```
!pip install torch torchvision torchaudio transformers pillow

from transformers import CLIPProcessor, CLIPModel
from PIL import Image
from google.colab import files
import torch

# ============================================
# 🧠 Define the Agent Class
# ============================================

class VisionAgent:
    def __init__(self, model_name="openai/clip-vit-base-patch32"):
        print("🧩 Initializing VisionAgent with CLIP model...")
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        self.model = CLIPModel.from_pretrained(model_name).to(self.device)
        self.processor = CLIPProcessor.from_pretrained(model_name)

        # You can easily extend this list
        self.knowledge_base = self.knowledge_base = [
    # Large house items & furniture
    "sofa", "sectional", "couch", "recliner", "loveseat", "armchair", "bean bag",
    "rocking chair", "bench", "coffee table", "side table", "end table", "console table",
    "bookshelf", "TV", "TV stand", "home theater", "soundbar", "entertainment unit",
    "dining table", "dining chair", "bar stool", "high chair", "breakfast table",
    "bed", "single bed", "queen bed", "king bed", "bunk bed", "daybed", "trundle bed",
    "crib", "baby cot", "headboard", "footboard", "nightstand", "vanity", "desk",
    "study table", "computer table", "workstation", "dresser", "wardrobe", "closet", "cabinet",
    "locker", "safe", "chest", "drawer", "shelf", "bookcase", "filing cabinet",
    "mirror (full)", "mirror (wall)", "sliding door", "door", "window", "staircase", "railing",
    "curtain", "blind", "shutter", "mat", "rug", "carpet", "floor tiles", "doorstop",
    # Large appliances
    "refrigerator", "freezer", "mini fridge", "oven", "microwave", "stove", "cooktop",
    "dishwasher", "washing machine", "clothes dryer", "air conditioner", "heater",
    "water heater", "geyser", "fan", "exhaust fan", "dehumidifier", "humidifier", "air purifier",
    "vacuum cleaner", "steam cleaner", "robot vacuum", "iron", "ironing board",
    # Medium appliances and utilities
    "blender", "mixer", "food processor", "toaster", "coffee maker", "espresso machine",
    "rice cooker", "pressure cooker", "kettle", "juicer", "grill", "slow cooker",
    "induction cooktop", "hot plate", "electric skillet", "water purifier",
    # Kitchenware & utensils (from big to tiny)
    "pot", "pan", "casserole", "wok", "baking tray", "baking sheet", "griddle", "roasting pan",
    "colander", "strainer", "salad spinner", "cutting board", "chopping block", "rolling pin",
    "mortar", "pestle", "measuring cup", "measuring spoon", "mixing bowl", "bowl", "plate",
    "dish", "serving tray", "serving spoon", "ladle", "spatula", "whisk", "tongs", "fork",
    "knife", "teaspoon", "tablespoon", "chopsticks", "peeler", "grater", "slicer", "can opener",
    "bottle opener", "corkscrew", "jar opener", "ice tray", "egg cup", "egg slicer", "apple corer",
    "pizza cutter", "nutcracker", "garlic press", "tea infuser", "tea strainer",
    "butter dish", "bread box", "salt shaker", "pepper grinder", "condiment bottle", "sugar bowl",
    "oil dispenser", "spice rack", "spice jar", "herb container", "tin", "plastic container",
    "cling film", "foil", "zipper bag", "silicone mat", "napkin holder", "placemat", "coaster",
    # Tableware & drinkware
    "cup", "mug", "teacup", "saucer", "glass", "wine glass", "champagne flute", "water bottle",
    "thermos", "jug", "pitcher", "decanter", "shot glass", "beer stein",
    # Cleaning & maintenance (major to micro)
    "broom", "mop", "bucket", "dustpan", "scrub brush", "toilet brush", "cleaning cloth",
    "rag", "towel", "duster", "sponges", "dish scrubber", "scouring pad", "gloves",
    "wipes", "spray bottle", "detergent", "soap bar", "liquid soap", "hand sanitizer",
    "bleach", "air freshener", "disinfectant", "cleaner", "toilet cleaner", "drain opener",
    "pest spray", "pesticide", "herbicide", "mosquito coil", "mosquito net",
    "fly swatter", "mouse trap",
    # Bathroom & personal care
    "shower curtain", "soap dish", "toothbrush", "toothpaste", "mouthwash", "floss",
    "razor", "shaving foam", "shaving kit", "hairbrush", "comb", "hair dryer", "straightener",
    "makeup kit", "perfume", "deodorant", "shampoo", "conditioner", "bath mat", "bath towel",
    "face towel", "hand towel", "loofah", "bath sponge", "nail clipper", "tweezers",
    "earbuds", "cotton balls", "facial tissue", "tissue box", "plasters", "bandage", "first aid kit",
    "thermometer",
    # Bedroom & clothing (all sizes/types)
    "clothes", "shirt","t-shirt", "jeans", "trousers", "shorts", "skirt", "dress", "gown", "nightwear",
    "undergarments", "vest", "socks", "stockings", "hat", "cap", "belt", "tie", "scarf", "gloves",
    "coat", "jacket", "sweater", "blazer", "raincoat", "shoes", "slippers", "sandals", "boots",
    "flip-flops",
    "wardrobe", "closet", "hanger", "drawer organizer", "shoe rack", "shoe organizer",
    "laundry basket", "clothespin", "clothesline", "ironing board", "stepladder",
```

```
        # Decor & small accessories
        "candle", "incense", "vase", "flower", "plant", "artificial plant", "bonsai", "terrarium",
        "planter", "pot", "picture frame", "painting", "poster", "statue", "figurine", "magnet",
        "snow globe", "wind chime", "clock", "wall clock", "alarm clock", "table clock",
        "calendar", "diary", "notebook", "journal", "pen", "pencil", "marker", "highlighter",
        # Office & tech (big to tiny)
        "laptop", "desktop computer", "monitor", "keyboard", "mouse", "printer", "scanner",
        "router", "modem", "wifi extender", "flash drive", "SSD", "hard drive", "memory card",
        "card reader", "battery", "power bank", "charger", "extension cord", "power strip",
        "adapter", "earphones", "headphones", "speaker", "camera", "tripod", "webcam",
        "microphone", "joystick", "video game console", "controller", "remote", "SIM card",
        "stylus", "phone", "tablet", "case", "screen protector", "charger cable",
        # Stationery & micro objects
        "stapler", "staple pin", "pin", "thumbtack", "pushpin", "paper clip", "binder clip",
        "rubber band", "button", "needle", "thread", "string", "lace", "cord", "shoelace",
        "zipper", "snap", "clip", "tag", "label", "badge", "ID card", "safety pin", "hairpin",
        # Workshop & utility (tiny to large)
        "toolbox", "screwdriver", "hammer", "wrench", "spanner", "pliers", "tape measure",
        "level", "saw", "chisel", "file", "drill", "drill bit", "nut", "bolt", "screw", "nail",
        "hook", "bracket", "plate", "hinge", "spring", "washer", "grommet", "plug", "socket",
        "wire", "cable", "fuse", "switch", "bulb", "tube light", "LED", "night lamp", "shade",
        # Kitchen: every micro item
        "spice jar", "herb jar", "tea bag", "coffee pod", "bottle cap", "straw", "toothpick",
        "cocktail stick", "pick", "lemon squeezer", "muddler", "rim salt", "sachet", "wrapper",
        "packet", "label", "coupon", "recipe card",
        # Living room/balcony/garden
        "garden chair", "garden table", "patio furniture", "swing", "hammock", "umbrella",
        "grill", "BBQ", "fire pit", "watering can", "shovel", "rake", "hoe", "spade",
        # Miscellaneous tiny household items
        "coin", "stamp", "seal", "ring", "necklace", "bracelet", "earring", "anklet", "brooch",
        "cufflink", "tie pin", "lapel pin", "locket", "pendant",
        "key", "keychain", "lock", "padlock", "comb", "brush", "lint roller", "sticky note",
        "envelope", "postcard", "ticket", "token", "chip", "dice", "board game token",
        "playing card", "domino", "puzzle piece", "rubik's cube", "marble", "bead", "button battery",
        # Pets & animals
        "pet bed", "pet collar", "leash", "tag", "dog bowl", "cat bowl", "litter tray",
        "scratching post", "cat tree", "feeder", "waterer", "aquarium", "fish tank", "birdcage",
        "pet toy", "ball", "chew toy", "treat container",
        # Kids, babies, toys
        "baby bottle", "pacifier", "teether", "rattle", "play mat", "playpen", "blocks",
        "building set", "stuffed toy", "teddy bear", "doll", "puppet", "car toy", "train set",
        "board game", "game console", "controller", "puzzle", "coloring book", "crayon", "paint",
        "brush", "finger paint", "sticker", "stamp", "glitter", "googly eyes", "craft supplies",
        # Outdoors/sports
        "bicycle", "helmet", "bike lock", "skateboard", "basketball", "football", "tennis racket",
        "shuttlecock", "badminton racket", "cricket bat", "ball", "glove", "jersey", "sports shoe",
        # Emergency, safety, security
        "fire extinguisher", "smoke detector", "alarm", "flashlight", "emergency light",
        "battery backup", "whistle", "pepper spray", "first aid kit", "bandage", "plaster",
        "splint", "ice pack", "hot pack",
        # Home office: every object
        "pen holder", "desk organizer", "paper tray", "letter opener", "envelope", "stamp",
        "post-it note", "marker", "eraser", "whiteboard", "whiteboard marker", "chalk",
        "chalkboard", "magnet", "pin", "flag", "file folder", "divider", "index tab"
    ]


    def analyze_image(self, image: Image.Image):
        """Analyze an image and detect the most likely object name."""
        prompts = [f"a photo of a {label}" for label in self.knowledge_base]
        inputs = self.processor(text=prompts, images=image, return_tensors="pt", padding=True).to(self.device)

        with torch.no_grad():
            outputs = self.model(**inputs)
            probs = outputs.logits_per_image.softmax(dim=1)

        best_idx = probs.argmax().item()
        confidence = probs[0, best_idx].item() * 100
        detected_object = self.knowledge_base[best_idx]

        reasoning = (
            f"🧠 Reasoning: Based on CLIP's vision-text similarity, "
            f"the uploaded image is most similar to '{detected_object}'.\n"
            f"Confidence: {confidence:.2f}%."
        )

        return {
```

```
            "object_detected": detected_object,
            "confidence": confidence,
            "reasoning": reasoning
        }

# ==========================================
# 🚀 Run the Agent
# ==========================================

print("📤 Please upload an image of a single object (like shoes, bottle, laptop, etc.)")
uploaded = files.upload()

if uploaded:
    image_path = list(uploaded.keys())[0]
    image = Image.open(image_path).convert("RGB")

    # Initialize our VisionAgent
    agent = VisionAgent()

    # Agent processes the image
    result = agent.analyze_image(image)

    print("\n🤖 Agent Report:")
    print(f"Detected Object: {result['object_detected']}")
    print(f"Confidence: {result['confidence']:.2f}%")
    print(result['reasoning'])
else:
    print("❌ No image uploaded.")
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: torchvision in /usr/local/lib/python3.12/dist-packages (0.23.0+cu126)
Requirement already satisfied: torchaudio in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.1)
Requirement already satisfied: pillow in /usr/local/lib/python3.12/dist-packages (11.3.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from torch) (3.20.0)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.7
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.77)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1.2)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4.2)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.85)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from torchvision) (2.0.2)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.36
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22.
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.3)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2.5.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2025
📩 Please upload an image of a single object (like shoes, bottle, laptop, etc.)
Choose files fridge.jpeg
fridge.jpeg(image/jpeg) - 4053 bytes, last modified: 29/10/2025 - 100% done
Saving fridge.jpeg to fridge (2).jpeg
🧩 Initializing VisionAgent with CLIP model...

🤖 Agent Report:
Detected Object: refrigerator
Confidence: 64.56%
🧠 Reasoning: Based on CLIP's vision-text similarity, the uploaded image is most similar to 'refrigerator'.
Confidence: 64.56%.
```

```
!pip install transformers torch torchvision torchaudio pillow

from transformers import CLIPProcessor, CLIPModel
from google.colab import files
from PIL import Image
import torch

# ============================================
# 🧠 Define VisionAgent
# ============================================

class VisionAgent:
    def __init__(self, model_name="openai/clip-vit-base-patch32"):
        print("🧩 Initializing VisionAgent with CLIP model...")
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        self.model = CLIPModel.from_pretrained(model_name).to(self.device)
        self.processor = CLIPProcessor.from_pretrained(model_name)

        # 🎯 Object vocabulary — extendable knowledge base
        self.knowledge_base = self.knowledge_base = [
    # Large house items & furniture
    "sofa", "sectional", "couch", "recliner", "loveseat", "armchair", "bean bag",
    "rocking chair", "bench", "coffee table", "side table", "end table", "console table",
    "bookshelf", "TV", "TV stand", "home theater", "soundbar", "entertainment unit",
    "dining table", "dining chair", "bar stool", "high chair", "breakfast table",
```

```
"bed", "single bed", "queen bed", "king bed", "bunk bed", "daybed", "trundle bed",
"crib", "baby cot", "headboard", "footboard", "nightstand", "vanity", "desk",
"study table", "computer table", "workstation", "dresser", "wardrobe", "closet", "cabinet",
"locker", "safe", "chest", "drawer", "shelf", "bookcase", "filing cabinet",
"mirror (full)", "mirror (wall)", "sliding door", "door", "window", "staircase", "railing",
"curtain", "blind", "shutter", "mat", "rug", "carpet", "floor tiles", "doorstop",
# Large appliances
"refrigerator", "freezer", "mini fridge", "oven", "microwave", "stove", "cooktop",
"dishwasher", "washing machine", "clothes dryer", "air conditioner", "heater",
"water heater", "geyser", "fan", "exhaust fan", "dehumidifier", "humidifier", "air purifier",
"vacuum cleaner", "steam cleaner", "robot vacuum", "iron", "ironing board",
# Medium appliances and utilities
"blender", "mixer", "food processor", "toaster", "coffee maker", "espresso machine",
"rice cooker", "pressure cooker", "kettle", "juicer", "grill", "slow cooker",
"induction cooktop", "hot plate", "electric skillet", "water purifier",
# Kitchenware & utensils (from big to tiny)
"pot", "pan", "casserole", "wok", "baking tray", "baking sheet", "griddle", "roasting pan",
"colander", "strainer", "salad spinner", "cutting board", "chopping block", "rolling pin",
"mortar", "pestle", "measuring cup", "measuring spoon", "mixing bowl", "bowl", "plate",
"dish", "serving tray", "serving spoon", "ladle", "spatula", "whisk", "tongs", "fork",
"knife", "teaspoon", "tablespoon", "chopsticks", "peeler", "grater", "slicer", "can opener",
"bottle opener", "corkscrew", "jar opener", "ice tray", "egg cup", "egg slicer", "apple corer",
"pizza cutter", "nutcracker", "garlic press", "tea infuser", "tea strainer",
"butter dish", "bread box", "salt shaker", "pepper grinder", "condiment bottle", "sugar bowl",
"oil dispenser", "spice rack", "spice jar", "herb container", "tin", "plastic container",
"cling film", "foil", "zipper bag", "silicone mat", "napkin holder", "placemat", "coaster",
# Tableware & drinkware
"cup", "mug", "teacup", "saucer", "glass", "wine glass", "champagne flute", "water bottle",
"thermos", "jug", "pitcher", "decanter", "shot glass", "beer stein",
# Cleaning & maintenance (major to micro)
"broom", "mop", "bucket", "dustpan", "scrub brush", "toilet brush", "cleaning cloth",
"rag", "towel", "duster", "sponges", "dish scrubber", "scouring pad", "gloves",
"wipes", "spray bottle", "detergent", "soap bar", "liquid soap", "hand sanitizer",
"bleach", "air freshener", "disinfectant", "cleaner", "toilet cleaner", "drain opener",
"pest spray", "pesticide", "herbicide", "mosquito coil", "mosquito net",
"fly swatter", "mouse trap",
# Bathroom & personal care
"shower curtain", "soap dish", "toothbrush", "toothpaste", "mouthwash", "floss",
"razor", "shaving foam", "shaving kit", "hairbrush", "comb", "hair dryer", "straightener",
"makeup kit", "perfume", "deodorant", "shampoo", "conditioner", "bath mat", "bath towel",
"face towel", "hand towel", "loofah", "bath sponge", "nail clipper", "tweezers",
"earbuds", "cotton balls", "facial tissue", "tissue box", "plasters", "bandage", "first aid kit",
"thermometer",
# Bedroom & clothing (all sizes/types)
"clothes", "shirt","t-shirt", "jeans", "trousers", "shorts", "skirt", "dress", "gown", "nightwear",
"undergarments", "vest", "socks", "stockings", "hat", "cap", "belt", "tie", "scarf", "gloves",
"coat", "jacket", "sweater", "blazer", "raincoat", "shoes", "slippers", "sandals", "boots",
"flip-flops",
"wardrobe", "closet", "hanger", "drawer organizer", "shoe rack", "shoe organizer",
"laundry basket", "clothespin", "clothesline", "ironing board", "stepladder",
# Decor & small accessories
"candle", "incense", "vase", "flower", "plant", "artificial plant", "bonsai", "terrarium",
"planter", "pot", "picture frame", "painting", "poster", "statue", "figurine", "magnet",
"snow globe", "wind chime", "clock", "wall clock", "alarm clock", "table clock",
"calendar", "diary", "notebook", "journal", "pen", "pencil", "marker", "highlighter",
# Office & tech (big to tiny)
"laptop", "desktop computer", "monitor", "keyboard", "mouse", "printer", "scanner",
"router", "modem", "wifi extender", "flash drive", "SSD", "hard drive", "memory card",
"card reader", "battery", "power bank", "charger", "extension cord", "power strip",
"adapter", "earphones", "headphones", "speaker", "camera", "tripod", "webcam",
"microphone", "joystick", "video game console", "controller", "remote", "SIM card",
"stylus", "phone", "tablet", "case", "screen protector", "charger cable",
# Stationery & micro objects
"stapler", "staple pin", "pin", "thumbtack", "pushpin", "paper clip", "binder clip",
"rubber band", "button", "needle", "thread", "string", "lace", "cord", "shoelace",
"zipper", "snap", "clip", "tag", "label", "badge", "ID card", "safety pin", "hairpin",
# Workshop & utility (tiny to large)
"toolbox", "screwdriver", "hammer", "wrench", "spanner", "pliers", "tape measure",
"level", "saw", "chisel", "file", "drill", "drill bit", "nut", "bolt", "screw", "nail",
"hook", "bracket", "plate", "hinge", "spring", "washer", "grommet", "plug", "socket",
"wire", "cable", "fuse", "switch", "bulb", "tube light", "LED", "night lamp", "shade",
# Kitchen: every micro item
"spice jar", "herb jar", "tea bag", "coffee pod", "bottle cap", "straw", "toothpick",
"cocktail stick", "pick", "lemon squeezer", "muddler", "rim salt", "sachet", "wrapper",
"packet", "label", "coupon", "recipe card",
# Living room/balcony/garden
"garden chair", "garden table", "patio furniture", "swing", "hammock", "umbrella",
```

```
        "grill", "BBQ", "fire pit", "watering can", "shovel", "rake", "hoe", "spade",
        # Miscellaneous tiny household items
        "coin", "stamp", "seal", "ring", "necklace", "bracelet", "earring", "anklet", "brooch",
        "cufflink", "tie pin", "lapel pin", "locket", "pendant",
        "key", "keychain", "lock", "padlock", "comb", "brush", "lint roller", "sticky note",
        "envelope", "postcard", "ticket", "token", "chip", "dice", "board game token",
        "playing card", "domino", "puzzle piece", "rubik's cube", "marble", "bead", "button battery",
        # Pets & animals
        "pet bed", "pet collar", "leash", "tag", "dog bowl", "cat bowl", "litter tray",
        "scratching post", "cat tree", "feeder", "waterer", "aquarium", "fish tank", "birdcage",
        "pet toy", "ball", "chew toy", "treat container",
        # Kids, babies, toys
        "baby bottle", "pacifier", "teether", "rattle", "play mat", "playpen", "blocks",
        "building set", "stuffed toy", "teddy bear", "doll", "puppet", "car toy", "train set",
        "board game", "game console", "controller", "puzzle", "coloring book", "crayon", "paint",
        "brush", "finger paint", "sticker", "stamp", "glitter", "googly eyes", "craft supplies",
        # Outdoors/sports
        "bicycle", "helmet", "bike lock", "skateboard", "basketball", "football", "tennis racket",
        "shuttlecock", "badminton racket", "cricket bat", "ball", "glove", "jersey", "sports shoe",
        # Emergency, safety, security
        "fire extinguisher", "smoke detector", "alarm", "flashlight", "emergency light",
        "battery backup", "whistle", "pepper spray", "first aid kit", "bandage", "plaster",
        "splint", "ice pack", "hot pack",
        # Home office: every object
        "pen holder", "desk organizer", "paper tray", "letter opener", "envelope", "stamp",
        "post-it note", "marker", "eraser", "whiteboard", "whiteboard marker", "chalk",
        "chalkboard", "magnet", "pin", "flag", "file folder", "divider", "index tab"
    ]


    def recognize_object(self, image_path):
        """Identify the object in the given image"""
        image = Image.open(image_path).convert("RGB")
        prompts = [f"a photo of a {label}" for label in self.knowledge_base]

        inputs = self.processor(
            text=prompts, images=image, return_tensors="pt", padding=True
        ).to(self.device)

        with torch.no_grad():
            outputs = self.model(**inputs)
            probs = outputs.logits_per_image.softmax(dim=1)

        best_idx = probs.argmax().item()
        confidence = probs[0, best_idx].item() * 100
        detected = self.knowledge_base[best_idx]

        return {
            "file": image_path,
            "detected_object": detected,
            "confidence": confidence,
        }

# ============================================
# 🚀 Run Agent Loop
# ============================================

def run_agent():
    print("📤 Upload images one by one (type 'ok' when done)\n")
    agent = VisionAgent()
    results = []
    count = 0

    while True:
        user_input = input("Press Enter to upload next image or type 'ok' to finish: ").strip().lower()
        if user_input == "ok":
            break

        uploaded = files.upload()
        if not uploaded:
            print("❌ No image uploaded. Try again.")
            continue

        image_name = list(uploaded.keys())[0]
        count += 1

        result = agent.recognize_object(image_name)
        results.append(result)
```

```
            results.append(result)

            print(f"✅ Image {count}: '{image_name}' → Detected: {result['detected_object']} "
                  f"(Confidence: {result['confidence']:.2f}%)\n")

        # 📋 Summary
        if results:
            print("\n📊 Final Agent Summary:")
            for i, res in enumerate(results, start=1):
                print(f"{i}. {res['file']} → {res['detected_object']} ({res['confidence']:.1f}%)")
        else:
            print("No images were processed.")

# ============================================
# ▶️ Execute
# ============================================

run_agent()
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.1)
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: torchvision in /usr/local/lib/python3.12/dist-packages (0.23.0+cu126)
Requirement already satisfied: torchaudio in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: pillow in /usr/local/lib/python3.12/dist-packages (11.3.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.20.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.3
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
```

```
!pip install transformers torch torchvision torchaudio pillow

from transformers import CLIPProcessor, CLIPModel
from google.colab import files
from PIL import Image
import torch

# ============================================
# 🧠 Define SceneAgent
# ============================================

class SceneAgent:
    def __init__(self, model_name="openai/clip-vit-large-patch14"):
        print("🏗️ Initializing SceneAgent with model:", model_name)
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        self.model = CLIPModel.from_pretrained(model_name).to(self.device)
        self.processor = CLIPProcessor.from_pretrained(model_name)

        # 🧩 Extensive indoor vocabulary
        self.scene_labels = [
            "bedroom", "kitchen", "living room", "dining room", "hall", "bathroom", "balcony", "corridor",
            "wardrobe area", "workspace", "study room", "office desk", "library corner", "kids playroom",
            "laundry area", "washing area", "pooja room", "shoe rack area", "garage", "storage room",
            "pantry", "staircase area", "home theater", "veranda", "garden area", "terrace", "store room",
            "mirror area", "sink area", "computer desk", "TV unit area", "couch seating area", "bookshelf area",
            "kitchen shelf", "kitchen counter", "fridge corner", "dining table area", "bedside area",
            "window side corner", "plant corner", "dresser area", "closet interior", "entryway", "hallway",
            "patio", "game room", "sofa lounge", "guest room", "music room", "craft room", "art studio",
            "exercise or gym area", "walk-in closet", "washing machine area", "cupboard area", "sink counter",
            "shoe closet", "toilet area", "mirror wall", "minimal empty room", "messy room", "clean modern interior",
            "storage shelf", "wardrobe with clothes", "study desk", "makeup table", "bedside table",
            "book storage rack", "TV cabinet", "sideboard area", "kitchen utensils area", "small balcony",
            "large hall", "tiny bedroom", "luxury kitchen", "modular wardrobe", "pooja shelf", "fridge area",
            "kitchen stove area", "microwave corner", "plant shelf", "window sitting area", "mirror dressing table"
        ]

    def classify_scene(self, image_path):
        """Identify what kind of room or area the image shows"""
        image = Image.open(image_path).convert("RGB")
        text_prompts = [f"This image shows a {label}." for label in self.scene_labels]

        inputs = self.processor(text=text_prompts, images=image, return_tensors="pt", padding=True).to(self.device)
        with torch.no_grad():
            outputs = self.model(**inputs)
            probs = outputs.logits_per_image.softmax(dim=1)

        best_idx = probs.argmax().item()
        confidence = probs[0, best_idx].item() * 100
        detected_scene = self.scene_labels[best_idx]

        return {
            "file": image_path,
            "detected_scene": detected_scene,
            "confidence": confidence
        }


# ============================================
# 🚀 Run SceneAgent
# ============================================
```

```
def run_scene_agent():
    print("📤 Please upload a room image (any type: kitchen, hall, bedroom, balcony, pooja room, workspace, etc.)")
    uploaded = files.upload()

    if not uploaded:
        print("❌ No image uploaded.")
        return

    image_path = list(uploaded.keys())[0]
    agent = SceneAgent()

    result = agent.classify_scene(image_path)

    print("\n🏠 Scene Detection Result:")
    print(f"This image shows a **{result['detected_scene']}** ({result['confidence']:.2f}% confidence)")

# ============================================
# ▶️ Execute
# ============================================

run_scene_agent()
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.1)
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: torchvision in /usr/local/lib/python3.12/dist-packages (0.23.0+cu126)
Requirement already satisfied: torchaudio in /usr/local/lib/python3.12/dist-packages (2.8.0+cu126)
Requirement already satisfied: pillow in /usr/local/lib/python3.12/dist-packages (11.3.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.20.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.36
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.3)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22.
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.7
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.80)
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.21)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
```