Product Requirements Document (PRD)

Project Name: WATCHWITHAI - AI-Powered Video Recommendation System for News Platforms

Author: Ishaan Goyal

# 1. Overview

WATCHWITHAI is a multimedia video metadata extraction and embedding pipeline designed to enhance video understanding and enable intelligent video recommendations. The system extracts rich metadata and multimodal embeddings from uploaded videos, leveraging AI models for transcription, summarization, and feature extraction. The data is stored and managed within Firebase to enable fast retrieval and personalized recommendations in the WATCHWITHAI platform.

# 2. Goals & Objectives

* Extract comprehensive metadata from videos including transcript, summary, tags, duration, thumbnails, and categories.
* Generate multimodal embeddings representing visual, audio, and textual content to support similarity search and recommendation.
* Provide an efficient pipeline for on-demand metadata and embedding extraction for each video.
* Store and organize extracted metadata and embeddings in Firebase Firestore for integration with frontend services.
* Support extensibility for adding new embedding types or metadata fields in the future.

# 3. Target Users / Personas

|  |  |  |
| --- | --- | --- |
| Persona | Description | Needs / Goals |
| Casual Viewers | Visit occasionally, not logged in | Browse trending/top news videos quickly, no login required |
| Regular Viewers | Watch videos frequently, logged in | Personalized video recommendations based on interests |
| Subscribed Users | Logged-in users with watch history and preferences | Highly relevant, tailored video recommendations |

# 4. Use Cases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Use Case ID | Description | Actors | Preconditions | Flow / Steps | Expected Outcome |
| UC1 | Show related videos after a video is watched | User | User watches a video | 1. User finishes watching. 2. System fetches similar videos based on video embeddings. | 10 related videos appear in sidebar. |
| UC2 | Display personalized recommended videos on homepage | Logged-in User | User is authenticated | 1. User opens homepage. 2. System fetches personalized recommendations based on watch history. | Personalized recommendations displayed. |
| UC3 | Provide category-based trending videos for new users | New User | No watch history available | 1. User visits homepage. 2. System fetches trending videos per selected categories. | Trending videos shown without login. |
| UC4 | Allow search by keyword or tag | Anonymous User | Videos have metadata and tags | 1. User inputs search keyword/tag. 2. System filters videos matching search. | Relevant videos matching keywords/tags displayed. |

# 5. Key Features & Functionalities

## Video Metadata Extraction

* Transcription: Use Whisper ASR to generate accurate, timestamped transcripts from video audio.
* Summary Generation: Use a transformer-based summarization model to produce concise video summaries.
* Duration & Thumbnails: Extract exact video duration and generate representative thumbnail images.
* Tags & Categories: Extract or infer relevant tags and categories from transcripts and metadata.

## Multimodal Embedding Extraction

* Visual Embeddings: Extract frame-level visual features using CLIP or similar vision transformer models.
* Audio Embeddings: Extract audio features such as MFCCs or other spectral representations using librosa.
* Textual Embeddings: Generate semantic text embeddings of transcripts using SentenceTransformers.

## Data Storage & Access

* Organize and store metadata and embeddings in Firebase Firestore under each video document.
* Maintain normalized and indexed structures to allow efficient querying and retrieval by other services.

# 6. Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement Description | Priority | Notes |
| FR1 | The system shall transcribe audio to text using Whisper. | High | Timestamped transcripts required. |
| FR2 | The system shall generate a natural language summary from transcripts. | High | Use transformer summarization pipeline. |
| FR3 | The system shall extract and store video duration and thumbnails. | High | Thumbnails should be representative. |
| FR4 | The system shall extract visual embeddings for sampled video frames. | High | Use CLIP or ViT-based models. |
| FR5 | The system shall extract audio embeddings (e.g., MFCC) from audio. | High | Use librosa or similar library. |
| FR6 | The system shall extract textual embeddings of the transcript. | High | Use SentenceTransformer models. |
| FR7 | The system shall store all metadata and embeddings in Firebase Firestore. | High | Structured per video document. |
| FR8 | The system shall allow on-demand extraction per video via API or command. | Medium | Processing status must be tracked. |
| FR9 | The system shall provide error handling and logging for extraction failures. | Medium |  |

# 7. Non-Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement Description | Priority | Notes |
| NFR1 | Processing should be completed within reasonable time limit (e.g., under 5 minutes/video) | Medium | Dependent on video length and hardware. |
| NFR2 | Data stored in Firebase must be consistent and queryable in real-time. | High |  |
| NFR3 | The pipeline should be modular to enable easy updates of individual models or extraction methods. | Medium |  |
| NFR4 | The system should provide detailed logs for debugging and auditing. | Medium |  |

# 8. Architecture Overview

Processing Components:  
 • Audio extraction and Whisper transcription  
 • Transcript summarization using transformer pipeline  
 • Visual feature extraction from sampled frames with CLIP/ViT  
 • Audio feature extraction using librosa (MFCC)  
 • Text embedding generation with SentenceTransformer  
Storage: Firebase Firestore for storing metadata and embeddings  
Frontend: Uses stored metadata and embeddings for recommendations and display

# 9. Timeline & Milestones

|  |  |  |
| --- | --- | --- |
| Phase | Tasks | Estimated Duration |
| Phase 1: Setup | Configure environment, Firebase integration, Whisper setup | 1 day |
| Phase 2: Metadata Extraction | Implement transcription, summarization, duration, thumbnails | 12 hours |
| Phase 3: Embedding Extraction | Implement visual, audio, text embedding extraction | 12 hours |
| Phase 4: Storage & API | Integrate with Firebase, build API for on-demand processing | 1 day |
| Phase 5: Testing & Optimization | End-to-end testing, error handling, performance tuning | 2 days |
| Phase 6: Documentation & Deployment | Final docs, deployment to production environment | 2 days |

# 10. Risks & Mitigation

* Model performance variability: Use pre-trained models with fine-tuning options and fallback mechanisms.
* Processing latency: Optimize pipeline and consider batch processing for heavy loads.
* Storage limits: Monitor Firebase usage and plan data lifecycle management.

# 11. Future Enhancements

* Support additional embedding types (e.g., motion features).
* Add multilingual transcription and summarization.
* Integrate feedback loop for recommendation improvement using user engagement data.