# CAB432 REST API — Evidence Report

## 1. Summary

The submitted application is a Video Transcoder API deployed on AWS EC2, implemented with FastAPI + Uvicorn running inside a Docker container. The API supports JWT authentication with role-based access (admin and user), and all endpoints are versioned under /api/v1. A simple web client is included for login, upload, listing media, and triggering video transcoding tasks.

## 2. Deployment Details

* Platform: EC2 (Ubuntu 24.04)
* Container: videoapi-ecr
* Image: 901444280953.dkr.ecr.ap-southeast-2.amazonaws.com/srimanjary/cab432/123:latest
* Port Mapping: 8080 → 8000
* Health Check: GET /health returns 200 (see attached screenshot).

## 3. Core Criteria (All Met)

|  |  |  |
| --- | --- | --- |
| Criterion | How It’s Met | Primary Evidence |
| Containerise the app | Dockerfile builds the image, which is pushed to ECR. | docker ps shows running ECR image; push logs. |
| Deploy the container | Image pulled from ECR and run on EC2 with port mapping. | docker ps with ECR URL; /health returns 200. |
| User login (JWT + roles) | POST /api/v1/auth/login issues JWT. Admin-only endpoint /api/v1/admin/metrics requires role. | 403 for user, 200 for admin (curl test). |
| REST API | Endpoints under /api/v1/\*, OpenAPI at /docs and /openapi.json, correct status codes. | Health + Docs screenshots. |
| Data types (≥2 non-auth) | Stores Media records (uploaded videos) and Jobs (transcoding tasks). | UI screenshot: upload returns video\_id, job status with ffmpeg output. |
| CPU-intensive task | ffmpeg transcoding (H.264, 720p) triggered by API/UI. | Transcode UI output showing ffmpeg timings. |
| CPU load testing | Transcoding loads container CPU >80% for several minutes. | Screenshot showing >1000% CPU during job execution. |

## 4. Additional Criteria

|  |  |  |
| --- | --- | --- |
| Criterion | Implementation | Evidence |
| Extensive API features | Versioning implemented. Listing endpoints support pagination & sorting. | OpenAPI schema (page, per\_page, sort). |
| Web client | Browser UI supports login → upload → list → transcode workflow. | Two UI screenshots showing process flow. |
| Not Claimed | External APIs, extra data types beyond media/jobs, custom processing, IaC, special features. | — |

## 5. Reproduction (Verification Steps)

Markers can quickly verify the application with the following commands:

# Health check  
curl -sI http://13.54.32.123:8080/health | head -n1  
  
# Docs endpoint  
curl -sI http://13.54.32.123:8080/docs | head -n1  
  
# JWT login (admin)  
AT=$(curl -s -X POST http://13.54.32.123:8080/api/v1/auth/login \  
 -H 'Content-Type: application/json' \  
 -d '{"username":"admin","password":"admin123"}' | jq -r '.access\_token')  
  
# Role-based endpoint  
curl -i -H "Authorization: Bearer $AT" http://13.54.32.123:8080/api/v1/admin/metrics

## 6. Screenshot Index

1. UI – Login, Upload, Transcode: token issued, upload returns video\_id, job with status 'completed' and ffmpeg summary.
2. UI – Second Run: additional uploads and jobs displayed.
3. Health Endpoint: /health returns { 'ok': true }.
4. CPU Load: Container CPU >1000% during transcoding.
5. (Optional) docker ps showing ECR image and port mapping.

## 7. Notes

• The application demonstrates persistence of media records and transcoding jobs.  
• CPU load evidence shows sustained heavy usage, fulfilling the load test requirement.  
• OpenAPI documentation confirms pagination and sorting parameters.  
• Features map directly to the CAB432 marking rubric, with clear, verifiable proof attached.

## ✅ Final Outcome:

All Core criteria are fully met. Two Additional criteria (Extensive API features, Web Client) are demonstrated. Optional features such as custom ML processing, external APIs, or IaC were not implemented and are not claimed for marks.