CrymadX Admin Dashboard

Complete API Documentation for Backend Developers

Version: 1.0.0 **Base URL:** https://api.crymadx.com/admin/v1 **Last Updated:** December 28, 2025

Table of Contents

100+	18	37
API Endpoints	Feature Modules	Admin Screens

Authentication Required: All admin API requests require a valid JWT Bearer token with admin privileges. Include the token in the Authorization header.

1. Authentication

POST `/auth/login`

Admin login to dashboard.

Request Body:

```
{
  "email": "admin@crymadx.com",
  "password": "securePassword123",
  "rememberMe": true
}
```

Response (200):

```
{
  "success": true,
  "data": {
    "admin": {
      "id": "adm_123456",
      "name": "John Admin",
      "email": "admin@crymadx.com",
      "role": "super_admin",
      "permissions": ["users.read", "users.write", ...],
      "lastLogin": "2025-12-27T10:30:00Z"
    },
    "accessToken": "eyJhbGciOiJIUzI1NiIs...",
    "refreshToken": "eyJhbGciOiJIUzI1NiIs...",
    "expiresIn": 3600
  }
}
```

POST `/auth/logout`

Logout admin session.

POST `/auth/refresh`

Refresh access token.

GET `/auth/me`

Get current admin profile.

2. Dashboard Analytics

GET /dashboard/stats

Get main dashboard statistics.

Response (200):


```
{
  "success": true,
  "data": {
    "totalUsers": { "value": 12543, "change": 12.5, "trend": "up" },
    "activeUsers24h": { "value": 1893, "change": 8.2, "trend": "up" },
    "tradingVolume24h": { "value": 2450000, "change": 15.3, "trend": "up" },
    "revenueToday": { "value": 12500, "change": -2.1, "trend": "down" },
    "pendingItems": {
      "kycPending": 23,
      "withdrawalsPending": 15,
      "ticketsOpen": 8
    }
  }
}
```


GET /dashboard/user-growth

Get user growth chart data.

Parameter	Type	Default	Description
period	string	"6months"	Time period: 7days, 30days, 6months, 1year

GET /dashboard/trading-volume

Get trading volume chart data.

GET /dashboard/revenue-breakdown

Get revenue breakdown by source.

GET /dashboard/recent-activity

Get recent platform activity feed.

GET /dashboard/top-traders

Get top traders leaderboard.

3. User Management

GET `/users`

Get paginated list of all users.

Parameter	Type	Default	Description
page	number	1	Page number
limit	number	20	Items per page
search	string	-	Search by name/email
status	string	-	Filter: active, inactive, suspended, pending
kycLevel	number	-	Filter by KYC level (0-3)
sortBy	string	"createdAt"	Sort field
sortOrder	string	"desc"	Sort direction: asc, desc

GET `/users/:userId`

Get detailed user profile.

PUT `/users/:userId`

Update user profile.

POST `/users/:userId/suspend`

Suspend a user account.

POST `/users/:userId/unsuspend`

Reactivate a suspended user.

POST `/users/:userId/adjust-balance`

Manually adjust user balance (admin action).

GET **/users/:userId/wallet**

Get user's wallet details.

GET **/users/:userId/trades**

Get user's trading history.

GET **/users/:userId/kyc**

Get user's KYC documents.

GET **/users/:userId/sessions**

Get user's active sessions.

GET **/users/:userId/activity**

Get user's activity log.

POST **/users/bulk-action**

Perform bulk action on multiple users.

4. KYC Management

GET `/kyc/queue`

Get pending KYC verification requests.

Parameter	Type	Default	Description
page	number	1	Page number
limit	number	20	Items per page
level	number	-	Filter by requested level (1, 2, 3)

GET `/kyc/:kycId`

Get KYC request details for review.

POST `/kyc/:kycId/approve`

Approve KYC verification request.

POST `/kyc/:kycId/reject`

Reject KYC verification request.

5. Wallet Management

GET /wallets/overview

Get platform wallet overview.

GET /wallets/assets

Get all platform supported assets.

POST /wallets/assets

Add new supported asset.

PUT /wallets/assets/:assetId

Update asset configuration.

DELETE /wallets/assets/:assetId

Remove asset (soft delete).

GET /wallets/deposits

Get all deposit transactions.

Parameter	Type	Description
status	string	Filter: pending, processing, completed, failed
asset	string	Filter by asset symbol
userId	string	Filter by user
startDate	string	Start date (ISO 8601)
endDate	string	End date (ISO 8601)

GET /wallets/withdrawals

Get all withdrawal requests.

POST `/wallets/withdrawals/:withdrawalId/approve`

Approve withdrawal request.

POST `/wallets/withdrawals/:withdrawalId/reject`

Reject withdrawal request.

POST `/wallets/sync-balances`

Sync platform balances with blockchain.

6. Trading Management

GET `/trading/pairs`

Get all trading pairs.

POST `/trading/pairs`

Create new trading pair.

PUT `/trading/pairs/:pairId`

Update trading pair configuration.

DELETE `/trading/pairs/:pairId`

Delete trading pair (soft delete).

POST `/trading/pairs/:pairId/toggle`

Activate/Deactivate trading pair.

GET `/trading/fees`

Get fee tier configuration.

Fee Tiers:

Tier	30d Volume	Maker Fee	Taker Fee
Standard	$0 - $10K	0.10%	0.15%
Bronze	$10K - $50K	0.09%	0.14%
Silver	$50K - $100K	0.08%	0.12%
Gold	$100K - $500K	0.06%	0.10%
Platinum	$500K - $1M	0.04%	0.08%
VIP	$1M+	0.02%	0.05%

PUT /trading/fees/:tier

Update fee tier.

7. P2P Management

GET `/p2p/traders`

Get all P2P traders.

GET `/p2p/traders/:traderId`

Get trader details.

POST `/p2p/traders/:traderId/ban`

Ban a P2P trader.

POST `/p2p/traders/:traderId/unban`

Unban a P2P trader.

GET `/p2p/orders`

Get all P2P orders.

GET `/p2p/orders/:orderId`

Get P2P order details.

GET `/p2p/disputes`

Get all P2P disputes.

GET `/p2p/disputes/:disputeId`

Get dispute details with chat history.

POST `/p2p/disputes/:disputeId/resolve`

Resolve a dispute.

POST `/p2p/disputes/:disputeId/message`

Send admin message in dispute chat.

GET /p2p/payment-methods

Get all P2P payment methods.

POST /p2p/payment-methods

Add new payment method.

PUT /p2p/payment-methods/:methodId

Update payment method.

DELETE /p2p/payment-methods/:methodId

Remove payment method.

8. Rewards & Tasks

GET `/rewards/tasks`

Get all reward tasks.

POST `/rewards/tasks`

Create new reward task.

PUT `/rewards/tasks/:taskId`

Update reward task.

DELETE `/rewards/tasks/:taskId`

Delete reward task.

GET `/rewards/vip-tiers`

Get VIP tier configuration.

VIP Tiers:

Tier	Points Required	Benefits
Bronze	0	Basic support, Standard fees
Silver	1,000	Priority support, 5% fee discount
Gold	5,000	24/7 support, 15% fee discount, Exclusive events
Platinum	20,000	Dedicated manager, 25% fee discount, VIP events

PUT `/rewards/vip-tiers/:tier`

Update VIP tier configuration.

9. Savings Products

GET `/savings/products`

Get all savings products.

POST `/savings/products`

Create new savings product.

PUT `/savings/products/:productId`

Update savings product.

DELETE `/savings/products/:productId`

Deactivate savings product.

GET `/savings/deposits`

Get all user savings deposits.

10. Support & Tickets

`GET` **`/support/tickets`**

Get all support tickets.

`GET` **`/support/tickets/:ticketId`**

Get ticket details with message thread.

`POST` **`/support/tickets/:ticketId/reply`**

Reply to a support ticket.

`POST` **`/support/tickets/:ticketId/resolve`**

Mark ticket as resolved.

`PUT` **`/support/tickets/:ticketId/priority`**

Update ticket priority.

`PUT` **`/support/tickets/:ticketId/assign`**

Assign ticket to admin.

`GET` **`/support/canned-responses`**

Get canned response templates.

`POST` **`/support/canned-responses`**

Create canned response.

11. Referrals

`GET` **`/referrals/codes`**

Get all referral codes.

`POST` **`/referrals/codes/generate`**

Generate referral code for user.

`PUT` **`/referrals/codes/:codeId`**

Update referral code.

`DELETE` **`/referrals/codes/:codeId`**

Deactivate referral code.

`GET` **`/referrals/commissions`**

Get commission payouts.

12. CMS & Content

`GET` **`/cms/homepage`**

Get homepage content.

`PUT` **`/cms/homepage`**

Update homepage content.

`POST` **`/cms/homepage/hero-image`**

Upload hero background image.

`GET` **`/cms/announcements`**

Get platform announcements.

`POST` **`/cms/announcements`**

Create announcement.

`PUT` **`/cms/announcements/:announcementId`**

Update announcement.

`DELETE` **`/cms/announcements/:announcementId`**

Delete announcement.

`GET` **`/cms/pages`**

Get custom pages.

`POST` **`/cms/pages`**

Create custom page.

13. Settings

`GET` `/settings/general`

Get general platform settings.

`PUT` `/settings/general`

Update general settings.

`GET` `/settings/security`

Get security settings.

`PUT` `/settings/security`

Update security settings.

`GET` `/settings/notifications`

Get notification settings.

`PUT` `/settings/notifications`

Update notification settings.

14. Admin & Roles

GET `/admins`

Get all admin users.

POST `/admins`

Create new admin.

PUT `/admins/:adminId`

Update admin.

DELETE `/admins/:adminId`

Remove admin (cannot remove super_admin).

GET `/roles`

Get all roles with permissions.

Available Roles:

Role	Permissions
Super Admin	All permissions (*)
Admin	users.*, kyc.*, withdrawals.*
Moderator	users.read, tickets.*, disputes.*
Support	users.read, tickets.*
Analyst	dashboard.read, analytics.read, reports.read

POST `/roles`

Create new role.

PUT `/roles/:roleId`

Update role permissions.

DELETE `/roles/:roleId`

Delete role.

15. Audit Logs

GET `/audit-logs`

Get admin audit logs.

Parameter	Type	Description
page	number	Page number
limit	number	Items per page
adminId	string	Filter by admin
action	string	Filter by action type
startDate	string	Start date
endDate	string	End date

GET `/audit-logs/export`

Export audit logs (CSV or JSON).

16. WebSocket Events

WebSocket URL: `wss://api.crymadx.com/admin/ws`

Events (Server to Client)

Event	Description
`dashboard:stats_update`	Real-time dashboard statistics update
`user:registered`	New user registration
`kyc:submitted`	New KYC submission
`withdrawal:requested`	New withdrawal request
`ticket:created`	New support ticket
`dispute:opened`	New P2P dispute
`trade:executed`	Trade executed (high volume)

17. Data Models

Admin

```
interface Admin {
  id: string;
  name: string;
  email: string;
  avatar?: string;
  role: 'super_admin' | 'admin' | 'moderator' | 'support' | 'analyst';
  permissions: string[];
  status: 'active' | 'inactive';
  twoFactorEnabled: boolean;
  lastLogin: string;
  createdAt: string;
}
```

User

```
interface User {
  id: string;
  name: string;
  email: string;
  phone?: string;
  avatar?: string;
  status: 'active' | 'inactive' | 'suspended' | 'pending';
  role: 'user' | 'vip';
  kycLevel: 0 | 1 | 2 | 3;
  country?: string;
  city?: string;
  joinedAt: string;
  lastActive: string;
  security: {
    twoFactorEnabled: boolean;
    emailVerified: boolean;
    phoneVerified: boolean;
  };
  wallet: {
    totalBalance: number;
    availableBalance: number;
    lockedBalance: number;
  };
  referral: {
    code: string;
    referredBy?: string;
    earnings: number;
  };
}
```

Withdrawal

```
interface Withdrawal {
  id: string;
  userId: string;
  userName: string;
  asset: string;
  amount: number;
  fee: number;
  netAmount: number;
  usdValue: number;
  address: string;
  network: string;
  riskScore: 'low' | 'medium' | 'high';
  status: 'pending' | 'approved' | 'processing' | 'completed' | 'rejected';
  txHash?: string;
  createdAt: string;
  processedAt?: string;
}
```

TradingPair

```
interface TradingPair {
  id: string;
  pair: string;
  baseAsset: string;
  quoteAsset: string;
  status: 'active' | 'inactive';
  volume24h: number;
  trades24h: number;
  makerFee: number;
  takerFee: number;
  minOrderSize: number;
  maxOrderSize: number;
}
```

Ticket

```
interface Ticket {
  id: string;
  userId: string;
  userName: string;
  subject: string;
  category: string;
  priority: 'low' | 'medium' | 'high';
  status: 'new' | 'pending' | 'resolved' | 'closed';
  assignedTo?: string;
  createdAt: string;
  lastReply: string;
}
```

18. Error Handling

Error Response Format

```
{
  "success": false,
  "error": {
    "code": "ERROR_CODE",
    "message": "Human readable error message",
    "details": {}
  }
}
```

Common Error Codes

Code	HTTP Status	Description
UNAUTHORIZED	401	Invalid or expired token
FORBIDDEN	403	Insufficient permissions
NOT_FOUND	404	Resource not found
VALIDATION_ERROR	400	Invalid request data
DUPLICATE_ENTRY	409	Resource already exists
RATE_LIMITED	429	Too many requests
INTERNAL_ERROR	500	Server error

Permission Codes

Permission	Description
users.read	View user data
users.write	Modify user data
users.suspend	Suspend/unsuspend users
kyc.read	View KYC requests
kyc.write	Approve/reject KYC
withdrawals.read	View withdrawals
withdrawals.approve	Approve/reject withdrawals
trading.read	View trading data
trading.write	Modify trading pairs/fees
tickets.read	View tickets
tickets.write	Reply to tickets
settings.read	View settings
settings.write	Modify settings
admins.read	View admins
admins.write	Manage admins
analytics.read	View analytics
audit.read	View audit logs

Rate Limiting

Endpoint Type	Limit
Standard endpoints	100 requests/minute
Sensitive endpoints (auth, withdrawals)	20 requests/minute
Export endpoints	5 requests/minute

CrymadX Admin Dashboard API Documentation

Version 1.0.0 | Last Updated: December 28, 2025

For User Dashboard API, see `CrymadX_API_Documentation.md`