Data Foundation Systems
CS4.409

AI/CV Pipeline (images)

Project Category: AI / CV Pipeline
Team Name: Straw Hats
Team Members:

Name	Roll no.
Jasika Shah	2021201063
Vishal Mani Mishra	2021201050
Nileshkumar Jha	2021202017

Table of Contents

1. Introduction

Building an Artificial Intelligence (AI) / Computer Vision (CV) project for autonomous driving, medical imaging or robotics, etc can be difficult and will require tracking, logging, analyzing and micromanaging multiple tasks. The enormous volume of data and multiplicity of iterations in the project necessitates the requirement of a proper scalable AI/CV Pipeline. This aids teams in creating, managing, and maintaining the constant inflow of high-quality training data within their computer vision pipelines.

2. Project Description

- Build AI/CV pipeline platform that enables users to add multiple Tools and provide input images to get the desired output.
- Build multiple services as a part of our platform to provide different functionalities.
- Provide an interactive and secure user interface that allows users to drag and drop required CV Tools and upload input images and requirements.
- Integrated multiple AI/CV and Data Processing components that will be required throughout the life cycle of the services.
- Deployment Scripts: Scripts for deploying the AI/CV pipeline in a production environment, including any necessary infrastructure setup and configuration.
- Execution of the CV Tools and providing the final pre-processed data to the end user.

3. Requirements

3.1 Functional Requirements

3.1.1 AI/CV Components:

The application must provide great performing computer vision tools that work well with C++ as well as Python. We analyzed already existing open source tools like OpenCV, which is prebuilt with all the necessary techniques and algorithms to perform several image and video processing tasks. Further, the implementation of various necessary tools was done to provide users with different data processing components.

3.1.2 Authentication:

Authentication ensures that users validate their identity before performing certain software functions. This functional requirement is used to specify the role of the end-users to ensure that proper individuals have access to the sensitive data. End-Users can register themselves as Admin or User. The admin has the right to add components to the system. Users can upload their images and use those components in their pipeline.

3.1.3 Input/Output Interface:

The pipeline has clear input and output interfaces that allow for the seamless transfer of data between different stages of the pipeline. The pipeline has well-defined protocols and standards for data transfer, such as APIs or file formats. This helps to ensure that the data is consistent and

compatible between different stages of the pipeline, regardless of the technology or tools used. For any two consecutive components in the pipeline the output of the first component must be consistent with input of the next component.

3.1.4 AI/CV Pipeline Preview

This is a high-level overview of the AI/CV pipeline as a whole that depicts the various components that are selected by the end-user from CV Toolbar provided on the front-end. This preview will help the end-user to analyze and verify the working of the pipeline.

3.1.5 UI requirements

- The user interface is key to application usability. The application will have a secure, convenient and extensible user interface. It would enable end-users to select tools for preprocessing of their images from the toolbar provided and then order them appropriately to set-up the pipeline. An efficient user-interface in terms of both speed and use for preprocessing their images will provide seamless experience to build AI/CV projects.
- The admin will be additionally provided the feature to upload a component as a zip file containing the source code, requirements.txt and config file. The newly added component will be immediately displayed on the CV toolbar in the front-end.

3.1.6 Pipeline History

The end-user can view their previously created pipeline results (intermediate and final) by logging in into their account. The web page displays a history section that lists all the previous pipelines, where users can view the results by clicking on the desired pipeline name.

3.2 Non-Functional Requirements

3.2.1 Portability:

This application will have the ability to be easily transferred to different hardware or software environments, which will be achieved through the use of virtual machines. VMs allow multiple languages to be compiled down to machine code without needing separate hardware-specific compilers for each individual language. This increases code portability across hardware, efficiency for developers, and makes for a better development ecosystem.

3.2.2 Scalability and Maintainability

The implementation of the application is separated into independent modules. The internal details of individual modules will be hidden behind a public interface, making each module easier to understand, test and refactor independently of others. The AI/CV pipeline should also be scalable and it must be able to handle increasing amounts of data and computational load without sacrificing performance. The key aspect for scalable systems is Data Scalability, Model Scalability, Computational

Scalability, Cloud Scalability, Load Balancing, Optimal Resource Allocation.

3.3.3 Security

- The platform will use HTTP connection and not HTTPS, thus we will encrypt passwords with a secure hashing algorithm such as SHA 256 with random nonce generated from the server. Instead the password was sent as plaintext in the request body which can be read by any attacker eavesdropping on the network.
- Whenever a client registers, the password is hashed and encrypted using SHA256 and stored in the database. SHA256 uses a one-way hash function, which means that it takes an input message of any size and produces a fixed-size output of 256 bits (32 bytes), which is unique for every input message. It is stored in the database in hexadecimal values.
- Whenever a client logs in, its password is hashed and compared with the original hashed value stored in the database. Since a secret key is used to hash passwords, the same hash values are generated if the genuine user login with the correct password. User is directed to the home page after valid authentication.

3.3.4 Reliability

The software system needs to consistently perform the specified functions without failure. The reliability requirements will consider needs regarding possible causes of AI/CV pipeline system failure, preventative actions or

procedures necessary to avoid failure, failure classes, and reliability metrics.

3.3.5 Performance

Performance is a key non-functional requirement of an AI/Computer Vision (CV) pipeline, as it directly impacts the quality and effectiveness of the results produced. Some specific performance-related considerations for an AI/CV pipeline include:

- **Processing Time:** The time it takes for the AI/CV pipeline to process a given dataset or image should be within an acceptable range and should be efficient enough for the desired use case.
- **Latency:** The latency of the AI/CV pipeline, or the time it takes to produce a result after a request is made, should be low enough for the desired use case.
- **Throughput:** The throughput of the AI/CV pipeline, or the number of requests it can handle per unit of time, should be high enough for the desired use case.
- **Resource Utilization:** The AI/CV pipeline should be optimized to minimize the use of resources such as memory, CPU, and storage, to ensure efficient performance.

4. Architecture

We propose a distributed platform that provides build, development and deployment functionalities for different AI/CV Tools. Most data becomes useless just seconds after it is generated, so having the lowest latency possible between the data and the decision is critical. With this platform, we bring AI preprocessing capabilities to the edge gateway. The platform consists of the following microservices:

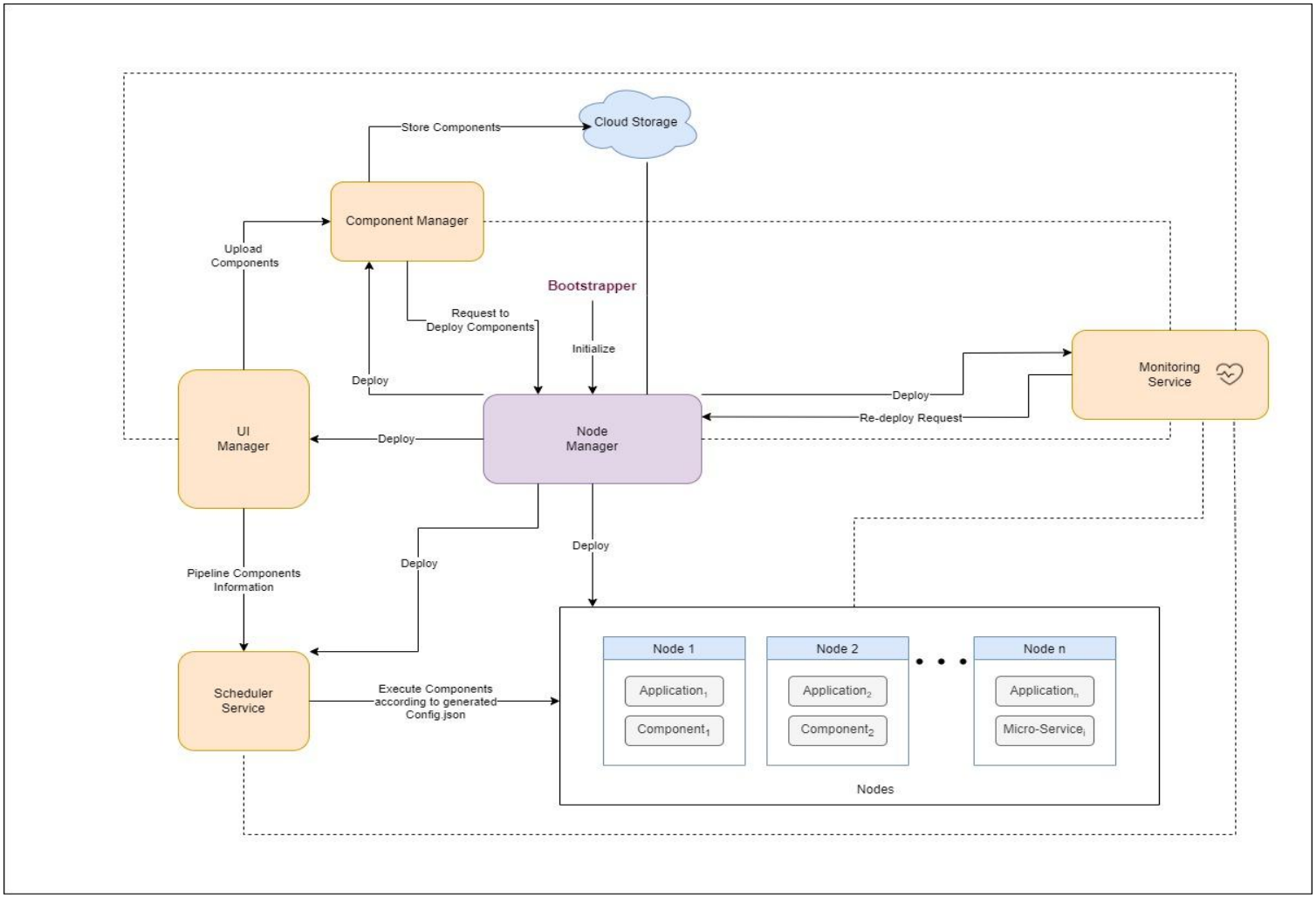

Figure 1: Overall System Architecture

4.1 UI Manager:

- Provides a CV components toolbar to create a customized CV pipeline.
- Provide a functionality for the admin to upload components as a zip file, which is then displayed on CV Toolbar ready for use.
- Redirects the requests to deploy Components uploaded by admin to the Component Manager.
- Display intermediate and final output to the end users after the execution of the pipeline.
- Provide preview of the pipeline and enable removing and adding Components from the pipeline before the execution.
- Enable the users to view their previous pipeline results in the history section.

4.2 Component Manager:

- Admin can upload Component as a zip file. The contents of the zip file should be as follows.

```
└── Component
    ├── config.json
    ├── utility.py
    ├── requriements.txt
    └── src
        └── src.py
```

- Wraps a node.js server with /preprocess endpoint to call preprocess function inside utility.py file.

4.3 Node Manager

- First service to be initialized by the bootstrapper.
- Deploy all services such as UI-Manager, Component Manager, Scheduler, Monitoring Service.
- Whenever an Admin uploads a component as a zip file, Node Manager downloads it, decompresses it, adds all the dependencies and deploys them.
- Run these nodes with the least load.
- Perform load balancing.

4.4 Scheduler

- Generates a config.json using the pipeline Components specified by the end-user through UI.
- Execute the Component according to config.json by providing the input and fetching the output while checking the consistency i.e. output of one Component must match the input of the next Component in the pipeline.
- Scheduler executes each step of the described pipeline sequentially.
- Provides the end output of the pipeline to the UI as well as intermediate steps.
- Scheduler will be based on dispatcher-worker model which is a variant of master-slave model.

4.5 Nodes

The nodes are running Virtual Machines of different microservices and CV Tools which are deployed on the platform. A node is a VM that is created and managed by a virtualization platform, Amazon EC2. Each node runs its own operating system and can be configured with specific resource allocations, such as CPU, memory, and storage. EC2 is an on-demand computing service on the AWS cloud platform. EC2 provides us the facility to scale up or scale down as per the needs. All dynamic scenarios can be easily tackled by EC2 with the help of this feature. And because of the flexibility of volumes and snapshots, it is highly reliable for its users.

4.6 Monitoring Service:

- Health check calls /health endpoint of every registered service to check if the service is running. If found not running, it communicates with Node Manager to get a machine with least load and run the service on that machine.
- Health check pings all the service's IP to check if any machine is down or out of the network. If found not reachable, another instance of the same service is relaunched.

5. REST API's

5.1 UI Manager

1. `/login`:
2. `/signup`:
3. `/logout`:
4. `/upload`:
5. `/compUpload`:
6. `/postjson`:
7. `/viewResult`:
8. `/health`:

5.2 Scheduler

1. `/createPipeline`:
2. `/health`:

5.3 Node Manager

1. `/deploy`:
2. `/redeploy`:
3. `/health`:

5.5 Components

1. `/run`:
2. `/health`:

6. Representation

6.1 Use Case Diagram

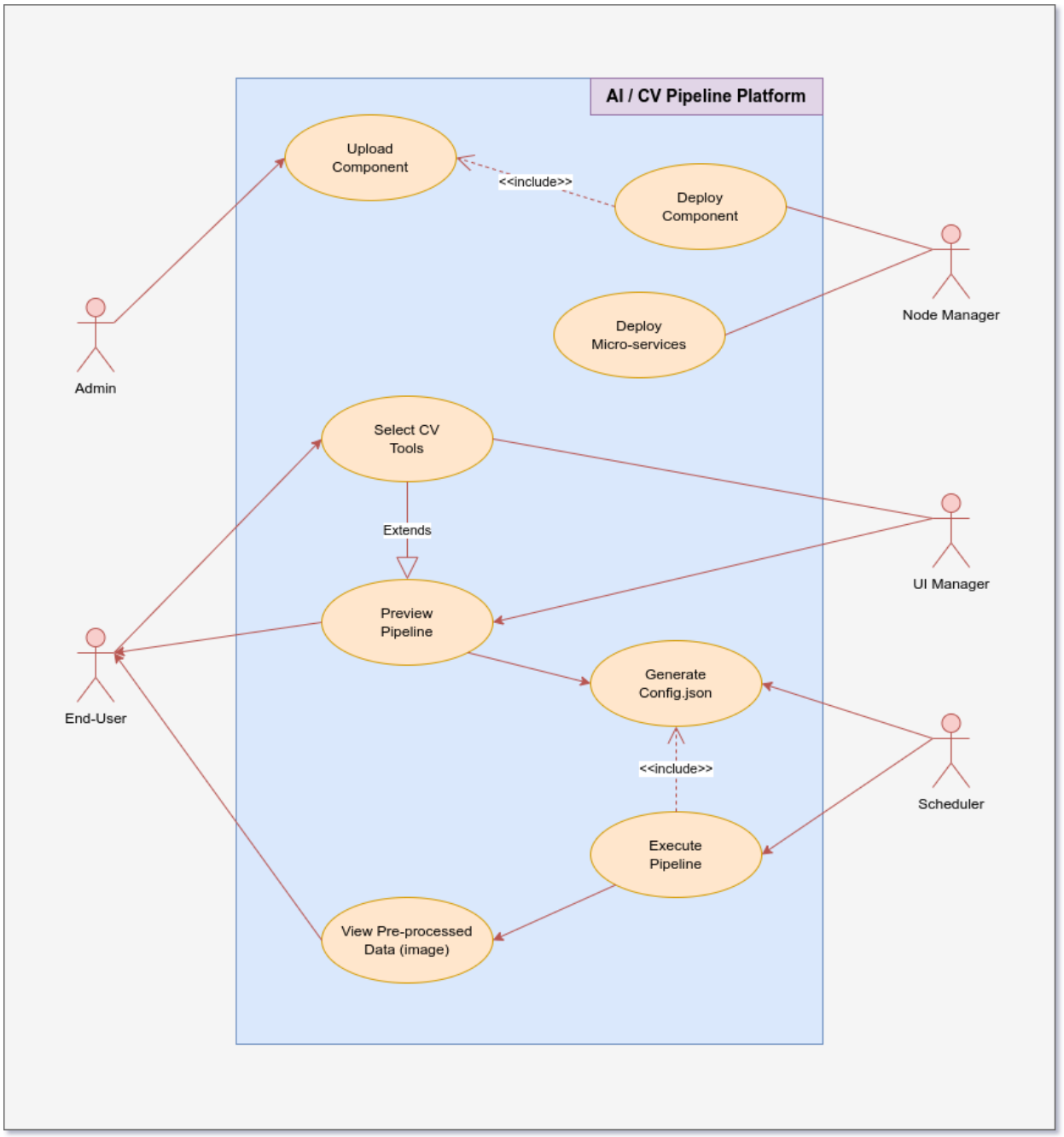

Figure 2: Use case diagram

6.2 ER Diagram

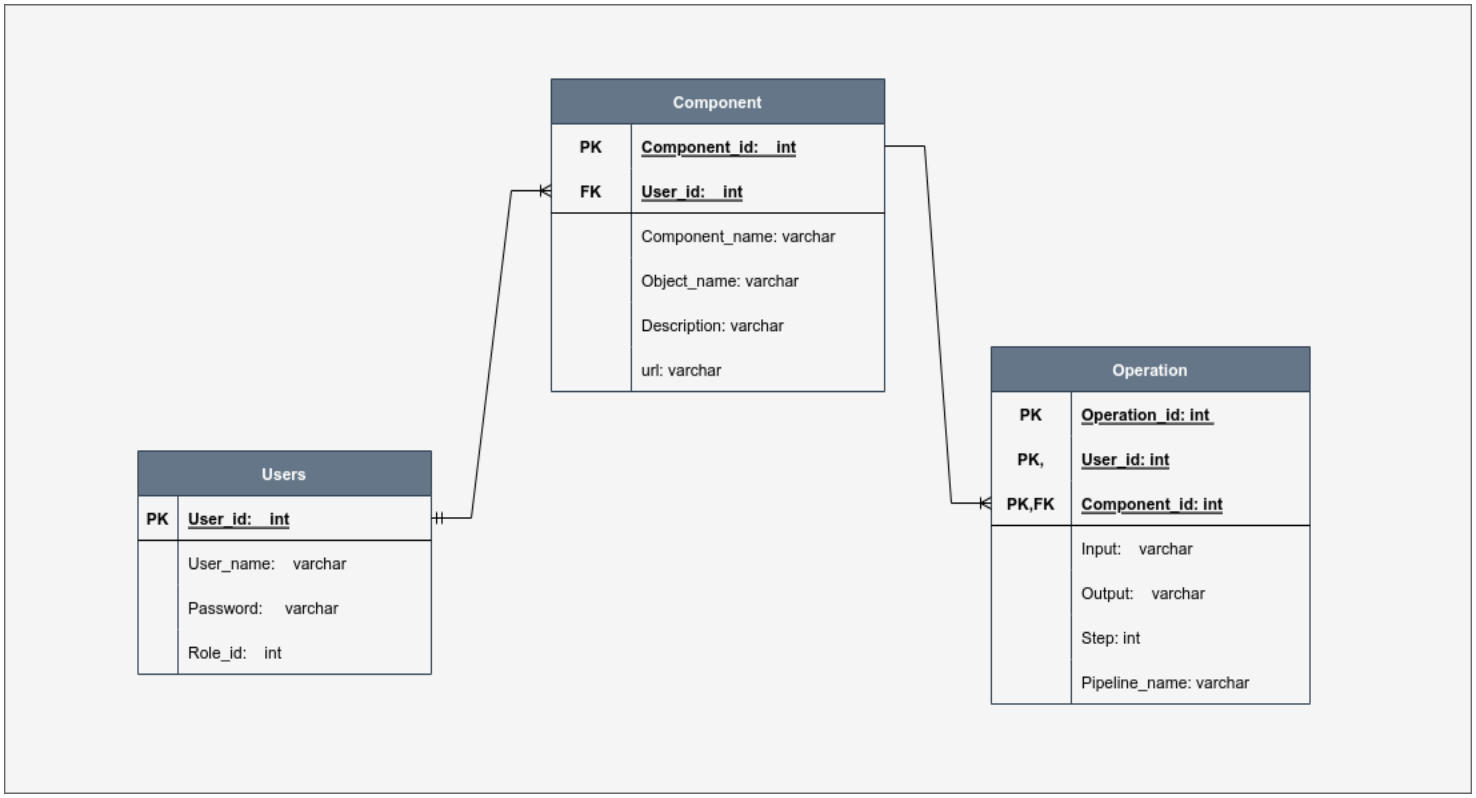

Figure 3: ER Diagram

6.3 Sequence Diagram

6.3.1 Component Upload Sequence Diagram

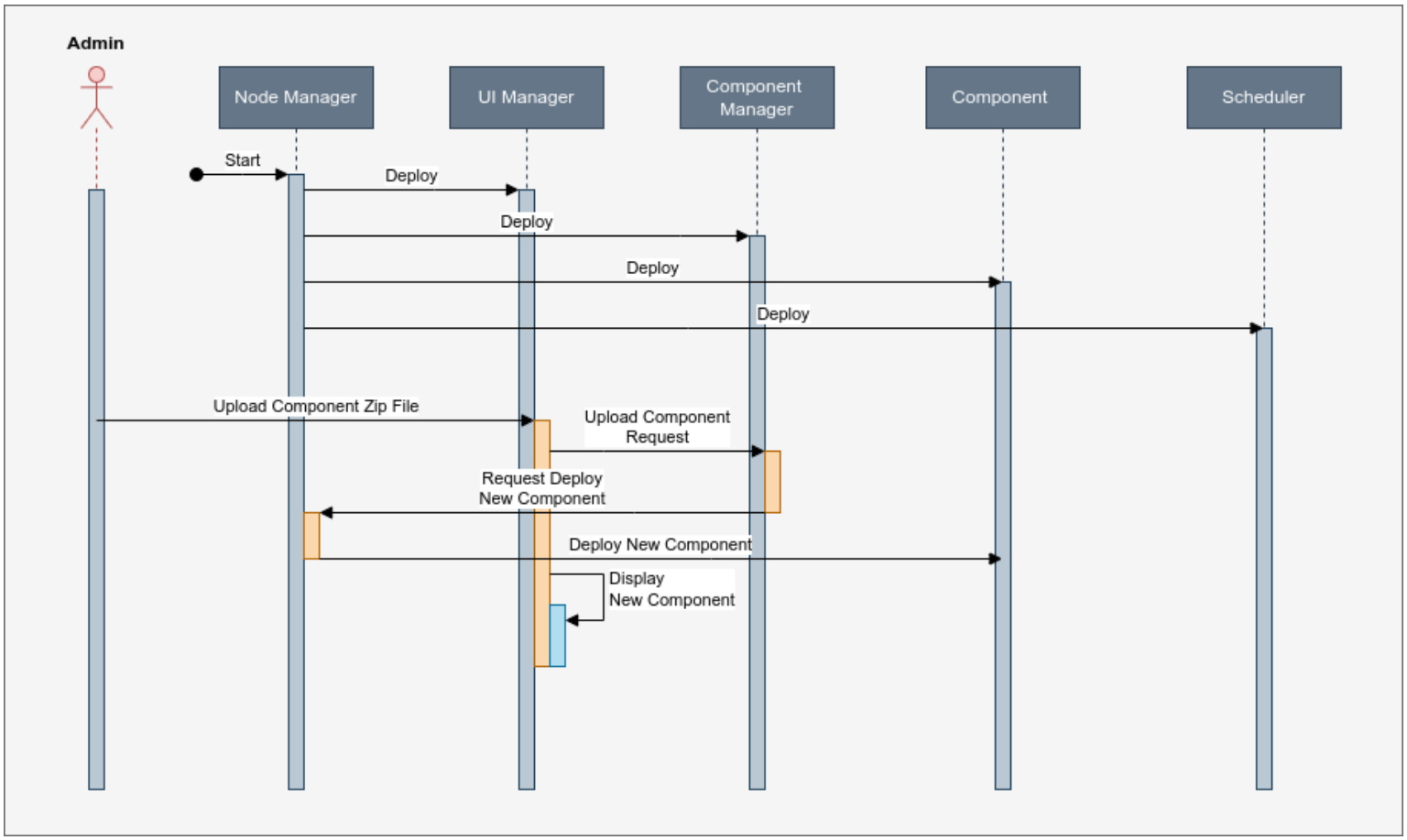

Figure 4: Component Upload Sequence Diagram

6.3.2 CV Pipeline Working Sequence Diagram

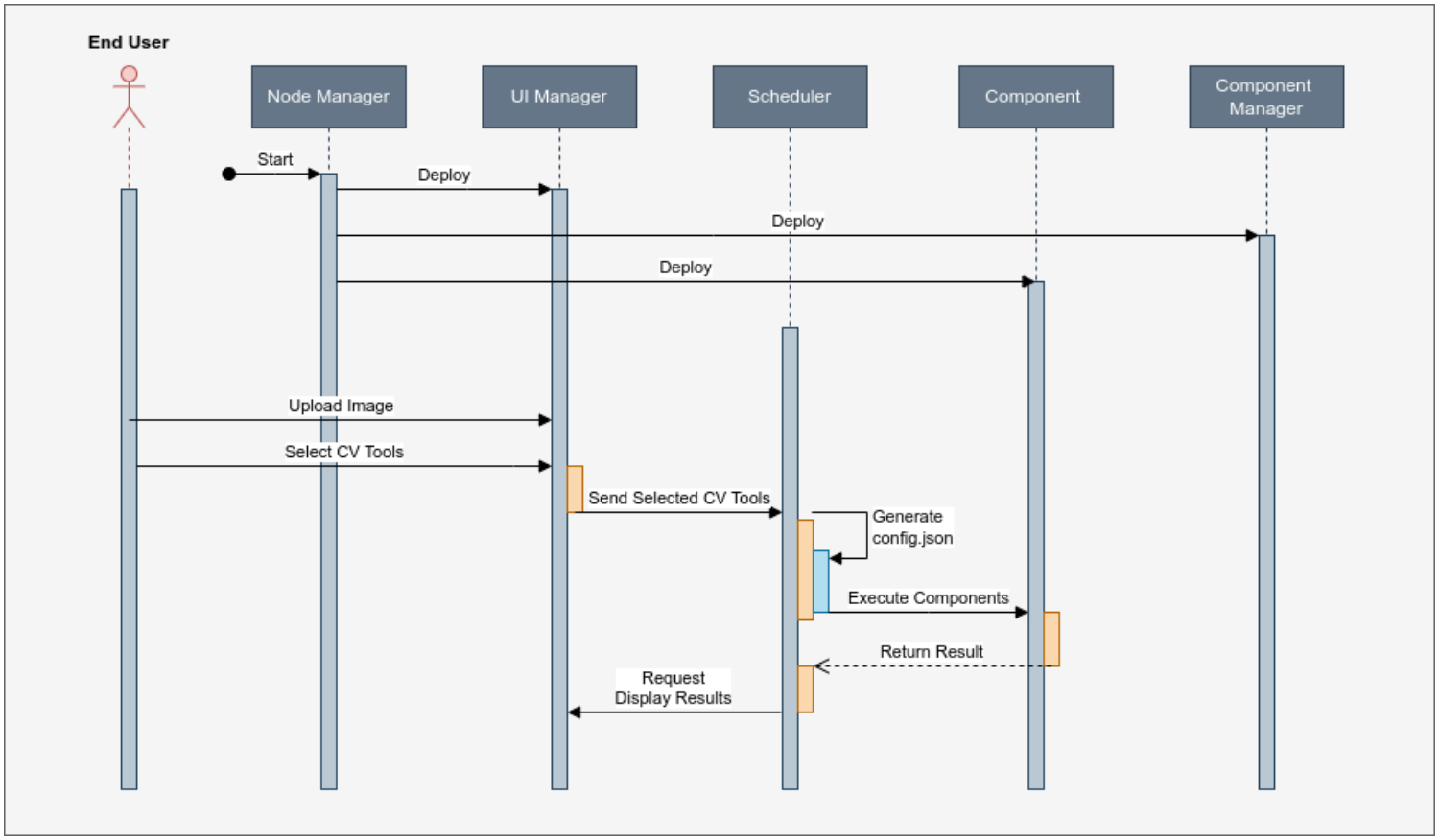

Figure 5: CV Pipeline working Sequence Diagram

6.4 Front-end

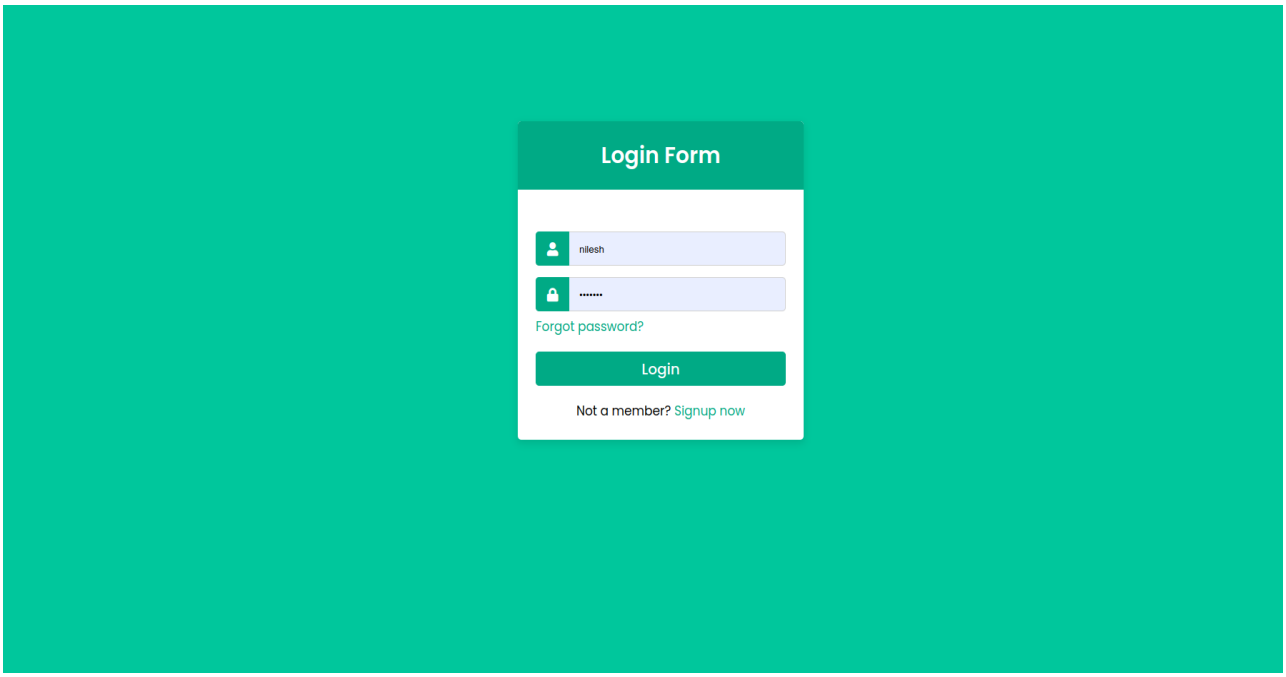

Figure 6: Login Page

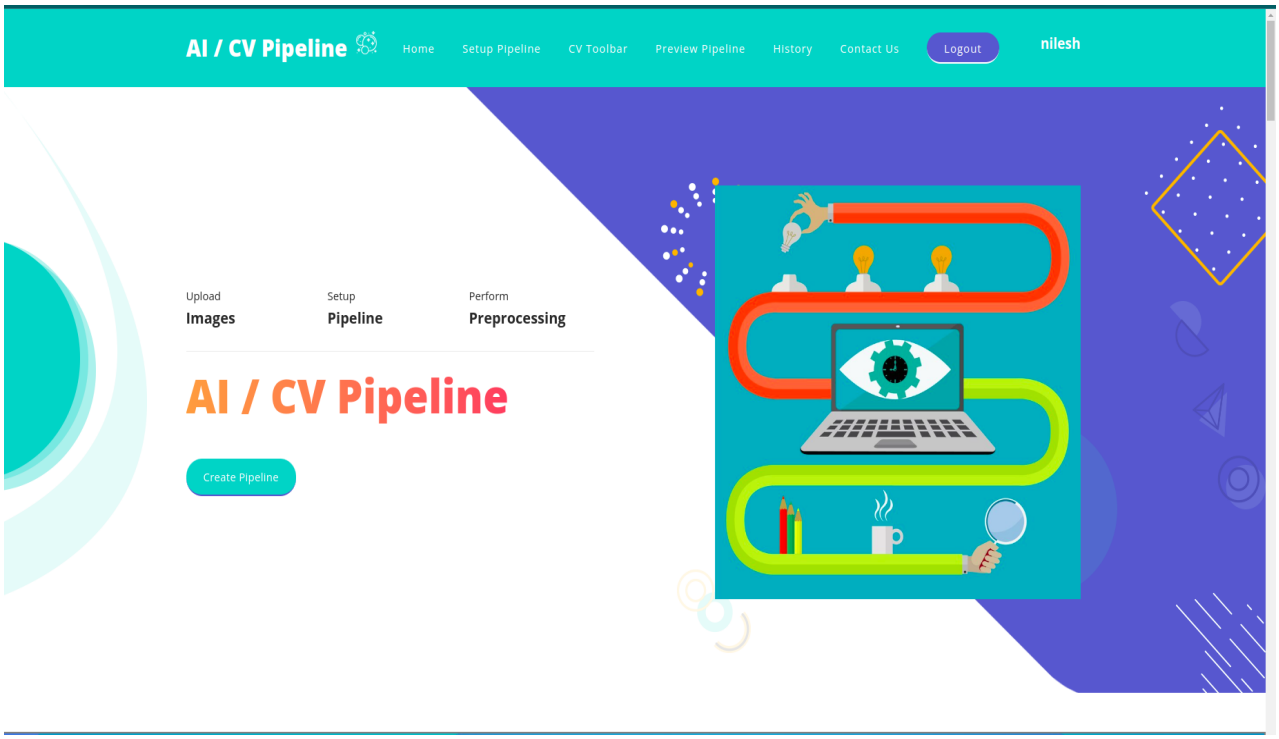

Figure 7: Home Page

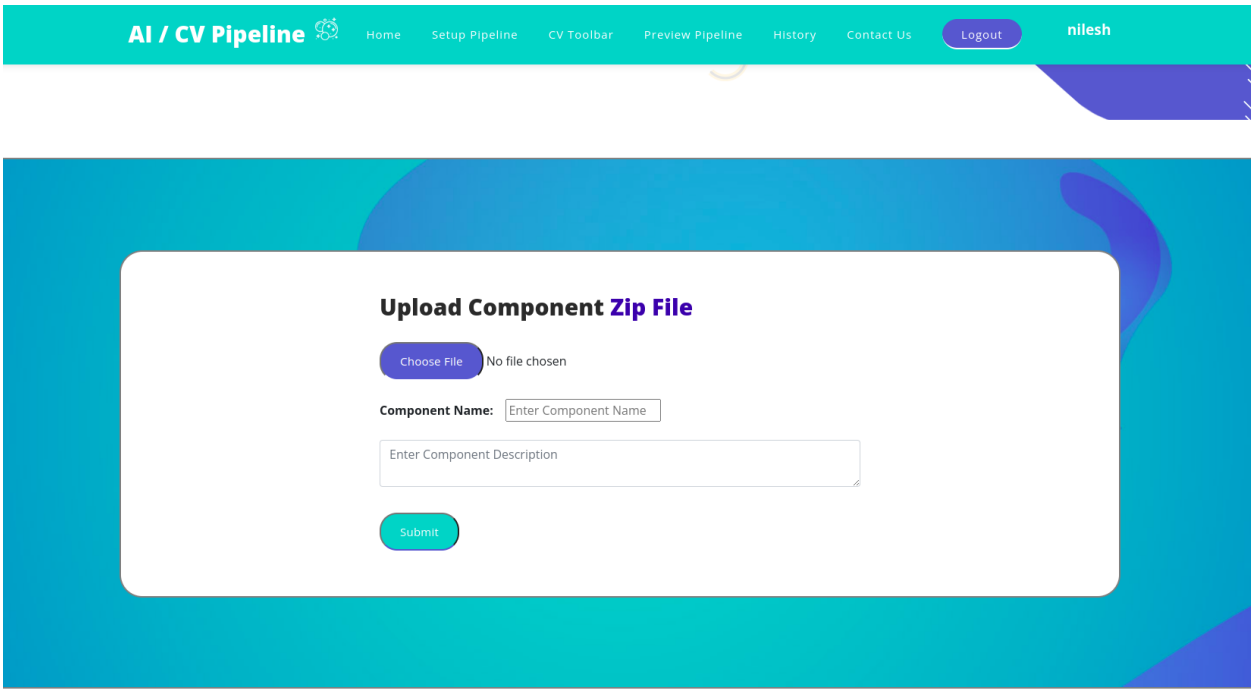

Figure 8: Upload Component Section (only for admin)

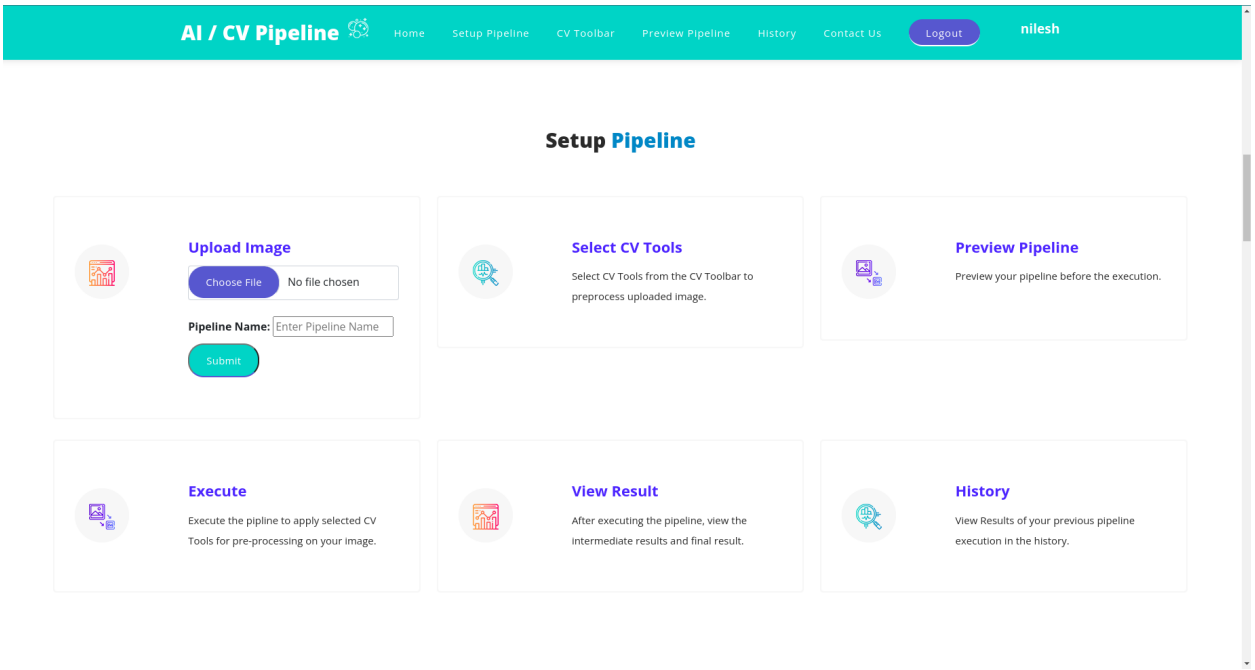

Figure 9: Setup Pipeline Section

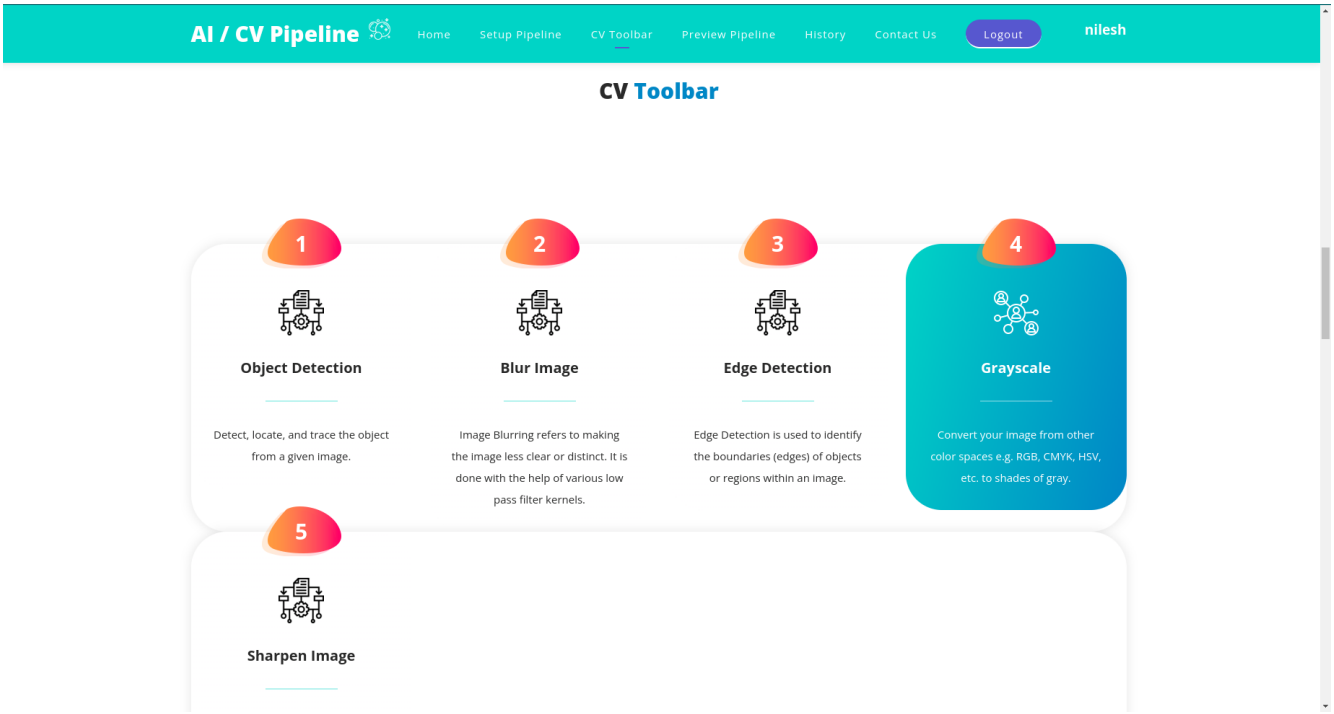

Figure 10: CV Toolbar

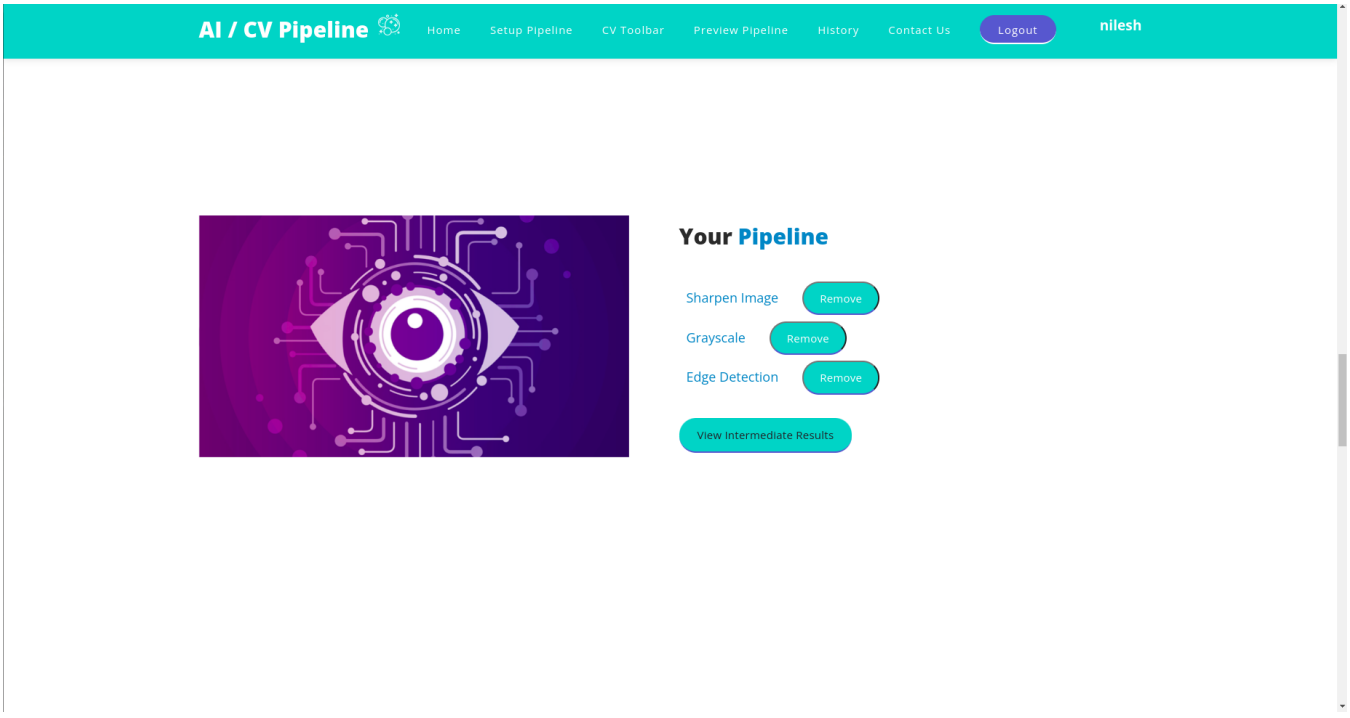

Figure 11: Pipeline Preview

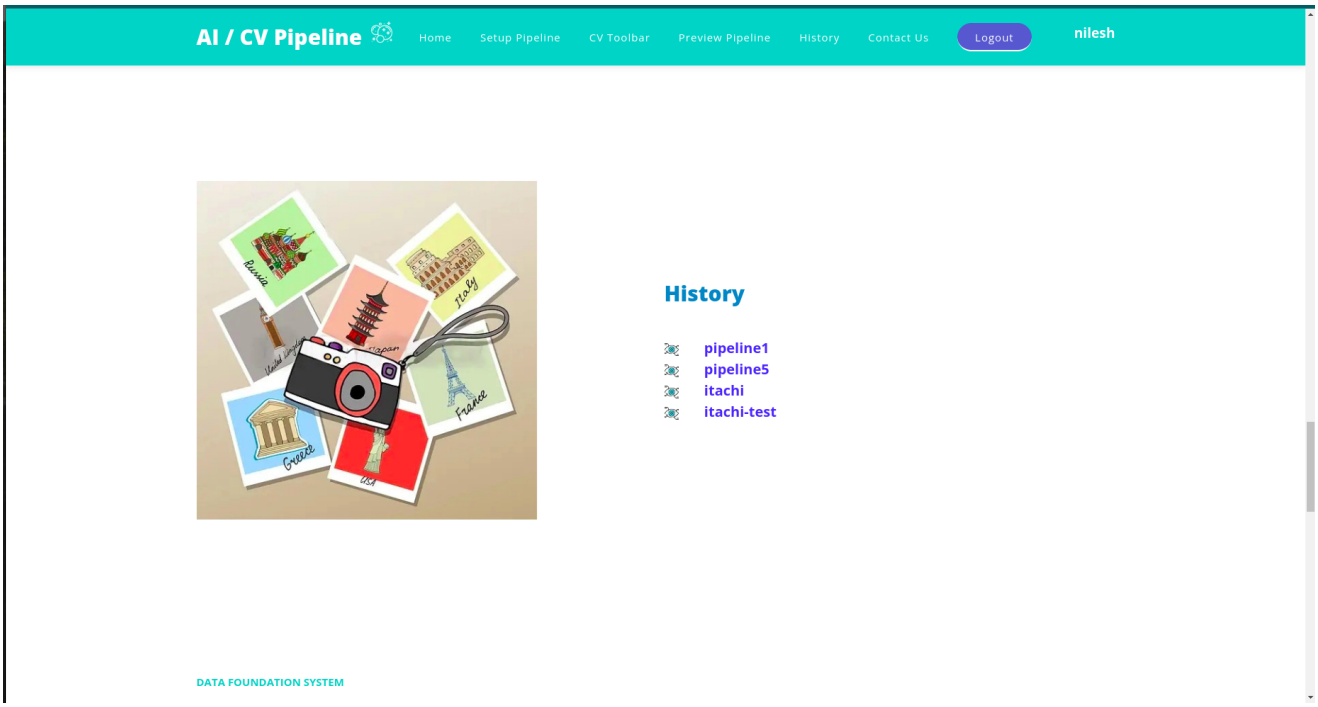

Figure 12: Pipeline History

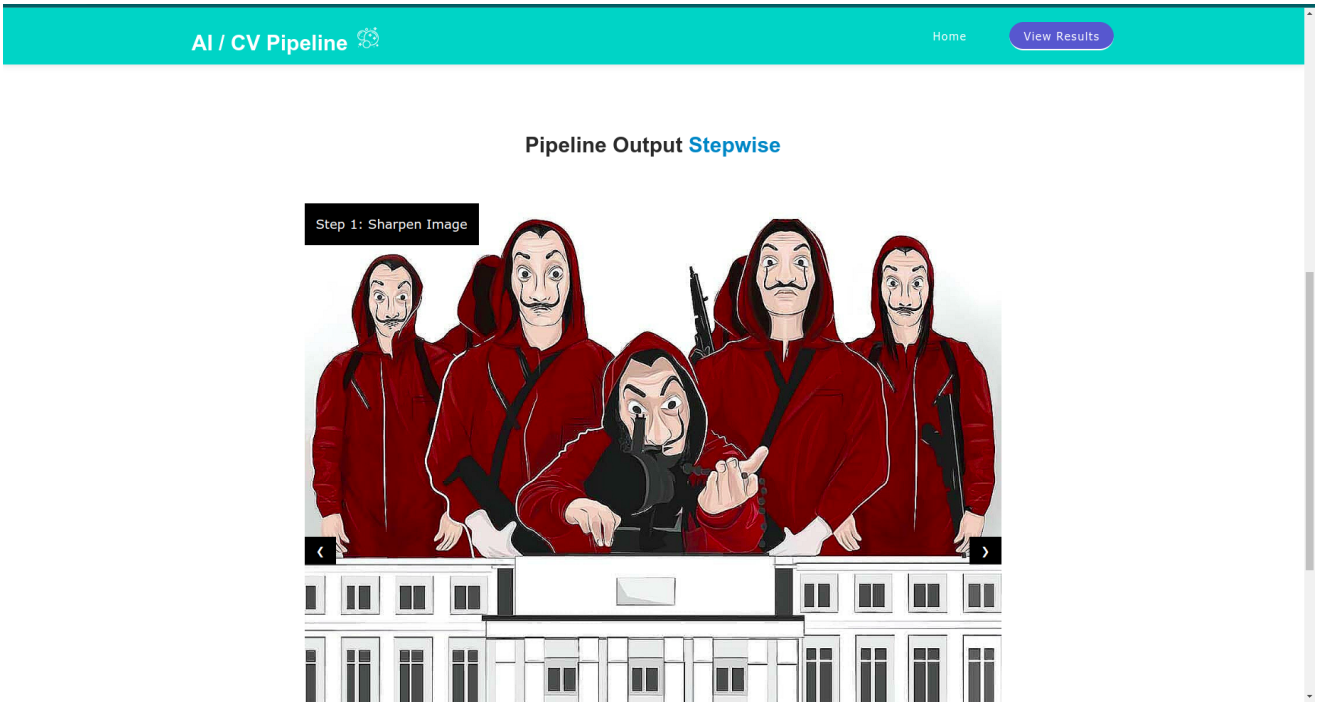

Figure 13: View Pipeline Result