```
{
  "resetToken": "abc123resetToken",
  "newPassword": "NewP@ssw0rd!"
}
```

Response:

```
{
  "message": "Password has been reset successfully"
}
```

## 1.8 Email Verification

### 1.8.1 Send Verification To Email

POST /api/v1/auth/email/verify

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "message": "Verification email sent successfully"
}
```

**Commented [שפ4]:** USED:  
After registration  
When user requests resend

### 1.8.2 Verify Email (Confirm)

POST /api/v1/auth/email/confirm

Request:

```
{
  "verificationToken": "abc123emailVerificationToken"
}
```

Response:

```
{
  "message": "Password has been reset successfully"
}
```

**Commented [שפ5]:** Triggered when the user clicks the link  
in the email.

Classification Endpoints

	Purpose	Endpoint
2.1	Get next batch	GET /classifications/next-batch
2.2	Submit classification	POST /classifications/{id}/submit
2.3	User progress	GET /classifications/progress
2.4.1	User overview	GET /statistics/me
2.4.2	User vs experts	GET /statistics/me/vs-experts
2.4.3	User vs users	GET /statistics/me/vs-users
2.4.4	Performance breakdown	GET /statistics/me/breakdown
2.4.5	Time series	GET /statistics/me/timeseries

## 2.1 Get Next Batch

GET /api/v1/classifications/next-batch?count=10

Request:

Headers:

Authorization: Bearer <accessToken>

```
{
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

Response:

```
{
  "images": [
    {
      "imageId": 1234,
      "taskId": 5,
      "question": "Is this a wasp?",
      "image": {
        "contentType": "image/jpeg",
        "data": "<base64-encoded-image>"
      },
      "referenceImages": [
        {
          "contentType": "image/jpeg",
          "data": "<base64-encoded-image>",
          "caption": "Example of a wasp - note the narrow waist"
        }
      ]
    }
  ]
}
```

## 2.2 Submit Classification

POST /api/v1/classifications/{classificationId}/submit

Request:

Headers:

Authorization: Bearer <accessToken>

```
{
  "imageId": 1234,
  "taskId": 5,
  "question": "Is this a wasp?",
  "decision": "YES",
  "responseTimeMs": 2340
}
```

Response:

```
{
  "images": [
    {
      "id": 1234,
      "taskId": 5,
      "question": "Is this a wasp?",
      "image": {
        "contentType": "image/jpeg",
        "data": "<base64-encoded-image>"
      }
    }
  ],
  "referenceImages": [
    {
      "contentType": "image/jpeg",
      "data": "<base64-encoded-image>",
      "caption": "Example of a wasp - note the narrow waist"
    }
  ]
}
```

## 2.3 Progress

GET /api/v1/classifications/progress

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "completed": 120,
  "accuracy": 0.91
}
```

## 2.4 User Statistics

**Base Path:**

*/api/v1/statistics*

**All endpoints require authentication.**

#### 2.4.1 User Classification Statistics

```
GET /api/v1/statistics/me
Request:
Headers:
Authorization: Bearer <accessToken>
Body:
  (none)

Response:
{
  "summary": {
    "totalClassifications": 320,
    "correctClassifications": 295,
    "accuracy": 0.92,
    "contributionPercentage": 0.42, // % of total dataset
    "rank": {
      "daily": 3,
      "weekly": 5,
      "monthly": 12,
      "allTime": 45
    },
    "rankPercentile": 87
  },
  "trend": {
    "byDay": [
      { "date": "2025-12-10", "accuracy": 0.89 },
      { "date": "2025-12-11", "accuracy": 0.91 },
      { "date": "2025-12-12", "accuracy": 0.94 }
    ]
  }
}
```

#### 2.4.2 User Vs Experts

```
GET /api/v1/statistics/me/vs-experts
Request:
Headers:
Authorization: Bearer <accessToken>
Body:
  (none)
```

Response:

```
{
  "user": {
    "accuracy": 0.92
  },
  "experts": {
    "accuracy": 0.97
  },
  "difference": {
    "accuracy": -0.05,
  }
}
```

#### 2.4.3 User Vs Community Comparison

GET /api/v1/statistics/me/vs-users

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "user": {
    "accuracy": 0.92
  },
  "experts": {
    "accuracy": 0.97
  },
  "difference": {
    "accuracy": -0.05
  }
}
```

#### 2.4.4 Detailed Performance Breakdown

GET /api/v1/statistics/me/breakdown

Request:

Headers:

Authorization: Bearer <accessToken>

Body:  
(none)

Response:

```
{
  "byCategory": [
    {
      "category": "WASP",
      "accuracy": 0.95,
      "total": 120
    },
    {
      "category": "BEE",
      "accuracy": 0.88,
      "total": 200
    }
  ]
}
```

#### 2.4.5 Time-Series Data (Graph-Ready)

GET /api/v1/statistics/me/timeseries?metric=accuracy&period=30d

Request:

Headers:

Authorization: Bearer <accessToken>

Body:  
(none)

Response:

```
{
  "metric": "accuracy",
  "points": [
    { "timestamp": "2025-12-01", "value": 0.88 },
    { "timestamp": "2025-12-02", "value": 0.90 }
  ]
}
```

## Leaderboard Endpoints

```
GET /api/v1/leaderboard/weekly?page=0&size=10
```

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "period": "weekly",
  "startDate": "2024-12-09",
  "endDate": "2024-12-15",
  "entries": [
    {
      "rank": 1,
      "username": "top_classifier",
      "totalPoints": 2500,
      "totalClassifications": 250,
      "accuracy": 89.5
    }
  ]
}
```

```
],  
  "currentUserRank": {  
    "rank": 15,  
    "totalPoints": 1050  
  },  
  "pagination": {  
    "page": 0,  
    "size": 10,  
    "totalPages": 5,  
    "totalElements": 48  
  }  
}
```

## Dashboard Endpoints

### Admin

	Purpose	Endpoint
3.1	View Tasks	GET /dashboard/tasks
3.2	Create Task	POST /dashboard/tasks/create
3.3	Activate / pause	POST /dashboard/tasks/{id}/activate
3.4	Task analytics	GET /dashboard/tasks/{id}/analytics
3.5	Export results	POST /dashboard/exports
3.6	Download export	GET /dashboard/exports/{id}/download
3.7	View Gold Images	GET /dashboard/images/
3.8	Upload Gold Image	POST /dashboard/images/upload
3.9	View Recipients List	GET /dashboard/recipients/
4.0	Create Recipients List	POST /dashboard/recipients/create
4.1	View Taxonomy	GET /dashboard/taxonomy
4.2	Delete Task	

User

	Purpose	Endpoint
4.2	My tasks	GET /dashboard/my-tasks
4.3	Task details	GET /dashboard/my-tasks/{id}
4.4	Play Task	GET /dashboard/my-tasks/{id}/play

GET /api/v1/leaderboard/weekly?page=0&size=10

Request:

Headers:

Authorization: Bearer <accessToken>

Body:

(none)

Response:

```
{
  "period": "weekly",
  "startDate": "2024-12-09",
  "endDate": "2024-12-15",
  "entries": [
    {
      "rank": 1,
      "userId": 42,
      "username": "top_classifier",
      "displayName": "Top Classifier",
      "totalPoints": 2500,
      "totalClassifications": 250,
      "accuracy": 89.5
    }
  ],
  "currentUserRank": {
    "rank": 15,
    "totalPoints": 1050
  },
  "pagination": {
    "page": 0,
    "size": 10,
    "totalPages": 5,
```

```
    "totalElements": 48  
  }  
}
```