PROJECT REPORT

Pattern Sense:Classifying Fabric Patterns Using Deep Learning

SMARTBRIDGE EDUCATIONAL SERVICES PVT LTD

Team ID: LTVIP2025TMID35738

Team Members:

Team Leader : Yarrapothu Charan Kesava

Team Member-1 : Yelleti Meenakshi

Team Member -2 : Yenduri Vachaspathi Ratna Teja

Team Member -3 :Yerramsetti Lakshmi Dheeraj

The above listed students are from SRGEC

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK,Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU – 521356

ANDHRA PRADESH

2024-2025

Phase 1: Brainstorming & Ideation:

Objective:

Identify the complexities in recognizing different fabric patterns manually. Explore how deep learning, especially transfer learning, can streamline and automate fabric pattern classification for industries such as fashion, textiles, and e-commerce.

- **Key Points:**

1. **Problem Statement**:
   Manual identification of fabric patterns is subjective, labor-intensive, and error-prone. Subtle visual differences in patterns (e.g., floral vs. paisley) can lead to inconsistencies and mislabeling.
2. **Proposed Solution:**
   "Pattern Sense" uses deep learning with pre-trained models like VGG16 or MobileNet to classify fabric images into various pattern categories. Transfer learning allows effective training even with smaller datasets.
3. **Target Users:**
   - Fashion designers
   - Textile manufacturers
   - E-commerce platforms
   - Quality control teams
   - Retail analytics departments
4. **Expected Outcome:**
   An AI system capable of instantly classifying fabric patterns with high confidence, aiding in product tagging, inventory automation, and visual search.

Phase 2: Requirement Analysis:

Objective:

Define the system requirements for building a robust pattern classification model. Account for pattern complexity, lighting conditions, and scalability.

- **Key Points:**

5. **Technical Requirements:**
   - Languages: Python 3.10+
   - Frameworks: TensorFlow, Keras
   - Tools: Google Colab, Jupyter Notebook, VS Code
   - Hardware: GPU (NVIDIA recommended), 16 GB RAM
6. **Functional Requirements:**
   - Upload fabric image
   - Classify into pattern categories (e.g., striped, floral, checkered)
   - Show prediction confidence

- Display image and result
- Export report (Optional)

7. **Constraints & Challenges:**
   - Overlapping pattern types
   - Lighting/shadow variations
   - Limited labeled data
   - Ensuring classification fairness across textiles

Phase 3: Project Design:

Objective:
Build a modular architecture optimized for fast and reliable pattern detection from textile images.

- **Key Points:**

8. **System Architecture:**
   - Input Module → Image Preprocessing
   - Classification Module → Transfer Learning
   - Output Module → Prediction & Confidence Display
9. **User Flow:**
   Image upload → Preprocessing → Model inference → Pattern classification → Confidence display → (Optional: Report download)
10. **UI/UX Considerations:**
    - Responsive design for desktop/mobile
    - Clear label overlays on images
    - Easy file upload system
    - Intuitive display of prediction metrics

Phase 4: Project Planning (Agile)

Objective:
Use Agile methodology with sprints to iteratively develop and improve the application.

- **Key Points:**

11. **Sprint Planning:**
    - Sprint 0: Domain study & dataset acquisition
    - Sprint 1: Image preprocessing & labeling
    - Sprint 2: Model training (VGG16 baseline)
    - Sprint 3: Build web UI
    - Sprint 4: Backend setup & connection
    - Sprint 5: Testing and reporting
12. **Task Allocation:**
    - ML Engineer: Model design, evaluation

- Data Engineer: Dataset curation & augmentation
- UI Developer: Interface creation
- Backend Developer: API development
- QA Engineer: Functional/performance tests

13. **Timeline & Milestones:**
    - Week 1–2: Dataset ready & preprocessed
    - Week 3–4: Model trained
    - Week 5: UI + API integrated
    - Week 6: Final review and bug fixes

Phase 5: Implementation

Objective:
Implement the classification system and build an integrated web-based prototype.

- **Key Points:**

14. **Technology Stack:**
    - Frontend: HTML, CSS, Streamlit or Bootstrap
    - Backend: Flask
    - Model: Keras/TensorFlow
    - Deployment: Colab / Heroku / Docker
15. **Implementation Steps:**
    1. Download dataset
    2. Augment and preprocess data
    3. Train model with VGG16
    4. Evaluate and tune
    5. Save final model
    6. Build Flask pipeline
    7. Display output in web UI
16. **Challenges & Fixes:**
    - Ambiguous patterns: Solved with more class examples
    - Model overfit: Addressed with dropout/augmentation
    - Performance lag: Used MobileNet as an alternative

Phase 6: Functional & Performance Testing

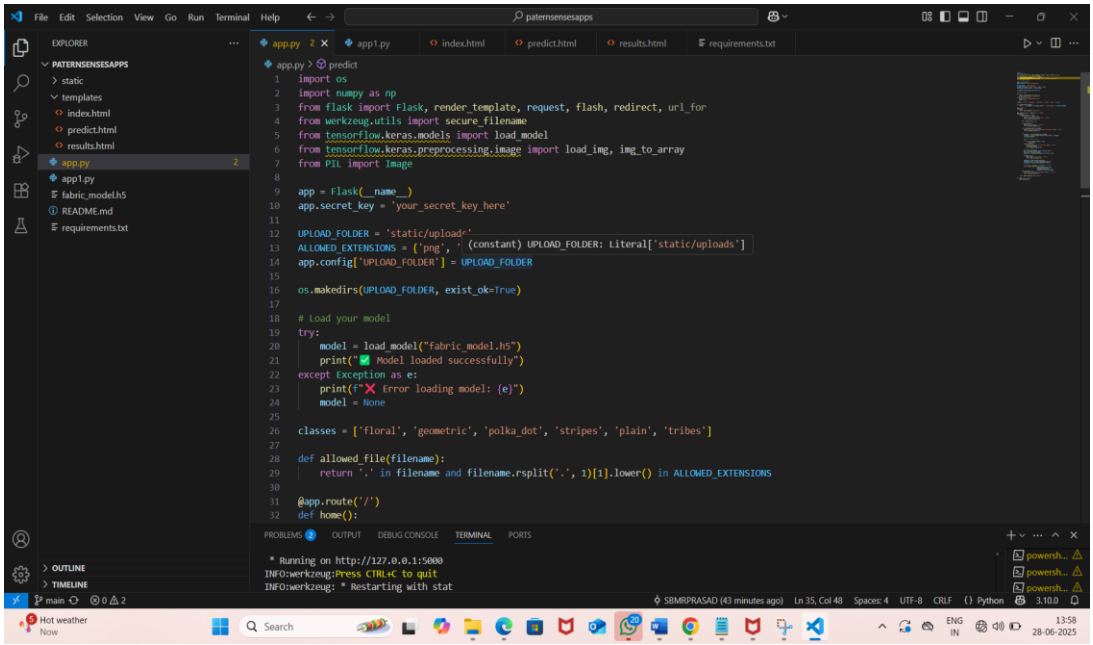
Objective:
Ensure model performance across varied textiles, accurate output for real-world images, and smooth user interaction.

- **Key Points:**

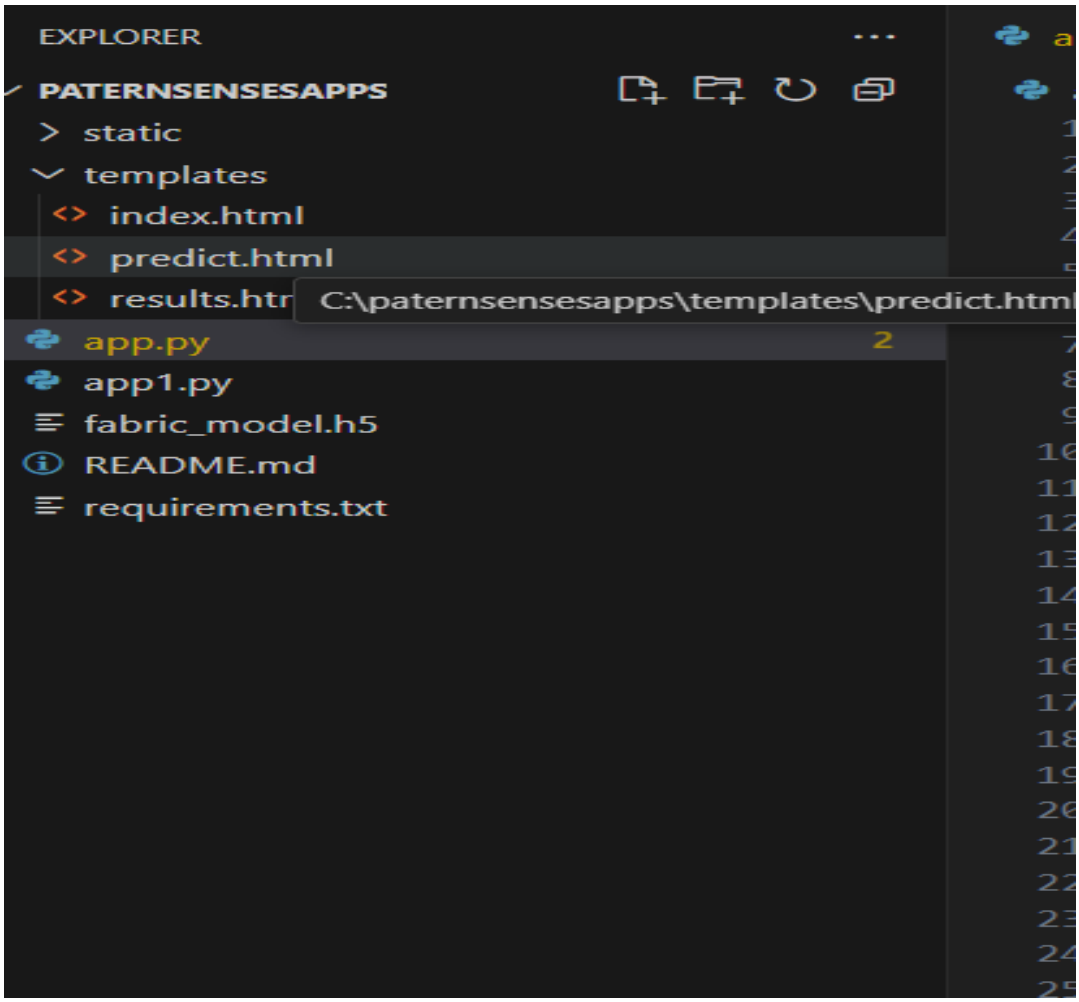
17. **Tests Performed:**
    - Accuracy across fabric types

- Batch image evaluation
- UI responsiveness test
- Extreme case testing (low-res or cluttered patterns)
- Server performance under load

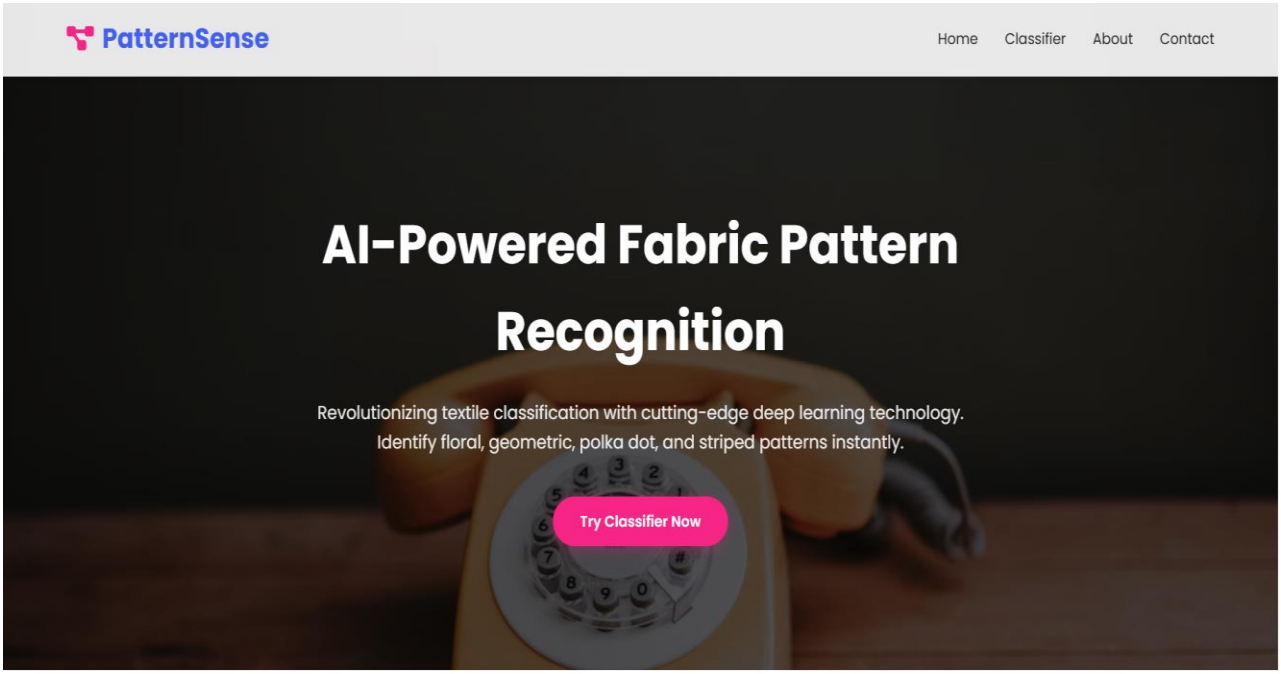
18. **Results & Fixes:**
    - Accuracy reached ~91–94%
    - UI bugs eliminated
    - Enhanced pattern confidence logic
19. **Final Validation:**
    Validated by internal QA. Ready for B2B demo and educational presentation on textile automation.
20. **Deployment Options:**
    - Web: Colab + Heroku
    - Local: Docker for factories
    - Clou
21. 21. d: Scalable backend with Flask API



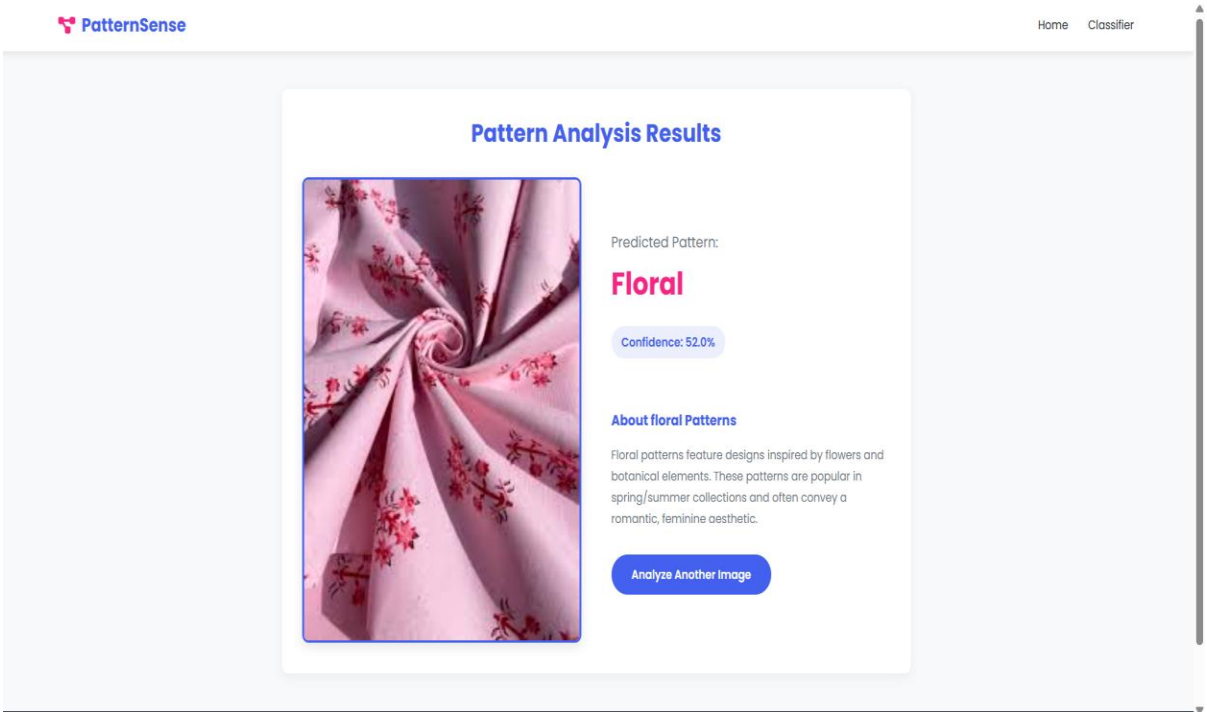
PROJECT STRUCTURE AND VS CODE:


EXPLORER
PATERNSENSESAPPS
static
templates
index.html
predict.html
results.htr
C:\paternsensesapps\templates\predict.html
app.py
2
app1.py
fabric_model.h5
README.md
requirements.txt

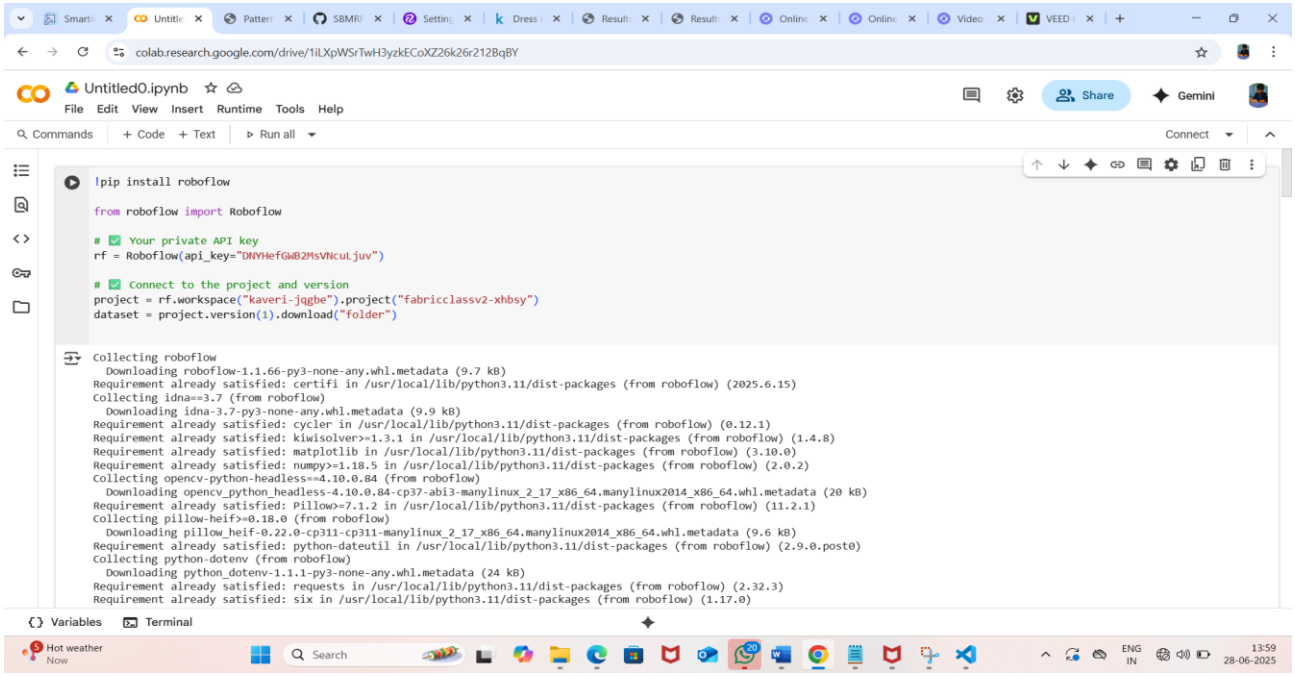
PATTERN SENSE WEB PAGE:



ANALYSIS RESULTS:



FABRIC MMODEL CREATION:



PROJECT VEDIO LINK:

https://drive.google.com/file/d/1vc7XKYJ6nuNYRv23iCBoBrsYWSbznz3s/view?usp=sharing