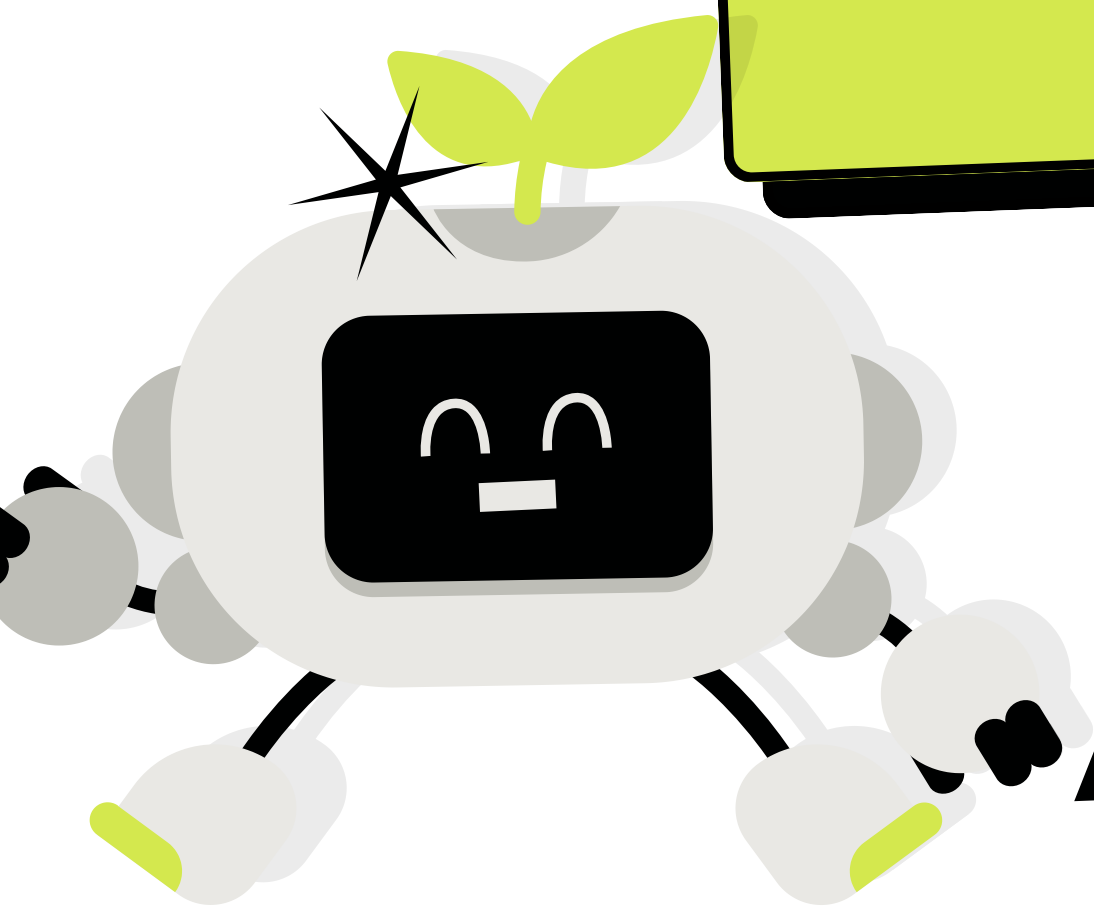


**SMART**

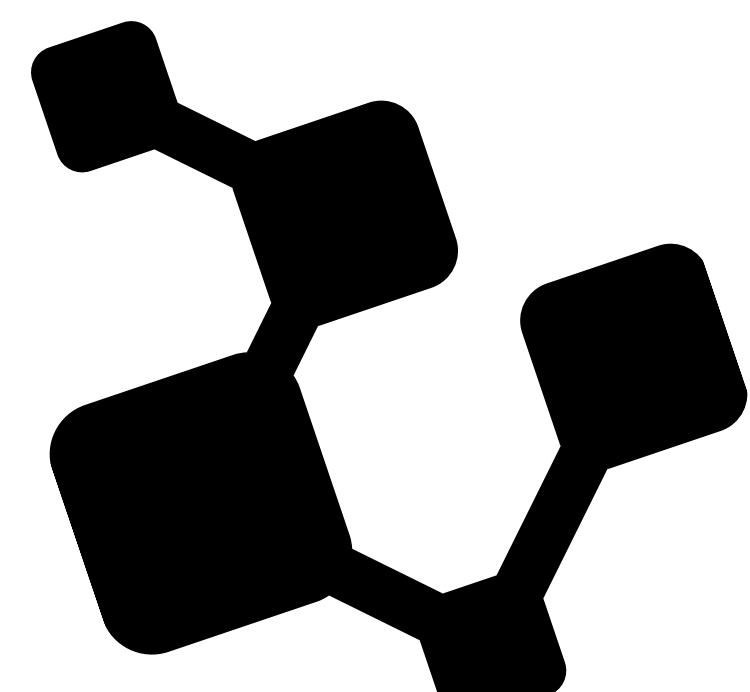


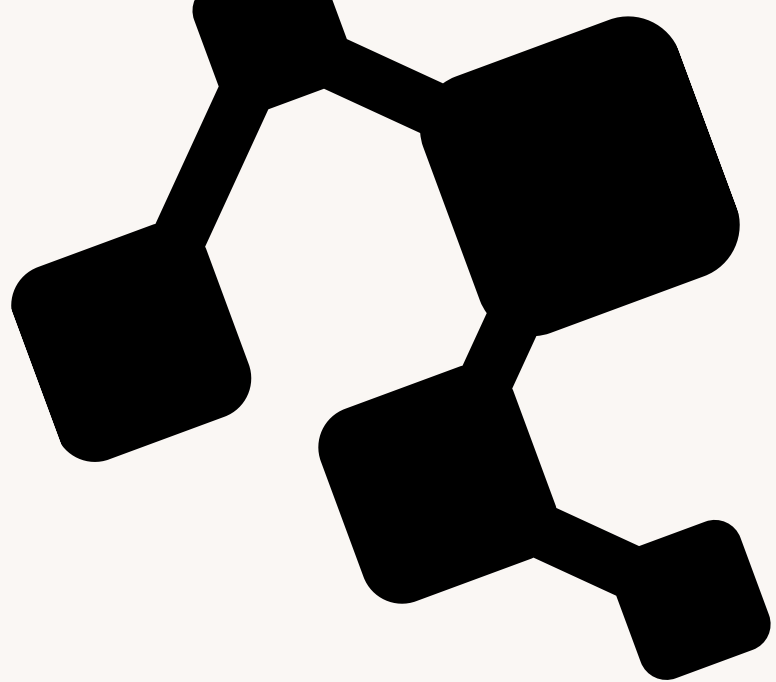
**WASTE**

**DETECTION**



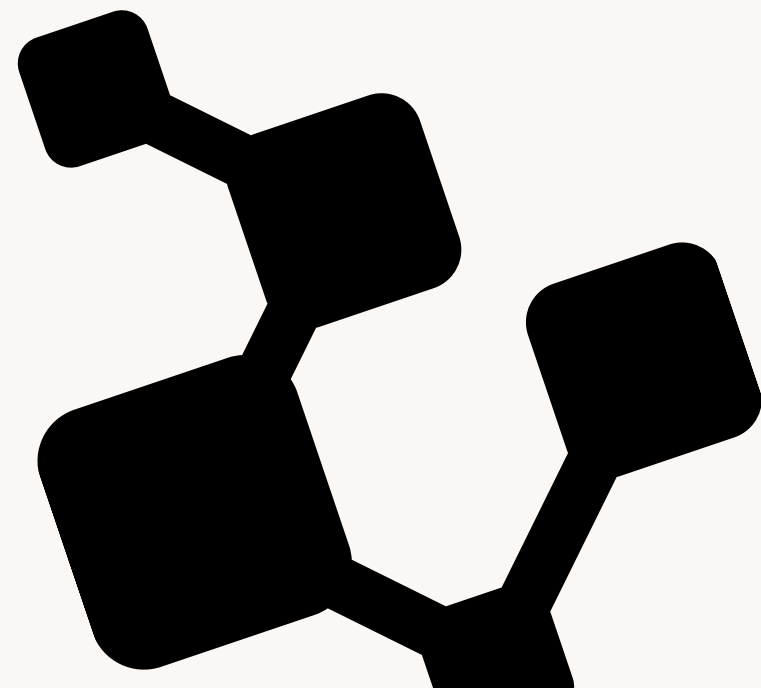
**AND SORTING**

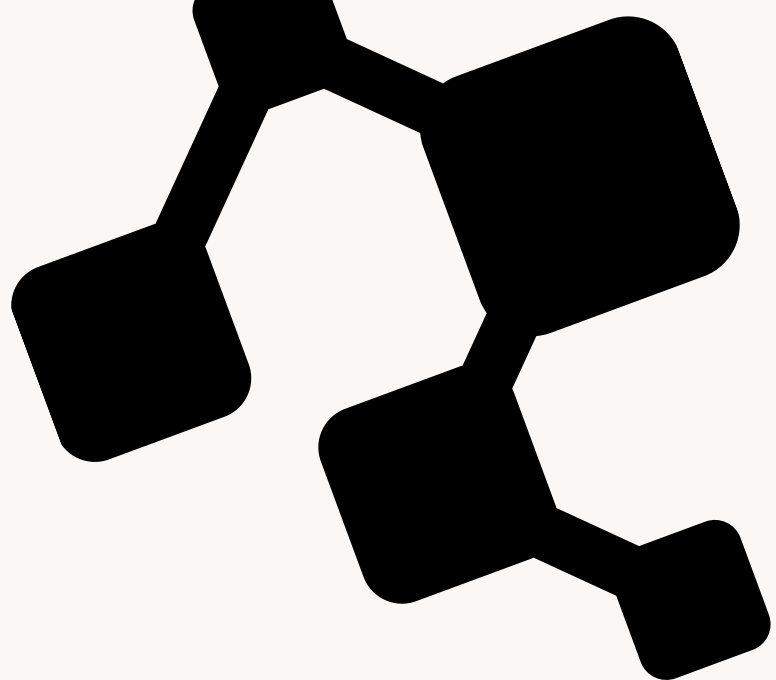




# **PROJECT**

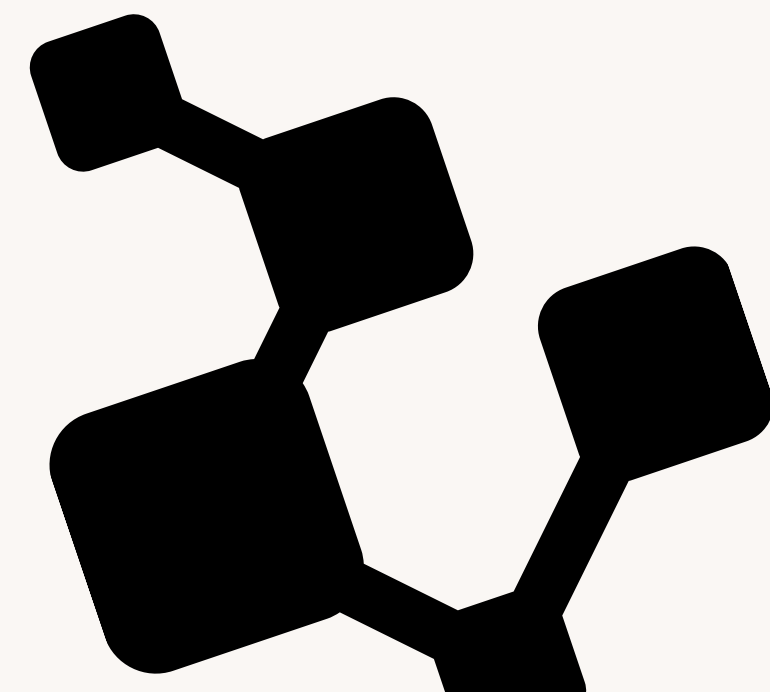
# **OVERVIEW**

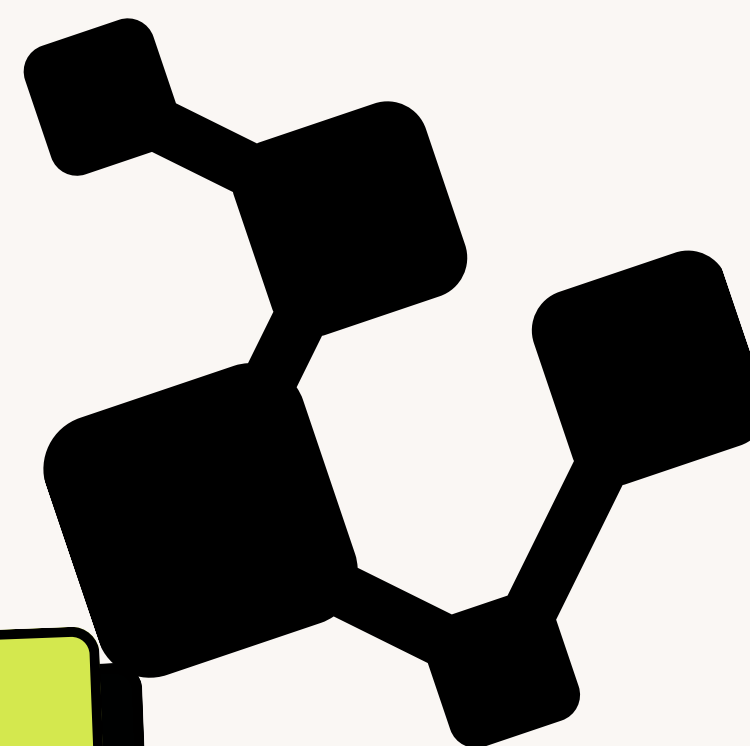
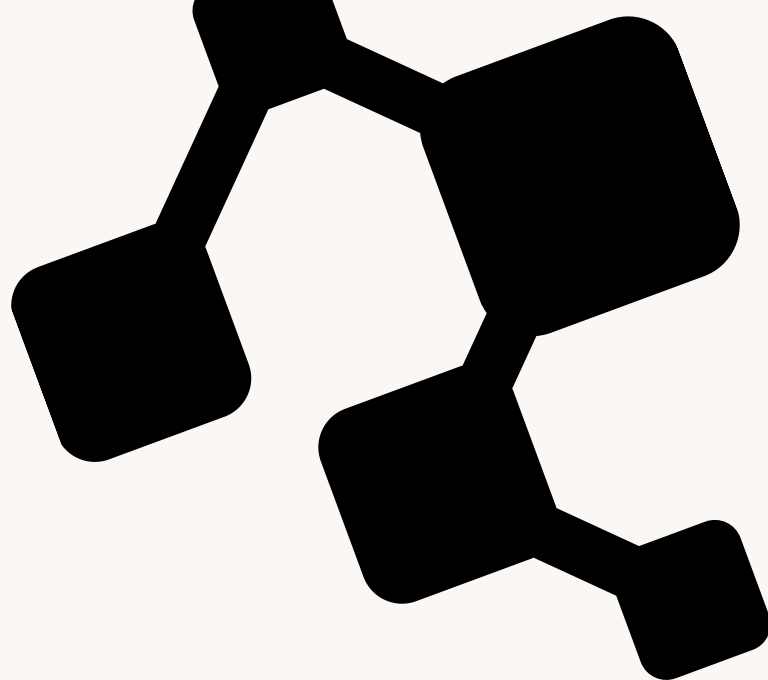




**DATA**

**COLLECTION**





 **DATA**

**PREPROCESSING**

# DATA PREPROCESSING

**IMAGE  
CLEANING**

01

**BALANCED  
CLASSES**

02

**IMAGE  
RESIZING**

03

# DATA PREPROCESSING

**DATA  
AUGMENTATION**

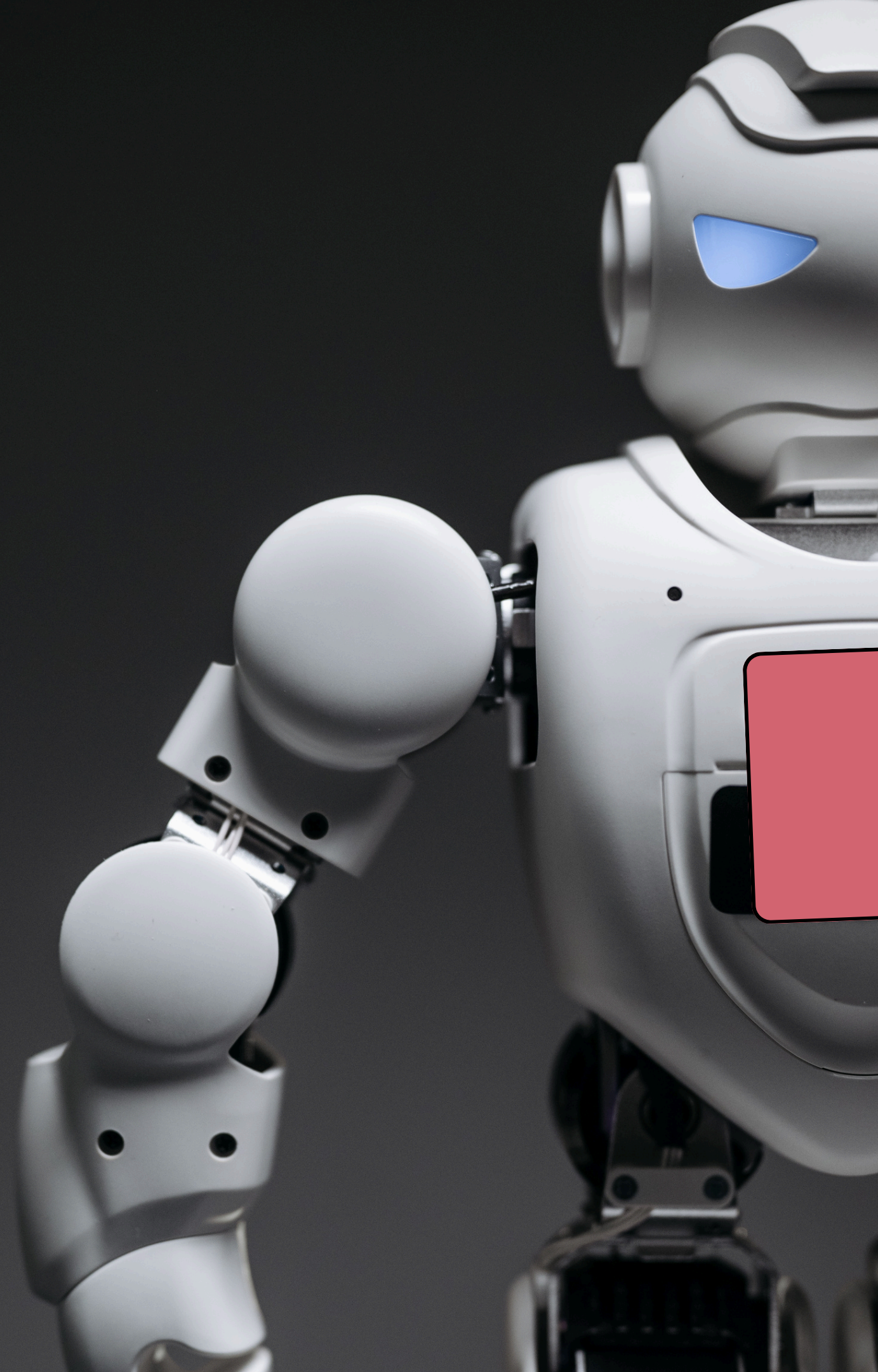
04

**REMOVE BAD  
SAMPLES**

05

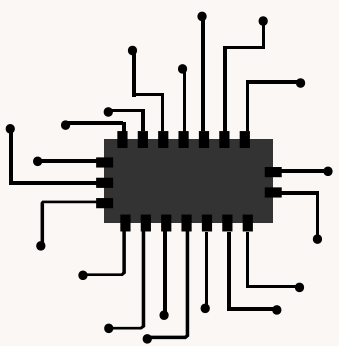
**TRAIN / VAL /  
TEST SPLIT**

06



MODEL

BUILDING

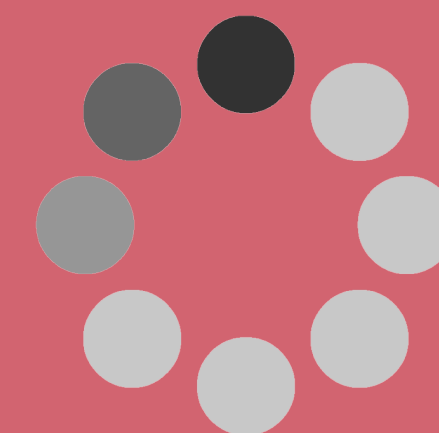
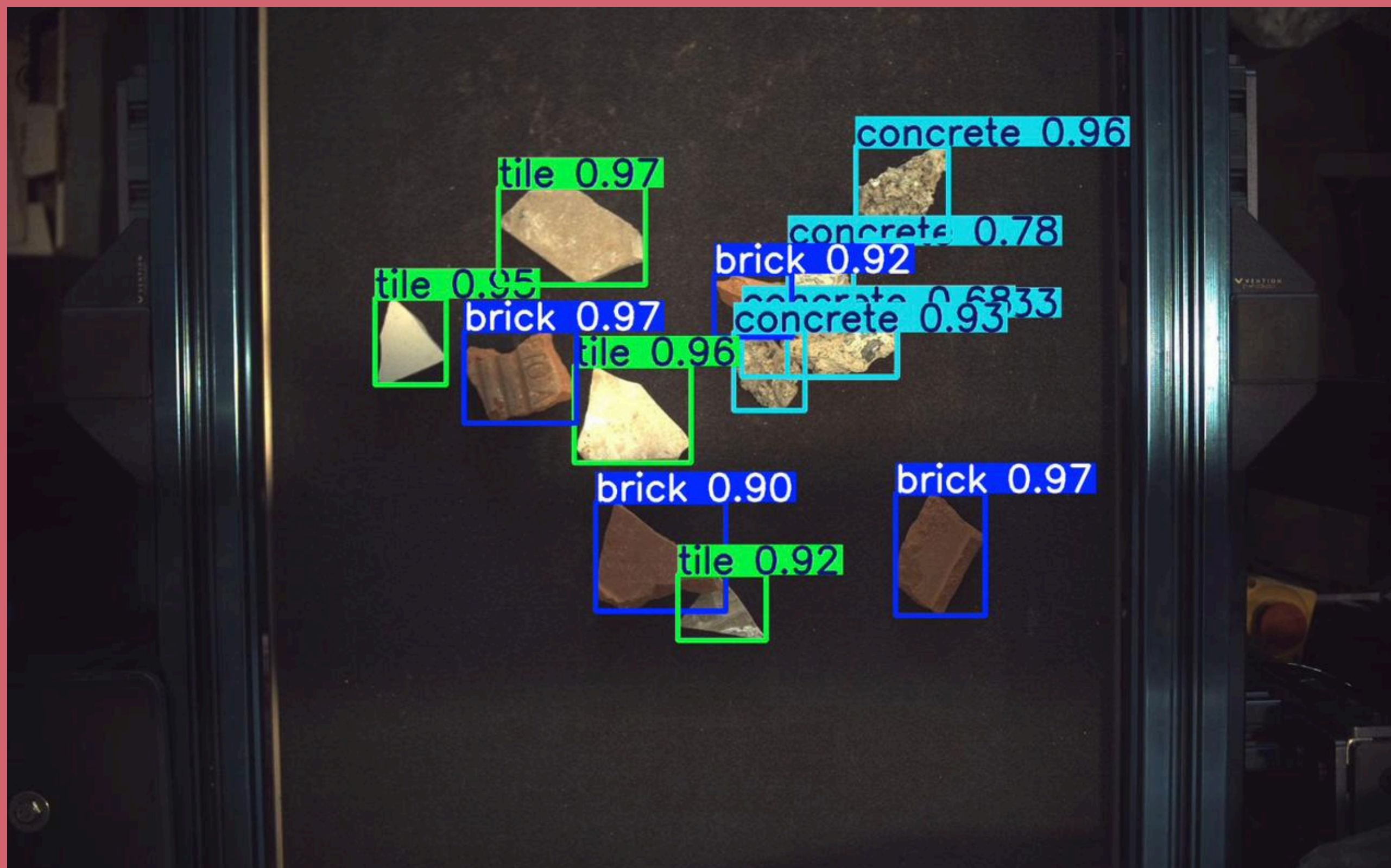


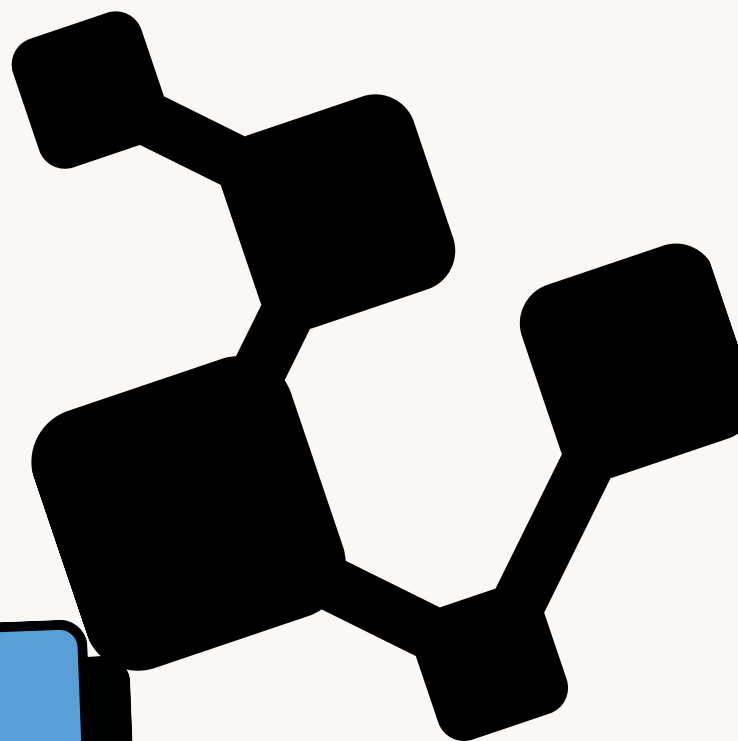
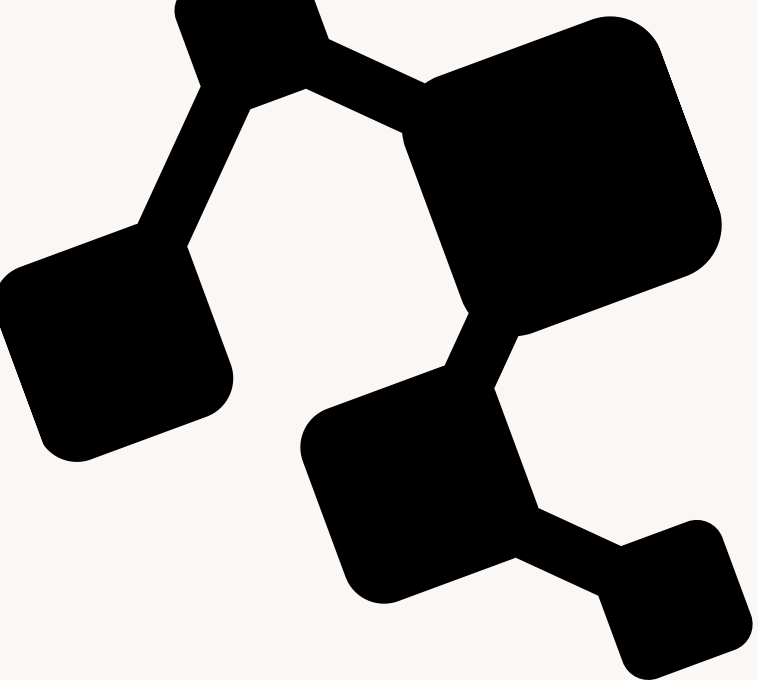
## **MODEL DEVELOPMENT & TRAINING PHASE**

**THE MODEL DEVELOPMENT PHASE FOCUSED ON SELECTING A HIGH-QUALITY DATASET, OPTIMIZING CLASS BALANCE, AND TRAINING AN OBJECT DETECTION MODEL CAPABLE OF ACCURATELY CLASSIFYING DIFFERENT TYPES OF WASTE IN REAL TIME. SEVERAL DATASETS WERE EVALUATED AND TESTED TO DETERMINE WHICH ONE PROVIDED THE BEST PERFORMANCE IN TERMS OF IMAGE QUALITY, ANNOTATION ACCURACY, CLASS DISTRIBUTION, AND COMPATIBILITY WITH YOLO MODELS. THROUGHOUT THIS PROCESS, DIFFERENT YOLO VERSIONS WERE EXPERIMENTED WITH, AND DATA AUGMENTATION WAS APPLIED WHERE NECESSARY TO IMPROVE ROBUSTNESS.**

**AFTER EXTENSIVE EXPERIMENTATION, THE CONSTRUCTION AND DEMOLITION WASTE OBJECT DETECTION DATASET (Codd) WAS IDENTIFIED AS THE MOST SUITABLE DATASET. IT OFFERED STRONG IMAGE DIVERSITY AND BALANCED WASTE CATEGORIES, LEADING TO STABLE TRAINING RESULTS. THE FINAL YOLOV8 MODEL ACHIEVED 97% ACCURACY ON THE TRAINING SET AND 90% ACCURACY ON THE VALIDATION SET, MAKING IT THE HIGHEST-PERFORMING MODEL AMONG ALL ATTEMPTS.**







 **MODEL**

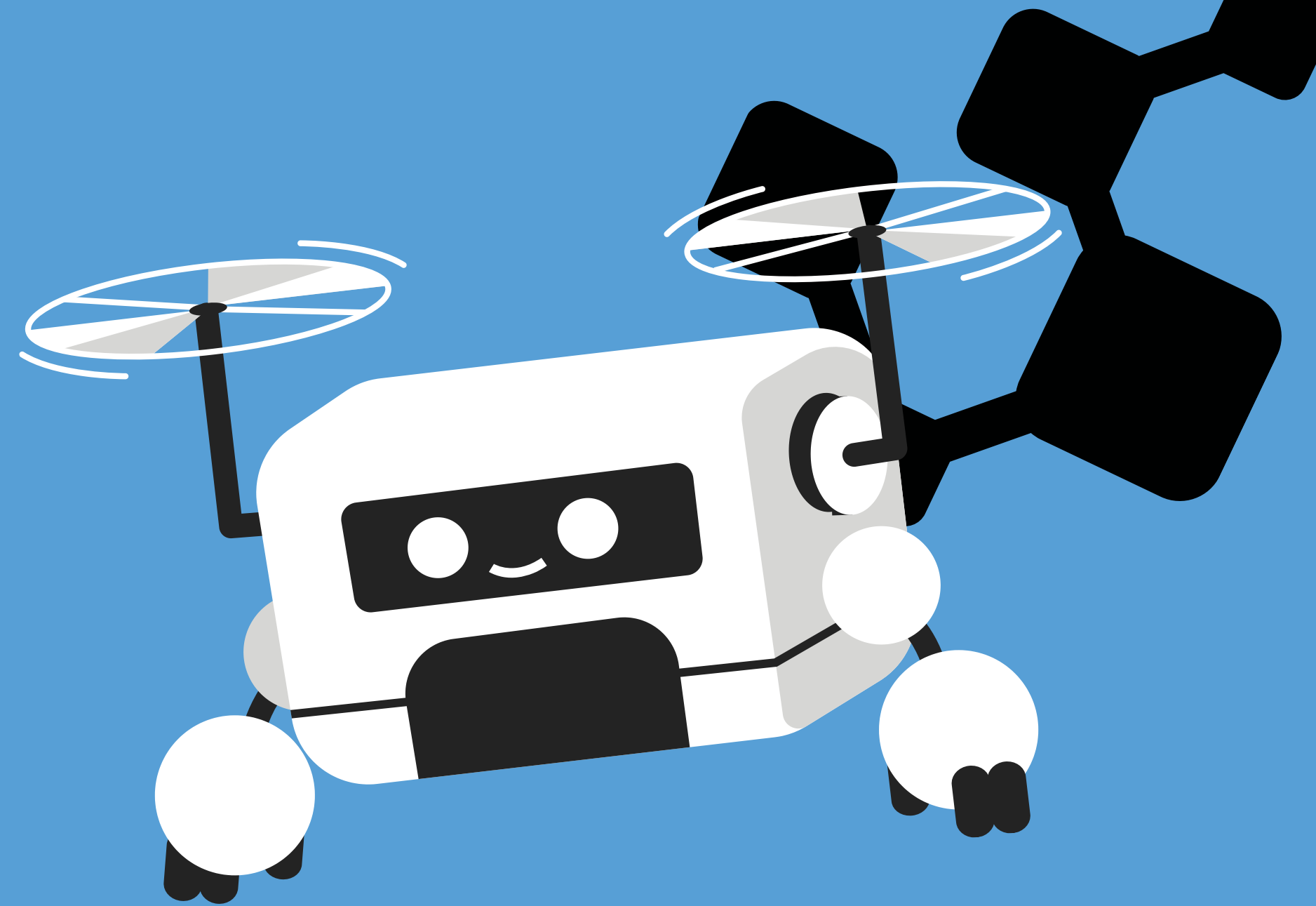
**EVALUATION**

1. Evaluate on Training Set

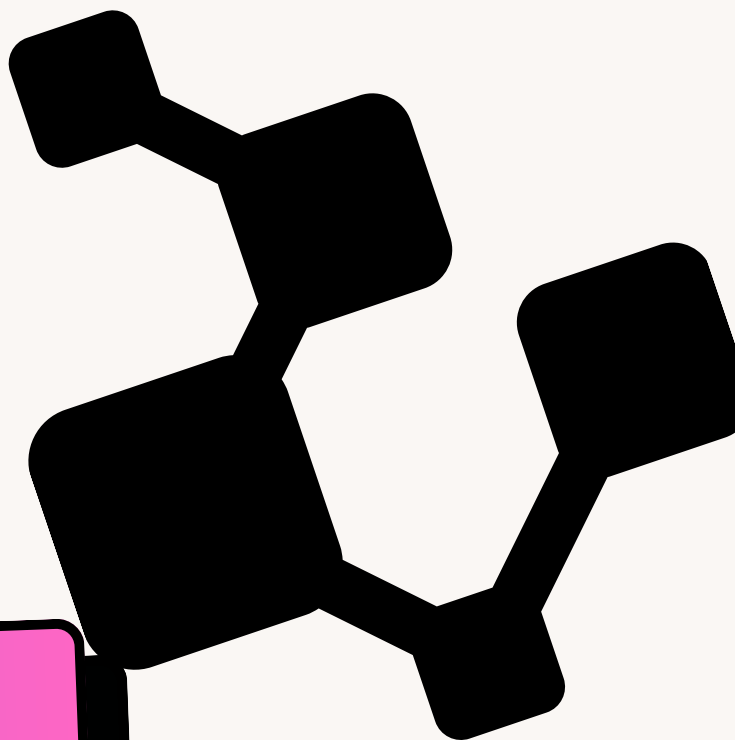
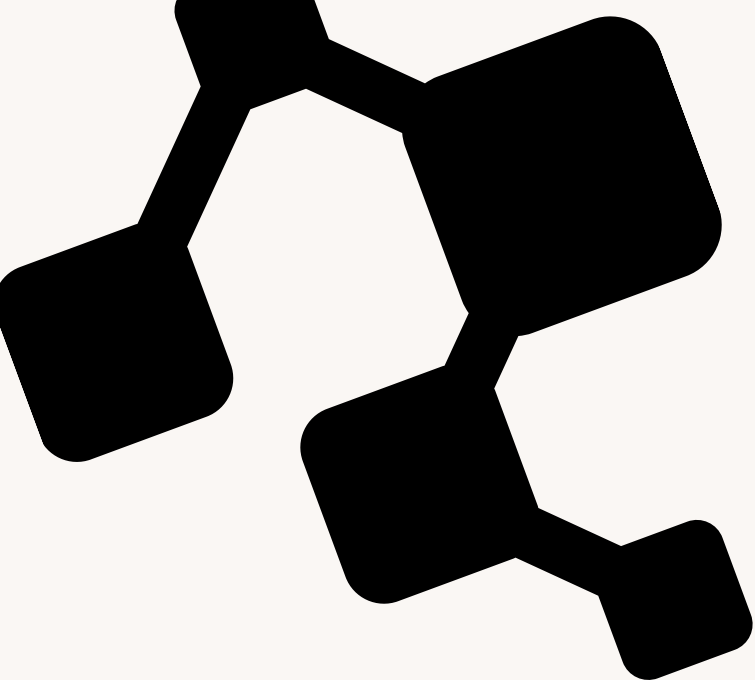
2. EVALUATE ON  
VALIDATION SET

3. EVALUATE ON  
TEST SET

4. FINE-TUNING &  
ITERATION



**MODEL  
EVALUATION**



**DEPLOYMENT**

# DEPLOYMENT

1

**PREPARE THE  
TRAINED  
MODEL**

2

**CHOOSE  
DEPLOYMENT  
PLATFORM**

3

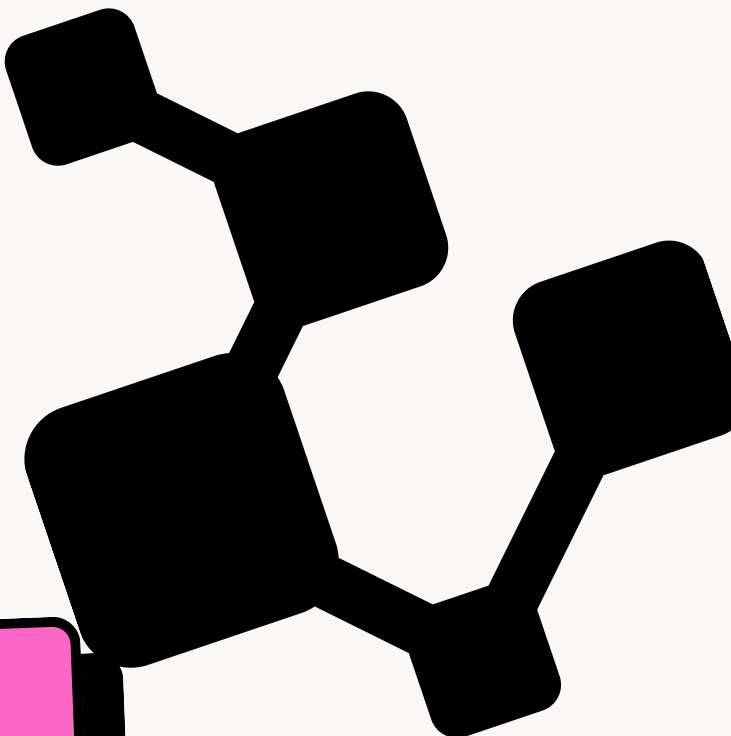
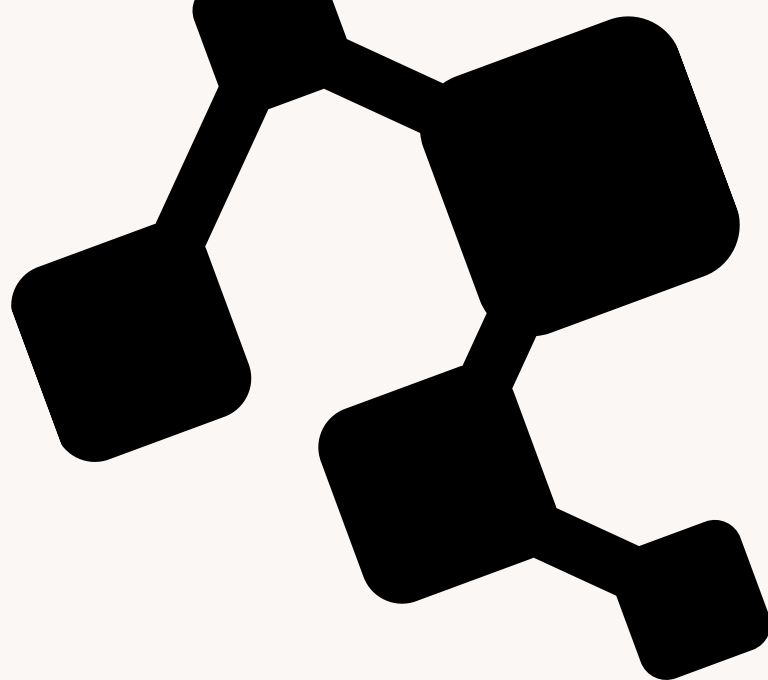
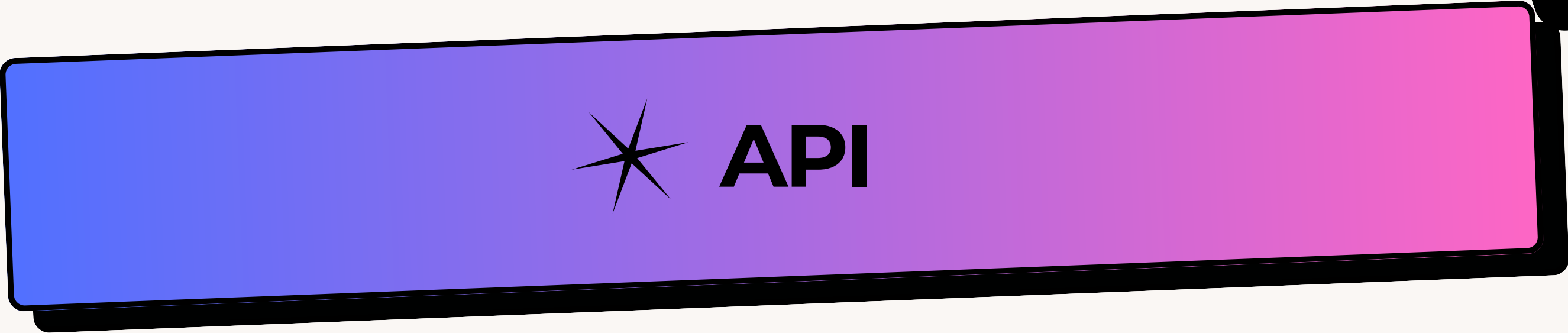
**DEVELOP  
INFERENCE  
PIPELINE**

4

**INTEGRATE WITH  
APPLICATION**

5

**TEST  
DEPLOYMENT**





**BUILT USING  
FASTAPI AND  
ONNX RUNTIME**

1

**RECEIVES AN  
IMAGE AND  
RETURNS  
DETECTED  
OBJECTS**

2

**STREAMLIT  
INTEGRATION**

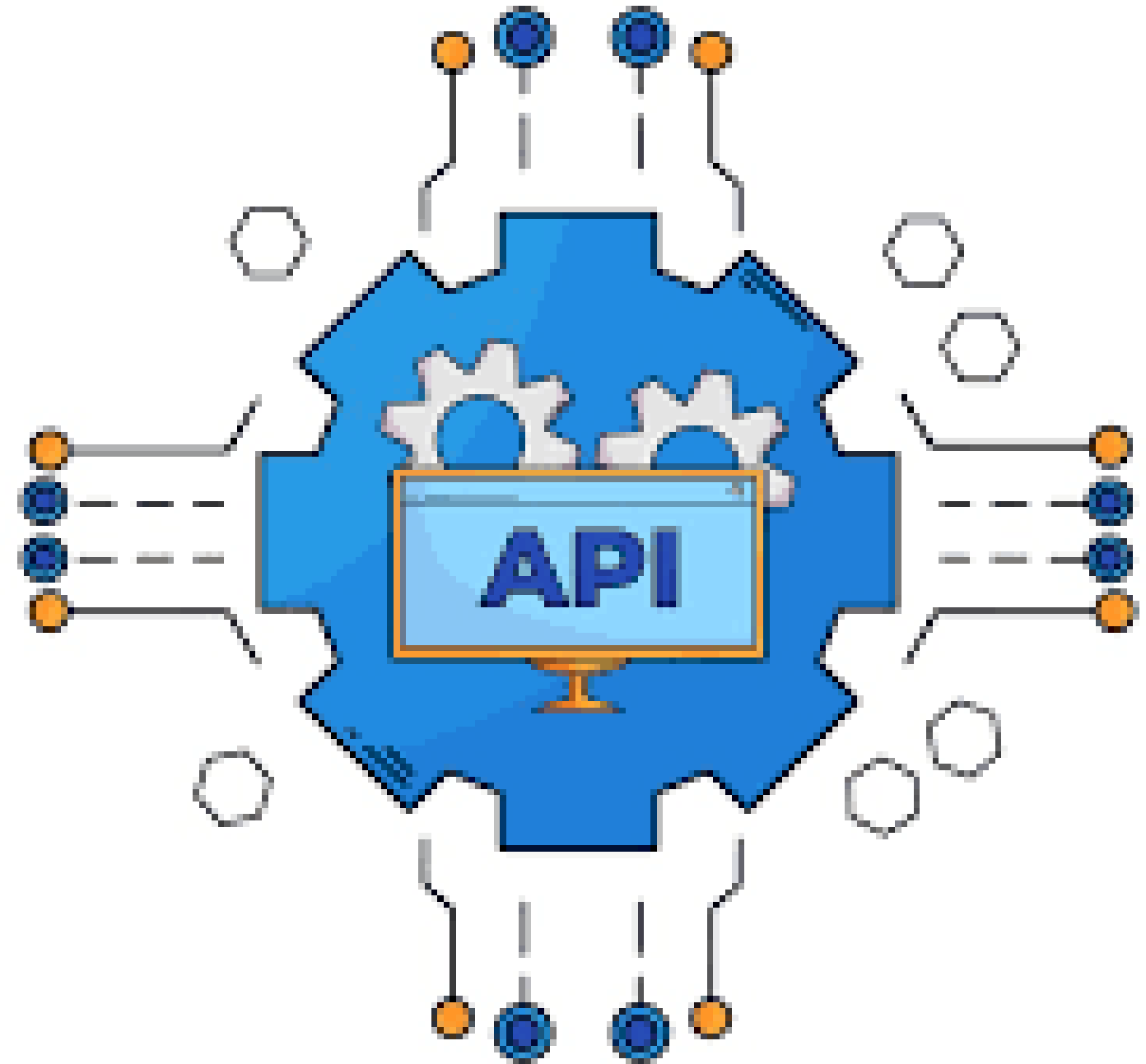
3

# API PREPROCESSING

01 | **FILTER & CLASSIFY**

02 | **NORMALIZATION**

03 | **CONVERT (BGR TO RGB)**



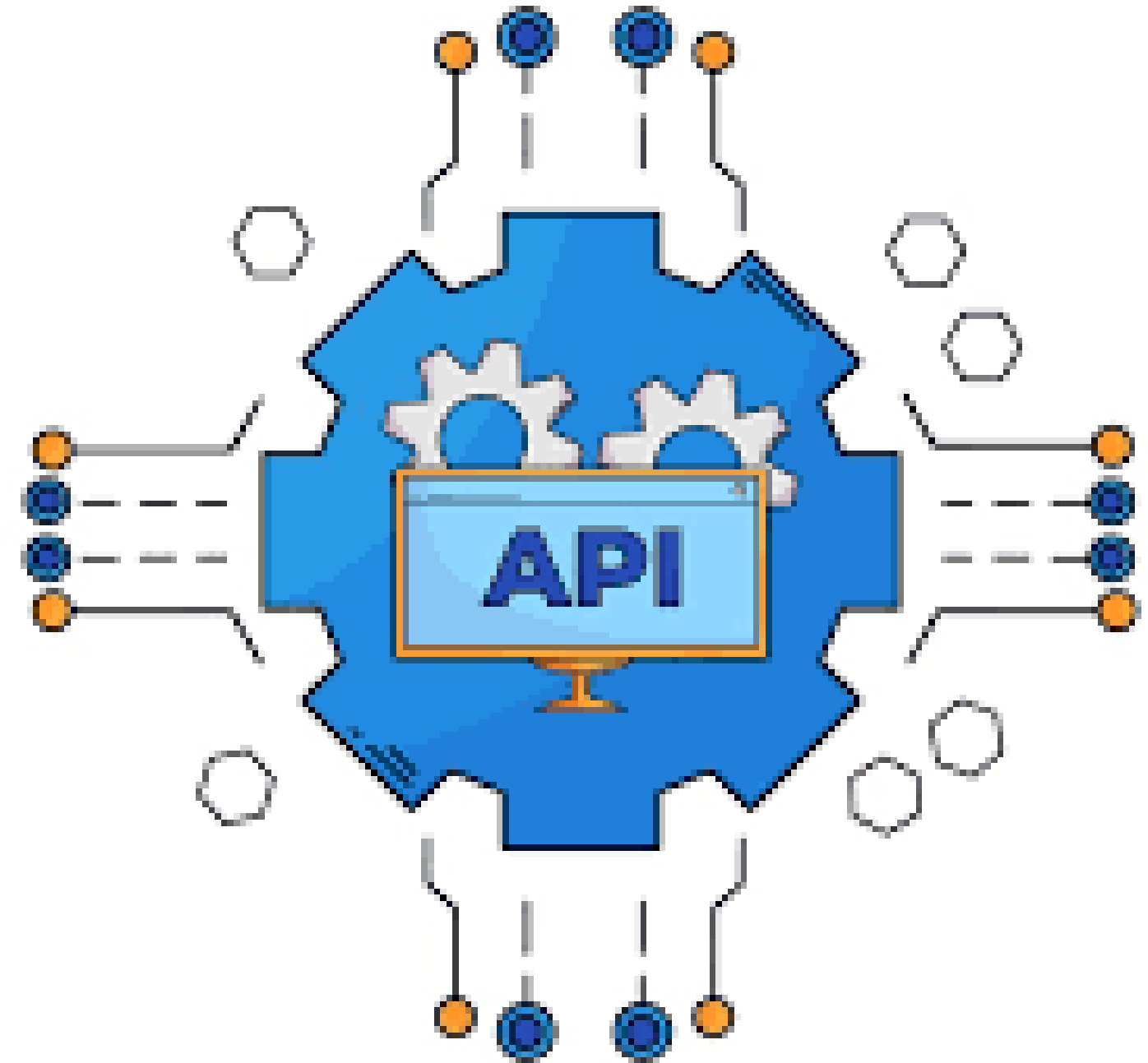
# API

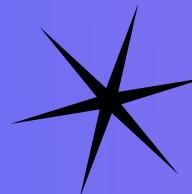
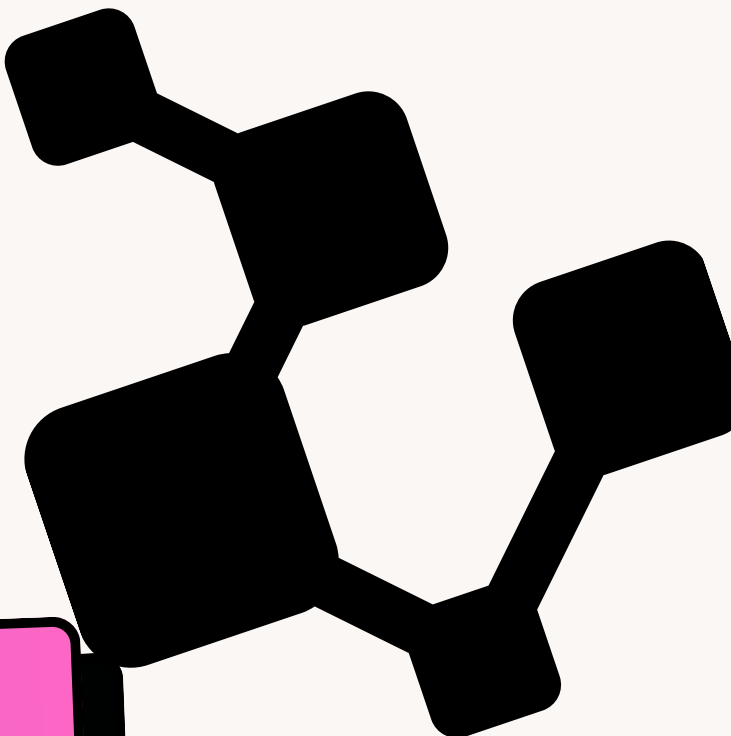
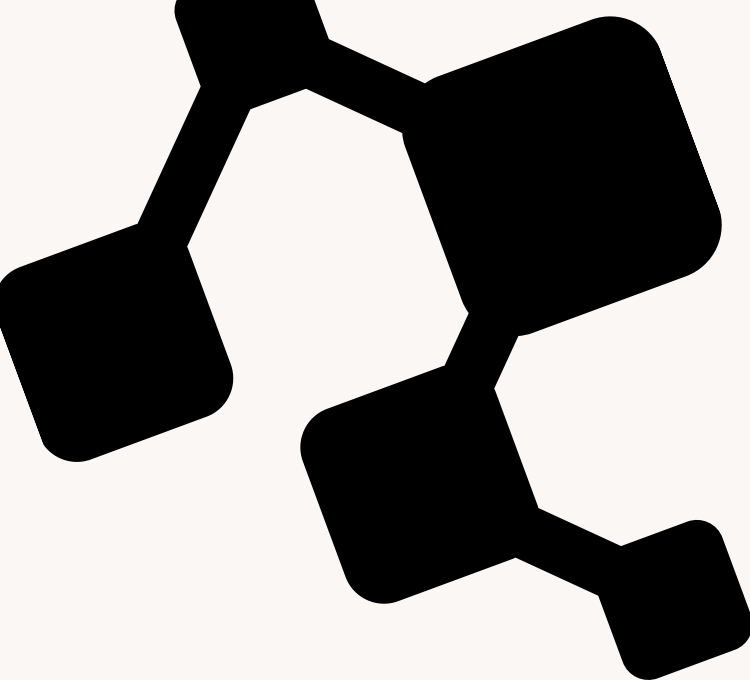
## POSTPROCESSING

01 **FILTER & CLASSIFY**

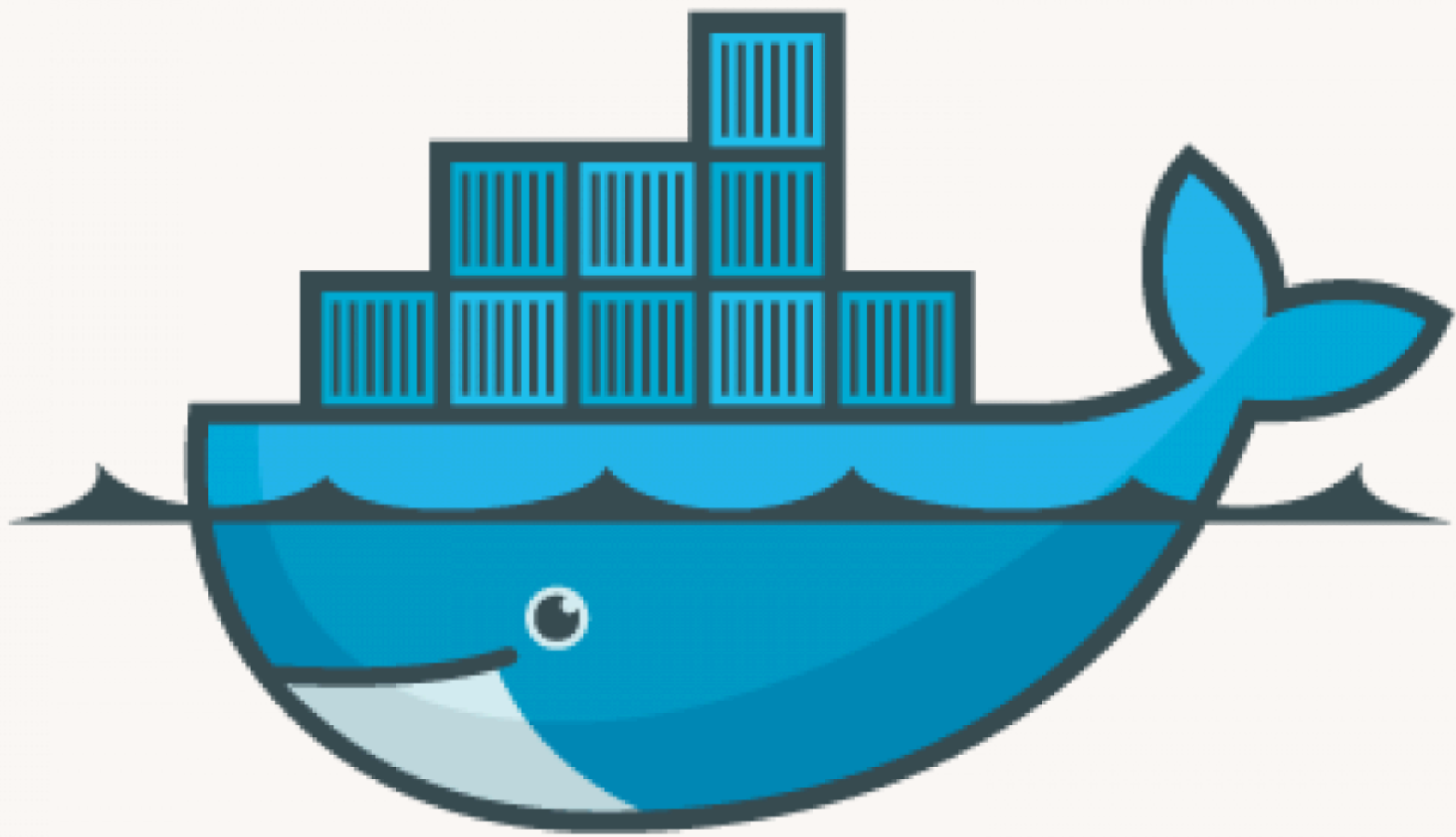
02 **CONVERT & SCALE BOXES**

03 **SUPPRESS OVERLAPS**





**DOCKERIZATION**



docker

## DOCKERIZATION

**WHAT IS  
DOCKER**

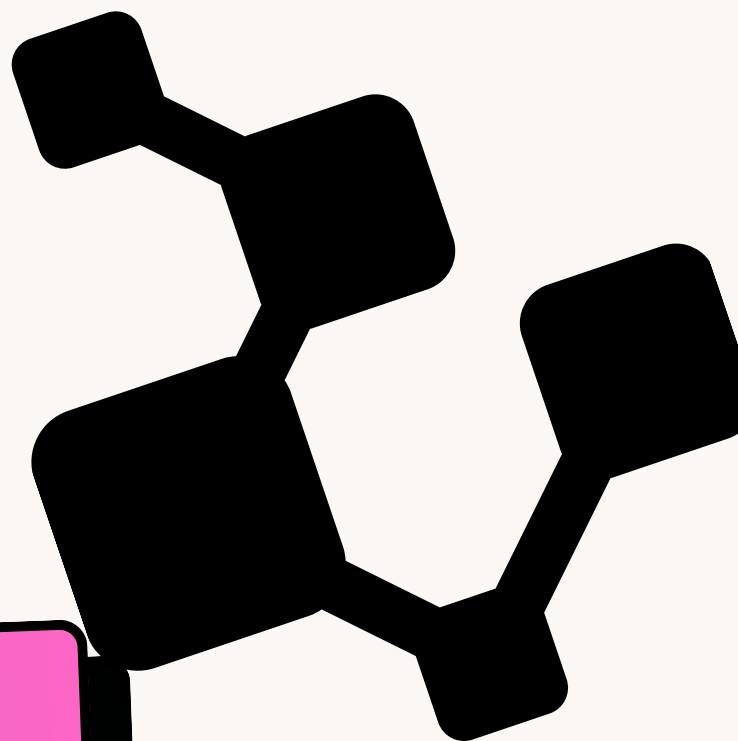
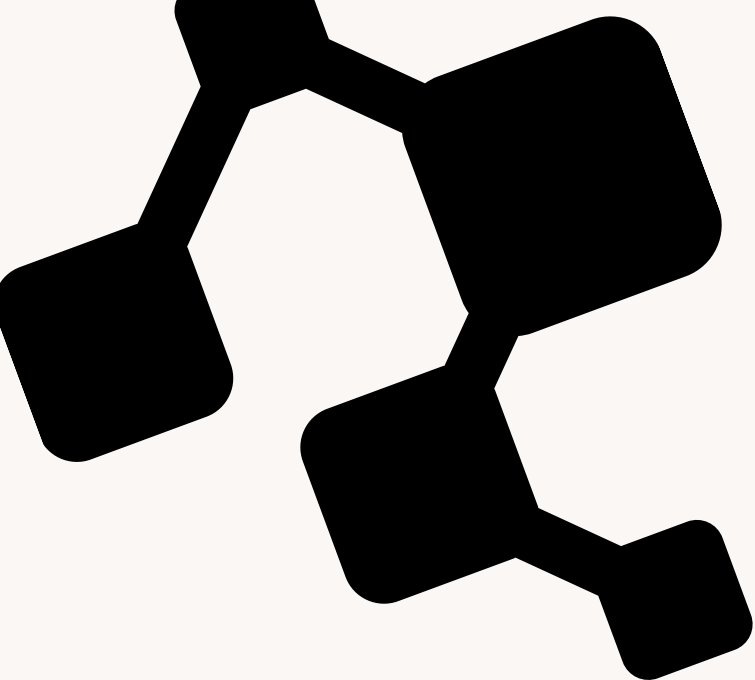
1

**WHY  
DOCKER IS  
USEFUL**

2

**BASIC  
TERMINOLOGY**

3



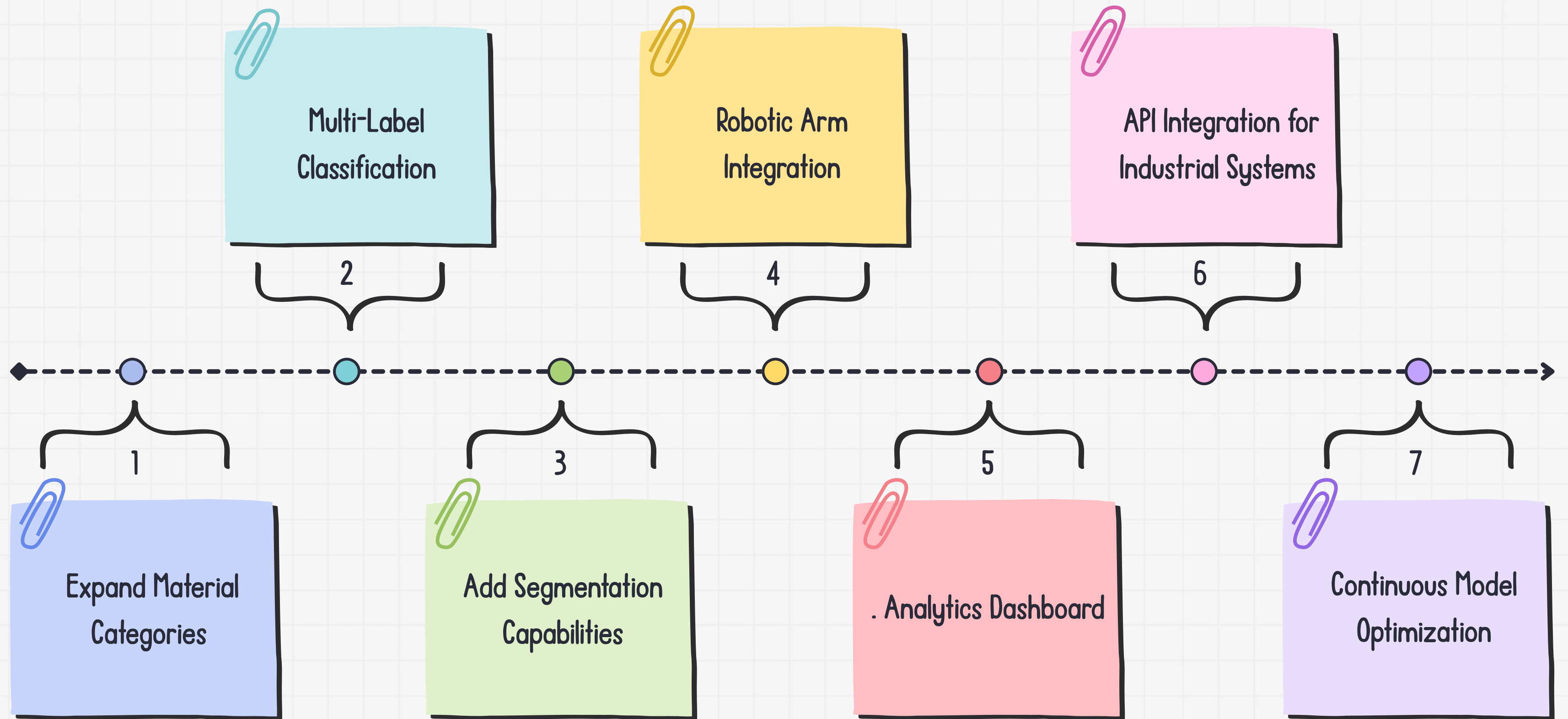
**APP REVIEW**

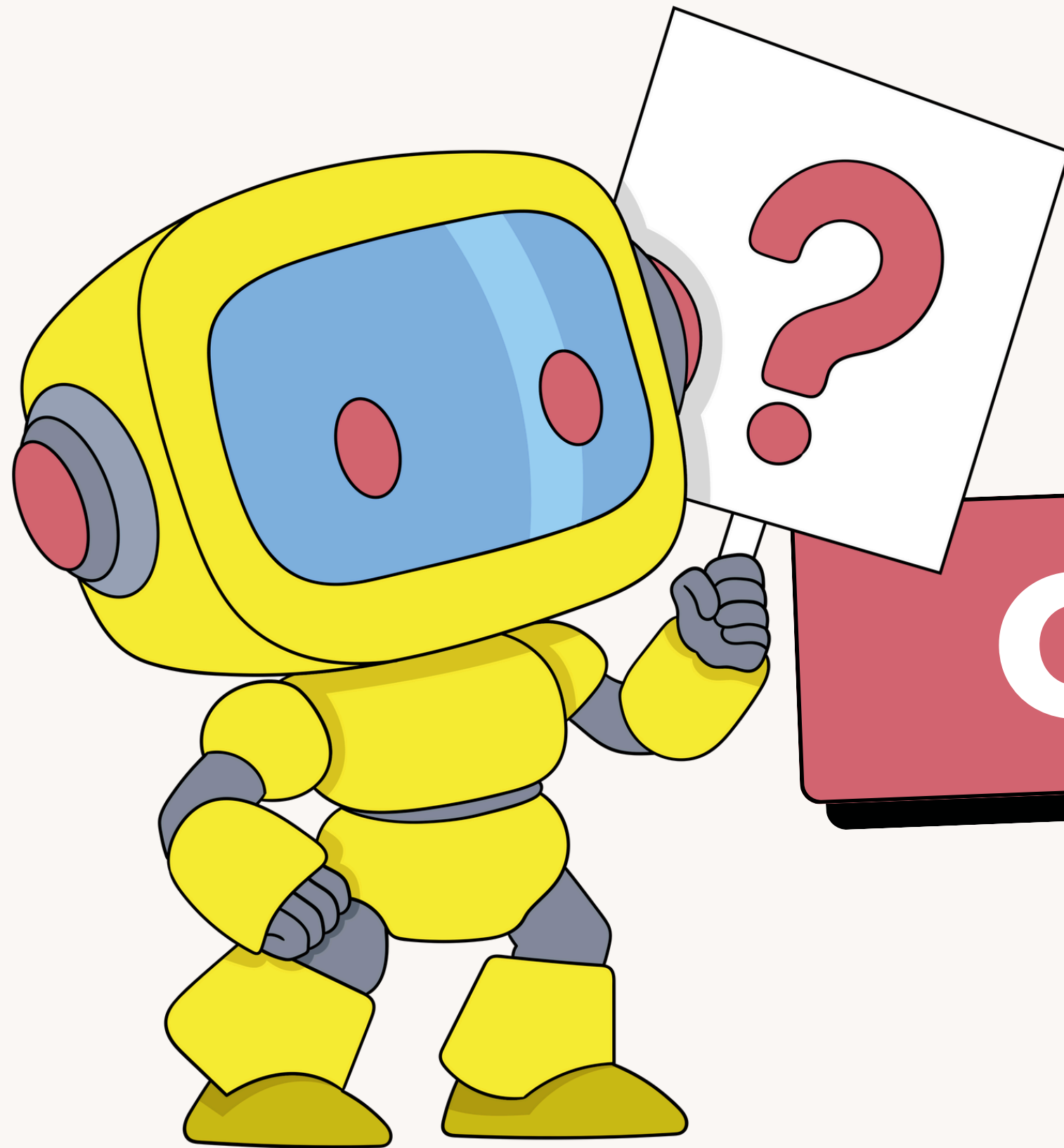
Abstract geometric shapes made of black squares and lines, located in the top-left and top-right corners of the image.

**FUTURE**

 **DEVELOPMENT**

**AND MARKETING**





**QUESTIONS**

**THANK YOU**