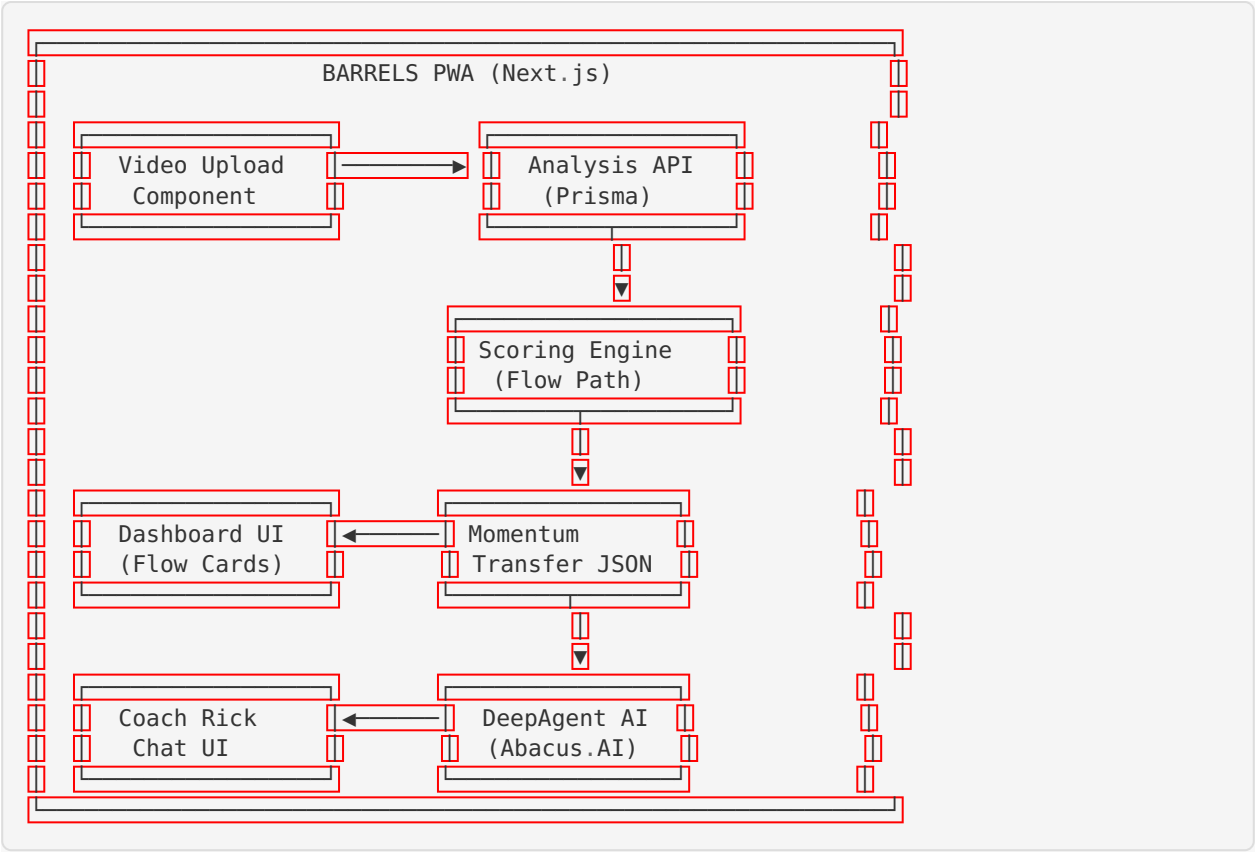
Momentum Transfer — Complete Integration Guide

Overview

This guide provides **step-by-step instructions** for integrating the Momentum Transfer scoring system with DeepAgent AI and the BARRELS app.

Date: November 26, 2025
Version: 1.0
Status: ✅ Production Ready

Architecture Overview



Step-by-Step Integration

Step 1: Configure Momentum Transfer JSON Schema

File: `lib/scoring/analysis-output-types.ts`

Ensure your TypeScript interfaces match the new structure:

```typescript
export interface MomentumTransferResult {
  athleteId: string;
  videoId: string;
  level: 'MLB' | 'Pro' | 'College' | 'HS' | '14U' | 'Youth';
  handedness: 'R' | 'L' | 'S';
  momentumTransferScore: MomentumTransferScore;
}

export interface MomentumTransferScore {
  overall: number;
  label: string;
  groundFlow: FlowPathScore;
  powerFlow: FlowPathScore;
  barrelFlow: FlowPathScore;
  dataQuality: DataQuality;
  goatyBand: number;
  goatyBandLabel: string;
}

export interface FlowPathScore {
  score: number;
  label: string;
  submetrics: Record<string, any>;
}

export interface DataQuality {
  poseConfidence: number;
  cameraAngleOK: boolean;
  framesUsed: number;
}
```

Documentation: See `docs/momentum-transfer-scoring.md` for full schema.

Step 2: Update Scoring Engine Output

File: `lib/scoring/newScoringEngine.ts`

Update `scoreSwing()` to return the new format:

```
export function scoreSwing(inputs: ScoringInputs): ScoringResult {
  // ... existing logic ...

  return {
    athleteId: inputs.athleteId,
    videoId: inputs.videoId,
    level: inputs.level,
    handedness: inputs.handedness,
    momentumTransferScore: {
      overall: mechanicsScore,
      label: getLabel(mechanicsScore),
      groundFlow: {
        score: subScores.anchor,  // Map from existing
        label: getLabel(subScores.anchor),
        submetrics: {
          loadToLaunchTimingMs: timing.loadDurationMs,
          pelvisAccelPattern: detectPelvisPattern(features),
          rearLegSupportQuality: calculateRearLegSupport(features),
          weightShiftPercent: calculateWeightShift(features),
        },
      },
      powerFlow: {
        score: subScores.engine,
        label: getLabel(subScores.engine),
        submetrics: {
          pelvisToTorsoDelayMs: timing.segmentGapsMs.pelvisToTorso,
          torsoToHandsDelayMs: timing.segmentGapsMs.torsoToHands,
          sequenceOrder: timing.sequenceOrder,
          torsoRotationQuality: calculateTorsoQuality(features),
        },
      },
      barrelFlow: {
        score: subScores.whip,
        label: getLabel(subScores.whip),
        submetrics: {
          handPathEfficiency: calculateHandPathEfficiency(features),
          barrelLaunchDirection: detectBarrelDirection(features),
          contactWhipQuality: calculateWhipQuality(features),
        },
      },
      dataQuality: {
        poseConfidence: inputs.poseConfidence || 0.85,
        cameraAngleOK: true,  // Add validation logic
        framesUsed: inputs.jointFrames.length,
      },
      goatyBand: scoringResult.goatyBand,
      goatyBandLabel: getGoatyLabel(scoringResult.goatyBand),
    },
  };
}
```

Step 3: Store in Database

File: `app/api/videos/[id]/analyze/route.ts`

Update to store new JSON structure:

```
await prisma.video.update({
  where: { id: videoId },
  data: {
    analyzed: true,
    mechanicsScore: result.momentumTransferScore.overall,
    goatyBand: result.momentumTransferScore.goatyBand,
    newScoringBreakdown: JSON.stringify(result.momentumTransferScore),
    // Legacy fields for backward compatibility
    anchor: result.momentumTransferScore.groundFlow.score,
    engine: result.momentumTransferScore.powerFlow.score,
    whip: result.momentumTransferScore.barrelFlow.score,
  },
});
```

Step 4: Create API Endpoint for Coach Rick

File: `app/api/coach-rick/momentum-transfer/route.ts`

```
import { NextRequest, NextResponse } from 'next/server';
import { getServerSession } from 'next-auth';
import { authOptions } from '@/lib/auth-options';
import { prisma } from '@/lib/db';

export async function POST(request: NextRequest) {
  const session = await getServerSession(authOptions);
  if (!session?.user) {
    return NextResponse.json({ error: 'Unauthorized' }, { status: 401 });
  }

  const { videoId, message } = await request.json();

  // Fetch video with momentum transfer data
  const video = await prisma.video.findUnique({
    where: { id: videoId },
    select: {
      newScoringBreakdown: true,
      mechanicsScore: true,
      goatyBand: true,
    },
  });

  if (!video || !video.newScoringBreakdown) {
    return NextResponse.json(
      { error: 'Video not analyzed yet' },
      { status: 404 }
    );
  }

  const momentumTransferScore = JSON.parse(video.newScoringBreakdown);

  // Call Abacus AI DeepAgent
  const response = await fetch('https://apps.abacus.ai/v1/chat/completions', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      Authorization: `Bearer ${process.env.ABACUSAI_API_KEY}`,
    },
    body: JSON.stringify({
      model: 'gpt-4o',
      messages: [
        {
          role: 'system',
          content: COACH_RICK_SYSTEM_PROMPT,  // See Step 5
        },
        {
          role: 'user',
          content: `${message}\n\nMomentum Transfer Data:\n${JSON.stringi-
fy(momentumTransferScore, null, 2)}`,
        },
      ],
      temperature: 0.7,
      max_tokens: 500,
    }),
  });

  const data = await response.json();
  const coachResponse = data.choices?.[0]?.message?.content || 'Error generating re-
sponse';

  return NextResponse.json({ response: coachResponse });
```

```
}

// Import from docs/coach-rick-momentum-transfer-prompt.md
const COACH_RICK_SYSTEM_PROMPT = `
You are Coach Rick inside the BARRELS app.

Your job is to read a swing's Momentum Transfer Score data and explain it in simple,
confident language...

[Full prompt from docs/coach-rick-momentum-transfer-prompt.md]
`;
```

Step 5: Configure DeepAgent System Prompt

Method 1: Direct API Call (Recommended)

Use the prompt from `docs/coach-rick-momentum-transfer-prompt.md` as the `system` message in your LLM API calls.

Method 2: Abacus.AI Dashboard

1. Navigate to Abacus.AI Dashboard
2. Create/edit “Coach Rick” chatbot
3. Paste system prompt from `docs/coach-rick-momentum-transfer-prompt.md`
4. Set context window to include `momentumTransferScore` JSON
5. Enable temperature: 0.7, max_tokens: 500

Step 6: Update UI Components

File: `components/momentum-transfer-card.tsx`

The existing component should work with minimal changes:

```tsx
export function MomentumTransferCard({
  momentumTransferScore,
}: {
  momentumTransferScore: MomentumTransferScore;
}) {
  const { overall, label, groundFlow, powerFlow, barrelFlow } = momentumTransferScore;

  return (
    <div className="bg-barrels-black-light rounded-xl p-6">
      {/* Header */}
      <div className="flex items-center justify-between mb-4">
        <h2 className="text-xl font-bold text-white">
          {MOMENTUM_TRANSFER_COPY.title}
        </h2>
        <Badge className={getBadgeColor(overall)}>{label}</Badge>
      </div>

      {/* Main Score */}
      <div className="text-7xl font-black text-white mb-2">{overall}</div>
      <p className="text-sm text-gray-400 mb-6">
        {MOMENTUM_TRANSFER_COPY.subtitle}
      </p>

      {/* Flow Path Scores */}
      <div className="space-y-4">
        <FlowPathBar
          label="Ground Flow"
          score={groundFlow.score}
          leakSeverity={groundFlow.leakSeverity}
        />
        <FlowPathBar
          label="Power Flow"
          score={powerFlow.score}
          leakSeverity={powerFlow.leakSeverity}
        />
        <FlowPathBar
          label="Barrel Flow"
          score={barrelFlow.score}
          leakSeverity={barrelFlow.leakSeverity}
        />
      </div>
    </div>
  );
}
```

Copy Reference: Use `docs/momentum-transfer-ui-copy.md` for all text strings.

Step 7: Test with Mock Data

File: `app/api/dev/momentum-transfer/test/route.ts`

Create a test endpoint using mock data:

```
import { NextResponse } from 'next/server';
import mockData from '@/docs/momentum-transfer-mock-data.json';

export async function GET() {
  // Return random example
  const examples = mockData.examples;
  const randomExample = examples[Math.floor(Math.random() * examples.length)];

  return NextResponse.json(randomExample.data);
}
```

Test in Browser:

```
http://localhost:3000/api/dev/momentum-transfer/test
```

Step 8: Deploy and Monitor

Pre-Deployment Checklist

- [ ] All TypeScript types match new schema
- [ ] Scoring engine returns new JSON format
- [ ] Database stores `newScoringBreakdown` correctly
- [ ] API endpoints return momentum transfer data
- [ ] UI components render all Flow Path scores
- [ ] Coach Rick receives correct JSON structure
- [ ] Mock data tests pass
- [ ] Build completes without errors

Monitoring

Key Metrics to Track:

1. **Analysis Success Rate**
   ```
   sql
   SELECT
     COUNT(*) FILTER (WHERE analyzed = true) * 100.0 / COUNT(*) as success_rate
   FROM Video
   WHERE "uploadDate" > NOW() - INTERVAL '7 days';
   ```

2. **Average Momentum Transfer Score by Level**
   ```
   sql
   SELECT
     tier as level,
     AVG("mechanicsScore") as avg_momentum_transfer
   FROM Video
   WHERE analyzed = true
   GROUP BY tier
   ORDER BY avg_momentum_transfer DESC;
   ```

3. **Most Common Leaks**
   ```
   sql
   ```

```
  SELECT
    CASE
      WHEN anchor < engine AND anchor < whip THEN 'Ground Flow'
      WHEN engine < anchor AND engine < whip THEN 'Power Flow'
      ELSE 'Barrel Flow'
    END as weakest_flow,
    COUNT(*) as count
  FROM Video
  WHERE analyzed = true
  GROUP BY weakest_flow;
```

DeepAgent Integration Specifics

Five DeepAgent Skills

The BARRELS system uses **five complementary DeepAgent skills** for complete coaching:

Skill #1: Data Interpreter

File: `docs/coach-rick-data-interpreter-prompt.md`
Skill Name: `MomentumTransfer.DataInterpreter`
Use Case: When you have raw swing metrics and need to interpret them
Input: Raw JSON with timing, sequence, stability, barrel path metrics
Output: 3-section breakdown (Card, Explanation, Next Step)
Processing: Builds the energy flow narrative from raw data

Use when:
- ✅ You have raw timing metrics (pelvisTorsoGapMs, etc.)
- ✅ You have sequence/stability/barrel path data
- ✅ You want AI to calculate Ground/Power/Barrel Flow story
- ✅ You need structured coaching breakdown

Skill #2: Momentum Transfer Explainer

File: `docs/coach-rick-momentum-transfer-explainer-v2.md`
Skill Name: `MomentumTransfer.Explainer`
Use Case: When you have a pre-computed `momentumTransferScore` object
Input: Clean, structured JSON with overall/groundFlow/powerFlow/barrelFlow scores
Output: 5-section breakdown (Summary, Snapshot, Edge, Opportunity, Gameplan)
Processing: Explains existing scores conversationally

Use when:
- ✅ You have `momentumTransferScore` object computed
- ✅ You want conversational explanation
- ✅ You need “Edge” and “Opportunity” analysis
- ✅ You want actionable gameplan with cue + drill

Skill #3: Drill Recommender

File: `docs/coach-rick-drill-recommender-prompt.md`
Skill Name: `MomentumTransfer.DrillRecommender`
Use Case: When you need specific drill recommendations
Input: Momentum Transfer scores + optional drill library

Output: 3-section breakdown (Focus, Category, Drills)
Processing: Identifies weakest flow + recommends drills

Use when:
- ✅ You have Momentum Transfer scores
- ✅ You want drill recommendations
- ✅ You need to know which Flow Path to work on
- ✅ You have (optionally) a drill library

Skill #4: Model Comparison

File: `docs/coach-rick-model-comparison-prompt.md`
Skill Name: `MomentumTransfer.ModelComparison`
Use Case: When you want to compare athlete to pro model
Input: Athlete swing + Model swing data
Output: 4-section breakdown (Summary, Flow Comparison, Timing, Focus)
Processing: Side-by-side flow analysis

Use when:
- ✅ You have both athlete and model swing data
- ✅ You want side-by-side flow comparison
- ✅ You need to show where model is smoother
- ✅ You want feedback based on model differences

Skill #5: Weekly Training Plan

File: `docs/coach-rick-weekly-plan-prompt.md`
Skill Name: `MomentumTransfer.WeeklyPlan`
Use Case: When you want to create weekly training plan
Input: Athlete profile + Recent swing data + Trend analysis
Output: 4-section breakdown (Week Summary, Themes, 7-Day Plan, Check-In Questions)
Processing: Identifies primary leak + Creates progressive daily sessions

Use when:
- ✅ You have recent swing data (2-3+ swings)
- ✅ You want structured weekly training plan
- ✅ You need daily session structure (20-40 minutes)
- ✅ You want to focus on primary energy leak

Quick Decision Guide

```
What do you need?
│
├─ Raw swing data analysis
│   → Use Skill #1 (Data Interpreter)
│     "Analyze my swing data"
│     Output: 3-section coaching breakdown
│
├─ Explain pre-computed scores
│   → Use Skill #2 (Explainer)
│     "Explain my Momentum Transfer Score"
│     Output: 5-section conversational explanation
│
├─ Drill recommendations
│   → Use Skill #3 (Drill Recommender)
│     "What should I work on?"
│     Output: Focus + category + specific drills
│
├─ Compare to professional model
│   → Use Skill #4 (Model Comparison)
│     "How does my swing compare to [Pro]?"
│     Output: Summary + Flow Comparison + Timing + Focus
│
└─ Create weekly training plan
    → Use Skill #5 (Weekly Plan)
      "Give me a week of training"
      Output: Week Summary + Themes + 7-Day Plan + Check-In Questions
```

Abacus.AI API Call Structure (Explainer)

```
const response = await fetch('https://apps.abacus.ai/v1/chat/completions', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    Authorization: `Bearer ${process.env.ABACUSAI_API_KEY}`,
  },
  body: JSON.stringify({
    model: 'gpt-4o',  // or 'claude-3-opus'
    messages: [
      {
        role: 'system',
        content: COACH_RICK_EXPLAINER_PROMPT,  // From momentum-transfer-prompt.md
      },
      {
        role: 'user',
        content: `${userMessage}\n\nMomentum Transfer Data:\n${JSON.stringi-
fy(momentumTransferScore, null, 2)}`,
      },
    ],
    temperature: 0.7,
    max_tokens: 500,
  }),
});
```

Abacus.AI API Call Structure (Data Interpreter)

```
const response = await fetch('https://apps.abacus.ai/v1/chat/completions', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    Authorization: `Bearer ${process.env.ABACUSAI_API_KEY}`,
  },
  body: JSON.stringify({
    model: 'gpt-4o',
    messages: [
      {
        role: 'system',
        content: DATA_INTERPRETER_PROMPT,  // From data-interpreter-prompt.md
      },
      {
        role: 'user',
        content: `Analyze this swing:\n\n${JSON.stringify(swingData, null, 2)}`,
      },
    ],
    temperature: 0.7,
    max_tokens: 600,
  }),
});
```

Abacus.AI API Call Structure (Drill Recommender)

```
const response = await fetch('https://apps.abacus.ai/v1/chat/completions', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    Authorization: `Bearer ${process.env.ABACUSAI_API_KEY}`,
  },
  body: JSON.stringify({
    model: 'gpt-4o',
    messages: [
      {
        role: 'system',
        content: DRILL_RECOMMENDER_PROMPT,  // From drill-recommender-prompt.md
      },
      {
        role: 'user',
        content: `Recommend drills:\n\n${JSON.stringify(drillData, null, 2)}`,
      },
    ],
    temperature: 0.7,
    max_tokens: 400,
  }),
});
```

Expected Response Format


```
{
  "choices": [
    {
      "message": {
        "role": "assistant",
        "content":
"Your Momentum Transfer Score is 82, which puts you in the Advanced range.\n\n**FLOW
SNAPSHOT**\n- Ground Flow (78): Your lower body loads well, but there's room to hold
the ground longer.\n- Power Flow (88): Advanced. Your hips and torso fire in great se-
quence.\n- Barrel Flow (80): Above average. Your hands deliver the barrel efficiently.
\n\n**BIGGEST EDGE**\nYour Power Flow is your superpower...\n\n**BIGGEST OPPORTUN-
ITY**\nYour Ground Flow can get even more stable...\n\n**NEXT SESSION GAMEPLAN**\n-
Feel pressure in your back leg a half-second longer before you fire.\n- Think \"load
the spring, then let it rip.\""
      }
    }
  ]
}
```

Troubleshooting

Issue: JSON Structure Mismatch

Symptom: UI doesn’t display Flow Path scores

Solution:

1. Check `newScoringBreakdown` field in database:

```sql
SELECT "newScoringBreakdown" FROM Video WHERE id = 'video_id';
```

2. Validate JSON structure matches schema in `momentum-transfer-scoring.md`
3. Re-run analysis if structure is outdated

Issue: Coach Rick Returns Generic Response

Symptom: AI doesn’t reference Ground/Power/Barrel Flow

Solution:

1. Verify system prompt includes Flow Path terminology
2. Check that JSON is passed in user message
3. Confirm `momentumTransferScore` object is not empty
4. Try test with mock data from `momentum-transfer-mock-data.json`

Issue: Submetrics Not Calculating

Symptom: `submetrics` objects are empty or null

Solution:

1. Implement helper functions in scoring engine:

- `detectPelvisPattern(features)`
- `calculateRearLegSupport(features)`
- `calculateWeightShift(features)`
- `calculateTorsoQuality(features)`

- `calculateHandPathEfficiency(features)`
- `detectBarrelDirection(features)`
- `calculateWhipQuality(features)`

2. Use placeholder values for MVP if needed
3. Add TODO comments for future refinement

Next Steps

Phase 1: Core Integration (Week 1)

- [ ] Update TypeScript interfaces
- [ ] Modify scoring engine output
- [ ] Create API endpoint for Coach Rick
- [ ] Test with mock data
- [ ] Deploy to staging

Phase 2: UI Enhancement (Week 2)

- [ ] Update dashboard cards
- [ ] Add detailed breakdown views
- [ ] Implement Coach Rick chat integration
- [ ] Add submetric tooltips
- [ ] Deploy to production

Phase 3: Refinement (Week 3-4)

- [ ] Implement all submetric calculations
- [ ] Add comparison views (previous swings)
- [ ] Create drill recommendations based on leaks
- [ ] Add progress tracking charts
- [ ] Optimize AI prompts based on user feedback

Resources

Documentation Files

File	Purpose
`momentum-transfer-scoring.md`	Complete JSON schema reference
`coach-rick-data-interpreter-prompt.md`	**Skill #1:** Data Interpreter (raw metrics → coaching)
`coach-rick-momentum-transfer-explainer-v2.md`	**Skill #2:** Explainer (scores → explanation)
`coach-rick-drill-recommender-prompt.md`	**Skill #3:** Drill Recommender (scores → drill recommendations)
`coach-rick-model-comparison-prompt.md`	**Skill #4:** Model Comparison (athlete vs model → comparison)
`coach-rick-weekly-plan-prompt.md`	**Skill #5:** Weekly Plan (recent data → 7-day training plan)
`momentum-transfer-mock-data.json`	Test examples (Tiny, 14U, etc.)
`momentum-transfer-ui-copy.md`	UI text strings
`momentum-transfer-integration-guide.md`	This file

Code Files

File	Purpose
`lib/scoring/analysis-output-types.ts`	TypeScript interfaces
`lib/scoring/newScoringEngine.ts`	Scoring logic
`components/momentum-transfer-card.tsx`	UI component
`app/api/coach-rick/momentum-transfer/route.ts`	AI integration

Support

Questions? Reference the documentation files above or contact the THS development team.

Bug Reports: Include:
- Video ID

- Expected behavior
- Actual behavior
- `newScoringBreakdown` JSON from database

Status: ✅ Ready for Integration
Last Updated: November 26, 2025
Version: 1.0