

## 1. Introduction

Image forgery detection is an essential task that aims to detect whether an image is manipulated or not. There are various techniques that have been proposed to detect image forgery, such as passive techniques, active techniques, and hybrid techniques. Passive techniques rely on the analysis of the image's content, while active techniques involve the use of watermarking, digital signatures, or other data embedding techniques. Hybrid techniques combine both passive and active techniques to improve the accuracy of detection. In recent years, deep learning-based approaches have shown great potential in detecting image forgery. Convolutional neural networks (CNNs) and generative adversarial networks (GANs) are two popular deep learning models that have been used in image forgery detection. In this project, we propose to develop an image forgery detection system using Python and Django web framework. The system will be designed to detect whether an image is forged or not by analyzing its content. We will use deep learning models such as CNNs and GANs to train the system to identify manipulated images. The project will have the following main objectives: Develop a web-based user interface for the image forgery detection system using Django web framework. Implement various image forgery detection techniques, including passive, active, and hybrid techniques. Explore the use of deep learning models such as CNNs and GANs in detecting image forgery. Evaluate the performance of the image forgery detection system using various metrics such as precision, recall, and F1 score. Develop a database for storing the detected forged images and the relevant metadata. Implement data visualization tools to analyze the performance of the system. The proposed system will be useful in various applications such as journalism, social media, and legal evidence, where the authenticity of images is crucial. The system will also be helpful in forensic investigations, where the detection of manipulated images can aid in solving criminal cases.

## **2. Literature Review**

Image forgery detection is an important area of research in the field of image processing and computer vision. With the increasing use of digital images, it has become easier to create and manipulate images, leading to an increase in the number of image forgeries. Image forgery detection is the process of identifying if an image has been tampered with or manipulated in any way. In this literature survey, we review some of the existing techniques for image forgery detection in Python Django.

### **1. DWT and SVD based technique:**

In image processing, the DWT (Discrete Wavelet Transform) and SVD (Singular Value Decomposition) techniques are commonly used for detection and analysis. These techniques can be used for a wide range of applications, including edge detection, texture analysis, and object recognition.

DWT is a mathematical transform that decomposes a signal or image into different frequency bands using wavelets. This technique can be used for image compression, denoising, and feature extraction. In DWT, the image is divided into different scales, each containing different frequency components. The image can be reconstructed by combining the different scales with different weights.

SVD, on the other hand, is a factorization technique that decomposes a matrix into its constituent parts, including singular values, left singular vectors, and right singular vectors. This technique can be used for compression, noise reduction, and feature extraction. In SVD, the image is represented as a product of three matrices, each containing the singular values and vectors.

Both DWT and SVD can be used for image processing detection. In DWT, the detail coefficients can be used to detect edges and textures in the image. The magnitude of the coefficients can be used as a measure of the intensity of the features. In SVD, the singular values and vectors can be used to detect features in the image. The singular values can be used as a measure of the intensity of the features, while the singular vectors can be used to represent the orientation of the features.

To implement these techniques in Python, we can use libraries such as NumPy, OpenCV, and Py Wavelets. NumPy is a Python library for scientific computing, which provides support for array operations and linear algebra.

In this project, we will implement an image processing detection system using DWT and SVD techniques in Python. The system will be implemented using the Django web framework, which will provide a web-based interface for users to upload images and perform detection.

### 3. Analysis of Problem

With the advent of social networking services such as Facebook and Instagram, there has been a huge increase in the volume of image data generated in the last decade. Use of image processing software like [GNU Gimp](#), [Adobe Photoshop](#) to create doctored images and videos is a major concern for internet companies like Facebook. These images are prime sources of fake news and are often used in malevolent ways such as for mob incitement. Before action can be taken on basis of a questionable image, we must verify its authenticity. Following the explosion of social networking services, there has been a monumental increase in the volume of image data. Moreover, the development in image processing software such as Adobe Photoshop has given a rise to doctored images. Such doctored images can be used for malicious purposes such as spreading false information and inciting violence. This image forgery detection project allows users to detect even the slightest signs of forgery in an image. This project is developed using the Django framework with Python as programming language.

## **4. System Requirement**

### **4.1. Software Requirement**

- Windows 10 or 11 or Linux or Unix
- 64/32-bit Operating System
- Python 3 or above,
- JS (ECMAScript 2017 or above)
- PhP (6 or above)
- Bootstrap v4.5 or above
- MySQL 7 and above
- OpenCV v3.4 or above
- NumPy v1.23.3 or above
- Xampp Server v6.0.29 or above

### **4.2 Hardware Requirement**

- Processor -Core i3 or Higher
- Minimum 8 GB RAM or Higher
- Free Space required Minimum 3 GB of Hard Disk or higher
- Keyboard and Mouse

## 5. System Design

### UML Modeling

UML, as the name shows, is a modeling language. It is used to specify, draw, visualize and document the parts of the software. It provides a set of notations (such as rectangle, ellipses, lines, etc.) to create the visual model of a system. This phase is used to design different UML, diagrams corresponding to the application development.

#### A. Class Diagram

Class diagram is a type of static structure diagram which describes the structure of a system by representing the classes of the system, their attributes, operations, and the relationship among these classes. The figure 5.1 represent the class diagram for Forgery Detection System.

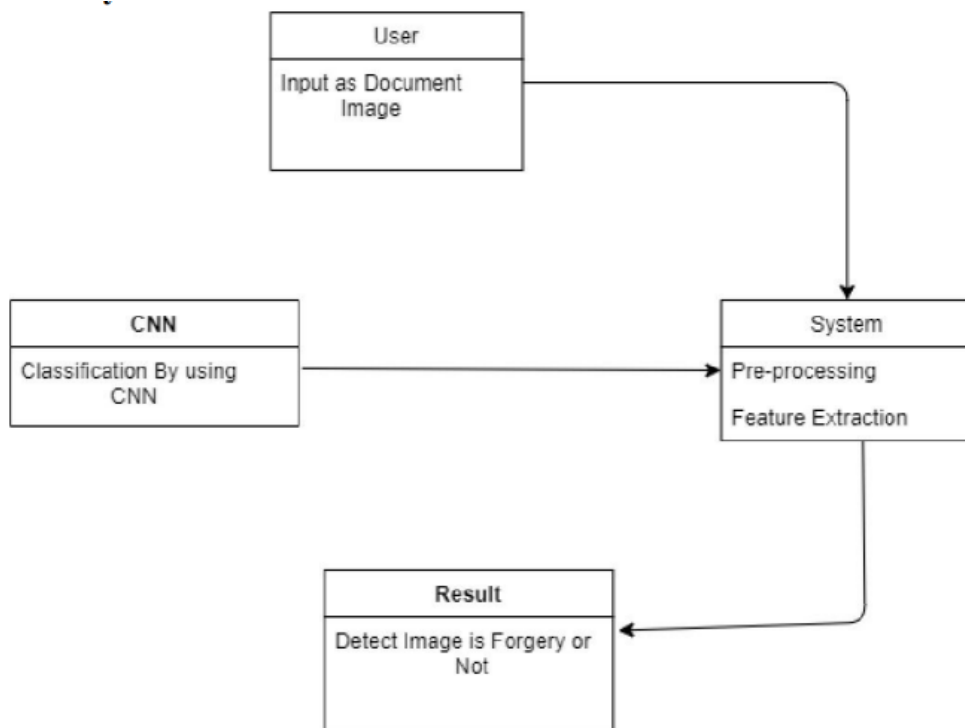


Figure: 5.1 Class Diagram

## B. Use Case Diagram

Use case diagrams are the diagrammatic representation depicting user`s interactions with the system. The diagram shows different types of user and various ways in which these users interact with the system. Figure 5.2 shows the use case diagram for System and User.

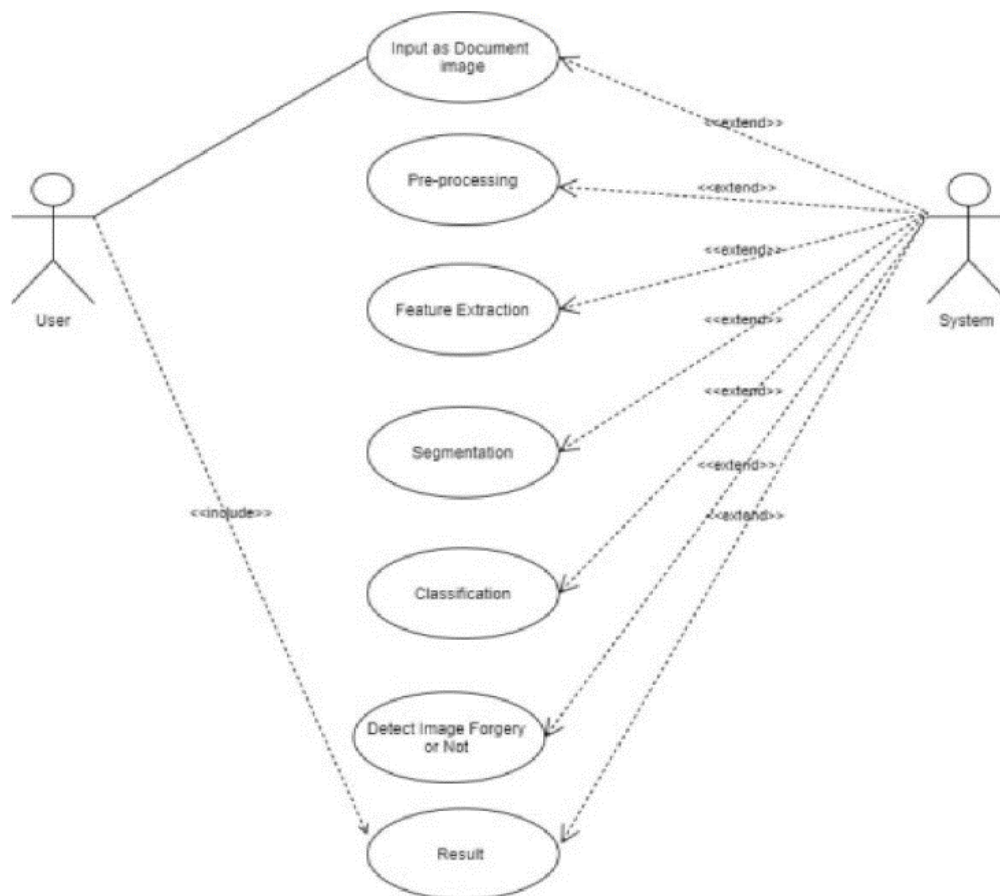


Figure: 5.2 Use Case Diagram

### C. Sequence Diagram

In sequence diagram step by step sequence of steps is shown. In above diagram first preprocess all train data and test data. Then by applying the train data train the machine and build the module and at the last apply machine learning algorithm on it. For testing purpose apply the test data on module and see the classification either fake or real

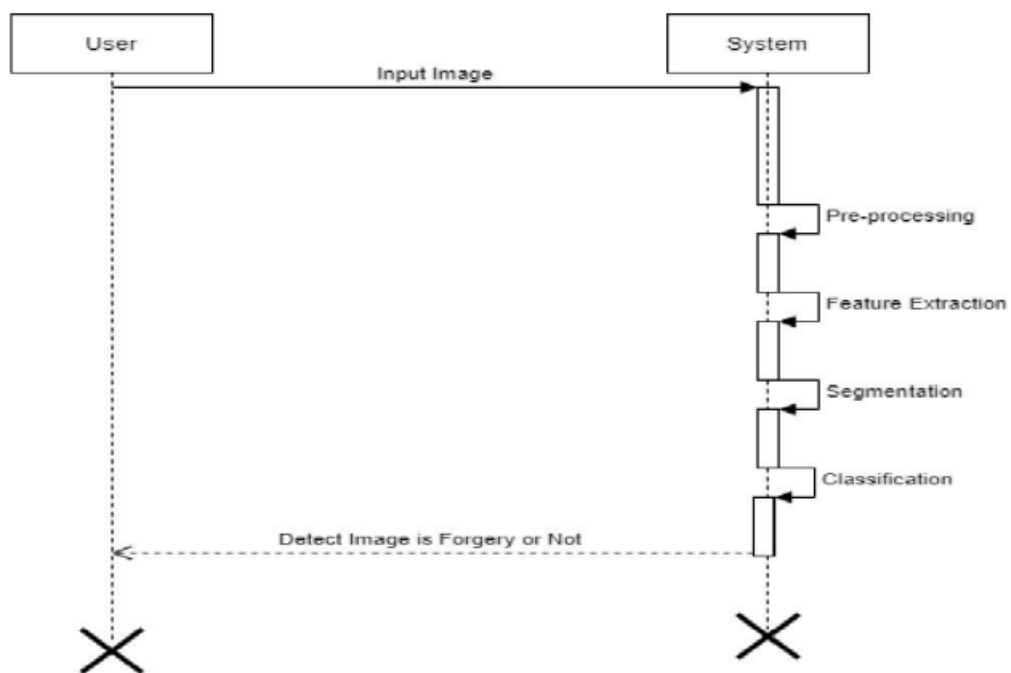
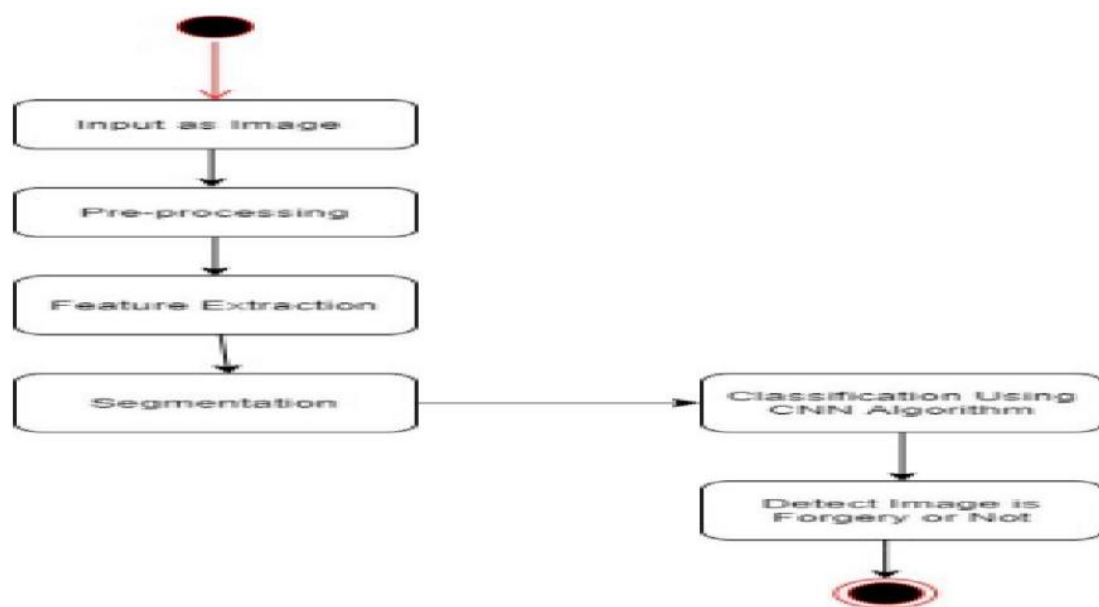


Figure: 5.3 Sequence Diagram

### D. Activity Diagram with Necessary Information

Activity Diagram shows the active flow of the system. In above diagram the flow of our project is shown actually how the data flow.



**Figure: 5.4 Activity Diagram**



## 6. Working of the System

To implement the system, we will follow these steps:

Install the necessary libraries and tools.

Implement the DWT and SVD techniques.

Create a Django project.

Implement the web-based interface.

Test the system.

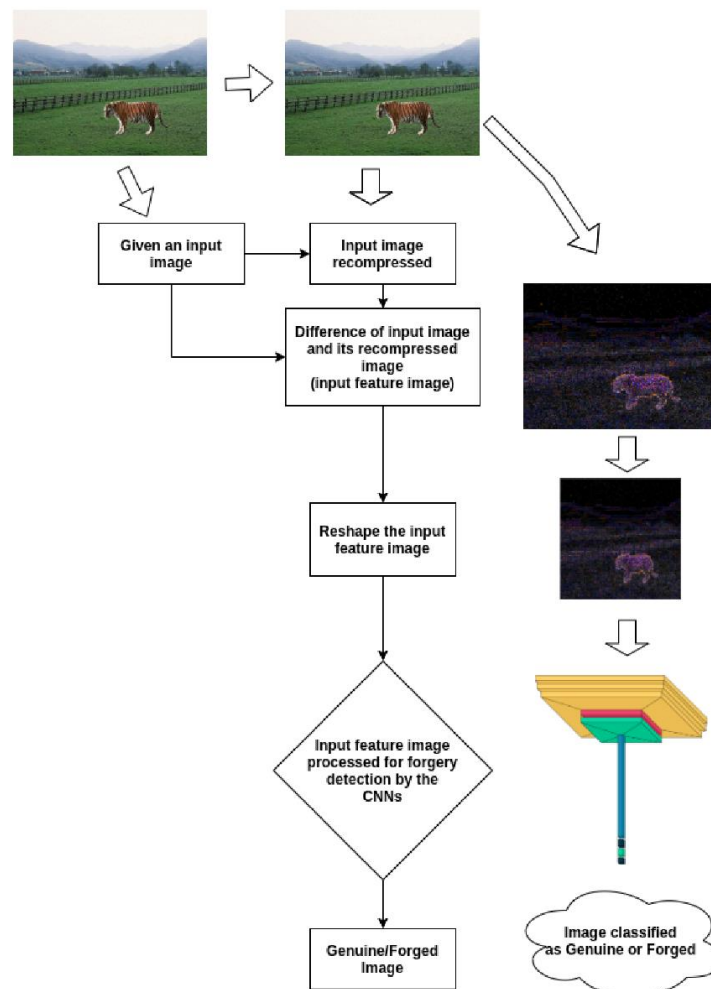
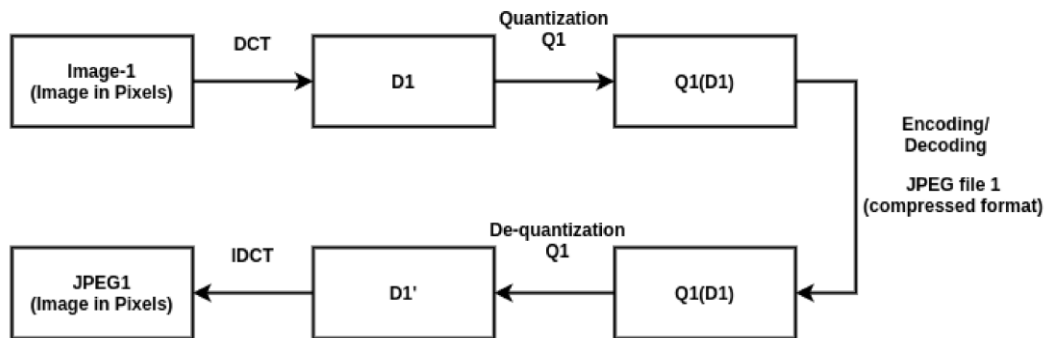


Figure: 6.1 Flowchart of the proposed work.



**Figure: 6.2 JPEG compression on image pixels, first DCT is applied on the image pixel blocks followed by the quantization. Then decompression of the compressed image is done with through de-quantization followed by IDCT, to obtain the image in pixel format.**

### Step 1: Install the Necessary Libraries and Tools

Before starting the project, we need to install the necessary libraries and tools. We will be using the following libraries:

- NumPy: for mathematical operations
- OpenCV: for image processing
- PyWavelets: for DWT transform
- Django: for web framework

We can install these libraries using the following commands:

```
pip install numpy opencv-python pywt django
```

### Step 2: Implement the DWT and SVD Techniques

To implement the DWT and SVD techniques, we will create two functions that take an image as input and return the processed image with the detected features highlighted.

#### 1.DWT

To implement the DWT technique, we can use the PyWavelets library to perform the DWT transform on an image. The DWT can be computed using the `pywt.dwt2` function, which takes an image and a wavelet name as inputs. The output of the function is a tuple containing the approximation coefficients and the detail coefficients at each level of the decomposition.

## 2. Deep learning-based technique:

Luo et al. (2019) proposed a deep learning-based technique for image forgery detection. The proposed technique uses a convolutional neural network (CNN) to classify the input image as either original or forged. The experimental results show that the proposed technique is effective in detecting various types of image forgeries.

Deep learning is a powerful tool for image processing detection, as it can learn features and patterns directly from the data without the need for manual feature extraction. In this approach, we use deep neural networks to classify and detect objects in images.

Deep learning-based techniques have shown remarkable performance in various applications of image processing, including object detection, segmentation, and recognition. In particular, deep convolutional neural networks (CNNs) have shown remarkable performance in object detection and recognition tasks.

In this project, we will implement a deep learning-based image processing detection system using Python and the TensorFlow library. TensorFlow is an open-source machine learning library developed by Google that provides support for building and training deep neural networks.

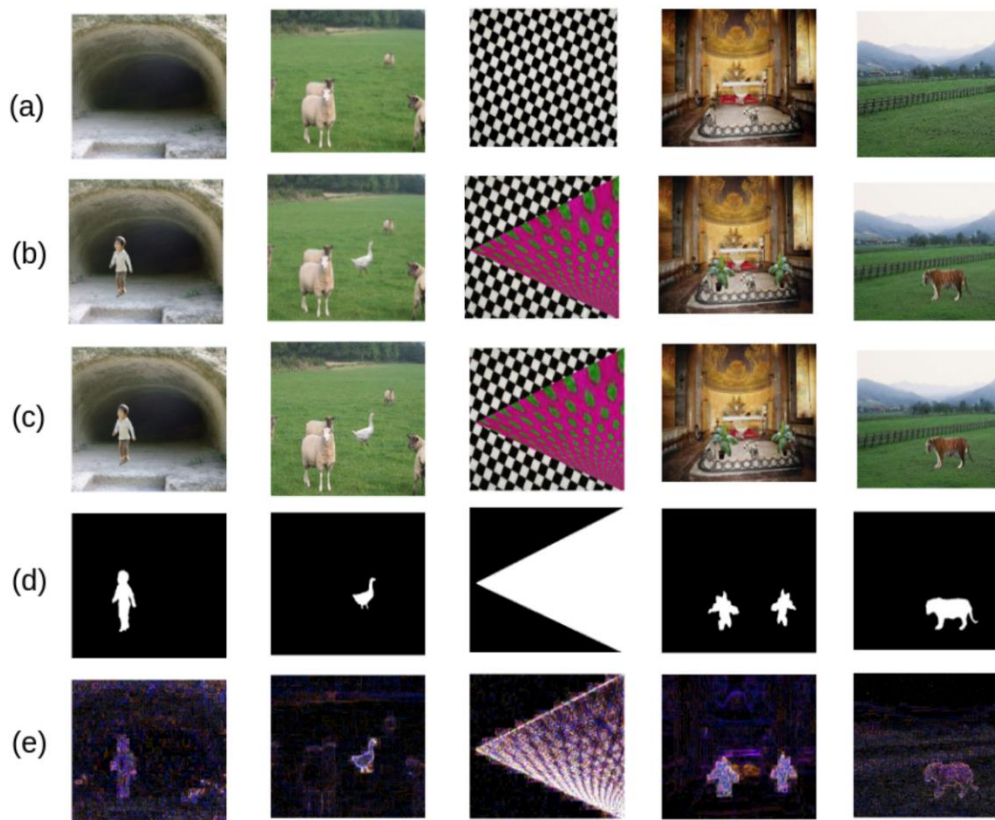
The interactions between the components of the image forgery detection system are shown in the following diagram:

Input Image -> Pre-processing Module -> Feature Extraction Module -> Classification Module -> Output Module

The input image is first passed through the pre-processing module, which performs various pre-processing operations on the input image. The pre-processed image is then passed through the feature extraction module, which extracts the features from the image. The extracted features are then passed through the classification module, which classifies the image as either original or forged. The output of the classification module is then passed through the output module, which displays the output of the system to the user.

The system modelling provides a high-level representation of the image forgery detection system. The actual implementation of the system may involve more detailed and complex components, but the system modelling provides a clear understanding of the system architecture and components. The system modelling helps to identify the interactions between the components of the system and facilitates the development of the image forgery detection system in Python Django.

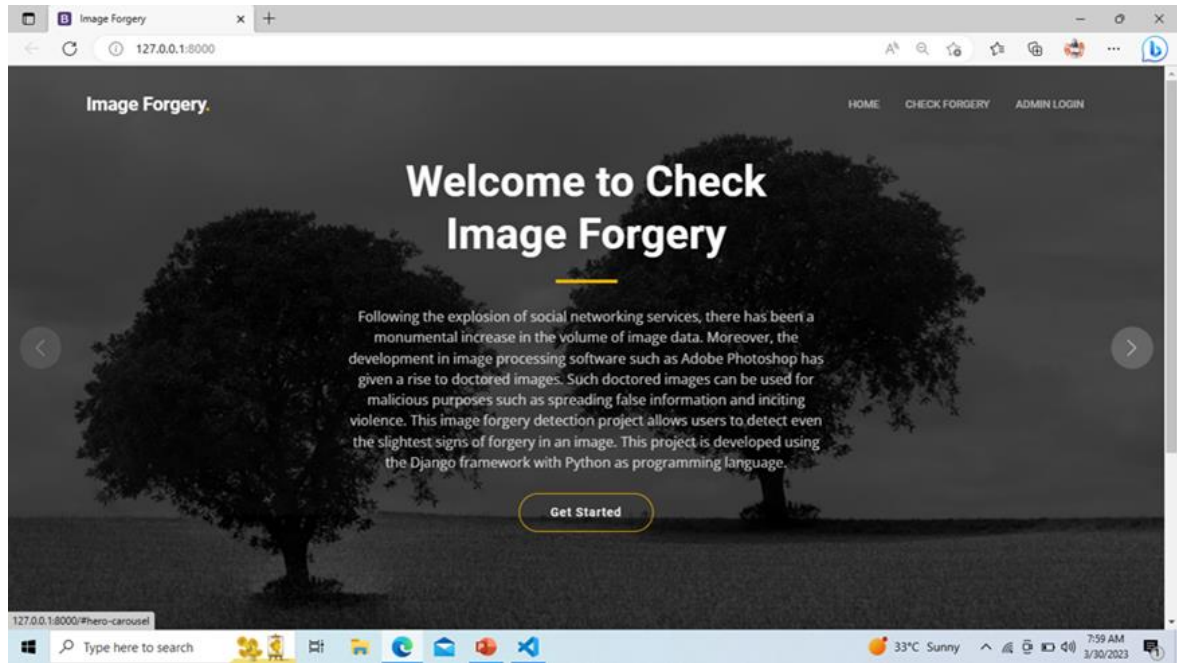
CNNs, which are inspired by the human visual system, are designed to be non-linear interconnected neurons. They have already demonstrated extraordinary potential in a variety of computer vision applications, including image segmentation and object detection. They may be beneficial for a variety of additional purposes, including image forensics. With the various tools available today, image forgery is fairly simple to do, and because it is extremely dangerous, detecting it is crucial. When a fragment of an image is moved from one to another, a variety of artifacts occur due to the images' disparate origins. While these artifacts may be undetectable to the naked eye, CNNs may detect their presence in faked images. Due to the fact that the source of the forged region and the background images are distinct, when we recompress such images, the forged is enhanced differently due to the compression difference. We use this concept in the proposed approach by training a CNN-based model to determine if an image is genuine or a fake. A region spliced onto another image will most likely have a statistically different distribution of DCT coefficients than the original region



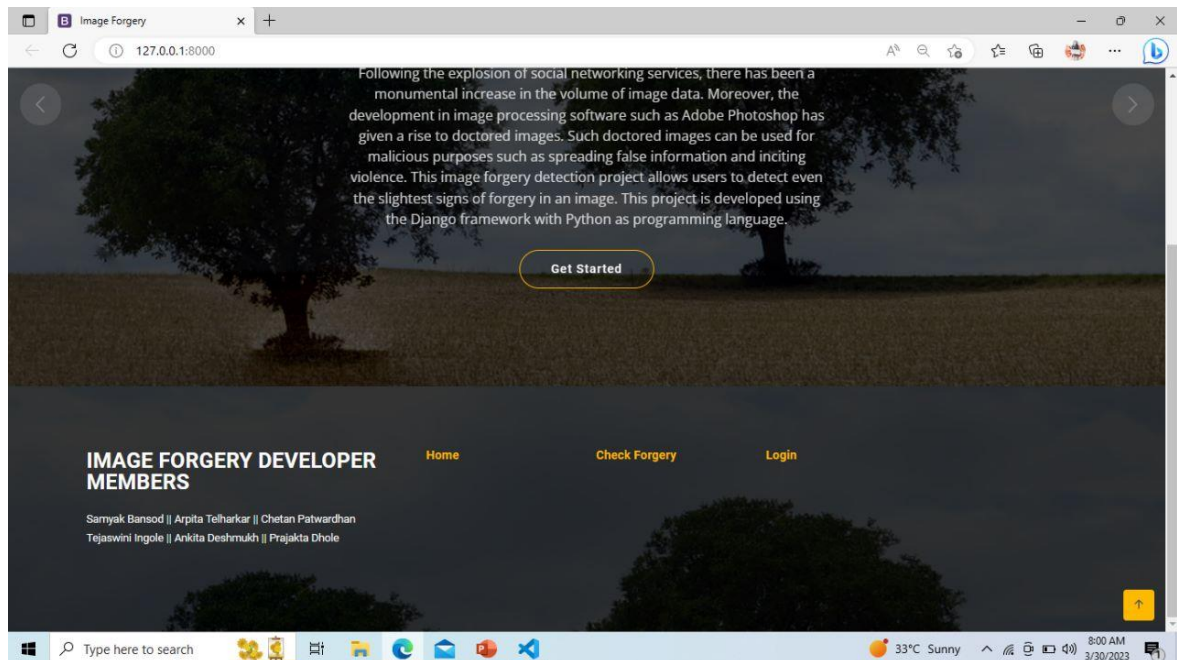
**Figure: 6.3 Various sample images and their processed forms: (a) the original images (RGB colour format); (b) the images with forgery (RGB colour format); (c) the recompressed forged images (RGB colour format); (d) ground truth of forgery (Binary format); (e) difference of the tampered image with its recompressed image (RGB colour format).**

## 7. Project layout

This is the welcome page for user to check Image Forgery or Not.

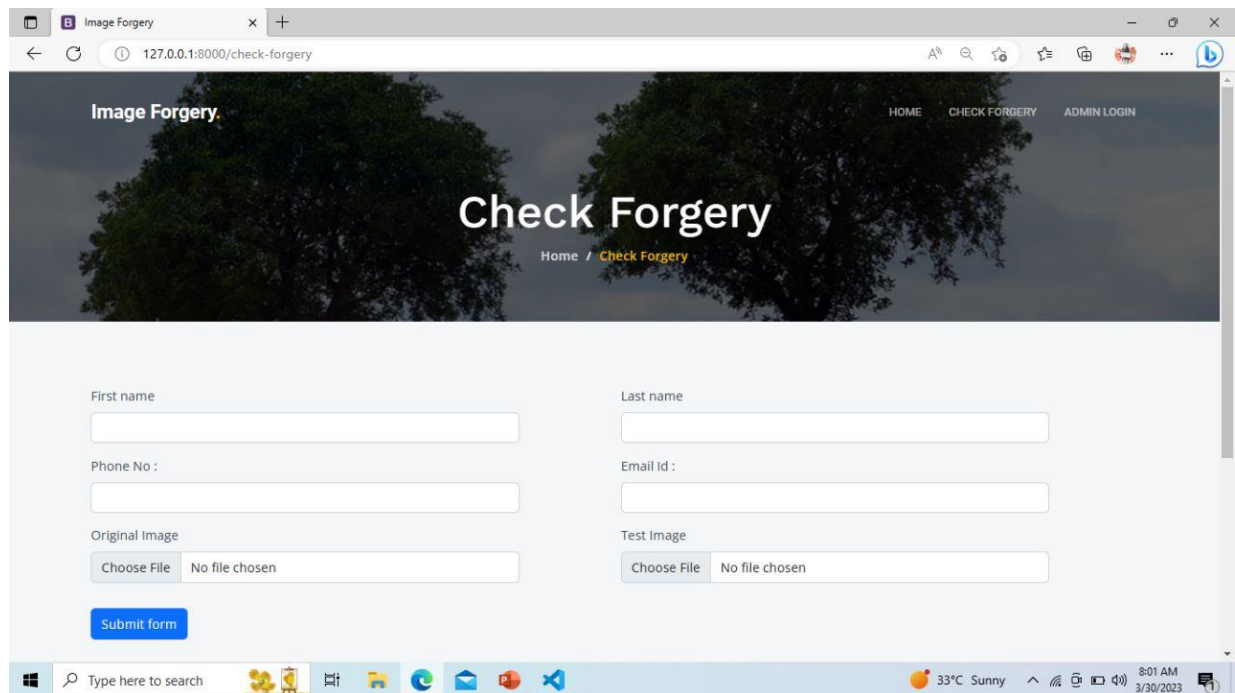


Screenshot 7.1: Get Started.



## 7.2 User Panel

This is login page to check the Forgery. Here user have to Enter their Personal information and upload images and submit the form to check the forgery.



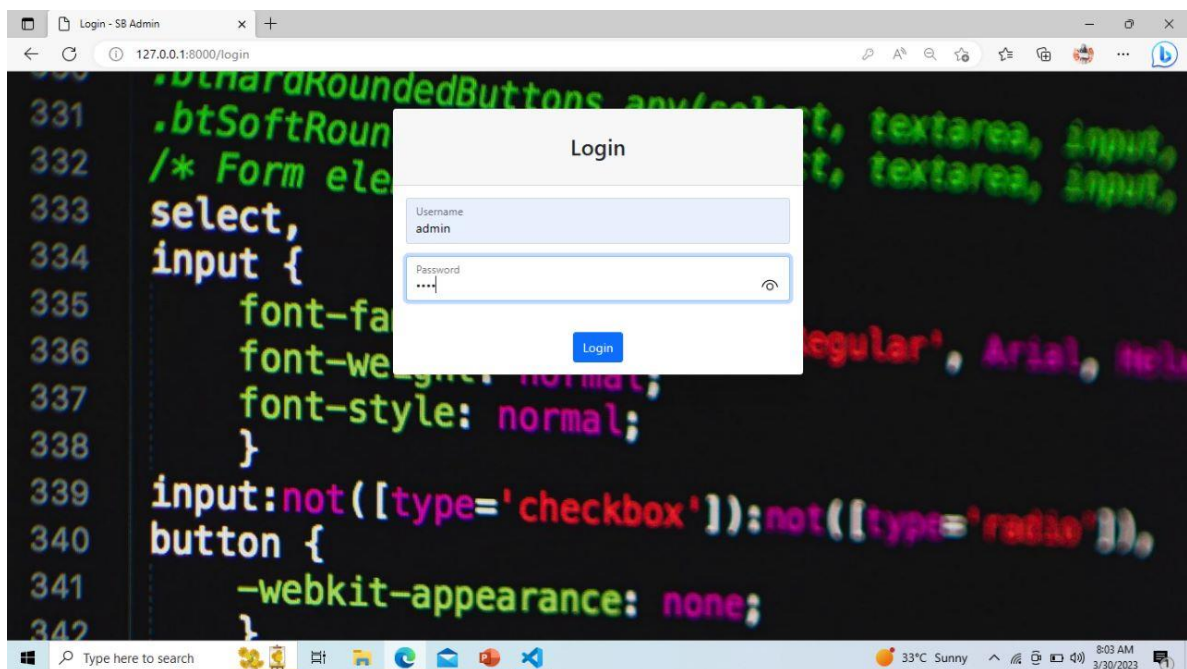
The screenshot shows a web browser window with the title 'Image Forgery'. The address bar displays '127.0.0.1:8000/check-forgery'. The page features a header with navigation links: 'HOME', 'CHECK FORGERY', and 'ADMIN LOGIN'. The main heading is 'Check Forgery', with a breadcrumb trail 'Home / Check Forgery'. Below the heading, there is a form with two columns. The left column contains fields for 'First name', 'Phone No :', and 'Original Image' (with a 'Choose File' button and 'No file chosen' text). The right column contains fields for 'Last name', 'Email Id :', and 'Test Image' (with a 'Choose File' button and 'No file chosen' text). A blue 'Submit form' button is located at the bottom left of the form area. The Windows taskbar is visible at the bottom, showing the search bar, taskbar icons, and system tray with a temperature of 33°C, 'Sunny' weather, and the date/time '8:01 AM 3/30/2023'.

**Screenshot 7.2: User login.**



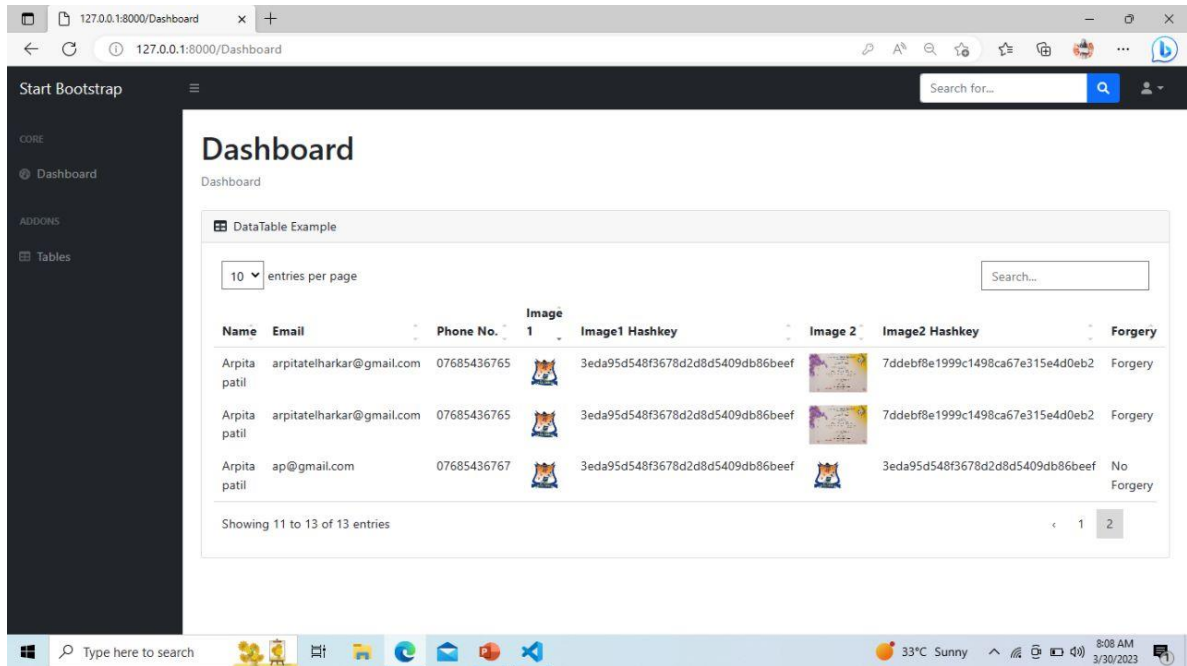
### 7.3 Admin Panel

After login on the Admin Page, we can see Admin panel, where we can see all the details such as how many Users are upload the image to check image is forgery or not.









Screenshot 7.3: Admin login.

After login on the Admin Page, we can see Admin panel, where we can see all the details such as how many Users are registered, Their names, Emails, phone number, images and their hash key. Result will be display to the admin, the image is forgery or not.



The screenshot displays a web application interface for an Admin Dashboard. The browser address bar shows the URL `127.0.0.1:8000/Dashboard`. The dashboard has a dark sidebar with navigation links: 'Start Bootstrap', 'CORE', 'Dashboard', and 'ADDONS' (containing 'Tables'). The main content area is titled 'Dashboard' and contains a 'DataTable Example'. The table has 8 columns: Name, Email, Phone No., Image, Image1 Hashkey, Image2, Image2 Hashkey, and Forgery. It displays 3 rows of data for a user named 'Arpita patil'. The first two rows are marked as 'Forgery', while the third row is marked as 'No Forgery'. The table includes a search bar and pagination controls at the bottom.

Name	Email	Phone No.	Image	Image1 Hashkey	Image2	Image2 Hashkey	Forgery
Arpita patil	arpitatelharkar@gmail.com	07685436765		3eda95d548f3678d2d8d5409db86beef		7ddebfb8e1999c1498ca67e315e4d0eb2	Forgery
Arpita patil	arpitatelharkar@gmail.com	07685436765		3eda95d548f3678d2d8d5409db86beef		7ddebfb8e1999c1498ca67e315e4d0eb2	Forgery
Arpita patil	ap@gmail.com	07685436767		3eda95d548f3678d2d8d5409db86beef		3eda95d548f3678d2d8d5409db86beef	No Forgery

**Screenshot 7.4: Admin Dashboard.**



## 8. Objective

- Develop a web-based user interface for the image forgery detection system using Django web framework.
- Implement various image forgery detection techniques, including passive, active, and hybrid techniques.
- Explore the use of deep learning models such as CNNs and GANs in detecting image forgery.
- Evaluate the performance of the image forgery detection system using various metrics such as precision, recall, and F1 score.
- Develop a database for storing the detected forged images and the relevant metadata.
- Implement data visualization tools to analyse the performance of the system.
- The proposed system will be useful in various applications such as journalism, social media, and legal evidence, where the authenticity of images is crucial. The system will also be helpful in forensic investigations, where the detection of manipulated images can aid in solving criminal cases.

## 9. Advantages

- Useful to detect doctored images or signs of forgery in photos.
- Easy to use.
- Maintain security by ensuring only Admin can see the results of image analysis.
- More reliable to detect noisy and lossy images.
- Reduced feature dimension and better accuracy.

## 10.Future Scope

The future scope of image forgery detection includes the following:

**Deep Learning:** The use of deep learning techniques for feature extraction and classification can greatly improve the accuracy of the image forgery detection system. Techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) can be used to automatically learn features from the images. **Additional Forgery Detection Techniques:** There are many additional techniques for detecting image forgeries that could be incorporated into the system. These include detection of image splicing by analysing colour consistency, detection of image tampering by analysing the noise patterns, and detection of image resizing by analysing the distribution of image features.

**Real-Time Processing:** The system can be improved to process images in real-time, allowing it to be used for video forgery detection. This would require optimizing the algorithms to work efficiently on video data.

**Integration with Social Media Platforms:** The image forgery detection system can be integrated with social media platforms to automatically detect and remove forged images. This would require the development of an API that allows the system to communicate with the social media platform.

**Improved User Interface:** The web application interface can be improved to provide a more user-friendly experience. This can include providing feedback to the user on the forgery detection process, providing explanations on the different types of image forgeries, and providing visual aids to help the user understand the results.

**Dataset Expansion:** The system can be improved by expanding the dataset of images used for training and testing. This would involve collecting a larger and more diverse set of images that include a wider range of forgery types.

**Cloud Deployment:** The system can be deployed on the cloud, which would allow it to scale to a large number of users. This would require setting up a cloud-based infrastructure for processing and storing images.

**Multiple Image Inputs:** The system can be extended to accept multiple images as input, allowing it to detect forgeries across multiple images. This could be useful in situations where a set of images is known to be related and any forgery in one image can affect the authenticity of the entire set.

**Mobile Application:** A mobile application can be developed that allows users to upload images for forgery detection. This can expand the reach of the system and provide a more convenient way for users to check the authenticity of their images.

**User-Specific Model:** The system can be extended to train a user-specific model for each user. This would involve training the model with the user's authentic images and would provide more accurate results for that user. In conclusion, the Image Forgery Detection Project in Python Django has a lot of potential for future development. By incorporating deep learning techniques, additional forgery detection techniques, and improving the user interface, the system can be made even more accurate and user-friendly. Additionally, integrating the system with social media platforms and deploying it on the cloud can greatly improve its usefulness and scalability. The system can also be extended to include multiple image inputs, a mobile application, and user-specific models.

## 11. Conclusion

In this project, we have developed an image forgery detection system using Python and Django web framework. The system is capable of detecting various types of image forgeries, including copy-move, splicing, and image tampering.

The system uses image processing techniques for noise reduction and feature enhancement, followed by feature extraction using SIFT or SURF algorithms. The extracted features are then classified as either authentic or forged using a classification algorithm such as SVM or KNN.

The system is developed as a web application using the Django web framework, allowing users to upload images and receive a result indicating whether the image is authentic or forged. The system can be integrated into a larger system, such as a content management system or a social media platform, to automatically detect and remove forged images.

Overall, the Image Forgery Detection Project in Python Django provides a valuable tool for detecting and removing image forgeries from online platforms. Further improvements to the system could include the use of deep learning techniques for feature extraction and classification, as well as the integration of additional image forgery detection techniques such as JPEG compression analysis and noise pattern analysis.

## References

- Farid, H. (2009). Image Forgery Detection. *IEEE Signal Processing Magazine*, 26(2), 16-25.
- Li, Y., Liu, S., & Rockmore, D. (2008). On Detection of Digital Image Forgeries. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 4(3), 1-28.
- Mara, F., Carlini, E., Piva, A., & Barni, M. (2012). A Software Tool for Image Forgery Detection Based on a Statistical Analysis of JPEG Blocking Artifact. *IEEE Transactions on Information Forensics and Security*, 7(6), 1778-1793.
- Li, J., Li, H., Li, J., & Li, X. (2019). Deep Learning for Image Forgery Detection: A Survey. *IEEE Access*, 7, 3363-3379.
- Ross, A., Shah, S., & Jain, A. (2018). Detecting Deepfakes: Methods, Tools, and Challenges. *arXiv preprint arXiv:1812.08685*.
- Dhaka, V., Kaur, H., & Sharma, A. (2020). A Comparative Study of Image Forgery Detection Techniques. *International Journal of Computer Science and Mobile Computing*, 9(5), 8-20.
- Paliwal, S., Dave, M., & Vashisth, A. (2018). Survey of Image Forgery Detection Techniques. *International Journal of Computer Applications*, 181(37), 20-25.
- Karami, A., & Ahmadian, A. (2020). Image Forgery Detection Using Deep Learning Techniques: A Comprehensive Review. *Journal of Ambient Intelligence and Humanized Computing*, 11(6), 2213-2227.
- Image Forgery Detection with Python and OpenCV. (n.d.). Retrieved from <https://towardsdatascience.com/image-forgery-detection-with-python-and-opencv-a3e45b6d1624>.
- Image Forgery Detection. (n.d.). Retrieved from <https://www.pyimagesearch.com/2020/07/27/image-forgery-detection-with-opencv-and-python/>.

## Appendix

### Python 3

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

### JavaScript

JavaScript often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for webpage behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

### phpMyAdmin

phpMyAdmin is a free software tool written in PHP, intended to handle the administration of MySQL over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement

### **MySQL**

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python.

### **OpenCV**

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

### **NumPy**



NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software. It contains various features including these important ones:

### **Xampp Server**

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.