# **Documentation**

## **1. Project Overview**

This project aims to automate the identification of industrial components and process flow arrows in SVG diagrams, converted to PNG format. The goal is to detect the components, determine their dimensions (width and height), and find the start and end points of arrows in process flow diagrams. This is accomplished using **YOLOv11** for component detection and keypoints for identifying the arrow points in the diagrams.

## **2. Project Components**

The project consists of the following components:

* **Data Preprocessing**: Conversion of SVG files to PNG images.
* **Training Dataset Creation**: Split of PNG files into training and validation sets.
* **YOLOv11 for Component Detection**: Used for detecting components in the process flow diagrams.
* **YOLOv11-Pose for Keypoint Detection**: Used to identify start and end points of arrows.
* **Post-Processing**: Extracting detected components' width, height, and arrow start/end points.

## **3. Technical Requirements**

The project requires the following tools and libraries:

* **Python 3.8+**
* **Flask** (for backend API development)
* **YOLOv11** (for object detection)
* **OpenCV** (for image processing)
* **Pillow** (for image handling)
* **NumPy** (for numerical operations)
* **PyTorch** (for deep learning model training)
* **matplotlib** (for visualization)
* **scikit-learn** (for data splitting and model evaluation)
* **TensorFlow** (if needed for other components)

## **4. Functional Requirements**

* **Component Detection**: The system should be able to detect industrial components (e.g., valves, pumps, etc.) in process flow diagrams.
* **Dimension Calculation**: For each detected component, the width and height should be recorded.
* **Arrow Keypoint Detection**: The system should identify the start and end points of arrows in the process flow diagram, marking their coordinates.
* **Accuracy Evaluation**: The detection should be evaluated based on precision, recall, and F1 score.
* **Integration with Backend**: The processed information should be forwarded to the backend for further integration into hospital or industrial systems.

## **5. System Architecture**

The system follows a modular architecture:

1. **Frontend**: (if applicable, for viewing detected components and arrows).
2. **Backend (Flask)**: API endpoints for uploading and processing diagrams.
3. **Detection Model (YOLOv11 + Pose)**: Detects components and keypoints.
4. **Data Management**: Manages and stores component details (width, height) and arrow keypoints.
5. **Post-Processing**: Includes logic to calculate and store component dimensions and arrow coordinates.

## **6. Model Training and Testing**

### **6.1 Data Preprocessing**

* Convert SVG diagrams to PNG images using a custom script.
* Split dataset into training and validation sets (80% training, 20% validation).

### **6.2 YOLOv11 Training**

* Train the **YOLOv11** model for component detection using labeled data (images with bounding boxes around components).
* Use keypoint detection (**YOLOv11-Pose**) for identifying the start and end points of arrows.

### **6.3 Keypoint Detection**

* Utilize **YOLOv11-Pose** to identify keypoints in the process flow diagrams.
* Post-process the keypoint data to extract the coordinates of arrow start and end points.

## **7. Results**

The model should provide accurate detection of components and arrows with the following expected results:

* **Component Detection**: Bounding boxes around each industrial component.
* **Keypoint Detection**: Accurate start and end points for arrows.

## **8. Challenges and Limitations**

* Variations in diagram quality (e.g., resolution, noise) can affect detection accuracy.
* Complex diagrams with overlapping components may reduce detection performance.
* The model might require fine-tuning depending on the quality of labeled training data.

## **9. Future Work**

* Improve model accuracy by gathering more training data and refining model hyperparameters.
* Enhance backend integration to automate the processing of multiple diagrams at once.
* Add support for detecting additional components and different types of arrows.