

Internship Project Report

Title: VeinSecure- Palm Vein Authentication System



Submitted By: Anjori Sarabhai, Apoorva Kashyap

Submitted To: Mr. Prashant Tiwari
SAG Lab, DRDO

GitHub Repository: <https://github.com/anjorisarabhai/veinsecure-palm-vein-authentication>

1. Introduction

VeinSecure is a secure biometric authentication system that uses palm vein patterns to verify identity. Palm vein recognition offers several advantages over other biometrics such as fingerprints or facial recognition due to its internal nature, resistance to forgery, and high accuracy. This project focuses on building an end-to-end authentication system that includes image analysis, deep learning-based classification, a Flask-based backend, and a modern web interface.

2. Dataset Description

- **Source:** [BMPD Dataset on Kaggle](#)
 - **Content:** Palm vein images of 41 individuals.
 - **Format:** JPG/PNG image format, organized into folders by user ID (e.g., 001/, 002/, ..., 041/).
 - **Size:** Varies per image; preprocessing ensures uniformity.
 - **Structure Screenshot Placeholder:** *Insert screenshot showing folder structure of raw dataset.*
-

3. Exploratory Data Analysis (EDA)

Image Structure & Format Analysis

- Counted number of images per user.
- Verified folder consistency programmatically.
- Visualized class distribution using bar charts.
- Calculated image sizes, resolutions, and flagged corrupted files.

Class Distribution:

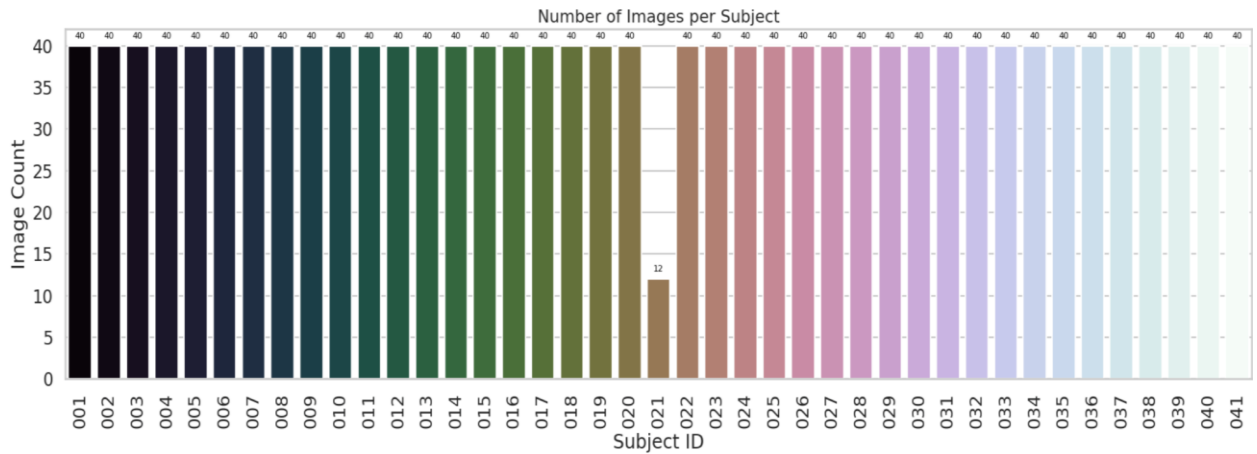
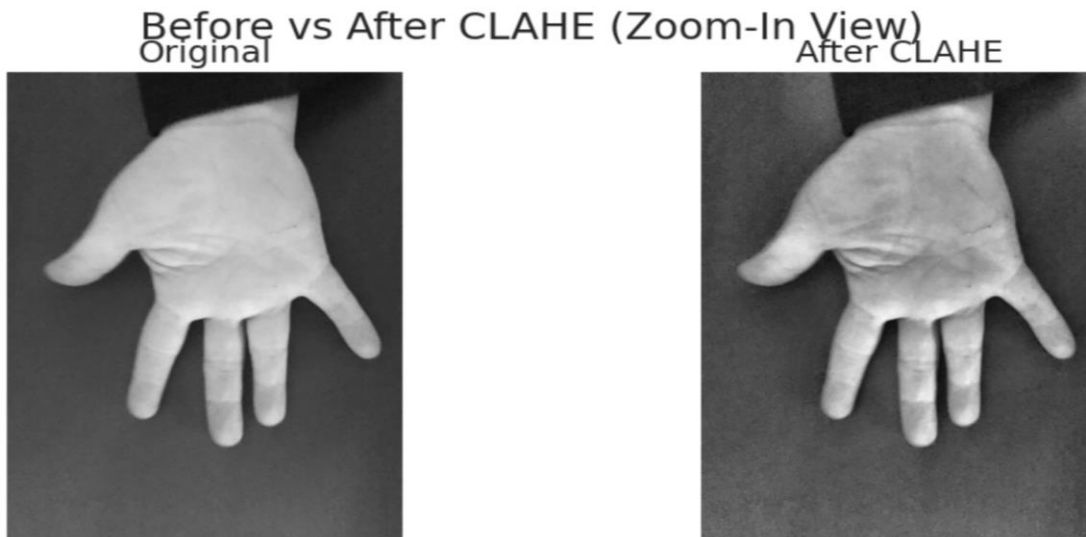


Image Intensity & Quality Assessment

- Plotted pixel intensity histograms (RGB & grayscale).
- Applied CLAHE (Contrast Limited Adaptive Histogram Equalization) to improve contrast.
- Computed brightness statistics (mean, variance).
- Identified images that were too dark or too bright.

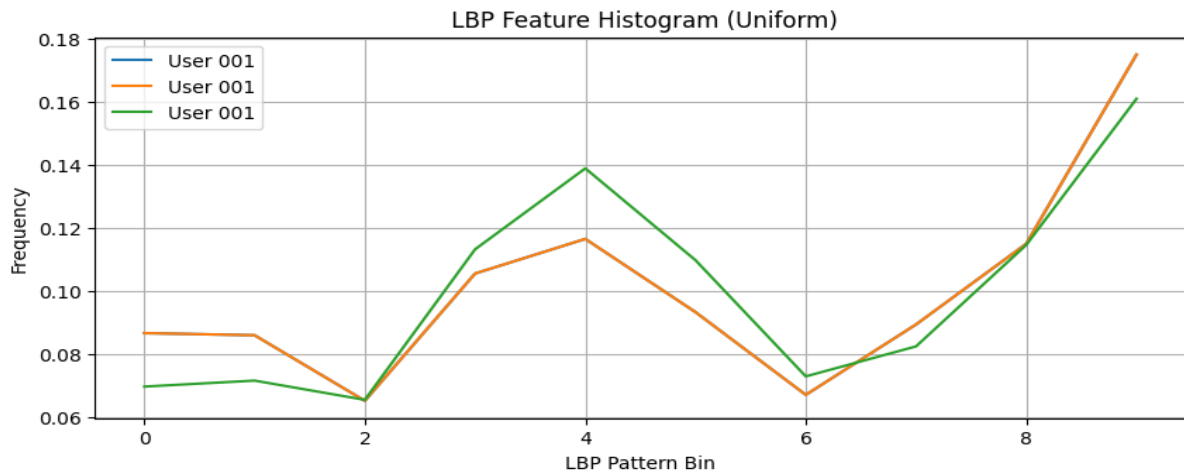
before/after CLAHE comparison image grid:



Intra-Class vs Inter-Class Variation

- Computed SSIM and MSE to compare images within the same user vs across users.
- Started PCA/t-SNE for dimensionality reduction.
- Visualized separability using 2D scatter plots.

PCA/t-SNE plot colored by user:



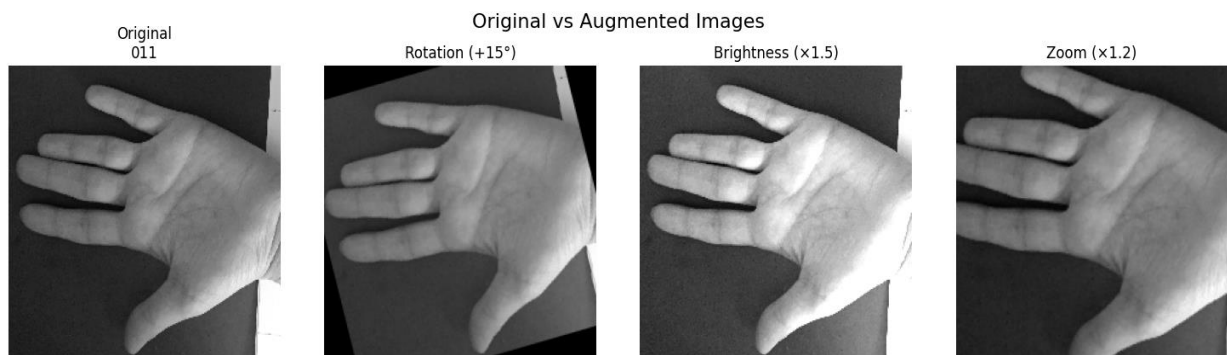
Classification Report:

	precision	recall	f1-score	support
001	0.50	1.00	0.67	1
002	1.00	1.00	1.00	1
003	0.00	0.00	0.00	1
004	1.00	1.00	1.00	1
accuracy			0.75	4
macro avg	0.62	0.75	0.67	4
weighted avg	0.62	0.75	0.67	4
Accuracy: 0.75				

Augmentation Feasibility

- Applied augmentations: rotation, zoom, flips, brightness/contrast shifts.
- Measured SSIM before/after augmentation.
- Flagged augmentations that distorted identity significantly.

image grid showing raw vs augmented samples:

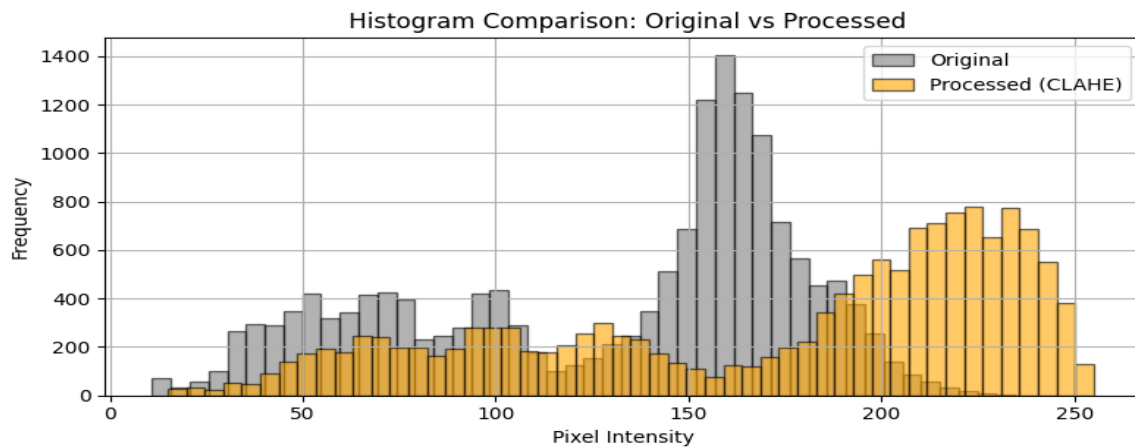


4. Preprocessing Pipeline

- Developed reusable functions for grayscale conversion, resizing, normalization.
- Saved processed images to a separate directory (`/data/processed/`).
- Visual comparison of raw vs processed images.

- Created data loaders to visualize processed batches.

plot comparing original and processed images:

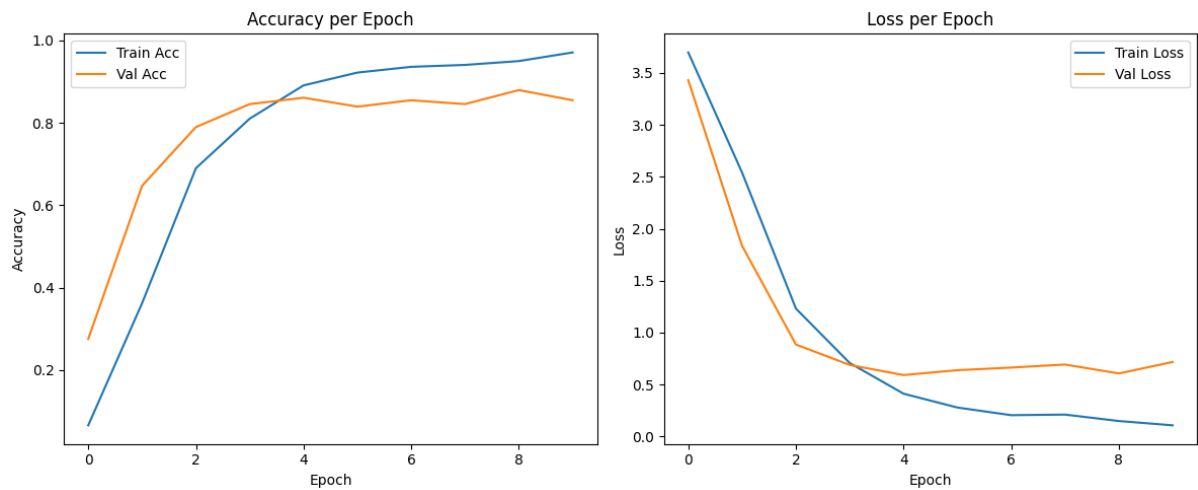
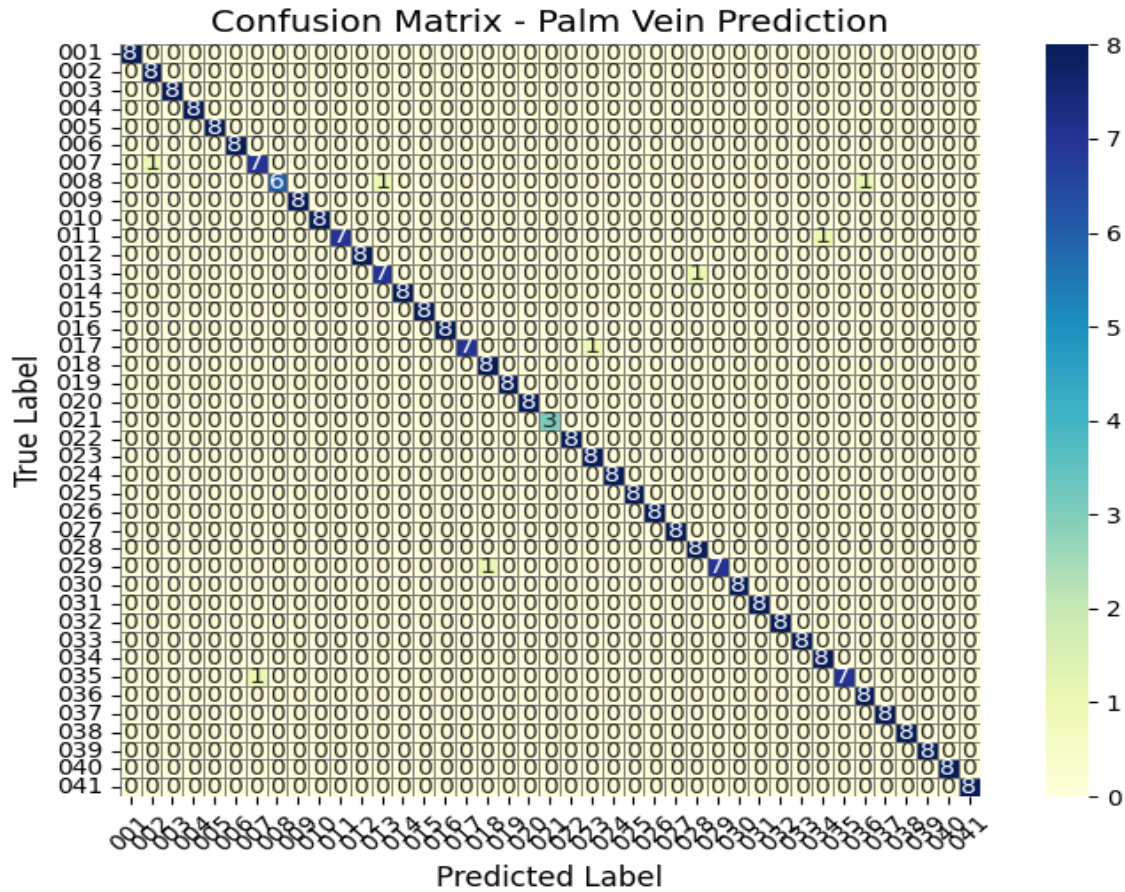


5. Model Development

Baseline CNN

- Created a simple CNN in `baseline_model.ipynb`.
- Stratified dataset split (user-wise) into train/test.
- Tracked training accuracy, loss, and plotted confusion matrix.

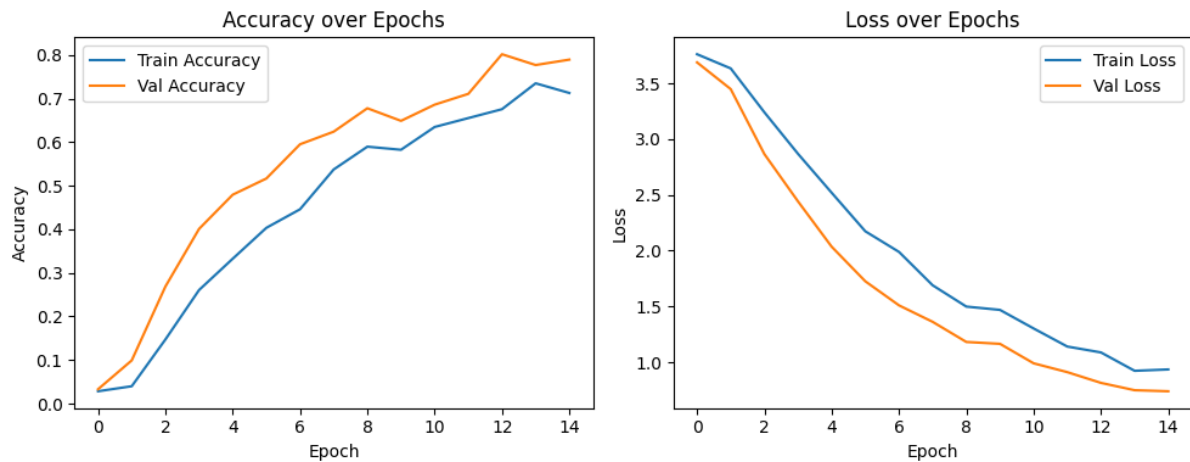
confusion matrix and training curve:



Final CNN Model

- Modular training setup in `models/train_model.py`.
- Used Keras ImageDataGenerator with augmentation.
- Hyperparameters: optimizer (Adam), dropout, batch size = 32.
- Early stopping and model checkpoint callbacks.
- Final model saved as `final_model.h5`.

learning curve plot (loss vs epochs):



6. Web Application (Flask + Bootstrap)

Backend (`localapp.py`)

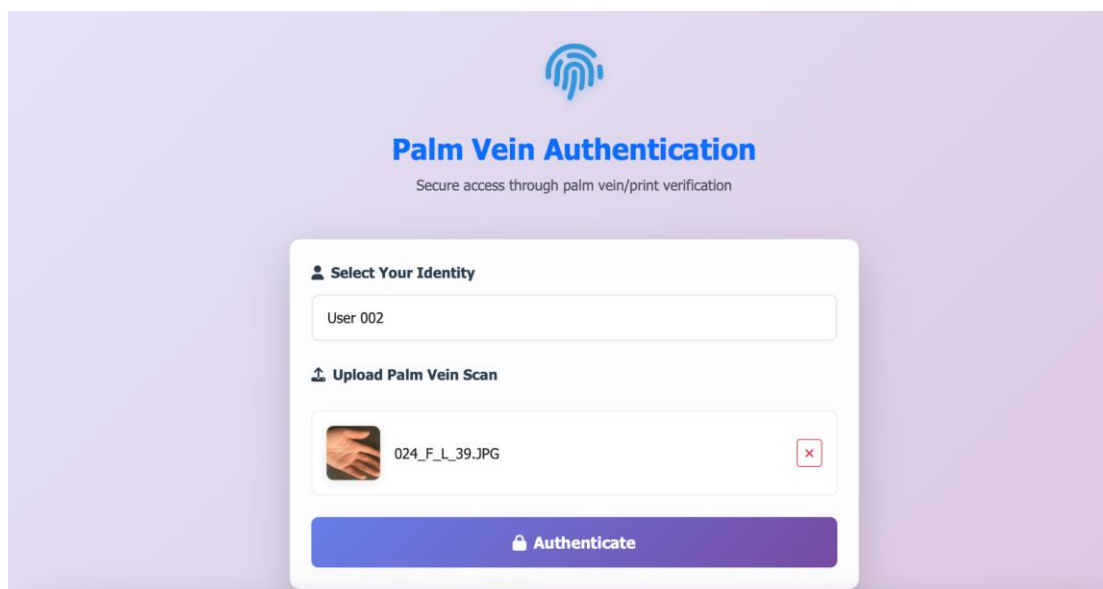
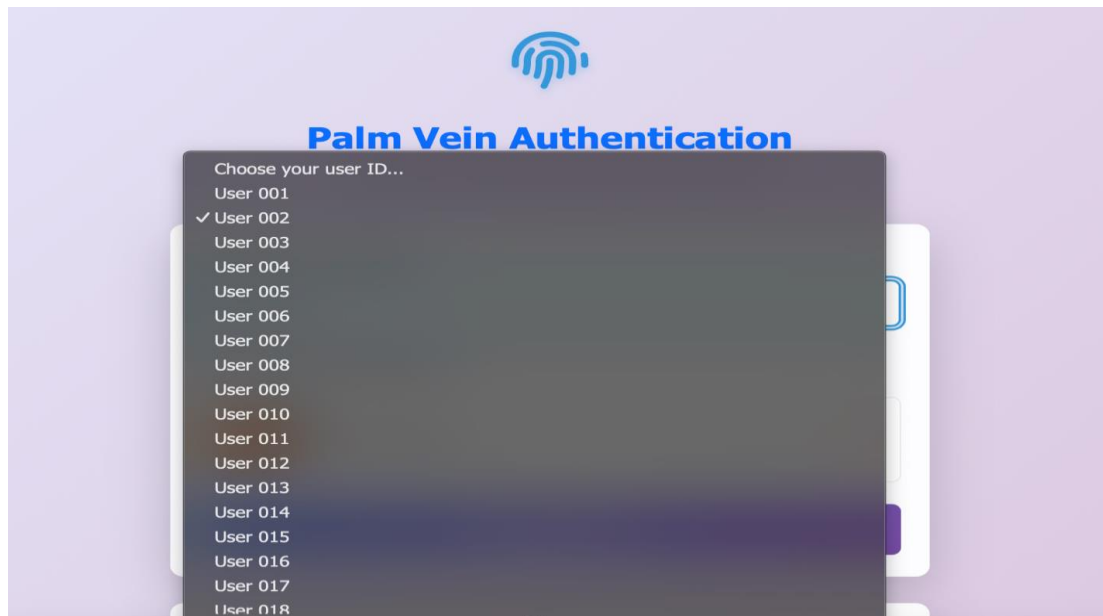
- Loads the pre-trained CNN model.
- Accepts uploaded image and claimed user ID via `/authenticate` POST route.
- Preprocesses image, predicts user ID.
- Logs authentication attempts.
- Implements rate-limiting: locks out user after 5 failed attempts for 60 seconds.

Stack: Flask, TensorFlow, OpenCV, Python, JSON, Logging



Frontend (HTML + Bootstrap)

- User-friendly, responsive interface.
- Lavender gradient theme with clean form layout.
- Features:
 - Dropdown to select claimed identity (User 001 - 041)
 - Drag-and-drop or file select for image upload
 - Live preview of uploaded image
 - Disabled submit until valid input

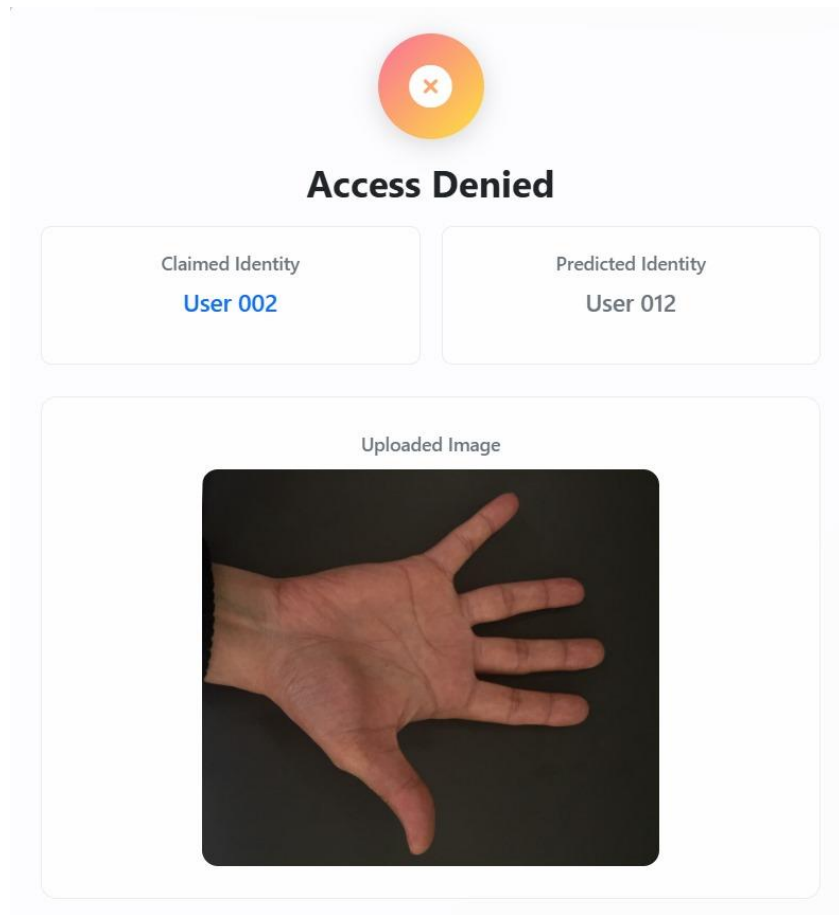
UI showing form with file upload & dropdown:



Results Display

- Displays predicted ID, claimed ID, and access decision (granted/denied).
- Shows uploaded image in result card.
- Visual indicators:  for success,  for denial.

example output with granted/denied view:



7. Technologies Used

Component	Tech Stack
Model Training	TensorFlow, Keras
Image Handling	OpenCV, NumPy, Matplotlib
Data Analysis	pandas, seaborn, scikit-image
Web App Backend	Flask, Python

Frontend UI	HTML, CSS, Bootstrap
Deployment	Localhost Testing
Logging	Python Logging, CSV Logger

8. Limitations & Future Work

- Static model: Requires retraining if new users are added.
- Plan to shift to dynamic authentication using Siamese or Triplet networks.
- Add support for user management and admin dashboard.
- Consider real-time camera input and mobile-friendly layout.

9. Literature Review

- [Deep Learning Techniques for Hand Vein Biometrics: A Comprehensive Review](#)
Mustapha Hemis et al., 2024
Provides a broad overview of deep learning applications in hand vein biometrics, including palm vein. It introduced us to important datasets, CNN variants, and performance metrics.
 - [N-shot Palm Vein Verification Using Siamese Networks](#)
Felix Marattukalam et al., 2021
Introduced a Siamese neural network approach for few-shot verification of palm veins. This research laid the groundwork for our future work on dynamic user addition.
 - [Palm Vein Recognition via Multi-task Loss Function and Attention Layer](#)
Jiashu Lou et al., 2022
Proposed using attention layers and a multi-task loss to boost accuracy in palm vein classification tasks. It supported our interest in enhancing feature extraction and robustness.
-

10. References

- [TensorFlow Documentation](#)
- [OpenCV Python Docs](#)
- [Keras API Reference](#)
- [Flask Documentation](#)
- [scikit-image Library](#)