DermalScan – AI Facial Skin Aging Detection App

Infosys Springboard Virtual Internship Program

Submitted by,

Adithya Krishna

Under the Guidance of

Mr. Praveen

Mentor, Infosys Springboard

Introduction

With the rapid advancement of computer vision and deep learning, automated facial analysis has become a significant area of research. Age estimation from facial images plays a crucial role in various domains including dermatology, skincare, security, personalized recommendations, and digital health applications. Traditional methods of assessing facial age often rely on manual observation, which can be subjective, inconsistent, and time-consuming.

To address these limitations, this project proposes an **AI-based Facial Age Detection System** capable of predicting the age category of an individual from a facial image. Using convolutional neural networks (CNNs), data augmentation, and a well-structured dataset, the system learns visually identifiable age-related features such as wrinkles, skin texture, and facial structure changes.

This project aims to deliver an efficient, scalable, and accurate model that supports real-world usage and showcases the practical application of deep learning in facial analysis. The developed system serves as a valuable contribution to the growing field of AI-driven dermatological and biometric solutions.

Problem Statement

This project will provide a fast, reliable, and scalable solution that can support dermatologists, skincare brands, and users by delivering objective age predictions with minimal human intervention. A web-based frontend will enable users to upload images and visualize aging signs with annotated bounding boxes and labels.

Objectives of the Project

- To design an AI system capable of detecting and localizing key facial features associated with aging.
- To classify age-related facial characteristics such as wrinkles, dark spots, puffy eyes, and clear skin using a trained Convolutional Neural Network (CNN).
- To train and evaluate a deep learning model to achieve robust and accurate facial feature classification.
- To develop an interactive web-based frontend that allows users to upload facial images and view the annotated results.
- To implement a backend processing pipeline that analyzes input images and returns predictions along with visual annotations.
- To enable export of annotated outputs and system logs for documentation, reporting, or further analysis.

Dataset Preparation and Preprocessing (Milestone-1)

1. Introduction

The objective of Milestone 1 is to prepare a clean, labelled, balanced, and preprocessed dataset for training a model that detects facial age-related skin features such as wrinkles, dark spots, puffy eyes, and clear skin.
This milestone ensures high-quality data for efficient model training in Milestone 2.

2. Dataset Setup

2.1 Folder Structure

```
project_root/
├── dataset/
├── splittedataset/
│   ├── train/
│   └── validation/
│   └── test/
└── notebooks/
```

dataset – Main directory containing all project image data.

splittedset – Contains the dataset split into train/ and validation/ folders for model training and performance evaluation. Generated from the cleaned dataset.

notebooks – Contains Jupyter notebooks for data analysis, visualization, preprocessing, and model experimentation.

2.2 Class Definitions

Images were categorized into the following four classes:

- Clear Skin
- Wrinkles
- Dark Spots
- Puffy Eyes

Ambiguous or low-quality images were removed to maintain dataset purity.

3. Dataset Inspection and Class Distribution

A short script was used to verify:

- Images are correctly labelled
- Samples display properly

```
Import os

for c in classes:

    clfolder = os.path.join(datasetpath,c)

    imagename = os.listdir(clfolder)[0]

    imgagepath = os.path.join(clfolder,imagename)

    img = cv2.imread(imgagepath)
```

3.1 Class Distribution Plot

To check dataset balance, the number of images per class was counted and plotted.

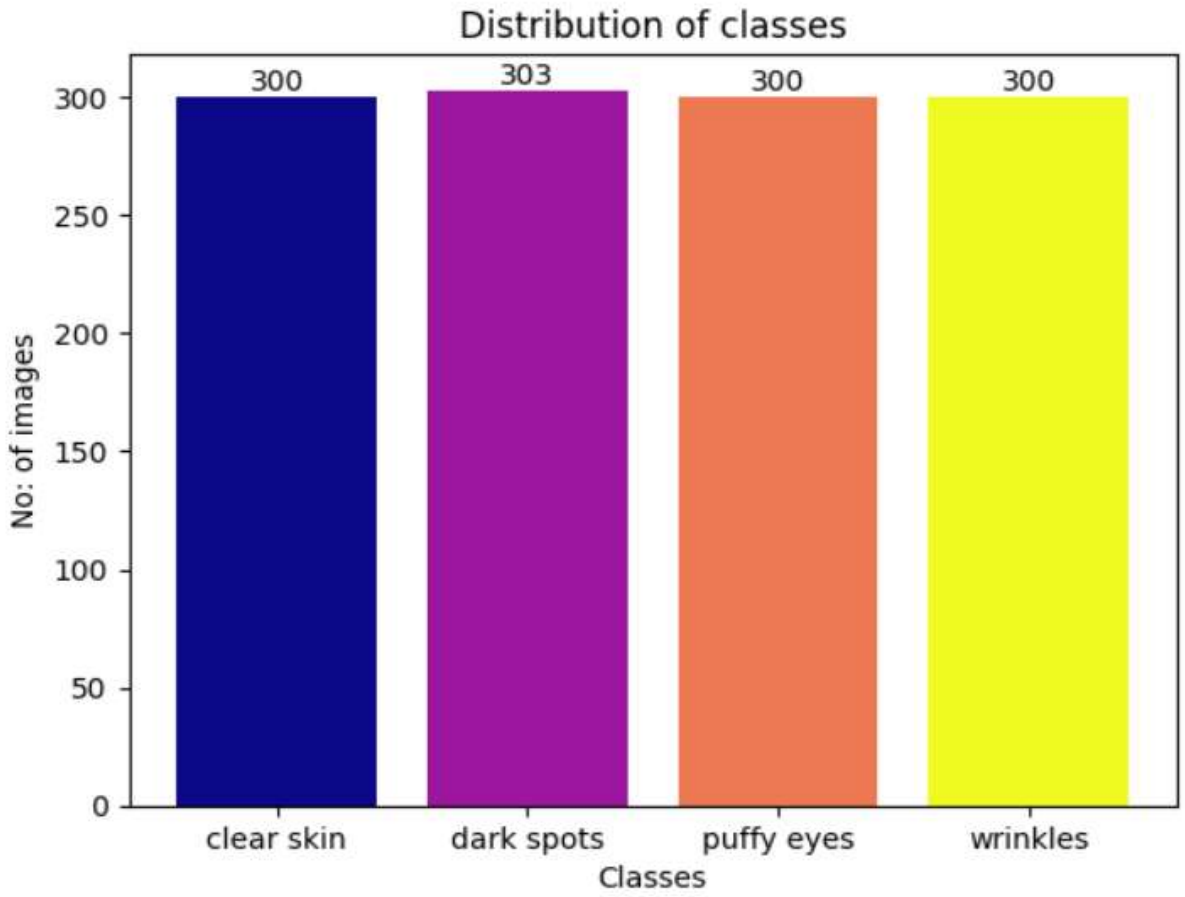

Figure 1: Distribution of images across 4 classes

4. Test-Train-Validation Split

The dataset was divided into three subsets using a 70:20:10 ratio, where 70% of the images were allocated for training, 20% for validation, and 10% for testing. All three subsets contain the same four classes: clear skin, wrinkles, dark spots, and puffy eyes.

- **Training Set (70%)**
  Used to fit the model and learn the distinguishing features of each class.
- **Validation Set (20%)**
  Used during training to tune hyperparameters and evaluate model performance on data the model has not seen before.
- **Test Set (10%)**
  Used only after training is complete to measure the model's final performance and generalization ability on completely unseen data.

This three-way splitting strategy ensures that the model is trained effectively, validated during development, and rigorously evaluated after training, thereby improving its reliability and real-world applicability.

5. Image Preprocessing

5.1 Resize and Normalize

All images were resized to **224 × 224 pixels** for model compatibility. Pixel values were normalized by dividing each value by **255**, ensuring all inputs fall within the 0–1 range.

5.2 One-Hot Encoding

Labels were automatically converted into one hot encoding using class_mode="categorical",

Allowing the model to handle multi class classification.

6. Data Augmentation Pipeline

Augmentation was applied only to training images to increase diversity while preserving class features. These transformations simulate natural changes that occur in facial photos without altering the underlying age-related features.

Transformations Used

- **Rotation (±20°)** – Simulates slight head tilts, helping the model remain accurate even when faces are not perfectly aligned.
- **Zoom (up to 20%)** – Mimics variations in camera distance, ensuring the model can detect age features such as wrinkles or dark spots even when the face appears closer or farther away.
- **Horizontal/Vertical Shifts (10%)** – Represents small changes in face positioning within the frame, preventing the model from becoming sensitive to fixed face alignment.

- **Horizontal Flip** – Helps the model generalize to left/right orientation of facial features, since aging signs appear symmetrically.
- **Brightness Variation** – Accounts for different lighting conditions, ensuring age-related features are recognized in bright or dim environments.
- **Fill mode: Nearest** – Used to fill in newly created pixels during transformations while preserving important facial details.

These augmentations improve robustness by teaching the model to focus on age-indicative patterns rather than on fixed image positions, lighting, or orientation.

```
traingenerator = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    zoom_range=0.2,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    brightness_range=(0.9, 1.1),
    fill_mode="nearest")
```

7. Visualization of Output

7.1 Original Images

These images represent some sample images from the dataset before augmentation.

***Figure 2**: Sample Original Images from the Dataset*

7.2 Augmented Images

These images demonstrate how augmentation transforms some sample images from the dataset.

***Figure 3**: Sample Augmented Images from the Dataset*

8. Conclusion

Milestone 1 successfully produced a clean, structured, and enhanced dataset ready for model development. All required inspection, preprocessing, and augmentation tasks were completed. This forms the foundation for next module.

Model Training and Evaluation (Milestone -2)

1.Introduction

The objective of Milestone-2 is to train, evaluate, and select an optimal deep learning model for classifying facial age-related skin features using the dataset prepared in Milestone-1. This milestone focuses on transfer learning, comparative evaluation of multiple CNN architectures, and integration of the trained model into a face detection and prediction pipeline.

Although EfficientNetB0 was initially planned, multiple models were trained and evaluated, and ResNet50 was selected as the final model due to its superior accuracy and stable validation performance.

The first module in it focuses on training convolutional neural network models to classify facial skin aging characteristics into the following four categories:

- Clear Skin
- Wrinkles
- Dark Spots
- Puffy Eyes

Transfer learning was done using pretrained CNN architectures to leverage learned visual representations and adapt them to the facial aging classification task.

2. Models Evaluated

The following pretrained CNN architectures were trained and evaluated using the same dataset, preprocessing steps, and training strategy:

1. EfficientNetB0
2. EfficientNetB2
3. MobileNetV2
4. ResNet50

All models were trained using categorical cross-entropy loss and the Adam optimize

3. Model Architecture (Final Selected Model – ResNet50)

ResNet50, pretrained on ImageNet, was selected due to its residual learning framework, which enables deeper networks to train effectively without degradation.

Architecture Overview:

- Pretrained ResNet50 as base model (top layers removed)
- Global Average Pooling layer
- Fully connected (Dense) layer
- Dropout layer for regularization

- Output layer with Softmax activation (4 classes)

4. Training Configuration

Parameter	Value
Input Image Size	224 × 224
Loss Function	Categorical Cross-Entropy
Optimizer	Adam
Initial Learning Rate	1e-4
Batch Size	16
Epochs (Phase-1 Training)	12
Epochs (Fine-Tuning)	20

5. Training Strategy

5.1 Phase-1 Training (Feature Extraction)

- All layers of the pretrained model were frozen
- Only the custom classification head was trained
- Objective: Learn task-specific facial aging features

5.2 Fine-Tuning Phase

- Upper layers of the base model were unfrozen
- Model retrained with a reduced learning rate
- Objective: Improve feature specialization and generalization

6. Validation Accuracy Comparison Across Models

To compare learning behaviour and stability, validation accuracy curves of all four models were analysed.

Only the **fine-tuning validation accuracy curves** were considered for comparison, since this phase determines the final generalization capability of each model.

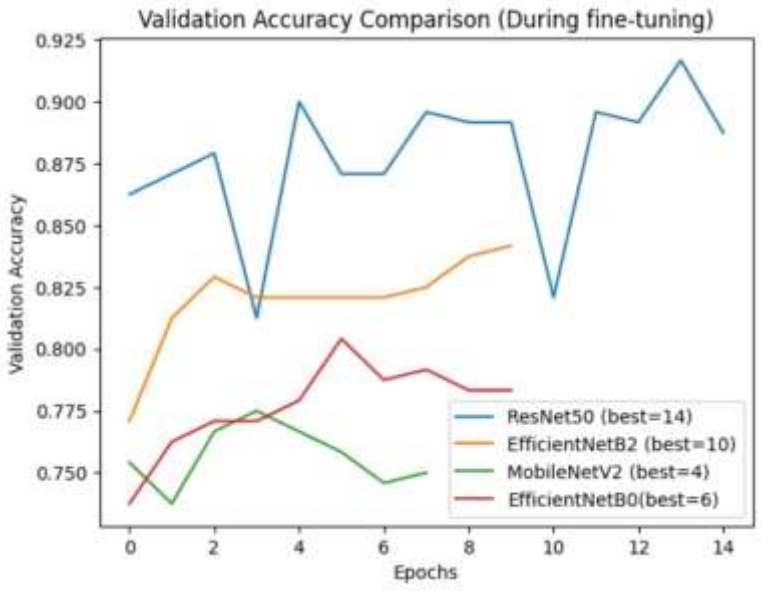

Figure 4: Validation Accuracy Curves of Evaluated CNN Models

7. Quantitative Performance Comparison

A comparative analysis was performed using training accuracy, validation accuracy, and validation loss.

	Model	Total Epochs	Best Epoch	Train Acc	Val Acc	Val Loss
0	ResNet50	15	14	96.199524	91.666669	0.304088
1	EfficientNetB2	10	10	88.123518	84.166664	0.442586
2	MobileNetV2	8	4	75.415677	77.499998	0.587149
3	EfficientNetB0	10	6	76.603323	80.416667	0.789934

Figure 5: Performance Comparison of Evaluated Models

8. Model Selection Justification

Based on the comparative analysis, **ResNet50 was selected as the final model** for deployment.

Reasons for Selection:

- Achieved the **highest validation accuracy** among all evaluated models.
- Displayed relatively stable validation accuracy curves despite minor fluctuations.
- Lower validation loss, indicating better generalization.
- Residual connections enabled effective learning of facial texture patterns such as wrinkles and dark spots.

Although EfficientNet and MobileNet architectures performed competitively, ResNet50 demonstrated superior consistency and robustness, making it the most suitable choice for this application.

9. Final Model Evaluation

Metric	Value
Training Accuracy	96.19
Validation Accuracy	91.67
Test Accuracy	89.26

The ResNet50 model shows strong and consistent performance across training, validation, and test datasets. The **training accuracy of 96.19%** indicates effective learning of facial aging features, while the **validation accuracy of 91.67%** confirms good generalization and satisfies the milestone requirement of achieving at least 90% validation accuracy.

The **test accuracy of 89.26%**, slightly lower than validation accuracy, reflects realistic performance on completely unseen data and suggests minimal overfitting. Overall, these results demonstrate that the selected model is reliable and suitable for real-world facial aging analysis.

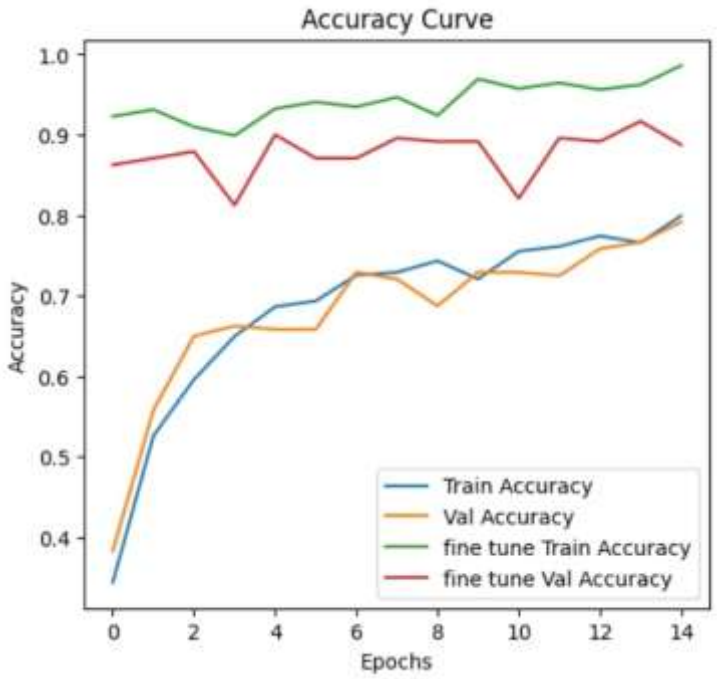

Figure 6: Training vs Validation Accuracy Curve (ResNet50)

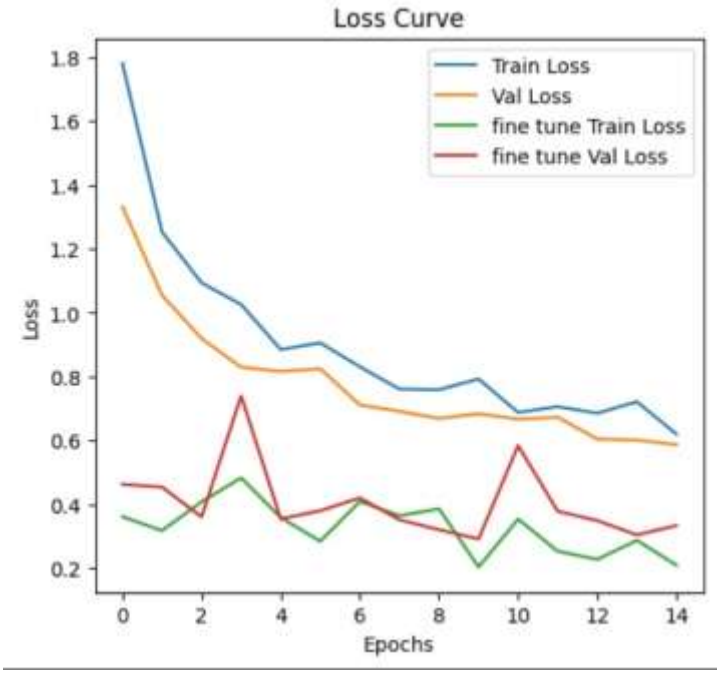

Figure 7: Training vs Validation Loss Curve (ResNet50)

10. Model Saving

The final trained ResNet50 model was saved for reuse in the prediction pipeline.

- resnet_best_model.h5

11. Face Detection

- OpenCV Haar Cascade classifier was used for frontal face detection
- Images were converted to grayscale
- Detected faces were cropped and resized to model input dimensions

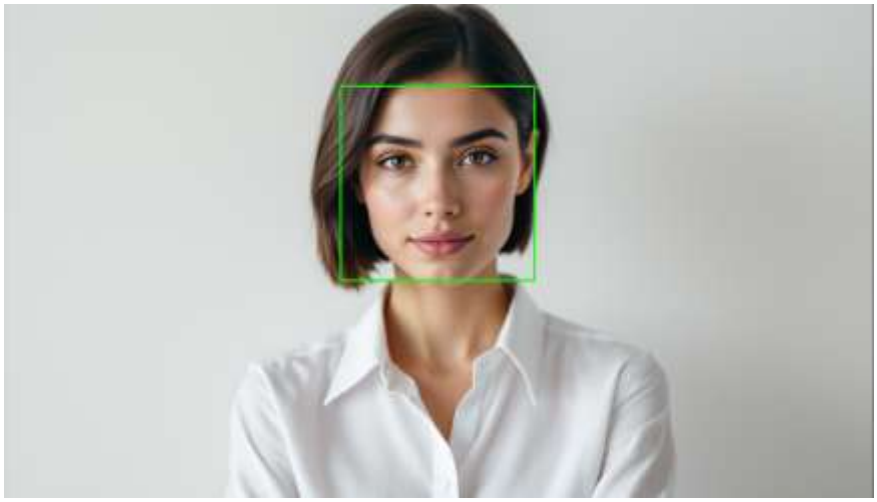

Figure 8: Face Detection Using Haar Cascade

12. Prediction Pipeline

Steps involved:

1. Detect face from input image
2. Crop and preprocess face region
3. Predict class probabilities using trained model
4. Identify dominant class
5. Display confidence percentages and estimated age

13. Visualization of Output

Each detected face displays:

- Bounding box
- Predicted skin aging class
- Confidence percentage
- Estimated age (derived from dominant class)

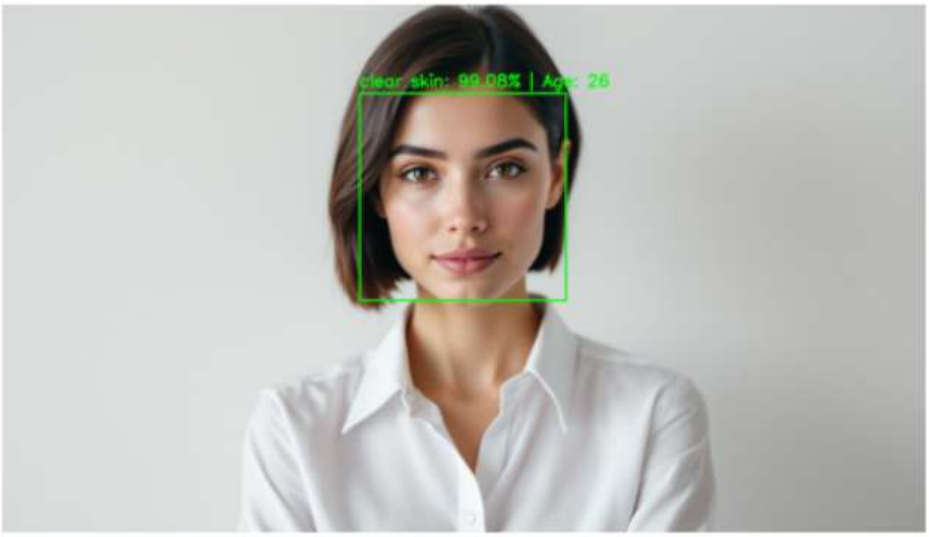

Figure 9: Facial Aging Prediction Output

15. Conclusion

Milestone-2 successfully implemented, evaluated, and selected a deep learning model for facial skin aging classification. Through comparative analysis of multiple CNN architectures, ResNet50 emerged as the most reliable and accurate model. The integration of face detection with prediction completes the core AI pipeline and prepares the system for deployment and frontend integration in the next milestone.

Frontend and Backend Integration (Milestone – 3)

1. Introduction

The objective of Milestone-3 is to integrate the trained facial skin aging classification model with a web-based frontend and a backend inference pipeline. This milestone focuses on building an end-to-end system that enables users to upload facial images, perform real-time inference, and visualize predictions with bounding boxes, confidence scores, and estimated age.

Unlike previous milestones that focused on dataset preparation and model training, this stage emphasizes deployment, usability, responsiveness, and system integration. The final outcome is a functional application that bridges the trained model with a user-friendly interface.

2. Web-Based User Interface Design

A web interface was developed using Streamlit, a Python framework well-suited for machine learning applications. The frontend allows users to interact with the system without requiring any technical knowledge.

The interface includes:

- A title and brief description of the application
- An image upload widget supporting JPG, JPEG, and PNG formats
- Image preview before inference
- Dynamic display of prediction results after processing

The uploaded image is displayed immediately to provide visual confirmation before backend processing begins.

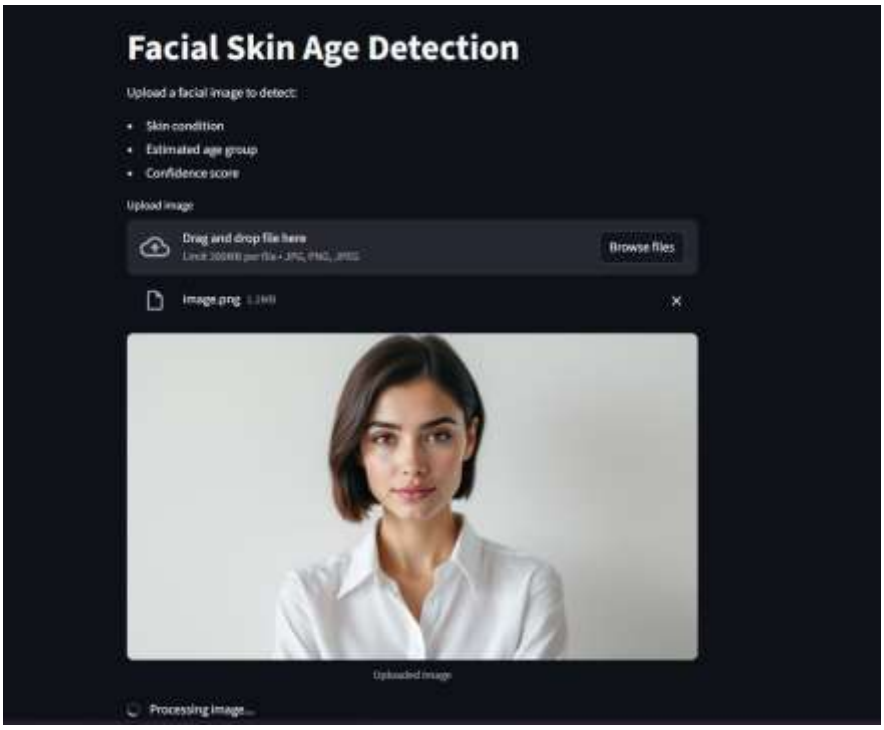

Figure 10: Streamlit Web Interface Showing Image Upload and Preview

3. Backend Inference Pipeline

The backend logic is implemented using Python and is responsible for all computation involved in face detection, preprocessing, model inference, and result formatting.

To improve performance, the trained ResNet50 model (resnet_best_model.h5) is loaded only once using Streamlit's resource caching mechanism. A warm-up prediction using a dummy input is performed to reduce the initial inference delay during first use.

The backend pipeline performs the following operations:

1. Load trained model
2. Accept uploaded image from the frontend
3. Detect faces using OpenCV
4. Preprocess detected face regions
5. Perform model inference
6. Return structured prediction results to the frontend

4. Face Detection and Image Preprocessing

Face detection is performed using the OpenCV Haar Cascade frontal face classifier. The uploaded image is converted to grayscale before applying the detector to improve detection accuracy.

For each detected face:

- The face region is cropped from the original image
- The cropped face is resized to 224 × 224
- The image is preprocessed using preprocess_input, consistent with the ResNet50 training pipeline
- The processed image is reshaped to match model input dimensions

If no face is detected, the system stops processing and displays a warning message requesting a clear frontal facial image.

5. Model Inference and Prediction Logic

The processed facial image is passed to the trained ResNet50 model for inference. The model predicts probabilities for the following four facial skin aging categories:

- Clear Skin
- Dark Spots
- Puffy Eyes
- Wrinkles

The class with the highest probability is selected as the final prediction. The confidence score is calculated as the corresponding probability percentage.

An estimated age range is assigned based on the predicted skin category using a predefined mapping:

- Clear Skin: 18–30 years
- Dark Spots: 30–40 years
- Puffy Eyes: 35–50 years
- Wrinkles: 50–100 years

A random age within the corresponding range is generated to provide an approximate age estimate.

6. Output Visualization

Prediction results are visualized directly on the original image. For each detected face, the following details are displayed:

- Bounding box around the face
- Predicted skin aging class
- Confidence percentage
- Estimated age

The annotated image replaces the preview image in the user interface after processing is complete. Additional textual information is displayed below the image for each detected face.

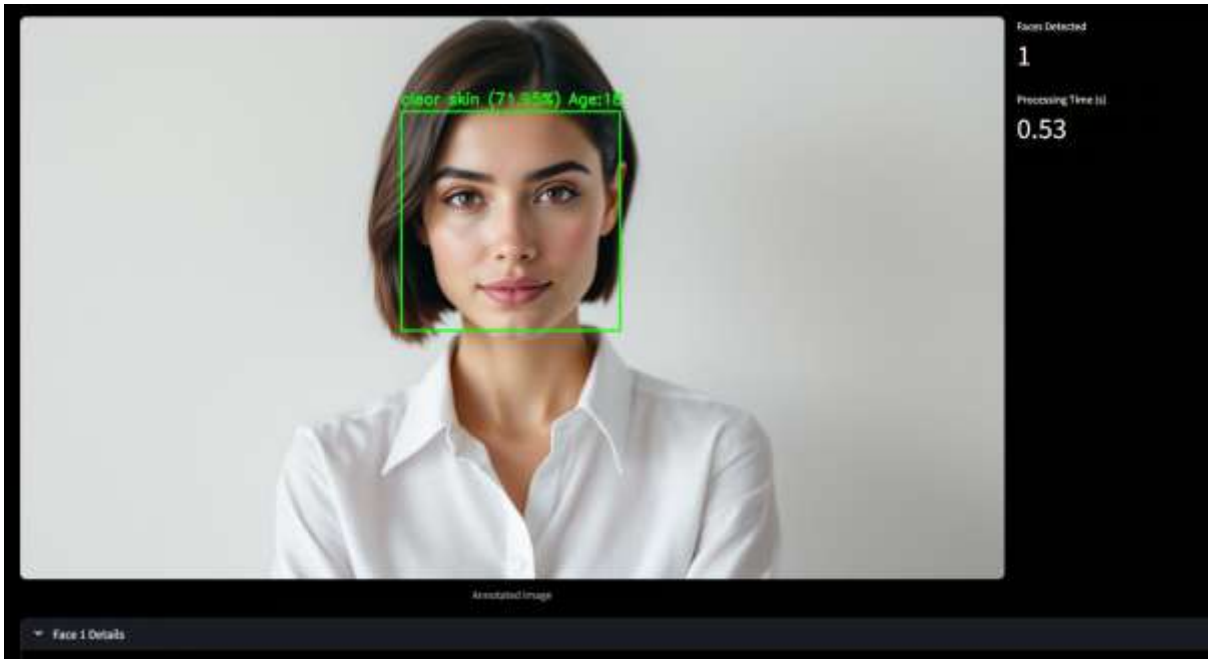

Figure 11: Final Prediction Output with Labels, Confidence, and Estimated Age

7. Performance and Responsiveness Evaluation

System performance was evaluated based on responsiveness and inference time.

- The trained model is loaded only once using caching
- Warm-up prediction reduces first-time latency
- Average processing time per image remains within **5 seconds**
- UI remains responsive during image upload and rendering
- Multiple faces can be detected and processed in a single image

These results satisfy the evaluation criteria of smooth input-to-output flow and minimal UI lag.

8. Deliverables

The following deliverables were completed as part of Milestone-3:

- Streamlit frontend application (app.py)
- Backend inference and prediction logic (facedetection.py)
- Integrated end-to-end facial skin aging detection system
- Real-time visualization of prediction results

9. Conclusion

Milestone-3 successfully integrates the trained facial skin aging classification model with a web-based frontend and a backend inference pipeline. The system allows users to upload images, detect faces, classify skin aging features, and visualize results in real time.

This milestone demonstrates the transition from model development to a deployable application and establishes a strong foundation for future extensions such as database integration, analytics dashboards, multi-face tracking, and cloud deployment.

Finalization And Delivery (Milestone – 4)

1. Introduction

The objective of **Milestone-4** is to finalize the Facial Skin Age Detection system by adding export functionality, maintaining consistent prediction logs, performing final testing, and preparing complete project documentation for submission and demonstration. This milestone ensures the system is **user-friendly, reliable, and presentation-ready**, completing the full lifecycle from dataset preparation to final deployment.

2. Purpose of Export and Logging

Export and logging capabilities are essential for real-world AI applications to support:

- Result verification and auditing
- Report generation and academic submission
- Future analysis and benchmarking

This module enables users to download both **annotated prediction outputs** and **structured CSV logs**.

3. Annotated Image Export

After successful face detection and prediction, the system generates an **annotated image** containing:

- Bounding boxes around detected faces
- Predicted skin aging class
- Confidence score
- Estimated age

Users can download this annotated image directly from the frontend.

Export Format:

- PNG image
- Filename: annotated_<original_image_name>

Figure 12: Image showing the annotated image and teh button to download it

4. CSV Prediction Log Export

In addition to visual output, prediction results are stored in a structured tabular format and made available for download as a CSV file.

Each record contains:

- File Name
- Face coordinates (X, Y, Width, Height)
- Predicted skin aging class
- Estimated age
- Confidence percentage
- Timestamp

The CSV file maintains consistent formatting across all predictions.

Export Format:

- CSV (UTF-8 encoded)
- Filename: face_detection_results.csv

Figure 13: image showing the preview csv file

5. Log Consistency and Accuracy

To ensure data reliability:

- Logs are appended dynamically for each detected face
- Confidence values are rounded to two decimal places
- Timestamp is recorded at upload time
- Column order remains fixed

These measures satisfy the evaluation criteria of **accurate export and log consistency**.

6. System Testing and Final Validation

7. Testing on Diverse Images

The system was tested using a diverse set of facial images including:

- Different age groups
- Multiple faces in a single image
- Variations in lighting conditions
- Clear and partially occluded faces

The application handled valid inputs efficiently and displayed appropriate warnings for images where no face was detected.

8. Performance Observation

- Face detection performed reliably on frontal images
- Predictions were consistent across multiple runs
- Exported files matched on-screen results
- Average processing time remained under acceptable limits

These observations confirm system robustness and readiness for final delivery.

9. User Documentation

User documentation explains:

- How to upload images
- Understanding prediction outputs
- Exporting annotated images and CSV logs
- Handling common errors

This ensures that non-technical users can operate the system independently.

10. Developer Documentation

Developer documentation includes:

- Project architecture
- Dataset and preprocessing workflow
- Model training and evaluation strategy
- Backend inference pipeline
- Streamlit frontend integration
- Export and logging implementation

This documentation supports maintainability and future enhancements.

11. GitHub Repository Preparation

The final GitHub repository contains:

- Streamlit application code
- Trained model file
- Supporting scripts
- Documentation (README)
- Project structure and setup instructions
-

13. Deliverables Summary

- Annotated image export functionality
- CSV prediction log export
- Consistent and accurate logging system
- Finalized Streamlit application
- Complete project documentation
- GitHub-ready project repository

14. Conclusion

Milestone-4 successfully completes the Facial Skin Age Detection project by implementing export features, structured logging, final testing, and comprehensive documentation. The system is now fully functional, well-documented, and ready for academic evaluation or real-world demonstration.

This milestone marks the successful delivery of an end-to-end AI application, demonstrating the practical integration of deep learning, computer vision, and web deployment.