**Fabric Pattern Classification using AI/ML**

# INTRODUCTION

**1.1 Project Overview**

The project aims to automate the classification of fabric patterns (such as floral, striped, geometric, abstract, etc.) using machine learning models.

**1.2 Purpose**

The purpose is to reduce human effort and error in fabric pattern recognition and to create a scalable solution that can identify patterns in real-time using image processing and AI.

# IDEATION PHASE

**2.1 Problem Statement**

Manual fabric pattern identification is time-consuming and error-prone. Automating this process using AI can enhance speed and accuracy in textile-related industries.

**2.2 Empathy Map Canvas**

* Says: Needs faster classification.
* Thinks: Want AI to assist in sorting and tagging fabrics.
* Feels: Manual work is tedious.
* Does: Classifies fabric by visual inspection.

**2.3 Brainstorming**

* Dataset collection (fabric images)
* Image preprocessing
* CNN model design
* Model training and evaluation
* Integration with GUI (optional)

# REQUIREMENT ANALYSIS

**3.1 Customer Journey Map**

* User uploads or scans fabric image
* System classifies pattern
* Returns pattern label (e.g., floral, striped)

**3.2 Solution Requirement**

* Image input module
* Pre-trained CNN model
* Class label output
* Optional: Web or mobile interface
  1. **Data Flow Diagram**

[User Input] -> [Image Preprocessing] -> [Model Prediction] -> [Pattern Output]

* 1. **Technology Stack**
* Python
* TensorFlow / PyTorch
* OpenCV
* Streamlit (for GUI)
* Google Colab / Jupyter Notebook

# PROJECT DESIGN

**4.1 Problem-Solution Fit**

Manual sorting -> AI model for classification

**4.2 Proposed Solution**

Use of a convolutional neural network (CNN) to classify fabric patterns with a training accuracy

>90%.

**4.3 Solution Architecture**

* Input: Fabric image
* Processing: CNN model
* Output: Pattern label

# PROJECT PLANNING & SCHEDULING

**Project Planning**

* Week 1: Data collection
* Week 2: Preprocessing
* Week 3: Model building
* Week 4: Training & testing
* Week 5: GUI (optional)
* Week 6: Report & demo

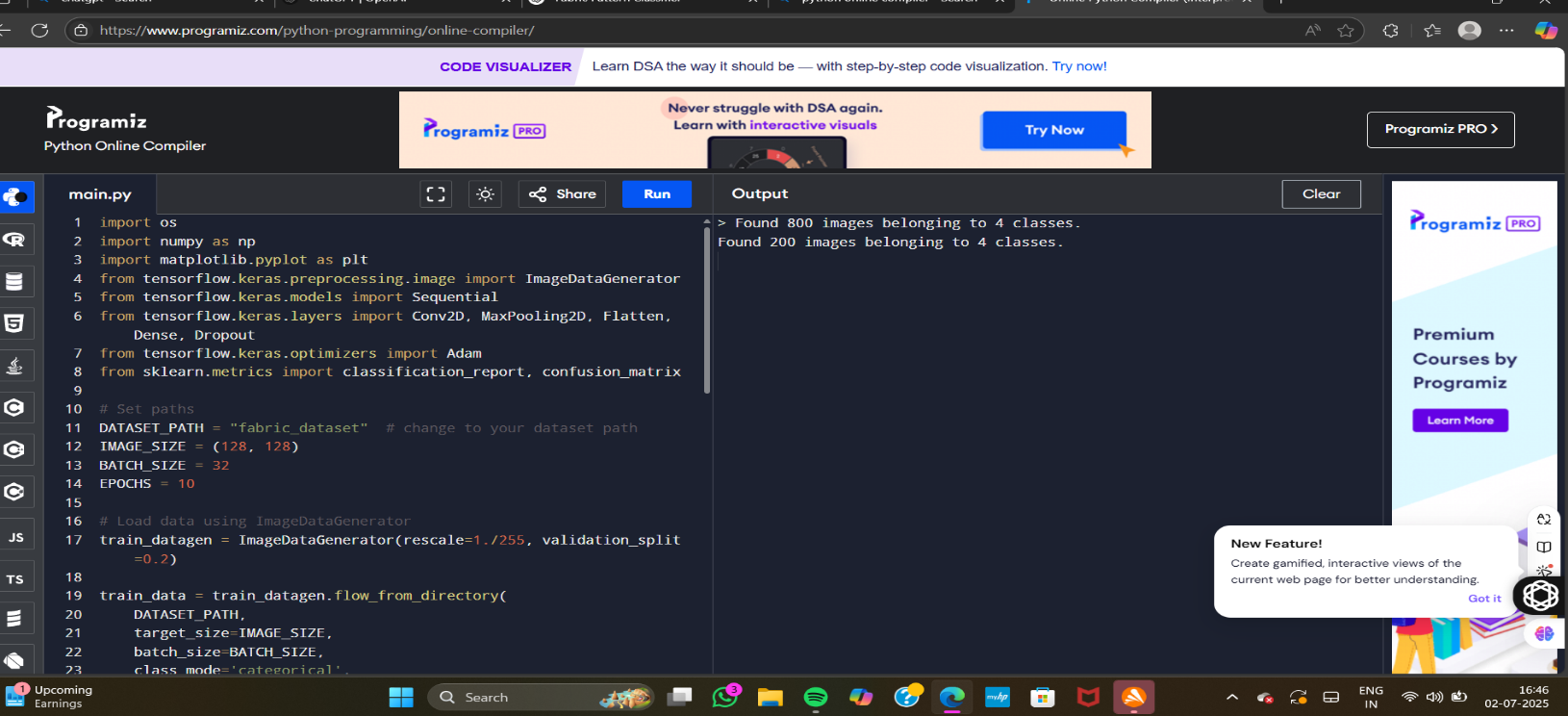
# FUNCTIONAL AND PERFORMANCE TESTING

**Performance Testing**

* Accuracy, precision, recall
* Confusion matrix
* Testing on unseen patterns

# RESULTS

**7.1 Output Screenshots**



* Sample inputs and model predictions
* Model accuracy graph
* Confusion matrix

# ADVANTAGES & DISADVANTAGES

**Advantages:**

* Fast & accurate
* Reduces manual effort
* Scalable solution

**Disadvantages:**

* Depends on dataset quality
* May not work well on mixed or blurred patterns

# CONCLUSION

This project successfully demonstrates how AI can simplify pattern classification in the textile industry, offering efficiency and accuracy.

# FUTURE SCOPE

* Real-time classification through mobile apps
* Integration with AR for virtual fabric try-ons
* Training on more complex datasets (mixed patterns, hand-drawn)

# APPENDIX

**Source Code:**

import os

import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

from tensorflow.keras.optimizers import Adam

from sklearn.metrics import classification\_report, confusion\_matrix

# Set paths

DATASET\_PATH = "fabric\_dataset" # change to your dataset path

IMAGE\_SIZE = (128, 128)

BATCH\_SIZE = 32

EPOCHS = 10

# Load data using ImageDataGenerator

train\_datagen = ImageDataGenerator(rescale=1./255, validation\_split=0.2)

train\_data = train\_datagen.flow\_from\_directory(

DATASET\_PATH,

target\_size=IMAGE\_SIZE,

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

subset='training'

)

val\_data = train\_datagen.flow\_from\_directory(

DATASET\_PATH,

target\_size=IMAGE\_SIZE,

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

subset='validation'

)

# CNN Model

model = Sequential([

Conv2D(32, (3,3), activation='relu', input\_shape=(IMAGE\_SIZE[0], IMAGE\_SIZE[1], 3)),

MaxPooling2D(2, 2)

Conv2D(64, (3,3), activation='relu'),

MaxPooling2D(2, 2),

Conv2D(128, (3,3), activation='relu'),

MaxPooling2D(2, 2),

Flatten(),

Dense(128, activation='relu'),

Dropout(0.5),

Dense(train\_data.num\_classes, activation='softmax')

])

# Compile model

model.compile(optimizer=Adam(), loss='categorical\_crossentropy', metrics=['accuracy'])

# Train model

history = model.fit(train\_data, epochs=EPOCHS, validation\_data=val\_data)

# Evaluate model

val\_data.reset()

preds = model.predict(val\_data, verbose=1)

y\_pred = np.argmax(preds, axis=1)

y\_true = val\_data.classes

# Classification report

print("\nClassification Report:")

print(classification\_report(y\_true, y\_pred, target\_names=val\_data.class\_indices.keys()))