

# ProjectX: Complete User Flows Explained Simply

## Imagine This Like a Game...


Think of ProjectX like a **video game marketplace** where:

- **USDC** = Game coins you buy stuff with (small purchases)
  - **AVAX** = Real treasure you lock in a vault (big payments)
  - **Badge NFTs** = Achievement badges that prove you're skilled
  - **Community NFT** = VIP membership pass
  - **Smart Contracts** = Robot referees that hold money until work is done
- 

## FLOW 1: Regular User (Worker) Journey

**Scenario: Sarah wants to find gigs and get paid**

## STEP 1: First Time Visiting ProjectX

- ➔ Sarah visits ProjectX.com
- ➔ Sees: "Browse 500+ gigs" and "Connect Wallet" button
- ➔ Decision Point: Does she want to browse or join?
- ➔ Option A: Browse FREE (no wallet needed)
  - ↳ Can see gig titles and basic info
  - ↳ Can't see full details or apply
- ↳ Option B: Connect Wallet
  - ↳ Clicks "Connect Wallet"
  - ↳ MetaMask pops up → Select Avalanche C-Chain
  - ↳ Wallet connected! 
  - ↳ Dashboard appears

## STEP 2: Choose Membership Model

- ➔ Sarah sees 3 options:

 **OPTION 1: Community NFT (VIP Pass)**

Price: \$49 USDC (one-time)

Benefits:

- Unlimited gig applications
- Access to premium gigs
- Early access to new features
- Community Discord role
- NFT shows in wallet (bragging rights!)

[Buy Community NFT - \$49 USDC]

 **OPTION 2: Monthly Subscription**

Price: \$9.99 USDC/month (via x402)

Benefits:

- Unlimited applications this month
- Auto-renews each month
- Cancel anytime
- Same features as Community NFT

[Subscribe - \$9.99/month]

 **OPTION 3: Pay-Per-Apply (Free Tier)**

Price: \$0.02 USDC per application

Benefits:

- No commitment
- Only pay when you apply
- Perfect for casual job seekers

[Use Free Tier]

→ Sarah chooses: Monthly Subscription (\$9.99)

### STEP 3: Subscribe via x402

→ Clicks "Subscribe - \$9.99/month"

→ Payment modal appears:

 Confirm Subscription

Amount: \$9.99 USDC

Frequency: Monthly (auto-renew)

Network: Avalanche C-Chain

You'll be charged \$9.99 USDC every month starting today.

[Cancel] [Pay with USDC] 

→ Sarah clicks "Pay with USDC"

→ MetaMask pops up: "Sign transaction"

→ x402 processes payment (2 seconds)

→  Subscription active!

→ Sarah's wallet now shows:

→ USDC: 40.01 (was 50, now -9.99)

→ Subscription badge appears in profile

### STEP 4: Get a Skill Badge (Optional but Recommended)

→ Sarah sees banner: "Get verified to stand out! 

→ Clicks "Get Verified"

→ Verification page opens:

 Skill Verification

Choose skill: [Web Development ▼]

Portfolio URL: [\_\_\_\_\_]

GitHub: [\_\_\_\_\_]

Cost: \$5.00 USDC


Review time: 24 hours


[Submit for Verification]

- Sarah pays \$5 USDC via x402
- Admin reviews her portfolio (24 hrs)
-  Admin approves!
- Smart contract mints Badge NFT to Sarah's wallet
- Sarah's profile now shows:
  - "Web Development" badge  (verified on blockchain)


#### STEP 5: Browse and Apply to Gigs

- Sarah goes to "Browse Gigs"
- Sees list of gigs:


 Build Landing Page (Featured)
Payment: 0.5 AVAX (\$20)
Required: Web Development badge
Employer: 0xABC... (  Verified)
[View Details] [Apply Now]

- Sarah clicks "Apply Now"
- Since she has monthly subscription: FREE APPLICATION!   
(If she was pay-per-apply: Would cost \$0.02 USDC)
- Application submitted!
- Sarah sees: "Application sent to employer"

#### STEP 6: Get Selected and Work

- Employer reviews Sarah's profile
- Sees her "Web Development" badge 
- Employer assigns gig to Sarah
- Sarah receives notification: "You got the gig! 🎉"
- Sarah sees gig status: "In Progress"
- Payment shown: "0.5 AVAX locked in escrow"
- Sarah builds the landing page (3 days)
- Uploads work, clicks "Submit Work"
- Status changes: "Waiting for employer approval"

#### STEP 7: Get Paid!

- Employer reviews work → Approves
- Employer clicks "Release Payment"
- Smart contract automatically sends 0.5 AVAX to Sarah
- Sarah's wallet: +0.5 AVAX (\$20) 

→ Success screen:

🎉 Payment Received!	
Amount: 0.5 AVAX (\$20)	
From: 0xABC...	
Gig: Build Landing Page	
Your earnings: 2.5 AVAX this month	
Completed gigs: 5	

#### STEP 8: Monthly Subscription Renewal

- 30 days pass...
- x402 automatically charges Sarah \$9.99 USDC
- Sarah receives email: "Subscription renewed ✅"
- No interruption in service
- If Sarah's USDC balance is too low:
  - x402 payment fails
  - Sarah receives notification
  - Account switches to "Free Tier" (pay-per-apply)
  - Sarah can top up and resubscribe anytime

## 👛 FLOW 2: Employer Journey

**Scenario: John wants to hire a developer**

## JOHN'S JOURNEY

### STEP 1: Discover ProjectX



- ➔ John searches: "hire web developer no commission"
- ➔ Finds ProjectX: "0% commission gig marketplace"
- ➔ Clicks "Post a Gig"
- ➔ Redirected to connect wallet

### STEP 2: Connect Wallet & Setup

- ➔ Connects MetaMask (Avalanche C-Chain)
- ➔ Wallet connected: 0xDEF...
- ➔ Wallet balance:
  - ➔ AVAX: 5.0 AVAX (for escrow payments)
  - ➔ USDC: 50 USDC (for platform fees)
- ➔ Account type: Employer 

### STEP 3: Create a Gig (FREE)

- ➔ Clicks "Post a Gig"
- ➔ Fill out form:

	Create New Gig
Title: [Build Landing Page_____]	
Description:	
[Need a responsive landing page with contact form. Must work on mobile...]	
Payment: [0.5] AVAX ( ~ \$20)	
Required Badge: [Web Development ▼]	
Deadline: [7 days ▼]	
Cost to post: FREE 	
[Create Gig]	

- ➔ John clicks "Create Gig"
- ➔ Backend saves gig to database
- ➔ Gig appears in "Browse Gigs" (basic listing)
- ➔ Success! Gig ID: #123 created

### STEP 4: Feature the Gig (OPTIONAL - \$0.50)

- ➔ John sees his gig at bottom of list
- ➔ Wants more visibility
- ➔ Clicks "★ Feature This Gig"

→ Payment modal appears:

★ Feature Your Gig
Your gig will appear at the top of search results for 24 hours.
Cost: \$0.50 USDC
Network: Avalanche C-Chain
[Cancel] [Pay \$0.50 USDC] <input checked="" type="checkbox"/>

→ John pays \$0.50 USDC via x402

→ ☒ Payment confirmed (2 seconds)

→ Gig now shows at top with ★ badge

→ Platform earned: \$0.50 USDC

#### STEP 5: Lock Payment in Escrow

→ John sees banner: "⚠ Lock payment to activate gig"

→ Clicks "Lock Payment in Escrow"

→ MetaMask pops up:

🔒 Lock Payment in Escrow
Amount: 0.5 AVAX
To: GigEscrow Contract
Gas: ~\$0.002
This payment will be held by the smart contract until work is completed and approved.
[Reject] [Confirm] <input checked="" type="checkbox"/>

→ John confirms transaction

→ Smart contract locks 0.5 AVAX

→ Transaction on Snowtrace: <https://testnet.snowtrace.io/tx/0x...>

→ Gig status: "Open - Payment Locked" ☒

#### STEP 6: Review Applications

→ 3 workers apply (Sarah, Mike, Lisa)

→ John sees applications dashboard:

📄 Applications (3)

👤 Sarah

Badges: ★ Web Development (Verified)

Completed: 12 gigs

Rating: 4.8/5.0

[View Profile] [Assign Gig]

👤 Mike

Badges: (none)

Completed: 2 gigs

Rating: 4.0/5.0

[View Profile] [Assign Gig]

➡ John picks Sarah (has verified badge)

➡ Clicks "Assign Gig"

➡ Sarah receives notification: "You got the gig!"

#### STEP 7: Worker Submits Work

➡ Sarah builds landing page (3 days)

➡ Sarah clicks "Submit Work"

➡ John receives notification: "Work submitted for review"

➡ John reviews:

🔍 Review Submitted Work

Worker: Sarah

Deliverable: <https://landing-page.com>

Message: "All requirements completed!"

Escrow: 0.5 AVAX (locked)

[Request Changes] [Approve & Pay] ✓

➡ John tests the website → Works perfectly!

#### STEP 8: Approve and Release Payment

➡ John clicks "Approve & Pay"

➡ MetaMask pops up: "Call releasePayment() function"

➡ Smart contract executes:

1. Checks gig status = SUBMITTED ✓

2. Checks caller = employer ✓

3. Transfers 0.5 AVAX → Sarah's wallet

4. Updates status = COMPLETED

5. Emit's PaymentReleased event

5. Emits PaymentReleased event

Transaction confirmed!

Sarah receives 0.5 AVAX instantly ✓

Success screen:

✓ Payment Released!

Amount: 0.5 AVAX

To: Sarah (0xABC...)

Total platform fees: \$0.50

(vs Upwork would charge: \$4.00)

[Post Another Gig]

#### STEP 9: Get Verified Employer Badge (OPTIONAL)

John posts 5 successful gigs

Pays \$10 USDC for "Verified Employer" badge

Admin reviews history → Approves

Badge NFT minted to John's wallet

Future gigs show "★ Verified Employer"

Workers trust John more → More applications

## FLOW 3: Admin Journey

**Scenario: You (the platform admin) managing ProjectX**

## ADMIN JOURNEY

### STEP 1: Admin Dashboard Login


- ➔ Visit: admin.projectx.com
- ➔ Connect admin wallet: 0xADMIN...
- ➔ Two-factor authentication ✓

➔ Admin dashboard loads:

 ProjectX Admin Dashboard
Today's Stats:
• Revenue: \$127.50 USDC
• New Users: 23
• Active Gigs: 156
• Pending Verifications: 8
[Badge Verification] [Revenue] [Users]

### STEP 2: Review Badge Verification Requests

- ➔ Click "Badge Verification" tab
- ➔ See pending requests:

 Pending Verifications (8)
Request #1:
User: Sarah (0xABC...)
Skill: Web Development
Paid: \$5.00 USDC ✓
Submitted: 2 hours ago
Portfolio: https://sarah.dev
GitHub: github.com/sarah (234 repos)
LinkedIn: linkedin.com/in/sarah
[View Portfolio] [Approve] [Reject]

- ➔ Admin reviews Sarah's portfolio:
  - ➔ Checks GitHub: Has React projects ✓
  - ➔ Checks portfolio: Looks professional ✓
  - ➔ Decision: APPROVE ✓
- ➔ Admin clicks "Approve"

### STEP 3: Mint Badge NFT (On-Chain)

STEP 3: Mint Badge for User (On-chain)

➔ "Approve" button triggers backend API:

POST /api/admin/badges/mint

```
{
  "verificationId": "ver_123",
  "userAddress": "0xABC...",
  "skillName": "Web Development",
  "level": "INTERMEDIATE"
}
```

➔ Backend calls smart contract:

```
const tx = await badgeContract.mintBadge(
  "0xABC...", // Sarah's address
  "Web Development", // Skill name
  "ipfs://badge-icon.png", // Icon
  BadgeLevel.INTERMEDIATE // Level
);
```

➔ Transaction broadcast to Avalanche

➔ Mining... (2 seconds)

➔  Badge NFT minted! Token ID: 42

➔ Sarah receives notification:


"🎉 Your Web Development badge is verified!"

➔ Badge appears in Sarah's wallet immediately

#### STEP 4: Monitor Platform Revenue

➔ Click "Revenue" tab

➔ See revenue dashboard:

 Revenue Analytics	
Today: \$127.50 USDC	
This Week: \$890.00 USDC	
This Month: \$3,240.00 USDC	
Breakdown:	
• Featured Gigs: \$65.00 (130 × \$0.50)	
• Badge Verifications: \$50.00 (10 × \$5)	
• Subscriptions: \$119.88 (12 × \$9.99)	
• Pay-per-apply: \$2.50 (125 × \$0.02)	
[Export CSV] [View Transactions]	

➔ All revenue collected in admin wallet on Avalanche

#### STEP 5: Handle Disputes (Rare)

➔ User reports: "Employer won't release payment"

→ Dispute ticket created: #DIS-456

→ Admin reviews:

 Dispute Resolution
Gig: #123 "Build Landing Page"
Employer: John (0xDEF...)
Worker: Sarah (0xABC...)
Escrow: 0.5 AVAX (locked)
Worker claims: "I completed all tasks"
Deliverable: <a href="https://landing-page.com">https://landing-page.com</a>
Employer claims: "Missing mobile view"
<a href="#">[View Deliverable]</a> <a href="#">[Chat Thread]</a>
Decision:
<input type="radio"/> Release to Worker
<input type="radio"/> Refund to Employer
<input type="radio"/> Split 50/50
<a href="#">[Submit Decision]</a>

→ Admin reviews deliverable

→ Tests on mobile → Works fine 

→ Decision: Release to Worker

→ Admin calls smart contract (admin override):

```
const tx = await escrowContract.resolveDispute(  
  gigId,  
  RELEASE_TO_WORKER  
);
```

→ 0.5 AVAX sent to Sarah 

## STEP 6: Manage Community NFT Sales

→ Click "Community NFT" tab

→ Dashboard shows:

 Community NFT Management
Total Minted: 234 / 10,000
Revenue: \$11,466 (234 × \$49)
Recent Purchases:
• 0xAAA... - 2 hours ago
• 0xPPP... 5h

• 0xB... - 5 hours ago

#### NFT Holder Benefits:

- ✓ Unlimited applications
- ✓ Premium gigs access
- ✓ Community Discord role
- ✓ Early feature access

[\[Manage Benefits\]](#) [\[View Holders\]](#)

→ Admin can update benefits or mint promotional NFTs

### STEP 7: Platform Analytics

→ Click "Analytics" tab

→ See comprehensive metrics:

#### Platform Metrics

##### Users:

- Total: 1,234
- Active (30d): 567
- Community NFT holders: 234
- Monthly subscribers: 123

##### Gigs:

- Posted: 2,456
- Completed: 1,890
- Success rate: 77%

##### Badges:

- Total issued: 567
- Most popular: Web Development (234)

##### Revenue:

- Total: \$45,678 USDC
- Avg per user: \$37

[\[Export Report\]](#) [\[Share Dashboard\]](#)

→ Use data to make platform decisions

## SUBSCRIPTION MODELS EXPLAINED

### Option 1: Community NFT (VIP Pass) - \$49 One-Time

🎯 Target User: Serious freelancers who use platform regularly

## COMMUNITY NFT PURCHASE

STEP 1: User clicks "Buy Community NFT"

→ Payment page:

💎 Community NFT - Lifetime Access

Price: \$49 USDC (one-time payment)

What you get:

- ✓ Unlimited gig applications
- ✓ Access to premium gigs
- ✓ No monthly fees ever
- ✓ Community Discord access
- ✓ Voting rights on platform changes
- ✓ NFT badge in your wallet
- ✓ Early access to new features

This NFT is yours forever. Can't be transferred or sold (soulbound).

[Buy with USDC - \$49]

→ User clicks "Buy with USDC"

→ x402 processes payment (2 seconds)

→ Backend calls smart contract:

```
const tx = await communityNFT.mint(userAddress);
```

→ NFT minted to user's wallet ✓

→ User immediately gets all benefits

BENEFITS:

- ✓ One-time payment (no recurring fees)
- ✓ NFT ownership (visible in wallet)
- ✓ Community status
- ✓ Perfect for power users

REVENUE TO PLATFORM:

$\$49 \times 234 \text{ users} = \$11,466$  (one-time)

**Option 2: Monthly Subscription - \$9.99/month via x402**

## MONTHLY SUBSCRIPTION

### STEP 1: User clicks "Subscribe Monthly"

➔ Subscription page:

17

Monthly Subscription

Price: \$9.99 USDC/month

What you get:

✓

Unlimited applications this month

✓

Same features as Community NFT

✓

Auto-renews automatically

✓

Cancel anytime

How it works:

•

Pay \$9.99 USDC now via x402

•

Auto-charges every 30 days

•

Cancel from settings anytime

[Subscribe - \$9.99/month]

➔ User clicks "Subscribe"

➔ x402 creates recurring payment:

```
const subscription = await x402.createSubscription({  
  amount: "$9.99",  
  interval: "monthly",  
  userAddress: "0xABC..."  
});
```

➔ First payment: \$9.99 USDC charged ✓

➔ Subscription active!

➔ User sees "Subscribed ✓" badge

### STEP 2: Auto-Renewal (30 days later)

➔ x402 automatically attempts payment

➔ If user has \$9.99 USDC: ✓ Payment succeeds

➔ Subscription continues

➔ If user has insufficient USDC: ✗ Payment fails

➔ User receives email: "Payment failed"

➔ 3-day grace period

- 3 day grace period
- If still not paid: Account → Free tier
- User can resubscribe anytime

### STEP 3: Cancellation

- User clicks "Cancel Subscription"
- Confirmation modal:


Cancel Subscription?

You'll keep access until: Dec 31, 2025

No refunds for current period.

[Keep Subscription] [Cancel]

- User confirms cancellation
- x402 stops future payments ✓
- User keeps access until period ends

### BENEFITS:

- ✓ Flexible (cancel anytime)
- ✓ Lower upfront cost than Community NFT
- ✓ Perfect for testing the platform

### REVENUE TO PLATFORM:

$\$9.99 \times 123 \text{ subscribers} = \$1,228.77/\text{month (recurring)}$   
 $\$9.99 \times 123 \times 12 \text{ months} = \$14,745/\text{year}$


## Option 3: Pay-Per-Apply (Free Tier) - \$0.02 per application

## PAY-PER-APPLY MODEL

### STEP 1: User browses WITHOUT subscription

- ➡ Can see all gigs (free)
- ➡ Can connect wallet (free)
- ➡ Can view gig details (free)

➡ Wants to apply to gig:

 Build Landing Page

Payment: 0.5 AVAX

To apply to this gig:  
Pay \$0.02 USDC per application

OR

Subscribe for unlimited applications:

- Monthly: \$9.99/month
- Community NFT: \$49 (lifetime)

[Pay \$0.02] [Subscribe Instead]

### STEP 2: User chooses "Pay \$0.02"

- ➡ x402 payment modal appears
- ➡ User pays \$0.02 USDC
- ➡ Payment processed (2 seconds) ✅

➡ Application submitted!

### MATH:

- ➡ User applies to 10 gigs = \$0.20 total
- ➡ User applies to 50 gigs = \$1.00 total
- ➡ User applies to 500 gigs = \$10.00 total
- ➡ At this point, monthly sub is cheaper!
- ➡ Platform reminds: "💡 Subscribe for \$9.99 = unlimited!"

### BENEFITS:

- ✅ No commitment
- ✅ Only pay when you use it
- ✅ Perfect for casual users
- ✅ Can upgrade to subscription later

REVENUE TO PLATFORM:

$\$0.02 \times 5,000 \text{ applications/month} = \$100/\text{month}$



# COMPLETE API ENDPOINTS LIST

## Public Endpoints (No Authentication)

javascript

// ===== BROWSE GIGS =====

GET /api/gigs

Query params:

- status: OPEN | ASSIGNED | SUBMITTED | COMPLETED
- requiredBadge: string (e.g., "Web Development")
- featured: boolean
- limit: number (default: 20)
- skip: number (for pagination)

Response: { success: true, gigs: [...], total: number }

// ===== VIEW SINGLE GIG =====

GET /api/gigs/:gigId

Response: { success: true, gig: {...} }

// ===== VIEW USER BADGES =====

GET /api/badges/:address

Response: { success: true, badges: [...] }

// ===== BADGE TYPES LIST =====

GET /api/badges/types

Response: { success: true, badgeTypes: ["Web Development", ...] }

// ===== PLATFORM STATS =====

GET /api/stats

Response: {  
 success: true,  
 stats: {  
 totalGigs: number,  
 activeGigs: number,  
 completedGigs: number,  
 totalUsers: number,  
 totalBadgesIssued: number,  
 totalValueLocked: string (AVAX)  
 }  
}

// ===== HEALTH CHECK =====

GET /health

Response: { status: "healthy", timestamp: string }

## User Endpoints (Requires Wallet Connection)

javascript

// ===== CREATE GIG (FREE) =====

POST /api/gigs

Body: {

title: string,

description: string,

paymentAmount: string (AVAX),

requiredBadge: string,

deadline: number (days)

}

Response: { success: true, gigId: string }

// ===== FEATURE GIG (x402: \$0.50) =====

POST /api/gigs/featured

X-PAYMENT: \$0.50 USDC via x402

Body: { gigId: string }

Response: { success: true, message: "Gig featured for 24h" }

// ===== URGENT GIG BADGE (x402: \$1.00) =====

POST /api/gigs/urgent

X-PAYMENT: \$1.00 USDC via x402

Body: { gigId: string }

Response: { success: true, message: "Gig marked urgent" }

// ===== APPLY TO GIG =====

POST /api/gigs/:gigId/apply

X-PAYMENT: \$0.02 USDC via x402 (if no subscription)

Body: {

coverLetter: string,

estimatedTime: number (days)

}

Response: { success: true, applicationId: string }

// ===== UPDATE GIG STATUS =====

PUT /api/gigs/:gigId

Body: {

status: string,

worker: string (address),

txHash: string

}

Response: { success: true }

// ===== GET MY GIGS (as Employer) =====

GET /api/users/:address/gigs/employer

Response: { success: true, gigs: [...] }

// ===== GET MY GIGS (as Worker) =====

GET /api/users/:address/gigs/worker

```
Response: { success: true, gigs: [...] }
```

```
// ===== CREATE/UPDATE USER PROFILE =====
```

```
POST /api/users
```

```
Body: {
```

```
  address: string,
```

```
  displayName: string,
```

```
  bio: string,
```

```
  profileImage: string (URL)
```

```
}
```

```
Response: { success: true, message: "Profile updated" }
```

```
// ===== GET USER PROFILE =====
```

```
GET /api/users/:address
```

```
Response: {
```

```
  success: true,
```

```
  user: {
```

```
    address: string,
```

```
    displayName: string,
```

```
    bio: string,
```

```
    badges: [...],
```

```
    completedGigs: number,
```

```
    rating: number,
```

```
    joinedAt: string
```

```
  }
```

```
}
```

## Subscription Endpoints (x402 Payments)

javascript

```
// ===== SUBSCRIBE MONTHLY (x402: $9.99/month) =====  
POST /api/subscriptions/monthly  
X-PAYMENT: $9.99 USDC via x402 (recurring)  
Body: { userAddress: string }  
Response: {  
  success: true,  
  subscriptionId: string,  
  expiresAt: string,  
  autoRenew: true  
}  
  
// ===== CANCEL SUBSCRIPTION =====  
DELETE /api/subscriptions/:subscriptionId  
Response: { success: true, message: "Subscription cancelled" }  
  
// ===== CHECK SUBSCRIPTION STATUS =====  
GET /api/subscriptions/:address/status  
Response: {  
  success: true,  
  active: boolean,  
  type: "none" | "monthly" | "community_nft",  
  expiresAt: string | null  
}  
  
// ===== BUY COMMUNITY NFT (x402: $49) =====  
POST /api/community-nft/purchase  
X-PAYMENT: $49 USDC via x402  
Body: { userAddress: string }  
Response: {  
  success: true,  
  tokenId: number,  
  txHash: string,  
  message: "Community NFT minted"  
}  
  
// ===== CHECK COMMUNITY NFT OWNERSHIP =====  
GET /api/community-nft/:address/owns  
Response: {  
  success: true,  
  owns: boolean,  
  tokenId: number | null  
}
```

## Badge Verification Endpoints

javascript

```
// ===== REQUEST BADGE VERIFICATION (x402: $5) =====  
  
POST /api/badges/verify  
X-PAYMENT: $5.00 USDC via x402  
Body: {  
  userAddress: string,  
  skillName: string,  
  portfolioUrl: string,  
  githubUrl: string (optional),  
  linkedinUrl: string (optional)  
}  
Response: {  
  success: true,  
  verificationId: string,  
  message: "Verification request submitted. Review within 24h."  
}  
  
// ===== CHECK VERIFICATION STATUS =====  
  
GET /api/badges/verify/:verificationId  
Response: {  
  success: true,  
  status: "pending" | "approved" | "rejected",  
  submittedAt: string,  
  reviewedAt: string | null  
}
```

## Premium Features (x402 Micropayments)

javascript

```
// ===== ADVANCED SEARCH (x402: $0.10) =====  
POST /api/search/advanced  
X-PAYMENT: $0.10 USDC via x402  
Body: {  
  keywords: string[],  
  paymentRange: { min: number, max: number },  
  badges: string[],  
  location: string (optional),  
  aiMatch: boolean  
}  
Response: { success: true, gigs: [...], matchScore: number }  
  
// ===== ANALYTICS DASHBOARD (x402: $2/month) =====  
GET /api/analytics/earnings  
X-PAYMENT: $2.00 USDC via x402 (monthly access)  
Response: {  
  success: true,  
  earnings: {  
    thisWeek: number,  
    thisMonth: number,  
    allTime: number,  
    chart: [...]  
  }  
}  
  
// ===== AI GIG MATCHING (x402: $0.10) =====  
POST /api/ai/match  
X-PAYMENT: $0.10 USDC via x402  
Body: {  
  userAddress: string,  
  preferences: {...}  
}  
Response: {  
  success: true,  
  recommendations: [...],  
  matchScores: [...]  
}
```

## Admin Endpoints (Requires Admin Role)

javascript

// ===== MINT BADGE (Admin only) =====

POST /api/admin/badges/mint

Headers: { Authorization: "Bearer ADMIN\_TOKEN" }

Body: {

  userAddress: string,

  skillName: string,

  iconURI: string,

  level: "BEGINNER" | "INTERMEDIATE" | "EXPERT"

}

Response: {

  success: true,

  tokenId: number,

  txHash: string

}

// ===== REVIEW VERIFICATION REQUESTS =====

GET /api/admin/verifications/pending

Response: {

  success: true,

  verifications: [...]

}

POST /api/admin/verifications/:id/approve

Response: { success: true, badgeTokenId: number }

POST /api/admin/verifications/:id/reject

Body: { reason: string }

Response: { success: true }

// ===== VIEW REVENUE =====

GET /api/admin/revenue

Query params:

- period: "today" | "week" | "month" | "all"

Response: {

  success: true,

  revenue: {

    total: number,

    breakdown: {

      featuredGigs: number,

      badgeVerifications: number,

      subscriptions: number,

      payPerApply: number,

      communityNFT: number

  }

}

}

}

“  
DECODE PRIVATE

// ===== RESOLVE DISPUTE =====

POST /api/admin/disputes/:disputeId/resolve

Body: {

  decision: "release\_to\_worker" | "refund\_to\_employer" | "split\_50\_50"

}

Response: { success: true, txHash: string }

// ===== MINT COMMUNITY NFT (Promotional) =====

POST /api/admin/community-nft/mint

Body: {

  userAddress: string,

  reason: string (e.g., "Beta tester reward")

}

Response: { success: true, tokenId: number }

## Blockchain Interaction Endpoints

javascript

```
// ===== VERIFY ESCROW PAYMENT =====
```

```
POST /api/blockchain/verify-payment
```

```
Body: {
```

```
  txHash: string,
```

```
  gigId: string
```

```
}
```

```
Response: {
```

```
  success: true,
```

```
  blockchainGigId: number,
```

```
  paymentLocked: string (AVAX),
```

```
  message: "Payment verified"
```

```
}
```

```
// ===== GET GIG FROM BLOCKCHAIN =====
```

```
GET /api/blockchain/gigs/:blockchainGigId
```

```
Response: {
```

```
  success: true,
```

```
  gig: {
```

```
    employer: string,
```

```
    worker: string,
```

```
    paymentAmount: string (AVAX),
```

```
    status: string,
```

```
    createdAt: number,
```

```
    completedAt: number
```

```
  }
```

```
}
```

```
// ===== GET BADGE FROM BLOCKCHAIN =====
```

```
GET /api/blockchain/badges/:tokenId
```

```
Response: {
```

```
  success: true,
```

```
  badge: {
```

```
    skillName: string,
```

```
    iconURI: string,
```

```
    holder: string,
```

```
    issuedAt: number,
```

```
    level: string
```

```
  }
```

```
}
```

---

## COMMUNITY NFT BENEFITS EXPLAINED

### What is the Community NFT?






Think of it like a **lifetime gym membership**:

- Pay once (\$49)
- Use forever
- Can't be transferred or sold
- Proves you're a committed community member





## **Full Benefits List**

## COMMUNITY NFT HOLDER BENEFITS



### CORE BENEFITS:

-  Unlimited gig applications (no \$0.02 per apply fee)
-  Access to premium/exclusive gigs
  - ↳ Some employers only hire NFT holders
-  Priority in application queue
  - ↳ Your application shows first
-  No monthly fees (vs \$9.99/month subscription)
- ↳  NFT visible in wallet (status symbol)




### COMMUNITY ACCESS:

-  Private Discord server for NFT holders
-  Direct chat with platform team
-  Networking events (virtual/in-person)
- ↳  Job referral channel

### GOVERNANCE:

-  Voting rights on platform changes
  - ↳ Vote on new features
  - ↳ Vote on fee structures
  - ↳ Vote on dispute resolutions
- ↳  Proposals: Suggest new features

### EARLY ACCESS:

-  Beta test new features first
-  Early access to new gig categories
- ↳  Special promotions

### FINANCIAL PERKS:

-  Discounted badge verifications
  - ↳ \$3 instead of \$5 per badge
-  Free featured gig (1× per month)
  - ↳ \$0.50 value
- ↳  Revenue share (future):
  - ↳ NFT holders get % of platform revenue

### EXCLUSIVE FEATURES:

-  Custom profile badge " Community Member"
-  Access to enterprise/high-value gigs
  - ↳ Gigs \$500+ AVAX
- ↳  Priority support (24h response time)

solidity

*// SPDX-License-Identifier: MIT*

pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC721/ERC721.sol";

import "@openzeppelin/contracts/access/Ownable.sol";

contract CommunityNFT is ERC721, Ownable {

uint256 public constant MAX\_SUPPLY = 10000;

uint256 public nextTokenId;

uint256 public price = 49 \* 10\*\*6; *// \$49 in USDC (6 decimals)*

IERC20 public usdcToken; *// USDC contract address*

mapping(uint256 => uint256) public mintedAt;

mapping(address => bool) public hasMinted;

event CommunityNFTMinted(

uint256 indexed tokenId,

address indexed holder,

uint256 price

);

constructor(address \_usdcToken)

ERC721("ProjectX Community", "PXCOM")

Ownable(msg.sender)

{

usdcToken = IERC20(\_usdcToken);

}

*// User purchases Community NFT*

function mint() external returns (uint256) {

require(nextTokenId < MAX\_SUPPLY, "Max supply reached");

require(!hasMinted[msg.sender], "Already owns Community NFT");

*// Transfer USDC from user to platform*

require(

usdcToken.transferFrom(msg.sender, owner(), price),

"USDC payment failed"

);

uint256 tokenId = nextTokenId++;

\_safeMint(msg.sender, tokenId);

mintedAt[tokenId] = block.timestamp;

hasMinted[msg.sender] = true;

emit CommunityNFTMinted(tokenId, msg.sender, price);

```

    return tokenId;
}

// Check if address owns Community NFT
function isMember(address _address) external view returns (bool) {
    return balanceOf(_address) > 0;
}

// Make NFT non-transferable (soulbound)
function _update(
    address to,
    uint256 tokenId,
    address auth
) internal virtual override returns (address) {
    address from = _ownerOf(tokenId);

    // Allow minting (from = 0) but block transfers
    require(
        from == address(0) || to == address(0),
        "Community NFT is non-transferable"
    );

    return super._update(to, tokenId, auth);
}

// Admin can gift NFTs (for promotions)
function mintPromo(address _to) external onlyOwner returns (uint256) {
    require(nextTokenId < MAX_SUPPLY, "Max supply reached");
    require(!hasMinted[_to], "Already owns Community NFT");

    uint256 tokenId = nextTokenId++;
    _safeMint(_to, tokenId);

    mintedAt[tokenId] = block.timestamp;
    hasMinted[_to] = true;

    emit CommunityNFTMinted(tokenId, _to, 0); // Free promo
    return tokenId;
}

// Update price (owner only)
function updatePrice(uint256 _newPrice) external onlyOwner {
    price = _newPrice;
}
}

```

## Frontend: Community NFT Purchase Flow

javascript

*// Component: CommunityNFTPurchase.jsx*

```
import { useState } from 'react';
import { useAccount, useContract, useSigner } from 'wagmi';
import { parseUnits } from 'ethers';

export function CommunityNFTPurchase() {
  const { address } = useAccount();
  const { data: signer } = useSigner();
  const [isPurchasing, setIsPurchasing] = useState(false);

  const COMMUNITY_NFT_ADDRESS = import.meta.env.VITE_COMMUNITY_NFT_ADDRESS;
  const USDC_ADDRESS = import.meta.env.VITE_USDC_ADDRESS;

  const handlePurchase = async () => {
    if (!signer) return alert('Please connect wallet');

    setIsPurchasing(true);

    try {
      // Step 1: Approve USDC spending
      const usdcContract = new ethers.Contract(
        USDC_ADDRESS,
        ['function approve(address spender, uint256 amount) returns (bool)'],
        signer
      );

      console.log('Approving USDC...');
      const approveTx = await usdcContract.approve(
        COMMUNITY_NFT_ADDRESS,
        parseUnits('49', 6) // $49 USDC (6 decimals)
      );
      await approveTx.wait();
      console.log('USDC approved ✅');

      // Step 2: Mint Community NFT
      const nftContract = new ethers.Contract(
        COMMUNITY_NFT_ADDRESS,
        ['function mint() returns (uint256)'],
        signer
      );

      console.log('Minting Community NFT...');
      const mintTx = await nftContract.mint();
      const receipt = await mintTx.wait();

      // Extract tokenId from event
```

```

const event = receipt.logs.find(log =>
  log.topics[0] === ethers.id('CommunityNFTMinted(uint256,address,uint256)')
);
const tokenId = ethers.BigNumber.from(event.topics[1]).toString();

console.log('Community NFT minted! Token ID:', tokenId);

// Step 3: Show success
alert(🎉 Community NFT purchased! Token ID: ${tokenId});

// Reload page to show new benefits
window.location.reload();

} catch (error) {
  console.error('Purchase failed:', error);
  alert('Purchase failed: ' + error.message);
} finally {
  setIsPurchasing(false);
}
};

return (
  <div className="max-w-2xl mx-auto p-8 bg-white rounded-lg shadow-lg">
    <div className="text-center mb-8">
      <h2 className="text-3xl font-bold mb-2">💎 Community NFT</h2>
      <p className="text-xl text-purple-600 font-bold">$49 USDC - Lifetime Access</p>
    </div>

    <div className="space-y-4 mb-8">
      <BenefitItem icon="✅" text="Unlimited gig applications" />
      <BenefitItem icon="✅" text="Access to premium gigs" />
      <BenefitItem icon="✅" text="Community Discord access" />
      <BenefitItem icon="✅" text="Voting rights on platform changes" />
      <BenefitItem icon="✅" text="Early access to new features" />
      <BenefitItem icon="✅" text="Priority support (24h response)" />
      <BenefitItem icon="✅" text="Monthly free featured gig ($0.50 value)" />
    </div>

    <div className="bg-purple-50 p-6 rounded-lg mb-8">
      <h3 className="font-bold mb-2">💰 Compare:</h3>
      <div className="space-y-2">
        <div>Pay-per-apply: $0.02 × 500 apps = <span className="font-bold">$10.00</span></div>
        <div>Monthly sub: $9.99 × 12 months = <span className="font-bold">$119.88/year</span></div>
        <div className="text-green-600 font-bold">Community NFT: $49.00 FOREVER ✅</div>
      </div>
    </div>

    <button
      onClick={handlePurchase}
      disabled={isPurchasing || !address}
    >

```

```

disabled={isPurchasing || !address}
className="w-full bg-purple-600 hover:bg-purple-700 text-white py-4 rounded-lg
font-bold text-lg disabled:opacity-50 disabled:cursor-not-allowed"
>
{isPurchasing ? 'Processing...' : 'Buy Community NFT - $49 USDC'}
</button>

<p className="text-sm text-gray-500 text-center mt-4">
  * NFT is non-transferable (soulbound). Limited to 10,000 mints.
</p>
</div>
);
}

function BenefitItem({ icon, text }) {
  return (
    <div className="flex items-center gap-3">
      <span className="text-2xl">{icon}</span>
      <span className="text-lg">{text}</span>
    </div>
  );
}

```

## 🎯 COMPARISON: All Membership Options

MEMBERSHIP COMPARISON TABLE			
Feature	Free Tier	Monthly Sub	Community NFT
	(Pay-per-use)	(\$9.99/mo)	(\$49 one-time)
Browse Gigs	✓ Unlimited	✓ Unlimited	✓ Unlimited
View Details	✓ Unlimited	✓ Unlimited	✓ Unlimited
Apply to Gigs	\$0.02 each	✓ Unlimited	✓ Unlimited
Premium Gigs	✗ No access	✓ Yes	✓ Yes
Community Discord	✗ No	✗ No	✓ Yes
Voting Rights	✗ No	✗ No	✓ Yes
Early Access	✗ No	✗ No	✓ Yes
Monthly Free	✗ No	✗ No	✓ 1 featured gig
Featured Gig			(\$0.50 value)
Badge Discount	✗ \$5.00	✗ \$5.00	✓ \$3.00
Priority Support	✗ No	✗ No	✓ 24h response
NFT in Wallet	✗ No	✗ No	✓ Yes
Status Symbol	✗ No	✗ No	✓ "💎 Member"
Cost per year	Varies	\$119.88	\$49 (forever)
(100 applies)	\$2.00		
(500 applies)	\$10.00		
Break-even	N/A	5 months	Immediate if
		using often	

#### 🎯 RECOMMENDATION:

- ➡ Casual users (1-10 applies/month): Free Tier
- ➡ Regular users (50+ applies/month): Monthly Sub
- ➡ Power users (daily usage): Community NFT (best value)

## 🚀 QUICK SUMMARY

### As a User (Worker):

1. Connect wallet → Choose membership
2. Get skill badge verified (\$5)
3. Browse & apply to gigs
4. Complete work → Get paid in AVAX

### As an Employer:

1. Connect wallet
2. Post gig (FREE) or feature it (\$0.50)
3. Lock payment in escrow (AVAX)
4. Review applications → Assign worker
5. Approve work → Payment auto-released

#### **As Admin:**

1. Review badge verifications
2. Mint NFT badges on-chain
3. Monitor revenue dashboard
4. Resolve disputes (rare)
5. Manage Community NFT benefits

#### **Subscription Models:**

- **Community NFT:** \$49 one-time → Lifetime VIP
- **Monthly:** \$9.99/month → Flexible, cancel anytime
- **Pay-per-apply:** \$0.02/application → No commitment

#### **All Payments:**

- **Platform fees:** USDC via x402 (micropayments)
- **Gig payments:** AVAX via escrow (trustless)
- **Everything:** On Avalanche C-Chain (one network)

---

**That's the complete flow! Ready to build? 🚀 |**