YTEmpire Week 1 Execution Plan

Week 1 Overview

Focus: Core service implementation, API development, and integration foundations
Goal: Functional backend with YouTube integration, basic frontend, and first test video generated
Team Size: 17 specialists working in parallel streams
Sprint: Week 1 represents Sprint 1 of the 12-week MVP cycle

Leadership Team

Role: CEO/Founder

Task 1: Beta User Acquisition Campaign Launch (P0)

Description: Convert Week 0 leads into committed beta users and expand pipeline. **Steps**:

1. Schedule 15 discovery calls with Week 0 qualified leads (30-min slots)
2. Create beta user agreement with terms, expectations, and NDA
3. Develop onboarding packet with video tutorials and getting-started guide
4. Set up Calendly for automated demo scheduling
5. Launch LinkedIn outreach campaign to YouTube creators (target: 50 messages)
6. Create beta user Slack community for direct feedback
7. Prepare customized demos showing ROI potential ($10K/month target) **Duration**: 12 hours across week **Dependencies**: Legal review of beta agreement **Deliverable**: 5+ signed beta users, 50+ qualified pipeline

Task 2: Fundraising Materials Preparation (P1)

Description: Develop investor deck and financial projections based on MVP progress. **Steps**:

1. Update pitch deck with Week 0 technical achievements
2. Create financial model showing path to $50M ARR
3. Record 3-minute product demo video showing automation in action
4. Compile technical due diligence package with architecture diagrams
5. Schedule calls with 3 interested angel investors
6. Prepare competitive analysis showing advantages over TubeBuddy/VidIQ **Duration**: 8 hours **Dependencies**: Metrics from CTO and VP of AI **Deliverable**: Complete Series A prep package

Task 3: Content Marketing Initiative (P2)

Description: Begin building YTEMPIRE brand presence and thought leadership. **Steps**:

1. Write blog post: "Why 95% of YouTubers Fail (And How AI Changes Everything)"
2. Create Twitter/X account and post 5 daily insights about YouTube automation
3. Record podcast appearance prep for 2 scheduled shows
4. Design landing page copy for beta sign-ups
5. Create case study template for beta user success stories **Duration**: 6 hours **Dependencies**: Marketing assets from UI/UX Designer **Deliverable**: Published content and social media presence

Role: CTO/Technical Director

Task 1: Technical Architecture Review & Optimization (P0)

Description: Validate Week 0 implementation and optimize based on initial findings. **Steps**:

1. Conduct architecture review with all team leads (2-hour session)
2. Identify and document technical debt from Week 0 rapid development
3. Prioritize optimization opportunities (focus on <$3/video cost)
4. Create technical roadmap for Weeks 2-4
5. Implement critical performance fixes discovered
6. Set up weekly architecture review process
7. Document decision log for future reference **Duration**: 10 hours **Dependencies**: Week 0 deliverables from all teams **Deliverable**: Optimized architecture and technical roadmap

Task 2: Cross-Team Integration Coordination (P0)

Description: Ensure all teams are properly integrated and APIs are functioning. **Steps**:

1. Review and approve API contracts between teams
2. Facilitate integration testing session (Backend ↔ Frontend ↔ AI)
3. Resolve blocking issues between teams
4. Create integration test suite covering critical paths
5. Document data flow between services with sequence diagrams
6. Establish SLA agreements between services **Duration**: 8 hours **Dependencies**: API implementations from all teams **Deliverable**: Fully integrated service architecture

Task 3: Production Environment Preparation (P1)

Description: Begin setting up production infrastructure for beta launch. **Steps**:

1. Define production architecture requirements (99.9% uptime target)
2. Set up staging environment identical to planned production

3. Configure monitoring and alerting thresholds
4. Create deployment runbooks with rollback procedures
5. Implement zero-downtime deployment strategy
6. Set up backup and disaster recovery systems **Duration**: 8 hours **Dependencies**: Platform Ops infrastructure ready **Deliverable**: Production-ready environment plan

Role: VP of AI

Task 1: ML Pipeline End-to-End Implementation (P0)

Description: Complete the full AI pipeline from trend detection to video generation. **Steps**:

1. Integrate trend detection model with YouTube API data
2. Connect GPT-4 script generation with quality scoring
3. Implement voice synthesis pipeline with fallback options (ElevenLabs → Google TTS)
4. Set up thumbnail generation using Stable Diffusion XL
5. Create model monitoring dashboard with drift detection
6. Optimize for <$0.50 per video cost target
7. Implement emergency cost cutoff at $3.00 **Duration**: 12 hours **Dependencies**: API infrastructure from Backend team **Deliverable**: Functional ML pipeline generating videos

Task 2: Quality Assurance System (P1)

Description: Implement multi-stage quality checks for generated content. **Steps**:

1. Create content scoring rubric (0-100 scale)
2. Implement automated policy violation detection
3. Build confidence scoring for each generation stage
4. Set up human-in-the-loop for low-confidence content (<70%)
5. Create quality metrics dashboard
6. Implement A/B testing for quality improvements **Duration**: 8 hours **Dependencies**: ML pipeline functioning **Deliverable**: Quality assurance system with 90% accuracy

Task 3: Cost Optimization Sprint (P1)

Description: Reduce per-video costs through intelligent optimization. **Steps**:

1. Implement caching for common prompts and responses
2. Create dynamic model selection (GPT-3.5 vs GPT-4 based on complexity)
3. Batch API calls where possible (save 30% on costs)
4. Optimize prompt engineering for token efficiency

5. Set up cost tracking dashboard with real-time alerts
6. Implement progressive cost reduction (premium → standard → economy modes) **Duration**: 6 hours **Dependencies**: Cost data from Week 0 **Deliverable**: 30% cost reduction achieved

Role: Product Owner

Task 1: Sprint 1 Execution & Management (P0)

Description: Run first formal sprint with all ceremonies and tracking. **Steps**:

1. Conduct Sprint 1 planning meeting (4 hours, Monday morning)
2. Create and assign all Jira tickets with story points
3. Run daily standups at 9:30 AM (15 minutes each)
4. Remove blockers and manage dependencies
5. Conduct mid-sprint review (Wednesday)
6. Facilitate Sprint 1 retrospective (Friday)
7. Prepare Sprint 2 backlog based on learnings **Duration**: 15 hours across week **Dependencies**: All team members available **Deliverable**: Completed Sprint 1 with 80% story completion

Task 2: User Testing Sessions (P1)

Description: Conduct user testing with early beta users on core workflows. **Steps**:

1. Recruit 5 beta users for testing sessions
2. Create testing script for channel setup flow
3. Conduct 1-hour sessions via Zoom with recording
4. Document all feedback and pain points
5. Prioritize fixes for Sprint 2
6. Create user testing report with recommendations **Duration**: 10 hours **Dependencies**: Working MVP features **Deliverable**: User testing report with prioritized improvements

Task 3: Metrics Implementation (P2)

Description: Set up analytics and tracking for key product metrics. **Steps**:

1. Implement Mixpanel/Amplitude for user analytics
2. Create funnel tracking for onboarding flow
3. Set up custom events for key actions
4. Build metrics dashboard in analytics tool
5. Document metrics definitions

6. Set up weekly metrics review process **Duration**: 6 hours **Dependencies**: Frontend implementation **Deliverable**: Analytics tracking live with dashboards

---

Backend Team (Under CTO)

Role: Backend Team Lead

Task 1: YouTube Multi-Account Integration (P0)

Description: Implement rotation system for 15 YouTube accounts to manage quotas. **Steps**:

1. Create account pool management system with health scoring
2. Implement automatic failover when quota exceeded (10,000 units/day limit)
3. Build quota tracking and prediction system
4. Create account assignment algorithm (round-robin with weights)
5. Test with simultaneous uploads across accounts
6. Document account management strategy
7. Implement account rest periods to maintain health **Duration**: 10 hours **Dependencies**: YouTube OAuth from Week 0 **Deliverable**: Multi-account system handling 50 videos/day

Task 2: Video Processing Pipeline Implementation (P0)

Description: Build complete pipeline from request to published video. **Steps**:

1. Create video generation request API endpoint
2. Implement Celery task chain for processing stages
3. Add progress tracking with WebSocket updates (5-second intervals)
4. Build error recovery and retry logic (max 3 retries)
5. Create pipeline monitoring dashboard
6. Test with 10 concurrent video generations
7. Implement priority queue for premium users **Duration**: 12 hours **Dependencies**: Queue system from Week 0 **Deliverable**: Reliable pipeline processing 50+ videos/day

Task 3: Performance Optimization Sprint (P1)

Description: Optimize API response times and database queries. **Steps**:

1. Profile all API endpoints with Python profiler
2. Optimize N+1 queries with eager loading
3. Implement database connection pooling (100 connections)
4. Add Redis caching for expensive operations

5. Create performance monitoring dashboard
6. Implement query result caching (1-hour TTL) **Duration**: 6 hours **Dependencies**: Working API endpoints **Deliverable**: All APIs responding <500ms p95

Role: API Developer Engineer #1

Task 1: Channel Management Complete API (P0)

Description: Finish all channel-related endpoints with full functionality. **Steps**:

1. Implement channel creation with YouTube linking
2. Build channel settings and configuration endpoints
3. Create channel analytics aggregation endpoint
4. Add channel scheduling capabilities
5. Implement channel-specific templates
6. Write comprehensive API tests (>80% coverage)
7. Document all endpoints in OpenAPI spec **Duration**: 10 hours **Dependencies**: Database schema ready **Deliverable**: Complete channel management API

Task 2: User Dashboard API (P1)

Description: Create APIs for dashboard data and analytics. **Steps**:

1. Build dashboard summary endpoint with key metrics
2. Create time-series data endpoints for charts
3. Implement real-time metrics via WebSocket
4. Add cost breakdown API endpoint
5. Create revenue tracking endpoint
6. Implement caching for dashboard queries **Duration**: 8 hours **Dependencies**: Analytics data available **Deliverable**: Dashboard API with <200ms response

Task 3: Notification System (P2)

Description: Implement notification system for important events. **Steps**:

1. Create notification model and API
2. Implement email notifications via SendGrid
3. Add in-app notification system
4. Build notification preferences API
5. Create notification templates

6. Test notification delivery reliability **Duration**: 6 hours **Dependencies**: Event system functioning **Deliverable**: Working notification system

Role: API Developer Engineer #2

Task 1: Video Management API (P0)

Description: Build comprehensive video management endpoints. **Steps**:

1. Create video CRUD operations with validation
2. Implement video status tracking endpoint
3. Build video analytics endpoint
4. Add video scheduling API
5. Create bulk operations endpoint (up to 50 videos)
6. Implement video search and filtering
7. Add pagination for large result sets **Duration**: 10 hours **Dependencies**: Video model defined **Deliverable**: Complete video management API

Task 2: Cost Tracking API (P0)

Description: Implement detailed cost tracking and reporting APIs. **Steps**:

1. Create cost recording endpoint for each service
2. Build cost aggregation by time period
3. Implement cost prediction endpoint
4. Add budget alert system ($2.50 warning, $3.00 stop)
5. Create detailed cost breakdown API
6. Implement cost optimization suggestions endpoint **Duration**: 8 hours **Dependencies**: Cost model implemented **Deliverable**: Real-time cost tracking with <$3/video validation

Task 3: Webhook Management System (P1)

Description: Build system for managing external webhooks. **Steps**:

1. Create webhook registration endpoint
2. Implement webhook delivery system with queuing
3. Add retry logic with exponential backoff
4. Build webhook event log
5. Create webhook testing endpoint
6. Implement webhook signature verification **Duration**: 6 hours **Dependencies**: Event system ready **Deliverable**: Reliable webhook delivery system

Role: Data Pipeline Engineer #1

Task 1: Video Processing Pipeline Core (P0)

Description: Implement core video processing workflow with all stages. **Steps**:

1. Create script generation task with GPT integration
2. Implement audio synthesis task with TTS services
3. Build video assembly task using FFmpeg
4. Create thumbnail generation task with AI
5. Implement YouTube upload task with metadata
6. Add comprehensive error handling at each stage
7. Create pipeline performance metrics **Duration**: 12 hours **Dependencies**: AI services integrated **Deliverable**: End-to-end video processing pipeline

Task 2: GPU Resource Management (P1)

Description: Implement intelligent GPU scheduling for video rendering. **Steps**:

1. Create GPU resource pool manager (3 concurrent jobs max)
2. Implement job scheduling algorithm
3. Add memory monitoring and management
4. Build GPU utilization dashboard
5. Create fallback to CPU rendering
6. Implement priority-based scheduling **Duration**: 8 hours **Dependencies**: GPU drivers configured **Deliverable**: Efficient GPU utilization system

Role: Data Pipeline Engineer #2

Task 1: Analytics Data Pipeline (P0)

Description: Build data pipeline for analytics and reporting. **Steps**:

1. Create YouTube Analytics data fetcher
2. Implement data transformation pipeline
3. Build aggregation jobs for metrics
4. Create data warehouse schema
5. Implement incremental data updates
6. Add data quality checks
7. Set up anomaly detection **Duration**: 10 hours **Dependencies**: YouTube API access **Deliverable**: Analytics pipeline updating every hour

Task 2: Cost Aggregation Pipeline (P1)

Description: Build automated cost calculation and tracking pipeline. **Steps**:

1. Create cost collection from all services
2. Implement real-time cost aggregation
3. Build cost allocation by channel/video
4. Create cost prediction model
5. Add cost optimization recommendations
6. Implement cost alerting system **Duration**: 8 hours **Dependencies**: Service integrations complete **Deliverable**: Accurate cost tracking per video

Role: Integration Specialist

Task 1: N8N Production Workflows (P0)

Description: Create production-ready N8N workflows for automation. **Steps**:

1. Build main video generation workflow with error handling
2. Create channel monitoring workflow (every 6 hours)
3. Implement cost tracking workflow
4. Add comprehensive error handling and alerts
5. Create workflow monitoring dashboard
6. Test with 50 video generations
7. Document all workflow triggers and actions **Duration**: 10 hours **Dependencies**: N8N platform ready **Deliverable**: Automated workflows processing 50+ videos/day

Task 2: Payment System Integration (P1)

Description: Complete Stripe integration for subscriptions. **Steps**:

1. Implement subscription creation flow
2. Build payment webhook handlers
3. Create billing portal integration
4. Add usage-based billing for overages
5. Implement payment failure handling
6. Test with test credit cards **Duration**: 8 hours **Dependencies**: Stripe account configured **Deliverable**: Functional payment system

Task 3: Third-Party API Optimization (P2)

Description: Optimize all external API integrations for reliability. **Steps**:

1. Implement circuit breakers for all APIs
2. Add intelligent retry logic
3. Create fallback strategies
4. Build API health monitoring
5. Optimize API call batching
6. Document API limits and quotas **Duration**: 6 hours **Dependencies**: All APIs integrated **Deliverable**: 99% API reliability achieved

---

Frontend Team (Under CTO)

Role: Frontend Team Lead

Task 1: Dashboard Implementation (P0)

Description: Build main dashboard with real-time updates. **Steps**:

1. Create dashboard layout with Material-UI Grid
2. Implement chart components with Recharts
3. Add real-time WebSocket updates
4. Build metric cards with animations
5. Create responsive grid system
6. Implement loading states and error boundaries
7. Add data refresh controls **Duration**: 10 hours **Dependencies**: Dashboard API ready **Deliverable**: Functional dashboard showing key metrics

Task 2: State Management Optimization (P1)

Description: Optimize Zustand stores for performance. **Steps**:

1. Implement store persistence with localStorage
2. Add optimistic updates for better UX
3. Create computed values with memoization
4. Implement store DevTools integration
5. Add error boundary handling
6. Create store reset functionality **Duration**: 6 hours **Dependencies**: State structure defined **Deliverable**: Optimized state management

Task 3: Component Library Expansion (P2)

Description: Build additional reusable components. **Steps**:

1. Create data table component with sorting/filtering
2. Build file upload component with drag-and-drop
3. Create toast notification system
4. Add modal dialog system
5. Document all components in Storybook
6. Create component testing suite **Duration**: 8 hours **Dependencies**: Base components ready **Deliverable**: 20+ reusable components

Role: React Engineer

Task 1: Channel Management Interface (P0)

Description: Build complete channel management UI. **Steps**:

1. Create channel list view with cards
2. Build channel creation wizard (3-step process)
3. Implement channel settings form with validation
4. Add channel analytics view with charts
5. Create channel scheduling interface
6. Implement channel templates selection
7. Add channel health indicators **Duration**: 12 hours **Dependencies**: Channel API ready **Deliverable**: Full channel management functionality

Task 2: Video Queue Interface (P1)

Description: Create video queue management interface. **Steps**:

1. Build queue visualization with status indicators
2. Create video detail modal with metadata
3. Implement drag-and-drop reordering
4. Add bulk actions toolbar
5. Create filtering and search functionality
6. Add queue statistics summary **Duration**: 8 hours **Dependencies**: Video API ready **Deliverable**: Interactive video queue manager

Task 3: User Settings Pages (P2)

Description: Implement user account and settings pages. **Steps**:

1. Create account settings form

2. Build subscription management UI
3. Implement notification preferences
4. Add API key management interface
5. Create billing history view
6. Add data export functionality **Duration**: 6 hours **Dependencies**: User API endpoints **Deliverable**: Complete settings section

Role: Dashboard Specialist

Task 1: Real-Time Analytics Dashboard (P0)

Description: Build comprehensive analytics dashboard with live updates. **Steps**:

1. Implement revenue tracking chart (line chart)
2. Create video performance metrics (bar charts)
3. Build channel comparison charts (multi-series)
4. Add cost breakdown visualization (pie chart)
5. Implement date range selector component
6. Create export functionality (CSV/PDF)
7. Add chart drill-down capabilities **Duration**: 10 hours **Dependencies**: Analytics API ready **Deliverable**: Full analytics dashboard

Task 2: WebSocket Integration (P1)

Description: Implement WebSocket client for real-time updates. **Steps**:

1. Create WebSocket service class with TypeScript
2. Implement reconnection logic with exponential backoff
3. Add event handlers for different update types
4. Integrate with Zustand stores for state updates
5. Create connection status indicator component
6. Add debugging tools for WebSocket events **Duration**: 8 hours **Dependencies**: WebSocket server ready **Deliverable**: Real-time updates working

Role: UI/UX Designer

Task 1: Beta User Onboarding Flow (P0)

Description: Design complete onboarding experience for new users. **Steps**:

1. Create welcome screen designs with value proposition

2. Design step-by-step setup wizard (5 steps)
3. Build channel connection flow UI
4. Create first video generation interface
5. Design success/completion screens
6. Create helpful tooltips and contextual guides
7. Add progress indicators throughout **Duration**: 10 hours **Dependencies**: User journey defined **Deliverable**: Complete onboarding designs in Figma

Task 2: Mobile Responsive Designs (P1)

Description: Adapt desktop designs for tablet and mobile views. **Steps**:

1. Create responsive breakpoints (mobile: 375px, tablet: 768px)
2. Design mobile navigation (hamburger menu)
3. Adapt dashboard for mobile screens
4. Create touch-friendly interfaces
5. Design mobile-specific features
6. Document responsive behavior **Duration**: 8 hours **Dependencies**: Desktop designs approved **Deliverable**: Responsive design system

Task 3: Design System Documentation (P2)

Description: Document design system for developer use. **Steps**:

1. Create component usage guidelines
2. Document color palette and usage rules
3. Provide spacing and layout grid system
4. Create interaction patterns guide
5. Build accessibility guidelines (WCAG 2.1 AA)
6. Export design tokens for development **Duration**: 6 hours **Dependencies**: Design system stabilized **Deliverable**: Complete design documentation

---

Platform Operations Team (Under CTO)

Role: Platform Ops Lead

Task 1: Production Infrastructure Setup (P0)

Description: Configure production environment for beta launch. **Steps**:

1. Set up production Docker Swarm/Kubernetes cluster

2. Configure load balancing with Nginx
3. Implement SSL certificates for all domains
4. Set up CDN for static assets (Cloudflare)
5. Create production database with replication
6. Configure production monitoring with alerts
7. Test failover scenarios **Duration**: 12 hours **Dependencies**: Infrastructure plan approved **Deliverable**: Production environment ready

Task 2: Disaster Recovery Implementation (P1)

Description: Implement comprehensive backup and recovery system. **Steps**:

1. Set up automated database backups (every 6 hours)
2. Configure file system snapshots
3. Create disaster recovery runbook
4. Test full system recovery procedure
5. Implement backup monitoring and alerts
6. Document recovery time objectives (RTO: 4 hours) **Duration**: 8 hours **Dependencies**: Production environment ready **Deliverable**: DR system with 4-hour RTO

Task 3: Security Audit (P2)

Description: Conduct security audit and implement fixes. **Steps**:

1. Run vulnerability scanning with OWASP ZAP
2. Review access controls and permissions
3. Audit API security implementation
4. Check for exposed secrets in code
5. Implement identified security fixes
6. Create security checklist for releases **Duration**: 6 hours **Dependencies**: All services deployed **Deliverable**: Security audit report with fixes

Role: DevOps Engineer #1

Task 1: CI/CD Pipeline Enhancement (P0)

Description: Enhance CI/CD pipeline for production deployments. **Steps**:

1. Add staging deployment stage to pipeline
2. Implement blue-green deployment strategy
3. Add automated rollback triggers on failures

4. Create deployment notifications (Slack)
5. Implement feature flags system (LaunchDarkly)
6. Add performance testing stage
7. Create deployment approval workflow **Duration**: 10 hours **Dependencies**: Basic CI/CD working **Deliverable**: Production-grade CI/CD pipeline

Task 2: Container Optimization (P1)

Description: Optimize Docker containers for size and performance. **Steps**:

1. Implement multi-stage builds for all services
2. Optimize base images (Alpine Linux where possible)
3. Add health checks to all containers
4. Configure resource limits (CPU/Memory)
5. Implement container security scanning
6. Document container best practices **Duration**: 8 hours **Dependencies**: All services containerized **Deliverable**: Optimized containers with 50% size reduction

Role: DevOps Engineer #2

Task 1: Monitoring Enhancement (P0)

Description: Expand monitoring coverage and create dashboards. **Steps**:

1. Add application-level metrics with Prometheus
2. Create business metrics dashboard in Grafana
3. Implement log aggregation with Loki
4. Set up alert rules for critical metrics
5. Create on-call runbooks for alerts
6. Test alert escalation chain
7. Document monitoring architecture **Duration**: 10 hours **Dependencies**: Monitoring stack running **Deliverable**: Comprehensive monitoring system

Task 2: Auto-Scaling Implementation (P1)

Description: Implement auto-scaling for video processing. **Steps**:

1. Define scaling metrics and thresholds
2. Implement horizontal pod autoscaling (HPA)
3. Create scaling policies (min: 2, max: 10)
4. Test scaling under simulated load

5. Document scaling behavior
6. Set up cost controls for scaling **Duration**: 8 hours **Dependencies**: Container orchestration ready **Deliverable**: Auto-scaling handling 2x load

Role: Security Engineer #1

Task 1: API Security Implementation (P0)

Description: Implement comprehensive API security measures. **Steps**:

1. Implement rate limiting per endpoint (100 req/min default)
2. Add API key management system
3. Configure CORS properly for frontend
4. Implement request validation middleware
5. Add security headers (HSTS, CSP, etc.)
6. Create API audit logging system
7. Test with security tools (Burp Suite) **Duration**: 10 hours **Dependencies**: APIs functioning **Deliverable**: Secured API infrastructure

Task 2: Data Encryption (P1)

Description: Implement encryption for sensitive data. **Steps**:

1. Encrypt data at rest in database
2. Implement field-level encryption for PII
3. Configure TLS 1.3 for all connections
4. Implement key rotation schedule
5. Document encryption standards
6. Test encryption implementation **Duration**: 8 hours **Dependencies**: Database ready **Deliverable**: Full encryption implementation

Role: Security Engineer #2

Task 1: Access Control System (P0)

Description: Implement role-based access control. **Steps**:

1. Define role hierarchy (Admin, User, Viewer)
2. Implement RBAC in application layer
3. Create permission management interface
4. Add comprehensive audit logging
5. Test permission boundaries

6. Document access control policies
7. Implement principle of least privilege **Duration**: 10 hours **Dependencies**: Authentication working **Deliverable**: Complete RBAC system

Task 2: Security Monitoring (P1)

Description: Set up security monitoring and alerts. **Steps**:

1. Configure intrusion detection system
2. Set up failed login monitoring (lock after 5 attempts)
3. Implement anomaly detection for API usage
4. Create security dashboard in Grafana
5. Configure security alerts to PagerDuty
6. Document security incident response **Duration**: 8 hours **Dependencies**: Logging infrastructure ready **Deliverable**: Security monitoring active

Role: QA Engineer #1

Task 1: End-to-End Test Suite (P0)

Description: Create comprehensive E2E test suite. **Steps**:

1. Write tests for user registration flow
2. Create channel creation and setup tests
3. Test video generation pipeline end-to-end
4. Implement payment flow tests
5. Add cross-browser testing (Chrome, Firefox, Safari)
6. Set up test reporting with Allure
7. Integrate with CI/CD pipeline **Duration**: 12 hours **Dependencies**: Features implemented **Deliverable**: 50+ E2E tests passing

Task 2: Performance Testing (P1)

Description: Conduct performance testing and optimization. **Steps**:

1. Create load testing scripts with k6
2. Test API endpoints under load (100 concurrent users)
3. Identify performance bottlenecks
4. Test database query performance
5. Create performance baseline report

6. Recommend optimizations **Duration**: 8 hours **Dependencies**: System deployed **Deliverable**: Performance test report

Role: QA Engineer #2

Task 1: API Testing Framework (P0)

Description: Build automated API testing framework. **Steps**:

1. Set up Postman/Newman collections
2. Create automated API tests for all endpoints
3. Implement contract testing with Pact
4. Add negative test cases and edge cases
5. Create API test execution reports
6. Integrate with CI/CD pipeline
7. Document API testing strategy **Duration**: 10 hours **Dependencies**: APIs documented **Deliverable**: 100+ API tests automated

Task 2: Mobile Testing (P1)

Description: Test responsive design and mobile functionality. **Steps**:

1. Test on iOS devices (iPhone 12+)
2. Test on Android devices (Pixel, Samsung)
3. Verify responsive breakpoints work correctly
4. Test touch interactions and gestures
5. Create mobile compatibility matrix
6. Document mobile-specific issues **Duration**: 8 hours **Dependencies**: Mobile UI ready **Deliverable**: Mobile compatibility verified

---

AI Team (Under VP of AI)

Role: AI/ML Team Lead

Task 1: Model Deployment Pipeline (P0)

Description: Implement production model deployment system. **Steps**:

1. Set up model registry with MLflow
2. Implement A/B testing framework for models
3. Create model versioning system
4. Build gradual rollout mechanism (canary deployment)

5. Add comprehensive model monitoring
6. Create automated rollback procedures
7. Document deployment process **Duration**: 10 hours **Dependencies**: Models trained **Deliverable**: Production-ready model deployment

Task 2: Performance Optimization (P1)

Description: Optimize model inference for latency and cost. **Steps**:

1. Implement model quantization (reduce size by 75%)
2. Add batch inference capabilities
3. Optimize prompt engineering for fewer tokens
4. Cache common predictions in Redis
5. Create performance benchmarks
6. Implement model warm-up on startup **Duration**: 8 hours **Dependencies**: Models deployed **Deliverable**: 50% latency reduction achieved

Task 3: Team Coordination (P2)

Description: Coordinate AI team deliverables and integration. **Steps**:

1. Run daily AI team standups
2. Review model performance metrics
3. Prioritize optimization tasks
4. Coordinate with Backend team on API needs
5. Document AI system architecture
6. Prepare weekly AI progress report **Duration**: 6 hours ongoing **Dependencies**: Team available **Deliverable**: Coordinated AI team execution

Role: ML Engineer

Task 1: Trend Prediction Model Production (P0)

Description: Deploy trend prediction model to production. **Steps**:

1. Finalize model training with latest YouTube data
2. Implement feature pipeline for real-time features
3. Create prediction API endpoint with FastAPI
4. Add confidence scoring to predictions
5. Implement model monitoring for drift
6. Test with real-time data stream

7. Document model performance metrics **Duration**: 10 hours **Dependencies**: Training data ready **Deliverable**: 75% accuracy trend prediction live

Task 2: Content Quality Scorer (P1)

Description: Build model to score content quality. **Steps**:

1. Define quality metrics (engagement, retention, CTR)
2. Create labeled training dataset
3. Train quality scoring model (BERT-based)
4. Implement scoring API endpoint
5. Validate against human quality ratings
6. Create quality threshold configurations **Duration**: 8 hours **Dependencies**: Content samples available **Deliverable**: Quality scorer with 85% correlation to human ratings

Role: Data Engineer

Task 1: Feature Store Implementation (P0)

Description: Build feature store for ML models. **Steps**:

1. Design feature schema for all ML features
2. Implement feature ingestion pipeline with Kafka
3. Create feature serving API with low latency
4. Add feature versioning system
5. Build feature monitoring dashboard
6. Document feature definitions and lineage
7. Implement feature backfilling capability **Duration**: 10 hours **Dependencies**: Data pipeline ready **Deliverable**: Feature store serving 100+ features

Task 2: Training Pipeline Automation (P1)

Description: Automate model training pipelines. **Steps**:

1. Create training orchestration with Apache Airflow
2. Implement data validation checks
3. Add hyperparameter tuning with Optuna
4. Create model evaluation pipeline
5. Set up automated retraining triggers
6. Implement training monitoring dashboard **Duration**: 8 hours **Dependencies**: ML infrastructure ready **Deliverable**: Automated training pipeline

Role: Data Scientist/Analyst

Task 1: A/B Testing Framework (P0)

Description: Implement A/B testing for content optimization. **Steps**:

1. Design experiment framework with proper randomization
2. Create user assignment system
3. Build metrics collection pipeline
4. Implement statistical significance testing
5. Create A/B test results dashboard
6. Document testing methodology
7. Set up automated test analysis **Duration**: 10 hours **Dependencies**: Analytics pipeline ready **Deliverable**: A/B testing system with significance testing

Task 2: Business Metrics Dashboard (P1)

Description: Create comprehensive business analytics dashboard. **Steps**:

1. Define key business metrics (CAC, LTV, MRR)
2. Build ETL pipelines for metrics calculation
3. Create visualization dashboards in Tableau/Looker
4. Add predictive analytics for revenue
5. Implement anomaly detection for metrics
6. Set up automated reporting **Duration**: 8 hours **Dependencies**: Data warehouse ready **Deliverable**: Executive dashboard with KPIs

---

Week 1 Critical Milestones

Monday End-of-Day

- ✅ Sprint 1 planning complete with all tasks assigned
- ✅ All teams have clear priorities and dependencies mapped
- ✅ Development environment stable and accessible
- ✅ First integration test between Backend and Frontend

Tuesday End-of-Day

- ✅ Core APIs functional (authentication, channels, videos)
- ✅ YouTube multi-account system implemented
- ✅ ML pipeline components integrated

- ✅ Frontend dashboard skeleton rendering

Wednesday Mid-Week Checkpoint

- ✅ First end-to-end video generated successfully
- ✅ Cost tracking showing <$3 per video
- ✅ 5 beta users onboarded and testing
- ✅ Mid-sprint review identifying blockers

Thursday End-of-Day

- ✅ All P0 tasks completed or near completion
- ✅ Production infrastructure configured
- ✅ Security measures implemented
- ✅ Quality assurance tests passing

Friday Sprint Close

- ✅ Sprint 1 retrospective completed
- ✅ 10+ videos generated successfully
- ✅ All critical integrations working
- ✅ Sprint 2 backlog prepared
- ✅ Beta user feedback collected

Risk Mitigation & Contingency Plans

Risk 1: YouTube API Integration Delays

Mitigation:

- Have 15 accounts ready with OAuth completed
- Implement mock YouTube API for testing
- Create manual upload fallback process

Risk 2: Cost Per Video Exceeding $3

Mitigation:

- Implement progressive cost reduction (Premium → Economy mode)
- Cache all AI responses aggressively
- Switch to cheaper models when approaching limit

Risk 3: Video Generation Pipeline Failures

Mitigation:

- Implement comprehensive retry logic
- Create manual intervention workflow
- Set up alerts for pipeline failures

Risk 4: Beta User Onboarding Issues

Mitigation:

- Provide white-glove onboarding support
- Create video tutorials for common tasks
- Have dedicated Slack channel for support

Success Criteria for Week 1

Technical Achievements

- ✅ 50+ test videos generated successfully
- ✅ <$3 average cost per video achieved
- ✅ <10 minute end-to-end generation time
- ✅ 99% API uptime maintained
- ✅ All P0 and P1 tasks completed

Business Achievements

- ✅ 5+ beta users actively using platform
- ✅ First revenue-generating video published
- ✅ Investor deck updated with progress
- ✅ 100+ user waitlist accumulated

Team Performance

- ✅ 80% sprint velocity achieved
- ✅ All teams integrated successfully
- ✅ Daily standups maintaining alignment
- ✅ No critical blockers remaining

Week 2 Preview

Key Focus Areas

- Scaling to 100 videos/day capacity
- Implementing advanced ML features
- Enhancing user experience based on feedback
- Preparing for 10 beta user milestone
- Cost optimization to reach <$1.50/video

Major Deliverables

- Advanced analytics dashboard
- Multi-channel optimization
- Automated content scheduling
- Revenue tracking system
- Performance improvements

Communication Plan

Daily Standups

- **Time**: 9:30 AM PST
- **Duration**: 15 minutes
- **Format**: What I did / What I'm doing / Blockers
- **Tool**: Zoom with recording

Cross-Team Syncs

- **Backend ↔ Frontend**: Tuesday 2 PM
- **Backend ↔ AI**: Wednesday 3 PM
- **Platform Ops ↔ All**: Thursday 4 PM
- **Leadership Review**: Friday 2 PM

Documentation

- **Wiki**: Confluence for all documentation
- **Code**: GitHub with PR reviews
- **Metrics**: Daily dashboard updates
- **Progress**: Jira burn-down charts

Resource Allocation

Infrastructure

- **Local Server**: AMD Ryzen 9 9950X3D (100% allocated)
- **GPU**: NVIDIA RTX 5090 (3 concurrent video renders)
- **Memory**: 128GB (PostgreSQL: 16GB, Redis: 8GB, Apps: 40GB)
- **Storage**: 2TB NVMe + 8TB backup

API Quotas

- **YouTube**: 10,000 units/day per account (15 accounts)
- **OpenAI**: $500/day budget
- **ElevenLabs**: 100,000 characters/day
- **Google TTS**: Unlimited (pay-per-use)

Team Availability

- **Core Hours**: 9 AM - 6 PM PST
- **On-Call**: Platform Ops rotation
- **Weekend**: Emergency support only
- **Communication**: Slack primary, email backup

Document Status: Complete Week 1 Execution Plan **Last Updated**: Start of Week 1 **Next Review**: Friday Week 1 Retrospective **Owner**: CTO with input from all Team Leads **Approval**: Ready for execution