Low Level Design(LLD)

PlayWave: Video Streaming WebApp Remaining

Revision Number: 1.0
Last date of revision: 2025/02/03
Written By: Bishal Shahi

Document Version Control

Change Record:

Date Issued	Version	Description	Author
03/02/2025	1.0	Document Initialized	Bishal Shahi

Reviews:

Date Issued	Version	Description	Author
02/02/2025	1.0	Document Content	Bishal Shahi

Approval Status:

Date Issued	Version	Reviewed By	Approved By

Contents

1. Introduction
    1.1 Why this Low Level Design Document?
    1.2 Scope
    1.3 Definitions
2. Component-Level Design
    2.1 Backend Components
    2.2 Frontend Components
    2.3 Database Schema
3. API Endpoints and Routes
    3.1 Video Upload API
    3.2 Video Processing API
    3.3 Video Streaming API
4. Error Handling and Logging
5. Security Considerations
6. Conclusion

1. Introduction

1.1 Why this Low Level Design Document?

The purpose of this **Low Level Design (LLD)** document is to define the component-level architecture for **PlayWave Video Streaming App**. It provides in-depth technical details regarding **backend, frontend, APIs, and database schemas**, ensuring smooth implementation.

This document will:

- Describe the **detailed design of each module**.
- Define **API endpoints and their behavior**.
- Specify **error handling and security measures**.

1.2 Scope

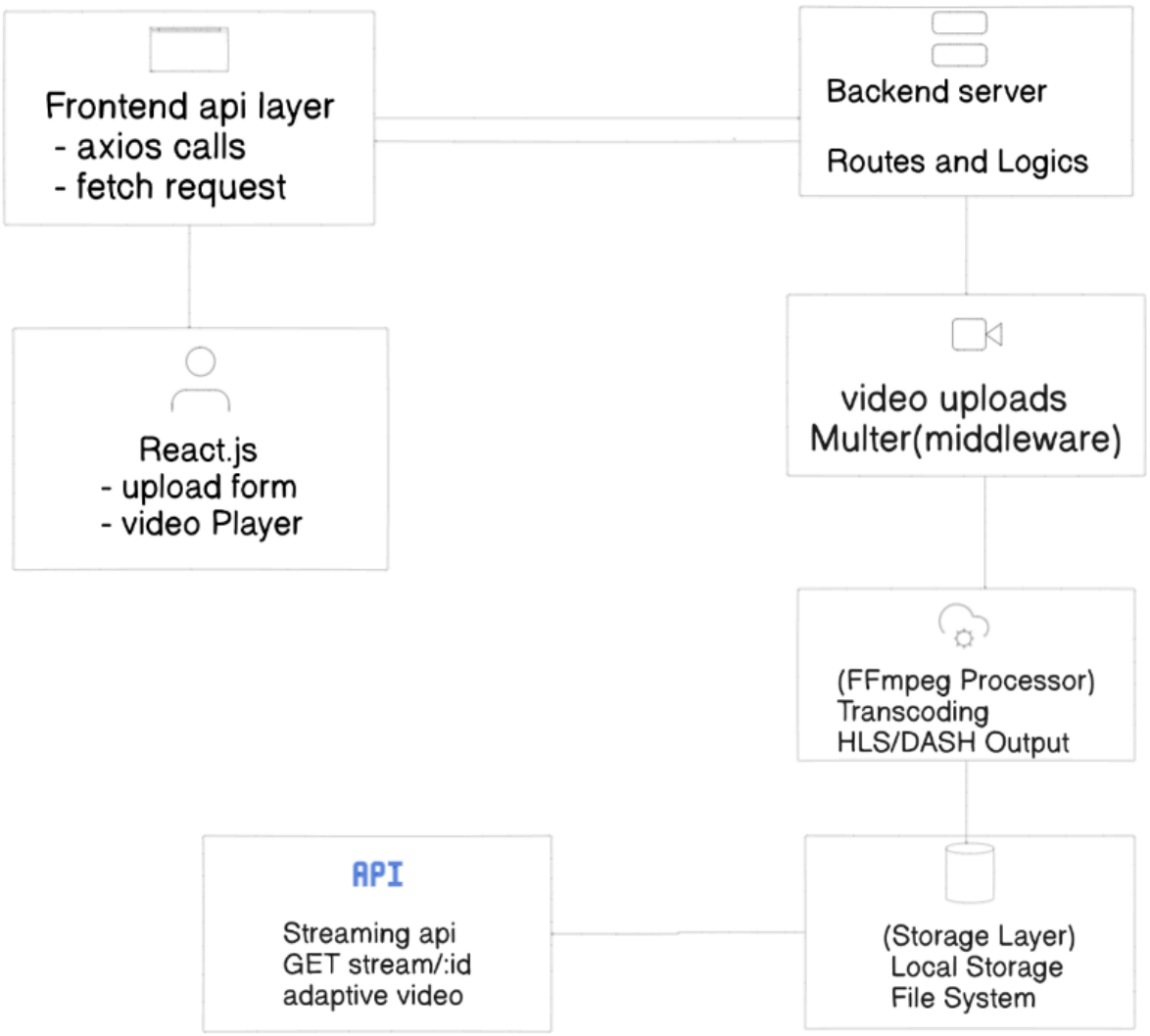
The **LLD** focuses on breaking down **each system component** into detailed **functional blocks**, including how they **interact with each other**.

1.3 Definitions

- **Multer**: Middleware for handling file uploads in Node.js.
- **FFmpeg**: A tool for video processing and conversion.
- **Express.js**: Web framework for handling API requests.

2. Component-Level Design

2.1 System Architecture (Enhanced Overview)



2.2 API Endpoints and Routes

2.2.1 Video Upload API

- **POST /upload** – Uploads video file to local storage.

- **Middleware**: Multer (for handling file uploads).

2.2.2 Video Processing API

- **POST /process/:id** – Converts uploaded video to HLS/DASH using FFmpeg.

2.2.3 Video Streaming API

- **GET /stream/:id** – Serves video files to the frontend player.

3. Error Handling and Logging

- **Try-Catch Blocks**: Used for handling errors in API routes.
- **Winston Logger**: Logs API calls and errors for debugging.
- **Validation Middleware**: Ensures only valid files are uploaded.

4. Security Considerations

- **Rate Limiting**: Prevents abuse of API calls.
- **CORS Policy**: Restricts access to trusted origins.

5. Conclusion

The **PlayWave Low Level Design (LLD)** provides a comprehensive breakdown of **backend, frontend, API endpoints, and security features**. This ensures **structured development, efficient debugging, and robust security** for a seamless video streaming experience.