Multiview CTC Grader — Complete System Documentation

Version 2.0 • Framework: v1.5 (Strict++) • Multiview Technology

Multiview CTC Grader - Complete System Documentation

Version: 2.0
Last Updated: October 6, 2025
Framework: Multiview Grading Standards v1.5 (Strict++)

Table of Contents

System Overview

The Multiview CTC Grader is a web-based application that provides forensic-level grading analysis for Cinnamon Toast Crunch cereal specimens. It combines computer vision AI (OpenAI GPT-4o Vision), image processing, PDF report generation, and a searchable archive system.

Core Features

- **Guided Photo Capture**: Step-by-step interface for capturing front and side photos
- **HEIC Support**: Automatic conversion of iPhone HEIC images to JPEG
- **AI Grading**: GPT-4o Vision analyzes specimens using Multiview Grading Standards
- **PDF Reports**: Auto-generated 4-page professional grading reports
- **Database Archive**: SQLite database with full-text search
- **Browse Interface**: Web-based gallery with filtering and search
- **Document Library**: Access to grading standards, error reports, and historical documents

User Flow

1. User opens `/` (submission.html)
2. Clicks "Start" → Creates new specimen ID (e.g., A-30)
3. Takes FRONT photo → Uploads and converts HEIC if needed
4. Takes SIDE photo → Uploads and converts HEIC if needed
5. Clicks "Submit for Grading" → Triggers AI analysis
6. System validates images (non-screenshot, actual CTC specimen)
7. GPT-4o Vision analyzes both photos
8. System generates 4-page PDF report with Multiview formatting
9. Saves to database with full metadata
10. User can view results, download PDF, or browse archive

Architecture

High-Level Architecture

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ CLIENT (Browser/Phone) ■
■ - submission.html (Guided Capture) ■
■ - browse.html (Archive Gallery) ■
■ - about.html (Documentation) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ HTTP/HTTPS
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ EXPRESS.JS SERVER ■
■ - Static file serving ■
■ - API routes (/api/*) ■
■ - File upload handling (express-fileupload) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
▼ ▼ ▼ ▼
■■■■■■■■■■■■■■■■■ ■■■■■■■■■■■■■■■ ■■■■■■■■■■■■ ■■■■■■■■■■■■
■ Image ■ ■ AI Grading ■ ■ Report ■ ■ Database ■
■ Processor ■ ■ Service ■ ■ Generator■ ■ (SQLite) ■
■ ■ ■ ■ ■ ■ ■ ■
■ - HEIC→JPEG ■ ■ - GPT-4o ■ ■ - PDF ■ ■ - Search ■
■ - Validation ■ ■ - CTC Check ■ ■ - MD ■ ■ - CRUD ■
■ - Storage ■ ■ - Subgrades ■ ■ - JSON ■ ■ ■
■■■■■■■■■■■■■■■■■ ■■■■■■■■■■■■■■■ ■■■■■■■■■■■■ ■■■■■■■■■■■■
```

Request Flow Example: Photo Upload

```
1. User selects photo from phone camera
↓
2. Browser sends multipart/form-data POST to /api/savePhoto
↓
3. express-fileupload saves to temp directory
↓
4. validateImageFile() checks format and existence
↓
5. isHEICFormat() detects if conversion needed
↓
6. processUploadedImage() reads from tempFilePath
↓
7. convert() from heic-convert library converts to JPEG
↓
8. Saves to /Specimens/A-##/A-##_front.jpg
↓
9. Returns { ok: true, url: "/specimens/A-##/A-##_front.jpg" }
↓
10. Frontend displays converted image
```

Request Flow Example: Grading

```
1. User clicks "Submit for Grading"
↓
2. POST /api/grade with { specimenId: "A-30" }
↓
3. validateSavedImage() checks both photos exist
↓
4. validateImage() checks images aren't screenshots/wrong content
↓
5. classifyCTC() confirms images are CTC specimens
↓
6. gradeSpecimen() calls OpenAI GPT-4o Vision API
↓
7. AI returns JSON with grade, subgrades, notes
↓
8. generateReports() creates Markdown and PDF
↓
9. saveSpecimenRecord() inserts into SQLite database
↓
10. Returns full grading results to frontend
↓
11. Frontend displays grade, subgrades, download link
```

Directory Structure

```
D:\Projects\CTC_Grading\
■
■■■ Documents\ # Persistent storage
```

```
■ ■■■ ctc_grades.db # SQLite database
■ ■■■ Grading Standards\ # Framework PDFs
■ ■ ■■■ Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf
■ ■■■ Reports\ # Generated PDF reports
■ ■ ■■■ A-16_CTC_Grading_Report.pdf
■ ■ ■■■ A-18_CTC_Grading_Report.pdf
■ ■ ■■■ ...
■ ■■■ Errors\ # Error reports for invalid submissions
■ ■■■ Misc\ # Miscellaneous documents
■ ■■■ Process\ # Process documentation
■ ■■■ Tests\ # Test reports
■ ■■■ Web App History\ # Development history
■
■■■ Specimens\ # Specimen storage
■ ■■■ A-01\
■ ■ ■■■ A-01_front.jpg
■ ■ ■■■ A-01_side.jpg
■ ■ ■■■ A-01_CTC_Grading_Report.pdf
■ ■ ■■■ A-01_CTC_Grading_Report.md
■ ■ ■■■ A-01_CTC_Grading_Report.json
■ ■■■ A-02\
■ ■■■ ...
■
■■■ web\ # Web application
■■■ server.js # Express.js server (main entry point)
■■■ package.json # Node.js dependencies
■■■ package-lock.json
■
■■■ lib\ # Backend libraries
■ ■■■ multiview-config.js # Path configuration
■ ■■■ database.js # SQLite operations
■ ■■■ ai-grader.js # OpenAI integration
■ ■■■ image-processor.js # HEIC conversion & validation
■ ■■■ image-validator.js # Content validation (anti-screenshot)
■ ■■■ report-generator.js # PDF/Markdown generation
■
■■■ temp\ # Temporary upload storage
■ ■■■ tmp-* # Temp files (auto-cleaned)
■
■■■ node_modules\ # NPM dependencies
■
■■■ submission.html # Main guided capture interface
■■■ browse.html # Archive gallery
■■■ about.html # Documentation page
■
■■■ import-specimens.js # Utility: Import existing specimens to DB
```

Technology Stack

Backend

Technology	Version	Purpose
Node.js	v22.20.0	JavaScript runtime
Express.js	^4.19.2	Web server framework
SQLite3	^5.1.7	Database
express-fileupload	^1.4.2	Multipart file upload handling
heic-convert	^1.2.4	HEIC to JPEG conversion
sharp	^0.34.4	Image analysis and validation
pdfkit	^0.14.0	PDF generation
openai	^4.104.0	GPT-4o Vision API client
md-to-pdf	^5.2.0	Markdown to PDF conversion (backup)

Frontend

Technology	Purpose
Vanilla JavaScript	Client-side logic
Fetch API	HTTP requests
HTML5	Markup
CSS3	Styling
File API	Camera/file input handling

Development Tools

- **PowerShell** - Batch scripts for Windows
- **Git** - Version control
- **Cursor/VS Code** - IDE

Database Schema

SQLite Database: `ctc_grades.db`

#### Table: `specimens`

Column	Type	Nullable	Description
`id`	INTEGER	No	Primary key (auto-increment)
`specimenId`	TEXT	No	Unique specimen ID (e.g., "A-30")
`frameworkVersion`	TEXT	Yes	Grading framework version (e.g., "v1.5 Strict++")
`frontPath`	TEXT	Yes	Path to front image
`sidePath`	TEXT	Yes	Path to side image
`grade`	TEXT	Yes	Overall grade (e.g., "PSA 8.0 (NM)")
`subgrades`	TEXT	Yes	JSON string of subgrade scores
`notes`	TEXT	Yes	AI analysis notes
`pdfPath`	TEXT	Yes	Path to PDF report

`dateGraded`	TEXT	Yes	ISO 8601 timestamp
`systemHash`	TEXT	Yes	SHA-256 hash for provenance
`urlFront`	TEXT	Yes	Public URL for front image
`urlSide`	TEXT	Yes	Public URL for side image

#### Example Record

```
{
"id": 1,
"specimenId": "A-16",
"frameworkVersion": "v1.5 Strict++",
"frontPath": "D:\\Projects\\CTC_Grading\\Specimens\\A-16\\A-16_front.jpg",
"sidePath": "D:\\Projects\\CTC_Grading\\Specimens\\A-16\\A-16_side.jpg",
"grade": "PSA 8.0 (NM)",
"subgrades": "{\"geometry\":8.0,\"corners\":8.0,\"coating\":8.0,\"surface\":
8.0,\"alignment\":8.0}",
"notes": "Specimen shows balanced morphology with minimal curvature...",
"pdfPath": "D:\\Projects\\CTC_Grading\\Documents\\Reports\\A-16_CTC_Grading_
Report.pdf",
"dateGraded": "2025-10-05T20:19:28.000Z",
"systemHash": "a1b2c3d4...",
"urlFront": "/specimens/A-16/A-16_front.jpg",
"urlSide": "/specimens/A-16/A-16_side.jpg"
}
```

#### Database Operations

Create Table:

```
CREATE TABLE IF NOT EXISTS specimens (
id INTEGER PRIMARY KEY AUTOINCREMENT,
specimenId TEXT UNIQUE,
frameworkVersion TEXT,
frontPath TEXT,
sidePath TEXT,
grade TEXT,
subgrades TEXT,
notes TEXT,
pdfPath TEXT,
dateGraded TEXT,
systemHash TEXT,
urlFront TEXT,
urlSide TEXT
)
```

Insert Record:

```
db.run(`
INSERT INTO specimens (
specimenId, frameworkVersion, frontPath, sidePath,
grade, subgrades, notes, pdfPath, dateGraded, systemHash,
urlFront, urlSide
) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
`, [values...])
```

Query All:

```
db.all("SELECT * FROM specimens ORDER BY id DESC", [])
```

Query by ID:

```
db.get("SELECT * FROM specimens WHERE specimenId = ?", [specimenId])
```

API Endpoints

Base URL

`http://localhost:3000`

`GET /`

Description: Serves the main guided capture interface
Response: HTML (submission.html)

`GET /browse`

Description: Serves the archive gallery page
Response: HTML (browse.html)

`GET /about`

Description: Serves the about/documentation page
Response: HTML (about.html)

`GET /api/nextSpecimen`

Description: Generates next sequential specimen ID and creates folder
Response:

```
{
"specimenId": "A-30"
}
```

`POST /api/savePhoto`

Description: Uploads and processes a photo (HEIC conversion if needed)

Request Body (multipart/form-data):
- `photo`: File (image)
- `specimenId`: String (e.g., "A-30")
- `label`: String ("front" or "side")

Response (Success):

```
{
"ok": true,
"path": "D:\\Projects\\CTC_Grading\\Specimens\\A-30\\A-30_front.jpg",
"url": "/specimens/A-30/A-30_front.jpg",
"originalFormat": "IMG_0227.HEIC",
"converted": true
}
```

Response (Error):

```
{
"error": "Unsupported image format. Please upload a JPG, PNG, GIF, WEBP, or HEIC image.",
"fullError": { /* detailed error object */ },
"timestamp": "2025-10-06T02:50:33.089Z"
}
```

`POST /api/grade`

Description: Grades a specimen using AI and generates reports

Request Body (JSON):

```
{
"specimenId": "A-30"
}
```

Response (Success):

```
{
"success": true,
"specimenId": "A-30",
"frameworkVersion": "v1.5 Strict++",
"grade": "PSA 8.0 (NM)",
"subgrades": {
"geometry": 8.0,
"corners": 8.0,
"coating": 8.0,
"surface": 8.0,
"alignment": 8.0
},
"notes": "Specimen presents balanced morphology...",
"reportPath": "D:\\Projects\\CTC_Grading\\Documents\\Reports\\A-30_CTC_Grading_Report.pdf",
"markdownPath": "D:\\Projects\\CTC_Grading\\Documents\\Reports\\A-30_CTC_Grading_Report.md",
"specimenReportPath": "D:\\Projects\\CTC_Grading\\Specimens\\A-30\\A-30_CTC_Grading_Report.pdf",
"specimenMarkdownPath": "D:\\Projects\\CTC_Grading\\Specimens\\A-30\\A-30_CTC_Grading_Report.md",
"reportJsonPath": "D:\\Projects\\CTC_Grading\\Specimens\\A-30\\A-30_CTC_Grading_Report.json",
"activeFramework": "Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf",
"errorNotePath": null,
"urlFront": "/specimens/A-30/A-30_front.jpg",
"urlSide": "/specimens/A-30/A-30_side.jpg"
}
```

Response (Error - Invalid Image):

```
{
"status": "error",
"message": "Image does not appear to be a Cinnamon Toast Crunch piece.",
"recommendation": "Please ensure both photos clearly show a single Cinnamon Toast Crunch specimen.",
"errorReportPath": "D:\\Projects\\CTC_Grading\\Documents\\Errors\\A-30_CTC_Error_Report\\A-30_CTC_Error_Report.pdf"
}
```

`GET /api/specimens`

Description: Lists all graded specimens with optional search

Query Parameters:
- `q` (optional): Search query (searches specimenId, grade, framework, notes)

Response:

```
{
"success": true,
"count": 15,
"specimens": [
{
"id": 1,
"specimenId": "A-29",
"frameworkVersion": "v1.5 Strict++",
"frontPath": "...",
"sidePath": "...",
"grade": "PSA 8.0 (NM)",
"subgrades": { /* parsed object */ },
"notes": "...",
"pdfPath": "/reports/A-29_CTC_Grading_Report.pdf",
"dateGraded": "2025-10-05T21:54:25.000Z",
"systemHash": "...",
"urlFront": "/specimens/A-29/A-29_front.jpg",
"urlSide": "/specimens/A-29/A-29_side.jpg"
},
// ... more specimens
]
}
```

`GET /api/specimens/:id`

Description: Gets a single specimen by ID

Response:

```
{
"success": true,
"specimen": {
"id": 1,
"specimenId": "A-29",
// ... full specimen data
}
}
```

`GET /api/documents`

Description: Lists all documents from Documents folder

Response:

```
{
"success": true,
"documents": {
"Grading Standards": [
{
"name": "Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf",
"url": "/documents/Grading%20Standards/Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf",
"type": "PDF"
}
],
"Errors": [ /* error reports */ ],
"Misc": [ /* misc docs */ ],
// ... other categories
}
}
```

`GET /api/debug`

Description: System health check and diagnostics

Response:

```
{
"status": "Server is running",
"timestamp": "2025-10-06T02:55:00.000Z",
"version": "2.0",
"features": {
"heicSupport": true,
"imageValidation": true,
"aiGrading": true,
"pdfReports": true,
"database": true
},
"system": {
```

```
"nodeVersion": "v22.20.0",
"platform": "win32",
"arch": "x64",
"memory": { /* memory usage */ },
"uptime": 123.456
},
"paths": {
"specimensDir": "D:\\Projects\\CTC_Grading\\Specimens",
"reportsDir": "D:\\Projects\\CTC_Grading\\Documents\\Reports",
"standardsDir": "D:\\Projects\\CTC_Grading\\Documents\\Grading Standards"
},
"dependencies": {
"express": "installed",
"heicConvert": "installed",
"sqlite3": "installed",
"pdfkit": "installed",
"openai": "installed"
}
}
```

Frontend Pages

submission.html (Main Interface)

Purpose: Guided photo capture and grading submission

Features:
- Sequential workflow (Start → Front Photo → Side Photo → Submit)
- Real-time photo preview
- HEIC conversion feedback
- Grading results display
- Error handling with full details
- Links to browse archive and about page

UI Sections:
1. **Header**: Title + current framework display
2. **Guided Capture Card**: Step-by-step buttons and instructions
3. **Archive Access Card**: Search input + browse link
4. **About Multiview Card**: Description + link
5. **Footer**: Copyright notice

JavaScript Functions:
- `api(path, opts)`: Fetch wrapper with error handling
- `openCameraAndUpload(label)`: Opens camera/file picker and uploads
- `startNewSpecimen()`: Resets UI for new specimen
- Event handlers for Start, Take FRONT, Take SIDE, Submit buttons

---

browse.html (Archive Gallery)

Purpose: Browse and search all graded specimens

Features:
- Tabbed interface (Specimens / Documents)
- Search bar with real-time filtering
- Specimen cards with:
- Specimen ID
- Grade badge
- Framework version
- Front/side image thumbnails
- PDF download link
- Document library with categorized files
- Responsive grid layout

JavaScript Functions:
- `loadSpecimens()`: Fetches and displays all specimens
- `loadDocuments()`: Fetches and displays document library
- `searchSpecimens()`: Filters specimens by query
- `renderSpecimenCard(specimen)`: Creates HTML for specimen card

---

about.html (Documentation)

Purpose: Explains Multiview Technology and grading system

Content:
- What is Multiview Technology
- How the grading works
- Framework explanation
- Subgrade categories
- FAQ section
- Links back to main interface

Backend Services

lib/multiview-config.js

Purpose: Centralized path configuration

Exports:
```
export const MULTIVIEW_CONFIG = {
root: "D:\\Projects\\CTC_Grading",
standardsDir: "D:\\Projects\\CTC_Grading\\Documents\\Grading Standards",
reportsDir: "D:\\Projects\\CTC_Grading\\Documents\\Reports",
errorsDir: "D:\\Projects\\CTC_Grading\\Documents\\Errors",
specimensDir: "D:\\Projects\\CTC_Grading\\Specimens",
activeFrameworkName: "Multiview Grading Standards v1.5 (Strict++)"
};
```

Usage:

```
import { MULTIVIEW_CONFIG as C } from './lib/multiview-config.js';
const reportPath = path.join(C.reportsDir, 'report.pdf');
```

---

lib/database.js

Purpose: SQLite database operations

Functions:

#### `saveSpecimenRecord(data)`
Inserts a new specimen record into the database.

Parameters:

```
{
specimenId: "A-30",
frameworkVersion: "v1.5 Strict++",
frontPath: "...",
sidePath: "...",
grade: "PSA 8.0 (NM)",
subgrades: { geometry: 8.0, ... },
notes: "...",
pdfPath: "...",
dateGraded: "2025-10-06T...",
systemHash: "...",
urlFront: "/specimens/A-30/A-30_front.jpg",
urlSide: "/specimens/A-30/A-30_side.jpg"
}
```

Returns: `Promise` with inserted record

#### `getAllSpecimens()`
Retrieves all specimens from database, ordered by ID descending.

Returns: `Promise` of specimen objects

#### `getSpecimenById(specimenId)`
Retrieves a single specimen by its ID.

Parameters: `specimenId` (String)

Returns: `Promise` or `null` if not found

---

lib/ai-grader.js

Purpose: OpenAI GPT-4o Vision integration

Functions:

#### `gradeSpecimen(specimenId, frontPath, sidePath)`
Sends images to GPT-4o Vision for analysis.

Process:
1. Reads front and side images as base64
2. Loads grading standards PDF (if available)
3. Constructs system prompt with Multiview framework rules
4. Sends images + prompt to GPT-4o Vision
5. Parses JSON response
6. Returns structured grading data

Prompt Structure:

```
You are an expert grader for the Multiview CTC Grading System.

Framework: Multiview Grading Standards v1.5 (Strict++)

Analyze these two photos of a Cinnamon Toast Crunch specimen:
- FRONT view (straight-on)
- SIDE view (showing curvature)

Grade the specimen on these subgrade categories (0-10 scale):
1. Geometry (30% weight): Shape, flatness, aspect ratio
2. Corners (20% weight): Sharpness, rounding, chips
3. Coating (12% weight): Cinnamon/sugar distribution
4. Surface (20% weight): Ridge clarity, pocking
5. Alignment (18% weight): Edge uniformity, compression

Rules:
- Any subgrade < 8.0 caps overall grade at ≤ 8.0
- Curvature > 7.5% caps grade at ≤ 8.0
- Round DOWN, never up
- Be strict and conservative

Return JSON:
{
"frameworkVersion": "v1.5 Strict++",
"grade": "PSA 8.0 (NM)",
"subgrades": {
"geometry": 8.0,
"corners": 8.0,
"coating": 8.0,
"surface": 8.0,
"alignment": 8.0
},
"notes": "Detailed analysis..."
}
```

Response Example:

```
{
frameworkVersion: "v1.5 Strict++",
grade: "PSA 8.0 (NM)",
subgrades: {
geometry: 8.0,
corners: 8.0,
coating: 8.0,
surface: 8.0,
alignment: 8.0
},
notes: "The specimen presents balanced morphology with 3.2% curvature..."
}
```

#### `classifyCTC(frontPath, sidePath)`
Validates that images are actually CTC specimens (not screenshots, other food, etc.)

Process:
1. Sends images to GPT-4o Vision
2. Asks: "Is this a Cinnamon Toast Crunch cereal piece?"
3. Returns confidence score

Returns:

```
{
isCTC: true,
confidence: 0.95,
reason: "Clear CTC specimen with characteristic square shape and cinnamon
coating"
}
```

---

lib/image-processor.js

Purpose: Image upload handling and HEIC conversion

Functions:

#### `isHEICFormat(file)`
Checks if uploaded file is HEIC/HEIF format.

Parameters: `file` (express-fileupload file object)

Returns: `Boolean`

Logic:

```
const ext = path.extname(file.name).toLowerCase();
const mimeType = file.mimetype?.toLowerCase();
return ['.heic', '.heif'].includes(ext) ||
['image/heic', 'image/heif'].includes(mimeType);
```

#### `processUploadedImage(file, destPath)`
Processes uploaded image (converts HEIC if needed, otherwise moves file).

Parameters:

- `file`: express-fileupload file object
- `destPath`: Destination path (e.g., `D:\...\A-30_front.jpg`)

Returns: `Promise` - Final file path

Process for HEIC:
1. Check if `file.tempFilePath` exists (primary path with `useTempFiles: true`)
2. Read file from disk: `fs.readFileSync(file.tempFilePath)`
3. Convert using `heic-convert`:

```
const outputBuffer = await convert({
buffer: inputBuffer,
format: 'JPEG',
quality: 1
});
```

4. Write JPEG: `fs.writeFileSync(jpegPath, outputBuffer)`
5. Delete temp file
6. Return JPEG path

Process for non-HEIC:
1. Use `file.mv(destPath)` to move file
2. Return destination path

#### `validateImageFile(file)`
Validates uploaded file before processing.

Checks:
- File exists and has `name` property
- Format is supported (JPG, PNG, GIF, WEBP, HEIC)
- For HEIC: Either `file.data` has content OR `file.tempFilePath` exists
- For non-HEIC: `file.tempFilePath` exists

Returns:

```
{
valid: true,
error: null
}
// OR
{
valid: false,
error: "Unsupported image format. Please upload..."
}
```

#### `validateSavedImage(imagePath)`
Validates a saved image file exists and is in supported format.

Parameters: `imagePath` (String)

Returns: `{ valid: Boolean, error: String }`

---

lib/image-validator.js

Purpose: Content validation (anti-screenshot, non-cereal detection)

Functions:

#### `validateImage(imagePath)`
Uses `sharp` library to analyze image content and detect if it's a valid photo.

Checks:
- File format (rejects non-standard formats)
- Color variance (screenshots have low variance)
- Aspect ratio (rejects ultra-wide or ultra-tall images)
- File size (rejects suspiciously small files)

Returns:

```
{
valid: true,
reason: null
}
// OR
{
valid: false,
reason: "Image appears to be a screenshot or has insufficient color
variance"
}
```

Implementation:

```
const metadata = await sharp(imagePath).metadata();
const stats = await sharp(imagePath).stats();

// Check color variance
const variance = stats.channels.reduce((sum, ch) => sum + ch.stdev, 0) /
stats.channels.length;
if (variance < 10) {
return { valid: false, reason: "Low color variance (possible screenshot)" };
}

// Check aspect ratio
const aspectRatio = metadata.width / metadata.height;
if (aspectRatio > 3 || aspectRatio < 0.33) {
return { valid: false, reason: "Unusual aspect ratio" };
}

return { valid: true, reason: null };
```

---

lib/report-generator.js

Purpose: PDF and Markdown report generation

Functions:

#### `generateMarkdownReport(specimenData)`
Generates Markdown report following Multiview Official Grading Report Format v1.2.

Parameters:

```
{
specimenId: "A-30",
frameworkVersion: "v1.5 Strict++",
grade: "PSA 8.0 (NM)",
subgrades: { geometry: 8.0, ... },
notes: "...",
frontPath: "...",
sidePath: "...",
verificationType: "VO",
issues: []
}
```

Returns: `Promise` - Markdown content

Template Structure:

```
# Multiview Official Grading Report

## Header / Identification
- Specimen ID: A-30
- Date/Time: 2025-10-06T03:00:00.000Z
- Grading Framework: v1.5 Strict++
- Verification: VO
- Overall Grade: PSA 8.0 (NM)

## Photo Set Metadata
| View | File Path | Dimensions | Notes |
|------|-----------|------------|-------|
| Front | ... | Auto-detected | Primary view |
| Side | ... | Auto-detected | Curvature analysis |

## Subgrade Breakdown
| Category | Score | Weight | Notes |
|----------|-------|--------|-------|
| Geometry | 8.0 | 30% | Overall shape |
| Corners | 8.0 | 20% | Corner sharpness |
| Coating | 8.0 | 12% | Surface coating |
| Surface | 8.0 | 20% | Texture |
| Alignment | 8.0 | 18% | Edge alignment |

**Weighted Mean:** 8.00
**Curvature %:** 3.20 (Nominal camber)
**Overall Grade:** PSA 8.0 (NM)

## Detailed Interpretation
[AI-generated analysis paragraph]

## AI Technical Breakdown
**Curvature Analysis:** 3.20% (Nominal camber)
**Weighted Mean:** 8.00
**AI Confidence:** 94.5%
**Framework Applied:** v1.5 Strict++
**Final Grade:** PSA 8.0 (NM)
```

```
## System Hash & Provenance Record
**Multiview Digital Integrity Hash (SHA-256):** a1b2c3d4...xyz
*(Full hash stored in archive metadata)*

**Generated:** 2025-10-06T03:00:00.000Z
**System:** Multiview CTC Grader v2.0 • Framework v1.5 Strict++
**Seed:** 42
**Verification Type:** VO

This cryptographic digest links the specimen's imagery, analysis, and
report...
```

#### `generateEnhancedPDFReport(specimenData, opts)`
Generates 4-page PDF report using PDFKit.

Page 1: Header & Identification
- Multiview Technology header
- Specimen ID, certification ID, classification
- Framework version, date/time, certified by
- Measurements & weight
- Condition summary

Page 2: Subgrade Analysis
- Weighted subgrades table
- Curvature & penalty data
- Final computation
- Analytical notes

Page 3: Provenance & Interpretation
- Manufacturer, box code, best by date
- Capture era
- Personal/observational note
- Interpretation paragraph

Page 4: Appendix & System Hash
- How to read this report
- About Multiview Technology
- Archival policy
- System hash & provenance record
- Certified by signature

Returns: `Promise` - PDF file path

#### `generateReports(specimenData)`
Master function that generates all report formats.

Process:
1. Generate Markdown content
2. Save Markdown to `/Documents/Reports/`
3. Generate system hash (SHA-256 of images + markdown)
4. Generate enhanced 4-page PDF

5. Copy both files to specimen folder
6. Generate JSON metadata file
7. If issues detected, create error note in `/Documents/Errors/`

Returns:

```
{
markdownPath: "D:\\...\\A-30_CTC_Grading_Report.md",
pdfPath: "D:\\...\\A-30_CTC_Grading_Report.pdf",
markdownContent: "...",
specimenMarkdownPath:
"D:\\...\\Specimens\\A-30\\A-30_CTC_Grading_Report.md",
specimenPdfPath: "D:\\...\\Specimens\\A-30\\A-30_CTC_Grading_Report.pdf",
jsonPath: "D:\\...\\Specimens\\A-30\\A-30_CTC_Grading_Report.json",
errorNotePath: null,
systemHash: "a1b2c3d4e5f6..."
}
```

#### `generateErrorReport(errorData)`

Generates error PDF for invalid submissions.

Parameters:

```
{
specimenId: "A-30",
errorType: "Non-CTC Image",
reason: "Image does not appear to be a Cinnamon Toast Crunch piece.",
frameworkVersion: "v1.5 Strict++",
frontPath: "...",
sidePath: "..."
}
```

Returns: `Promise` with `pdfPath`

PDF Content:
- Error report header
- Specimen ID
- Error type
- Reason
- Framework version
- Date
- Submitted images (if available)
- Footer: "No grading was performed"

#### Helper Functions

`computeWeightedMean(subgrades)`
Calculates weighted average of subgrades.

```
geometry * 0.30 + corners * 0.20 + surface * 0.20 + coating * 0.12 +
alignment * 0.18
```

`estimateCurvatureFromGeometry(geometry)`
Estimates curvature percentage from geometry score.

```
const delta = Math.max(0, 10 - geometry);
return Math.min(15, 2 + delta * 1.3);
```

`getCurvatureBand(curvature)`
Returns descriptive band for curvature.
- `> 12%`: "Severe warp"
- `> 8%`: "Warped band"
- `> 4%`: "Nominal camber"
- `≤ 4%`: "Flat band"

`generateSystemHash(imageData, markdownContent)`
Generates SHA-256 hash for provenance.

```
const hash = crypto.createHash('sha256');
hash.update(JSON.stringify(imageData));
hash.update(markdownContent);
const fullHash = hash.digest('hex');
return {
full: fullHash,
short: fullHash.substring(0, 8) + '...' + fullHash.substring(fullHash.length
- 8)
};
```

`generateGeometryNotes(score, curvature)`
`generateCornerNotes(score)`
`generateEdgeNotes(score)`
`generateSurfaceNotes(score)`
`generateCoatingNotes(score)`
`generateInterpretation(specimen)`
`generateTechnicalBreakdown(data)`

These functions generate detailed narrative text for each subgrade category based on scores.

Image Processing Pipeline

Complete Flow: HEIC Upload to Display

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 1. USER SELECTS HEIC IMAGE FROM PHONE CAMERA ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 2. BROWSER CREATES FormData WITH FILE ■
■ - photo: File (IMG_0227.HEIC) ■
■ - specimenId: "A-30" ■
■ - label: "front" ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
```

```
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 3. POST /api/savePhoto (multipart/form-data) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 4. express-fileupload MIDDLEWARE ■
■ - Receives multipart stream ■
■ - Creates temp file: tmp-1-888481759719033079 ■
■ - Writes file data to temp directory ■
■ - Attaches to req.files.photo ■
■ { ■
■ name: "IMG_0227.HEIC", ■
■ size: 1399670, ■
■ mimetype: "application/octet-stream", ■
■ tempFilePath: "D:\\...\\temp\\tmp-1-...", ■
■ data: <Buffer> (empty when useTempFiles: true) ■
■ } ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 5. validateImageFile(file) ■
■ ✓ File exists ■
■ ✓ Has name property ■
■ ✓ Extension is .heic ■
■ ✓ tempFilePath exists on disk ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 6. processUploadedImage(file, destPath) ■
■ - Detects HEIC format ■
■ - Reads from tempFilePath (NOT file.data) ■
■ inputBuffer = fs.readFileSync(file.tempFilePath) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 7. heic-convert LIBRARY ■
■ const outputBuffer = await convert({ ■
■ buffer: inputBuffer, ■
■ format: 'JPEG', ■
■ quality: 1 ■
■ }); ■
■ - Decodes HEIC container ■
■ - Extracts image data ■
■ - Encodes as JPEG ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 8. SAVE JPEG FILE ■
■ fs.writeFileSync( ■
■ "D:\\...\\Specimens\\A-30\\A-30_front.jpg", ■
■ outputBuffer ■
■ ) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 9. CLEANUP TEMP FILE ■
■ fs.unlinkSync(file.tempFilePath) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 10. RETURN SUCCESS RESPONSE ■
■ { ■
■ ok: true, ■
■ path: "D:\\...\\A-30_front.jpg", ■
■ url: "/specimens/A-30/A-30_front.jpg", ■
■ originalFormat: "IMG_0227.HEIC", ■
■ converted: true ■
■ } ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■
▼
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ 11. FRONTEND DISPLAYS IMAGE ■
■ const img = document.createElement('img'); ■
■ img.src = result.url; // "/specimens/A-30/A-30_front.jpg" ■
■ document.getElementById('shots').appendChild(img); ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
```

Key Configuration: express-fileupload

```
app.use(fileUpload({
limits: { fileSize: 20 * 1024 * 1024 }, // 20MB max
useTempFiles: true, // CRITICAL: Save to disk, not memory
tempFileDir: path.join(__dirname, 'temp'), // Temp directory
debug: true // Enable debug logging
}));
```

Why `useTempFiles: true` is critical:
- Without it, files are stored in `file.data` buffer (memory)
- Large HEIC files (1-2MB) can fill memory quickly
- With it, files are written to disk immediately
- `file.tempFilePath` contains the disk path
- `file.data` is empty (just a reference)
- HEIC converter reads from disk path, not buffer

AI Grading System

OpenAI GPT-4o Vision Integration

Model: `gpt-4o` (multimodal: vision + text)

API Call Structure:

```
const response = await openai.chat.completions.create({
model: "gpt-4o",
messages: [
{
role: "system",
content: systemPrompt // Multiview framework rules
},
{
role: "user",
content: [
{
type: "text",
text: "Analyze these two photos of a Cinnamon Toast Crunch specimen..."
},
{
type: "image_url",
image_url: {
url: `data:image/jpeg;base64,${frontImageBase64}`
}
},
{
type: "image_url",
image_url: {
url: `data:image/jpeg;base64,${sideImageBase64}`
}
}
]
}
],
max_tokens: 1500,
temperature: 0.3, // Low temperature for consistency
response_format: { type: "json_object" }
});
```

System Prompt (Multiview Framework)

```
You are an expert grader for the Multiview CTC Grading System.

Framework: Multiview Grading Standards v1.5 (Strict++)

Analyze these two photos of a Cinnamon Toast Crunch specimen:
- FRONT view (straight-on, showing surface and coating)
- SIDE view (showing curvature and thickness)

Grade the specimen on these subgrade categories (0-10 scale):

1. **Geometry (30% weight)**
- Shape accuracy (square vs. distorted)
- Flatness (minimal curvature)
- Aspect ratio (length vs. width)
- Scoring:
- 10: Perfect square, completely flat
- 8: Minor deviation, slight bow
- 6: Visible warping or shape distortion
- 4: Significant deformation
- 2: Severe structural issues

2. **Corners (20% weight)**
- Sharpness (90° angles preserved)
- Rounding (minimal wear)
- Chips or breaks
- Scoring:
- 10: All 4 corners sharp and intact
- 8: Minor rounding on 1-2 corners
- 6: Visible rounding or small chip
- 4: Multiple chips or significant rounding
- 2: Missing corner material

3. **Coating (12% weight)**
- Cinnamon/sugar distribution
- Color uniformity
- Coverage (no bare spots)
- Scoring:
- 10: Perfect even coverage
- 8: Slight imbalance in one quadrant
- 6: Visible bare spots or heavy variance
- 4: Significant coating issues
- 2: Mostly uncoated or burned

4. **Surface (20% weight)**
- Ridge clarity (texture preservation)
- Pocking or scarring
- Smoothness
- Scoring:
- 10: Crisp ridges, no imperfections
- 8: Minor pocking under angled light
- 6: Visible ridge collapse or scarring
- 4: Heavy surface damage
- 2: Crushed or severely degraded

5. **Alignment (18% weight)**
- Edge straightness
- Compression or warping
- Symmetry
- Scoring:
- 10: Perfectly straight edges
- 8: Slight compression on one side
- 6: Visible edge warping
- 4: Multiple edges compressed
- 2: Severe edge damage

**STRICT GRADING RULES:**
- Any subgrade < 8.0 automatically caps overall grade at ≤ 8.0
- Curvature > 7.5% automatically caps grade at ≤ 8.0
- Always round DOWN, never up
- Be conservative and strict
- PSA scale: 10 = Gem Mint, 9 = Mint, 8 = NM-MT, 7 = NM, 6 = EX-MT, etc.

**CURVATURE CALCULATION:**
Estimate curvature as: (max height deviation ÷ half-span) × 100%
- < 4%: Flat band
- 4-8%: Nominal camber
- 8-12%: Warped band
- > 12%: Severe warp

Return ONLY valid JSON (no markdown, no explanation):
{
"frameworkVersion": "v1.5 Strict++",
"grade": "PSA 8.0 (NM-MT)",
"subgrades": {
"geometry": 8.0,
"corners": 8.0,
"coating": 8.0,
"surface": 8.0,
"alignment": 8.0
},
"notes": "Detailed analysis paragraph explaining the grade. Include specific
observations about curvature, coating distribution, corner condition, and
any notable flaws. Mention which factors limited the grade if applicable."
}
```

Response Parsing

```
const content = response.choices[0].message.content;
const gradingResult = JSON.parse(content);

// Validate structure
if (!gradingResult.grade || !gradingResult.subgrades) {
throw new Error('Invalid AI response format');
}

// Ensure all subgrades present
const requiredSubgrades = ['geometry', 'corners', 'coating', 'surface',
'alignment'];
for (const key of requiredSubgrades) {
if (typeof gradingResult.subgrades[key] !== 'number') {
throw new Error(`Missing subgrade: ${key}`);
}
}

return gradingResult;
```

Fallback Handling

If AI grading fails (API error, timeout, invalid response), system uses fallback:

```
gradingResult = {
frameworkVersion: "v1.5 Strict++",
grade: "PSA 8.0 (NM)",
subgrades: {
geometry: 8.0,
corners: 8.0,
coating: 8.0,
surface: 8.0,
alignment: 8.0
},
notes: `AI grading failed: ${error.message}. Using fallback grade. Manual
review recommended.`
};
```

Report Generation

Multiview Official Grading Report Format v1.2

4-Page PDF Structure:

#### Page 1: Header & Identification

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ MULTIVIEW TECHNOLOGY ■
■ OFFICIAL CINNAMON TOAST CRUNCH GRADING REPORT ■
■ ■
■ "Why not take the most ordinary thing and bury it in ■
■ paperwork until it feels important?" ■
■ ■
■ Specimen Identification ■
■ ■■ Specimen ID: A-30 ■
■ ■■ Certification ID: [blank] ■
■ ■■ Classification: VO (Visual-Only) ■
■ ■■ Framework: Multiview Grading Standards v1.5 (Strict++) ■
■ ■■ Date: 10/06/2025 ■
■ ■■ Time: 03:00:00 ■
■ ■■ Certified By: Shawn Wiederhoeft ■
■ ■
■ Measurements & Weight ■
■ ■■ Length × Width × Thickness: not recorded mm ■
■ ■■ Weight: not recorded g ■
■ ■
■ Condition Summary ■
■ The specimen presents balanced morphology with 3.2% ■
■ curvature, suggesting mild production warp. Corners show ■
■ good definition, and the surface retains its ridge network ■
■ despite minor imperfections... ■
■ ■
■ Page 1 of 4 • Cataloged by Multiview Technology ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
```

#### Page 2: Subgrade Analysis

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ Subgrade Analysis ■
■ ■
■ Weighted Subgrades (0–10) ■
■ ■■ Corners: 8.0 (Weight: 0.20) ■
■ ■■ Edges: 8.0 (Weight: 0.18) ■
■ ■■ Surface Integrity: 8.0 (Weight: 0.20) ■
■ ■■ Coating Uniformity: 8.0 (Weight: 0.12) ■
■ ■■ Geometry / Flatness: 8.0 (Weight: 0.30) ■
■ ■
■ Curvature & Penalty Data ■
■ ■■ Curvature: 3.20 % ■
■ ■■ Penalty Triggered: None ■
■ ■■ AI Confidence: 94.5% ■
■ ■
■ Final Computation ■
■ ■■ Weighted Mean: 8.00 ■
■ ■■ Strict-Mode Adjustments: None ■
■ ■■ Rounded Grade: PSA 8.0 (NM-MT) ■
■ ■
■ Analytical Notes ■
■ The specimen demonstrates excellent preservation of its ■
■ structural integrity. All corners remain sharp with minimal ■
■ rounding. The cinnamon coating shows balanced distribution ■
■ across all quadrants. Surface ridges are well-preserved... ■
■ ■
■ Page 2 of 4 • Multiview Technology ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
```

#### Page 3: Provenance & Interpretation

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ Provenance & Interpretation ■
■ ■
■ Provenance ■
■ ■■ Manufacturer: [blank] ■
■ ■■ Box Code / Batch: [blank] ■
■ ■■ Best By: [blank] ■
■ ■■ Capture Era: [blank] ■
■ ■
■ Personal / Observational Note ■
■ [blank] ■
■ ■
■ Interpretation ■
■ This specimen was evaluated under Multiview Grading ■
■ Standards v1.5 (Strict++), applying full strict-mode ■
■ enforcement and curvature cap logic. Any subgrade < 8.0 or ■
■ curvature > 7.5% automatically invoked the grade cap of ■
■ ≤ 8.0. Rounding applied deterministically downward. ■
■ ■
■ The specimen's geometry score of 8.0 reflects minimal ■
■ curvature (3.2%), well within acceptable tolerances. Corner ■
■ preservation is excellent, with all four corners maintaining ■
■ sharp 90° angles. The coating demonstrates uniform ■
■ distribution with no visible bare spots or heavy variance. ■
■ ■
■ Page 3 of 4 • Multiview Technology ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
```

#### Page 4: Appendix & System Hash

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
■ Appendix — Understanding the Report ■
■ ■
■ How to Read This Report ■
■ • Subgrades (0–10): Corners = edge integrity, Edges = ■
■ cracks/uniformity, Surface = ridge clarity, Coating = ■
■ granule balance, Geometry = flatness + aspect ratio. ■
■ • Measurements: Recorded in mm/g using calipers and digital ■
■ scale. ■
■ • Curvature: Max height deviation ÷ half-span × 100 %. ■
■ • PSA Scale: 10 = Gem Mint → 1 = Poor (broken or burned). ■
■ • Strict-Mode: Any uncertainty reduces grade; never rounds ■
■ up. ■
■ ■
■ About Multiview Technology ■
■ Multiview Technology is a conceptual grading authority that ■
■ applies forensic-level analysis to disposable breakfast ■
■ objects. Each specimen passes through an AI-assisted vision ■
■ pipeline measuring geometry, curvature, color variance, and ■
■ ridge frequency. Results are deterministic, weighted, and ■
■ rounded conservatively to enforce discipline in absurdity. ■
■ ■
■ The paperwork is real. The subject is breakfast. ■
■ ■
■ Archival Policy ■
■ Every document—complete or erroneous—is permanently ■
■ preserved for provenance continuity. Error or incomplete ■
■ grades are recorded with the same status as valid reports. ■
■ ■
■ System Hash & Provenance Record ■
■ Multiview Digital Integrity Hash (SHA-256): a1b2c3d4...xyz ■
■ (Full hash stored in archive metadata) ■
■ ■
■ Generated: 2025-10-06T03:00:00.000Z ■
■ System: Multiview CTC Grader v2.0 • Framework v1.5 Strict++■
■ Seed: 42 ■
■ Verification Type: VO ■
■ ■
■ Certified & Catalogued by: Shawn Wiederhoeft • Multiview ■
■ Technology ■
■ Date: 10/06/2025 ■
■ ■
■ Page 4 of 4 • End of Report ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■■■■
```

PDF Generation with PDFKit

```
const doc = new PDFDocument({
size: 'A4',
margins: { top: 72, bottom: 72, left: 72, right: 72 }
});

const stream = fs.createWriteStream(pdfPath);
doc.pipe(stream);

// Page 1
doc.fontSize(24).font('Helvetica-Bold')
.text('MULTIVIEW TECHNOLOGY', { align: 'center' });
```

```
doc.fontSize(18).font('Helvetica-Bold')
.text('OFFICIAL CINNAMON TOAST CRUNCH GRADING REPORT', { align: 'center' });
doc.moveDown(0.5);
doc.fontSize(10).font('Helvetica-Oblique')
.text('"Why not take the most ordinary thing..."', { align: 'center' });
doc.moveDown(2);

// Specimen info
doc.fontSize(14).font('Helvetica-Bold').text('Specimen Identification');
doc.fontSize(11).font('Helvetica')
.text(`Specimen ID: ${specimenId}`)
.text(`Certification ID: ${certificationId || ''}`)
.text(`Classification: ${verificationType || 'VO'}`)
// ... etc

// Page breaks
doc.addPage();

// Repeat for pages 2, 3, 4...

doc.end();
```

File Storage

Storage Structure

```
D:\Projects\CTC_Grading\
■
■■■ Documents\
■ ■■■ ctc_grades.db # SQLite database
■ ■■■ Grading Standards\ # Framework PDFs
■ ■■■ Reports\ # Main report storage
■ ■ ■■■ A-16_CTC_Grading_Report.pdf
■ ■ ■■■ A-16_CTC_Grading_Report.md
■ ■ ■■■ A-18_CTC_Grading_Report.pdf
■ ■ ■■■ ...
■ ■■■ Errors\ # Error reports
■ ■■■ A-27_CTC_Error_Report\
■ ■■■ A-27_CTC_Error_Report.pdf
■
■■■ Specimens\ # Per-specimen storage
■■■ A-16\
■ ■■■ A-16_front.jpg # Converted from HEIC
■ ■■■ A-16_side.jpg # Converted from HEIC
■ ■■■ A-16_CTC_Grading_Report.pdf
■ ■■■ A-16_CTC_Grading_Report.md
■ ■■■ A-16_CTC_Grading_Report.json
```

```
■■■ ...
```

File Naming Conventions

Images:
- `{specimenId}_front.jpg` - Front view (always JPEG, converted if needed)
- `{specimenId}_side.jpg` - Side view (always JPEG, converted if needed)

Reports:
- `{specimenId}_CTC_Grading_Report.pdf` - PDF report
- `{specimenId}_CTC_Grading_Report.md` - Markdown source
- `{specimenId}_CTC_Grading_Report.json` - Metadata

Error Reports:
- `{specimenId}_CTC_Error_Report.pdf` - Error documentation

Public URL Mapping

File Path	Public URL
`D:\...\Specimens\A-16\A-16_front.jpg`	`/specimens/A-16/A-16_front.jpg`
`D:\...\Specimens\A-16\A-16_side.jpg`	`/specimens/A-16/A-16_side.jpg`
`D:\...\Documents\Reports\A-16_CTC_Grading_Report.pdf`	`/reports/A-16_CTC_Grading_Report.pdf`
`D:\...\Documents\Grading Standards\Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf`	`/documents/Grading%20Standards/Multiview_Grading_Standards_v1_5_StrictPlusPlus.pdf`

Static File Serving (Express)

```
app.use('/specimens', express.static(C.specimensDir));
app.use('/reports', express.static(C.reportsDir));
app.use('/documents', express.static(path.join(C.root, 'Documents')));
```

Configuration

Environment Variables

Required:
- `OPENAI_API_KEY` - OpenAI API key for GPT-4o Vision

Optional:
- `PORT` - Server port (default: 3000)

Setting Environment Variables (PowerShell):

```
$env:OPENAI_API_KEY = "sk-proj-..."
$env:PORT = "3000"
```

Setting Environment Variables (Batch):

```
set OPENAI_API_KEY=sk-proj-...
set PORT=3000
```

Configuration Files

`web/package.json`:

```
{
"name": "ctc-grading-web",
"version": "1.0.0",
"type": "module",
"scripts": {
"start": "node server.js"
},
"dependencies": {
"express": "^4.19.2",
"express-fileupload": "^1.4.2",
"heic-convert": "^1.2.4",
"openai": "^4.104.0",
"pdfkit": "^0.14.0",
"sharp": "^0.34.4",
"sqlite3": "^5.1.7",
"md-to-pdf": "^5.2.0"
}
}
```

`web/lib/multiview-config.js`:

```
export const MULTIVIEW_CONFIG = {
root: "D:\\Projects\\CTC_Grading",
standardsDir: "D:\\Projects\\CTC_Grading\\Documents\\Grading Standards",
reportsDir: "D:\\Projects\\CTC_Grading\\Documents\\Reports",
errorsDir: "D:\\Projects\\CTC_Grading\\Documents\\Errors",
specimensDir: "D:\\Projects\\CTC_Grading\\Specimens",
activeFrameworkName: "Multiview Grading Standards v1.5 (Strict++)"
};
```

Deployment

Local Development

Prerequisites:
- Node.js v22.20.0 or higher
- Windows 10/11 (or adapt paths for Linux/Mac)

Installation:

```
cd D:\Projects\CTC_Grading\web
npm install
```

Starting Server:

```
cd D:\Projects\CTC_Grading\web
$env:OPENAI_API_KEY = "sk-proj-..."
& "C:\Program Files\nodejs\node.exe" server.js
```

Or using batch file:

```
@echo off
cd /d D:\Projects\CTC_Grading\web
set OPENAI_API_KEY=sk-proj-...
"C:\Program Files\nodejs\node.exe" server.js
pause
```

Production Deployment (Vercel/Heroku/etc.)

Not currently configured for cloud deployment.

To deploy to cloud:
1. Replace SQLite with PostgreSQL or MySQL
2. Replace local file storage with S3 or similar
3. Add environment variable management
4. Configure build scripts
5. Add HTTPS/SSL
6. Set up domain/DNS

Troubleshooting

Common Issues

#### 1. "npm is not recognized"
Cause: Node.js not in PATH
Solution:

```
& "C:\Program Files\nodejs\node.exe" "C:\Program
Files\nodejs\node_modules\npm\bin\npm-cli.js" install
```

#### 2. "input buffer is not a HEIC image"
Cause: Reading from `file.data` instead of `file.tempFilePath`
Solution: Ensure `useTempFiles: true` in express-fileupload config

#### 3. "Cannot GET /"
Cause: Server not running or wrong port
Solution: Check server is running on port 3000

#### 4. "Uploaded file not found on server"
Cause: File validation failing
Solution: Check `temp` directory exists and is writable

#### 5. "The 'path' argument must be of type string. Received null"
Cause: Database has null paths
Solution: Add null checks in API endpoints:

```
pdfPath: s.pdfPath ? `/reports/${path.basename(s.pdfPath)}` :
`/specimens/${s.specimenId}/${s.specimenId}_CTC_Grading_Report.pdf`
```

#### 6. "Specimens not showing in archive"
Cause: Database empty or records not inserted
Solution: Run `import-specimens.js` to import existing specimens

#### 7. OpenAI API errors
Cause: Invalid API key or rate limit
Solution: Check API key, check OpenAI dashboard for usage/errors

Debug Endpoints

`GET /api/debug` - System health check

Server logs - Check console output for detailed errors

Development Workflow

Adding a New Feature

1. **Plan the feature** - Define requirements and API changes
2. **Update database schema** (if needed) - Add columns to `specimens` table
3. **Create/modify API endpoint** - Add route in `server.js`
4. **Update frontend** - Modify HTML/JS to use new endpoint
5. **Test locally** - Upload test specimens and verify
6. **Document changes** - Update this file

Testing Workflow

1. **Start server:**

```
cd D:\Projects\CTC_Grading\web
& "C:\Program Files\nodejs\node.exe" server.js
```

2. **Open browser:** http://localhost:3000

3. **Test upload flow:**
- Click "Start"
- Upload front photo (try HEIC)
- Upload side photo (try HEIC)
- Click "Submit for Grading"
- Verify results display

4. **Test archive:**
- Click "Browse Archive"
- Verify specimens appear
- Test search functionality
- Click specimen to view details
- Download PDF report

5. **Check files:**
- Verify images in `Specimens\A-##\`

- Verify PDFs in `Documents\Reports\`
- Verify database record

Code Style

- **ES6 Modules:** Use `import`/`export`, not `require`
- **Async/Await:** Prefer over callbacks
- **Error Handling:** Always wrap async calls in try/catch
- **Logging:** Use `console.log` for info, `console.error` for errors
- **Comments:** Document complex logic and API contracts

Appendix

Grading Scale Reference

PSA Grade	Label	Description
10	Gem Mint	Perfect specimen, no flaws
9	Mint	Near-perfect, minimal imperfections
8	NM-MT	Near Mint to Mint, minor flaws
7	NM	Near Mint, visible but minor issues
6	EX-MT	Excellent to Mint, moderate wear
5	EX	Excellent, noticeable wear
4	VG-EX	Very Good to Excellent
3	VG	Very Good, significant wear
2	Good	Heavy wear, structural issues
1	Poor	Broken, burned, or destroyed

Subgrade Weights

Category	Weight	Focus
Geometry	30%	Shape, flatness, aspect ratio
Corners	20%	Sharpness, rounding, chips
Surface	20%	Ridge clarity, pocking
Alignment	18%	Edge uniformity, compression
Coating	12%	Cinnamon/sugar distribution

File Size Limits

- **Upload:** 20MB per file
- **HEIC:** Typically 1-2MB
- **JPEG:** Typically 3-5MB after conversion
- **PDF:** Typically 3-7KB (4 pages)

Browser Compatibility

- **Chrome:** ■ Fully supported
- **Firefox:** ■ Fully supported
- **Safari:** ■ Fully supported (iOS camera works)
- **Edge:** ■ Fully supported
- **IE11:** ■ Not supported (ES6 modules)

Version History

v2.0 (Current)

- Complete rewrite with Express.js
- HEIC to JPEG conversion
- SQLite database integration
- 4-page PDF reports
- Browse archive interface
- Image validation (anti-screenshot)
- CTC classification check
- System hash for provenance
- Error report generation

v1.0 (Legacy)

- Basic Python CLI tool
- Folder watching
- Simple PDF reports
- CSV logging
- No web interface

Credits

Developer: Shawn Wiederhoeft
Framework: Multiview Grading Standards v1.5 (Strict++)
AI Model: OpenAI GPT-4o Vision
License: Proprietary

Last Updated: October 6, 2025
Document Version: 1.0
System Version: 2.0