ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

Project Documentation format

# Introduction

* + **Project Title: “**Pattern Sense : Classifying Fabric patterns using Deep Learning.

# Project Overview

* + **Purpose:** To automate the identification and classification of fabric patterns using deep learning.The project is designed to help industries like fashion,textiles and interior design by speeding up the pattern recognition process , reducing human error and enabling smart sorting and categorization of fabric types.
  + **Features:** a. Automated Pattern Recognition

b. Multi-Class Classification

c. User-friendly Interface

d. High Accuracy Model

# Architecture

* + **Frontend:** HTML,CSS
  + **Backend:** Python , OpenCV , CNN
  + **Database:** SQLite/MYSQL

# Setup Instructions

* + **Prerequisites:** Python , TensorFlow , Keras , Flask , Numpy , Pandas ,Scikit-learn.
  + **Installation:**

i.Clone the Repository :

git clone <https://github.com/LalithaGolla05/Pattern-Sense.git>

ii. Create a Virtual Environment :

python -m venv venv

Venv\Scripts\activate

iii. Install Dependencies :

pip install -r requirements.txt

iv. Setup the Environment Variables :

touch .env

v. Run the Application :

Flask run

# Folder Structure

* + **Client:** It contains the user interface (index.html , style ) in the frontend.
  + **Server:** It loads the deep learning model , process the uploaded image and returns the predicted class and communictes with the frontend using Flask routes(app.py) at the backend.

# Running the Application

Install dependencies , setup the environment and run the application using flask.

# Authentication

# Authentication : Verifies who the user is (identity)

# User Registration

# User Login

# Session Management

# 

# Authorization : Determines what the user can do (permissions)

# Use Flask-Login’s @login\_required decorator to protect sensitive routes.

# User Interface

# The Pattern Sense UI is a clean , simple , and user-friendly web interface designed allow to upload fabric images and view the predicted pattern type.

# Testing

# Model Testing(Deep Learning Pipeline)

# Tools Used :

# Python

# TensorFlow

# Scikit-learn

# Matplotlib/Seaborn

# Backend Testing(Flask API)

# Tools Used :

# Flask-Testing

# Frontend Testing(HTML- based Client)

# Tools Used :

# Manual Testing

# 

# Screenshots

# 

# WhatsApp Image 2025-06-27 at 4.37.21 PM

# WhatsApp Image 2025-06-27 at 4.39.54 PM

# Known Issues

# Limited Dataset Diversity

# Model Overfitting on Small Datasets

# Prediction Latency

# Error Handling for Invalid Inputs

# Authentication Token Expiry(if implemented)

# No Offline Functionality

# Frontend Compatability

# Lack of Real-Time Training Feedback

# Future Enhancements

1. Dataset Expansion
2. Mobile App Integration
3. Offline Prediction Mode
4. Pattern Detection and Segmentation
5. Pattern Similarity Search
6. User Feedback Loop
7. Multi-language Support
8. Voice-Based Interface