**Project Phases Template**

**Project Title:**

Pattern Sense: Classifying Fabric Patterns Using Deep Learning

**Team ID:**

LTVIP2025TMID33839

**Team Members:**

* Padamata Nagaraju
* Natta Harika
* Navyaka Matham
* Nimisha Kambham

**Phase-1:Brainstorming & Ideation**

**Objective:**

* Identify the problem statement
* Purpose and impact of the project

**Key points:**

**1.Problem statement:**

The textile and fashion industries lack scalable, automated systems to classify fabric patterns accurately. Manual classification is time-consuming, subjective, and inconsistent, especially in large-scale production or e-commerce cataloging

**2.Proposed Solution:**

**1. Data Collection**

**We plan to build or utilize a diverse dataset consisting of labeled images of different fabric patterns, including:**

* **Floral**
* **Striped**
* **Dotted**
* **Geometric**
* **Plain**
* **Abstract/Other**

**Data sources:**

* **Web scraping from fashion e-commerce sites**
* **Open-source fabric datasets.**

**2. Data Preprocessing**

**To improve model performance and generalization:**

* **Image resizing: Standardized to 224x224 pixels**
* **Normalization: Scale pixel values to [0, 1]**
* **Data augmentation: Apply transformations to increase diversity:**
  + **Rotation**
  + **Horizontal/vertical flips**
  + **Zooming and cropping**
  + **Brightness/contrast adjustments**
* **Label encoding: Convert class names to numerical values**

**3.Target Users:**

**1. Textile Industry Professionals**

* **Who: Fabric manufacturers, quality control teams, textile designers.**
* **Why: To automate fabric pattern inspection and reduce manual classification errors.**
* **Use Case: Rapid classification of fabric rolls during production or packaging.**

**2. E-commerce Platforms**

* **Who: Online clothing/fabric retailers like Amazon, Flipkart, Myntra, etc.**
* **Why: To automate product tagging and improve search relevance.**
* **Use Case: Classifying uploaded product images with appropriate pattern labels.**

**3. Fashion Designers**

* **Who: Designers working in clothing brands or boutiques.**
* **Why: To quickly browse categorized fabric patterns for inspiration or selection.**
* **Use Case: Upload swatch photos and get pattern classification for documentation or styling purposes.**

**4. Educational Institutions and Researchers**

* **Who: Students, research scholars, professors in textile engineering or computer vision.**
* **Why: To explore deep learning use cases in fashion tech.**
* **Use Case: As a research or teaching aid to understand image classification applications.**

**5. Mobile App Users (Future Scope)**

* **Who: Casual users, customers in textile stores, or small-scale tailors.**
* **Why: To identify and compare fabric patterns instantly using a phone.**
* **Use Case: Capture fabric photos and get immediate pattern classification for shopping or design.**

**4.Excepted Outcome:**

**The Pattern Sense project aims to develop an intelligent, automated system that can accurately classify different types of fabric patterns from images using deep learning techniques.**

**Phase-2: Requirement Analysis:**

**Objective:**

* Define technical and functional requirements

**Key Points:**

**1.Technical Requirements:**

**Programming Languages**

* **Python**
  + **For image preprocessing, model training, and backend integration.**
* **HTML**
  + **To build a simple and intuitive front-end web interface for user interaction.**

**🧠 Frameworks & Libraries**

**Data Handling:**

* **NumPy, Pandas**
  + **Used for reading datasets, processing metadata, and handling image-related data structures.**

**Image Processing:**

* **OpenCV**
  + **For loading, resizing, augmenting, and preprocessing fabric images before classification.**

**Deep Learning Frameworks:**

* **TensorFlow / Keras (or PyTorch)**
  + **To build and train the deep learning model (CNN or pre-trained architecture).**

**Model Training & Testing Tools:**

* **Google Colab**
  + **Used for model experimentation, training, and evaluation with GPU acceleration.**
* **Kaggle**
  + **As a source for publicly available fabric pattern datasets or data exploration.**

**Optional: Web Application Backend**

* **Flask / Streamlit**
  + **For integrating the trained model into a lightweight web app.**

**🌐 To Run the Application**

* **Anaconda Command Prompt**
  + **To manage the virtual environment, run Jupyter notebooks or Python scripts, and start the web application server.**

**2.Functional Requirements:**

**🖼️ Image Upload Feature**

* **Users should be able to upload a fabric image through the web interface (JPEG/PNG).**
* **The uploaded image is sent to the backend for preprocessing and prediction.**

**🎨 Fabric Pattern Classification**

* **The system should analyze the uploaded fabric image and classify it into one of the following pattern types:**
  + **🌼 Floral**
  + **🟦 Striped**
  + **🔴 Dotted**
  + **📐 Geometric**
  + **⬜ Plain**
  + **🌀 Abstract**

**🧠 Machine Learning Model Integration**

* **The trained CNN model (or fine-tuned transfer learning model) should:**
  + **Load and preprocess the uploaded image.**
  + **Perform prediction and identify the correct pattern class.**
  + **Return results along with confidence score/probability.**

**🖥️ User Interface (UI)**

* **A clean, simple HTML-based front-end should support:**
  + **Image upload via file picker.**
  + **“Classify” button to trigger prediction.**
  + **Display of:**
    - **Predicted pattern type**
    - **Confidence percentage**
    - **Optional: Display a sample image or description for that pattern type.**
  + **Status messages such as:**
    - **"Uploading image..."**
    - **"Prediction complete!"**
    - **"Invalid image format. Please upload a JPG/PNG**

**Phase-3:Project Design:**

**Objective:**

* Create the architecture and user flow

**Key points:**

**1.System Architecture Diagram:**

**PatternSense/**

**├── model\_cnn.h5**

**├── label\_map.json**

**├── streamlit\_app.py**

**├── requirements.txt**

**└── .streamlit/**

**└── config.toml (optional for theme)**

**2.User Flow:**

**🟢 1. Launch the Web App**

**🖼️ 2. Upload Fabric Image**

**🤖 3. Predict Fabric Pattern**

**🎯 4. Show Results**

**🔁 5. Upload Another Image (Optional)**

**Phase 4: Project Planning(Agile Methodologies):**

**Objective:**

* Break down the tasks using Agile methodologies

**Key Points:**

**1.Sprint Planning &Task Allocation:**

Padamata Nagaraju : Report making

Natta Harika : Project Development(Frontend), (Backend

Navyaka Matham : ppt making

Nimisha Kambham : ppt making

**2.Milestones & Timeline:**

* Day 1: Finalize project idea, define objectives, and assign team roles
* Day 2: Collect and download dataset from Kaggle (recyclable & non-recyclable waste)
* Day 3: Preprocess data (resize images, label, normalize, split into train/test sets)
* Day 4-5: Train the machine learning model and evaluate accuracy
* Day 6: Develop the frontend using HTML (upload feature)
* Day 7: Integrate the trained ML model with the backend
* Day 8: Test end-to-end functionality, fix bugs, and refine the UI
* Day 9: Evaluate final results, take screenshots, and prepare project output
* Day 10: Create and finalize the PowerPoint presentation & report for demonstration

**✅ Phase-5: Integration & Development**

**Objective:**Code the project and integrate components**.**

**Key Points:**

1. **Technology Stack Used:**
   * Frontend: HTML
   * Backend: Python
   * AI Model: pre-trained CNN, Keras/TensorFlow
   * Libraries: NumPy, OpenCV, Pillow, TensorFlow
2. **Development Process:**

**✅ Dataset manually collected across six fabric pattern categories: stripes, floral, geometric, plain, abstract, and polka\_dots.**

**✅ Data preprocessing included image verification, resizing to 224x224, normalization (rescale=1./255), and label encoding using Keras ImageDataGenerator.**

**✅ Model trained using transfer learning (MobileNetV2) with pre-trained ImageNet weights on local system using TensorFlow and Keras. EarlyStopping was applied to prevent overfitting.**

**✅ Inference pipeline integrated into a Streamlit web application, replacing Flask for simplified deployment and real-time interaction.**

**✅ Frontend UI developed using Streamlit components including file uploader, prediction trigger, and confidence score display, with image preview for better UX.**

**✅ App deployed using Streamlit Community Cloud for public accessibility, enabling users to test fabric classification instantly in the browser.**

1. **Challenges & Fixes:**
   * Challenge: Class imbalance in dataset  
     Fix: Used data augmentation and class weighting during training.
   * Challenge: Real-time image preview was not updating  
     Fix: Implemented FileReader preview in JS with error fallback.
   * Challenge: Inconsistent predictions due to poor lighting in test images  
     Fix: Added normalization and brightness-contrast adjustment during preprocessing.

**✅ Phase-6: Functional & Performance Testing**

**Objective:**Ensure the project works as expected.

**Key Points:**

1. **Test Cases Executed:**
   * Uploaded various test images (different types of fabrics).
   * Verified correct category output.
   * Tested responsiveness on mobile and desktop browsers.
   * Handled invalid file types and missing files gracefully.
2. **Bug Fixes & Improvements:**
   * Fixed image filename sanitization issues using secure\_filename.
   * Added custom modals instead of default alert() for better UX.
   * Improved prediction loading and error message display.

**3.Final Validation:**

* + ✅ Meets all functional requirements listed in Phase-2.
  + ✅ End-to-end flow (image → backend → prediction → frontend) works without manual intervention.

**4.Deployment (if applicable):**

* + **Can be hosted on Render / PythonAnywhere / Heroku or local deployment using:**

nginx

CopyEdit

python app.py

http://127.0.0.1:2222

**🏁 Final Submission**

1. **Project Report:  
   Completed using the given template (content from pattern sense.docx and this summary).**
2. **Demo Video (3–5 Minutes):**
   * **Record a walkthrough:**
     + **Uploading image**
     + **Real-time prediction**
     + **Description of technology used**
     + **Impact and applications**
3. **GitHub / Code Repository Link:  
   Include:**
   * **app.py**
   * **index.html**
   * **static/uploads (empty or placeholder)**
   * **Model file: vgg16.h5**
   * **README.md explaining setup**