Low-Level Design (LLD)

Overview

This document outlines the architecture and components of a system designed to process images based on data provided in CSV files. The system is built using modern technologies and follows a modular approach to ensure efficiency and flexibility.

Table of Contents

1. System Purpose

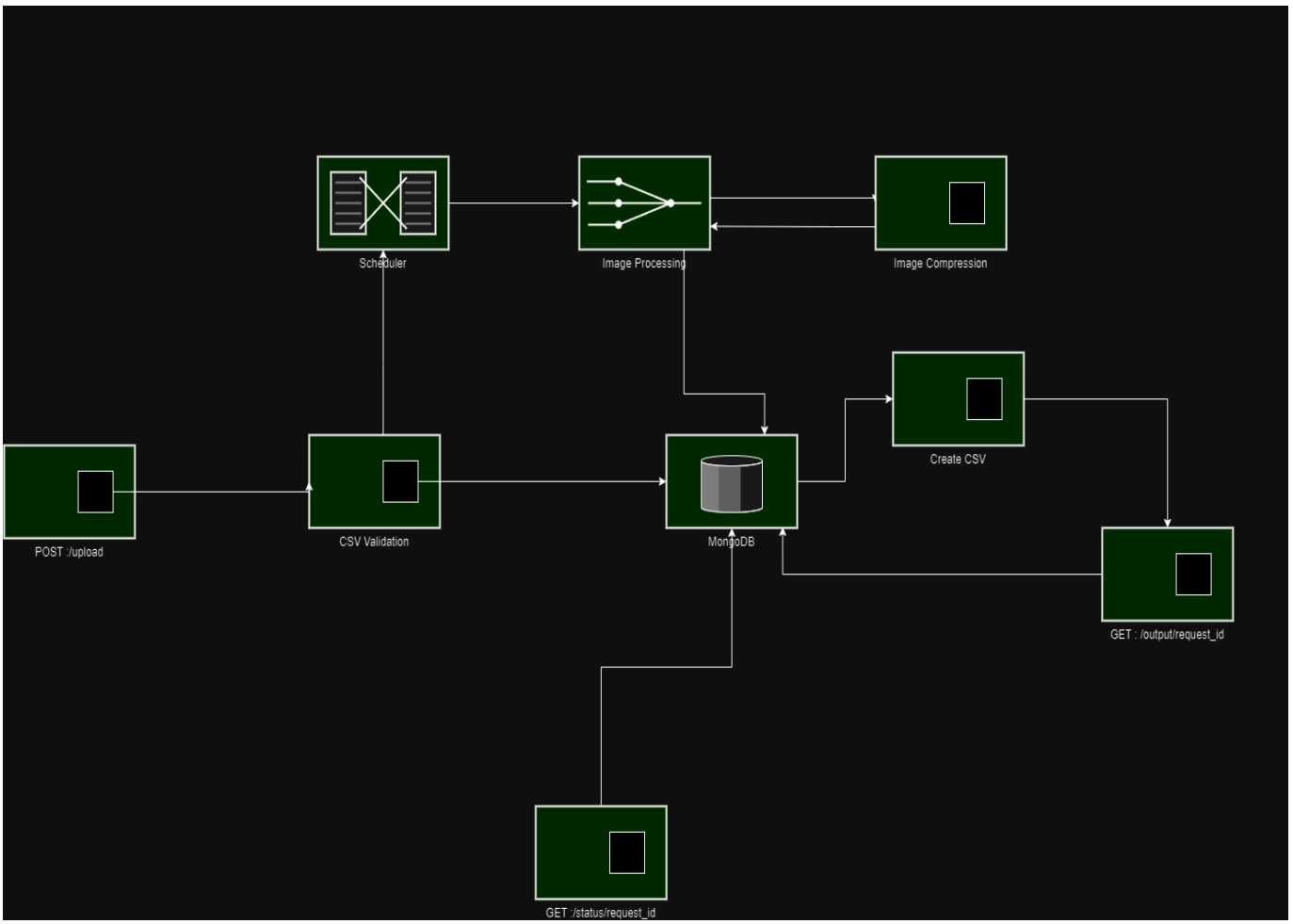
The system is designed to automate the processing of images based on a list of URLs contained in CSV files. It transforms these images according to specified parameters and provides users with processed images and metadata.

2. Architectural Overview

The system architecture is modular, dividing responsibilities among different components to promote scalability and ease of maintenance. Each module handles a specific aspect of the processing workflow, ensuring a clear separation of concerns.

3. Detailed Component Analysis

Component Diagram



3.1. Core Server

- **Functionality:** Acts as the entry point for all HTTP requests.
- **Responsibilities:** Manages endpoints for uploading files, checking processing status, and downloading results.
- **Technology Used:** Express.js

3.2. File Handling Middleware

- **Functionality:** Manages file uploads and temporary storage.
- **Responsibilities:** Handles the reception and temporary storage of CSV files during upload.
- **Technology Used:** Multer

3.3. Data Parsing Module

- **Functionality:** Converts CSV data into a structured format for processing.
- **Responsibilities:** Transforms CSV content into JSON objects for further handling.
- **Technology Used:** csv-parser

3.4. Image Transformation Engine

- **Functionality:** Performs image modifications such as resizing and format conversion.
- **Responsibilities:** Applies various transformations to images based on processing requirements.
- **Technology Used:** Sharp

3.5. Database Interface

- **Functionality:** Interfaces with the MongoDB database for data storage.
- **Responsibilities:** Handles database operations including storing metadata and processed image URLs.
- **Technology Used:** Mongoose

Request Field

Field Name	Data Type	Required	Unique	Default Value	Description
request_id	String	Yes	Yes	N/A	Unique ID for the request, used to track individual requests.
status	String	Yes	No	'PENDING'	Current status of the request. Possible values are 'PENDING', 'PROCESSING', 'COMPLETED', or 'FAILED'.
csv_data	String	No	No	N/A	Contains the CSV data related to the request, formatted as a JSON string.
created_at	Date	No	No	Current Date/Time	Timestamp when the request was created.
updated_at	Date	No	No	Current Date/Time	Timestamp when the request was last updated.

Product Field

Field Name	Data Type	Required	Unique	Default Value	Description
request_id	String	Yes	No	N/A	Links the product to a specific request using its unique ID.
product_name	String	Yes	No	N/A	The name of the product.
input_image_urls	Array of Strings	Yes	No	N/A	List of URLs pointing to the input images associated with the product.
output_image_urls	Array of Strings	No	No	[] (empty array)	List of URLs pointing to the processed images for the product.

3.6. Job Management

- **Functionality:** Schedules and manages asynchronous tasks.
- **Responsibilities:** Manages the execution of image processing jobs.
- **Technology Used:** Agenda

3.7. Network Request Handler

- **Functionality:** Handles external HTTP requests to retrieve images.
- **Responsibilities:** Downloads images from URLs specified in the CSV files.
- **Technology Used:** Axios

3.8. Output File Generator

- **Functionality:** Creates downloadable CSV files with processed image data.
- **Responsibilities:** Generates and provides CSV files containing the results of image processing.
- **Technology Used:** csv-writer

4. Processing Pipeline

1. **Upload CSV File:** Users submit a CSV file through the `/upload` route.
2. **Parse CSV Data:** The system parses the file into a format suitable for processing.
3. **Schedule Processing Jobs:** Each image processing task is queued using the job management system.
4. **Download and Transform Images:** Images are retrieved and processed according to specified rules.
5. **Store Data:** Results and metadata are stored in the database.
6. **Monitor Status:** Users can check the status of their processing request.
7. **Download Results:** Processed data is available for download as a CSV file.

5. Summary

This document provides a thorough overview of the system’s design and operational framework, detailing each component's role and how they work together to achieve the project's objectives.