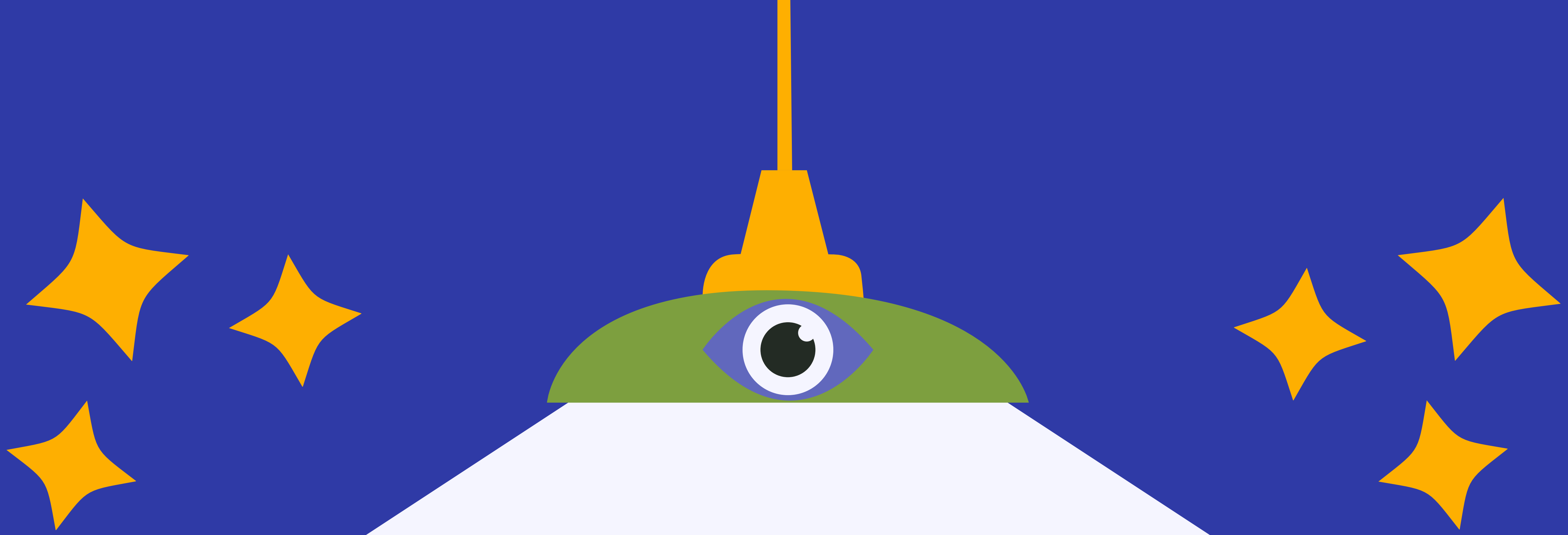
Link

YouTube Link:

https://youtu.be/P6dxnbqJPyA

Flowchart Link :

https://drive.google.com/file/d/1L2EBjenc9-rJo8wygOPm5BHVJjP3nBrR/view?usp=sharing



Computer Vision Project

Team Member

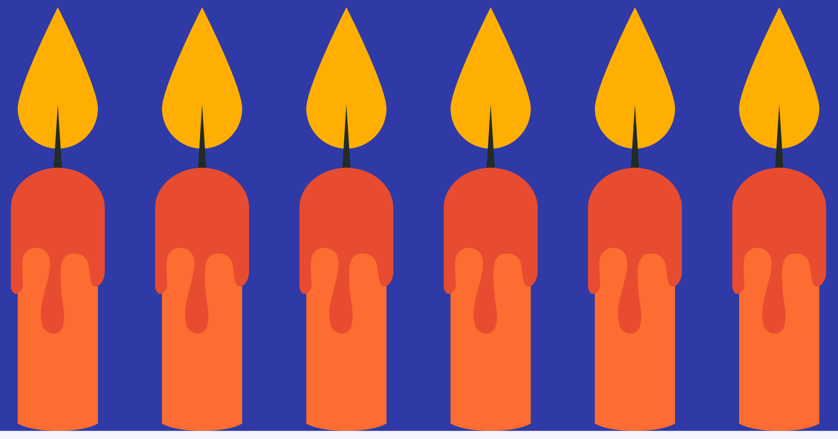
Akesit Akkharasasiri 65110131

Kunlanith Busabong 65110141

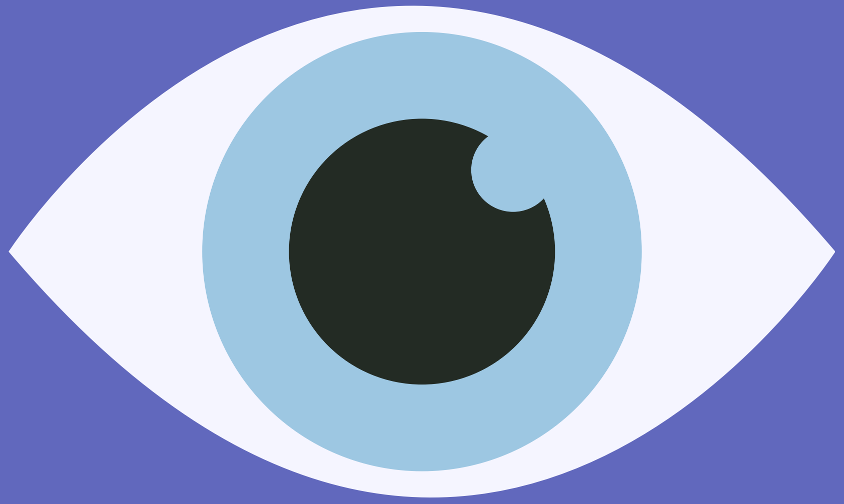
Paveetida Tiranatwittayakul 65110145

Rattapol Kitirak 65110149

Table of Content

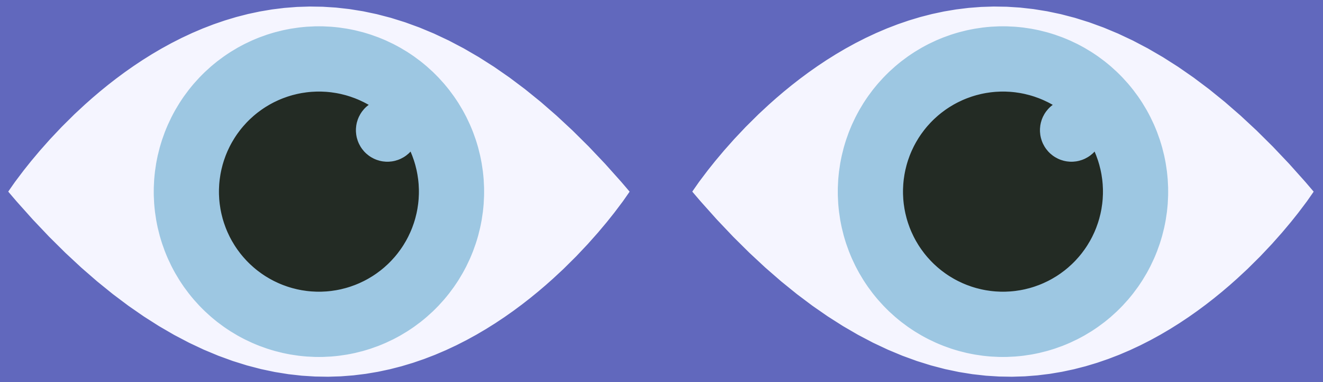


Why we choose this project?



Our project focuses on using computer vision to revolutionize package address sorting in response to the increasing demands of the e-commerce industry. By automating the interpretation of package labels, we aim to streamline sorting processes, minimize errors, and contribute to the trend of smart logistics. This project aligns with the evolving needs of logistics and courier services, ensuring timely and accurate deliveries while optimizing operational workflows.

Why we choose this process?



Objective Enhancement

Real-Time Processing

Compliance and Security

Technical Deep Dive

CV2 for Object Detection

Pytesseract for Enhanced OCR

Pandas for Efficient Data Management

Process Flow Expansion

Dynamic Image Adjustment

Size Analysis and Comparison

Item and Zip Code Validation

Automated Sorting Logic

System Outcomes and Benefits

Enhanced Decision Accuracy

Operational Efficiency

Data-Driven Insights

Scalable and Modular Design

Innovative Use Cases

Customs and Border Protection

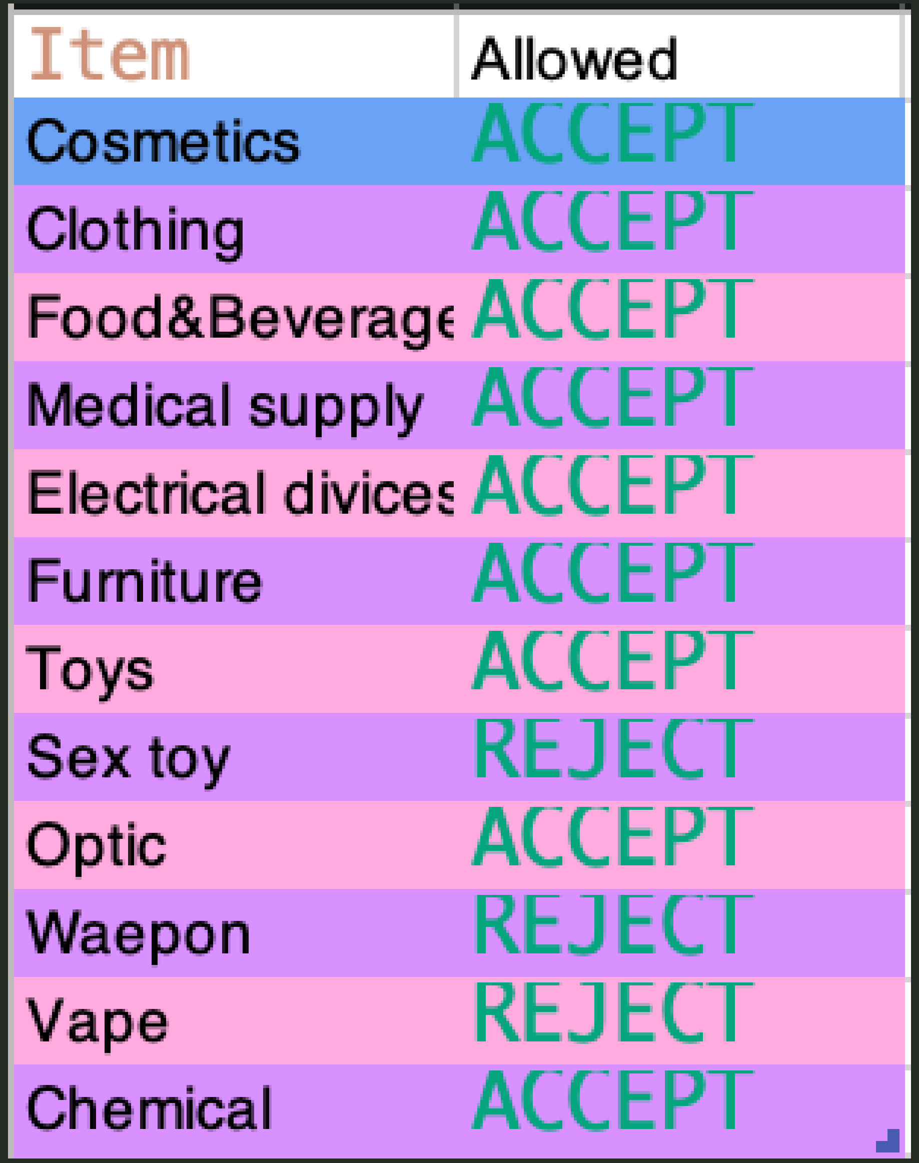
Warehouse Management

Warehouse Management

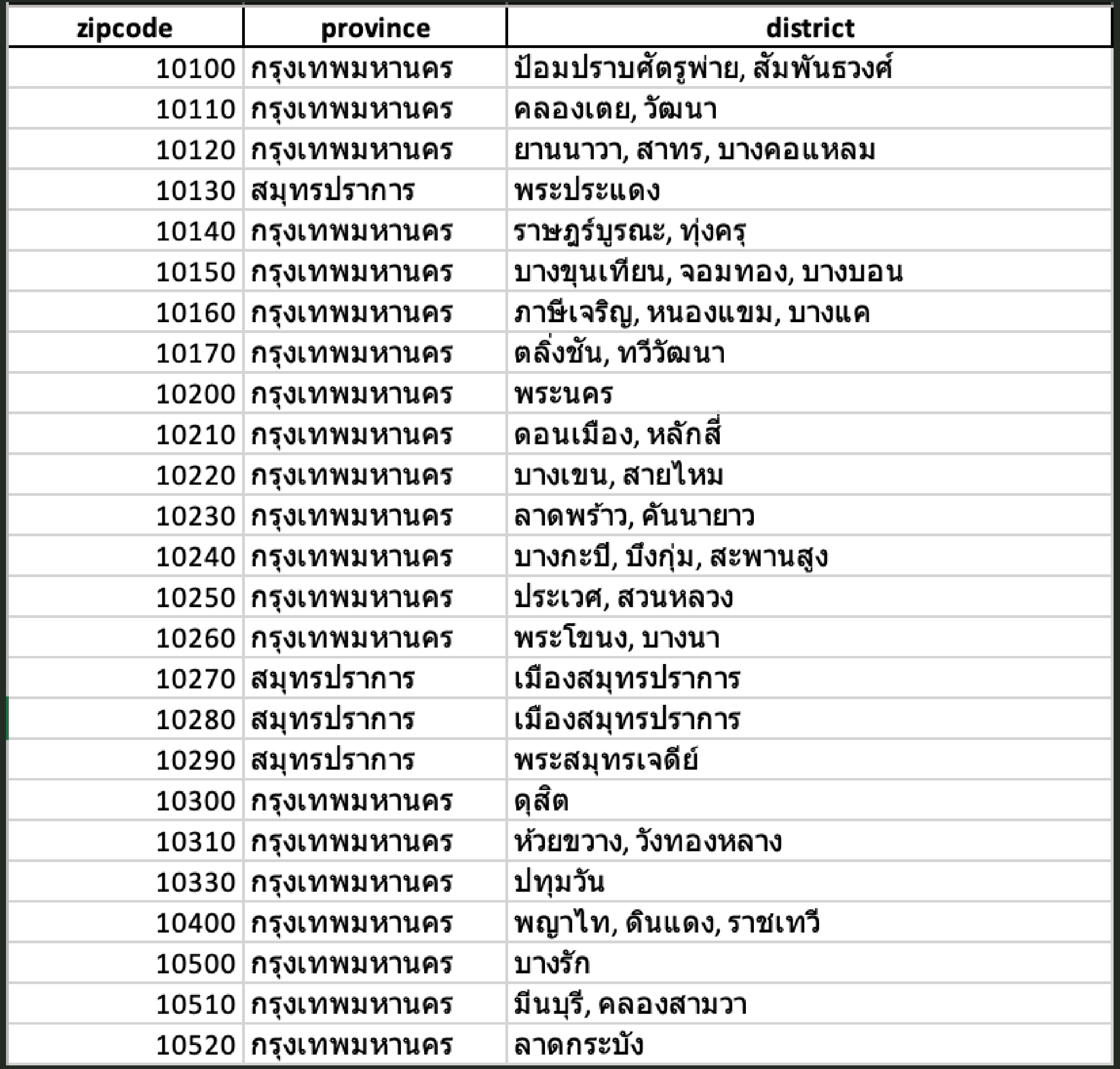
Flow Chart

https://drive.google.com/file/d/1YG1yhXGP2dJqlWmsKvJ3G0gSO5Z95xEq/view?usp=drive_link

Dataset



Item	Allowed
Cosmetics	ACCEPT
Clothing	ACCEPT
Food&Beverage	ACCEPT
Medical supply	ACCEPT
Electrical divices	ACCEPT
Furniture	ACCEPT
Toys	ACCEPT
Sex toy	REJECT
Optic	ACCEPT
Waepon	REJECT
Vape	REJECT
Chemical	ACCEPT



zipcode	province	district
10100	กรุงเทพมหานคร	ป้อมปราบศัตรูพ่าย, สัมพันธวงศ์
10110	กรุงเทพมหานคร	คลองเตย, วัฒนา
10120	กรุงเทพมหานคร	ยานนาวา, สาทร, บางคอแหลม
10130	สมุทรปราการ	พระประแดง
10140	กรุงเทพมหานคร	ราษฎร์บูรณะ, ทุ่งครุ
10150	กรุงเทพมหานคร	บางขุนเทียน, จอมทอง, บางบอน
10160	กรุงเทพมหานคร	ภาษีเจริญ, หนองแขม, บางแค
10170	กรุงเทพมหานคร	ตลิ่งชัน, ทวีวัฒนา
10200	กรุงเทพมหานคร	พระนคร
10210	กรุงเทพมหานคร	ดอนเมือง, หลักสี่
10220	กรุงเทพมหานคร	บางเขน, สายไหม
10230	กรุงเทพมหานคร	ลาดพร้าว, คันนายาว
10240	กรุงเทพมหานคร	บางกะปิ, บึงกุ่ม, สะพานสูง
10250	กรุงเทพมหานคร	ประเวศ, สวนหลวง
10260	กรุงเทพมหานคร	พระโขนง, บางนา
10270	สมุทรปราการ	เมืองสมุทรปราการ
10280	สมุทรปราการ	เมืองสมุทรปราการ
10290	สมุทรปราการ	พระสมุทรเจดีย์
10300	กรุงเทพมหานคร	ดุสิต
10310	กรุงเทพมหานคร	ห้วยขวาง, วังทองหลาง
10330	กรุงเทพมหานคร	ปทุมวัน
10400	กรุงเทพมหานคร	พญาไท, ดินแดง, ราชเทวี
10500	กรุงเทพมหานคร	บางรัก
10510	กรุงเทพมหานคร	มีนบุรี, คลองสามวา
10520	กรุงเทพมหานคร	ลาดกระบัง

Coding

✨ cv2

Purpose: OpenCV (Open Source Computer Vision Library) is used for image processing, video capture and processing, and object detection. In the code, it's used for capturing video frames, converting color spaces, detecting contours, cropping images, and displaying the results.

✨ numpy

Purpose: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. In the code, it's used for creating arrays to define color ranges for mask operations and other array manipulations.

✨ Regular Expressions (re)

Purpose: The re module offers a set of functions that allows us to search a string for a match. In the script, it's used to detect zip codes in the text extracted from images.

✨os

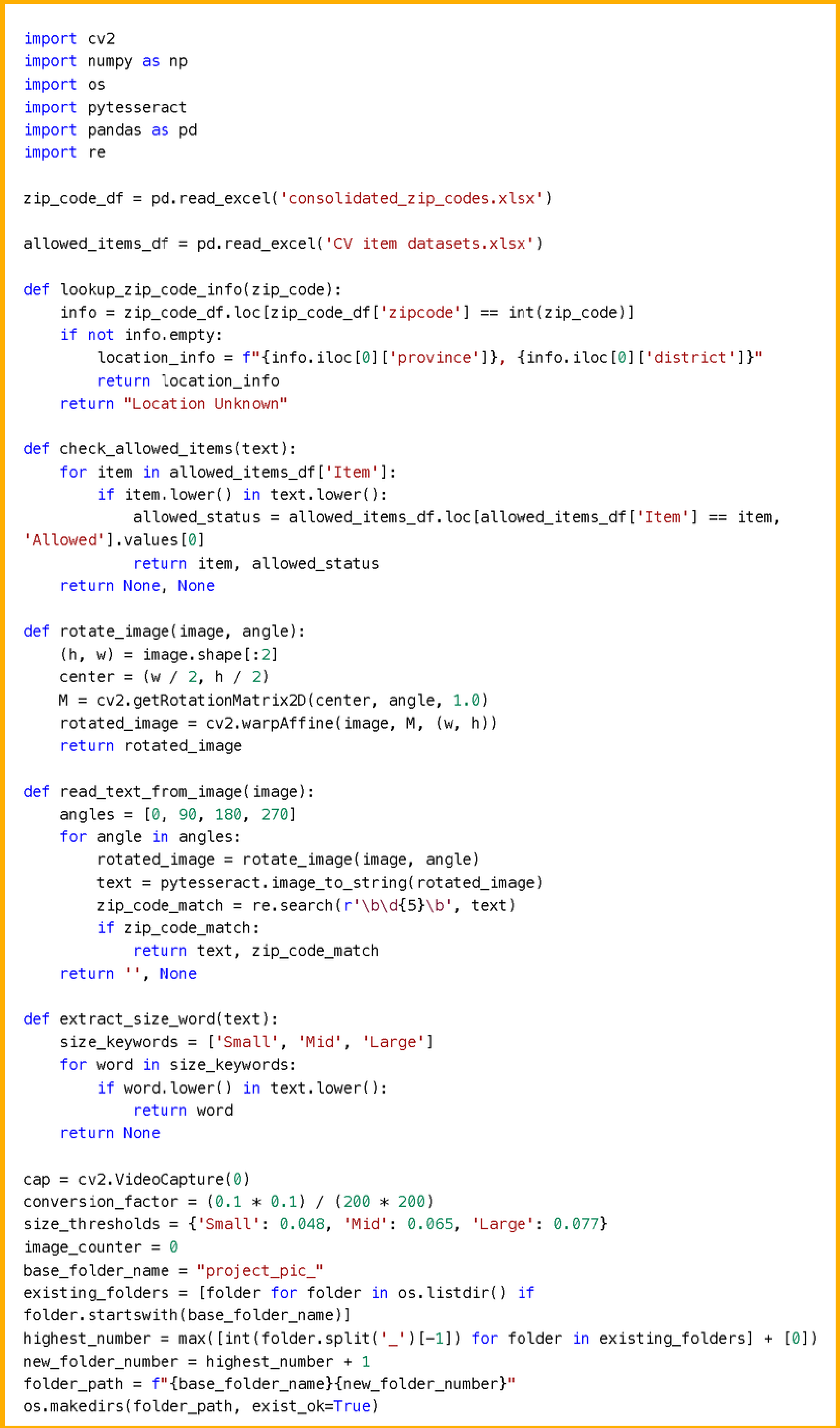
Purpose: The os module in Python provides a way of using operating system dependent functionality. In the code, it's used to handle directory paths, check for existing folders, and create new folders for saving images.

✨ pytesseract

Purpose: Pytesseract is a wrapper for Google's Tesseract-OCR Engine. It can read and recognize text in images. In the script, it's used to extract text from images for further processing, such as detecting zip codes and item names.

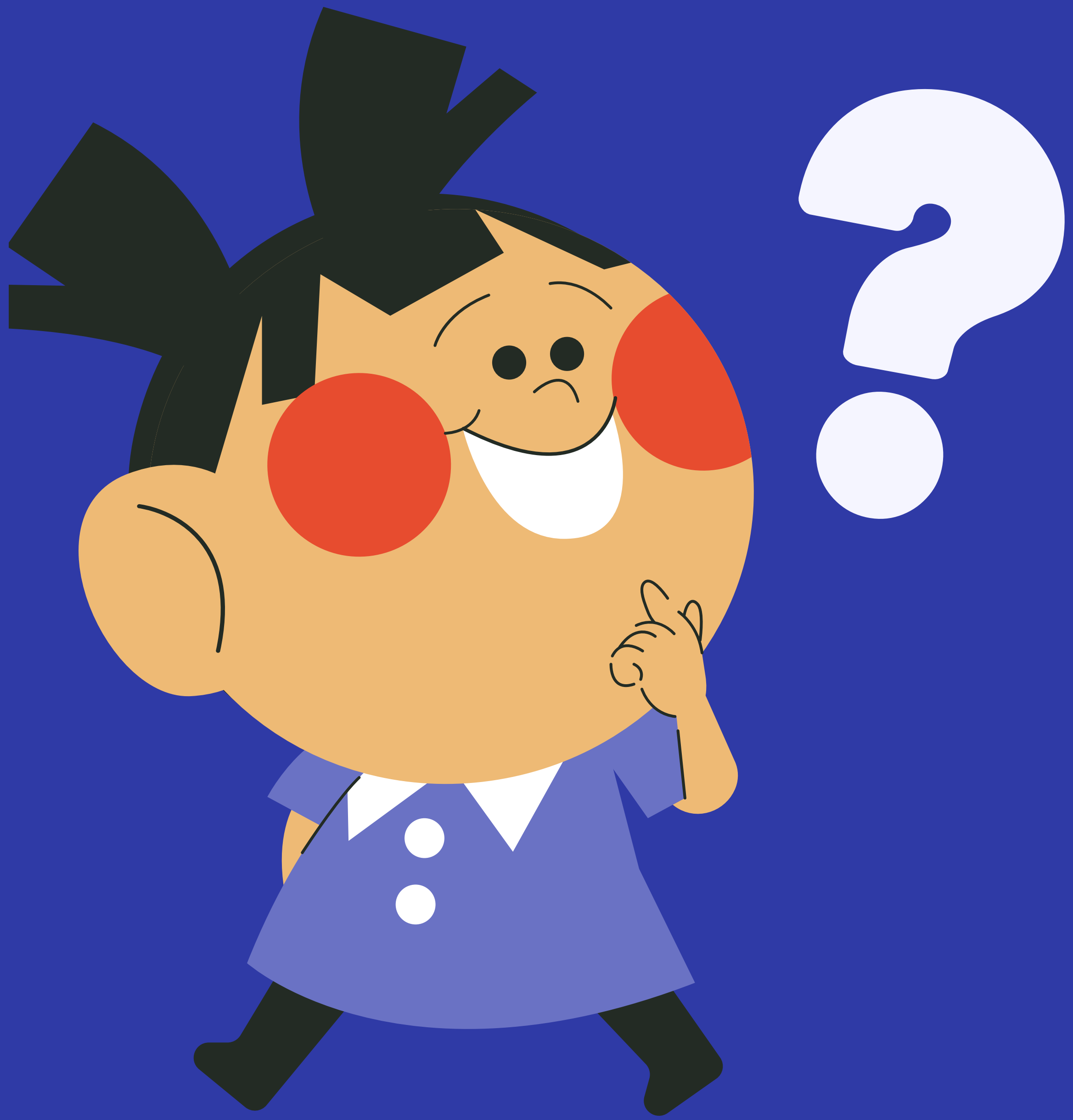
✨ pandas

Purpose: Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. In the code, it's used to read Excel files containing zip codes and item lists, and to search for specific information within these datasets.



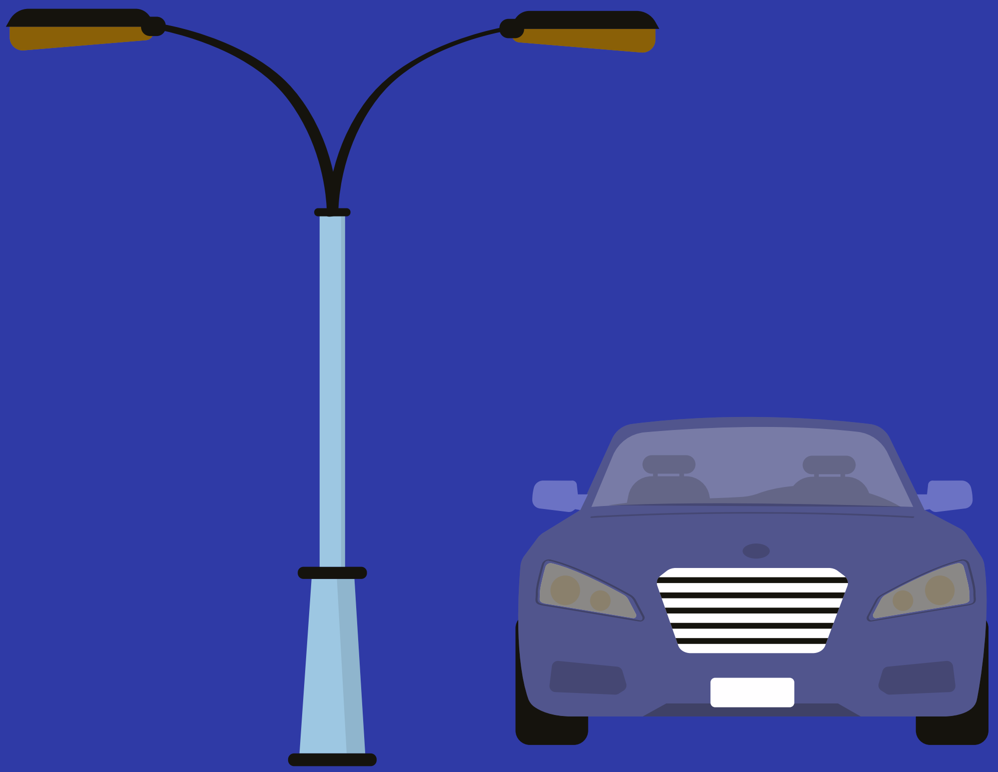
```python
import cv2
import numpy as np
import os
import pytesseract
import pandas as pd
import re

zip_code_df = pd.read_excel('consolidated_zip_codes.xlsx')

allowed_items_df = pd.read_excel('CV item datasets.xlsx')

def lookup_zip_code_info(zip_code):
    info = zip_code_df.loc[zip_code_df['zipcode'] == int(zip_code)]
    if not info.empty:
        location_info = f"{info.iloc[0]['province']}, {info.iloc[0]['district']}"
        return location_info
    return "Location Unknown"

def check_allowed_items(text):
    for item in allowed_items_df['Item']:
        if item.lower() in text.lower():
            allowed_status = allowed_items_df.loc[allowed_items_df['Item'] == item,
'Allowed'].values[0]
            return item, allowed_status
    return None, None

def rotate_image(image, angle):
    (h, w) = image.shape[:2]
    center = (w / 2, h / 2)
    M = cv2.getRotationMatrix2D(center, angle, 1.0)
    rotated_image = cv2.warpAffine(image, M, (w, h))
    return rotated_image

def read_text_from_image(image):
    angles = [0, 90, 180, 270]
    for angle in angles:
        rotated_image = rotate_image(image, angle)
        text = pytesseract.image_to_string(rotated_image)
        zip_code_match = re.search(r'\b\d{5}\b', text)
        if zip_code_match:
            return text, zip_code_match
    return '', None

def extract_size_word(text):
    size_keywords = ['Small', 'Mid', 'Large']
    for word in size_keywords:
        if word.lower() in text.lower():
            return word
    return None

cap = cv2.VideoCapture(0)
conversion_factor = (0.1 * 0.1) / (200 * 200)
size_thresholds = {'Small': 0.048, 'Mid': 0.065, 'Large': 0.077}
image_counter = 0
base_folder_name = "project_pic_"
existing_folders = [folder for folder in os.listdir() if
folder.startswith(base_folder_name)]
highest_number = max([int(folder.split('_')[-1]) for folder in existing_folders] + [0])
new_folder_number = highest_number + 1
folder_path = f"{base_folder_name}{new_folder_number}"
os.makedirs(folder_path, exist_ok=True)
```

```python
while True:
    ret, frame = cap.read()
    if ret:
        hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
        lower_color = np.array([10, 100, 20])
        upper_color = np.array([20, 255, 200])
        color_mask = cv2.inRange(hsv_frame, lower_color, upper_color)
        contours, _ = cv2.findContours(color_mask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        if contours:
            largest_contour = max(contours, key=cv2.contourArea)
            if cv2.contourArea(largest_contour) > 100:
                x, y, w, h = cv2.boundingRect(largest_contour)
                area_pixels = cv2.contourArea(largest_contour)
                area_meters = area_pixels * conversion_factor
                size_name = "Unknown"
                for size_category, threshold in size_thresholds.items():
                    if area_meters < threshold:
                        size_name = size_category
                        break
                if size_name == "Unknown":
                    size_name = "Large"
                cropped_frame = frame[y:y+h, x:x+w].copy()
                text, zip_code_match = read_text_from_image(cropped_frame)
                detected_size_word = extract_size_word(text)
                item, allowed_status = check_allowed_items(text)
                if zip_code_match:
                    zip_code = zip_code_match.group()
                    location = lookup_zip_code_info(zip_code)
                else:
                    zip_code = "N/A"
                    location = "Location Unknown"
                size_match = detected_size_word == size_name if detected_size_word else
"N/A"
                cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
                for express_name in ["SLOTH EXPRESS", "SNAIL EXPRESS", "DOG EXPRESS"]:
                    if express_name in text:
                        display_text = f"Express name: {express_name}, Size: {size_name},
Detected Size: {detected_size_word}, Size Match: {size_match}, Area: {area_meters:.3f}m^2,
Zip Code: {zip_code}, Location: {location}, Item: {item if item else ''}, Status:
{allowed_status if allowed_status else ''}"
                        cv2.putText(frame, display_text, (x, y-10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2)
                        print(display_text)
                        filename_suffix = f"_{zip_code}" if zip_code_match else ""
                        image_filename = os.path.join(folder_path, f'{express_name.replace("
", "_").lower()}{filename_suffix}_{image_counter}.jpg')
                        cv2.imwrite(image_filename, cropped_frame)
                        image_counter += 1
                        break
                cv2.imshow('Frame', frame)
                if cv2.waitKey(1) & 0xFF == ord('q'):
                    break
cap.release()
cv2.destroyAllWindows()
```



Result

Propose Idea Improvement



Integration of Machine Learning (ML) and Artificial Intelligence (AI)
- Advanced Object Recognition
- Natural Language Processing (NLP)

Enhanced Imaging Technologies
- 3D Scanning and Modeling
- High-Resolution Imaging

Internet of Things (IoT) Integration
- Real-Time Data Collection and Analysis

Adaptive and Predictive Algorithms
- Predictive Sorting
- Adaptive Processing Paths

Thank You