# Software Requirement Specification

June 19, 2024

| **Project ID & Name** | **Object Detection** | | |
|---|---|---|---|
| Project Manager | Md. Mohoiminul Islam Chowdure | | |
| Customer Request Reference | | | |
| Prepared by | Abu Darda | Date | 27 March, 2024 |
| Reviewed by | | Date | |

## Contents

## 1 Requirement Scope Summary

This project involves creating a Python script that utilizes a pre-trained object detection model to detect and label objects in an image. The script will load a model, process an image to detect objects, draw bounding boxes around the detected objects, and save the processed image to disk. The deliverables include the Python script and the processed image.

## 2 Functional Requirements Identification

1. Install necessary libraries.
2. Load a pre-trained object detection model.
3. Load and display an image.
4. Detect objects in the image using the loaded model.
5. Draw bounding boxes around detected objects with labels.
6. Save the processed image to disk.

## 3 Functional Requirements Details

### 3.1 FR1: Install necessary libraries

- The script must install or verify the installation of required libraries such as OpenCV, TensorFlow, PyTorch, or other dependencies needed for the chosen model.

### 3.2 FR2: Load a pre-trained object detection model

- The script must load a pre-trained object detection model, such as YOLO, SSD, or Faster R-CNN.
- The model chosen should be efficient and suitable for the system's resource constraints.

### 3.3 FR3: Load and display an image

- The script must load an image from a specified path.
- The script should have the capability to display the image using a library such as OpenCV or Matplotlib.

### 3.4 FR4: Detect objects in the image

- The script must use the loaded model to detect objects in the loaded image.
- The output should include the coordinates of bounding boxes and class labels for detected objects.

### 3.5 FR5: Draw bounding boxes around detected objects

- The script must draw bounding boxes around each detected object in the image.
- Each bounding box should be labeled with the corresponding object class.

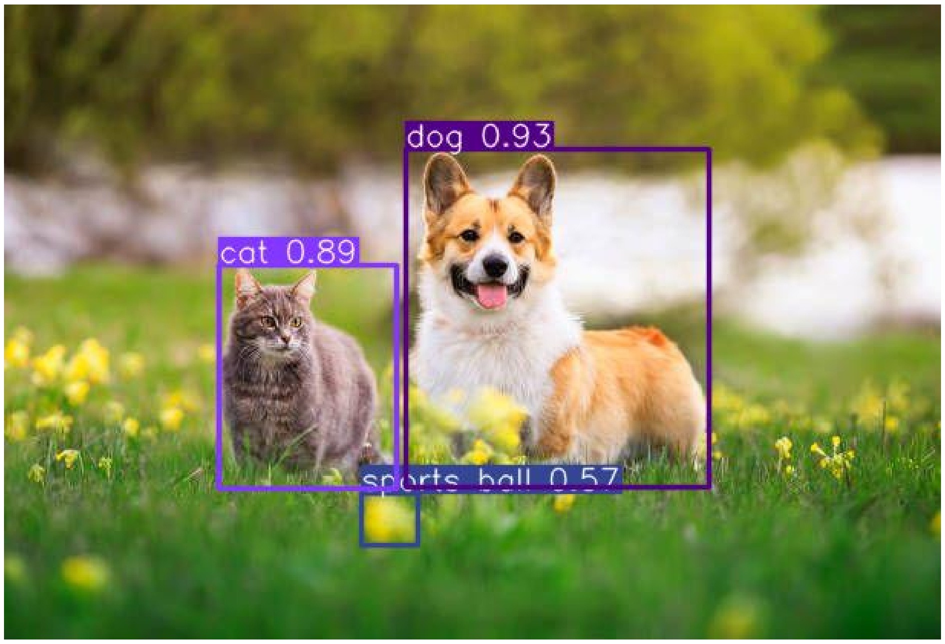

Figure 1: Objects Detected

### 3.6 FR6: Save the processed image to disk

- The script must save the processed image with drawn bounding boxes to a directory named `cv_advanced_output`.
- The filename should be descriptive or include a timestamp to prevent overwriting.

## 4 Deliverables

1. A Python script named `object_detection.py` implementing the above functionalities.
2. A main section in the script that processes an image and saves the result.
3. The processed image saved in the `cv_advanced_output` directory.