FlexoPlate Manager
Optimizing Plate Usage through Similarity Detection

Group ID : R25 - 082

SLIIT
FACULTY OF COMPUTING

SUPERVISOR DETAILS

SUPERVISOR

Ms Uthpala Samarakoon

CO-SUPERVISOR

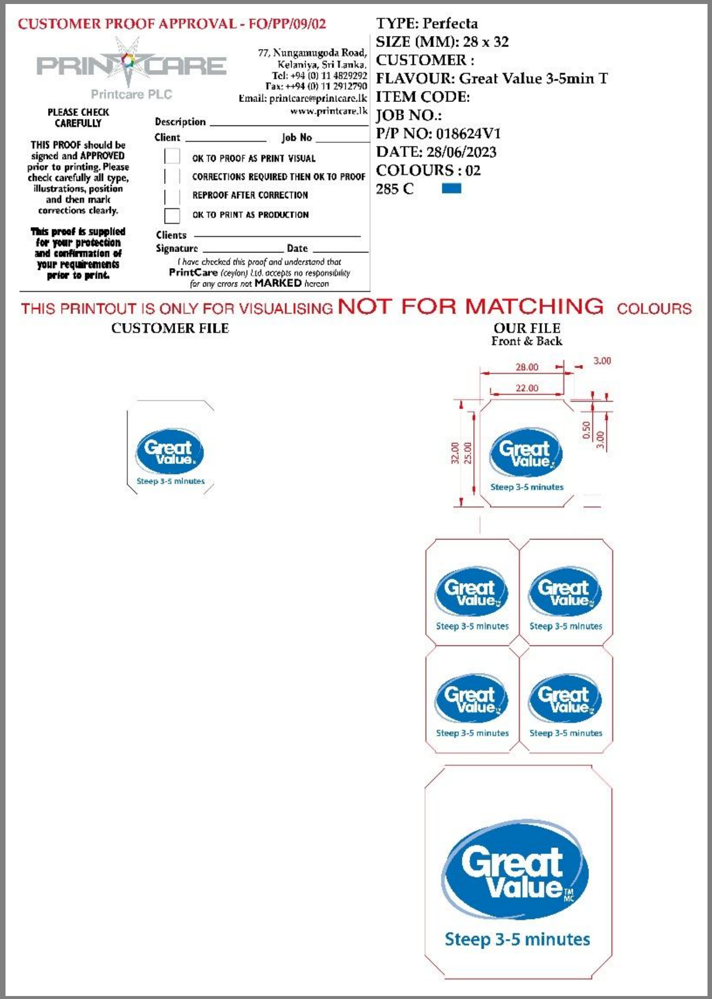
Ms Dushanthi Kuruppu

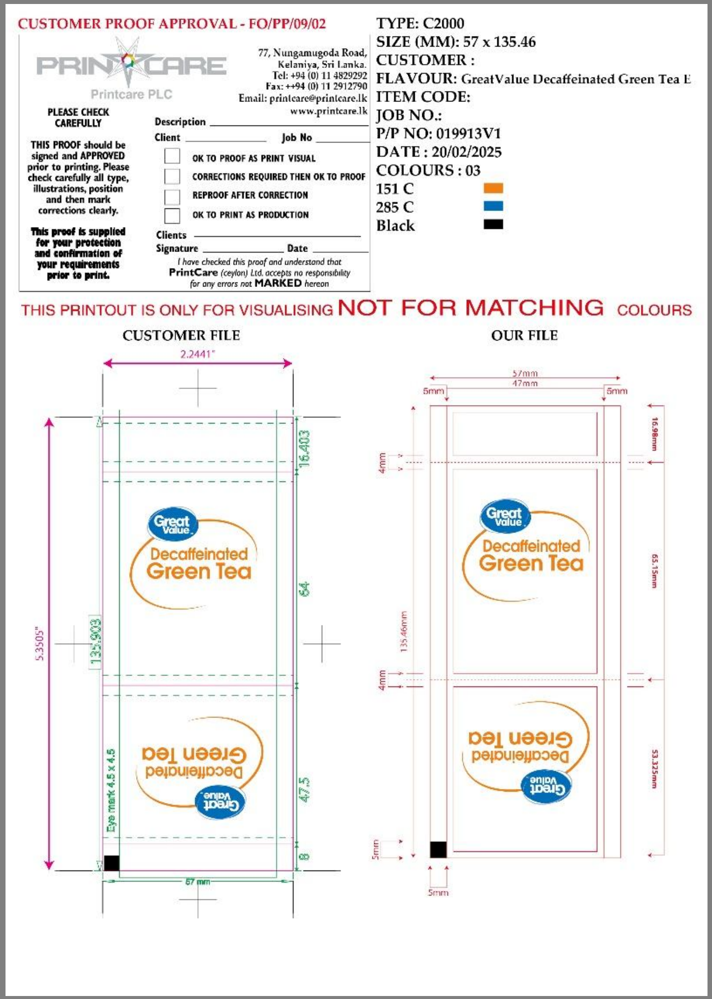
INTRODUCTION

Revolutionizing Flexo Printing: Harnessing AI to Reduce Waste, Cut Costs, and Optimize Plate Reuse.

Flexographic printing is essential in packaging, but manual plate reuse is inefficient and costly. This research leverages AI for artwork similarity detection and automated plate tracking to reduce waste, cut costs, and improve efficiency (1)

WHAT IS THE ARTWORK?





The number of polymer plates used for printing should be determined by the colors in the artwork.

OBJECTIVES

Harnessing AI to revolutionize flexo printing—minimizing waste, lowering costs, and enhancing efficiency.

AI-Based Artwork Similarity Detection

Automated Plate Tracking

Cost & Waste Reduction

Operational Efficiency

OBJECTIVES

Harnessing AI to revolutionize flexo printing—minimizing waste, lowering costs, and enhancing efficiency.

AI-Based Artwork Similarity Detection

Automated Plate Tracking

Cost & Waste Reduction

Operational Efficiency

DEEP LEARNING ARCHITECTURE

OCR-Based Specification Extraction - Duty: Extract technical

```python
import pytesseract
import re

def extract_specifications(image):
    # Multiple OCR configurations for accuracy
    text = pytesseract.image_to_string(image, config='--psm 6')

    # Extract PP number using regex
    pp_pattern = r'P/P\s+NO\s*[:\-]\s*([A-Z]\d+[-_]\d{4}[-_][A-Z]\d+)'
    pp_number = re.search(pp_pattern, text)

    # Extract dimensions
    size_pattern = r'SIZE\s*\(MM\)\s*[:\-]\s*(\d+\.?\d*)\s*x\s*(\d+\.?\d*)'
    dimensions = re.search(size_pattern, text)

    # Extract color count
    color_pattern = r'COLOURS?\s*[:\-]\s*(\d+)'
    colors = re.search(color_pattern, text)

    return {
        'pp_number': pp_number.group(1) if pp_number else None,
        'width': float(dimensions.group(1)) if dimensions else None,
        'height': float(dimensions.group(2)) if dimensions else None,
        'colors': int(colors.group(1)) if colors else None
    }
```

- pytesseract.image_to_string: Reads text from image using OCR
- Regex patterns (pp_pattern, size_pattern, color_pattern): Extract industry-specific info.
- Returns structured dictionary (pp_number, width, height, colors).
- **Why important?** This captures *exact technical details* that a CNN cannot recognize (e.g., plate numbers).

CNN-Based Visual Analysis

Duty: Extract deep **visual features** (logos, patterns, designs).

```python
def extract_visual_features(image):
    # Load pre-trained VGG16 model
    model = VGG16(weights='imagenet', include_top=False, pooling='avg')

    # Preprocess image
    image_224 = cv2.resize(image, (224, 224))
    image_array = np.expand_dims(image_224, axis=0)

    # Extract CNN features
    cnn_features = model.predict(image_array, verbose=0).flatten()

    # Traditional CV features
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    hist_h = cv2.calcHist([hsv], [0], None, [50], [0, 180])
    hist_s = cv2.calcHist([hsv], [1], None, [50], [0, 256])
    edges = cv2.Canny(gray, 50, 150)
    edge_density = np.sum(edges > 0) / edges.size

    # Combine deep + traditional features
    combined_features = np.concatenate([
        cnn_features,
        hist_h.flatten(),
        hist_s.flatten(),
        [edge_density]
    ])
```

- VGG16: Pre-trained CNN extracts **design patterns**.
- hist_h & hist_s: Color histograms (hue & saturation info).
- edge_density: Captures **line/outline details**.
- np.concatenate: Combines deep learning features + classic computer vision features.
- **Why important?** Captures what OCR misses — logos, fonts, layouts, background graphics.

Dual-Approach Similarity Calculation

Duty: Combine OCR and visual features into one **similarity score**.

```
def calculate_similarity(new_artwork, database_artwork):
    # Approach 1: OCR similarity
    ocr_sim = calculate_ocr_similarity(
        new_artwork['specs'],
        database_artwork['specs']
    )

    # Approach 2: Visual similarity
    visual_sim = cosine_similarity(
        [new_artwork['visual_features']],
        [database_artwork['visual_features']]
    )[0][0]

    # Weighted combination
    overall_similarity = (ocr_sim * 0.4) + (visual_sim * 0.6)

    return {
        'overall': overall_similarity,
        'ocr': ocr_sim,
        'visual': visual_sim
    }
```

```
def calculate_ocr_similarity(specs1, specs2):
    score = 0.0

    # PP number exact match
    if specs1['pp_number'] == specs2['pp_number']:
        score += 0.5

    # Size similarity (±2mm tolerance)
    if specs1['width'] and specs2['width']:
        if abs(specs1['width'] - specs2['width']) <= 2 and abs(specs1['height'] - specs2['height']
            score += 0.4

    # Color similarity
    if specs1['colors'] == specs2['colors']:
        score += 0.1

    return score
```

- calculate_ocr_similarity: Matches PP number, size, and colors.
- cosine_similarity: Compares deep visual features numerically.
- Weighted average → **60% visual + 40% OCR**.

Training & Database Building

Duty: Process dataset ⟶ store OCR + visual features ⟶ build database.

```
def train_dual_approach_model(artwork_dataset):
    artwork_database = []

    for image_path in artwork_dataset:
        image = cv2.imread(image_path)

        # Extract both feature sets
        ocr_specs = extract_specifications(image)
        visual_features = extract_visual_features(image)

        artwork_record = {
            'filename': os.path.basename(image_path),
            'ocr_specs': ocr_specs,
            'visual_features': visual_features
        }

        artwork_database.append(artwork_record)

    # Build searchable matrix
    visual_matrix = np.array([art['visual_features'] for art in artwork_database])

    return artwork_database, visual_matrix
```

- Loops over dataset ⟶ extracts OCR & CNN features.
- Saves each record with filename + features
- Builds visual_matrix (used for fast similarity search with cosine/FAISS).
- **Why important?** Turns raw dataset into **structured searchable index**.

Finding Similar Artworks

Duty: Query new artwork → find closest matches.

```python
def find_similar_artworks(new_artwork, database, top_k=10):
    similarities = []

    for db_artwork in database:
        sim_score = calculate_similarity(new_artwork, db_artwork)
        similarities.append((db_artwork, sim_score))

    # Sort by highest overall similarity
    similarities.sort(key=lambda x: x[1]['overall'], reverse=True)

    return similarities[:top_k]
```

- Compares new artwork against entire database.
- Uses dual-approach similarity.
- Returns **top K similar artworks** ranked by similarity.
- **Why important?** Suggests which **polymer plates can be reused**.

System Evaluation

Duty: Measure system accuracy (validation for research).

```
def evaluate_system_performance(test_dataset, ground_truth):
    correct_predictions = 0
    total_predictions = len(test_dataset)

    for test_artwork in test_dataset:
        similar_artworks = find_similar_artworks(test_artwork, database)

        # Top result check
        if similar_artworks[0][0]['filename'] in ground_truth[test_artwork['filename']]:
            correct_predictions += 1

    accuracy = correct_predictions / total_predictions
    return {
        'accuracy': accuracy
    }
```

- Compares system's top prediction with **ground truth labels**.
- Calculates accuracy (later can add precision/recall/F1).
- **Why important?** Provides **scientific validation** for research panel.

Key Technologies Summary

Component	Technology	Purpose
OCR Engine	Pytesseract	Text extraction from images
CNN Model	VGG16 (TensorFlow/Keras)	Deep visual feature extraction
Pattern Matching	Regex	Industry-specific text patterns
Similarity Calculation	Cosine Similarity (Scikit-learn)	Feature vector comparison
Image Processing	OpenCV	Traditional computer vision

Key Points

Dual-Approach Concept

Approach 1: OCR extracts exact specifications (PP numbers, dimensions, colors)
Approach 2: CNN analyzes visual patterns and design elements

Core Technologies

TensorFlow/Keras VGG16: Pre-trained CNN for visual analysis
Pytesseract: Multi-configuration OCR engine
OpenCV: Traditional computer vision features
Scikit-learn: Cosine similarity calculations

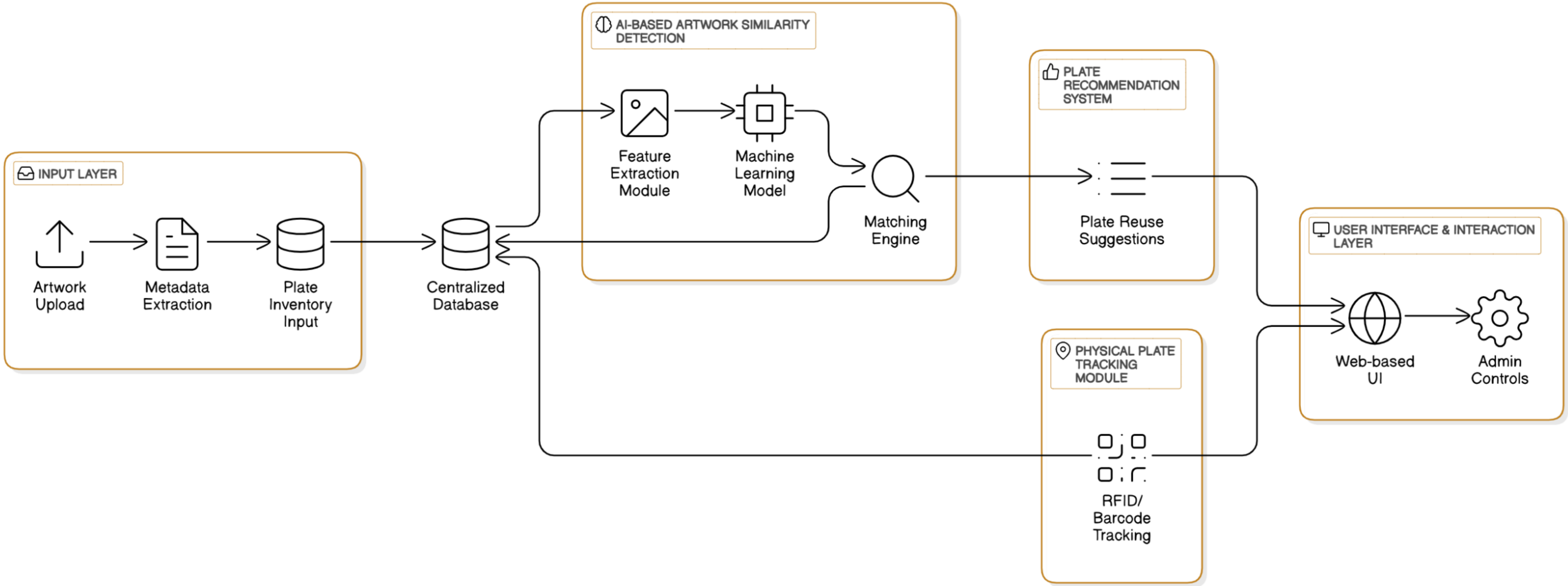
Integration Strategy

Weighted combination: 60% visual + 40% OCR
Handles both quantitative specs and qualitative design similarity
Multi-modal feature fusion

Research Innovation

First dual-approach system for flexographic printing
Combines deep learning with industrial pattern recognition
Scalable for manufacturing databases

OVERALL SYSTEM DIAGRAM



COMMERCIALIZATION PLAN

Basic Plan
Free

✓ Limited artwork similarity detection for up to 5 artwork uploads per month

✓ Basic plate tracking with manual entry

✓ View basic plate reuse suggestions without history tracking

Small businesses exploring AI for plate reuse.

Individual operators seeking basic tracking tools.

Premium Plan
($15/month)

✓ Unlimited artwork similarity detection using AI-powered analysis

✓ Automated plate tracking with barcode/RFID integration

✓ Advanced plate reuse recommendations with location and usage history

✓ AI-driven pattern recognition for better matching accuracy

Mid-sized businesses aiming for automated plate optimization.

Production managers needing detailed tracking and reports. Agency FB

Enterprise Plan
($150/month)

✓ All features from the Premium Plan

✓ Multi-user access for up to 20 users

✓ Centralized dashboard for managing multiple printing jobs and plate inventories

✓ Integration support for existing ERP or production systems

✓ Custom AI model training for industry-specific artwork styles

Large companies managing high-volume artwork and plates.

Corporations needing multi-user access and system integration.

IT21291500 | Premajayantha W.H.S.I.

Specialization: Information technology

INTRODUCTION

- Flexographic printing is essential in packaging, but manual plate reuse is ineffient and costly [2]
- Relies on operator experience, causing inconsistencies and unnecessary plate creation.
- Loss of knowledge when experienced operators leave or resign, creating a knowledge gap
- Time-consuming training for new operators, leading to further inefficiencies.
- High potential for mistakes, increasing material waste, production costs, and environmental impact.

CHALLENGES

- Polymer plates used in flexo printing are not environmentally friendly and contribute to waste
- Storage challenges—large amounts of plates require significant space to store properly.
- Tracking difficulty—managing and locating specific plates in a large inventory is time-consuming
- Job setup delays—finding the correct plates when loading a job into the machine can be complex.
- Duplicate plates issue—if multiple plates exist for the same design, identifying the correct one is difficult.

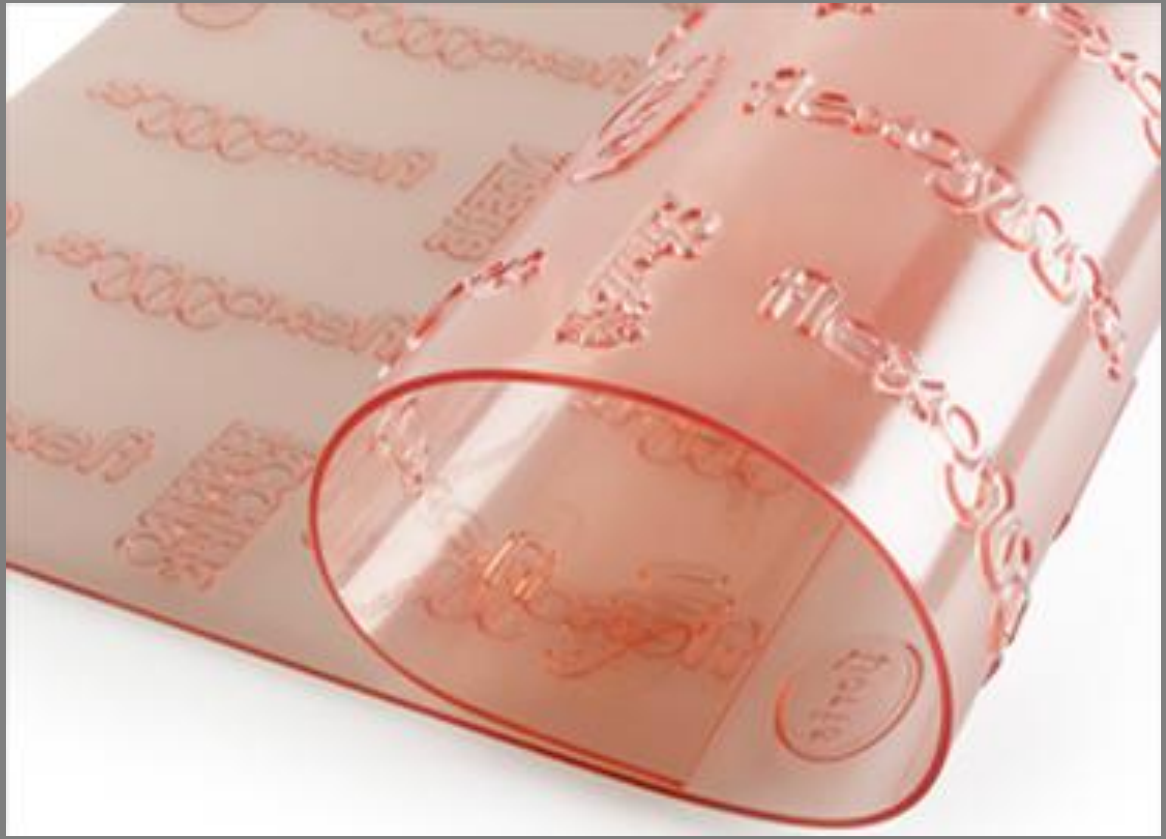
RESEARCH GAP

Component	[3]	[4]	[5]	[6]	Proposed System
AI-Based Artwork Similarity Detection	✗	✗	✓	✗	✓
Automated Plate Tracking (RFID/Barcode)	✗	✗	✗	✗	✓
Centralized Database for Plate Reuse	✗	✓	✗	✗	✓
Real-Time Plate Reuse Recommendation	✗	✗	✗	✗	✓
Knowledge Retention Without Operator Dependency	✗	✗	✗	✗	✓
Cost & Waste Reduction in Flexo Printing	✗	✗	✓	✗	✓

[3] **2022** - Study on AI applications in packaging industry automation.
[4] **2023** - Review of plate management challenges in flexographic printing.
[5] **2021** - Analysis of cost and material waste in flexo printing operations.
[6] **2024** - RFID and barcode tracking systems in industrial printing.

RESEARCH PROBLEM

How can we leverage AI-driven artwork similarity detection and automated plate tracking to optimize polymer plate reuse in flexographic printing, reducing material waste, production costs, and operational inefficiencies while improving sustainability?



OBJECTIVES

- **AI-Driven Artwork Similarity Detection**
  Automates plate reuse identification, reducing manual effort and costs.
- **Automated Knowledge Retention**
  Eliminates reliance on operator memory, ensuring consistent plate management even if experienced staff leave.
- **Efficient Plate Reuse & Waste Reduction**
  Identifies reusable plates to minimize material waste and unnecessary plate creation.

Real-Time Plate Tracking

Uses RFID/barcodes to track plate location and history, improving inventory management.

Faster Onboarding & Reduced Training Time

User-friendly system helps new operators adapt quickly, reducing disruptions.

Sustainability & Cost Savings

Optimizes resources, cutting costs while supporting eco-friendly practices in flexo printing.

TECHNOLOGIES

TensorFlow

JS

multer

React

Python

git

GitHub

Open CV

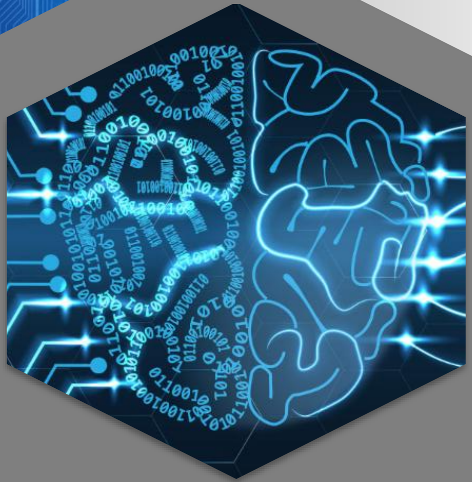
OpenCV

mongoDB GridFS

Machine Learning

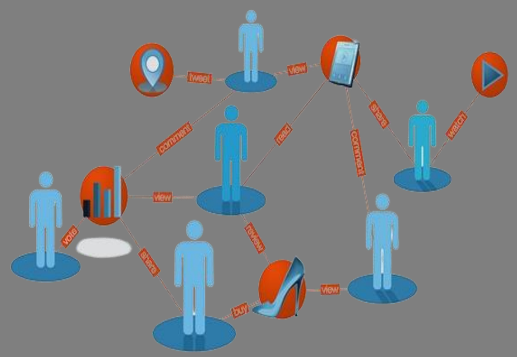
Automation & Tracking

Image Processing



KEY PILLARS

FUNCTIONAL REQUIREMENTS



- Data Collection and Processing
- AI-Powered Artwork Similarity Detection
- Adaptive Plate Identification & Tracking
- Interactive Plate Selection & Recommendation System
- User Interface & Experience

NON-FUNCTIONAL REQUIREMENTS

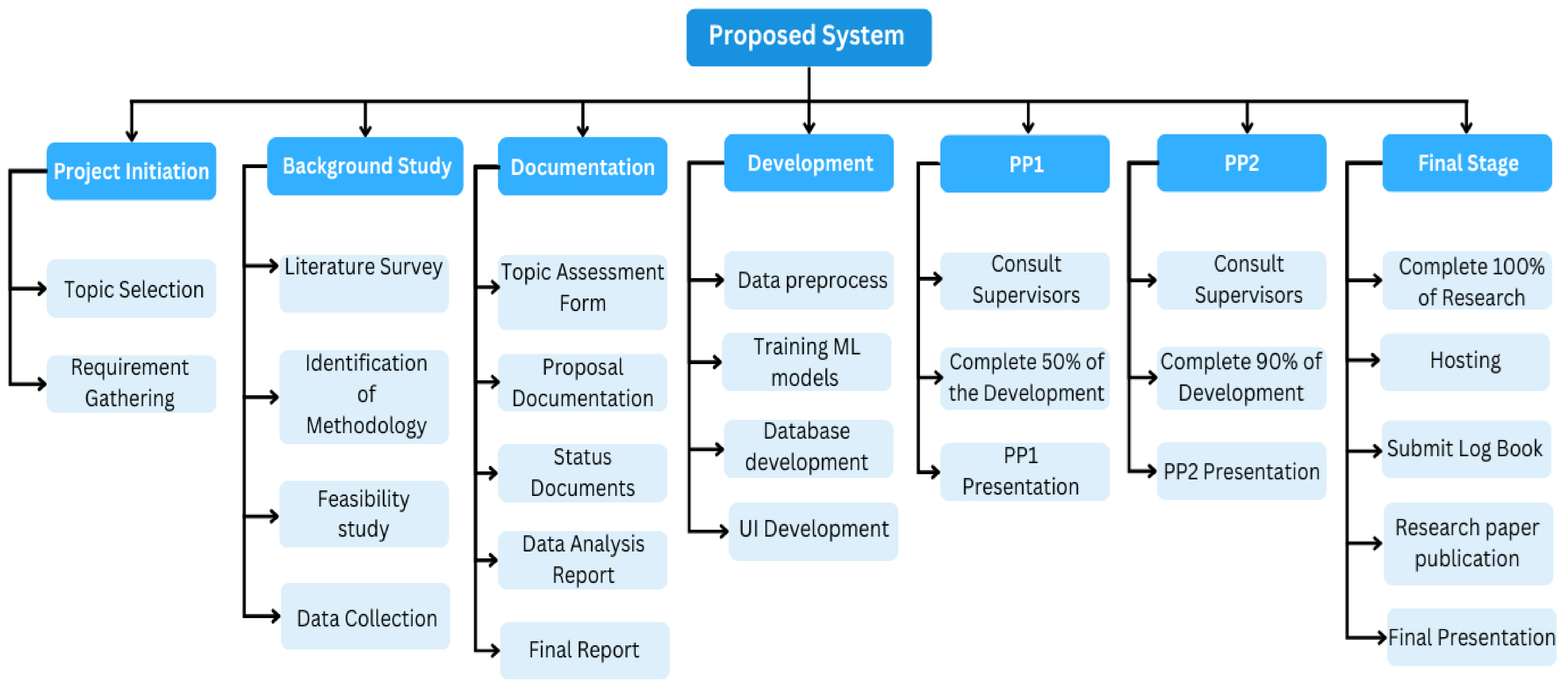
Performance

Reliability

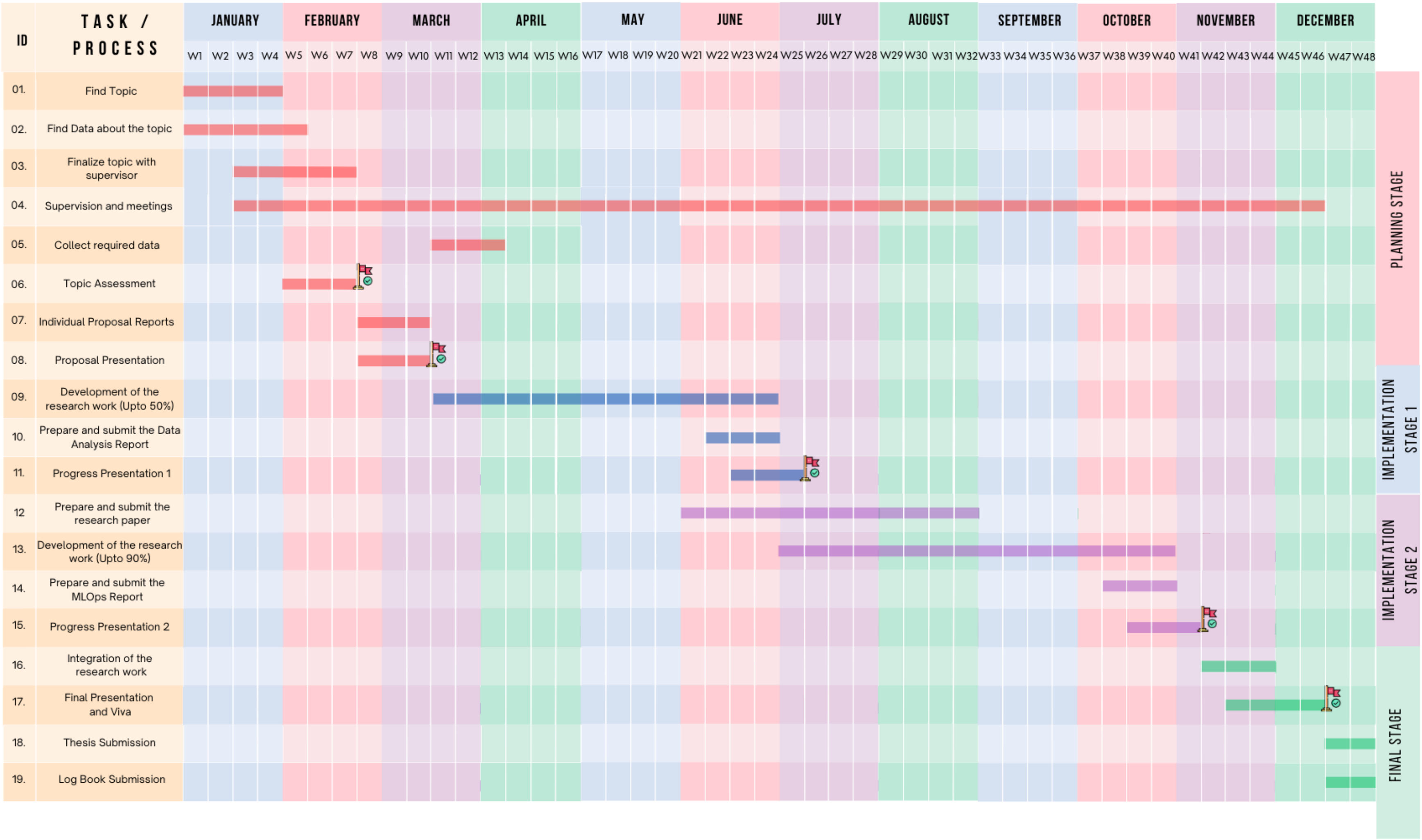
Usability

Security

WORK BREAKDOWN CHART



GANTT CHART


ID
TASK / PROCESS
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER
W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14 W15 W16 W17 W18 W19 W20 W21 W22 W23 W24 W25 W26 W27 W28 W29 W30 W31 W32 W33 W34 W35 W36 W37 W38 W39 W40 W41 W42 W43 W44 W45 W46 W47 W48
01. Find Topic
02. Find Data about the topic
03. Finalize topic with supervisor
04. Supervision and meetings
05. Collect required data
06. Topic Assessment
07. Individual Proposal Reports
08. Proposal Presentation
09. Development of the research work (Upto 50%)
10. Prepare and submit the Data Analysis Report
11. Progress Presentation 1
12 Prepare and submit the research paper
13. Development of the research work (Upto 90%)
14. Prepare and submit the MLOps Report
15. Progress Presentation 2
16. Integration of the research work
17. Final Presentation and Viva
18. Thesis Submission
19. Log Book Submission
PLANNING STAGE
IMPLEMENTATION STAGE 1
IMPLEMENTATION STAGE 2
FINAL STAGE

REFERENCES

[1] J. Smith and R. Kumar, "AI applications in packaging industry automation: A comprehensive study," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4561–4573, Jul. 2022.

[2] A. Johnson and M. Lee, "Review of plate management challenges in flexographic printing," *Journal of Printing Science and Technology*, vol. 35, no. 4, pp. 215–227, 2023.

[3] J. Smith and R. Kumar, "AI applications in packaging industry automation: A comprehensive study," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4561–4573, Jul. 2022.

[4] A. Lee and M. Fernandez, "Challenges in plate management for flexographic printing: A systematic review," *Journal of Packaging Science and Technology*, vol. 12, no. 3, pp. 112–125, Mar. 2023.

[5] K. Johnson and P. Wang, "Cost analysis and material waste reduction in flexo printing operations," *International Journal of Printing Technology*, vol. 29, no. 5, pp. 325–338, Oct. 2021.

[6] D. Brown and L. Carter, "RFID and barcode-based tracking systems in industrial printing," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 1, pp. 134–145, Jan. 2024.