Table of Contents

# Video Analytics - Executive Summary

## Problem Statement

Currently, video interactions are tracked in the Databricks storage layer as individual events (play, pause, resume, etc.). The challenge is to calculate meaningful metrics like **Total Watch Time** per user and video from these atomic events to understand video engagement.

**Complexity:** - Each interaction = 1 event row - User journey must be reconstructed manually - Edge cases like browser close, skips, multi-tab sessions

## Solution: Multi-Layer Aggregation

### Architecture

Raw Events (Databricks)  
 ↓ [Real-time]  
├─→ PySpark Aggregation Script  
 ↓ [Batch Processing]  
├─→ Aggregated User-Video Table (One row per User+Video)  
 ↓ [BI Connection]  
└─→ Dashboards & Analytics

## Implementation Timeline

**Duration:** 4 weeks (2 sprints) **Team:** 1 Scrum Master, 1 Product Owner, 2 Data Engineers, 1 Tester

### Sprint 1: Setup & Initial Implementation (Week 1-2)

**Week 1: Understanding & Adaptation** - Kickoff & team onboarding - Code adaptation to company infrastructure - Environment setup & configuration - First integration & smoke tests

**Week 2: Core Testing & Dashboard** - Core scenario testing (P0/P1) - Bug fixes & refinement - Basic dashboard creation - Sprint 1 review & demo

**Deliverables:** - Working aggregation on test data - All critical (P0) scenarios validated - Basic dashboards operational - Bug-free core functionality

### Sprint 2: Production & Optimization (Week 3-4)

**Week 3: Production Deployment** - Production environment setup - Full production deployment - Dashboard finalization - User training sessions

**Week 4: Monitoring & Handover** - Production monitoring - Performance optimization - Knowledge transfer - Final documentation & sign-off

**Deliverables:** - Production-ready pipeline - Complete dashboards - Comprehensive documentation - Team knowledge transfer - Stakeholder sign-off

## Why Only 4 Weeks?

**Key Success Factors:** 1. ✅ **Complete solution already exists** - Only needs adaptation, not development from scratch 2. ✅ **Comprehensive test coverage** - 25 test scenarios provided 3. ✅ **Production-ready code** - Tested and optimized 4. ✅ **Complete documentation** - Requirements, test specs, guides all ready 5. ✅ **Experienced team** - Data engineers familiar with PySpark/Databricks

**What’s included:** - Full code adaptation - Complete testing (90%+ coverage) - Dashboard creation - Production deployment - User training - Documentation

**See** [**IMPLEMENTATION\_PLAN.md**](../IMPLEMENTATION_PLAN.md) **for detailed day-by-day breakdown.**

## Key Metrics

### Video-Level Metrics

* **Unique Viewers**: Number of unique users
* **Total Watch Time**: Total time watched
* **Completion Rate**: % of sessions with “video\_ended” event
* **Avg Watch Time per Session**: Average playback time
* **Engagement Score**: Weighted metric from watch time, completions, sessions

### User-Level Metrics

* **Total Watch Time**: Sum across all videos
* **Videos Watched**: Number of videos started
* **Videos Completed**: Number of videos finished
* **Engagement Tier**: High/Medium/Low/Minimal based on score

### Session-Level Metrics

* **Watch Time**: Actually watched time (without skips)
* **Max Position**: Furthest point in video
* **Pause Count**: Number of pauses
* **Skip Count**: Number of forward/backward jumps
* **Completed**: Boolean, whether video was watched to end

## Scenario Coverage (18 scenarios identified)

### ✅ Fully solved:

1. **Straightforward Play → End** (Basic scenario)
2. **Play → Pause → Resume → End** (With pauses)
3. **Play → Pause → Browser Close** (Session timeout)
4. **Skip Forward/Backward** (Jump detection)
5. **Multiple Sessions (Replay)** (Replay recognition)
6. **Multiple Videos in Session** (Video switching)

### ⚠️ Partially solved (requires client-side changes):

1. **Tab Switch** (Heuristic via pause duration)
2. **Page Refresh** (Session restart detection)
3. **Resume from Last Position** (Requires position tracking)
4. **Autoplay** (Requires autoplay flag in event)

### 🔴 Not solvable with current events:

1. **Video completely ignored** (Requires page-view events with video list)

## Data Quality & Monitoring

### Validation Checks

* ✅ Watch Time ≤ Video Duration (+10% tolerance)
* ✅ No negative watch time
* ✅ Completion only with sufficient watch time (>75%)
* ✅ Session duration < 4 hours
* ✅ Events-per-session ratio: 3-5

### Alerts

* 🚨 Data freshness > 3 hours
* 🚨 Completion rate drop > 20%
* 🚨 Zero-watch-rate > 15%
* 🚨 Negative watch time detected

## Performance & Scaling

### Current Estimate (based on assumptions):

* **10,000 Users/Day** × 5 Videos × 4 Events = **200,000 Events/Day**
* **6M Events/Month**, **73M Events/Year**
* **Storage**: ~7.3 GB/year (Raw), ~1 GB/year (Aggregated)

### Query Performance Targets:

* Session aggregation (1 day): **< 5 seconds**
* User-video metrics (1 week): **< 10 seconds**
* Dashboard refresh: **< 15 seconds**
* Real-time (last hour): **< 2 seconds**

With optimized processing: **Sub-second queries** for aggregated data

## Cost-Benefit Analysis

### Effort Required

**Team:** 5 people (1 SM, 1 PO, 2 Engineers, 1 Tester) **Duration:** 4 weeks (2 sprints) **Total Effort:** ~530 hours

**Breakdown by Role:** - **Data Engineers (2):** 256 hours (~6.4 FTE weeks) - **Product Owner (1):** 98 hours (~2.5 FTE weeks) - **Tester (1):** 96 hours (~2.4 FTE weeks) - **Scrum Master (1):** 80 hours (~2 FTE weeks)

**Note:** Solution already exists - effort is for adaptation and deployment, not building from scratch.

### Benefits

**Immediate (Week 4):** - ✅ **Automated reporting** - No manual aggregation needed - ✅ **Data-driven decisions** - Know which videos work - ✅ **Content optimization** - Identify drop-off points - ✅ **User engagement tracking** - Segment users by behavior

**Ongoing:** - ✅ **ROI measurement** - Justify video production costs - ✅ **A/B testing capability** - Test thumbnails, titles, content - ✅ **Trend analysis** - Track engagement over time - ✅ **Personalization** - Recommend based on watch patterns

### Return on Investment

**Investment:** ~530 hours team effort **Payback:** Continuous automated insights, faster decision making

**Time Saved:** - Before: Manual analysis = 4-8 hours per report - After: Automated = seconds per query - **Annual savings:** 100+ hours of manual work

## Risks & Mitigations

| Risk | Probability | Impact | Mitigation |
| --- | --- | --- | --- |
| **Data structure mismatch** | Medium | High | Early validation in Week 1, flexible schema mapping |
| **Performance with large data** | Low | High | Code already optimized, baseline testing in Week 1 |
| **Missing requirements** | Low | Medium | PO validates against 25 test scenarios in Week 1 |
| **Team member unavailable** | Low | High | Knowledge sharing, pair programming, documentation |
| **BI tool integration issues** | Medium | Medium | Extra buffer in Week 2-3, fallback to basic visualization |
| **Stakeholder feedback delays** | Medium | Low | Early demos, continuous communication |

**Overall Risk:** Low - Solution is proven and complete, only needs adaptation.

## Recommendation

### ✅ Recommended Approach: 4-Week Implementation

**Why this timeline is realistic:** 1. Complete, tested solution already exists 2. Team only needs to adapt, not build from scratch 3. Comprehensive test coverage (25 scenarios) provided 4. All documentation ready 5. Clear implementation plan with daily breakdown

**Decision Points:** - **Week 2:** Go/No-Go for production deployment - **Week 3:** Production readiness review - **Week 4:** Final sign-off and handover

**If timeline is tight, can compress to 3 weeks** by cutting: - P2 edge case testing - Advanced dashboard features - Extended monitoring period

## Next Steps

### This Week (Preparation)

1. ✅ **Review documentation** - Team reads [INDEX.md](../INDEX.md)
2. 📅 **Schedule kickoff** - Book Sprint 1 planning
3. 🔐 **Grant access** - Databricks permissions for all team members
4. 📋 **Setup sprint board** - Create Jira/ADO tickets

### Sprint 1 Day 1 (Kickoff)

1. 🚀 **Sprint planning** - Review [IMPLEMENTATION\_PLAN.md](../IMPLEMENTATION_PLAN.md)
2. 🎯 **Assign responsibilities** - Clear ownership
3. 💻 **Environment setup** - All team members ready
4. ⚡ **Start parallel work** - Engineers, Tester, PO begin tasks

### Weekly Cadence

* **Monday:** Sprint planning (Sprint 2 only)
* **Daily:** 15-min standup
* **Friday:** Sprint review & demo to stakeholders
* **Friday:** Retrospective (end of sprint)

**Ready to start?** See [IMPLEMENTATION\_PLAN.md](../IMPLEMENTATION_PLAN.md) for detailed day-by-day breakdown.

## Appendices

**Detailed documentation:** - [IMPLEMENTATION\_PLAN.md](../IMPLEMENTATION_PLAN.md): Day-by-day plan (4 weeks) - [databricks\_video\_aggregation.py](../03_DEVELOPMENT/databricks_video_aggregation.py): Complete PySpark script - [TEST\_SCENARIOS\_COMPLETE.md](../04_TESTING/TEST_SCENARIOS_COMPLETE.md): 25 test scenarios - [VIDEO\_TRACKING\_SCENARIOS\_GUIDE.md](../02_BUSINESS_ANALYSIS/VIDEO_TRACKING_SCENARIOS_GUIDE.md): Business requirements - [GETTING\_STARTED.md](../05_REFERENCE/GETTING_STARTED.md): Technical setup guide

**Navigate by role:** [INDEX.md](../INDEX.md)