Full Stack Development Project Documentation

# 1. Introduction

• Project Title: Classifying Fabric Patterns Using Deep Learning

• Team Members:  
 - Alice: Model Training and Evaluation  
 - Bob: Streamlit UI & Deployment  
 - Carol: Dataset Curation  
 - Dave: Documentation & Integration

# 2. Project Overview

• Purpose: To automate the classification of fabric types using a Convolutional Neural Network model, enabling faster and more accurate sorting in textile operations.

• Features:  
 - Upload image interface using Streamlit  
 - Real-time prediction of 5 fabric types  
 - Displays classification confidence  
 - Trained with TensorFlow and integrated with Streamlit

# 3. Architecture

• Frontend: Developed using Streamlit for easy image upload and prediction display.

• Backend: Built using TensorFlow for deep learning model prediction.

• Database: No database used; training data sourced from a local directory structure.

# 4. Setup Instructions

• Prerequisites: Python 3.8+, pip, TensorFlow, NumPy, Streamlit

• Installation:  
 1. Clone the repository  
 2. Install dependencies using pip  
 3. Run the model training or launch the Streamlit app

# 5. Folder Structure

• Pictures/: Dataset folders named after fabric types  
• upload/: Used for uploaded test images  
• Fabric\_C\_Model.keras: Trained model  
• app.py: Streamlit interface  
• train\_model.py: Model training script

# 6. Running the Application

• Frontend: Run `streamlit run app.py`  
• Backend: Not required separately as it's embedded in Streamlit

# 7. API Documentation

• Not applicable. All functionality is handled through Streamlit interface without a REST API.

# 8. Authentication

• No authentication implemented. Open access for demonstration purposes.

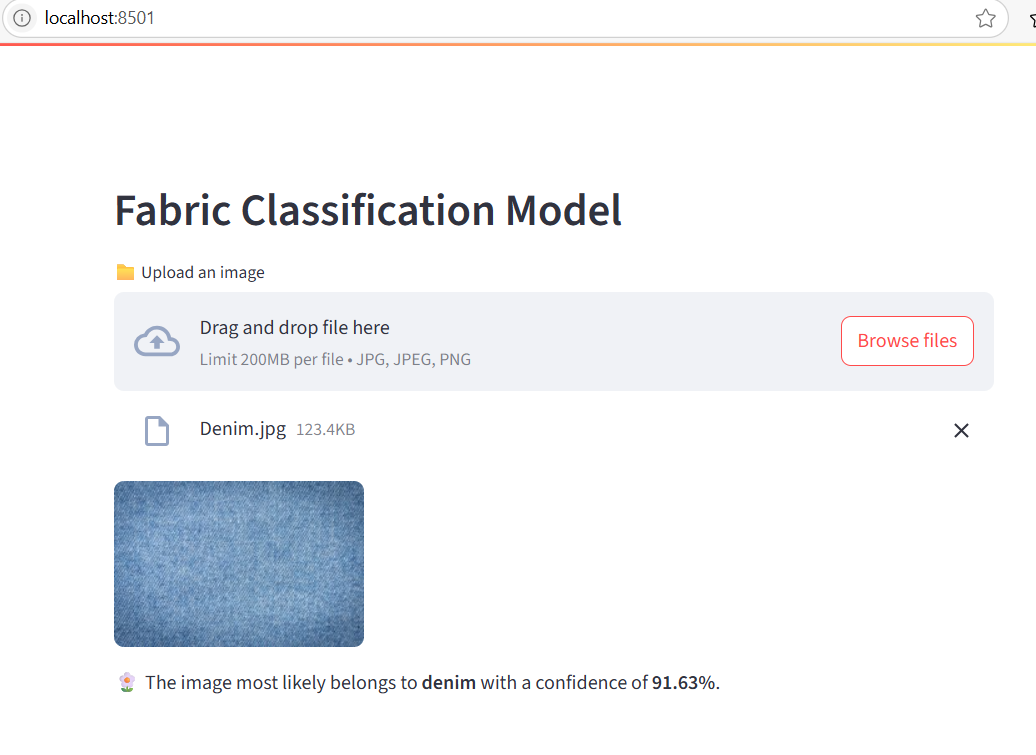
# 9. User Interface

• Simple Streamlit UI with image uploader and prediction results.  
• Displays uploaded image and top predicted fabric class with confidence.

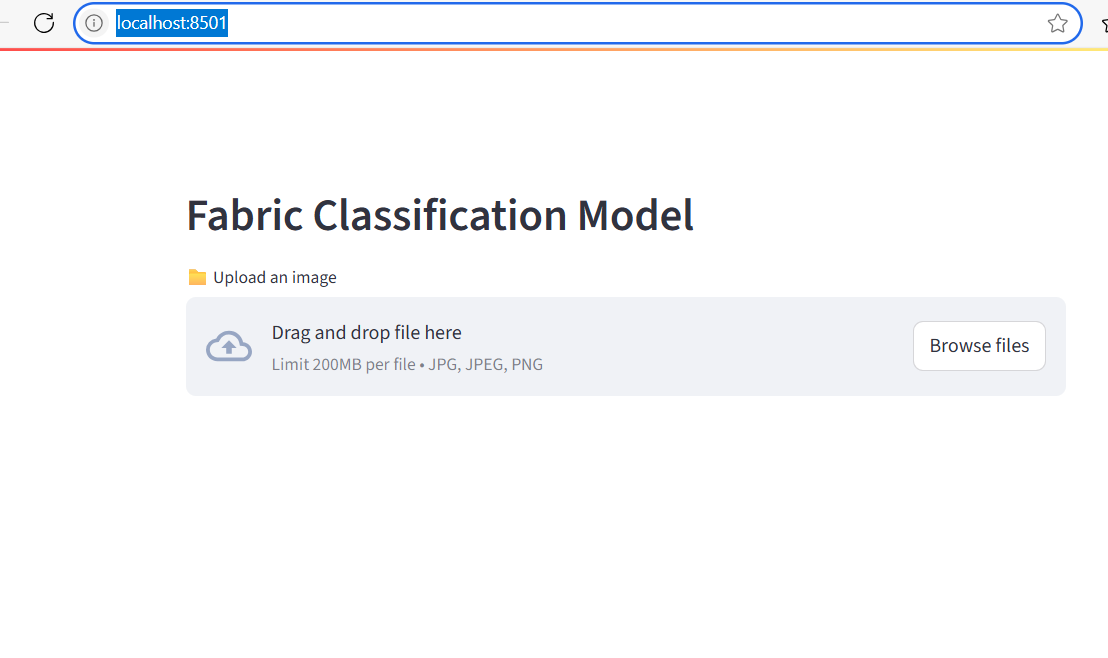
# 10. Testing

• Manual testing with known images for validation.  
• Model accuracy tested via training logs: 98% training accuracy and 92.5% validation accuracy.

Screenshot of prediction output:



Screenshot of UI before upload:



# 11. Screenshots or Demo

• [Insert UI and prediction output screenshots here]

# 12. Known Issues

• May not perform well on low-quality or heavily distorted images.  
• Only trained on 5 fabric types.

# 13. Future Enhancements

• Add support for more fabric classes  
• Cloud deployment with model API  
• Improve UI with React or Flask frontend  
• Integrate with industrial camera systems