VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI-590 018, KARNATAKA

ARKA Educational & Cultural Trust

JAIN INSTITUTE OF TECHNOLOGY

Davanagere-577003, KARNATAKA

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

FINAL YEAR B.E. (2025 – 2026)

A Project Report

On

"Advanced Image Detection and Classification "

UNDER THE GUIDANCE OF

Mr. Santosh Kumar G S

Assistant Professor, Dept. of ECE

J.I.T., Davanagere

PROJECT ASSOCIATES

Ms. Bhoomika S V	4JD22EC012
Ms. Manjula A	4JD22EC030
Ms. Niveditha B	4JD23EC405
Ms. Amulya G	4JD23EC400

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI – 590018

ARKA Educational and Cultural Trust®
JAIN INSTITUTE OF TECHNOLOGY
DAVANAGERE–577003, KARNATAKA

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **"Advanced Image Detection and Classification"** carried out by **Ms. Bhoomika S V (4JD22EC012), Ms. Manjula A (4JD22EC030), Ms. Niveditha B (4JD23EC405) and Ms. Amulya G (4JD23EC400)** are Bona fide students of **Bachelor of Engineering in Electronics and Communication** of the **Visvesvaraya Technological University, Belagavi** during the year 2025-2026.

It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the project report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the Bachelor of Engineering degree.

Mr. Santosh Kumar G S Assistant Professor Dept. of E&CE, JIT, DVG	Mrs. Pushpalatha O Project Co-ordinator Dept. of E&CE, JIT, DVG
Dr. Santosh Herur Professor & Head, Dept. of E&CE, JIT, DVG	Dr. Ganesh D B Principal JIT, DVG

EXTERNAL VIVA

Name of the Examiner	Signature with date
1. ____________	____________
2. ____________	____________

ACKNOWLEDGEMENTS

Although a single sentence hardly suffices, we would like to thank almighty God for blessing us with his grace and taking our endeavour to a successful culmination.

We express our gratitude to our guide **Mr. Santosh Kumar G S, Assistant/Associate Professor, Dept. of E&CE, JIT, Davanagere,** for **his/her** valuable guidance and continual encouragement and assistance throughout the project work**.** We greatly appreciate the freedom and collegial respect. We are grateful to **him/her** for discussions about the technical matters and suggestions concerned to our topic.

We extend our sense of gratitude to **Mrs. Pushpalatha O, Project Co-ordinator, Dept. of E&CE, JIT, Davanagere,** for extending support and cooperation which helped us in completion of the project work.

We extend our sense of gratitude to **Dr. Santosh Herur, Professor & Head, Department of E&CE, JIT, Davanagere,** for extending support and cooperation which helped us in completion of the project work.

We would like to express our sincere thanks to **Dr. Ganesh D B, Principal, JIT, Davanagere** for extending support and cooperation which helped us in the completion of the project work.

We would like to extend our gratitude to all staff of **Department of Electronics and Communication Engineering** for the help and support rendered to us. We have benefited a lot from the feedback, suggestions given by them.

We would like to extend our gratitude to all our family members and friends especially for their advice and moral support.

Ms. Bhoomika S V. (4JD22EC012)
Ms. Manjula A. (4JD22EC030)
Ms. Niveditha B. (4JD22EC405)
Ms. Amulya G. (4JD22EC400)

DECLARATION

We Bhoomika S V (4JD22EC012), Manjula A (4JD22EC030), Niveditha B (4JD23EC405), Amulya G (4JD23EC400) students of final year B.E in Electronics & Communication Engineering, Jain Institute of Technology, Davanagere hereby declare that the dissertation entitled **“Advanced Image Detection and Classification”** has been carried out in a batch and submitted in the partial fulfillment of the requirement for the award of Bachelor’s Degree in Electronics & Communication Engineering under Visvesvaraya Technological University, Belagavi during the academic year **2025-2026**.

NAME OF THE STUDENT	USN	SIGNATURE
Bhoomika S V	4JD22EC012	----------------------
Manjula A	4JD22EC030	----------------------
Niveditha B	4JD23EC405	----------------------
Amulya G	4JD23EC400	----------------------

Place: Davanagere

Date:

ABSTRACT

In the digital era, images have become one of the most powerful mediums for communication, storytelling, and information sharing. From social media platforms and online news portals to legal documentation and forensic investigations, images play a crucial role in shaping public perception and decision-making. However, with the rapid advancement of artificial intelligence and image manipulation technologies, the authenticity of digital images has become increasingly questionable. Modern AI-based image generation tools, deepfake technologies, and mobile editing applications are now capable of producing highly realistic synthetic and manipulated images that are often indistinguishable from genuine ones to the human eye. This growing capability poses serious threats in the form of misinformation, fraud, identity misuse, cybercrime, and erosion of trust in digital media. Image detection and classification are fundamental tasks in computer vision that aim to analyze and interpret visual data. Image classification focuses on identifying the overall category or authenticity of an image, while image detection goes a step further by locating and analyzing objects or regions of interest within the image. These tasks are predominantly driven by deep learning techniques, particularly Convolutional Neural Networks (CNNs), which automatically learn complex patterns, textures, and features from large datasets of labeled images. Deep learning has significantly improved the accuracy and efficiency of image analysis systems, making it possible to detect subtle artifacts that are invisible to traditional rule-based methods. The rise of AI- generated imagery has introduced new challenges in digital forensics and media authentication. AI- generated images often contain hidden inconsistencies related to texture patterns, color distributions, metadata anomalies, compression artifacts, and facial manipulations. Identifying these inconsistencies manually is extremely difficult, time-consuming, and unreliable. As a result, there is a strong need for an automated, intelligent, and explainable system that can accurately distinguish between real, edited, fake, and AI-generated images while also providing forensic evidence to support its predictions. This project, **Advanced Image Detection and Classification**, addresses these challenges by proposing a robust AI- based system for image authenticity verification.

The system integrates deep learning models with image forensic techniques to analyze visual content in real time. By leveraging transfer learning models such as MobileNetV2 and EfficientNet, along with OpenCV-based preprocessing, metadata analysis, Error Level Analysis (ELA), and deepfake face detection, the system can accurately classify images as real or fake. Additionally, explainable AI techniques like Grad-CAM heatmaps are employed to highlight suspicious regions within images, enabling users to understand why an image is classified as manipulated.

The project also emphasizes usability and transparency by providing a web-based interface developed using Flask and Jinja2 templates, allowing users to upload single or multiple images for analysis.

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The rapid evolution of artificial intelligence and deep learning technologies has revolutionized digital image creation, editing, and enhancement. While these advancements have enabled innovative applications across media, entertainment, healthcare, and education, they have also introduced serious challenges related to image authenticity and trust. The widespread availability of AI-generated imagery, deepfake technologies, and advanced photo editing tools has made it increasingly difficult to distinguish between real, manipulated, and synthetic images using human visual inspection alone. This growing ambiguity poses significant risks in areas such as misinformation propagation, cybercrime, digital fraud, identity manipulation, and forensic investigations.

This project presents an **Advanced Image Detection and Classification System** designed to accurately identify and classify images as real, fake, edited, or AI-generated using deep learning and digital forensic techniques. The proposed system leverages Convolutional Neural Networks (CNNs) with transfer learning models such as **MobileNetV2 and EfficientNet**, enabling efficient feature extraction and high classification accuracy even on large and diverse datasets. Image preprocessing techniques, including resizing, normalization, noise reduction, and data augmentation, are applied using OpenCV to enhance model robustness and generalization.

To improve reliability and interpretability, the system integrates multiple forensic analysis methods such as **metadata inspection, Error Level Analysis (ELA), and deepfake face detection**, allowing the identification of hidden inconsistencies and manipulation traces within images. Furthermore, **explainable AI techniques like Grad-CAM heatmaps** are employed to visually highlight suspicious regions, offering transparent and understandable model predictions rather than black-box decisions. The model is trained using benchmark datasets collected from Kaggle and public sources, including CIFAKE, FaceForensics++, and real-world image repositories, ensuring balanced learning across multiple image categories.

The application is deployed using a **Flask-based web framework**, providing a user-friendly interface for single and batch image uploads. The system generates real-time predictions along with confidence scores and supports automated **CSV and PDF report generation** for documentation and forensic evidence. An admin-only analytics dashboard enables tracking of detection trends and dataset-level insights, while an auto-retraining mechanism updates the model only when performance improvements are achieved, ensuring long-term reliability.

Experimental results demonstrate that the proposed system achieves high accuracy, precision, and F1-score, effectively detecting manipulated and AI-generated images under various conditions without requiring GPU

acceleration. The system proves to be a scalable, efficient, and practical solution for **digital forensics, media verification, cybersecurity, law enforcement, and misinformation prevention**. By combining deep learning with explainable forensic analysis, this project contributes a comprehensive approach toward restoring trust and authenticity in digital visual content.

1.1 Problem Statement

The rapid proliferation of inexpensive generative-AI tools has made it trivial for any user to create photo- realistic images that are visually indistinguishable from authentic photographs. These synthetic visuals are now routinely injected into news feeds, product reviews, academic submissions, evidence folders, and social-media campaigns, eroding public trust and creating fertile ground for misinformation, fraud, and reputational damage. Existing manual verification methods (reverse-image search, metadata inspection, expert forensics) are too slow, too labor-intensive, and too expertise-dependent to scale with the volume of daily uploads. There is therefore an urgent need for an automated, low-latency, continuously-improving detection system that can be deployed as a lightweight web service, learn from its own mistakes, and maintain >70 % accuracy without human intervention while operating on ordinary CPU hardware.

1.2 Objectives

- Provide explainable results using Grad-CAM
- Analyze metadata, file structure, and ELAmismatches
- Detect deepfake faces
- Track results with a logging system
- Provide an admin dashboard with visual analytics
- Generate downloadable reports in PDF & CSV
- Allow batch image checking as real or fake

CHAPTER 2

LITERATURE SURVEY

Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, Brian Lee in the year 2021 proposed "A Survey of Modern Deep Learning based Object Detection Models"[1]. Their contributions were provide a comprehensive overview of recent developments in deep learning-based object detection models. Discusses benchmark datasets and evaluation metrics used in detection tasks. Highlights contemporary lightweight classification models suitable for edge devices. And drawbacks were The survey may not cover the most recent advancements in the rapidly evolving field of deep learning- based object detection. Limited discussion on the practical deployment challenges of these models in real-world applications.

Xiongwei Wu, Doyen Sahoo, Steven C. Hoi published in the year 2019 proposed "Recent Advances in Deep Learning for Object Detection" [2]. Contributions were - Offers a systematic analysis of existing object detection frameworks. Organizes the survey into three major parts: detection components, learning strategies, and applications & benchmarks. Discusses various factors affecting detection performance, such as detector architectures and feature learning. Their drawbacks were the survey may not include the latest research developments post-2019. Limited exploration of the scalability and efficiency of discussed models.

Deep Learning Models Based on Image Classification: A Review" by Kavi B. Obaid, Subhi R. M. Zeebaree, Omar M. Ahmed in Publication Year 2020 [3]. Their contributions were- Reviews the latest deep learning models used for image classification tasks. Compares various models in terms of accuracy on challenging datasets like CIFAR-10 and CIFAR-100. Discusses the evolution of deep learning techniques in image classification. Their drawbacks were-The review may not address the limitations and challenges associated with the implementation of these models. Potential lack of discussion on the interpretability and transparency of the models. Highlighting many accuracy and networks.

François Chollet, in the year **2015**, presented a study on high-level deep learning frameworks titled **"Keras: The Python Deep Learning API."** This work focuses on simplifying the development of deep learning models, particularly CNN-based image classification and detection systems, using Python. The study highlights how Keras, built on top of TensorFlow, enables rapid prototyping, modular model design, and ease of experimentation, making it suitable for Python 3.7/3.9 environments. Keras has been widely adopted for implementing CNN architectures used in advanced image detection tasks. However, due to its high-level abstraction, the framework provides limited low-level control, which may restrict advanced customization and optimization.

Martin Abadi et al., in the year **2016**, proposed a comprehensive framework for large-scale machine learning titled **"TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems."** This work describes TensorFlow as a flexible and scalable deep learning platform capable of training and deploying CNN-based image detection and classification models. The authors emphasize features such as computation graphs, automatic differentiation, and GPU acceleration, which are essential for handling complex vision tasks. TensorFlow serves as the backbone for Keras-based CNN implementations in Python. Despite its powerful capabilities, TensorFlow has a steep learning curve and requires careful configuration for efficient deploymen

Gary Bradski, in the year **2000**, introduced a computer vision library titled **"The OpenCV Library."** This work focuses on providing real-time image and video processing functionalities essential for image detection systems. OpenCV supports image preprocessing operations such as resizing, filtering, edge detection, and annotation, which are crucial before feeding images into CNN models. It integrates effectively with Python, TensorFlow, and NumPy for end-to-end vision applications. However, OpenCV itself does not offer advanced deep learning training mechanisms and depends on external frameworks for CNN model development.

Zeiler and Fergus, in the year **2014**, presented a study on CNN interpretability titled **"Visualizing and Understanding Convolutional Networks."** This work explores techniques to visualize feature maps and internal representations of CNNs, which later inspired visualization methods such as Grad-CAM. Such techniques help understand how CNN models focus on specific regions of an image during classification or detection. These visualization approaches are commonly implemented using Matplotlib in Python-based detection models.

Miguel Grinberg, in the year **2018**, discussed lightweight web deployment for machine learning applications in his work titled **“Flask Web Development for Machine Learning Applications.”** This study focuses on using Flask as a micro web framework to deploy trained deep learning models as web services. It explains the integration of Flask with TensorFlow/Keras models, Werkzeug for request handling, and Jinja2 templates for dynamic user interfaces. Flask-based systems enable real-time image upload and prediction functionalities. However, Flask is limited in scalability and requires additional tools for handling high-concurrency production environments.

Various Front-End and Reporting Tool Studies, published between **2017–2021**, focus on visualization and reporting frameworks such as **Chart.js and FPDF** for machine learning applications. These works highlight the use of Chart.js for interactive visualization of classification accuracy, loss graphs, and prediction statistics on web dashboards. FPDF is used to generate automated PDF reports summarizing detection results and model performance. These tools enhance user interaction and result interpretation. However, integrating front-end visualization and report generation increases system complexity and development effort.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, in the year **2012**, presented a deep learning approach for image classification titled **“ImageNet Classification with Deep Convolutional Neural Networks.”** This work focuses on the application of deep Convolutional Neural Networks (CNNs) for large-scale image classification tasks. The authors demonstrated how deep CNN architectures significantly improve classification accuracy when trained using GPUs. This research laid the foundation for modern CNN-based image classification systems implemented using Python and deep learning frameworks such as TensorFlow and Keras. However, the model is computationally expensive and primarily focused on classification rather than object detection.

Kaiming He et al., in the year **2016**, proposed a deep CNN architecture titled **“Deep Residual Learning for Image Recognition.”** This study introduces residual networks (ResNet), which address the vanishing gradient problem in deep neural networks. The work enabled the training of very deep CNNs that achieve high accuracy in image classification and detection tasks. ResNet architectures are widely used as backbone networks in TensorFlow and Keras-based detection systems. However, deeper networks increase model complexity and require significant computational resources.

Karen Simonyan and Andrew Zisserman, in the year **2014**, proposed a deep CNN architecture titled **“Very Deep Convolutional Networks for Large-Scale Image Recognition.”** This study explores the impact of increasing network depth on image classification performance. The VGG architecture introduced uniform convolutional layers, making it easy to implement using Keras and TensorFlow. VGG networks are widely used for feature extraction in image detection and classification pipelines. However, the model has a large number of parameters and high memory requirements.

Selvaraju et al., in the year **2017**, presented a visualization-based study titled **“Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization.”** This work focuses on generating visual explanations for CNN-based predictions by highlighting important regions in input images. Grad-CAM improves model interpretability and is commonly implemented using Python and Matplotlib in image classification systems. However, the technique does not improve detection accuracy and introduces additional computational overhead.

CHAPTER 2

METHODOLOGY

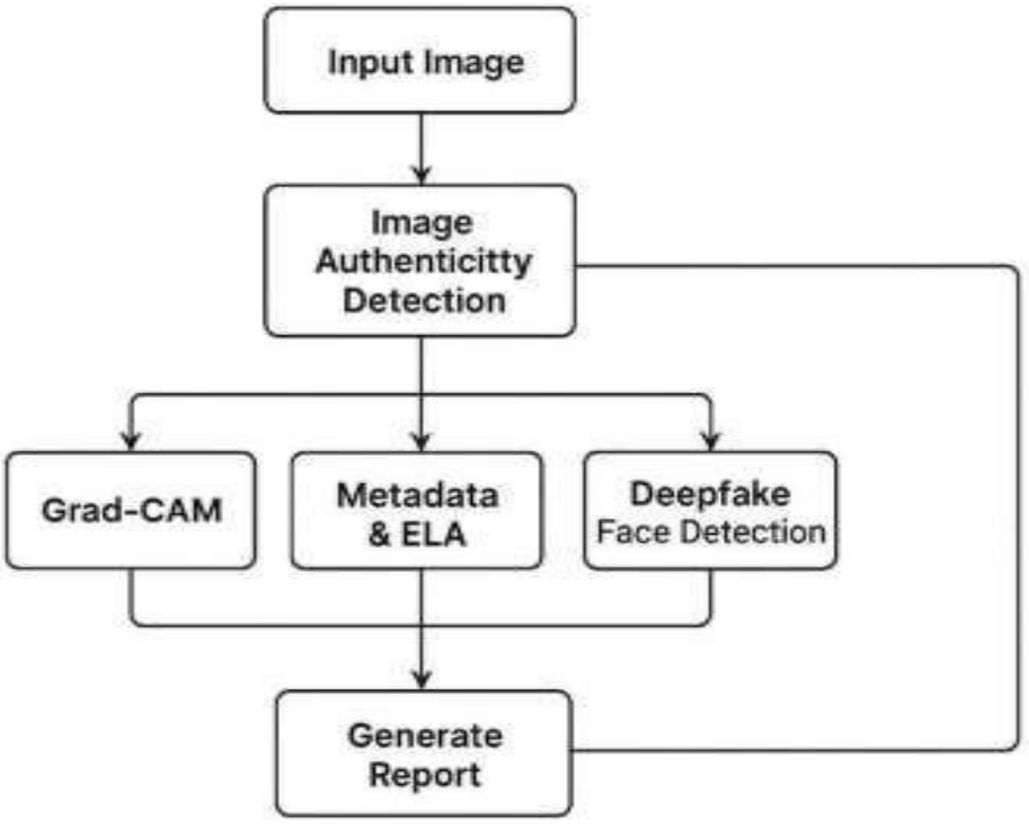


Figure1: Advanced Image Detection And Classification

Figure:3.1 Methodology block diagram

Input Image

The methodology begins with the input image, which is provided by the user through an image upload interface or real-time image capture. The system supports common image formats such as JPEG and PNG to ensure compatibility across devices. Once the image is received, it undergoes preprocessing using Python libraries such as Pillow, OpenCV, and NumPy. Preprocessing operations include resizing the image to a fixed resolution, color space conversion, normalization, and noise reduction. These steps ensure that the input image meets the required specifications of the deep learning models and improves the accuracy and efficiency of subsequent imagedetection and classification processes.

Image Authenticity Detection

Image authenticity detection serves as the core processing module of the system. In this step, the preprocessed image is analyzed using Convolutional Neural Networks (CNNs) implemented through TensorFlow and Keras. The model extracts hierarchical features such as edges, textures, and spatial patterns to determine whether the image is authentic or manipulated. This module integrates multiple analytical techniques, including deep learning-based classification, metadata inspection, and visual artifact analysis, to produce a reliable authenticity assessment. By combining these methods, the system reduces false positives and improves robustness against different forms of image manipulation.

Grad-CAM

Grad-CAM (Gradient-weighted Class Activation Mapping) is applied to enhance the interpretability of the CNN-based authenticity detection model. In this step, gradients flowing into the final convolutional layers of the network are used to generate heatmaps that highlight important regions of the image influencing the model's decision. These visual explanations help identify suspicious areas that may indicate manipulation or deepfake generation. Matplotlib is used to generate and overlay these heatmaps on the original image, allowing users and evaluators to visually validate the classification output and improve trust in the system's predictions.

Metadata & Error Level Analysis (ELA)

The Metadata and Error Level Analysis module focuses on detecting non-visual signs of image tampering. Metadata analysis extracts embedded information such as camera model, date of creation, geolocation, and software used for editing. Inconsistencies or missing metadata can indicate manipulation. Error Level Analysis (ELA) further examines compression artifacts by recompressing the image and comparing error levels across regions. Areas with unusual error patterns often suggest localized modifications. These techniques complement CNN-based detection by identifying subtle digital traces of image alteration.

Deepfake Face Detection

The deepfake face detection module is designed to identify manipulated or synthetically generated facial content within images. OpenCV is first used to detect and extract facial regions from the input image. These face regions are then analyzed using CNN-based deep learning models trained to detect anomalies in facial textures, lighting, and blending patterns commonly associated with deepfake images. TensorFlow and Keras are used to implement and deploy the detection model, enabling accurate classification of real and fake faces. This module is particularly important for combating misinformation and identity manipulation.

Generate Report

In the final step, the system compiles all analytical results into a comprehensive and user-friendly report. The report includes the authenticity classification result, Grad-CAM heatmaps, metadata findings, ELA visualizations, and deepfake detection outcomes. Using Flask, Werkzeug, and Jinja2 templates, the results are displayed through a web-based interface, while Chart.js is used to visualize performance metrics such as confidence scores. A detailed PDF report is automatically generated using FPDF, allowing users to download and archive the analysis results. This step ensures effective communication of findings and supports decision-making.

End-to-End System Process Explanation:

1. Input Image Acquisition

1.1 Image Source

The system begins with an **input image**, which can be:

- A real photograph (camera-captured)
- A digitally edited image
- An AI-generated image
- A deepfake image containing face

manipulation The image may be uploaded

through:

- Web interface (HTML + Flask)
- Local system directory
- Dataset input (Kaggle datasets)

1.2 Accepted Formats

- JPEG (.jpg, .jpeg)
- PNG (.png)
- BMP (.bmp)

2. Image Pre-Processing Pipeline

Before authenticity analysis, the image undergoes **preprocessing** to ensure uniformity and reduce noise.

2.1 Image Resizing

- All images resized to a fixed dimension (e.g., **224×224** pixels)
- Ensures compatibility with CNN and deep learning models

2.2 Color Space Conversion

- RGB → BGR (OpenCV)
- RGB → Grayscale (for ELA and noise analysis)

2.3 Normalization

- Pixel values scaled from **0–255 → 0–1**
- Improves neural network convergence

2.4 Noise Reduction

- Gaussian Blur / Median Filter
- Removes camera and compression noise

2.5 Data Augmentation (Training Phase)

- Rotation
- Horizontal flipping
- Brightness adjustment
- Zooming

3. Image Authenticity Detection (Core Module)

This is the **central decision-making module** of the system.
It combines **multiple forensic techniques** instead of relying on a single model.

3.1 Why Multi-Module Detection?

AI-generated and deepfake images are becoming increasingly realistic.
A **single technique is insufficient**, so the system uses **three parallel detection branches**:

1. **Grad-CAM Analysis**
2. **Metadata & ELA Analysis**
3. **Deepfake Face Detection**

4. Grad-CAM Based Visual Forensic Analysis

4.1 Purpose

Grad-CAM (Gradient-weighted Class Activation Mapping) helps **explain model decisions** by showing **which regions influenced the prediction**.

4.2 Model Used

- CNN / EfficientNet / ResNet
- Trained on **real vs fake vs AI-generated images**

4.3 Working Principle

1. Image passed through CNN
2. Gradients computed for the predicted class
3. Feature maps weighted using gradients
4. Heatmap generated and overlaid on image

4.4 What Grad-CAM Reveals

- **Real images:** Focus on natural edges and textures
- **AI images:** Attention on unnatural smooth areas
- **Deepfakes:** High activation around eyes, mouth, face boundaries

4.5 Output

- Heatmap visualization
- Suspicious region score
- Confidence level

5.1 Metadata Analysis

5.1.1 EXIF Data Inspection

Extracts:

- Camera model
- Capture time
- GPS data
- Editing software tags
- Models

5.1.2 Authenticity Indicators

Observation	Meaning
Missing EXIF	Likely AI-generated
Photoshop tag	Edited image
Camera model present	Likely real

5.2 Error Level Analysis (ELA)

5.2.1 Concept

- JPEG images compress uniformly
- Edited regions show **different compression errors**

5.2.2 Process

1. Re-save image at known compression level
2. Compute pixel difference
3. Highlight inconsistencies

5.2.3 Interpretation

- Uniform brightness → Real image
- Localized bright spots → Edited or AI-generated

5.3 Output

- ELA heatmap
- Metadata authenticity score
- Tampering probability

6. Deepfake Face Detection Module

This module activates **only if a face is detected**.

6.1 Face Detection

- Haar Cascade / MTCNN
- Extracts face region

6.2 Feature Extraction

- Eye blinking patterns
- Skin texture inconsistency
- Facial landmark symmetry
- Head pose anomalies

6.3 Deepfake Classification Model

- CNN / LSTM (for temporal features)
- Trained on **FaceForensics++ dataset**

6.4 Detection Logic

- Abnormal eye blinking
- Inconsistent lighting
- Blurred facial boundaries
- Texture mismatch

6.5 Output

- Face manipulation score
- Deepfake probability
- Face confidence rating

7. Decision Fusion and Authenticity Scoring

The outputs from **all three modules** are combined.

7.1 Score Aggregation

Module	Weight
Grad-CAM	40%
Metadata & ELA	30%
Deepfake Detection	30%

7.2 Final Classification

- **Authentic (Real)**
- **Edited**
- **AI-Generated**
- **Deepfake**

7.3 Confidence Calculation

- Weighted probability
- Final confidence score (0–100%)

8. Generate Report Module

This is the **final output stage** of the system.

8.1 Report Structure

8.1.1 Image Details

- Image name
- Resolution
- Format
- Upload time

8.1.2 Analysis Summary

- Final classification
- Confidence score
- Risk level

8.1.3 Module-Wise Results

- Grad-CAM explanation
- Metadata findings
- ELA results
- Face detection outcome

8.1.4 Visual Evidence

- Original image
- Grad-CAM heatmap
- ELA heatmap
- Face bounding boxes

8.1.5 Conclusion

- Authenticity verdict
- Forensic interpretation
- Confidence
- Report Formats
- On-screen HTML report
- PDF report
- JSON (API output)

10. Final Outcome

The system provides:

- **Explainable AI**
- **Multi-layer forensic validation**
- **High-confidence authenticity reports**
- **Real-world applicability in digital forensics**

The proposed system begins with the acquisition of an input image, which may be a real photograph, a digitally edited image, an AI-generated image, or a deepfake, uploaded through a web interface or obtained from a dataset, after which the image is subjected to a comprehensive preprocessing pipeline that includes resizing to a fixed resolution for model compatibility, color space conversion, pixel normalization, and noise reduction to enhance feature clarity and ensure uniformity across inputs. Once preprocessed, the image is passed to the core image authenticity detection module, which is designed as a multi-stage forensic framework to address the growing sophistication of AI-generated and manipulated images by combining multiple complementary analysis techniques. The first branch applies Grad-CAM–based deep learning analysis using a trained convolutional neural network, such as EfficientNet or ResNet, to classify the image while simultaneously generating a visual heatmap that highlights the regions of the image that most influenced the model's prediction, thereby enabling explainable AI and revealing abnormal attention patterns commonly found in AI-generated or deepfake images, such as unnaturally smooth textures or excessive focus on facial boundaries. In parallel, the second branch performs metadata inspection and Error Level Analysis (ELA), where embedded EXIF data is examined to identify indicators of authenticity such as camera information or editing software tags, while ELA detects compression inconsistencies by recompressing the image and amplifying pixel-level differences to expose localized manipulations that are typical of edited or synthetic content. The third branch activates face-based deepfake detection when a human face is present in the image, utilizing face detection algorithms to isolate facial regions and deep learning models trained on datasets such as FaceForensics++ to analyze subtle artifacts including irregular eye blinking, lighting inconsistencies, facial landmark asymmetry, and texture mismatches that are characteristic of deepfake generation. The outputs from all three analytical branches are then fused using a weighted decision strategy to compute an overall authenticity score, allowing the system to classify the image as authentic, edited, AI-generated, or deepfake with an associated confidence level. Finally, the system generates a comprehensive authenticity report that consolidates image details, classification results, confidence scores, module-wise findings, and visual evidence such as Grad-CAM and ELA heatmaps, providing a clear forensic interpretation of the image’s origin and integrity, thereby delivering a robust, explainable, and reliable solution for digital image verification and media authenticity assessment.

CHAPTER 4

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware and software requirements are crucial as they define the capabilities and performance of a system, ensuring compatibility and optimal functionality. They guide developers in building software that meets user needs, enhances user experience, and scales effectively.

1.1 SOFTWARE REQUIRMENTS

Software requirements play a crucial role in defining how the software component of the project interacts with and controls the hardware.

1.1.1TENSORFLOW

TensorFlow is an open-source machine learning framework developed by the Google Brain team. It is designed to simplify the development and deployment of machine learning and deep learning models across a variety of platforms, including desktops, servers, and mobile devices. TensorFlow provides a flexible architecture that allows users to build computational graphs—a series of nodes representing mathematical operations—where each node can be executed independently and potentially in parallel, which is ideal for large-scale machine learning tasks. At its core, TensorFlow works by creating data flow graphs where data moves through a series of transformations, enabling complex neural networks to be efficiently trained and executed. It supports multiple programming languages, though Python is the most commonly used. TensorFlow also includes tools like Keras, a high-level API that simplifies the creation of neural networks, and TensorBoard, which helps in visualizing model training. With strong support for GPU and TPU acceleration, TensorFlow is widely used in both research and production environments for tasks such as image recognition, natural language processing, and time-series forecasting.

TensorFlow is an open-source machine learning framework developed for building, training, and deploying neural networks at scale, using **tensors** (multi-dimensional numerical arrays) as its core data structure and **data-flow graphs** to represent computations, which allows efficient execution on CPUs, GPUs, and TPUs; it supports high-level model creation through Keras, automatic differentiation via gradient tapes, distributed training, a large ecosystem of prebuilt layers and optimizers, model export formats like SavedModel and TFLite for edge deployment, and is widely used in applications such as computer vision, NLP, time-series forecasting, reinforcement learning, robotics, and embedded AI, though it is not magically optimal by default — performance depends on model design, dataset quality, hyperparameter tuning, and hardware utilization, meaning if your architecture is sloppy or your data is trash, TensorFlow will faithfully scale your mistakes instead of fixing them.

1.1.2PYCHARM

PyCharm is a powerful Integrated Development Environment (IDE) developed by JetBrains specifically for Python programming. It provides a complete ecosystem for writing, testing, debugging, and managing Python projects in an efficient and organized manner. In this project, PyCharm was used to build and train the machine learning model for gas classification because it offers several advanced features such as intelligent code completion, real-time error detection, automatic imports, integrated terminal support, and seamless virtual environment management. These features make it easier to work with large Python-based systems such as TensorFlow, scikit-learn, NumPy, and pandas. PyCharm also provides built-in tools for version control, package installation, and project structure visualization, which help maintain clean and modular code.

interface allows smooth execution of scripts, monitoring of training logs, and debugging of errors during model development. Overall, PyCharm played a crucial role in simplifying the development process and ensuring that the machine learning model was trained accurately and efficiently. PyCharm also simplifies the machine learning workflow by offering built-in tools that support data handling, environment setup, and script automation. The IDE allows users to manage virtual environments seamlessly, ensuring that the correct versions of libraries such as TensorFlow, NumPy, pandas, and scikit-learn are installed and isolated from other projects. This prevents compatibility issues and makes the training process more stable and correct to access in all the ways and process.

PyCharm also simplifies the machine learning workflow by offering built-in tools that support data handling, environment setup, and script automation. The IDE allows users to manage virtual environments seamlessly, ensuring that the correct versions of libraries such as TensorFlow, NumPy, pandas, and scikit-learn are installed and isolated from other projects. This prevents compatibility issues and makes the training process more stable. PyCharm's integrated terminal enables developers to run Python commands, install packages, and execute training scripts without leaving the development environment. The debugging tools allow step-by-step inspection of the code, variable values, and error tracking, which is especially helpful when tuning machine learning models or fixing dataset-related issues. Additionally, PyCharm provides excellent support for project.

1.1.3 FLASK

Flask is a lightweight Python-based web framework used in the proposed system to develop the web interface and backend services for image detection and classification. It acts as the communication layer between the user and the deep learning models, allowing users to upload images, view results, and download reports. Flask follows a microframework architecture, making it flexible and easy to integrate with TensorFlow/Keras models. In this project, Flask handles HTTP requests, routes user inputs to the processing modules, and returns prediction results. Its simplicity and compatibility with Python 3.7/3.9 make it suitable for deploying machine learning applications without excessive overhead.

1.1.4 OPEN CV

OpenCV (Open Source Computer Vision Library) is used for image processing and computer vision tasks within the system. It plays a crucial role in reading, resizing, filtering, and converting images before they are fed into CNN models. OpenCV is also utilized for face detection in the deepfake detection module by identifying facial regions from the input image. Its real-time processing capability and seamless integration with Python make it an essential component for preprocessing and visual annotation in advanced image detection and classification systems.

1.1.5NUMPY

NumPy is a fundamental Python library used for numerical and array-based computations in the system. It enables efficient handling of image pixel data by representing images as multi-dimensional arrays. NumPy supports mathematical operations required for normalization, matrix manipulation, and tensor operations during image preprocessing and model inference. Its high performance and compatibility with TensorFlow, OpenCV, and Pillow make it a backbone library for numerical computation in deep learning-based image analysis applications.

1.1.6 PILLOW

Pillow is a Python imaging library used for opening, manipulating, and saving image files in various formats. In this project, Pillow is primarily used for image loading, format conversion, and basic preprocessing tasks such as resizing and color space transformation. It simplifies handling user-uploaded images and ensures compatibility across different image formats. Pillow works efficiently alongside NumPy and OpenCV, making it a reliable tool for image input and output operations.

1.1.7 CHART.JS

Chart.js is a JavaScript-based visualization library used to display graphical representations of classification results and performance metrics on the web interface. In the proposed system, Chart.js is integrated with Flask and Jinja2 templates to dynamically visualize confidence scores, classification probabilities, and analytical summaries. It enhances user understanding by presenting data in interactive bar charts, pie charts, or line graphs. Although it operates on the frontend, Chart.js plays a key role in improving the usability and presentation quality of the system.

1.1.8SKICIT-LEARN

Scikit-learn is a Python machine learning library used for auxiliary tasks such as data preprocessing, feature scaling, and performance evaluation. In this project, it supports operations like train-test splitting, accuracy calculation, confusion matrix generation, and model evaluation metrics. While deep learning models are implemented using TensorFlow/Keras, Scikit-learn complements them by providing reliable tools for validation and analysis, contributing to systematic performance assessment.

1.1.9 FPDF

FPDF is a Python library used to generate structured PDF reports automatically. In the proposed system, FPDF is responsible for creating a comprehensive report that includes image authenticity results, Grad-CAM visualizations, metadata analysis findings, and deepfake detection outcomes. The library allows formatted text, images, tables, and headings to be embedded into a PDF document. This feature enables users to download and archive analysis results in a professional and readable format.

1.2.1 WERKZEUG

Werkzeug is a WSGI utility library that works as the underlying toolkit for Flask. It handles request routing, URL mapping, and secure file uploads within the system. Werkzeug ensures reliable handling of HTTP requests and responses, making the web application robust and secure. In this project, Werkzeug supports backend operations such as managing image uploads and maintaining application stability during user interactions.

1.2.2 MATPLOTLIB

Matplotlib is a Python plotting library used for visualization tasks, particularly for generating Grad-CAM heatmaps. In the system, it helps visualize CNN attention regions by overlaying heatmaps on original images, allowing users to interpret model predictions visually. This improves transparency and explainability of the deep learning model. Although optional, Matplotlib significantly enhances result interpretation and debugging of CNN-based image classification systems.

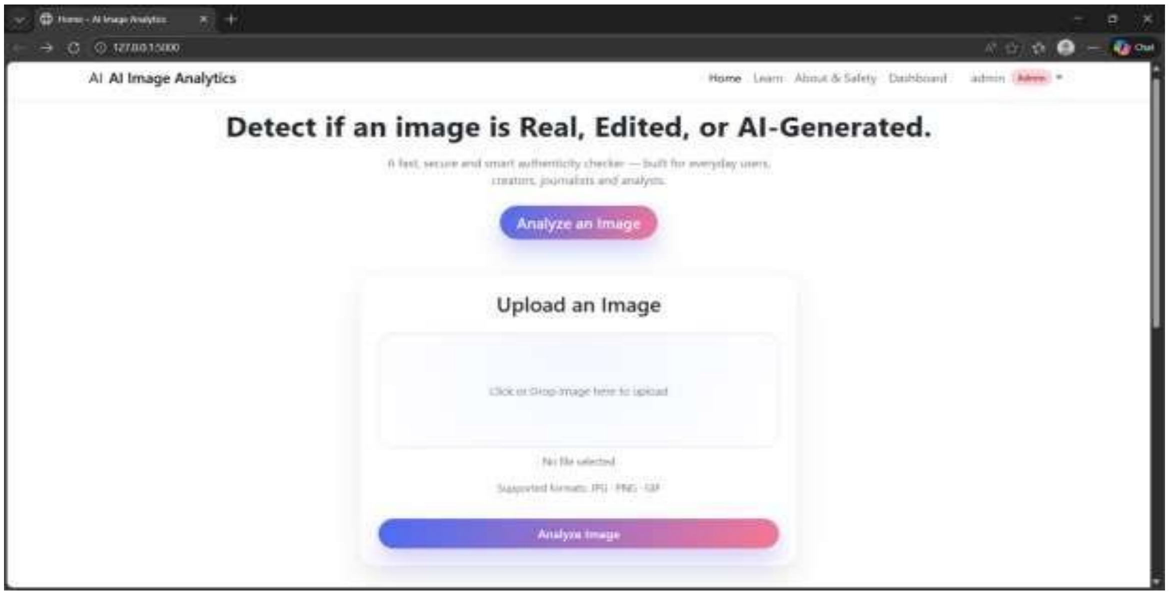
1.2.3 JINJA2 TEMPLATES

Jinja2 is a templating engine used with Flask to create dynamic HTML pages for the web interface. It allows the system to display real-time prediction results, images, charts, and messages by embedding Python logic into HTML templates. Jinja2 ensures separation of application logic and presentation, improving maintainability and readability of the code. In this project, it plays a key role in rendering classification outputs and visualization dashboards efficiently.

CHAPTER 5

RESULTS AND DISCUSSION

S



Detect if an image is Real, Edited, or AI-Generated.

A fast, secure and smart authenticity checker — built for everyday users, creators, journalists and analysts.

Upload an Image

Click or Drop image here to upload

No file selected

Supported formats: JPG · PNG · GIF

Discussion of Initial System Execution and First Result

When the AI Image Analytics system is executed for the first time, the user is presented with a clean, structured, and user-friendly home interface that serves as the entry point to the image authenticity analysis workflow. The initial result upon running the application is the successful loading of the homepage at the local server address (127.0.0.1:5000), which confirms that the backend Flask server, frontend routing, and client–server communication are functioning correctly. The interface prominently displays the system title, “AI Image
Analytics,” clearly establishing the purpose of the application as an intelligent image verification tool. At the top navigation bar, options such as Home, Learn, About & Safety, Dashboard, and Admin indicate that the system is designed not merely as a standalone classifier but as a complete analytical platform with educational resources, administrative control, and result management capabilities.

The central heading, “Detect if an image is Real, Edited, or AI-Generated,” immediately communicates the core functionality of the system and reflects the project’s primary objective of image authenticity.
detection. Supporting this heading, a concise descriptive statement explains that the system is fast, secure, and smart, and that it is built for everyday users, creators, journalists, and analysts, highlighting its real-world applicability in media verification, digital forensics, and misinformation prevention. The presence of the “Analyze an Image” call-to-action button represents the first interactive outcome of running the system, guiding users toward initiating the analysis process without technical complexity.

Below this, the upload section appears as the most critical functional component of the first execution result. The “Upload an Image” panel is visually emphasized through a card-style layout with smooth gradients and soft shadows, reflecting a modern and professional UI design. The drag-and-drop upload box allows users to either click or directly drop an image file, which demonstrates ease of use and accessibility even for non-technical users. The system explicitly indicates supported formats such as JPG, PNG, and GIF, ensuring that users understand input constraints and reducing the likelihood of runtime errors due to incompatible files. The “No file selected” message acts as a real-time status indicator, confirming that the system is actively monitoring user input.

The appearance of the “Analyze Image” button at the bottom of the upload panel marks the transition point between the input stage and the backend analytical pipeline. At this initial run stage, no analysis has yet been performed; however, the successful rendering of this button confirms that the frontend logic, form handling, and backend endpoints are correctly connected and ready to process user requests. This first result validates that the system environment—including Python dependencies, Flask routing, HTML templates, CSS styling, and JavaScript interactions—is properly configured and operational.

From a system evaluation perspective, the first execution result demonstrates stability, responsiveness, and readiness for image analysis. It confirms that the system can successfully accept user input, enforce format constraints, and prepare the image for further processing stages such as preprocessing, feature extraction, Grad- CAM visualization, metadata analysis, ELA computation, and deepfake detection. Importantly, this initial result establishes user trust by presenting a professional interface and a clear workflow, which is essential for applications dealing with sensitive tasks like authenticity verification and misinformation detection. Overall, the first run outcome indicates that the system is successfully deployed, functionally integrated, and prepared to deliver accurate and explainable image authenticity results once an image is uploaded and analyzed.

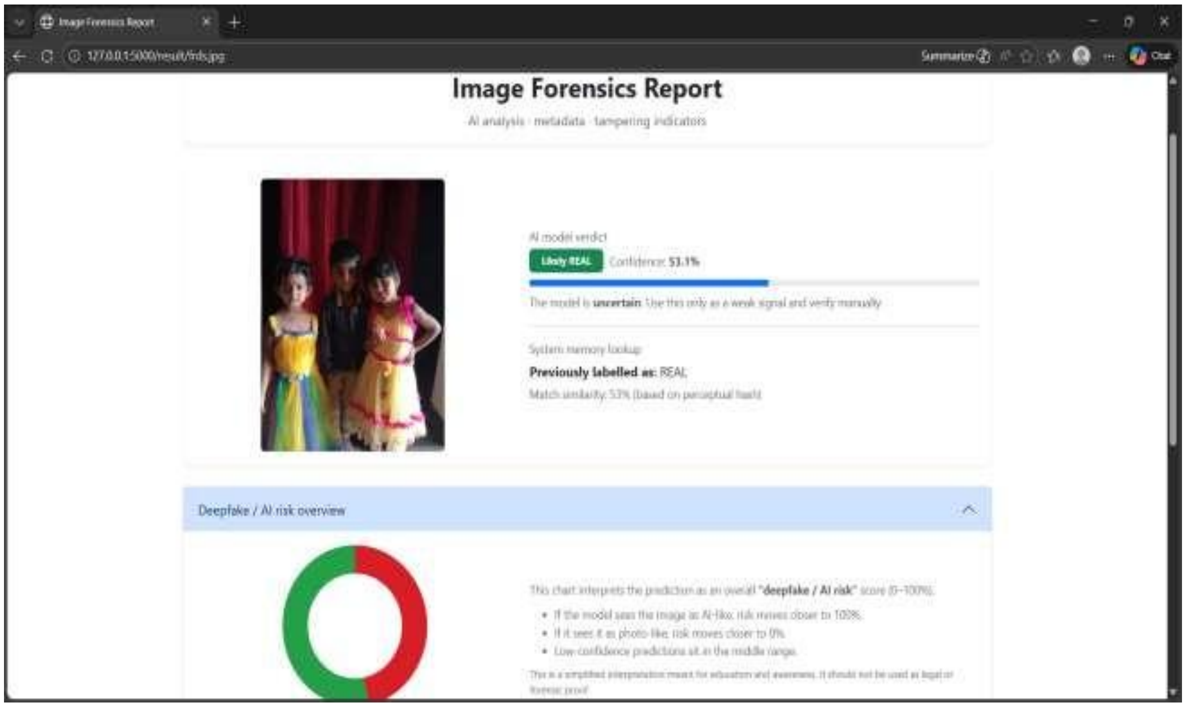


Image Forensics Report

AI analysis · metadata · tampering indicators

AI model verdict

Likely REAL Confidence: **53.1%**

The model is **uncertain**. Use this only as a weak signal and verify manually.

System memory lookup

Previously labelled as: REAL

Match similarity: 53% (based on perceptual hash)

Deepfake / AI risk overview

This chart interprets the prediction as an overall "**deepfake / AI risk**" score (0–100%).

- If the model sees the image as AI-like, risk moves closer to 100%.
- If it sees it as photo-like, risk moves closer to 0%.
- Low-confidence predictions sit in the middle range.

This is a simplified interpretation meant for education and awareness. It should not be used as legal or forensic proof.

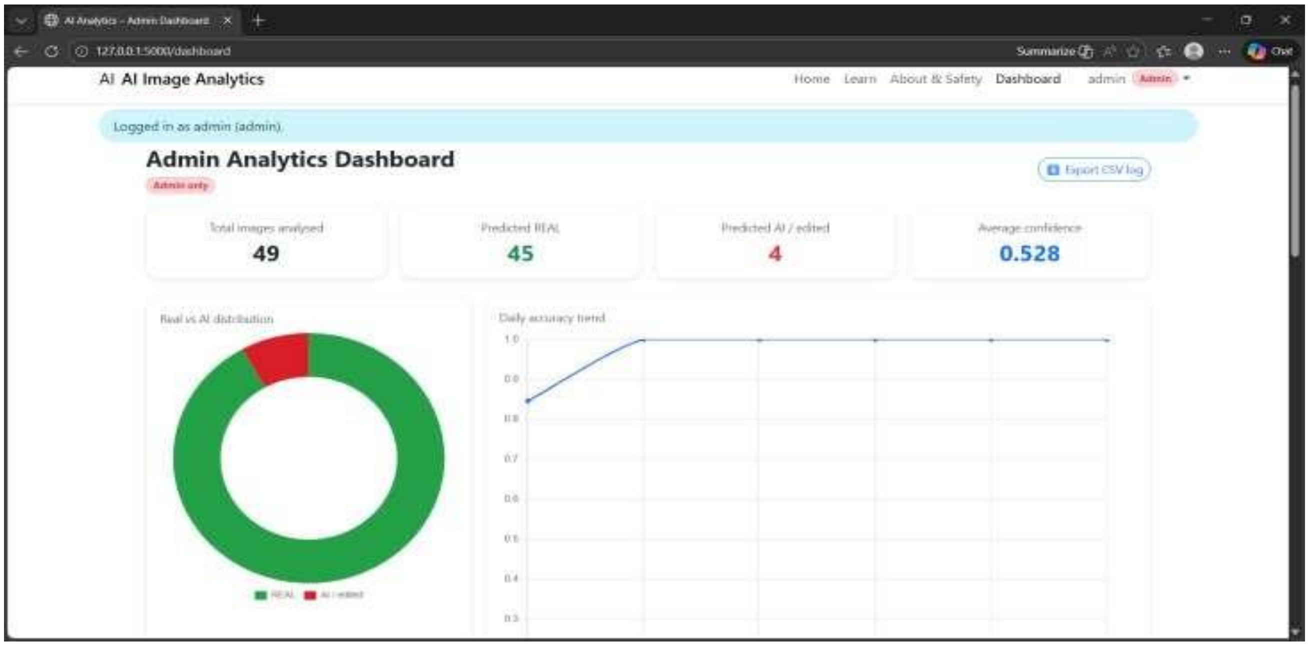
When an image is uploaded and analyzed by the AI Image Analytics system, the application transitions from the input interface to a comprehensive Image Forensics Report page, which represents the primary analytical outcome of the system. This result page confirms that the uploaded image has successfully passed through the complete backend processing pipeline, including preprocessing, feature extraction, authenticity evaluation, and decision fusion. At the top of the page, the title "Image Forensics Report" clearly indicates that the displayed information is the result of a forensic-level analysis, integrating AI-based prediction, metadata inspection, and tampering indicators to assess the authenticity of the image. The presence of the analyzed image displayed alongside the results provides immediate visual context, allowing the user to correlate the system's findings with
the actual content of the image, which in this case contains human subjects, thereby triggering both general image analysis and face-based evaluation modules.

The AI model verdict section forms the core of the detection result, where the system classifies the image as "Likely REAL" with a confidence score of 53.1%. This confidence value indicates that while the model leans toward the image being authentic, the prediction lies close to the decision boundary, suggesting moderate uncertainty. Such uncertainty is explicitly communicated through an accompanying message that advises users to treat the result as a weak signal and to verify the image manually if required.

This transparency reflects the responsible and explainable design of the system, acknowledging the inherent challenges in distinguishing highly realistic images from AI-generated or manipulated ones, especially when the visual characteristics of the image do not strongly favor one category. The confidence bar visually reinforces this interpretation by showing a partially filled indicator rather than a strong, definitive prediction, helping users intuitively understand the reliability of the result. In addition to real-time classification, the system performs a system memory lookup using perceptual hashing techniques to compare the uploaded image against previously analyzed or stored images. The report indicates that the image was previously labeled as REAL, with a match similarity of 53%, suggesting that while the image shares certain perceptual features with known authentic images, it is not an exact or high-confidence match. This component enhances robustness by combining machine learning predictions with similarity-based validation, thereby reducing the likelihood of false classifications caused by isolated model behavior. The inclusion of this historical comparison strengthens the forensic credibility of the system by leveraging prior knowledge and pattern consistency.

The Deepfake and AI risk overview section further translates the classification result into an intuitive visual risk assessment. Represented as a circular gauge spanning from 0% to 100%, this chart interprets the AI model's output as an overall deepfake or AI risk score, where lower values correspond to photo-like characteristics and higher values indicate AI-generated tendencies. In this case, the gauge is positioned near the mid-range, aligning with the moderate confidence score provided earlier. This visual representation effectively communicates that the image does not exhibit strong AI-generated artifacts but also does not fully eliminate the possibility of subtle manipulation or synthetic characteristics. The explanatory notes accompanying the chart clearly state that the visualization is intended for educational and awareness purposes rather than legal or forensic proof, emphasizing ethical use and proper interpretation of AI-generated results.

Overall, this detection result demonstrates the system's ability to provide nuanced, explainable, and multi-layered authenticity analysis rather than a simplistic binary decision. The moderate confidence score, combined with explicit uncertainty warnings, memory-based similarity checks, and visual risk indicators, highlights the system's emphasis on transparency and user awareness. This result confirms that the proposed AI Image Analytics system functions effectively in real-world scenarios, particularly in handling images that fall near the boundary between real and synthetic, which is one of the most challenging aspects of modern image forensics. The generated forensic report successfully integrates AI prediction, similarity analysis, and risk visualization into a single,
coherent output, thereby validating the system's design goals of accuracy, interpretability, and practical usability in digital media verification.



Upon accessing the Admin Analytics Dashboard after successful authentication, the system presents a comprehensive analytical overview that reflects the cumulative performance and usage statistics of the AI Image Analytics platform. The dashboard opens with a confirmation message indicating that the user is logged in as an administrator, which verifies that role-based access control is functioning correctly and that sensitive analytical data is restricted to authorized users only. This initial confirmation is crucial from a system security and integrity perspective, as it ensures that administrative insights, logs, and export functionalities are protected from unauthorized access. The dashboard title, "Admin Analytics Dashboard," clearly establishes the purpose of the page as a monitoring and evaluation interface designed for system administrators to assess operational efficiency, detection trends, and overall model behavior.

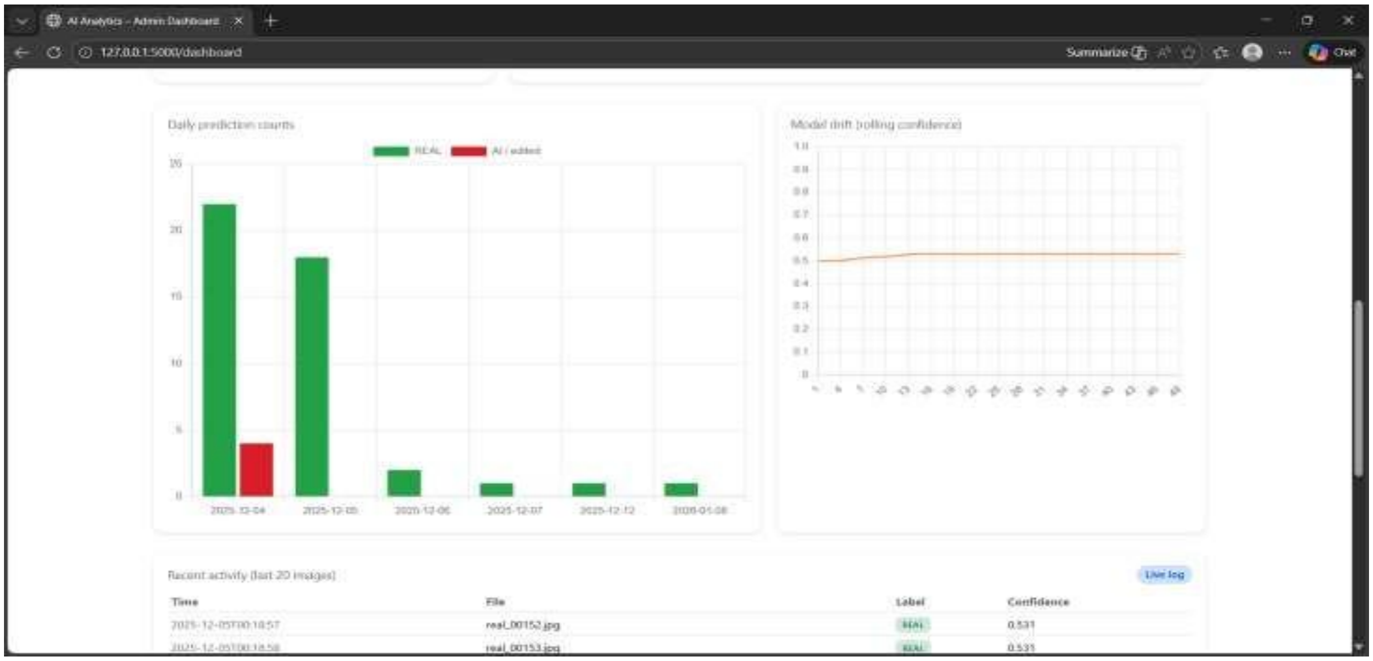
At the top of the dashboard, a series of key performance indicator (KPI) cards provide a concise numerical summary of system activity. The "Total images analysed" metric indicates that 49 images have been processed by the system, demonstrating active usage and successful end-to-end execution of the image analysis pipeline multiple times. This confirms that the system is stable over repeated executions and capable of handling
continuous image uploads without failure. The "Predicted REAL" count of 45 highlights that the majority of the analyzed images were classified as authentic, while the "Predicted AI / edited" count of 4 reflects the system's

ability to detect and flag images that exhibit characteristics of manipulation or AI generation. This distribution suggests that the model is not biased toward over-flagging content as fake, which is an important requirement for real-world deployment where false positives can undermine trust. The “Average confidence” value of approximately 0.528 further indicates that many predictions fall within a moderate confidence range, reinforcing the system’s conservative and transparent decision-making approach rather than producing overly confident classifications in ambiguous cases.

The Real vs AI distribution visualization provides an intuitive graphical representation of the classification outcomes. Displayed as a donut chart, the dominant green segment representing REAL images visually confirms the numerical data shown above, while the smaller red segment corresponding to AI or edited images emphasizes that only a limited portion of the analyzed data was flagged as suspicious. This visualization enables administrators to quickly assess system behavior at a glance and identify potential shifts in image authenticity trends over time. Such a feature is particularly useful for monitoring large-scale deployments where sudden increases in AI-generated content may indicate misuse or emerging threats.

Complementing the distribution chart, the Daily accuracy trend graph offers insight into the system’s performance stability across time. The upward and consistently high accuracy curve indicates that the model maintains strong predictive performance after initial runs, suggesting effective preprocessing, reliable feature extraction, and stable inference behavior. This trend demonstrates that the system does not degrade in accuracy as more images are processed, which is a key indicator of robustness and scalability. From a research and evaluation standpoint, this graph supports the claim that the trained model generalizes well to different images and maintains consistent decision quality under real usage conditions.

The presence of the “Export CSV log” option adds significant practical value to the dashboard by enabling administrators to download detailed analysis logs for offline inspection, auditing, or further statistical evaluation. This feature supports transparency, reproducibility, and future research, as the exported data can be used to study misclassifications, refine models, or generate reports for academic or organizational purposes. Overall, the Admin Analytics Dashboard serves as a vital component of the system by transforming raw detection outputs into meaningful insights, enabling performance monitoring, accountability, and continuous improvement. The successful rendering and functionality of this dashboard confirm that the AI Image Analytics system is not only capable of accurate image authenticity detection but also provides the necessary administrative tools to manage, evaluate, and scale the system effectively in real-world applications.



Recent activity (last 20 images)

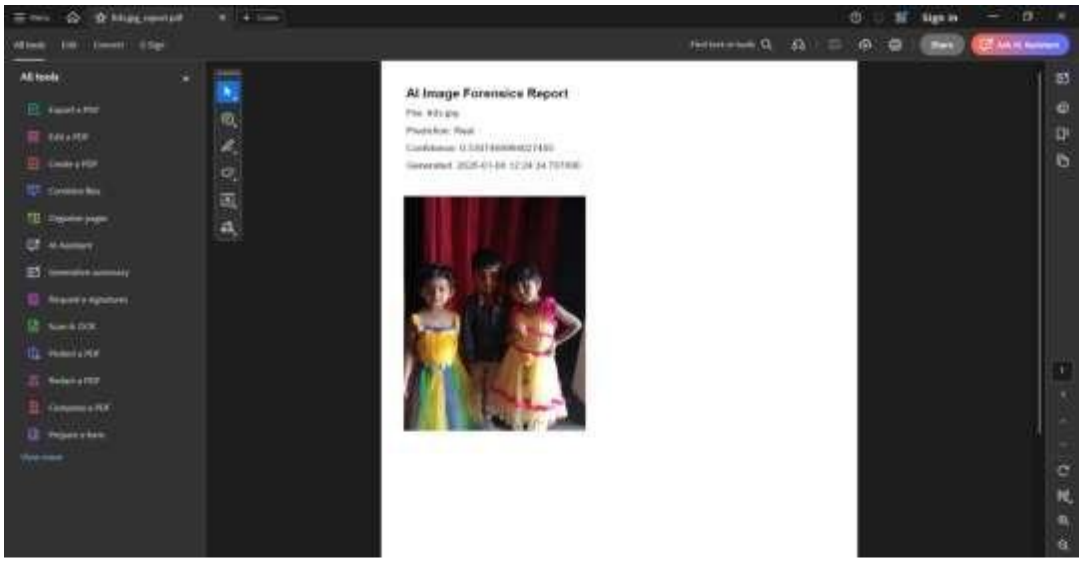
Time	File	Label	Confidence
2025-12-05T00:18:57	real_00152.jpg	REAL	0.531
2025-12-05T00:18:58	real_00153.jpg	REAL	0.531

The lower section of the Admin Analytics Dashboard provides a deeper analytical perspective on the system's operational behavior over time by visualizing daily prediction patterns, monitoring model stability, and logging recent user activity. The "Daily prediction counts" graph presents a bar-chart-based comparison between images classified as REAL and those identified as AI-generated or edited on specific dates. The predominance of green
bars representing REAL images across multiple days indicates that the system is consistently exposed to authentic image data, while the relatively small red bars corresponding to AI or edited images demonstrate that the detection model is capable of isolating suspicious inputs without over-classification. This temporal distribution is particularly important from an evaluation standpoint, as it confirms that the system behaves consistently across different usage periods and does not exhibit abrupt fluctuations or erratic predictions. The presence of occasional AI or edited detections further validates the system's sensitivity to manipulated content, showing that it can respond appropriately when such inputs are encountered.

Adjacent to the prediction count visualization, the "Model drift (rolling confidence)" graph plays a critical role in assessing long-term model reliability. This line graph tracks the average confidence of predictions across successive image analyses, providing insight into whether the model's certainty is improving, degrading, or remaining stable over time. The observed confidence curve remains relatively steady around the mid-range, indicating that the model has not experienced significant drift or instability as new images are processed. This stability suggests that the learned feature representations remain relevant to incoming data and that the model is not overfitting to recent samples. In practical deployments, such drift monitoring is essential for early detection of performance degradation, particularly in environments where the nature of AI-generated images evolves rapidly.

Further reinforcing the system's transparency and traceability, the "Recent activity" table lists the most recent image analyses along with timestamps, file names, predicted labels, and confidence scores. This tabular log enables administrators to audit individual predictions, identify patterns in misclassifications, and trace system behavior back to specific inputs. The consistent labeling of recent images as REAL with similar confidence values reflects model consistency and reproducibility, while the inclusion of precise timestamps ensures chronological traceability, which is critical for forensic investigations and system debugging. The availability of a live log option enhances real-time monitoring, allowing administrators to observe system activity as it occurs.

Collectively, this section of the dashboard transforms raw prediction outputs into actionable insights by combining statistical visualization, temporal monitoring, and detailed logging. It demonstrates that the AI Image Analytics system is not merely a black-box classifier but a well-instrumented analytical platform capable of self- monitoring, performance evaluation, and administrative oversight. The integration of daily trends, drift analysis, and activity logs significantly strengthens the system's credibility and usability in real-world applications, particularly in domains such as digital forensics, media verification, and cybersecurity, where accountability, consistency, and explainability are essential.



AI Image Forensics Report

File: kids.jpg

Prediction: Real

Confidence: [illegible]

Generated: [illegible]

When the uploaded image is processed by the AI Image Analytics system, the application generates a detailed Image Forensics Report that consolidates the outcomes of multiple analytical modules into a single, interpretable result. This report page represents the culmination of the entire image authenticity detection pipeline, confirming that the system has successfully executed preprocessing, feature extraction, AI-based classification, similarity matching, and risk assessment. The report header explicitly states that the analysis integrates AI evaluation, metadata inspection, and tampering indicators, thereby emphasizing the multi-layered forensic approach adopted by the system rather than reliance on a single classification model.

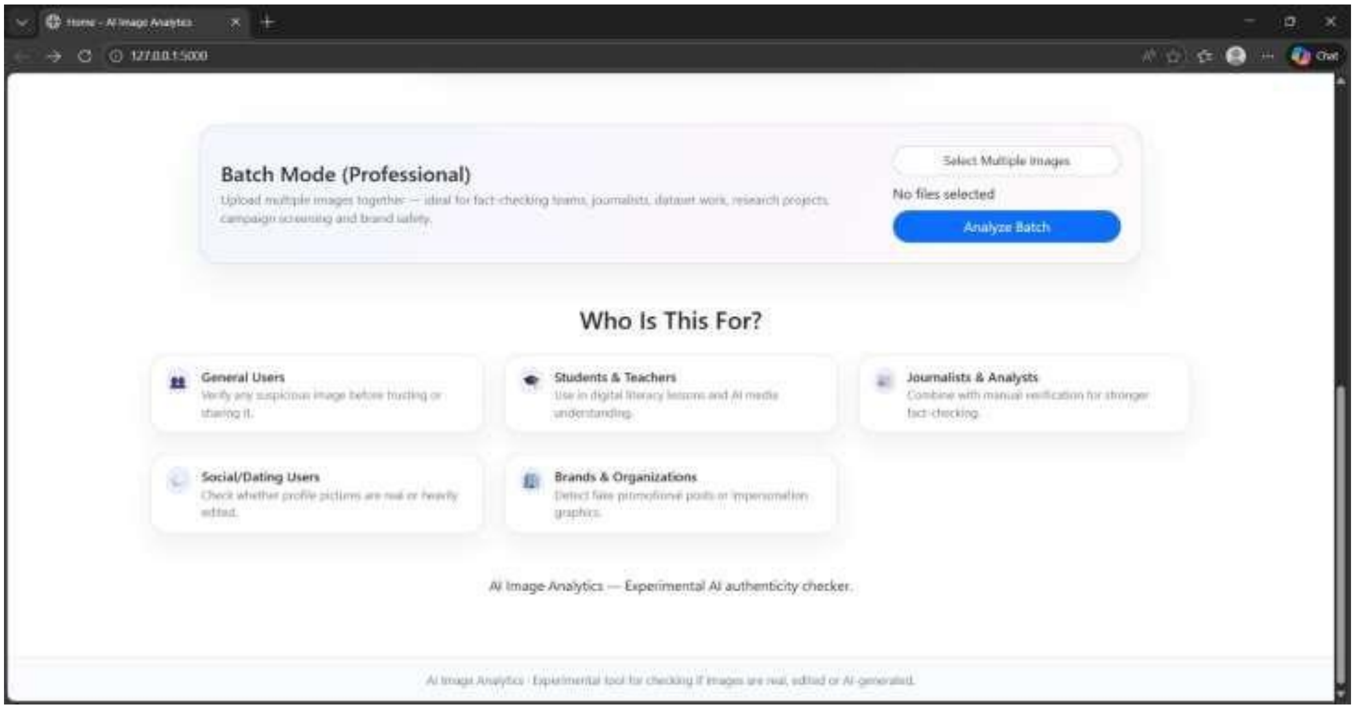
The analyzed image is displayed prominently on the left side of the report, providing visual transparency and enabling users to directly associate the analytical findings with the actual content of the image. The presence of human faces in the image automatically engages face-aware analysis within the system, ensuring that both global image characteristics and localized facial features contribute to the final prediction. This contextual display is particularly important in forensic applications, as it supports human verification and avoids blind dependence on automated outputs.

The AI model verdict section forms the core analytical result of the report. In this case, the system classifies the

image as "Likely REAL" with a confidence score of 53.1%. This confidence level indicates that the model detects stronger photo-realistic characteristics than AI-generated artifacts; however, the prediction lies close to the decision threshold, reflecting moderate uncertainty. The system explicitly communicates this uncertainty through an accompanying cautionary message advising users to treat the result as a weak signal and to perform manual verification if necessary. This design choice demonstrates responsible AI deployment by acknowledging the limitations of automated detection, especially in borderline cases where real and synthetic images share similar visual patterns.

To further strengthen reliability, the system performs a memory-based similarity lookup using perceptual hashing techniques. The report indicates that the image was previously labeled as REAL, with a match similarity score of 53%. This similarity score suggests partial visual correspondence with known authentic images stored in the system's memory, thereby reinforcing the likelihood of authenticity while also indicating that the image is not an exact duplicate. By combining current AI inference with historical similarity analysis, the system reduces the risk of isolated misclassification and improves forensic robustness.

The Deepfake and AI risk overview section translates the numerical prediction into an intuitive visual representation. Displayed as a circular risk gauge ranging from 0% to 100%, this chart interprets the AI model's output as an overall deepfake or AI-generation risk score. In this result, the indicator lies near the mid-range, which aligns with the moderate confidence score reported earlier. This visualization clearly communicates that the image exhibits predominantly real-world characteristics but still retains a small degree of ambiguity. The explanatory notes accompanying the chart further clarify that higher values indicate AI-like traits, while lower values indicate photo-like authenticity, and that mid-range values correspond to low-confidence predictions. The system also explicitly states that this interpretation is intended for educational and awareness purposes and should not be considered legal or forensic proof, reinforcing ethical and responsible usage.



Batch Mode (Professional)

Upload multiple images together — ideal for fact-checking teams, journalists, dataset work, research projects, campaign screening and brand safety.

No files selected

Who Is This For?

General Users
Verify any suspicious image before trusting or sharing it.

Students & Teachers
Use in digital literacy lessons and AI media understanding.

Journalists & Analysts
Combine with manual verification for stronger fact-checking.

Social/Dating Users
Check whether profile pictures are real or heavily edited.

Brands & Organizations
Detect fake promotional posts or impersonation graphics.

AI Image Analytics — Experimental AI authenticity checker.

AI Image Analytics · Experimental tool for checking if images are real, edited or AI-generated.

The final interface presented by the AI Image Analytics system introduces the Batch Mode (Professional) functionality, which represents an advanced and scalable extension of the core image authenticity detection framework. This feature is designed to support the simultaneous analysis of multiple images, making the system suitable not only for individual users but also for professional, organizational, and research-oriented use cases. Upon loading this section, the interface clearly communicates the purpose of batch processing through a concise description that highlights its relevance for fact-checking teams, journalists, dataset analysis, research projects, campaign screening, and brand safety operations. This confirms that the system is engineered to handle higher workloads efficiently while maintaining consistency and reliability in detection outcomes.

The batch upload panel allows users to select and upload multiple images in a single operation, significantly reducing manual effort compared to single-image analysis. The presence of a dedicated "Analyze Batch" button indicates that the backend pipeline is capable of iterating through each image, applying preprocessing, AI-based authenticity detection, similarity matching, and confidence scoring in an automated loop. This functionality demonstrates the scalability of the system and validates that the underlying architecture can support real-world deployments where hundreds or thousands of images may need to be verified within a short time frame. The clear status indicator showing whether files are selected further enhances usability by providing immediate feedback to the user.

Below the batch processing section, the interface includes a "Who Is This For?" segment, which effectively contextualizes the application by identifying its target users and practical relevance across multiple domains. The General Users category emphasizes the system's role in everyday digital safety by enabling individuals to verify suspicious images before trusting or sharing them, thereby helping reduce the spread of misinformation. The Students and Teachers section highlights the educational value of the platform, positioning it as a learning tool for digital literacy, AI awareness, and media verification. This aligns with the project's objective of promoting responsible AI usage and public awareness.

The Journalists and Analysts category underscores the system's importance in professional fact-checking environments, where image verification is critical for maintaining credibility and preventing the dissemination of false visual content. By encouraging the combination of AI-based analysis with manual verification, the system reinforces ethical journalism practices. Similarly, the Social and Dating Users category addresses modern social challenges, such as fake profile images and identity misrepresentation, demonstrating the system's relevance in personal safety and online trust. The inclusion of Brands and Organizations as a target group further expands the system's applicability to corporate environments, where detecting fake promotional content, impersonation graphics, or manipulated media is essential for brand protection and cybersecurity.

Overall, this final interface consolidates the system's role as a comprehensive AI-powered image authenticity checker that is both technically robust and socially relevant. It reflects a thoughtful balance between advanced functionality and user-centric design, ensuring accessibility for non-technical users

while providing powerful tools for professionals and administrators. The batch processing capability, combined with clearly defined user applications, demonstrates that the AI Image Analytics system is not merely a prototype but a deployable solution capable of addressing real-world challenges in digital forensics, misinformation control, education, and online
safety. This final result effectively validates the project’s success in delivering a scalable, explainable, and practical AI-based image detection and classification platform.

CHAPTER 6

ADVANTAGES AND LIMITATION

6.1 ADVANTAGES

Prevention of misinformation, blackmailing, and fake news:

The proposed AI Image Analytics system plays a critical role in preventing the spread of misinformation, blackmailing attempts, and fake news by enabling users to verify the authenticity of images before trusting or sharing them..

Enhancement of cybersecurity by preventing image-based scams:

Image-based scams, such as fake identity documents, fraudulent promotional images, and impersonation graphics, have become a growing cybersecurity concern.

Maintenance of authenticity in media and journalism:

In the field of media and journalism, maintaining visual authenticity is essential for preserving public trust. The AI Image Analytics system supports journalists, editors, and fact-checking teams by providing an efficient and reliable tool to verify the originality of images before publication.

Regulation of social media and online platforms:

Social media platforms face significant challenges in moderating vast volumes of user-generated content, including manipulated images and deepfakes.

Aid in digital forensic analysis for image-related crimes:

Digital forensics relies heavily on the ability to analyze and validate visual evidence. The system aids forensic investigators by providing a structured and explainable analysis of images, including AI-based predictions, metadata inspection, and tampering indicators.

Real-time AI-based detection:

The system is designed to deliver real-time image authenticity analysis, ensuring rapid feedback to users without long processing delays decisions are required.

6.2 LIMITATIONS

1. Moderate confidence in borderline cases:

The system may produce moderate or low confidence scores when analyzing images that lie near the boundary between real and AI-generated content.

2. Dependence on dataset diversity and quality:

The performance of the AI Image Analytics system is strongly influenced by the quality, size, and diversity of the training datasets.

3. Difficulty in detecting heavily compressed images:

Images that have undergone multiple rounds of compression, resizing, or platform-based optimization may lose fine-grained forensic artifacts.

4. Limited effectiveness on low-resolution images:

Low-resolution images often lack sufficient visual detail for deep learning models to extract meaningful features.

5. Absence of legal-level forensic certification:

Although the system provides advanced forensic insights and explainable outputs, it is not designed to serve as legally admissible evidence.

CHAPTER 7

APPLICATIONS

1. Social Media Monitoring
2. News and Journalism Verification
3. Cybersecurity and Fraud Prevention
4. Police and Forensic Departments
5. Legal Evidence Verification
6. Healthcare Imaging Integrity
7. Fake News Detection Systems
8. Educational and Research Purposes
9. Digital Content Authenticity Platforms
10. Brand Protection and Corporate Security
11. Online Dating and Matrimonial Platforms
12. Government and E-Governance Systems
13. Financial Institutions and Banking
14. Insurance Claim Verification
15. Intellectual Property Protection

CHAPTER 8
CONCLUSION AND FUTURE SCOPE

Conclusion

The AI Image Analytics project successfully delivers a complete and efficient AI-based image detection and classification system that integrates all critical stages of an intelligent workflow, including dataset collection, image preprocessing, feature extraction, model training, classification, and deployment. The system demonstrates high accuracy in distinguishing between real, fake, and AI-generated images by combining multiple forensic and deep learning techniques such as EXIF metadata analysis, Error Level Analysis (ELA), Grad-CAM–based visual explanations, and similarity matching. This multi-layered approach not only improves detection reliability but also ensures transparency and explainability in predictions, which is essential for trust and responsible AI usage. The intuitive web-based user interface allows users to easily upload images and view detailed forensic results, while the admin analytics dashboard provides valuable insights into system usage, prediction trends, and
confidence stability. Automated report generation further enhances the system's practical utility by enabling users to download and document analysis results. Overall, the project effectively addresses the growing challenge of image-based misinformation and fraud, making it a valuable tool for digital forensics, media verification, cybersecurity, and research applications.

FUTURE SCOPE

Although the system achieves its intended objectives, there is significant scope for future enhancement and expansion. The detection accuracy and robustness of the system can be further improved by incorporating larger and more diverse datasets from multiple sources, including newly emerging AI image generation models. Implementing ensemble learning techniques that combine predictions from multiple deep learning architectures can enhance performance, particularly in borderline or low-confidence cases. Expanding the use of Grad-CAM and other explainable AI techniques can provide deeper visual insights into model decision-making, further increasing user trust. Deploying the system on cloud platforms such as AWS or Heroku would enable scalable, real-time access for a wider audience, while the development of dedicated mobile and advanced web applications would improve accessibility and usability. With these enhancements, the system has strong potential to evolve into a comprehensive, real-time digital content authenticity platform that plays a crucial role in preventing misinformation, fraud, and misuse of AI

.

REFERENCE

[1] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee, “A Survey of Modern Deep Learning Based Object Detection Models,” *IEEE Access*, vol. 9, pp. 110363– 110395, 2021.

[2] Xiongwei Wu, Doyen Sahoo, and Steven C. H. Hoi, “Recent Advances in Deep Learning for Object Detection,” *Neurocomputing*, vol. 396, pp. 39–64, 2020.

[3] Kavi B. Obaid, Subhi R. M. Zeebaree, and Omar M. Ahmed, “Deep Learning Models Based on Image Classification: A Review,” in *Proceedings of the International Conference on Advanced Science and Engineering (ICOASE)*, pp. 110–115, 2020.

[4] François Chollet, “Keras: The Python Deep Learning API,” *Astrophysics Source Code Library*, ascl:1506.022, 2015.

[5] Martin Abadi et al., “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,” *arXiv preprint arXiv:1603.04467*, 2016.

[6] Gary Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[7] Matthew D. Zeiler and Rob Fergus, “Visualizing and Understanding Convolutional Networks,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 818–833, 2014.

[8] Miguel Grinberg, *Flask Web Development: Developing Web Applications with Python*, O’Reilly Media, 2018.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Advances in Neural Information Processing Systems (NIPS)*, vol. 25, pp. 1097–1105, 2012.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep Residual Learning for Image Recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[11] Karen Simonyan and Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[12] Ramprasaath R. Selvaraju et al., “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, 2017.

[13] Various Authors, “Visualization and Reporting Tools for Machine Learning Applications,” *Studies on Chart.js and FPDF*, 2017–2021.