Project Report: Object Detection using IBM Visual Recognition

Project Overview

The objective of this project is to develop an object detection system using IBM Visual Recognition. This report outlines the planning and initial setup phase.

I. Project Scope and Objectives

Project Goal

To create a robust object detection system capable of identifying and localizing objects in images.

Target Platform

Utilizing IBM Visual Recognition service for image analysis and object detection.

II. Planning and Requirements

Understanding the Problem

Researched and defined the requirements for the object detection system, including supported object classes, accuracy expectations, and desired features.

Setting Milestones

Established project milestones to track progress and ensure timely completion.

III. Tools and Technologies

IBM Visual Recognition

Chosen as the primary tool for object detection due to its powerful pre-trained models and ease of integration.

Python and Flask

Selected as the programming language and web framework for building the application.

Phase 2: System Architecture and Design

I. System Architecture

High-Level Overview

Designed a client-server architecture where the client interacts with a web interface for image upload, and the server handles image processing and interaction with IBM Visual Recognition.

Components

Client: Web interface for uploading images.

Server: Python-based backend using Flask for routing and handling requests.

II. User Interface Design

Web Interface Wireframe

Created a basic wireframe for the web interface using HTML/CSS, outlining the image upload form.

<!DOCTYPE html>

<html>

<head>

<title>Object Detection</title>

</head>

<body>

<h1>Object Detection System</h1>

<form action="/upload" method="post" enctype="multipart/form-data">

<input type="file" name="image" accept="image/\*">

<input type="submit" value="Upload Image">

</form>

</body>

</html>

III. IBM Visual Recognition Setup

IBM Cloud Account Creation

Created a new IBM Cloud account to gain access to services.

Setting Up Visual Recognition Service

Created a new Visual Recognition service in IBM Cloud Dashboard.

Obtained API keys for authentication.

Phase 3: Development Part 1

I. Setting Up Flask Application

Initializing Flask App

Created a new Flask application for handling web requests.

from flask import Flask, render\_template, request

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

return render\_template('index.html')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

Creating Image Upload Route

Implemented a route for handling image uploads.

@app.route('/upload', methods=['POST'])

def upload\_file():

uploaded\_file = request.files['image']

uploaded\_file.save('uploads/' + uploaded\_file.filename)

return 'File uploaded successfully!'

II. Uploading Images

Web Interface for Image Upload

Designed a web page allowing users to upload images.

Used the HTML code provided earlier.

Handling Image Uploads

Developed code to receive and store uploaded images on the server.

III. Integration with IBM Visual Recognition

Sending Images for Processing

Integrated the Flask application with IBM Visual Recognition service to send uploaded images for object detection.

IV. Viewing Detection Results

Displaying Object Detection Results

Implemented code to render detection results on the web interface.