3. Requirement Analysis
3.4 Technology Stack (Architecture & Stack)

Date	28 June 2025
Team ID	LTVIP2025TMID35678
Project Name	Pattern Sense: Classifying Fabric Patterns using Deep Learning
Maximum Marks	4 Marks

Technical Architecture:

The architecture of the Pattern Sense system is designed to allow users to upload images through a web interface, process the images via a trained CNN model, and display classification results (e.g., "Striped", "Polka-Dotted"). The architecture consists of the following components.

Table-1: Components & Technologies:

S. No	Component	Description	Technology
1	**User Interface**	Web-based UI for users to upload fabric images and view predictions	HTML, CSS, JavaScript, Bootstrap
2	**Application Logic-1**	Logic to handle image upload and routing	Python (Flask Framework)
3	**Application Logic-2**	Preprocessing logic (resize, normalize) before model prediction	OpenCV, NumPy, Keras
4	**Application Logic-3**	Logic to call CNN model and get prediction output	TensorFlow / Keras
5	**Database**	Store image history and prediction logs (optional)	MySQL / SQLite
6	**Cloud Database**	Store large dataset or model logs in scalable storage (optional)	Google Cloud Firestore / Firebase Realtime DB
7	**File Storage**	To store uploaded fabric images temporarily	Local Filesystem / Firebase Storage
8	**External API-1**	Optional: Geolocation API to enhance UX (if used)	Google Maps API (optional)

9	**Machine Learning Model**	Classify uploaded fabric image into pattern classes	CNN (model_cnn.h5) built with Keras / TensorFlow
10	**Infrastructure**	Local Deployment via Flask / Colab / Hosting on Glitch or Render	Localhost (Flask), Google Colab, Render.com

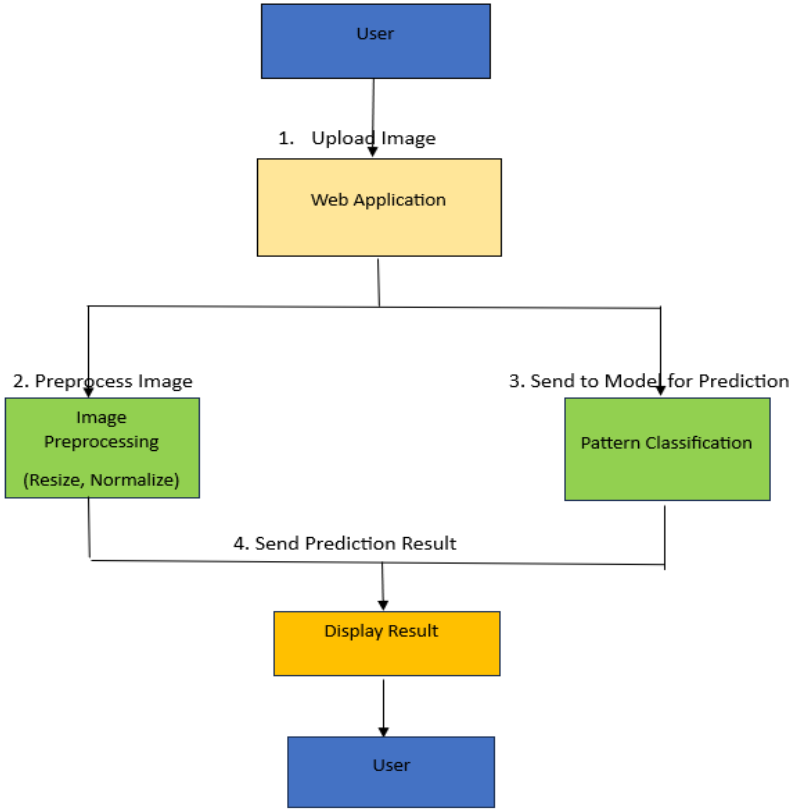

Table 2: Application Characteristics

S. No	Characteristic	Description	Technology Used
1	Open-Source Frameworks	Use of open-source frameworks for development	Flask, TensorFlow, Keras, Bootstrap

2	Security Implementations	Secure upload path, file validation, HTTPS if hosted publicly	File type checking, Flask CORS, HTTPS
3	Scalable Architecture	System can be containerized or deployed to cloud platforms	Docker (optional), Render, Firebase
4	Performance Optimization	Preloading model, limiting file size, and image size for faster prediction	Image size thresholding, model caching
5	Availability	System hosted on local machine or public server for demo	Glitch / Render / Localhost
6	Maintainability	Easy-to-maintain codebase using modular Flask routes and clean UI	Flask blueprinting, template inheritance

The Pattern Sense project is a deep learning-based image classification system developed to identify fabric patterns (e.g., Striped, Polka-Dotted, Plain, Checked) from uploaded images. The system is structured using a layered architecture to ensure modularity, performance, and scalability.

 Architecture Overview

The application consists of three main layers:

1. Presentation Layer (Frontend/UI):
    - This layer provides the user interface for uploading fabric images and displaying the predicted result.
    - It is built using standard web technologies like HTML, CSS, and JavaScript with styling support from Bootstrap.
    - The interface is simple, intuitive, and responsive for both desktop and mobile users.
2. Application Logic Layer (Backend):
    - This layer handles HTTP requests, routes the image to the CNN model, and returns the prediction results to the frontend.
    - It is implemented using the Flask framework in Python, ensuring lightweight performance and easy integration.
    - The uploaded images are preprocessed (resized, normalized) using OpenCV and NumPy before classification.
3. Model & Storage Layer:
    - The core of the system is a trained Convolutional Neural Network (CNN) built using TensorFlow/Keras.
    - The model (model_cnn.h5) is responsible for identifying the fabric pattern based on the input image.

- The system optionally stores uploaded images, prediction results, and logs using MySQL or SQLite for local development, and Firebase / Google Cloud Storage for cloud-based scaling.

🌐 External Integrations & Infrastructure

- The application can optionally integrate external APIs such as Google Maps (for geotagging images) or Feedback APIs.
- It is deployable on:
  - Local Systems using Flask
  - Cloud Platforms like Render, Firebase, or Google Colab for hosting and scalability.
- The system architecture is flexible and can be containerized using Docker for production-grade deployment.

🔒 Security & Maintainability

- The image uploads are validated (type/size) to prevent malicious file attacks.
- The backend uses Flask CORS and HTTPS (if cloud-hosted) to secure data flow.
- The system is built using open-source libraries and has a clean code structure, making it easily maintainable and extendable.

✅ Conclusion

The chosen technology stack ensures that Pattern Sense is:

- Easy to use
- Lightweight and responsive
- Modular and scalable
- Secure and adaptable for future extensions (like adding new pattern types or a mobile app)