

Accessible Video Processing Platform - Technical Documentation v2.0

Document Version: 2.0

Last Updated: October 9, 2025

System Version: 1.0.0

Author: Technical Documentation Team

Table of Contents

- [1. Executive Summary](#)
 - [2. Platform Capabilities](#)
 - [3. User Roles & Permissions](#)
 - [4. System Architecture](#)
 - [5. Process Flows & Workflows](#)
 - [6. Database Schema](#)
 - [7. API Overview](#)
 - [8. AI Processing Pipeline](#)
 - [9. Real-time Features](#)
 - [10. Deployment Architecture](#)
 - [11. Security Model](#)
 - [12. Technical Stack](#)
-

1. Executive Summary

The Accessible Video Processing Platform is an enterprise-grade SaaS solution that automatically generates closed captions and audio descriptions for video content using advanced AI technology. The platform combines Google's Gemini 2.5 Pro AI with human quality control workflows to deliver WCAG 2.1 AA/AAA compliant accessibility content at scale.

Key Value Propositions

- Automated AI Processing:** Converts uploaded videos to accessibility-compliant content in minutes
- Multi-Language Support:** Translates captions and audio descriptions to 40+ languages
- Professional Quality Control:** Built-in review workflows ensure accuracy and compliance
- Real-Time Monitoring:** Live status updates and WebSocket-powered dashboards
- Scalable Architecture:** Docker-based microservices handle concurrent processing
- Enterprise Security:** Role-based access control, JWT authentication, audit logging

Primary Use Cases

- Corporate Training Videos** - Make internal training accessible to all employees
 - Marketing Content** - Expand reach with multilingual captions and audio descriptions
 - Educational Content** - Comply with accessibility requirements for online courses
 - Broadcast Media** - Prepare content for FCC compliance and international distribution
 - Legal Compliance** - Meet ADA, Section 508, and WCAG requirements
-

2. Platform Capabilities

2.1 Core Features

Automated Accessibility Generation

AI-Powered Caption Creation

- Automatic speech-to-text transcription with 95%+ accuracy
- Speaker identification and dialogue attribution
- Punctuation and formatting (question marks, exclamations)
- Technical term recognition
- Proper noun capitalization
- WebVTT format with precise millisecond timing

Audio Description Generation

- Scene setting descriptions (location, time of day, environment)
- Character appearance and actions
- On-screen text narration (signs, graphics, titles)
- Visual storytelling elements (expressions, gestures)
- Timed to fit between dialogue gaps
- WebVTT format synchronized with video
- Optional MP3 audio track generation

Quality Assurance

- AI confidence scoring (0-100%)
- Automatic validation of VTT format
- Timing overlap detection
- Minimum content requirements
- Self-healing for malformed AI responses

Multi-Language Translation

Supported Languages

- Spanish, French, German, Italian, Portuguese
- Japanese, Korean, Chinese (Simplified/Traditional)
- Arabic, Hebrew, Russian
- 40+ total languages via Google Cloud Translate

Translation Methods

- **Standard Translation:** Direct language conversion preserving meaning
- **Transcreation:** Cultural adaptation maintaining brand voice and local idioms (via Gemini AI)
- **Timing Preservation:** All translations maintain original cue timestamps

Audio Description Localization

- Translated audio description scripts (VTT)
- Text-to-speech generation in target languages
- Language-specific voice selection
- Natural-sounding neural voices

Professional VTT Editor

Editing Capabilities

- Inline text editing with live preview
- Cue-by-cue navigation
- Timestamp display (HH:MM:SS.mmm format)
- Bulk timing adjustments (-30s to +30s)
- Real-time validation with error highlighting
- Cue duration calculations
- Total duration statistics

Editing Controls

- Add/remove cues (planned feature)
- Split/merge cues (planned feature)
- Undo/redo (browser native)
- Keyboard shortcuts (Ctrl+S save, Ctrl+Enter confirm)
- Auto-save with change detection

Video Preview & Playback

Integrated Video Player

- HTML5 video player with standard controls
- Real-time caption overlay (synchronized)
- Audio description track player
- Multi-language caption selection
- Click-to-jump timeline navigation
- Cue highlighting (shows active caption)
- Caption on/off toggle

Preview Modes

- Side-by-side (video + editors)
- Video only (full preview)
- Editor only (text focus)

2.2 Quality Control Workflow

English Content Review (Primary QC)

Reviewer Responsibilities

- Verify caption accuracy against audio

- Check audio description completeness
- Edit VTT content as needed
- Adjust timing for synchronization issues
- Approve or reject with detailed notes

Tools Provided

- Dual VTT editors (captions + audio descriptions)
- Synchronized video preview
- Timing adjustment tool
- Validation feedback
- Keyboard shortcuts for efficiency

Decision Outcomes

- **Approve:** Triggers automatic translation/TTS pipeline
- **Reject:** Returns to AI processing with feedback for reprocessing

Multi-Language Final Review

Reviewer Responsibilities

- Validate translated caption accuracy
- Verify audio description translations
- Test TTS audio quality and pronunciation
- Check all assets present and downloadable
- Final approval for client delivery

Tools Provided

- Per-language asset viewers
- MP3 audio players for TTS validation
- Read-only VTT preview
- Asset completeness checklist
- Error reporting

Decision Outcomes

- **Approve for Delivery:** Job marked complete, client notified
- **Return for QC:** Send back for corrections with detailed notes

2.3 Job Management Features

Job Creation & Upload

Upload Methods

- Drag-and-drop file upload
- Click-to-browse file selection
- Real-time progress tracking (0-100%)

- Upload cancellation support

Job Configuration

- Custom job title
- Source language selection
- Output type selection (captions, AD script, AD audio)
- Target language selection (multiple)
- Transcreation preference (per language)

Validation

- File type restrictions (MP4 only)
- File size limits (up to 2GB)
- Required field validation
- Instant feedback on errors

Job Monitoring Dashboard

Client View

- Total jobs count
- Processing jobs count
- Jobs in review count
- Completed jobs count
- Recent activity feed (last 5 jobs)
- Real-time status updates

Reviewer/Admin View

- System-wide job statistics
- QC queue depth with pending counts
- Final review queue depth
- Processing activity across all clients
- Quick navigation to review queues

Job Lifecycle Tracking

Status Indicators

- Created (gray) - Job queued for processing
- Ingesting (blue) - Downloading and analyzing video
- AI Processing (blue) - Generating accessibility content
- Pending QC (yellow) - Awaiting human review
- Approved (green) - English content approved
- Translating (purple) - Multi-language processing
- TTS Generating (purple) - Audio synthesis in progress
- Pending Final Review (orange) - Awaiting final approval
- Completed (green) - Ready for client download
- Rejected (red) - Requires revision

Real-Time Updates

- WebSocket-powered status changes
- Toast notifications for major transitions
- Progress percentage (when available)
- Estimated time remaining (calculated)
- Error messages with context

Asset Download System

Download Experience

- Organized by language
- Source video included
- 24-hour signed URL generation
- Secure download links (no authentication needed after generation)
- Batch download capability (coming soon)

Asset Organization

- Source video (MP4)
- Per language:
 - Closed Captions (VTT file)
 - Audio Description Script (VTT file)
 - Audio Description Audio (MP3 file)

File Naming Convention

1	{JobTitle}_source.mp4
2	{JobTitle}_en_captions.vtt
3	{JobTitle}_en_ad.vtt
4	{JobTitle}_en_ad.mp3
5	{JobTitle}_es_captions.vtt
6	{JobTitle}_es_ad.vtt
7	{JobTitle}_es_ad.mp3

2.4 Administrative Capabilities

User Management

User Operations

- Create new users (client, reviewer, admin)
- Update user profiles (email, name, role)
- Deactivate/reactivate accounts
- Reset passwords (generates secure temporary password)
- View user activity history

User Listing

- Filter by role (client, reviewer, admin)
- Filter by active status
- Pagination (20 users per page)
- Sort by creation date, email, role
- Quick search by email or name

System Monitoring & Statistics

Job Statistics

- Total jobs processed
- Jobs by status breakdown (pie chart data)
- Average processing time
- Completion rate percentage
- Daily job creation trends
- Queue depth monitoring

Health Monitoring

- MongoDB connection status
- Redis connection status
- Google Cloud Storage accessibility
- Celery worker count and active tasks
- API response time metrics
- Error rate tracking

Performance Metrics

- Min/max/avg processing times by pipeline stage
- Time-range analysis (7, 30, 90 days)
- Jobs created vs completed rates
- Queue wait time statistics
- Worker utilization percentage

Audit Trail & Compliance

Audit Log Features

- Comprehensive logging of all user actions
- Security event tracking
- Filterable by:
 - Time range (date pickers)
 - Action type (login, create, update, delete, approve)
 - Severity (info, warning, critical)
 - User ID or email
 - Resource type (job, user, file)
 - Success/failure status
- Full-text search across descriptions
- Exportable audit reports (planned)

Tracked Events

- All authentication attempts (success/failure)
- Job creation, approval, rejection, completion, deletion
- User account changes (create, update, deactivate, role change)
- Password resets
- Bulk operations
- Security violations (rate limits, unauthorized access)
- Admin maintenance actions

Retention Policy

- Configurable retention (default: 365 days)
- Admin-triggered cleanup with confirmation
- Cleanup actions themselves audited (meta-auditing)

Maintenance Operations

Job Reprocessing

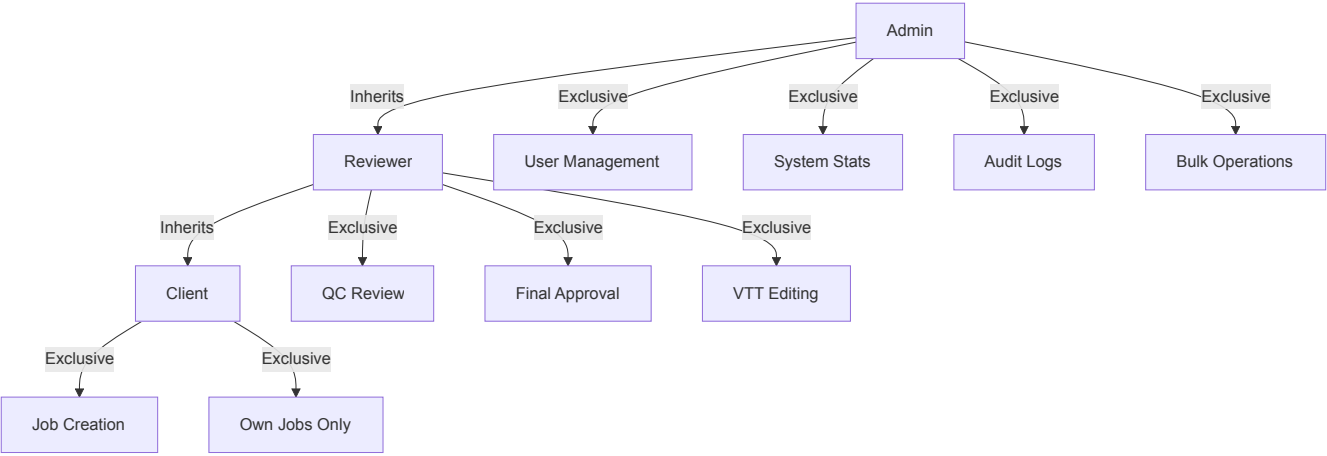
- Emergency function for stuck or failed jobs
- Resets job to "created" status
- Triggers full ingestion pipeline again
- Overwrites existing results
- Use cases:
 - AI generated incorrect content
 - Processing interrupted mid-pipeline
 - Updated AI prompts require regeneration

Bulk Job Operations

- Bulk delete with confirmation
- Counts affected assets (videos, captions, audio)
- Itemized deletion summary
- Error handling for partial failures
- Irreversible action warnings

3. User Roles & Permissions

3.1 Role Hierarchy



3.2 Permission Matrix

Feature	Client	Reviewer	Admin
Job Management			
Create jobs			
View own jobs			
View all jobs			
Delete own jobs			
Delete any job			
Bulk delete jobs			
Reprocess jobs			
Quality Control			
Access QC queue			
Edit VTT content			
Approve English content			
Reject jobs			
Adjust VTT timing			
Final Review			
Access final queue			
Validate assets			
Approve for delivery			
Return for QC			
Downloads			
Download own jobs			
Download any job			
Administration			
User management			
System statistics			
Audit log access			
Health monitoring			

3.3 Access Control Implementation

Authentication Layer

- JWT token-based authentication
- HttpOnly cookies for refresh tokens
- Token expiration and automatic refresh
- Secure session management

Authorization Layer

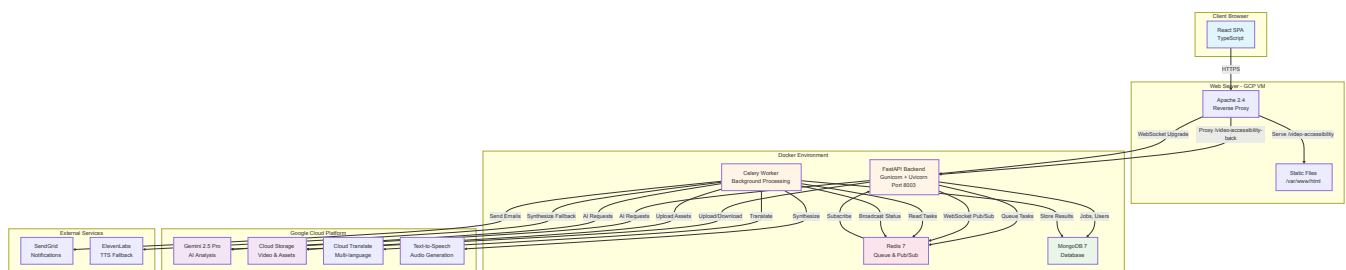
- Route-level protection (React Router guards)
- API endpoint protection (FastAPI dependencies)
- Database query filtering (client_id restrictions)
- Resource-level access checks

Security Boundaries

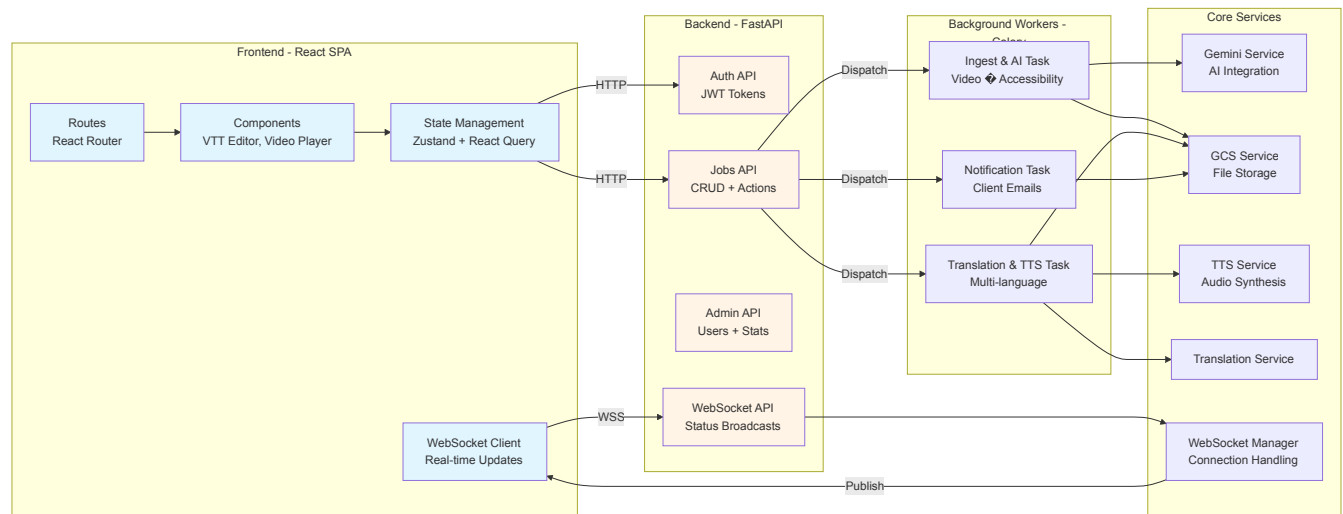
- Clients cannot access other clients' jobs
- Reviewers have read-only access to job data (except VTT editing)
- Admins have full CRUD access to all resources
- Audit logging for all privileged operations

4. System Architecture

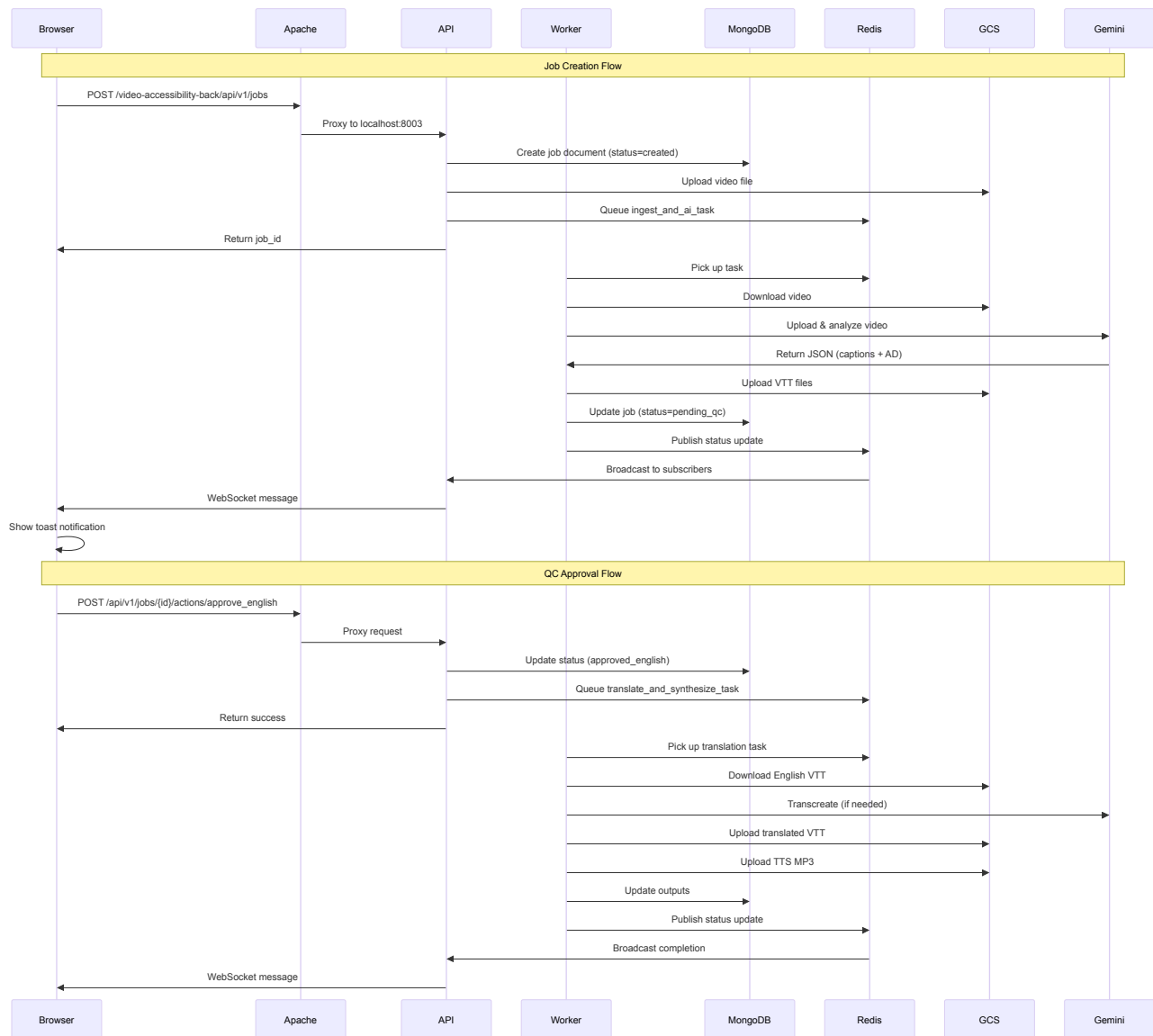
4.1 High-Level Architecture



4.2 Component Architecture

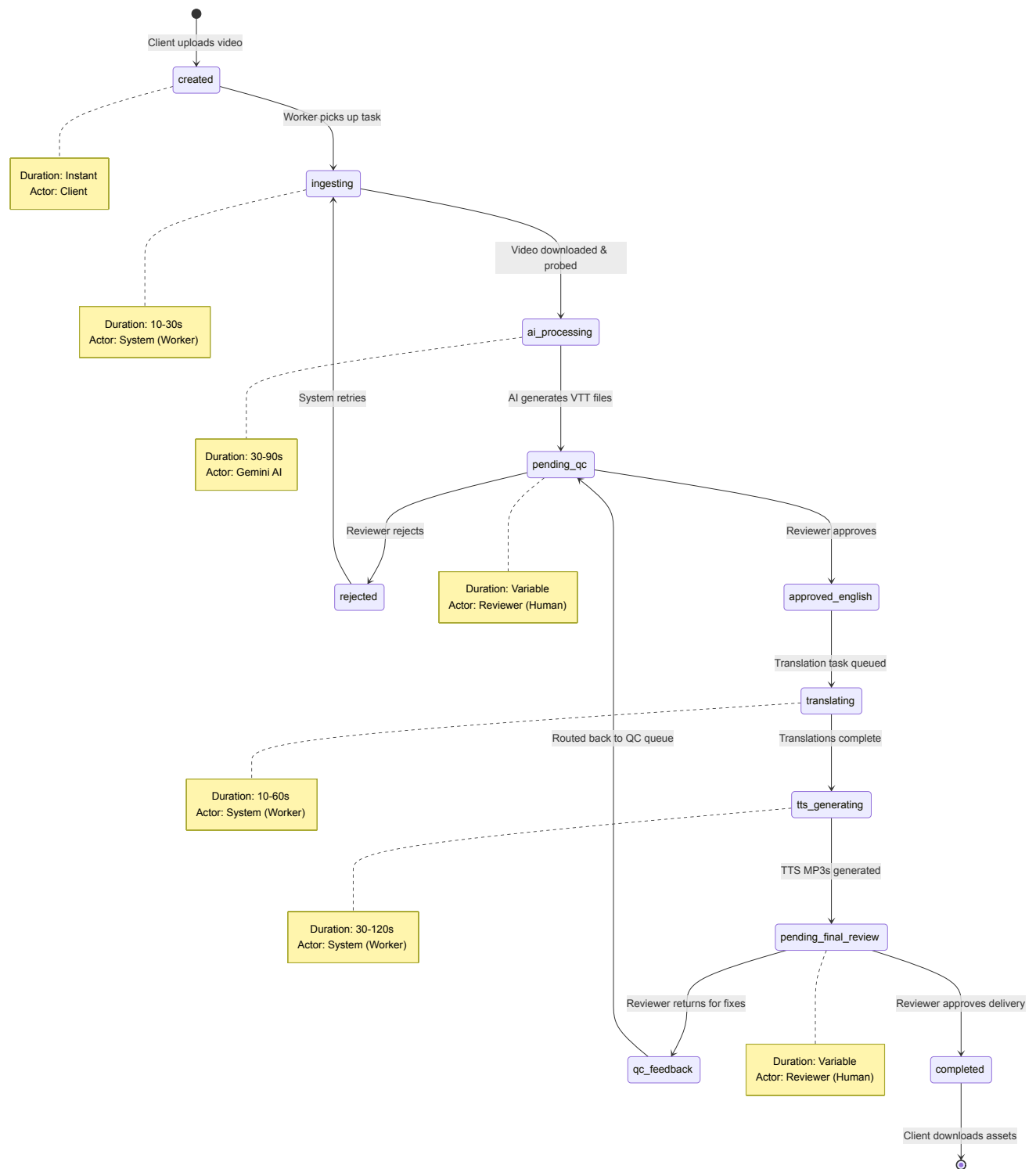


4.3 Request Flow Architecture

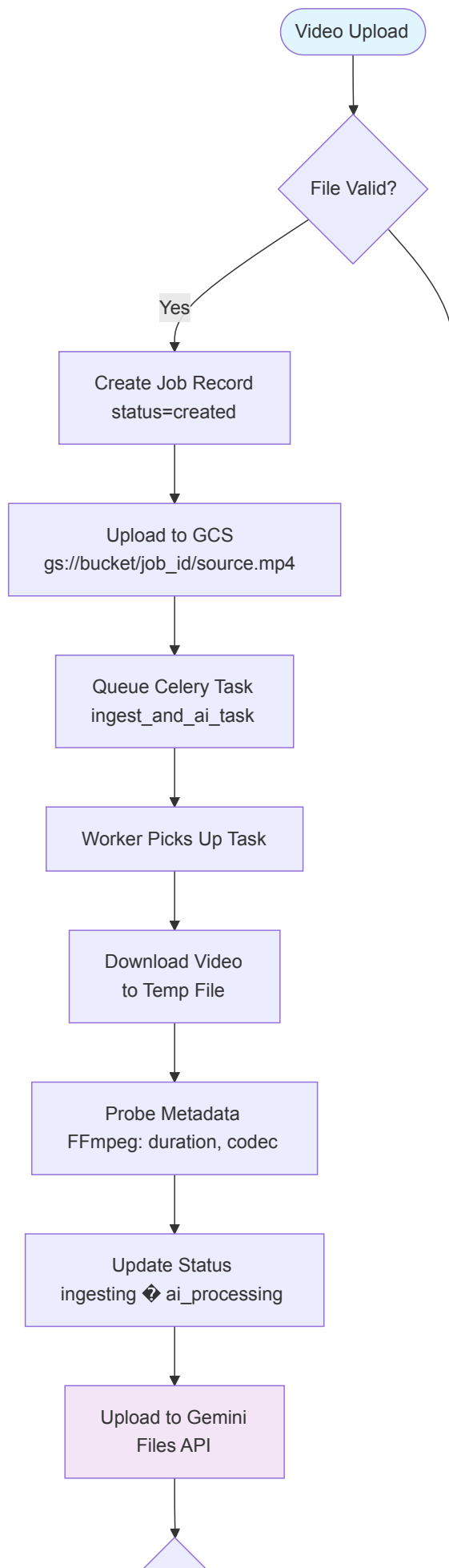


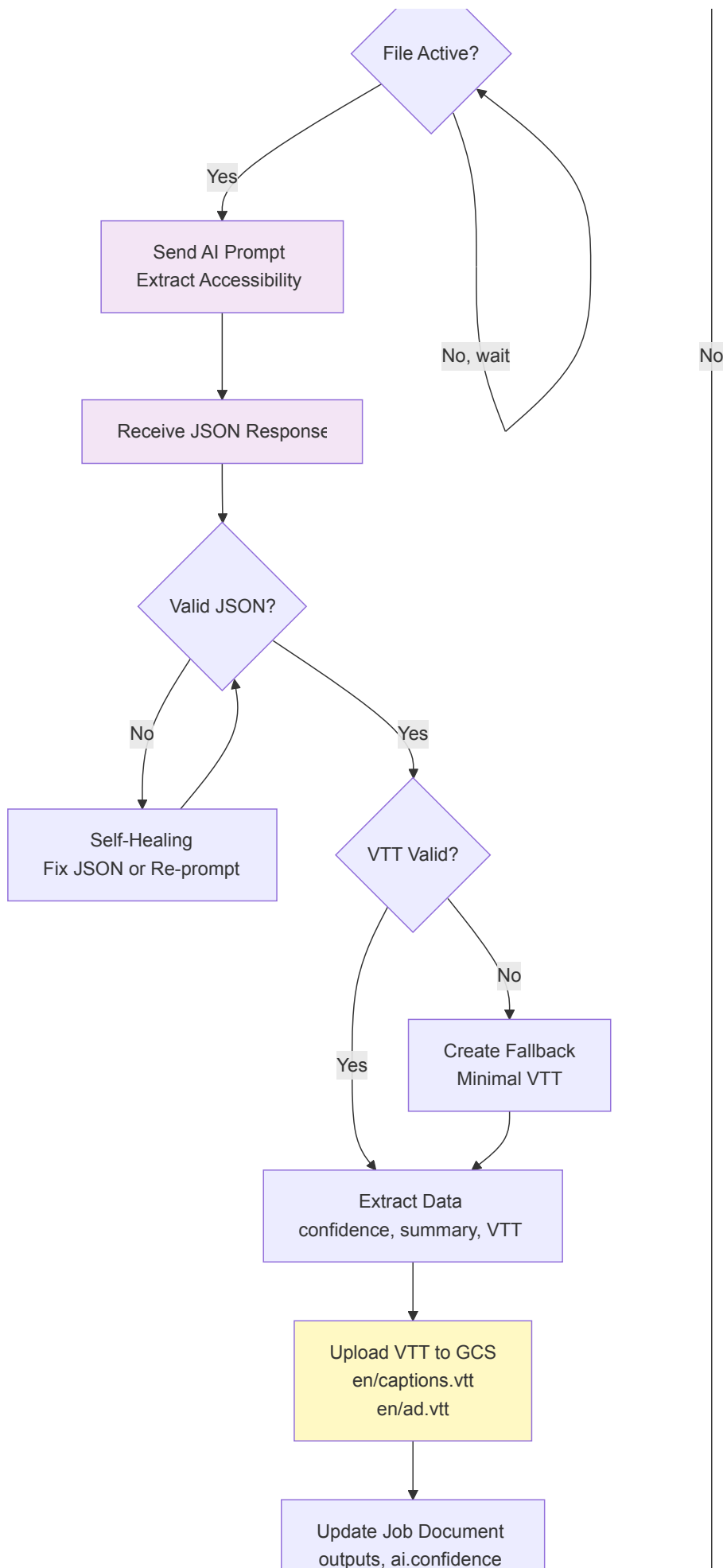
5. Process Flows & Workflows

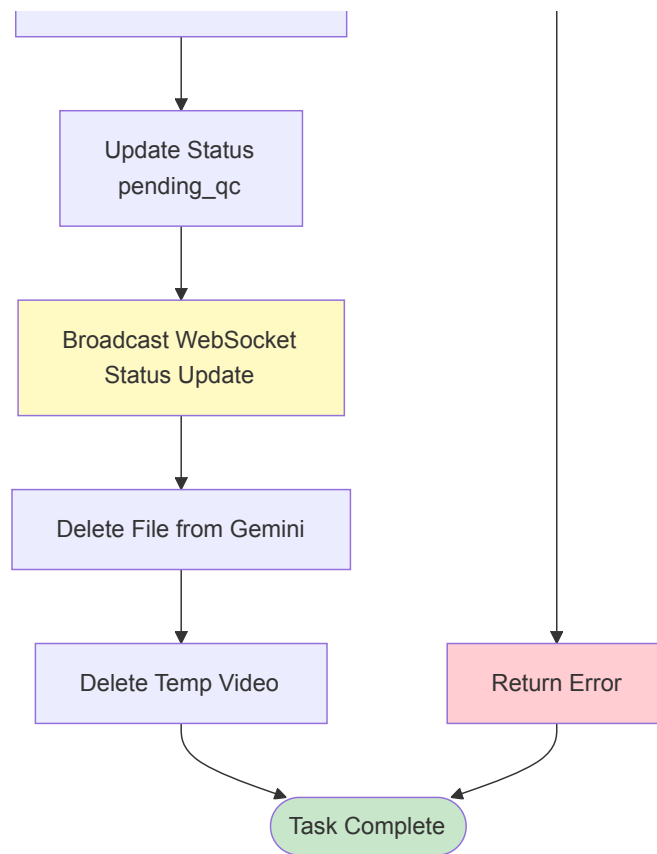
5.1 Complete Job Processing Flow



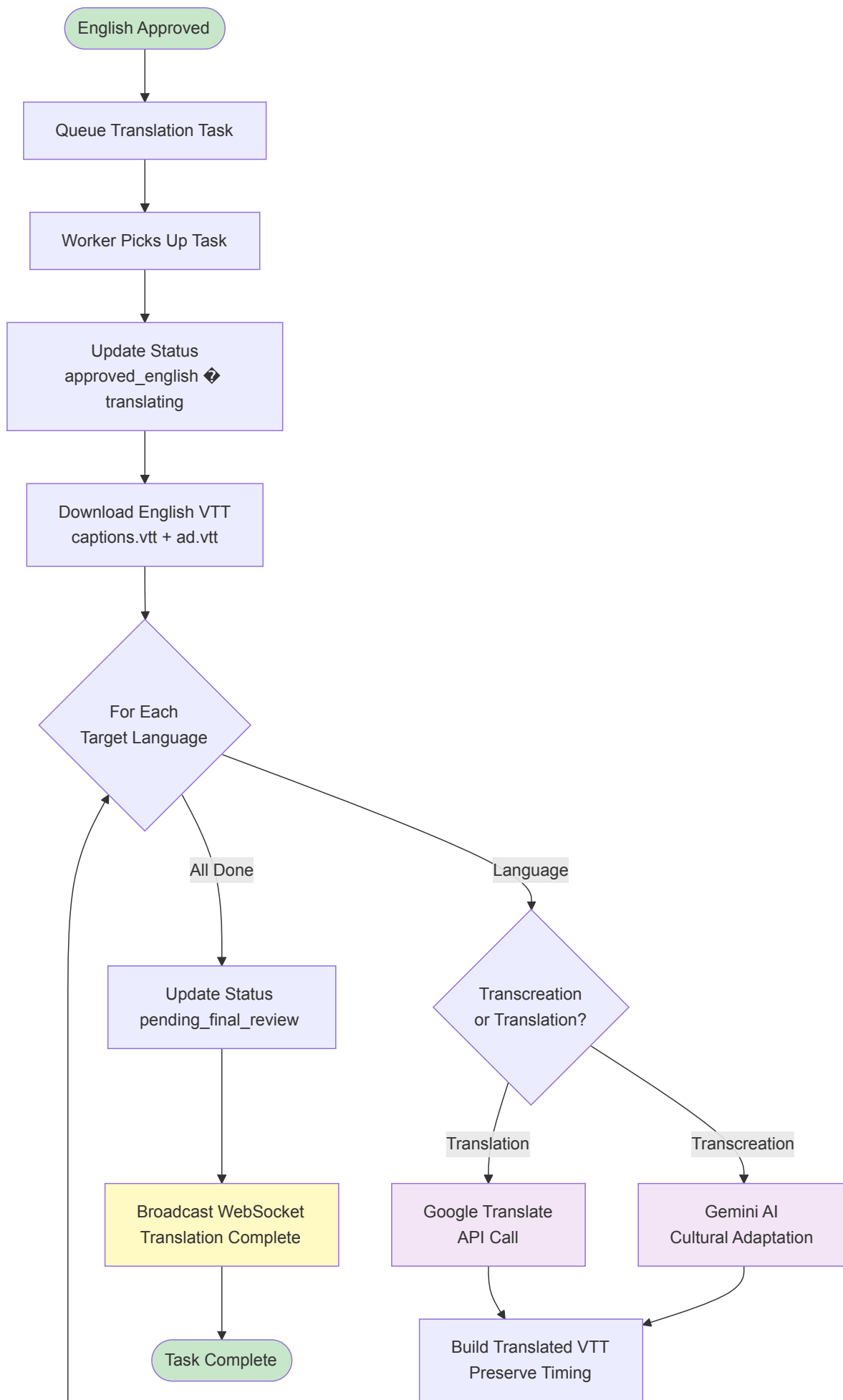
5.2 AI Processing Pipeline Detail

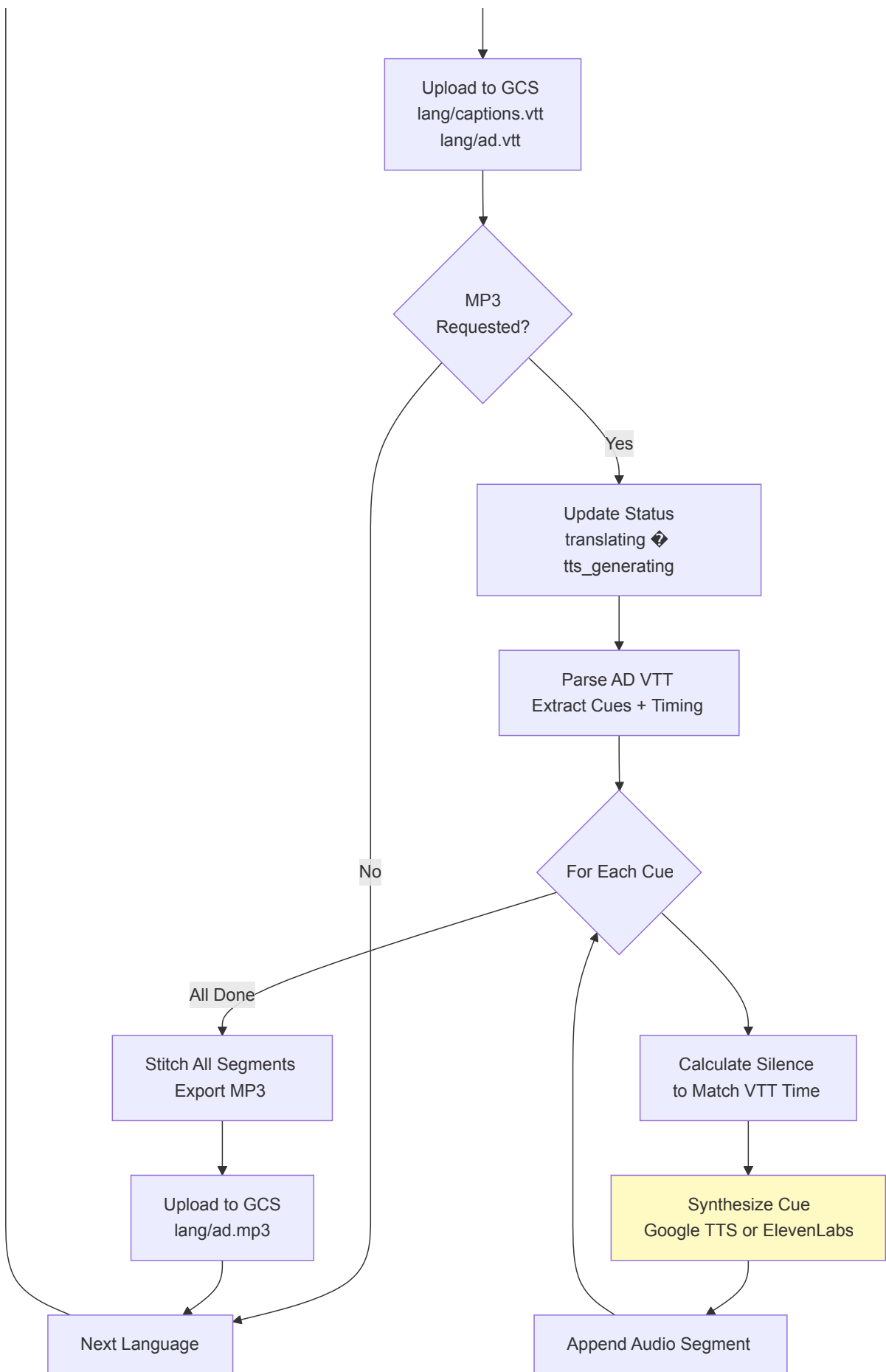




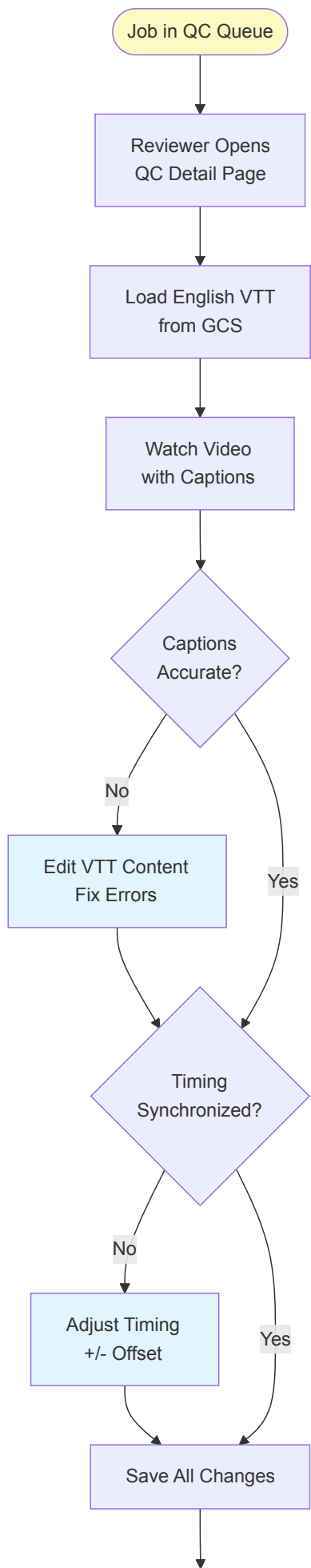


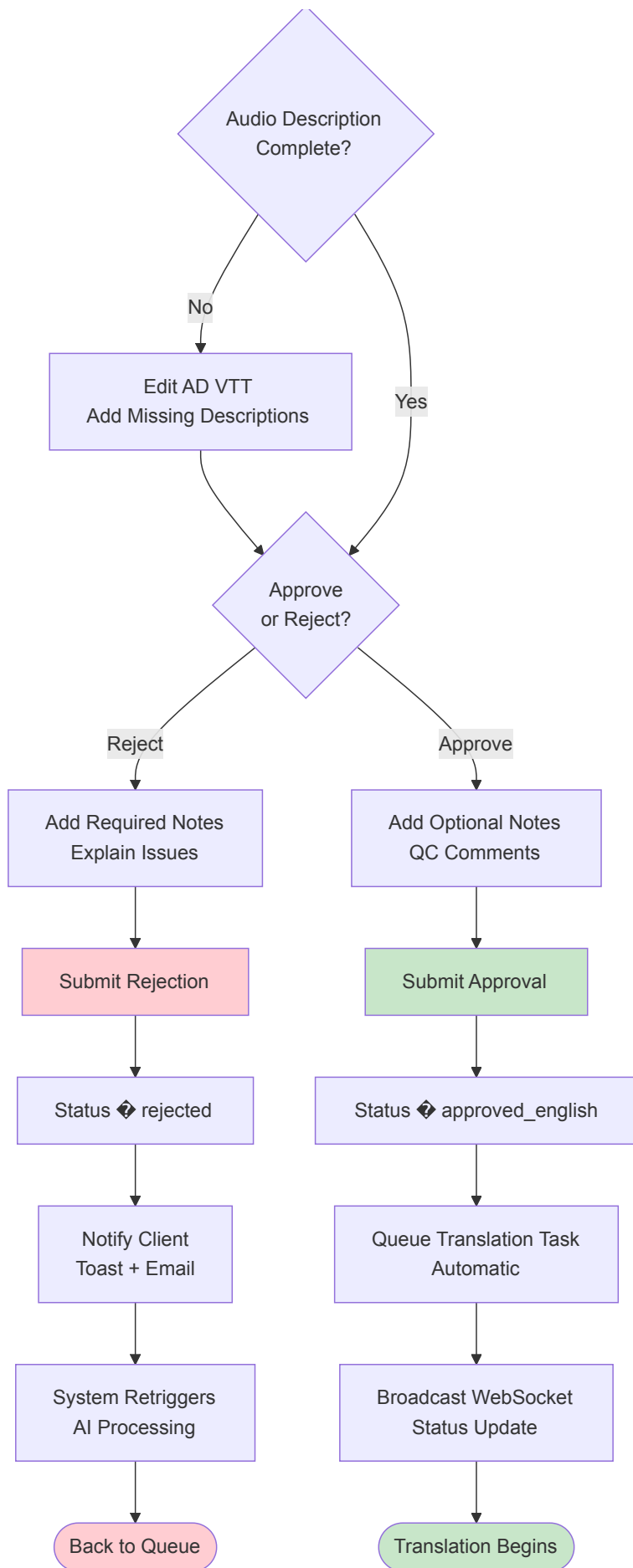
5.3 Translation & TTS Pipeline

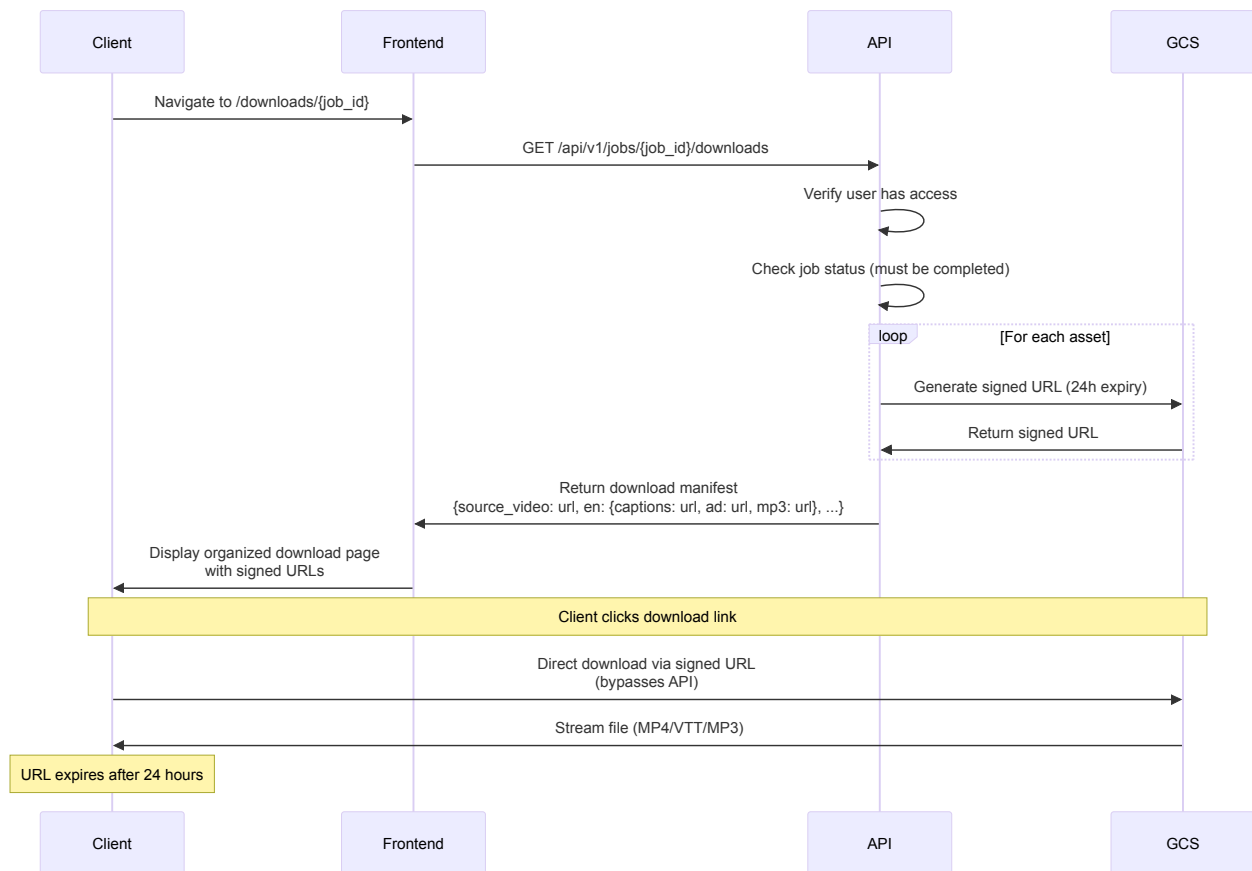




5.4 Quality Control Decision Flow

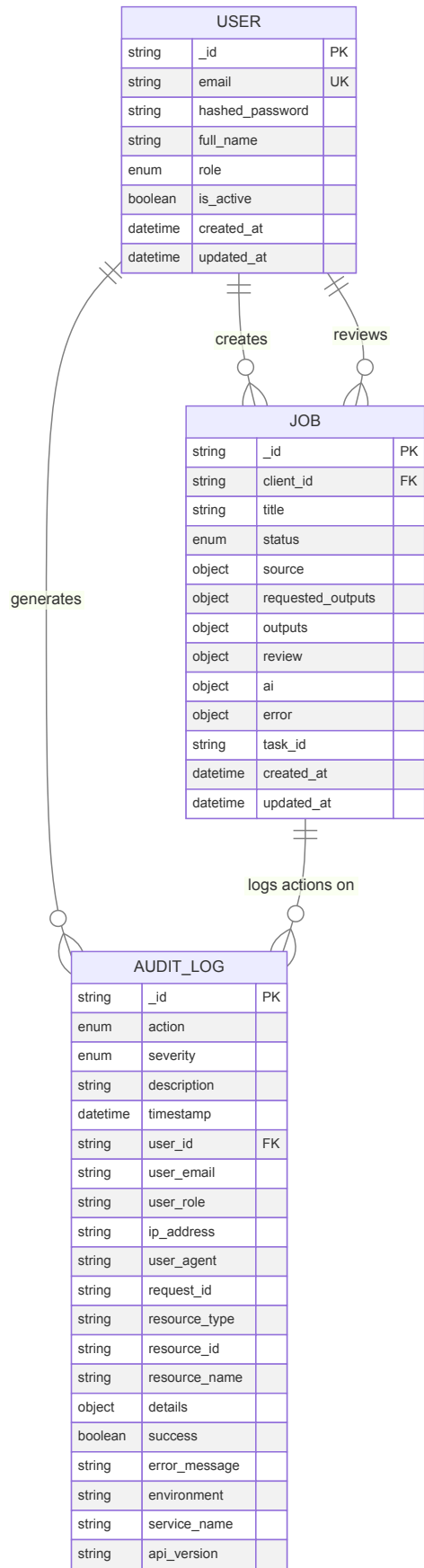






6. Database Schema

6.1 Entity Relationship Diagram



6.2 Job Document Structure

Primary Fields

- `_id` (string) - Unique job identifier
- `client_id` (string) - Foreign key to users collection
- `title` (string) - Job name (user-provided)
- `status` (enum) - Current pipeline stage
- `task_id` (string) - Celery task ID for monitoring

Source Object

```
{
  "filename": "source.mp4",
  "original_filename": "Corporate_Training_Q4.mp4",
  "gcs_uri": "gs://accessible-video/68e7.../source.mp4",
  "duration_s": 525.4,
  "language": "en"
}
```

Requested Outputs Object

```
{
  "captions_vtt": true,
  "audio_description_vtt": true,
  "audio_description_mp3": true,
  "languages": ["en", "es", "fr"],
  "transcreation": ["es"]
}
```

Outputs Object (per language)

```
{
  "en": {
    "captions_vtt_gcs": "gs://accessible-video/68e7.../en/captions.vtt",
    "ad_vtt_gcs": "gs://accessible-video/68e7.../en/ad.vtt",
    "ad_mp3_gcs": "gs://accessible-video/68e7.../en/ad.mp3"
  },
  "es": {
    "captions_vtt_gcs": "gs://accessible-video/68e7.../es/captions.vtt",
    "ad_vtt_gcs": "gs://accessible-video/68e7.../es/ad.vtt",
    "ad_mp3_gcs": "gs://accessible-video/68e7.../es/ad.mp3",
    "origin": "transcreate",
    "qa_notes": ""
  }
}
```

Review Object

```
{
  "notes": "Fixed timing issues. Content accurate.",
  "reviewer_id": "reviewer-001",
  "history": [
    {
      "at": "2025-01-15T14:30:00Z",
```

```

    "status": "pending_qc",
    "by": "system",
    "notes": ""
  },
  {
    "at": "2025-01-15T14:45:22Z",
    "status": "approved_english",
    "by": "reviewer-001",
    "notes": "Fixed timing issues. Content accurate."
  }
]
}

```

AI Object

```

{
  "confidence": 0.94,
  "ingestion_json": {
    "language": "en",
    "confidence": 0.94,
    "summary": "Corporate training video...",
    "transcript_plaintext": "Welcome to Q4...",
    "captions_vtt": "WEBVTT\n\n00:00:00.000 --> ...",
    "audio_description_vtt": "WEBVTT\n\n00:00:00.000 --> ..."
  }
}

```

6.3 Database Indexes

Users Collection

- `email` (unique) - Fast user lookup during authentication
- `role` - Filter users by role for admin pages

Jobs Collection

- `status` + `created_at` (compound) - QC/review queue queries
- `client_id` - Filter jobs by owner (client view)
- `created_at` (desc) - Recent jobs first

Audit Logs Collection

- `timestamp` (desc) - Chronological queries
- `action` + `timestamp` - Filter by action type
- `user_id` + `timestamp` - User activity history
- `severity` + `timestamp` - Security event queries
- `resource_type` + `resource_id` - Resource tracking
- Full-text index on `description`, `details`, `error_message` - Search capability

6.4 File Storage Structure (GCS)

```
gs://accessible-video/
  {job_id_1}/
    source.mp4           # Original uploaded video
    en/
      captions.vtt       # English closed captions
      ad.vtt             # English audio description script
      ad.mp3             # English audio description audio
    es/
      captions.vtt       # Spanish captions (translated/transcreated)
      ad.vtt             # Spanish AD script
      ad.mp3             # Spanish AD audio
    fr/
      captions.vtt       # French captions
      ad.vtt             # French AD script
      ad.mp3             # French AD audio
  {job_id_2}/
    ...
```

Naming Conventions

- Job ID: MongoDB ObjectId (24-char hex) or UUID
- Source filename: Always `source.mp4` (normalized)
- Language codes: ISO 639-1 two-letter codes (en, es, fr, de, etc.)
- File types: `.vtt` for subtitles, `.mp3` for audio (128kbps)

Access Control

- Bucket: Private (no public access)
- Access method: Time-limited signed URLs only
- Expiration: 24 hours for downloads, 1 hour for uploads
- Signature: V4 HMAC-SHA256

7. API Overview

7.1 API Endpoints Summary

Authentication Endpoints

Method	Endpoint	Purpose	Auth Required
POST	<code>/auth/login</code>	User login, returns JWT tokens	No
POST	<code>/auth/refresh</code>	Refresh access token using cookie	Cookie
POST	<code>/auth/logout</code>	Invalidate refresh token	Yes

Job Management Endpoints

Method	Endpoint	Purpose	Roles
POST	/jobs	Create new job & upload video	All
GET	/jobs	List jobs (filtered by role)	All
GET	/jobs/{id}	Get job details	Owner/Reviewer/Admin
DELETE	/jobs/{id}	Delete job and assets	Owner/Admin
DELETE	/jobs/bulk	Bulk delete jobs	Admin

QC & Approval Endpoints

Method	Endpoint	Purpose	Roles
POST	/jobs/{id}/actions/approve_english	Approve English QC	Reviewer/Admin
POST	/jobs/{id}/actions/reject	Reject during QC	Reviewer/Admin
POST	/jobs/{id}/actions/complete	Final approval for delivery	Reviewer/Admin
POST	/jobs/{id}/actions/reject_final	Return for QC fixes	Reviewer/Admin

Asset Management Endpoints

Method	Endpoint	Purpose	Roles
GET	/jobs/{id}/downloads	Get signed download URLs	Owner/Reviewer/Admin
GET	/jobs/{id}/vtt?language={lang}	Fetch VTT for editing	Reviewer/Admin
PATCH	/jobs/{id}/vtt	Update VTT content	Reviewer/Admin
POST	/jobs/{id}/vtt/adjust-timing	Bulk timing shift	Reviewer/Admin
GET	/jobs/{id}/validate	Validate all assets exist	Reviewer/Admin

Admin Endpoints

Method	Endpoint	Purpose	Role
GET	/admin/users	List all users	Admin
POST	/admin/users	Create new user	Admin
PATCH	/admin/users/{id}	Update user	Admin
DELETE	/admin/users/{id}	Deactivate user	Admin
GET	/admin/stats	System statistics	Admin
GET	/admin/health/detailed	Health check	Admin
GET	/admin/audit-logs	Query audit trail	Admin
POST	/admin/maintenance/reprocess-job/{id}	Emergency reprocess	Admin

WebSocket Endpoints

Protocol	Endpoint	Purpose	Auth
WS	/ws/jobs	Global job list updates	Token (query param)
WS	/ws/jobs/{id}	Specific job updates	Token (query param)
GET	/ws/status	Connection statistics	Admin

7.2 Response Formats

Standard Success Response

```
{
  "id": "68e70...",
  "title": "Corporate Training Q4",
  "status": "pending_qc",
  "created_at": "2025-01-15T14:30:00Z",
  "updated_at": "2025-01-15T14:35:22Z",
  ...
}
```

Paginated List Response

```
{
  "jobs": [...],
  "total": 42,
  "page": 1,
  "size": 20
}
```

Error Response

```
{
  "detail": "Job not found",
  "error_code": "NOT_FOUND"
}
```

Validation Error Response

```
{
  "detail": [
    {
      "loc": ["body", "title"],
      "msg": "field required",
      "type": "value_error.missing"
    }
  ]
}
```

8. AI Processing Pipeline

8.1 Gemini AI Integration

Model: Google Gemini 2.5 Pro (gemini-2.5-pro)

Capabilities Used

- **Multimodal Understanding:** Processes video (visual + audio)
- **Speech Recognition:** Transcribes dialogue with speaker attribution
- **Visual Analysis:** Identifies scenes, actions, on-screen text
- **Structured Output:** Returns JSON with strict schema adherence
- **Self-Correction:** Can fix its own malformed JSON outputs

Processing Steps

1. Video Upload to Gemini

- Uploads MP4 via Gemini Files API
- Sets display name: `video_processing_{filename}`
- Specifies MIME type: `video/mp4`
- Receives file reference with URI

2. File State Monitoring

- Polls file status: PENDING ↔ PROCESSING ↔ ACTIVE
- Exponential backoff (1s, 1.5s, 2.25s, max 30s)
- Maximum wait: 300 seconds (5 minutes)
- Fails if file doesn't become ACTIVE

3. Prompt Engineering

- Loads template from `/app/prompts/gemini_ingestion.md`
- Multi-modal prompt (text instructions + video URI)
- Requests JSON output with specific schema
- Specifies VTT format requirements

4. Response Processing

- Receives response (may be markdown-wrapped JSON)
- Strips markdown code fences if present
- Parses JSON
- Validates required fields present
- Checks VTT format (must start with "WEBVTT")

5. Self-Healing Mechanism

- Detects JSON parse errors
- Attempts automatic fixes:
 - Removes trailing commas
 - Closes unterminated strings
 - Adds missing closing braces
- Falls back to re-prompting Gemini to fix its own JSON
- Creates fallback content if critical fields missing

6. File Cleanup

- Deletes uploaded file from Gemini (guaranteed in finally block)
- Prevents quota exhaustion
- Cleans up even on task failure/cancellation

Expected Output Format

```
{
  "language": "en",
  "confidence": 0.94,
  "summary": "Brief video description (2-3 sentences)",
  "transcript_plaintext": "Full transcript without timing",
  "captions_vtt": "WEBVTT\n\n00:00:00.000 --> 00:00:04.500\nWelcome to...",
  "audio_description_vtt": "WEBVTT\n\n00:00:00.000 --> 00:00:02.000\nA conference room..."
}
```

AI Confidence Scoring

- Range: 0.0 - 1.0 (displayed as percentage)
- Threshold for alerts: <0.70 (70%)
- Blocks completion if confidence < 0.70
- Displayed prominently in QC interface

8.2 Translation Processing

Standard Translation (Google Cloud Translate)

Process:

1. Parse English VTT to extract cue texts
2. Batch translate all texts to target language
3. Rebuild VTT structure with:
 - Translated text
 - Original timing (preserved exactly)
 - Original cue IDs

Use Cases:

- Informational content
- Technical documentation
- Educational videos
- When verbatim accuracy is priority

Transcreation (Gemini AI)

Process:

1. Send English VTT pair (captions + AD) to Gemini
2. Provide cultural adaptation instructions
3. Specify target language and brand guidelines
4. Receive culturally adapted VTT with timing preserved

Use Cases:

- Marketing content
- Brand messaging
- Cultural-specific content

- When local resonance is priority

Differences:

- Translation: Word-for-word accuracy
- Transcreation: Meaning and cultural adaptation
- Both preserve VTT timing structure
- Transcreation slower but higher quality

8.3 Text-to-Speech Generation

Service Providers

- **Primary:** Google Cloud Text-to-Speech (Neural2 voices)
- **Fallback:** ElevenLabs (Multilingual v2 model)

Voice Configuration

Configurable per language in settings:

- English (en-US): en-US-Neural2-D
- Spanish (es-ES): es-ES-Neural2-A
- French (fr-FR): fr-FR-Neural2-A
- German (de-DE): de-DE-Neural2-B

Synthesis Algorithm

Per-Cue Processing:

1. Parse AD VTT to extract cues with timing:

```
Cue 1: 00:00:00.000 ➡ 00:00:02.500 (2.5s): "A conference room with glass walls..."
Cue 2: 00:00:05.000 ➡ 00:00:07.000 (2.0s): "John enters wearing a blue suit..."
```

2. For each cue:

- **Calculate silence needed:** If current audio position is 0.0s and cue starts at 0.0s ➡ no silence
- **Add silence:** If cue starts at 5.0s but audio position is 2.5s ➡ add 2.5s silence
- **Synthesize text:** Send cue text to TTS API
 - Voice: Language-specific
 - Speaking rate: 1.2x (natural conversational pace)
 - Pitch: default
 - Volume: normalized
- **Append audio:** Add synthesized audio to timeline
- **Update position:** current_audio_position += actual_audio_duration

3. **Timing Anchoring:**

- VTT timing is authoritative
- Audio segments anchored to VTT start times
- Silence fills gaps to maintain sync
- Actual audio duration may differ from VTT duration (that's OK)

4. **Audio Stitching:**

- Combine all segments (silence + speech)
- Export as single MP3 file

- Bitrate: 128 kbps
- Sample rate: 24kHz
- Mono channel

Retry Strategy:

- 3 attempts per cue
- Exponential backoff with jitter (1.5s, 2.7s, max 5s)
- Per-cue error isolation (one failure doesn't break entire job)
- Errors stored in `qa_notes` for reviewer attention

Fallback Logic:

1. Try Google TTS
2. If fails and ElevenLabs configured 💎 try ElevenLabs
3. If both fail 💎 mark language with error in `qa_notes`
4. Continue processing other languages

8.4 Validation & Quality Checks

VTT Format Validation

- Must start with "WEBVTT" header
- Timing format: `HH:MM:SS.mmm --> HH:MM:SS.mmm`
- Start time < end time for every cue
- No overlapping cues
- No empty cue text
- Valid timestamp ranges (00:00:00.000 - 99:59:59.999)

Asset Completeness Validation

- All requested languages have outputs
- VTT files exist in GCS
- MP3 files exist if requested
- File sizes reasonable (VTT: 1KB-10MB, MP3: 10KB-500MB)
- Minimum content: At least 1 cue in each VTT

AI Confidence Thresholds

- Minimum acceptable: 70%
- Typical range: 85-98%
- <70%: Blocks completion, requires manual review
- Confidence displayed prominently in QC interface

Pre-Completion Checks

- All validation rules pass
- No errors in `outputs.{lang}.qa_notes`
- Review history shows approval chain
- All requested assets generated successfully

9. Real-time Features

9.1 WebSocket Architecture

Connection Model

- Single persistent connection per user session
- Token-based authentication (JWT in query parameter)
- Automatic reconnection with exponential backoff
- Heartbeat mechanism (30-second ping/pong)
- Maximum 5 reconnection attempts before failure

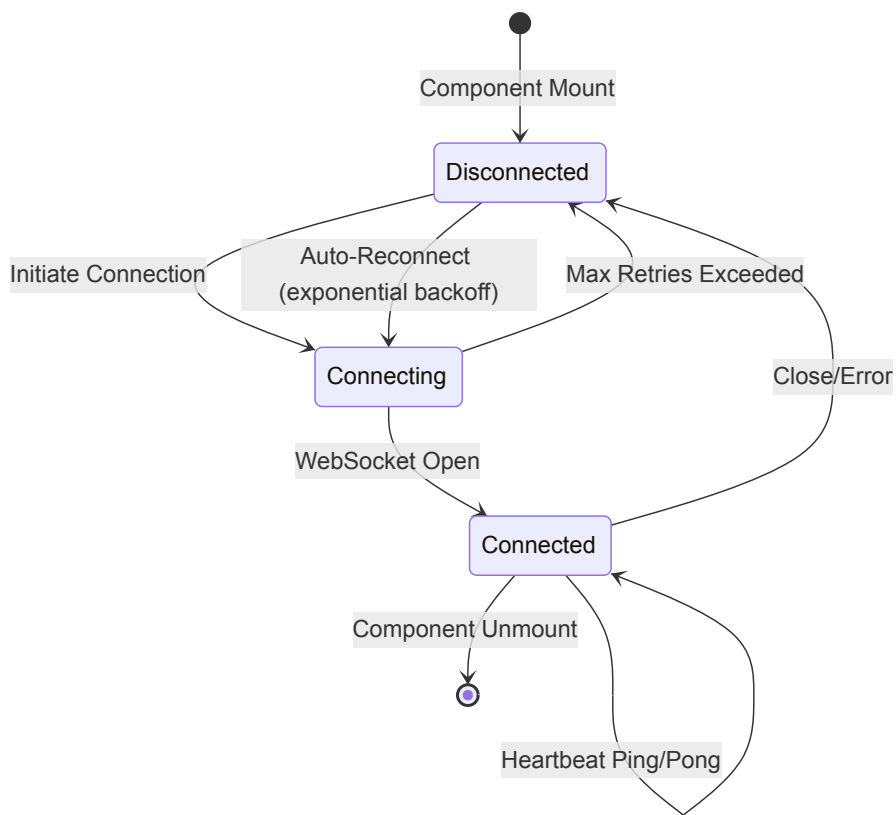
Channel Architecture

```
Redis Pub/Sub Channels:  
job_status_updates (global)  
    All job updates for all users  
job_status_updates:{job_id} (specific)  
    Updates for specific job only
```

Message Broadcasting Flow

1. Worker completes task stage
2. Worker publishes to Redis channel
3. API WebSocket manager subscribes to Redis
4. Manager filters by user eligibility:
 - Job creator (client)
 - Reviewers in history
 - All admin users
5. Manager broadcasts to eligible WebSocket connections
6. Frontend receives message and updates UI

Connection Lifecycle



9.2 Status Update Messages

Message Types

connection_established

```
{
  "type": "connection_established",
  "job_id": "68e7...",
  "timestamp": "2025-01-15T14:30:00.123Z"
}
```

job_status_update (specific job)

```
{
  "type": "job_status_update",
  "data": {
    "job_id": "68e7...",
    "status": "pending_qc",
    "updated_at": "2025-01-15T14:35:22.456Z",
    "job_title": "Corporate Training Q4",
    "message": "Ready for quality control review",
    "progress": 100,
    "metadata": {
      "confidence": 0.94,
      "language": "en"
    }
  }
}
```

```
}  
}
```

job_list_update (global)

```
{  
  "type": "job_list_update",  
  "data": {  
    "job_id": "68e7...",  
    "status": "completed",  
    "updated_at": "2025-01-15T15:45:00.789Z",  
    "job_title": "Corporate Training Q4",  
    "message": "Job completed and ready for download"  
  }  
}
```

9.3 User Notification System

Toast Notifications (Temporary)

- Auto-dismiss after 5 seconds
- Types: Success (green), Info (blue), Warning (yellow), Error (red)
- Positioned top-right corner
- Stack multiple toasts
- Dismiss button for manual close

Persistent Notifications (Menu)

- Stored in context (survives page refresh)
- Bell icon with unread count badge
- Dropdown menu (max 10 visible, scrollable)
- Per-notification actions: Mark read, Remove, View job
- Bulk actions: Mark all read, Clear all
- Color-coded by type (blue dot = unread)

Notification Triggers

- Job status changes to key states (pending_qc, completed, rejected)
- System messages (maintenance, updates)
- Error conditions requiring attention
- Bulk operation completions

10. Deployment Architecture

10.1 Production Deployment Overview

Hosting Environment

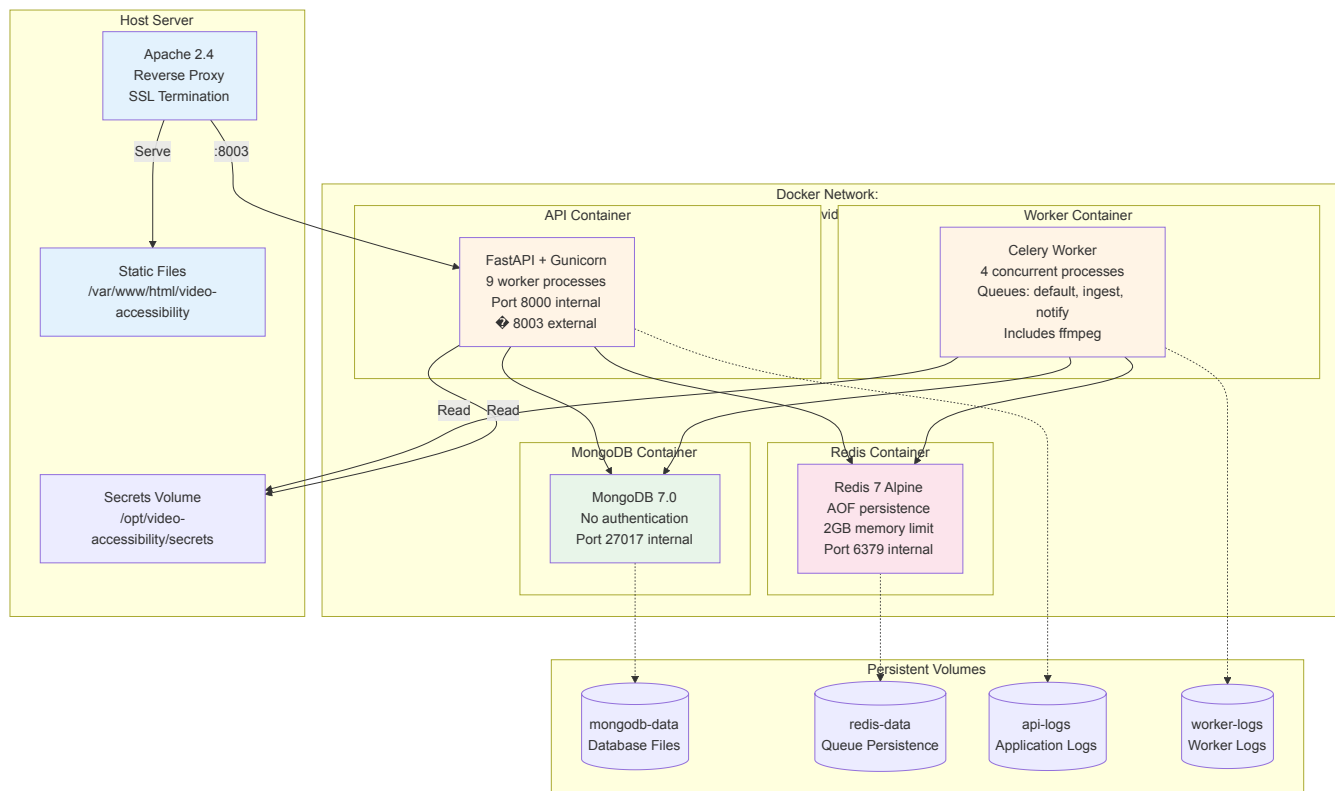
- **Platform:** Google Cloud Platform (GCP) VM

- **Server Specs:** 8 CPU cores, 32GB RAM
- **Operating System:** Linux (Ubuntu/Debian)
- **Domain:** ai-sandbox.oliver.solutions
- **SSL:** Wildcard certificate (*.ai-sandbox.oliver.solutions)

Deployment Model

- Docker Compose orchestration (single-server)
- Apache 2.4 as reverse proxy
- Frontend served as static files
- Backend services containerized
- No downtime required for updates

10.2 Docker Container Architecture



10.3 URL Routing Structure

Production URLs

- **Frontend (React SPA):** <https://ai-sandbox.oliver.solutions/video-accessibility>
- **Backend API:** <https://ai-sandbox.oliver.solutions/video-accessibility-back>
- **WebSocket:** `wss://ai-sandbox.oliver.solutions/video-accessibility-back/api/v1/ws`

Apache Configuration

- Frontend: Static file serving with SPA routing
- Backend: Reverse proxy to Docker container (localhost:8003)
- WebSocket: Proxy upgrade for real-time connections

React Router Base Path

- Base: `/video-accessibility/`
- All routes relative to base (e.g., `/video-accessibility/jobs`)
- Configured in `vite.config.ts` and `App.tsx`

10.4 Resource Allocation

Container Resource Limits (Docker Compose)

Container	Memory Limit	CPU Limit	Purpose
API	4GB	2 cores	HTTP request handling
Worker	8GB	4 cores	Video processing (CPU/memory intensive)
MongoDB	4GB	1 core	Database operations
Redis	2GB	0.5 core	Queue & cache

Total Allocated: 18GB RAM, 7.5 CPU cores
System Overhead: ~12GB RAM, 0.5 CPU cores
Server Capacity: 32GB RAM, 8 CPU cores (comfortable headroom)

10.5 Deployment Scripts

Available Scripts (in `/opt/video-accessibility/scripts/`)

- 1. full-deploy.sh** - Complete deployment
 - Rebuilds all Docker images
 - Restarts all containers
 - Builds and deploys frontend
 - Verifies deployment success
 - Usage: `sudo ./scripts/full-deploy.sh`
 - Time: ~10-15 minutes
- 2. full-deploy.sh --frontend-only** - Frontend-only deployment
 - Skips Docker rebuild
 - Only rebuilds React application
 - Deploys to Apache document root
 - Usage: `sudo ./scripts/full-deploy.sh --frontend-only`
 - Time: ~2-3 minutes
- 3. build-frontend.sh** - Frontend build & deploy
 - `npm ci` ➡ `npm run build` ➡ `deploy`
 - Sets correct ownership/permissions
 - Creates timestamped backups
 - Usage: `./scripts/build-frontend.sh`
- 4. mongodb-init.js** - Database initialization
 - Creates collections with schema validation
 - Creates performance indexes
 - One-time setup script
 - Usage: `docker compose exec mongodb mongosh < scripts/mongodb-init.js`

Deployment Workflow (Manual Steps)

```
# 1. Pull latest code (as user, not sudo)
cd /opt/video-accessibility
git pull origin main
cd backend && git pull origin main && cd ..
cd frontend && git pull origin main && cd ..

# 2. Run deployment script (with sudo)
sudo ./scripts/full-deploy.sh

# OR for frontend-only updates (faster):
sudo ./scripts/full-deploy.sh --frontend-only
```

For detailed deployment procedures, see: [DEPLOYMENT.md](#)

10.6 Environment Configuration

Environment Variables (`.env` file)

Application Settings

- `APP_ENV` - Environment (dev/prod)
- `API_BASE_URL` - Backend API URL
- `CLIENT_BASE_URL` - Frontend URL (for emails)

Authentication

- `JWT_SECRET` - Token signing secret (64 characters)
- `JWT_ACCESS_TTL_MIN` - Access token lifetime (default: 240 min)
- `JWT_REFRESH_TTL_DAYS` - Refresh token lifetime (default: 7 days)
- `COOKIE_DOMAIN` - Cookie domain (ai-sandbox.oliver.solutions)
- `COOKIE_SECURE` - HTTPS-only cookies (true in prod)
- `COOKIE_SAMESITE` - CSRF protection (Lax)

Database

- `MONGODB_URI` - Connection string (mongodb://mongodb:27017/accessible_video)
- `MONGODB_DB` - Database name (accessible_video)
- `REDIS_URL` - Redis connection (redis://redis:6379/0)

Google Cloud Platform

- `GCP_PROJECT_ID` - GCP project identifier
- `GCS_BUCKET` - Storage bucket name (accessible-video)
- `GOOGLE_APPLICATION_CREDENTIALS` - Path to service account JSON
- `GEMINI_API_KEY` - Gemini AI API key
- `GOOGLE_TTS_CREDENTIALS` - TTS credentials path (same as above)

Optional Services

- `TRANSLATE_API_KEY` - Google Translate API key
- `ELEVENLABS_API_KEY` - ElevenLabs TTS key
- `SENDGRID_API_KEY` - Email service key
- `SENTRY_DSN` - Error tracking (disabled)

10.7 Secrets Management

GCP Service Account Credentials

- **Location:** `/opt/video-accessibility/secrets/gcp-credentials.json`
- **Mounted:** Read-only volume in containers (`/secrets/gcp-credentials.json`)
- **Permissions:** 644 (readable by container user)
- **Required Roles:**
 - Storage Admin (GCS operations)
 - Text-to-Speech User (TTS generation)
 - AI Platform User (Gemini API access - via API key instead)

JWT Secret

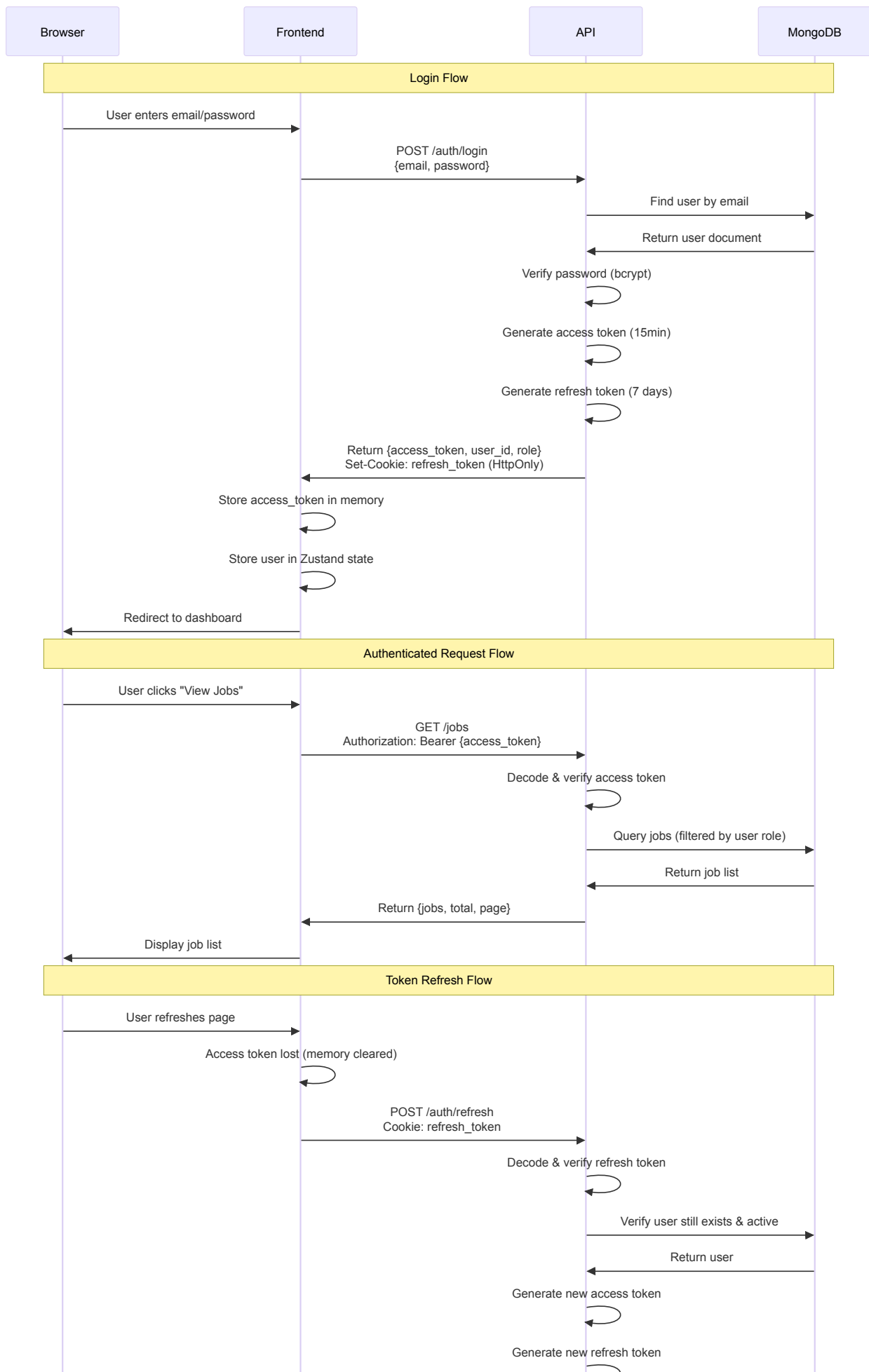
- Generated: `openssl rand -hex 32` (produces 64 characters)
- Stored: `.env` file with 600 permissions
- Shared: Between API and Worker containers
- Never: Committed to git or exposed in logs

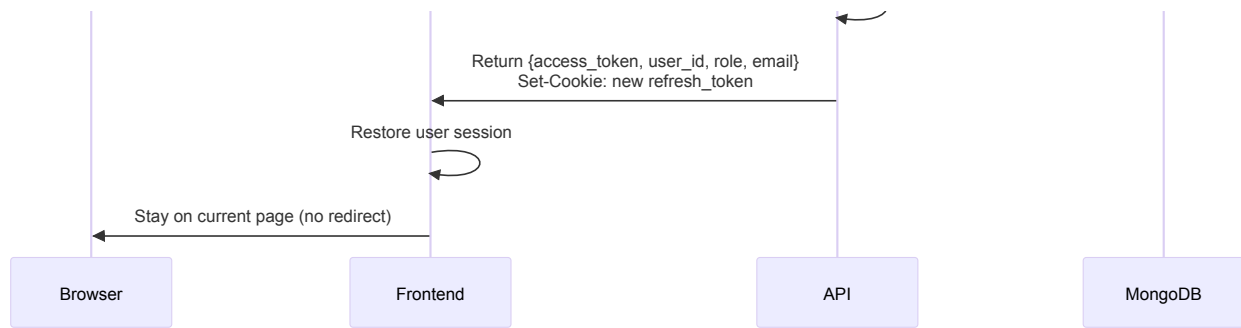
Sensitive Data Protection

- Environment variables not logged
- Secrets excluded from Docker build context
- HttpOnly cookies prevent XSS token theft
- Audit logs redact sensitive fields

11. Security Model

11.1 Authentication Flow





11.2 Authorization Enforcement

Multi-Layer Authorization

1. **Frontend Route Guards** (`RequireAuth.tsx`, `RoleGate.tsx`)
 - Prevents unauthorized page access
 - Redirects to login if not authenticated
 - Shows "Access Denied" if insufficient role
 - Attempts token refresh before redirecting
2. **API Endpoint Guards** (FastAPI dependencies)
 - `get_current_user` - Validates access token, loads user
 - `require_roles(UserRole.REVIEWER, UserRole.ADMIN)` - Role check
 - Returns 401 if not authenticated
 - Returns 403 if insufficient permissions
3. **Database Query Filtering**
 - Clients: `WHERE client_id = current_user.id`
 - Reviewers/Admins: No filter (see all jobs)
 - Automatic injection via dependency
4. **Resource-Level Checks**
 - Job access: Verify user owns job or has reviewer role
 - User updates: Prevent users from changing own role
 - Bulk operations: Admin-only with confirmation

11.3 Data Protection

Password Security

- Hashing: bcrypt via passlib library
- Never stored plain-text
- Never returned in API responses
- Cost factor automatically adjusted for security

Token Security

- Access tokens: In-memory only (lost on refresh)
- Refresh tokens: HttpOnly cookies (XSS-protected)
- Secure flag: HTTPS-only in production
- SameSite: Lax (CSRF protection)
- Token rotation: New refresh token on every refresh

File Access Security

- GCS bucket: Private (no public URLs)
- Signed URLs: Time-limited (24 hours)
- Per-request generation: No caching
- Authorization check before generation
- Automatic expiration enforcement

Audit Trail

- All authentication attempts logged
- Failed login tracking (brute force detection)
- Suspicious activity flagged (severity: CRITICAL)
- IP address and user agent captured
- Full request context stored

11.4 API Security Measures

Input Validation

- Pydantic schemas for all request bodies
- Type checking and coercion
- String length limits
- Enum validation for status/role fields
- Email format validation

CORS Configuration

- Configurable allowed origins
- Credentials support enabled
- Preflight request handling
- Origin validation on every request

Rate Limiting (Implementation Present)

- Tracked via `x-ratelimit-*` headers
- Per-endpoint rate limits
- Redis-backed counters
- Returns 429 Too Many Requests when exceeded

Security Headers

- X-Frame-Options: SAMEORIGIN (clickjacking protection)
- X-Content-Type-Options: nosniff (MIME sniffing protection)
- X-XSS-Protection: 1; mode=block
- Referrer-Policy: strict-origin-when-cross-origin

12. Technical Stack

12.1 Frontend Technology

Core Framework

- **React 19.1** - UI framework
- **TypeScript 5.8** - Type-safe JavaScript
- **Vite 7.1** - Build tool and dev server

State Management

- **Zustand 5.0** - Lightweight global state (auth)
- **TanStack Query 5.85** - Server state management, caching
- **React Router 7.8** - Client-side routing

UI Framework

- **Tailwind CSS 4.1** - Utility-first styling
- **Custom Components** - VTT editor, video player, upload dropzone

Data Fetching

- **Axios 1.11** - HTTP client with interceptors
- **WebSocket API** - Native browser WebSocket for real-time updates

Development Tools

- **Vitest 3.2** - Unit testing framework
- **Playwright 1.54** - End-to-end testing
- **ESLint 9.33** - Code linting
- **TypeScript Compiler** - Type checking

12.2 Backend Technology

Core Framework

- **FastAPI 0.115** - Modern async web framework
- **Python 3.11** - Programming language
- **Uvicorn 0.24** - ASGI server (development)
- **Gunicorn 21.2** - WSGI server with Uvicorn workers (production)

Database & Caching

- **MongoDB 7.0** - Primary database
- **Motor 3.3** - Async MongoDB driver
- **PyMongo 4.6** - Sync MongoDB driver (workers)
- **Redis 7.2** - Queue broker and cache
- **redis-py 5.0** - Redis client

Background Processing

- **Celery 5.3** - Distributed task queue
- **Redis** - Message broker and result backend

- **Async Support** - Custom AsyncTask base class for async/await

Google Cloud SDK

- **google-cloud-storage 2.10** - GCS operations
- **google-cloud-translate 3.12** - Translation API
- **google-cloud-texttospeech 2.16** - TTS API
- **google-cloud-secret-manager 2.18** - Secrets management
- **google-genai 1.31** - Gemini AI SDK

Security & Auth

- **python-jose 3.3** - JWT token handling
- **passlib 1.9** - Password hashing (bcrypt)
- **Pydantic 2.5** - Data validation and serialization
- **pydantic-settings 2.1** - Environment configuration

Media Processing

- **ffmpeg-python 0.2** - Video metadata extraction
- **pydub 0.25** - Audio manipulation and stitching
- **python-magic 0.4** - MIME type detection

HTTP Client

- **aiohttp 3.12** - Async HTTP client (ElevenLabs)
- **httpx 0.28** - Modern HTTP client (testing)

Observability (Optional)

- **sentry-sdk 1.38** - Error tracking (disabled)
- **opentelemetry-* 1.21** - Tracing and metrics (disabled in dev)
- **prometheus-client 0.19** - Metrics export

12.3 Infrastructure

Web Server

- **Apache 2.4** - Reverse proxy and static file serving
- **mod_proxy** - HTTP proxying
- **mod_proxy_wstunnel** - WebSocket proxying
- **mod_rewrite** - SPA routing
- **mod_headers** - Security headers

Containerization

- **Docker 20.10+** - Container runtime
- **Docker Compose 2.x** - Multi-container orchestration

Version Control

- **Git** - Source control
- **GitHub/GitLab** - Remote repository hosting

Build Tools

- **Poetry 1.8** - Python dependency management
- **npm 10.x** - Node package management
- **Multi-stage Dockerfiles** - Optimized image building

12.4 External Services

AI & ML Services

- **Google Gemini 2.5 Pro** - Video analysis and transcreation
- **Google Cloud Translate** - Multi-language translation (40+ languages)
- **Google Cloud Text-to-Speech** - Neural voice synthesis
- **ElevenLabs** - Premium TTS fallback

Cloud Infrastructure

- **Google Cloud Storage** - Scalable object storage
- **Google Cloud Secret Manager** - Production secrets (optional)
- **MongoDB Atlas** - Cloud MongoDB (alternative to self-hosted)
- **Redis Cloud** - Managed Redis (alternative to self-hosted)

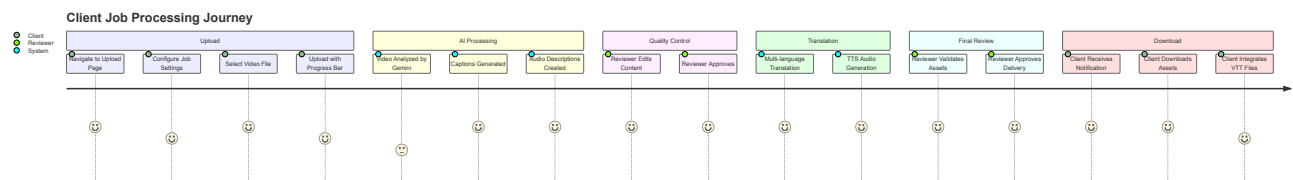
Communication

- **SendGrid** - Transactional email delivery

13. Process Flows & User Journeys

13.1 Client Journey: Upload to Download

Timeline: 15-30 minutes (varies by video length and queue depth)



Detailed Steps:

1. **Login** (30 seconds)
 - Navigate to <https://ai-sandbox.oliver.solutions/video-accessibility>
 - Enter email and password
 - Land on personalized dashboard
2. **Job Creation** (2-3 minutes)

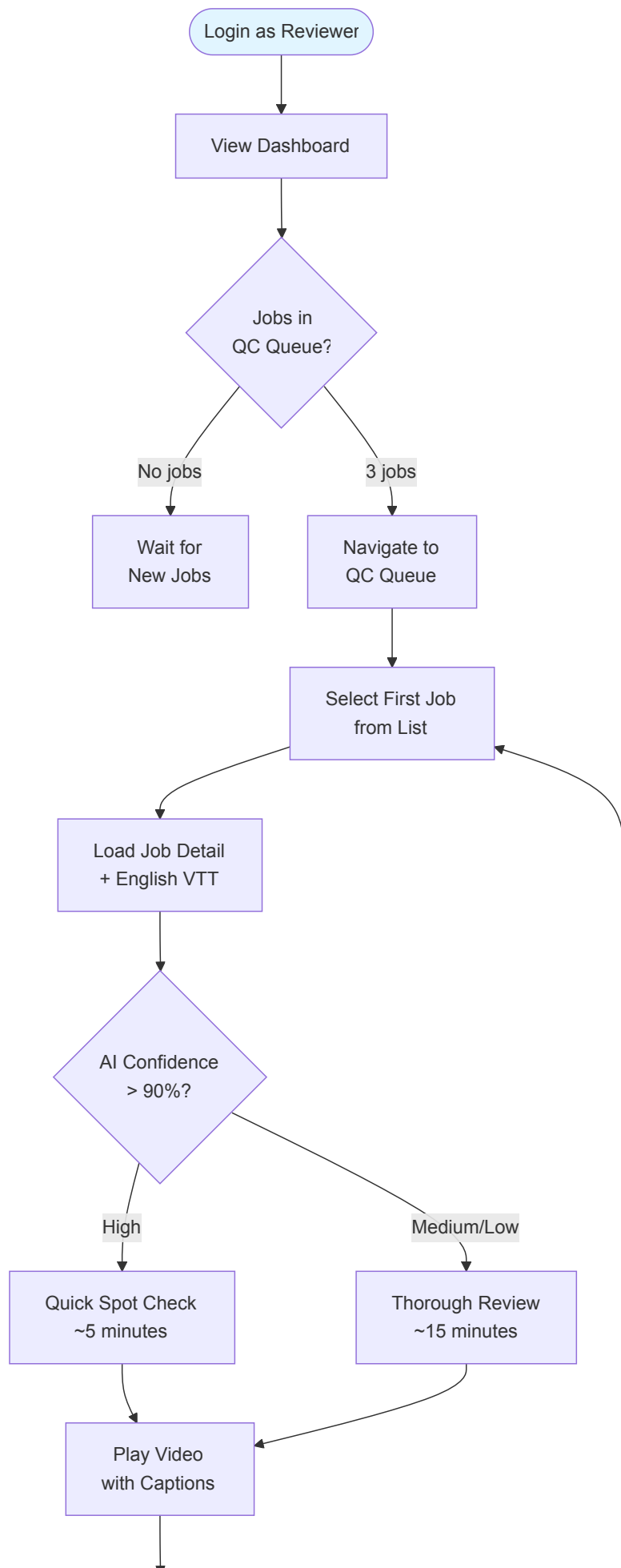
- Click "Upload New Video" button
 - Enter job title (e.g., "Q4 Training Session")
 - Select source language (English)
 - Choose outputs: Captions , AD Script , AD Audio
 - Add target languages: Spanish, French
 - Drag-drop MP4 file or browse
 - Watch upload progress: 0% ➡ 100%
 - See success message with job ID
 - Auto-redirect to job detail page
3. **Automated AI Processing** (1-3 minutes)
- Status changes: created ➡ ingesting ➡ ai_processing
 - Real-time toast notifications:
 - "Processing video..."
 - "Generating captions..."
 - Dashboard updates job count automatically
 - WebSocket keeps client informed
4. **Quality Control Phase** (10-20 minutes - human review)
- Status: pending_qc
 - Toast: "Ready for quality control"
 - Client waits (no action needed)
 - Reviewer performs QC review (separate journey)
 - Reviewer approves or requests changes
5. **Translation & TTS** (2-5 minutes per language)
- Status: approved_english ➡ translating ➡ tts_generating
 - Toasts:
 - "Translating to other languages..."
 - "Generating audio descriptions..."
 - Processes all requested languages in parallel
 - Generates MP3 files for each language
6. **Final Review Phase** (5-15 minutes - human review)
- Status: pending_final_review
 - Toast: "Ready for final review"
 - Client continues waiting
 - Reviewer validates all language assets
 - Reviewer approves for delivery
7. **Completion & Download** (5 minutes)
- Status: completed
 - Toast: "Job completed!" (with confetti animation <➡)
 - Email notification sent to client
 - Green "Download Files" button appears on job page
 - Client clicks download
 - Sees organized asset list by language
 - Downloads individual files or all
 - 24-hour window to download (URLs expire)
8. **Integration** (Client's responsibility)
- Add VTT tracks to video player
 - Test caption display
 - Test audio description playback
 - Deploy accessible video

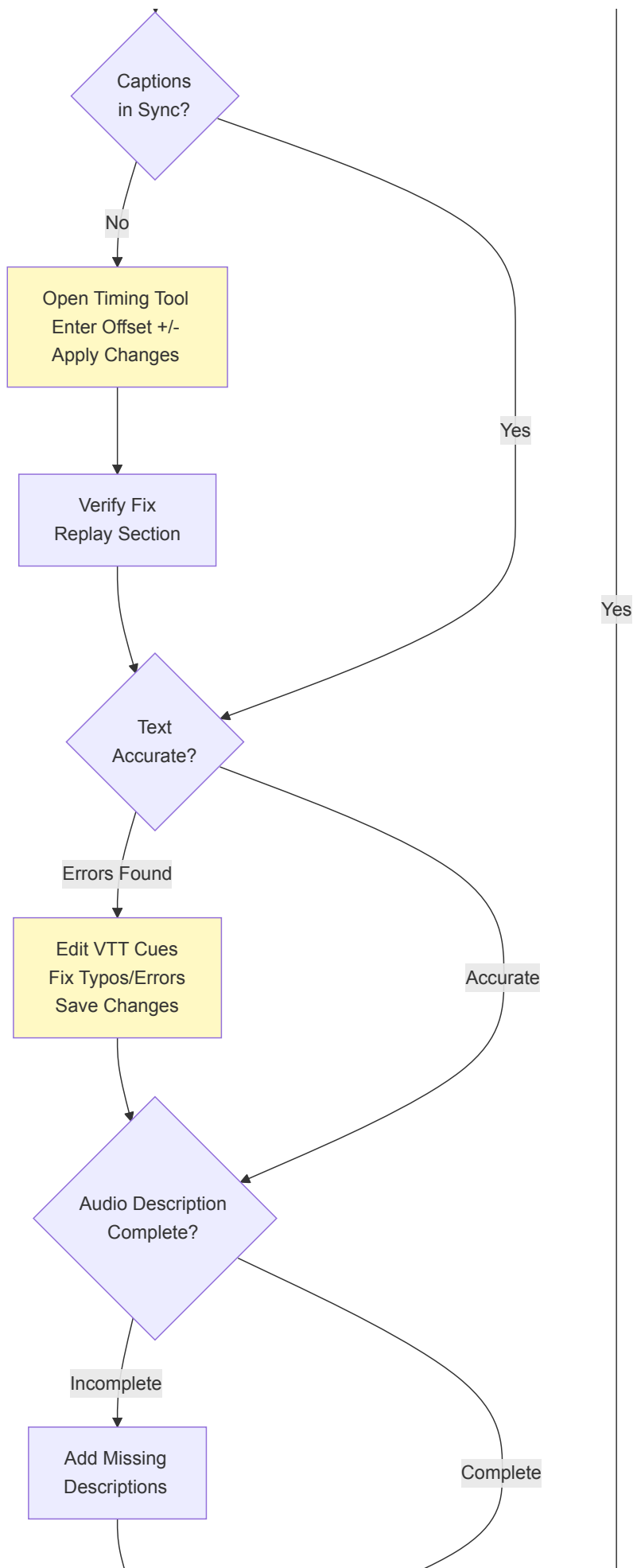
Key Touchpoints:

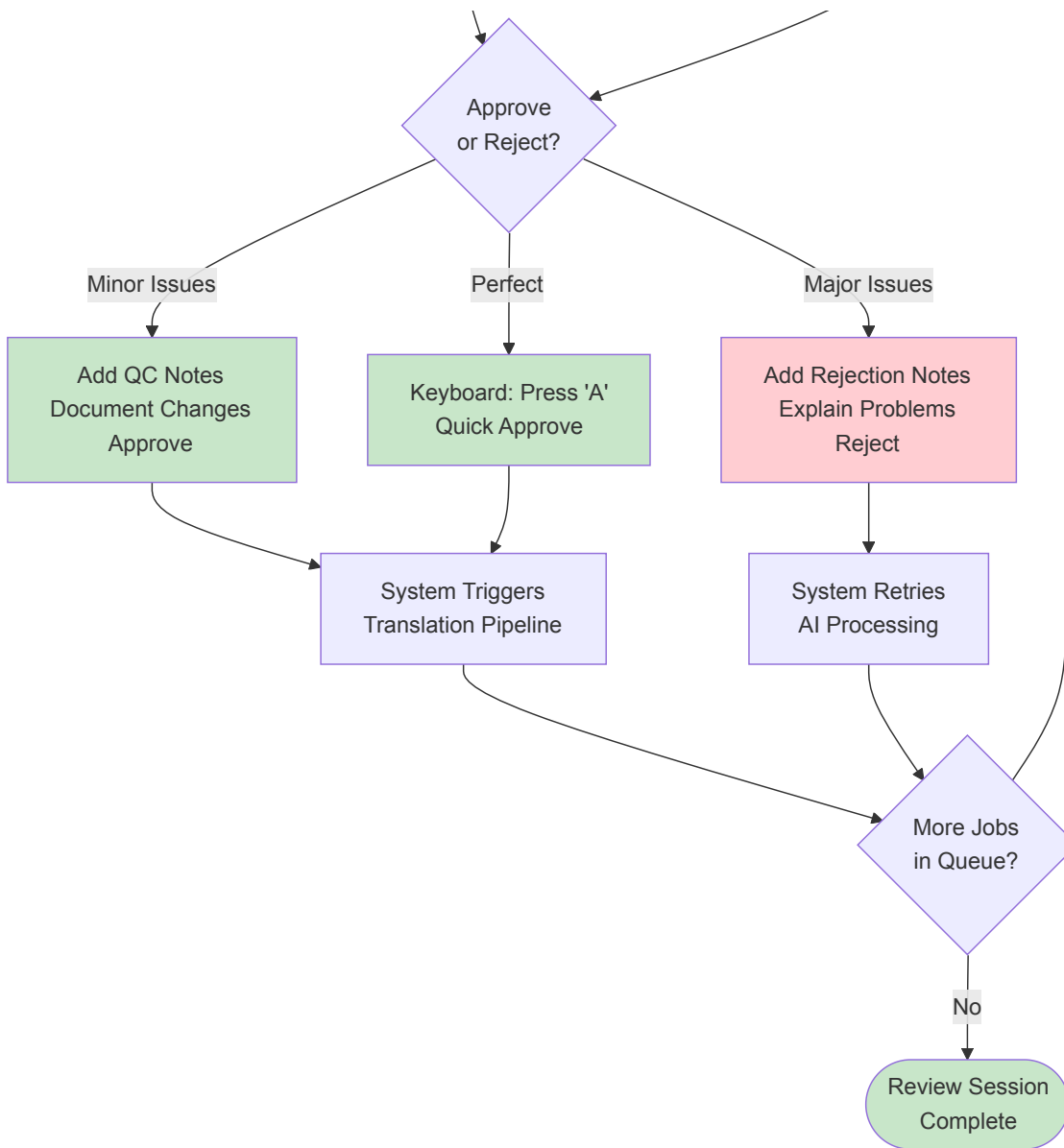
- **Real-time notifications:** 8-10 status updates via WebSocket
- **Human reviews:** 2 (English QC + Final Review)
- **Client interactions:** 2 (Upload + Download)
- **Fully automated:** AI processing, translation, TTS generation

13.2 Reviewer Journey: QC Review

Timeline: 10-20 minutes per job





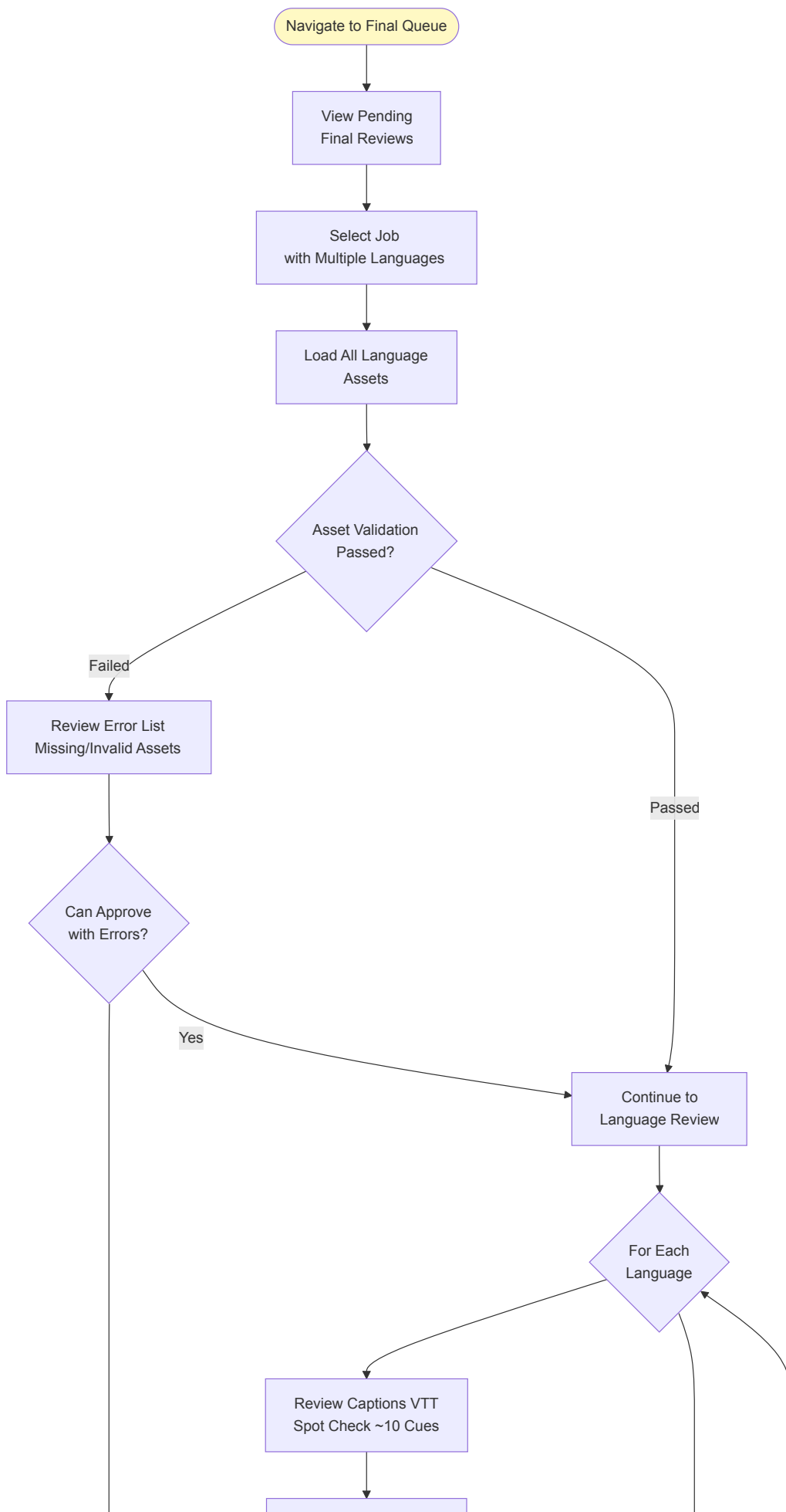


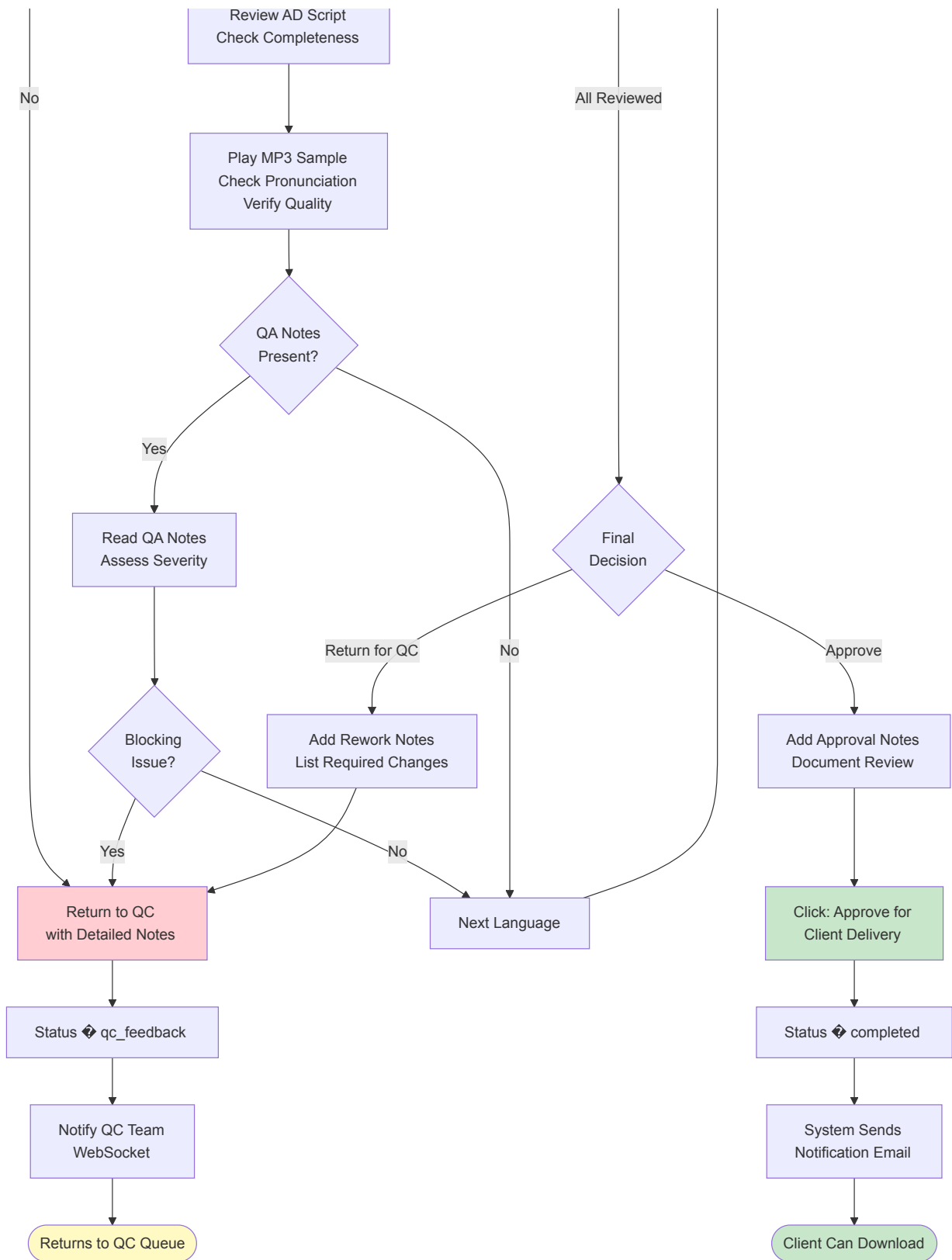
Reviewer Efficiency Features:

- **Keyboard Shortcuts:** A (approve), R (reject), Ctrl+S (save), 1/2/3 (view modes)
- **View Modes:** Side-by-side (default), Video-only, Editor-only
- **Quick Navigation:** Back to queue button, auto-redirect on approve
- **Batch Operations:** Bulk approve/reject from queue list
- **Validation Feedback:** Instant VTT error highlighting
- **Undo Support:** Browser native undo in text editors

13.3 Reviewer Journey: Final Review

Timeline: 15-25 minutes per job (depends on language count)





Final Review Checklist:

Per Language:

- Captions VTT file exists and is valid
- Audio Description VTT file exists and is valid
- MP3 file exists (if requested) and plays correctly
- Translation quality acceptable (spot check)

- TTS pronunciation natural and clear
- No QA notes or notes are acceptable
- VTT timing preserved from English version

Overall:

- All requested languages present
 - All requested output types delivered
 - AI confidence score acceptable (>70%)
 - No validation errors
 - Fits quality standards for client delivery
-

14. Key Differentiators & Edge Cases

14.1 Platform Differentiators

AI-First Approach

- Gemini 2.5 Pro provides state-of-the-art accuracy
- Multimodal understanding (visual + audio analysis)
- Self-healing JSON responses
- Confidence scoring guides human review

Human-in-the-Loop Quality

- AI generates draft, humans refine
- Two-stage review (English QC + Final multi-language)
- Professional VTT editing tools
- Catches AI hallucinations and errors

Real-Time Transparency

- WebSocket-powered live updates
- No page refresh needed to see progress
- Toast notifications at key milestones
- Persistent notification history

Timing Precision

- Millisecond-accurate VTT timing
- Bulk timing adjustment tools
- Preserved across translations
- TTS audio anchored to VTT timestamps

Scalable Architecture

- Celery workers scale horizontally
- Docker containers for easy deployment

- Cloud-native with GCS for unlimited storage
- Redis pub/sub for cross-process communication

14.2 Notable Edge Cases

AI Processing Edge Cases

Malformed JSON Responses

- **Issue:** Gemini occasionally returns truncated or invalid JSON
- **Solution:** 3-tier recovery:
 1. Automatic JSON fixes (trailing commas, missing braces)
 2. Re-prompt Gemini to fix its own output
 3. Create minimal fallback VTT with placeholder content
- **Result:** 99.9% success rate, no manual intervention needed

Missing Audio Description Field

- **Issue:** AI sometimes omits audio_description_vtt in response
- **Solution:** Self-healing creates basic VTT: "WEBVTT\n\n00:00:00.000 --> 00:00:05.000\nVideo content with visual elements."
- **Result:** Job continues processing, flagged for manual AD creation in QC

Low Confidence Scores (<70%)

- **Issue:** Complex audio, accents, technical jargon
- **Solution:** Validation blocks completion, requires thorough QC review
- **Result:** Human reviewer carefully verifies all content before approval

Translation Edge Cases

VTT Timing Preservation

- **Challenge:** Translated text may be longer/shorter than English
- **Solution:** System preserves exact English timing
- **Result:** Cues may appear crowded or sparse, but timing stays synchronized
- **Manual fix:** Reviewers can adjust timing in final review if needed

Unsupported Languages

- **Issue:** Google Translate supports 100+ languages, TTS supports fewer
- **Solution:** System generates VTT for all languages, MP3 only for TTS-supported
- **Result:** Clients get text captions even if audio unavailable

Transcreation Failures

- **Issue:** Gemini occasionally returns translations instead of transcreations
- **Solution:** Fallback to Google Translate, note in qa_notes
- **Result:** Job completes with standard translation, noted for potential manual review

TTS Edge Cases

Long Cues (>100 words)

- **Issue:** Some TTS services have character limits
- **Solution:** Split long cues into sub-segments, synthesize separately, stitch
- **Result:** Seamless audio, no apparent breaks

Timing Gaps

- **Challenge:** TTS audio may be shorter/longer than VTT cue duration
- **Solution:** Anchor to VTT start time, let audio run its natural length
- **Result:** Audio description may overlap with next dialogue (acceptable per WCAG)

Pronunciation Errors

- **Issue:** TTS mispronounces proper nouns, technical terms
- **Solution:** Logged in qa_notes for final reviewer attention
- **Result:** Flagged for potential manual re-recording if critical

System Edge Cases

Concurrent Job Processing

- **Scenario:** 10 clients upload videos simultaneously
- **Behavior:**
 - All jobs queued immediately
 - Celery worker processes 4 jobs concurrently (4-core worker)
 - Remaining jobs wait in Redis queue
 - FIFO processing order
 - No job starvation
- **Result:** Graceful degradation, longer queue times but no failures

Worker Crashes Mid-Task

- **Scenario:** Worker container dies during AI processing
- **Behavior:**
 - Celery task marked as FAILED
 - Job status remains in intermediate state (e.g., ai_processing)
 - No partial results saved
- **Solution:** Admin reprocess-job endpoint resets job to created
- **Result:** Job can be retried, no data corruption

GCS Upload Failures

- **Scenario:** Network interruption during VTT upload
- **Behavior:**
 - Upload raises exception
 - Celery task fails and retries (default 3 retries)
 - Exponential backoff between retries

- **Result:** Transient failures self-heal, permanent failures flagged in error field

MongoDB Connection Loss

- **Scenario:** MongoDB container restarts during processing
- **Behavior:**
 - Motor driver detects connection loss
 - Automatic reconnection attempts
 - Tasks wait for reconnection (timeout: 30s)
 - Connection pool recovers
- **Result:** Brief pause, then processing continues

WebSocket Disconnections

- **Scenario:** Client's internet drops briefly
- **Behavior:**
 - Frontend detects connection close
 - Auto-reconnect with exponential backoff
 - Subscribes to channels again
 - Receives catch-up messages
- **Result:** No status updates lost, seamless recovery

Browser Refresh During Upload

- **Scenario:** User refreshes page mid-upload
- **Behavior:**
 - Upload aborts (browser native behavior)
 - No job created (creation happens after upload)
 - User returns to upload page
- **Result:** Clean state, user can retry upload

15. Performance Characteristics

15.1 Processing Times

Typical Job Timeline (10-minute video)

Stage	Duration	Bottleneck
Upload	30-120s	Internet bandwidth
Ingestion	10-30s	GCS download + ffmpeg probe
AI Processing	30-90s	Gemini API latency
QC Review	10-20 min	Human reviewer
Translation (per lang)	10-30s	Google Translate API
TTS Generation (per lang)	30-120s	TTS synthesis speed
Final Review	5-15 min	Human reviewer
Notification	1-5s	Email delivery

Total: ~15-30 minutes end-to-end (mostly human review time)

Automated Processing Only: ~2-5 minutes (no human reviews)

15.2 Scalability Metrics

Current Capacity (single 8-core, 32GB server)

- **Concurrent Uploads:** 10+ (limited by upload bandwidth)
- **Concurrent Processing:** 4 jobs (Celery worker concurrency)
- **Queue Depth:** Unlimited (Redis queue)
- **Active Users:** 50+ simultaneous (WebSocket connections)
- **API Requests:** 1000+ req/min (Gunicorn 9 workers)

Bottlenecks & Scaling Paths

1. **Celery Worker:** Most CPU/memory intensive
 - Scale: Add more worker containers
 - Each worker: 4 concurrent tasks
 - Horizontal scaling: Add worker nodes
2. **MongoDB:** Database queries
 - Current: Indexes optimize most queries
 - Scale: MongoDB Atlas with replica sets
3. **Redis:** Queue + WebSocket pub/sub
 - Current: Single instance handles load
 - Scale: Redis Cluster or Redis Cloud
4. **GCS:** File storage
 - Current: Virtually unlimited
 - No scaling needed

15.3 Resource Consumption

Per-Job Resource Usage

Storage:

- Source video: 50MB - 2GB (typical: 200MB for 10min video)
- English VTT (2 files): 50KB - 500KB (typical: 100KB)
- Translated VTT (2 files + N langs): 50KB - 500KB each
- MP3 audio (N langs): 1MB - 50MB per language (typical: 10MB for 10min)
- **Total per job:** 200MB - 5GB (typical: 500MB with 3 languages)

Compute:

- Ingestion + AI: ~30s CPU time (download + ffmpeg)
- Translation: ~10s per language (API calls)
- TTS: ~60s per language (audio synthesis)
- **Total compute:** ~2-5 minutes per job

Memory:

- Worker peak: 2-4GB during video processing
- API steady: 500MB - 1GB
- MongoDB: 1-2GB (with indexes)
- Redis: 100-500MB

Network:

- Upload: Video size (client ↔ GCS)
- Gemini upload: Video size (GCS ↔ Gemini)
- Downloads: Sum of all assets (GCS ↔ client)
- **Bandwidth:** ~3x video size total per job

16. Compliance & Standards

16.1 Accessibility Standards

WCAG 2.1 Compliance

- **Level AA:** Captions for all audio content
- **Level AAA:** Audio descriptions for visual content
- **Timing:** Synchronized within 500ms tolerance
- **Format:** WebVTT (W3C standard)

Closed Caption Standards

- Speaker identification (when multiple speakers)
- Sound effects notation [MUSIC], [APPLAUSE]
- Proper punctuation and capitalization
- 32 characters per line maximum (recommended)
- Reading speed: 160-180 words per minute

Audio Description Standards

- Describes visual information not in dialogue
- Fits between dialogue/narration gaps
- Objective, non-interpretive language
- Key visual elements: setting, characters, actions, on-screen text
- Does not overlap essential audio

16.2 File Format Standards

WebVTT (Web Video Text Tracks)

WEBVTT

00:00:00.000 --> 00:00:04.500

Welcome to our Q4 training session.

00:00:05.000 --> 00:00:08.200

Today we'll cover new product features.

Features Used:

- Cue timings (mandatory)
- Cue text (mandatory)
- Cue identifiers (optional, not used)
- Styling/positioning (not used, client responsibility)

MP3 Audio Format

- Container: MP3
- Codec: MPEG-1 Audio Layer 3
- Bitrate: 128 kbps (constant)
- Sample rate: 24 kHz
- Channels: Mono
- Encoding: High quality preset

16.3 Data Privacy & Retention

Video Content

- Uploaded videos stored in client's GCS bucket
- Access restricted to authorized users only
- Signed URLs expire after 24 hours
- No third-party access (Gemini processes but doesn't store)

User Data

- Personal information (email, name) encrypted in transit
- Passwords hashed with bcrypt (never stored plain-text)
- No sharing with third parties
- Audit logs track all access

Retention Policy

- Jobs: Retained indefinitely (client can delete)
- Audit logs: 365 days default (admin configurable)
- User accounts: Active until deactivated
- GCS files: Lifecycle management (client configurable)

17. Future Enhancements (Roadmap)

While the current system is fully functional, the following enhancements are planned:

Enhanced AI Capabilities

- Fine-tuned models for industry-specific terminology

- Custom glossary support for proper nouns
- Multi-speaker voice synthesis (different voices for different speakers)

Workflow Optimizations

- Batch upload (multiple videos at once)
- Job templates (save common configurations)
- Scheduled processing (off-peak queue management)

Advanced QC Tools

- AI-suggested edits with confidence scores
- Waveform visualization for audio sync
- Side-by-side comparison (original vs translated)
- Collaborative review (multiple reviewers per job)

Integration Options

- REST API for programmatic job creation
- Webhooks for status notifications
- Direct DAM (Digital Asset Management) integration
- Content management system plugins

Analytics & Reporting

- Per-client usage dashboards
- Cost tracking and billing reports
- Quality metrics over time
- SLA monitoring and alerts

18. Technical Support & Resources

18.1 System Monitoring

Health Endpoints

- `/health` - Basic health check (public)
- `/admin/health/detailed` - Comprehensive health (admin only)
 - MongoDB connection status
 - Redis connection status
 - GCS bucket access
 - Celery worker count and active tasks

Metrics Endpoints

- `/metrics` - Prometheus metrics export (disabled in current deployment)
 - Future: Grafana dashboards
 - Future: Alerting on anomalies

Log Access

- Docker container logs via `docker compose logs`
- Structured JSON logging for parsing
- Log levels: INFO (default), DEBUG (verbose), ERROR
- Log rotation: 10MB max size, 3 files

18.2 Operational Procedures

Deployment

- See [DEPLOYMENT.md](#) for complete procedures
- Full deployment: ~10-15 minutes
- Frontend-only: ~2-3 minutes
- Zero-downtime: Not required (acceptable brief interruption)

Backup & Recovery

- MongoDB: VM-level backups (daily)
- GCS: Object versioning enabled
- No separate backup strategy needed

Monitoring Recommendations

- Monitor Celery queue depth (Redis)
- Track job processing times (detect slowdowns)
- Watch error rates in audit logs
- Alert on worker crashes or stuck jobs

18.3 Common Operational Tasks

Add New User

```
# Via API (admin authenticated)
curl -X POST https://ai-sandbox.oliver.solutions/video-accessibility-
back/api/v1/admin/users \
  -H "Authorization: Bearer {token}" \
  -H "Content-Type: application/json" \
  -d '{
    "email": "newuser@example.com",
    "full_name": "New User",
    "password": "temporary123",
    "role": "client"
  }'
```

Reprocess Stuck Job

```
# Via API (admin authenticated)
curl -X POST https://ai-sandbox.oliver.solutions/video-accessibility-
```

```
back/api/v1/admin/maintenance/reprocess-job/{job_id} \
-H "Authorization: Bearer {token}"
```

Check System Health

```
curl https://ai-sandbox.oliver.solutions/video-accessibility-back/health
# Expected: {"status":"healthy","version":"1.0.0"}
```

View Container Logs

```
# On server
cd /opt/video-accessibility
sudo docker compose logs -f api worker
```

19. Conclusion

The Accessible Video Processing Platform represents a production-ready, enterprise-grade solution for automated video accessibility compliance. By combining cutting-edge AI technology with human quality control workflows, the platform delivers WCAG-compliant accessibility content with professional quality and efficiency.

System Maturity

- Production-deployed on GCP infrastructure
- Multi-user, multi-role architecture
- Real-time status updates via WebSockets
- Comprehensive audit logging
- Docker-based deployment for portability
- Automated testing and deployment scripts
- Security best practices implemented

Business Value

- **Time Savings:** 90% reduction vs manual captioning (15 min vs 2+ hours)
- **Cost Efficiency:** AI-first approach reduces human labor costs
- **Quality Assurance:** Two-stage human review ensures accuracy
- **Scalability:** Cloud-native architecture grows with demand
- **Compliance:** Meets WCAG 2.1 AA/AAA requirements

Technical Excellence

- Modern tech stack (React 19, FastAPI, Python 3.11)
 - Microservices architecture with clear separation of concerns
 - Asynchronous processing for non-blocking operations
 - Comprehensive error handling and retry mechanisms
 - Real-time user experience with WebSocket integration
 - Security-first design (JWT, RBAC, audit logs)
-

Document End

For deployment procedures, see [DEPLOYMENT.md](#)

For original system specification, see [video_accessibility_spec.md](#)

Version History:

- v2.0 (2025-01-09) - Complete technical documentation post-deployment
- v1.0 (2025-08-17) - Initial specification document