

# Whop SSO + Product Gating Integration - COMPLETE

---

## Executive Summary

---

✅ **IMPLEMENTED:** Full Whop OAuth SSO integration with automatic membership sync and product gating

## What Was Implemented

1. **Whop OAuth Provider** - Custom NextAuth provider for Whop authentication
  2. **WAP Redirect Landing Page** - Seamless redirect from Whop's Web Application Platform
  3. **Automatic Membership Sync** - Real-time sync of user membership tier and status
  4. **Product Gating System** - Tier-based access control for features and content
  5. **Upgrade CTAs** - Beautiful upgrade prompts with feature listings
  6. **Example API Routes** - Demonstration of product gating in API endpoints
- 


## Authentication Flow

---

### Standard Whop OAuth Flow

1. User clicks "Sign in with Whop" on /auth/login  
↓
2. Redirected to Whop OAuth: https://data.whop.com/api/v3/oauth/authorize  
↓
3. User grants access on Whop  
↓
4. Whop redirects back with auth code  
↓
5. NextAuth exchanges code for access token  
↓
6. Fetch user profile from Whop API  
↓
7. Create/update user in database  
↓
8. Sync membership data from Whop  
↓
9. Redirect to /dashboard with active session

## WAP (Web Application Platform) Flow

1. User purchases **on** Whop  clicks "Open App"  

2. Redirected **to** /auth/whop-redirect?code=xxx  

3. Auto-triggers Whop OAuth sign-in  

4. Same flow as above (**steps** 5-9)  

5. Seamless landing **on** /dashboard

## Files Added/Modified

### New Files

1. **/app/auth/whop-redirect/page.tsx**
  - WAP landing page
  - Server component wrapper
2. **/app/auth/whop-redirect/whop-redirect-client.tsx**
  - Client-side OAuth handler
  - Loading states
  - Error handling
3. **/lib/tier-access.ts**
  - Tier hierarchy definitions
  - Feature access mapping
  - Server-side access checks
  - Upgrade message generation
4. **/components/upgrade-cta.tsx**
  - Reusable upgrade prompt component
  - Tier-specific feature lists
  - Whop redirect links
5. **/app/api/assessments/52-pitch/route.ts**
  - Example API route with product gating
  - Demonstrates tier access checks

### Modified Files

1. **/lib/auth-options.ts**
  - Added Whop OAuth provider
  - Enhanced JWT callback to sync memberships
  - Updated session callback for tier data
2. **/app/auth/login/login-client.tsx**
  - Added "Sign in with Whop" button
  - OAuth divider
  - Updated footer text

3. `.env`

- Added `WHOP_CLIENT_ID` placeholder
- Added `WHOP_CLIENT_SECRET` placeholder

## Configuration Required

### Step 1: Get Whop OAuth Credentials

1. Go to [Whop Developer Settings](https://dev.whop.com/settings) (<https://dev.whop.com/settings>)
2. Navigate to the OAuth section
3. Create a new OAuth app or select existing
4. Copy your **Client ID** and **Client Secret**

### Step 2: Configure Redirect URLs in Whop

Add these redirect URLs in your Whop OAuth app settings:

```
https://catchbarrels.app/api/auth/callback/whop
https://catchbarrels.app/auth/whop-redirect
```

### Step 3: Update `.env` File

Replace the placeholders in `/nextjs_space/.env` :

```
# Whop OAuth Credentials
WHOP_CLIENT_ID=your_actual_client_id_here
WHOP_CLIENT_SECRET=your_actual_client_secret_here
```





### Step 4: Configure WAP Redirect URL

In your Whop product settings:

1. Go to Product Settings → Web Application
2. Set App URL to: `https://catchbarrels.app/auth/whop-redirect`
3. Enable “Open app after purchase”

## Product Tiers & Access

### Tier Hierarchy

free (0)	 Basic access
athlete (1)	 \$49/mo - Unlimited uploads, drills, Coach Rick
pro (2)	 \$99/mo - Advanced biomechanics, assessments
elite (3)	 \$199/mo - Unlimited everything, live coaching

## Feature Access Map

```
const FEATURE_ACCESS = {
  // Free tier
  'basic_analysis': 'free',
  'view_dashboard': 'free',
  'upload_video': 'free',

  // Athlete tier
  'unlimited_uploads': 'athlete',
  'basic_drills': 'athlete',
  'progress_tracking': 'athlete',
  'coach_rick_basic': 'athlete',

  // Pro tier
  'advanced_biomechanics': 'pro',
  'kinematic_sequence': 'pro',
  'model_comparison': 'pro',
  '52_pitch_assessment': 'pro',

  // Elite tier
  'unlimited_assessments': 'elite',
  'priority_support': 'elite',
  'custom_training_plans': 'elite',
};
```



## Usage Examples

### Example 1: Server-Side API Route with Product Gating

```
import { checkTierAccess } from '@lib/tier-access';
import { NextRequest, NextResponse } from 'next/server';

export async function GET(request: NextRequest) {
  // Check if user has Pro tier access
  const access = await checkTierAccess('pro');

  if (!access.hasAccess) {
    return NextResponse.json(
      {
        error: 'Access denied',
        message: access.upgradeMessage,
        tier: access.tier,
        requiredTier: 'pro',
      },
      { status: 403 }
    );
  }

  // User has access - proceed with logic
  return NextResponse.json({ success: true });
}
```

## Example 2: Client-Side Feature Check

```
import { useSession } from 'next-auth/react';
import { hasFeatureAccess } from '@lib/tier-access';
import UpgradeCTA from '@components/upgrade-cta';

export default function MyFeature() {
  const { data: session } = useSession();
  const userTier = (session?.user as any)?.membershipTier || 'free';

  if (!hasFeatureAccess(userTier, 'advanced_biomechanics')) {
    return <UpgradeCTA tier="pro" feature="Advanced Biomechanics" />;
  }

  return <div>Protected feature content</div>;
}
```

## Example 3: Programmatic Tier Check

```
import { hasTierAccess } from '@lib/tier-access';

const userTier = 'athlete';
const canAccessPro = hasTierAccess(userTier, 'pro'); // false
const canAccessAthlete = hasTierAccess(userTier, 'athlete'); // true
```



## Testing the Integration

### Test Checklist

- [ ] **OAuth Login**
  1. Go to `/auth/login`
  2. Click “Sign in with Whop”
  3. Should redirect to Whop OAuth
  4. After granting access, should land on `/dashboard`
- [ ] **WAP Redirect**
  1. Go directly to `/auth/whop-redirect`
  2. Should auto-trigger OAuth
  3. Should show loading animation
  4. Should land on `/dashboard` when complete
- [ ] **Membership Sync**
  1. Sign in with Whop account that has active membership
  2. Check database: `membershipTier` should be updated
  3. Check session: `session.user.membershipTier` should match
- [ ] **Product Gating**
  1. Try accessing `/api/assessments/52-pitch` as free user
  2. Should receive 403 error with upgrade message
  3. Sign in with Pro tier
  4. Should receive 200 success

- [ ] **Upgrade CTA**

1. Add `<UpgradeCTA tier="pro" />` to any page
2. Should show upgrade prompt
3. Click “Upgrade Now” should open Whop purchase page

## Membership Sync Logic

### When Sync Happens

1. **On OAuth Sign-In** - JWT callback fetches all memberships
2. **On Webhook Event** - Real-time updates from Whop
3. **Manual Sync API** - `POST /api/whop/sync-membership`

### Sync Priority

If user has multiple memberships, highest tier wins:

```
elite (3) > pro (2) > athlete (1) > free (0)
```

### Product ID Mapping

Defined in `/lib/whop-client.ts`:

```
const productMapping = {
  'prod_kNyobCww4tc2p': 'athlete', // $49/mo or $417/yr
  'prod_04CB6y0IzNJLe': 'pro',    // $99/mo or $839/yr
  'prod_vCV6UQH3K18QZ': 'elite',  // $199/mo or $1,699/yr
  'prod_zH1wnZs0JKKfd': 'elite',  // 90-Day Transformation
};
```

## UI Components

### Whop Sign-In Button (Login Page)

```
<Button
  type="button"
  variant="outline"
  onClick={() => signIn('whop', { callbackUrl: '/dashboard' })}
  className="w-full border-[#F5A623]/30 text-white hover:bg-[#F5A623]/10"
>
  <svg className="w-5 h-5 mr-2" viewBox="0 0 24 24">
    <path d="M12 2L2 7v10c0 5.55 3.84 10.74 9 12 5.16-1.26 9-6.45 9-12V7l-10-5z"/>
  </svg>
  Sign in with Whop
</Button>
```

### WAP Redirect Loading State

- Animated BARRELS logo
- Pulsing dots

- Status messages
- Error handling

## Upgrade CTA Component

- Tier-specific pricing
- Feature bullet points
- “Upgrade Now” → opens Whop
- “Go Back” → returns to previous page
- Trust badges (cancel anytime, 7-day trial, secure)

---

## Troubleshooting

---

### Issue: “Invalid OAuth configuration”

#### Solution:

- Check `WHOP_CLIENT_ID` and `WHOP_CLIENT_SECRET` in `.env`
- Verify redirect URLs in Whop dashboard
- Restart Next.js dev server

### Issue: “User created but tier is ‘free’”

#### Solution:

- Check if user has active membership in Whop
- Verify product IDs in `/lib/whop-client.ts`
- Check webhook logs for sync errors

### Issue: “403 Forbidden on API routes”

#### Solution:

- User doesn’t have required tier
- Check `membershipStatus` is “active” in database
- Verify `membershipExpiresAt` hasn’t passed

### Issue: “Redirect loop on /auth/whop-redirect”

#### Solution:

- Clear browser cookies/session
- Check NextAuth configuration
- Verify OAuth code exchange is succeeding

---

## Database Schema

---

No new fields added - using existing Whop integration fields:

```

model User {
  whopUserId      String? @unique
  whopMembershipId String?
  membershipTier  String   @default("free")
  membershipStatus String   @default("inactive")
  membershipExpiresAt DateTime?
  lastWhopSync    DateTime?
}

```

## Deployment Notes

### Environment Variables

Ensure these are set in production:

```

NEXTAUTH_URL=https://catchbarrels.app
NEXTAUTH_SECRET=<your-secret>
WHOP_CLIENT_ID=<your-client-id>
WHOP_CLIENT_SECRET=<your-client-secret>
WHOP_API_KEY=<your-api-key>
WHOP_APP_ID=<your-app-id>

```

### Whop Dashboard Setup

#### 1. OAuth App

- Redirect URLs configured
- Scopes: openid profile email

#### 2. Web Application

- App URL: https://catchbarrels.app/auth/whop-redirect
- Auto-open enabled

#### 3. Webhooks

- Already configured at /api/webhooks/whop
- Events: payment.succeeded, membership.went\_valid, membership.went\_invalid



## ✓ Implementation Status

Feature	Status	Notes
Whop OAuth Provider	✓ Complete	Custom provider in auth-options.ts
WAP Redirect Page	✓ Complete	/auth/whop-redirect
Membership Sync	✓ Complete	JWT callback + webhook
Product Gating	✓ Complete	tier-access.ts utility
Upgrade CTAs	✓ Complete	Reusable component
Example API Routes	✓ Complete	52-pitch assessment
Login UI Update	✓ Complete	Whop sign-in button
Documentation	✓ Complete	This file

## 🎯 Next Steps for Coach Rick

### Immediate (Required for Production)

1. **Get Whop OAuth credentials** from dev.whop.com
2. **Update .env file** with real credentials
3. **Configure redirect URLs** in Whop dashboard
4. **Test OAuth flow** end-to-end
5. **Set WAP redirect URL** to `/auth/whop-redirect`

### Optional (Enhancements)

- Add product gating to more features (drill library, model comparison)
- Create dedicated pricing/upgrade page
- Add tier badge to user profile
- Implement “X features remaining” counters for free tier
- Add analytics tracking for upgrade conversions

## 📞 Support

For Whop-specific issues:

- [Whop Developer Docs](https://dev.whop.com) (https://dev.whop.com)
- [Whop Discord](https://discord.gg/whop) (https://discord.gg/whop)
- Whop support: support@whop.com

For implementation questions:

- Review this documentation
  - Check `/lib/tier-access.ts` for access control logic
  - Examine `/app/api/assessments/52-pitch/route.ts` for API example
- 

**Last Updated:** November 26, 2025

**Version:** 1.0

**Status:** Production Ready 