

Zus External Integration API

Version: 1.0.0

Last Updated: December 2025

This document is the authoritative API specification for integrating an external platform with the Zus Trading Agent backend (Cloudflare Workers). It covers authentication, endpoints, request/response schemas, webhooks, and working examples.

Base URL

- Production: `https://trading-agent-engine.sherry-mcadams001.workers.dev`
- Development (Wrangler): `http://127.0.0.1:8787`

Optional client env for the included frontend: set `VITE_API_URL` to your base URL.

Authentication

- Scheme: `Bearer JWT`
- Header: `Authorization: Bearer <token>`
- Public (no auth) endpoints:
 - `POST /api/auth/register`
 - `POST /api/auth/login`
 - `GET /api/pools`
 - `POST /api/invite-codes/validate`
 - `POST /api/webhook/nowpayments` (server-to-server webhook)
- All other endpoints require a valid token.

CORS

- Origin: `*`
- Methods: `GET, POST, PUT, DELETE, OPTIONS`
- Headers: `Content-Type, Authorization`

Common Response Shapes

- Standard success: `{ "status": "success", "data": <any> }`
- Standard error: `{ "status": "error", "error": "Message" }`
- Notifications endpoints use `{ "success": boolean, ... }` instead.

Errors

- 400: Validation error or bad request
- 401: Unauthorized / missing or invalid token
- 404: Not found
- 500: Internal error

Pagination

Where supported: `?limit=<number>&offset=<number>` (defaults vary by endpoint).

Endpoints

Auth

POST /api/auth/register

- Body: `{ "email": string, "password": string, "referralCode"??: string }`
- 201: `{ "status": "success", "data": { "user": { id, email, role, kyc_status, referral_code, created_at, updated_at }, "token": "<JWT>" } }`

POST /api/auth/login

- Body: `{ "email": string, "password": string }`
- 200: `{ "status": "success", "data": { "user": { ... }, "token": "<JWT>" } }`

POST /api/auth/logout

- 200: `{ "status": "success", "message": "Logged out successfully" }`

GET /api/auth/me

- 200: `{ "status": "success", "data": { id, email, role, kyc_status, referral_code, created_at, updated_at } }`

Examples:

```

BASE="https://trading-agent-engine.sherry-mcadams001.workers.dev"
curl -X POST "$BASE/api/auth/register" \
-H "Content-Type: application/json" \
-d '{"email":"user@example.com","password":"secret","referralCode":"ABC123"}'

TOKEN=$(curl -s -X POST "$BASE/api/auth/login" -H "Content-Type: application/json" -d
'{"email":"user@example.com","password":"secret"})' | jq -r .data.token)
curl -H "Authorization: Bearer $TOKEN" "$BASE/api/auth/me"

```

Wallet

All wallet routes require auth.

GET /api/wallet

- 200: { "status": "success", "data": { id, user_id, available_balance, locked_balance, pending_balance, currency, updated_at } }

GET /api/wallet/deposit-address

- Legacy static address (if configured).
- 200: { "status": "success", "data": { "address": string, "network": "TRC20" } }

POST /api/wallet/create-payment

- Body: { "amount": number >= 20, "currency": "btc" | "eth" | "ltc" | "usdttrc20" }
- 201: { "status": "success", "data": { transactionId, paymentId, payAddress, payAmount, payCurrency, priceAmount, priceCurrency } }

GET /api/wallet/payment-status/:paymentId

- 200: { "status": "success", "data": { transactionId, status, amount, metadata } }

POST /api/wallet/deposit

- Body: { "amount": number > 0, "description"? : string, "txHash"? : string, "network"? : string }
- 201: { "status": "success", "data": { ...transaction } }

POST /api/wallet/withdraw

- Body: { "amount": number > 0, "description"? : string }
- 201: { "status": "success", "data": { ...transaction } }

GET /api/wallet/transactions?limit=50&offset=0

- 200: { "status": "success", "data": [...transactions] }
-

Portfolio

All portfolio routes require auth.

GET /api/portfolio

- 200: { "status": "success", "data": { id, user_id, total_invested, total_pnl, total_trades, winning_trades, losing_trades, current_bot } }

GET /api/portfolio/trades?limit=50&offset=0

- 200: { "status": "success", "data": [{ id, user_id, session_id, symbol, side, price, quantity, pnl, created_at }...] }
-

Pools (Staking)

GET /api/pools (public)

- 200: { "status": "success", "data": [{ id, name, bot_tier, min_stake, max_stake, total_capacity, current_staked, roi_min, roi_max, lock_p }...] }

Auth required for the following:

GET /api/pools/stakes

- 200: { "status": "success", "data": [{ id, user_id, pool_id, amount, status, staked_at, unstake_available_at, unstaked_at, total_earned }...] }

GET /api/pools/stakes/active

- 200: { "status": "success", "data": [...activeStakes] }

GET /api/pools/summary

- 200: { "status": "success", "data": { "total_staked": number, "total_earned": number, "active_stakes_count": number } }

POST /api/pools/stake

- Body: { "pool_id": number, "amount": number>0 }
 - 201: { "status": "success", "data": { ...stake } }
-

Referrals

All referral routes require auth.

GET /api/referrals

- 200: { "status": "success", "data": [...directReferrals] }

GET /api/referrals/network

- 200: { "status": "success", "data": { /* nested levels */ } }

GET /api/referrals/volume

- 200: { "status": "success", "data": { "total_volume": number } }

GET /api/referrals/volume/breakdown

- 200: { "status": "success", "data": [{ level: number, volume: number }...] }

GET /api/referrals/stats

- 200: { "status": "success", "data": { /* aggregate stats */ } }
-

Profile

All profile routes require auth.

GET /api/profile

- 200: { "status": "success", "data": { "profile": { ... }, "tiers": [{ id, name, /* ROI config */ , eligible: true }...] } }

PUT /api/profile

- Body: `Partial<Profile>` (server validates)
- 200: { "status": "success", "data": { ...updatedProfile } }

POST /api/profile/strategy

- Body: { "tier": "delta"|"gamma"|"alpha"|"omega" }
- 200: { "status": "success", "data": { ... } }

GET /api/profile/strategy

- 200: { "status": "success", "data": { "tier": string } }
-

Dashboard (Aggregate)

All dashboard routes require auth.

GET /api/dashboard

- 200: { "status": "success", "data": { wallet, portfolio, trades, transactions, staking: { activeStakes, totalStaked, totalEarned }, referrals: { partnerVolume, stats }, roi: { currentRatePercent, actualDailyRatePercent, currentHourlyEarning, projectedDailyEarning, actualDailyEarning, rateMultiplier, marketSentiment, volatility, displayRate, tier, history }, timestamp } }
-

Invite Codes**POST /api/invite-codes/validate (public)**

- Body: { "code": string }
- 200/400: { "status": "success"|"error", ... }

Auth required for the following:

POST /api/invite-codes

- 201: { "status": "success", "data": { code, status, created_at, ... } }

GET /api/invite-codes

- 200: { "status": "success", "data": [...codes] }

GET /api/invite-codes/active

- 200: { "status": "success", "data": [...activeCodes] }

GET /api/invite-codes/stats

- 200: { "status": "success", "data": { total, active, redeemed, ... } }
-

Notifications

All notification routes require auth. Response convention is `success: boolean`.

GET /api/notifications?limit=50

- 200: { "success": true, "data": { "notifications": [...], "unreadCount": number } }

GET /api/notifications/unread-count

- 200: { "success": true, "data": { "unreadCount": number } }

POST /api/notifications/read/:id

- 200: { "success": true, "message": "Notification marked as read" }

POST /api/notifications/read-all

- 200: { "success": true, "message": "Marked N notifications as read" }

DELETE /api/notifications/:id

- 200: { "success": true, "message": "Notification deleted" }

DELETE /api/notifications

- 200: { "success": true, "message": "Deleted N notifications" }
-

Admin (Admin token required)**GET /api/admin/deposits/pending**

- 200: { "status": "success", "data": [...pendingDeposits] }

POST /api/admin/deposits/:id/approve

- 200: { "status": "success", "data": { ...updatedTransaction } }

GET /api/admin/users

- 200: { "status": "success", "data": [{ id, email, role, created_at, kyc_status, balance, last_login }...] }

GET /api/admin/settings

- 200: { "status": "success", "data": { [key:string]: string } }

POST /api/admin/settings

- Body: { "key": string, "value": string }
- 200: { "status": "success", "data": { key, value } }

GET /api/admin/users/:id

- 200: { "status": "success", "data": { user, wallet, transactions:[...] } }

POST /api/admin/users/:id/balance

- Body: { "amount": number, "description?": string }
- 200: { "status": "success", "data": { ...transaction } }

Webhooks

NowPayments IPN**POST /api/webhook/nowpayments**

- Header: `x-nowpayments-sig: <signature>` (verified when secret set)
- Body: NowPayments IPN fields including `payment_id`, `payment_status`, `order_id`, `price_amount`, `actually_paid`
- Behavior:
 - Validates signature if provided
 - Updates the corresponding transaction status
 - When completed, credits the user wallet with `price_amount` (USD)
- 200: { "status": "success" }

Order ID format used by the system: `user_{userId}_tx_{transactionId}`

Usage Guide

1) Register, Login, and Fetch Dashboard

```
BASE="https://trading-agent-engine.sherry-mcadams001.workers.dev"

curl -X POST "$BASE/api/auth/register" \
-H "Content-Type: application/json" \
-d '{"email":"user@example.com","password":"secret"}'

TOKEN=$(curl -s -X POST "$BASE/api/auth/login" \
-H "Content-Type: application/json" \
-d '{"email":"user@example.com","password":"secret"}' | jq -r .data.token)

curl -H "Authorization: Bearer $TOKEN" "$BASE/api/dashboard"
```

2) Create a Crypto Deposit (NowPayments)

```
curl -X POST "$BASE/api/wallet/create-payment" \
-H "Authorization: Bearer $TOKEN" \
-H "Content-Type: application/json" \
-d '{"amount": 100, "currency": "usdttrc20"}'

# Response includes payAddress/payAmount. User pays to that address.
# Later, check status:
curl "$BASE/api/wallet/payment-status/<paymentId>" -H "Authorization: Bearer $TOKEN"
```

3) Stake Into a Pool

```
curl "$BASE/api/pools" # public catalog

curl -X POST "$BASE/api/pools/stake" \
-H "Authorization: Bearer $TOKEN" \
-H "Content-Type: application/json" \
-d '{"pool_id": 1, "amount": 500}'
```

4) Programmatic Client (Axios)

```
import axios from 'axios';

const api = axios.create({ baseURL: import.meta.env.VITE_API_URL });
api.interceptors.request.use(cfg => {
  const t = localStorage.getItem('authToken');
  if (t) cfg.headers.Authorization = `Bearer ${t}`;
  return cfg;
});

export async function getDashboard() {
  const res = await api.get('/api/dashboard');
  return res.data.data;
}
```

Bot Tiers

IDs: `delta` , `gamma` , `alpha` , `omega` . Tier ROI ranges and configuration are returned via profile and pools endpoints. Users can set a preferred strategy via `POST /api/profile/strategy` .

Security Notes

- Always use HTTPS for public endpoints.
 - Store JWTs securely; rotate tokens as needed.
 - Verify NowPayments IPN with `x-nowpayments-sig` when `NOWPAYMENTS_IPN_SECRET` is configured.
 - Apply principle of least privilege for admin tokens.
-

Changelog

- Dec 2025: First consolidated external API spec covering auth, wallet, portfolio, pools, referrals, profile, notifications, dashboard, admin, and NowPayments webhook.