

# Expense Tracker API

## Complete API Documentation

Version: 1.0.0

Generated: 2025-12-25 19:37

Base URL: <http://localhost:3001>

# Table of Contents

- 1. Authentication
- 2. Transactions
- 3. Categories
- 4. Items
- 5. Analytics
- 6. Export
- 7. AI Analysis
- 8. Error Codes

# 1. Authentication

All protected endpoints require a token in the request header.

```
Header: token: <your_jwt_token>
```

## ***POST /auth/signup***

Register a new user and send OTP verification

**Authentication:** No

**Request Body:**

```
{
  "firstName": "John",
  "lastName": "Doe",
  "email": "john@example.com",
  "password": "password123",
  "phone": "1234567890",
  "countryCode": "+20",
  "country": "Egypt"
}
```

**Response:**

```
{
  "message": "Signup successful, OTP sent to email"
}
```

## ***POST /auth/signup/configurationOTP***

Verify OTP and complete registration

**Authentication:** No

**Request Body:**

```
{
  "otp": "123456"
}
```

**Response:**

```
{
  "message": "User added successfully",
  "user": {...},
  "token": "jwt_token"
}
```

## ***POST /auth/signin***

Login with email and password

**Authentication:** No

**Request Body:**

```
{
  "email": "john@example.com",
  "password": "password123"
}
```

**Response:**

```
{
  "message": "Login successfully",
  "user": {...},
  "token": "jwt_token"
}
```

### ***GET /auth/profile***

Get current user profile

**Authentication:** Yes

**Request Body:**

None

**Response:**

```
{
  "message": "Profile retrieved successfully",
  "user": {...}
}
```

### ***PUT /auth/profile***

Update user profile

**Authentication:** Yes

**Request Body:**

```
{
  "firstName": "John",
  "lastName": "Smith",
  "phone": "0987654321"
}
```

**Response:**

```
{
  "message": "Profile updated successfully",
  "user": {...}
}
```

### ***POST /auth/changePassword***

Change user password

**Authentication:** Yes

**Request Body:**

```
{
  "oldPassword": "password123",
  "newPassword": "newpassword456"
}
```

**Response:**

```
{
  "message": "Password changed successfully",
  "token": "new_jwt_token"
}
```

### ***POST /auth/forgetPassword***

Request password reset link

**Authentication:** No

**Request Body:**

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

**Response:**

```
{
  "message": "Password reset email sent successfully"
}
```

***POST /auth/setNewPassword***

Set new password using reset token

**Authentication:** No (requires token in query)

**Request Body:**

```
{
  "newPassword": "newpassword789"
}
```

**Response:**

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

***POST /auth/resendOTP***

Resend OTP to email

**Authentication:** No

**Request Body:**

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

**Response:**

```
{
  "message": "resend OTP successful, OTP sent to email"
}
```

***DELETE /auth/account***

Delete user account and all data

**Authentication:** Yes

**Request Body:**

```
{
  "password": "password123"
}
```

**Response:**

```
{
  "message": "Account deleted successfully"
}
```

## 2. Transactions

Manage financial transactions with text, voice, or image input.

### ***POST /transactions/createWithText***

Create transaction using text input

**Auth:** Yes | **Body:** { "text": "Bought coffee", "items": [{"\_id": "item\_id"}], "price": 50 }

### ***POST /transactions/createWithVoice***

Create transaction using voice recording

**Auth:** Yes | **Body:** Form-data: voice\_path (file), items (JSON), price (number)

### ***POST /transactions/createWithOCR***

Create transaction using image/receipt

**Auth:** Yes | **Body:** Form-data: OCR\_path (file), items (JSON), price (number)

### ***GET /transactions***

Get all transactions with pagination

**Auth:** No | **Body:** Query: page, limit, sort, search, fields

### ***GET /transactions/my***

Get current user's transactions

**Auth:** Yes | **Body:** Query: page, limit, sort

### ***GET /transactions/:id***

Get single transaction by ID

**Auth:** Yes | **Body:** None

### ***GET /transactions/category/:categoryId***

Get transactions by category

**Auth:** Yes | **Body:** None

### ***GET /transactions/date-range***

Get transactions within date range

**Auth:** Yes | **Body:** Query: startDate, endDate (ISO format)

### ***PUT /transactions/:id***

Update transaction

**Auth:** Yes | **Body:** { "text": "Updated", "price": 75 }

### ***DELETE /transactions/:id***

Delete transaction

**Auth:** Yes | **Body:** None

### 3. Categories

Organize expenses into categories.

#### ***POST /category***

Create new category | Auth: Yes

#### ***GET /category***

Get all user categories | Auth: Yes

#### ***GET /category/:id***

Get single category | Auth: Yes

#### ***GET /category/:id/items***

Get category with populated items | Auth: Yes

#### ***PUT /category/:id***

Update category | Auth: Yes

#### ***DELETE /category/:id***

Delete category | Auth: Yes

#### ***POST /category/:id/items***

Add items to category | Auth: Yes

#### ***DELETE /category/:id/items***

Remove items from category | Auth: Yes



## 4. Items

Manage expense items.

***POST /items - Create new item***

***GET /items - Get all user items***

***GET /items/:id - Get single item***

***PUT /items/:id - Update item***

***DELETE /items/:id - Delete item***

***POST /items/add-to-category - Add item to category***

***POST /items/remove-from-category - Remove item from category***

## 5. Analytics

Get insights and statistics about spending.

***GET /analytics/summary - Get overall spending summary***

***GET /analytics/by-category - Get spending breakdown by category***

***GET /analytics/by-date - Get spending by date (daily/weekly/monthly)***

***GET /analytics/top-categories - Get top spending categories***

***GET /analytics/trends - Compare current vs previous month***

## 6. Export

Export transactions in various formats.

***GET /export/pdf - Export to PDF***

Query: startDate, endDate, categoryId (all optional)

***GET /export/csv - Export to CSV***

Query: startDate, endDate, categoryId (all optional)

***GET /export/json - Export to JSON***

Query: startDate, endDate, categoryId (all optional)

## 7. AI Analysis

AI-powered expense analysis.

***POST /ai/analyze - Analyze text/image for financial data***

***POST /ai/voice - Analyze voice recording (Speech-to-Text + Analysis)***

## 8. Error Codes

Code	Description
200	Success
201	Created
400	Bad Request - Invalid input
401	Unauthorized - Invalid/missing token
404	Not Found
409	Conflict - Resource already exists
429	Too Many Requests - Rate limit exceeded
500	Internal Server Error

### WebSocket Server: ws://localhost:3002

Real-time updates for transaction analysis results.