

Coach Admin Panel v1 Documentation

Overview

The Coach Admin Panel provides coaches with a **read-only control room** for deep analysis of athlete performance data. This is v1 - focused on viewing, analyzing, and understanding data patterns.

Access: `/admin`

Features

1. Coach Control Room Dashboard (`/admin`)

Purpose: High-level overview of roster and recent activity

Sections:



Summary Stats

- **Total Athletes:** Count of all players in the system
- **Active (7 Days):** Athletes who uploaded sessions in the last week
- **Avg Momentum Score:** Average score across last 30 sessions

Roster Snapshot

- Table/card view of all athletes
- Sorted by most recent activity
- Shows:
 - Player name
 - Level (Pro, College, HS, Youth)
 - Last session date
 - 30-day average momentum score
 - Trend arrow (up/flat/down based on last 3 sessions)
 - Total sessions in last 30 days
- **Clickable rows:** Navigate to `/admin/athlete/[id]` (placeholder - coming in v2)
- **Expand/collapse:** Show 5 or all athletes

Recent Sessions & Flags

- Last 30 sessions across all athletes
- Shows:
 - Player name
 - Date/time
 - Momentum Transfer Score
 - Weakest flow path (category)
- Flag indicators:
 -  **Low score:** < 60
 -  **Large drop:** 10+ points below player's 30-day avg

- **Clickable rows:** Deep dive into `/admin/session/[id]`
-

2. Session Detail View (`/admin/session/[id]`)

Purpose: Deep biomechanical analysis of a single session

Layout:

Header

- Player name, level, date/time
- Large momentum transfer score display
- GOATY band label (Elite, Advanced, Developing, Needs Work)

Biomechanical Categories (Left Column)

- Five-category breakdown from new scoring engine:
 - **Timing & Rhythm**
 - **Sequence & Braking**
 - **Stability & Balance**
 - **Directional Barrels**
 - **Posture & Spine**
- Each shows:
 - Subscore (0-100)
 - Visual progress bar
 - Tag: “Big strength”, “On track”, or “Needs attention”
- Highlights:
 - **Strongest** category (green border)
 - **Weakest** category (red border)

Flags & Critical Issues

- Displays system-generated flags:
 - Low overall score (< 60)
 - Broken sequence (< 50)
 - Poor timing/rhythm (< 50)
 - Stability issues (< 50)

Video Player

- Embedded video preview (placeholder in v1)
- Button: “Open Full Player in New Tab” → `/video/[id]`

AI Analyst Panel (Right Column)

- **Coach Rick - Session Analyst**
- Button: “Explain This Session”
- AI-generated analysis focusing on:
 1. What stood out mechanically?
 2. What’s the story the numbers are telling?
 3. What should I watch on video?
 4. What should we track in the next session?
 5. Specific coaching cues or drills

- Uses **coach-to-coach tone** (not kid-friendly)
- Technical and actionable

Legacy Scores

- Anchor / Engine / Whip scores for reference

Access Control

How It Works

User Roles

The `User` model has two fields for coach access:

- `role: String` ("player", "coach", "admin")
- `isCoach: Boolean` (quick boolean check)

Middleware Protection

All `/admin` routes are protected by `middleware.ts`:

```
if (pathname.startsWith('/admin')) {
  const isCoach = token.isCoach || false;

  if (!isCoach) {
    // Redirect to dashboard with error
    redirect('/dashboard?error=unauthorized');
  }
}
```

Session Data

The `isCoach` field is included in NextAuth JWT and session:

- JWT callback: `token.isCoach = user.isCoach`
- Session callback: `session.user.isCoach = token.isCoach`
- Available in middleware, server components, and client hooks

Marking a User as Coach

Method 1: Direct Database Update (Production)

Using **Prisma Studio** (recommended for production):

```
cd /home/ubuntu/barrels_pwa/nextjs_space
yarn prisma studio
```

1. Open Prisma Studio (opens in browser)
2. Navigate to `User` model
3. Find the user by email or username
4. Set:
 - `isCoach = true`
 - `role = "coach"`

5. Save changes

Method 2: SQL Query

Using PostgreSQL directly:

```
UPDATE "User"
SET "isCoach" = true, "role" = 'coach'
WHERE email = 'coach@example.com';
```

Method 3: Seed Script (Development)

Add to `scripts/seed.ts`:

```
// Create coach user
const coach = await prisma.user.upsert({
  where: { username: 'coach' },
  update: {},
  create: {
    username: 'coach',
    email: 'coach@catchbarrels.app',
    password: await bcrypt.hash('CoachPassword123!', 10),
    name: 'Coach Rick',
    isCoach: true,
    role: 'coach',
    profileComplete: true,
  },
});
```

Method 4: Email Whitelist (Future v2)

Configure in `.env`:

```
COACH_EMAILS="rick@catchbarrels.app,assistant@catchbarrels.app"
```

Then add logic in `auth-options.ts` to auto-set `isCoach=true` for these emails.

Data Access APIs

`lib/admin/getAdminDashboardData.ts`

Function: `getAdminDashboardData()`

Returns:

```
interface AdminDashboardData {
  rosterSummary: AdminRosterAthlete[];
  recentSessions: AdminRecentSession[];
  totalAthletes: number;
  activeLast7Days: number;
  avgMomentumScore: number;
}
```

Performance:

- Fetches last 30 days of data
- Calculates 30-day averages per player
- Computes trend arrows from last 3 sessions
- Flags low scores and large drops

Usage:

```
import { getAdminDashboardData } from '@lib/admin/getAdminDashboardData';

const data = await getAdminDashboardData();
```

lib/admin/getSessionDetail.ts

Function: getSessionDetail(videoId: string)

Returns:

```
interface AdminSessionDetail {
  // Session info
  id: string;
  videoId: string;
  playerId: string;
  playerName: string;
  playerLevel: string | null;
  date: Date;

  // Scores
  momentumScore: number;
  goatyBand: number | null;

  // Category scores
  timing: number;
  sequence: number;
  stability: number;
  directional: number;
  posture: number;

  // Analysis
  weakestCategory: string;
  weakestScore: number;
  strongestCategory: string;
  strongestScore: number;

  // Flags
  flags: string[];

  // Video
  videoUrl: string | null;

  // Raw data
  rawScoringData: any;
}
```

Usage:

```
import { getSessionDetail } from '@lib/admin/getSessionDetail';

const session = await getSessionDetail('video-id-123');
```

AI Analyst Integration

API Endpoint

Route: POST /api/coach-rick/analyze-session

Authentication: Requires coach role

Request Body:

```
{
  "sessionId": "video-123",
  "playerName": "John Doe",
  "playerLevel": "High School (13-18)",
  "momentumScore": 72,
  "categoryScores": {
    "timing": 68,
    "sequence": 54,
    "stability": 75,
    "directional": 80,
    "posture": 71
  },
  "flags": ["Broken sequence"],
  "context": "admin_view"
}
```

Response:

```
{
  "analysis": "Mechanically, the big story here is...",
  "sessionId": "video-123",
  "timestamp": "2025-11-26T16:30:00Z"
}
```

System Prompt

The AI uses a **coach-to-coach** system prompt:

You are Coach Rick, an elite hitting strategist.
Analyzing **for** a coach, not a player.

Be:

- Technical and detailed
- Strategic (focus on "what to watch on video")
- Actionable (specific coaching cues)
- Story-focused ("what the numbers tell me")

Provide analysis covering:

1. What stood out mechanically?
2. What's the story the numbers are telling?
3. What should I watch on video?
4. What should we track in the next session?
5. Specific coaching cues or drills

Model: gpt-4o-mini via Abacus AI

Temperature: 0.7

Max tokens: 1000

Navigation

For Coach Users

The "Coach Admin" link appears in:


- **Hamburger menu** (mobile & desktop)
- Styled with gold color (`text-barrels-gold`)
- Only visible when `session.user.isCoach === true`

Header Component

File: `components/layout/BarrelsHeader.tsx`

Logic:

```
const isCoach = (session?.user as any)?.isCoach || false;

{isCoach && (
  <Link href="/admin">
     Coach Admin
  </Link>
)}
```

Future Enhancements (v2+)

Planned Features

`/admin/athlete/[id]` - **Athlete Deep Dive**

- All sessions for a single athlete
- Progress charts over time
- Video + metrics + notes for each session

- Export data (CSV, PDF reports)

Edit Capabilities

- Add notes to sessions
- Flag specific swings for review
- Create custom drill assignments
- Send coaching feedback to players

Advanced Analytics

- Cohort analysis (level, age group)
- Drill effectiveness tracking
- Comparative analysis (player vs. player)
- Model swing overlay comparisons

Bulk Operations

- Multi-session comparison
- Batch export
- Group messaging

Coach Collaboration

- Multi-coach access
- Shared notes
- Role-based permissions (head coach, assistant)

Troubleshooting

Issue: “Unauthorized” error when accessing /admin

Cause: User's `isCoach` field is `false`

Fix:

1. Check user in database:

```
bash
yarn prisma studio
```
2. Verify `isCoach` and `role` fields
3. Set `isCoach = true` and `role = "coach"`
4. Have user log out and log back in (to refresh session)

Issue: AI Analyst not working

Cause: Missing or invalid `ABACUSAI_API_KEY`

Fix:

1. Check `.env` file:

```
bash
grep ABACUSAI_API_KEY .env
```
2. Ensure key is present and valid
3. Restart dev server:

```
bash
yarn dev
```

Issue: Roster shows no athletes

Cause: Database has no users with `role = "player"`

Fix:

1. Verify users in database:

```
bash
```

```
yarn prisma studio
```

2. Ensure test users have `role = "player"`

3. Run seed script if needed:

```
bash
```

```
yarn prisma db seed
```

Issue: Session detail page shows 404

Cause: Video ID doesn't exist or is invalid

Fix:

1. Check video exists:

```
sql
```

```
SELECT id, "userId", "momentumTransferScore"
```

```
FROM "Video"
```

```
WHERE id = 'video-id-here';
```

2. Verify video has scoring data (`newScoringBreakdown`)
3. Navigate from dashboard (don't type URL manually)

Security Notes

Data Access

- Admin APIs only accessible to authenticated coaches
- All routes protected by middleware
- Session data fetched server-side (no client exposure)

Role Validation

- Checked in:
 - Middleware (route-level)
 - API routes (endpoint-level)
 - Server components (page-level)
- Never rely on client-side checks alone

PII Handling

- Email addresses only shown to coaches
 - Player data isolated by coach access
 - No cross-organization data leakage
-

Performance Considerations

Dashboard Query Optimization

- Fetches 30 days of data (configurable)
- Limits to last 30 sessions for recent feed
- Roster collapses to 5 by default (expand on demand)

Caching Strategy (Future)

- Dashboard stats: Cache for 5 minutes
- Roster data: Cache for 10 minutes
- Session detail: No cache (always fresh)

Pagination (Future)

- Recent sessions: Paginate after 50+ sessions
 - Roster: Virtual scrolling for 100+ athletes
-

API Documentation

Admin Dashboard API

Not exposed as REST API - server-side only

Usage in Server Components:

```
import { getAdminDashboardData } from '@lib/admin/getAdminDashboardData';  
  
const data = await getAdminDashboardData();
```

Coach Rick Analyze Session API

Endpoint: POST /api/coach-rick/analyze-session

Auth: Requires coach session

Rate Limit: None (v1)

Timeout: 30 seconds

Summary

What v1 Provides

- ✓ **Coach-only access** with role-based security
- ✓ **Dashboard overview** with roster and recent sessions
- ✓ **Session deep dive** with biomechanical breakdown
- ✓ **AI Analyst** for coaching insights
- ✓ **Flag system** for low scores and drops
- ✓ **Navigation** via hamburger menu

What v1 Does NOT Provide

- ✗ **Athlete detail pages** (`/admin/athlete/[id]`)
- ✗ **Editing capabilities** (notes, flags, assignments)
- ✗ **Bulk operations** (multi-session compare, export)
- ✗ **Progress charts** (time-series analytics)
- ✗ **Coach collaboration** (multi-coach, shared notes)

Next Steps

1. **Mark yourself as coach** in the database
2. **Log in** and access `/admin`
3. **Review athlete roster** and recent sessions
4. **Click into a session** to see deep dive
5. **Use AI Analyst** to get coaching insights

Version: 1.0

Last Updated: November 26, 2025

Status: Production Ready

Access: <https://catchbarrels.app/admin>