



# A PROJECT REPORT

<b>S. Kanmani</b>	<b>211320104005</b>
<b>S. Sowmiya</b>	<b>211320104009</b>

*Of*

IN

**P. B. COLLEGE OF ENGINEERING, IRUNGATTUKOTTAI**

MAY 2024

**ANNA UNIVERSITY::CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**Identifying Forgery Detection**” is the bonafide work of **S. Kanmani -211320104005** and **S. Sowmiya – 211320104009** who carry out the project work under my supervision.

**SIGNATURE**

**Mrs. T.SUNITHA M.E, MBA, (Ph.D)**

**HEAD OF THE DEPARTMENT  
ASSOCIATE PROFESSOR**

Computer Science & Engineering

P. B. College of Engineering

Irungattukottai

Chennai–602117

**SIGNATURE**

**Mrs. K.L. MUTHAMIZH M.E**

**SUPERVISOR  
ASSISSTANT PROFESSOR**

Computer Science & Engineering

P. B. College of Engineering

Irungattukottai

Chennai –602117

Submitted for the Anna University project evaluation held on \_\_\_\_\_

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

I would like to thank whole heartedly and express our sincere gratitude to our beloved chairman **Sri G.VENKATARAMAN**, respected Director **Dr. G.SUBBALAKSHMI, Ph.D.**, for their constant encouragement in the development of this project.

I would like to take the opportunity to thank our beloved Principal **Dr P.SENTHIL KUMAR M.E, MBA, Ph.D.**, for providing a great support focus in completing our project and for giving us the opportunity of doing the project.

I profusely thank our project coordinator, **Mrs. K.L.MUTHAMIZH M.E**, Assistant Professor, Computer Science and Engineering, for his exhilarating supervision, timely suggestions and encouragement during all phases of this work.

I thank our guide **Mrs. T.SUNITHA M.E, MBA, (Ph.D)** Head of the Department for his assistance in completing this project.

I thank all our staff members of CSE department for their assistance in completing this project .Profound thanks goes our family members and friends for their help in this end.

## ABSTRACT

Digital images are susceptible to a range of vulnerabilities and threats that can compromise security and privacy in online social networking sites. Image tampering attacks involve the unauthorized or deceptive alteration of digital images, often for the purpose of misrepresenting their content or context. Once the images are manipulated, it is hard for current techniques to reproduce the original contents. Image Processing and Machine Learning techniques have bolstered image forgery detection, primarily focusing on noise-level manipulation detection. In this context, this project introduces an enhanced scheme known as Image Immunizer for image tampering resistance and lossless auto – recovery using Vaccinator and Invertible Neural Network a Deep Learning Approach. Multitask learning is used to train the network, encompassing four key modules: apply vaccine to the uploaded image, ensuring consistency between the immunized and original images, classifying tampered pixels, and encouraging image self-recovery to closely resemble the original image. During the forward pass, both the original image and its corresponding edge map undergo transformation, resulting in the creation of an immunized version. Upon receiving an attacked image, a localizer identifies tampered areas by predicting a tamper mask. This proposed technique achieves promising results in real-world tests where experiments show accurate tamper localization as well as high-fidelity content recovery.

## TABLE OF CONTENT

C. NO	TITLE	PAGE NO
	<b>ABSTRACT</b>	iv
	<b>LIST OF FIGURES</b>	viii
	<b>LIST OF SYMBOLS</b>	ix
	<b>LIST OF ABBREVIATIONS</b>	x
<b>1</b>	<b>INTRODUCTION</b>	1
	1.1. General Introduction	1
	1.1.1. Purpose of Social networking	2
	1.1.2. Social media Privacy issues	2
	1.2. Importance of the Study	4
	1.3. Problem Statement	5
	1.4. Aim and Objective	6
	1.5. Scope and Limitation of the study	7
	1.6. Contribution of the Study	8
<b>2</b>	<b>LITERATURE REVIEW</b>	9
<b>3</b>	<b>SYSTEM ANALYSIS</b>	17
	3.1. Existing System	17
	3.1.1. Disadvantages	18
	3.2. Proposed System	18
	3.2.1. Social Network web App	18
	3.2.4. Image Immunizer Middleware	21
	3.2.4.1. Cyber Vaccinator	21
	3.2.4.2. Shared Image Detector	22
	3.2.5. Objective Loss Function	23
	3.2.6. Notification	23

	3.2.7. Advantages	24
<b>4</b>	<b>SYSTEM CONFIGURATION</b>	25
	4.1. Hardware Requirements	25
	4.2. Software Requirements	25
<b>5</b>	<b>SYSTEM DESIGN</b>	26
	5.1. System Architecture	26
	5.2. Use Case	27
	5.3. Class Diagram	28
	5.4. Activity Diagram	29
	5.5. Sequence Diagram	30
	5.6. Collaboration Diagram	31
	5.7. Deployment Diagram	32
	5.8. ER Diagram	33
	5.9. Data Flow Diagram	34
<b>6</b>	<b>PROJECT DESCRIPTION</b>	35
	6.1. Deep Learning	35
	6.1.1. Multi Task Learning	35
	6.2. Invertible Neural Network	36
	6.3. Dataset Details	37
	6.4. Performance Analysis	38
	6.5. Performance Measures	39
	6.6. Input & Output	41
	6.6.1. Input Design	41
	6.6.2. Output Design	42
<b>7</b>	<b>IMPLEMENTATION</b>	44
	7.1. Source Code	44
<b>8</b>	<b>TESTING</b>	61

	8.1. Testing Objectives	61
	8.2. Types Of Testing	61
	8.2.1. Unit Testing	61
	8.2.2. Functional Testing	61
	8.2.3. White Box Testing	61
<b>9</b>	<b>CONCLUSION &amp; FUTURE ENHANCEMENT</b>	62
<b>10</b>	<b>EXPERIMENTAL RESULTS</b>	63
	10.1. Results	63
	10.2. Experimental Results	63
<b>11</b>	<b>REFERENCES</b>	73

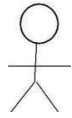
## LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO
5.1	Architecture Diagram	26
5.2	Use Case Diagram	27
5.3	Class Diagram	28
5.4	Activity Diagram	29
5.5	Sequence Diagram	30
5.6	Collaboration Diagram	31
5.7	Deployment Diagram	32
5.8	ER Diagram	33
5.9	Data Flow Diagram	34
10.2.1	User Signup	63
10.2.2	User Signin	64
10.2.3	Home Page	65
10.2.4	Need To Vaccinate	65
10.2.5	Social Media Signin	66
10.2.6	Dashboard	66
10.2.7	Subjected To Image Tampering	67
10.2.8	Tampered Section	67
10.2.9	Marking Tampering Layer	68
10.2.12	Original Image	69
10.2.13	Upload Post	70
10.2.15	Login For Image Immunizer	71
10.2.16	Report For Image	71
10.2.17	Tamper Layer Graph	72
10.2.18	Tamper with highlight	72



## LIST OF SYMBOLS

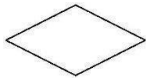
### SYMBOLS



Actor



Flow Direction



Decision



Initial



Final

## LIST OF ABBREVIATIONS

OSN	-	Open Storage Network
INN	-	Invertible Neural Network
ELA	-	Error Level Analysis
RLE	-	Run-Length Encoding
PSNR	-	Peal Signal-To-Noise Ratio
MSE	-	Mean Squared Error
IDE	-	Integrated Development Environment
MTL	-	Multi Task Learning
SRM	-	Spatial Rich Model
SSI	-	Structural Similarity Index

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1. GENERAL INTRODUCTION**

Social networking refers to using internet-based social media sites to stay connected with friends, family, colleagues, or customers. Social networking can have a social purpose, a business purpose, or both through sites like Facebook, Twitter, Instagram, and Pinterest. Social networking is also a significant opportunity for marketers seeking to engage customers. Facebook remains the largest and most popular social network, with 2 billion people using the platform daily, as of Feb 1, 2023.<sup>1</sup> Other popular platforms in the U.S. are Instagram, Twitter, WhatsApp, TikTok, and Pinterest. With the broad spectrum of websites, apps and services that exist online, there is no single exact definition of a social network. Generally, though, social networks have a few common attributes that set them apart. A social network will focus on user-generated content. Users primarily view and interact with content made by other users. They are encouraged to post text, status updates or pictures for viewing by others. Social networks allow the user or organization to create a profile. The profile contains information about the person and a centralized page with the content posted by them. Their profile may be associated with their real name. A social network has a way to form a lasting connection with other users. These connections are commonly called friending or following the other user. They allow the users to find other users and form webs of relationships. Often an algorithm will recommend other users and organizations they may want to form a connection with. Although often used interchangeably, social network is different than social media. A social network focuses on the connections and relationships between individuals. Social media is more focused on an individual sharing with a large

audience. In this case, media is used in the same sense as in mass media. Most social networks can also be used as social media sites.

### **1.1.1 Purpose of Social networking**

Social networking fulfils the following four main objectives:

- **Sharing.** Friends or family members who are geographically dispersed can connect remotely and share information, updates, photos and videos. Social networking also enables individuals to meet other people with similar interests or to expand their current social networks.
- **Learning.** Social networks serve as great learning platforms. Consumers can instantly receive breaking news, get updates regarding friends and family, or learn about what's happening in their community.
- **Interacting.** Social networking enhances user interactions by breaking the barriers of time and distance. With cloud-based video communication technologies such as WhatsApp or Instagram Live, people can talk face to face with anyone in the world.
- **Marketing.** Companies may tap into social networking services to enhance brand awareness with the platform's users, improve customer retention and conversion rates, and promote brand and voice identity.

### **1.1.2 Social Media Privacy Issues**

One of the important factors to consider is that while social media has several advantages, another side of the story needs awareness. Social media is often referred to as a

double-edged sword. The impact of social media is often subtle and becomes more pronounced with repeated usage. Social media and privacy are mostly incompatible

- **Oversharing Personal Information :** When it comes to social media privacy, we are our own first line of defence. Even if you have your profile set to

viewing by friends only, you should be conservative with the information you share.

Information you should never add to your social media profile includes:

- Your home address
- Your phone number or personal email address
- Any financial information
- Exact location tags
- Images that make your house identifiable from the street
- Images of children that make their schools identifiable
- Exact travel dates and information about when you'll be out of the house

Social media companies regularly change their policies and features — and some of those changes can cause serious data privacy issues.

- **Social Media Privacy: The Friend Loophole**

While it may seem harmless to add people you don't know on social media, this can open you up to various privacy issues.

For example, scammers may use fake accounts for spear-phishing campaigns. By adding a person on a platform such as Facebook, they can glean information that will make scam emails more convincing.

Another way it can compromise your security is when apps are able to access your information through a friend's account. This is what made the Cambridge Analytica scandal so shocking. Most people whose information was harvested didn't even take the quiz that was used to mine data.

Rather, the Facebook quiz used a loophole to access the information of all the friends of people who took the quiz. While Facebook has tightened up privacy on the network in this regard, the various ways friends can compromise our privacy is still very much a concern.

- **Privacy setting loopholes**

Data privacy is an important issue. Most social media companies amended their privacy policies in response to stricter privacy laws and regulations in Europe. They now allow you to tweak your settings and make your accounts more private. What that you, changing your privacy settings doesn't always guarantee privacy? Most of the time, something you shared with a closed group of friends gives them the ability to share it with others. Your friend's can then see the content you posted, which might not be your intention. Your friends might not even have stringent privacy policies, meaning that others can now access information that was supposed to stay within your friends' circle.

The same applies to closed social groups and forums. Sky News has previously found that comments and user lists in private health groups on Facebook could be easily searchable and discoverable by insurance companies and employers. So think twice before posting or commenting about controversial issues. That information could ruin your reputation or lead to identity theft.

- **Trouble with privacy**

The trouble with privacy is arguably one of the most significantly experienced disadvantages. when listing the advantages and disadvantages of social media. Tweeting inappropriate information, sharing too much of your everyday life with a large audience, or unknowingly sharing your online location are some of the disadvantages of frequent social media usage.

It is also important to note that security agencies worldwide have access to all the information you post on social media platforms. As a result, your privacy is almost always compromised.

## **1.2 IMPORTANCE OF THE STUDY**

The significance of the Image Immunizer Middleware for Online Social Networks lies in safeguarding the integrity and security of shared images within the dynamic landscape of social media platforms.

- **Ensuring Image Integrity:** With the proliferation of digital manipulation techniques, ensuring the integrity of shared images on social media platforms has become crucial. This project addresses this concern by providing a robust mechanism to detect and counter digital attacks, thereby safeguarding the authenticity of shared images.
- **Protecting User Trust:** Users often share personal and sensitive images on social media, expecting them to remain unaltered. By implementing advanced techniques such as Invertible Neural Networks and tamper detection algorithms, this project helps maintain user trust by ensuring that shared images are not maliciously tampered with.
- **Combating Digital Manipulation:** The project's focus on detecting various forms of digital manipulation, including copy-move, splicing, and inpainting attacks, contributes to the broader effort of combating misinformation and deceptive practices prevalent on social media platforms.
- **Enhancing Online Security:** By integrating with existing OSN architectures and providing user awareness notifications, the project contributes to enhancing overall online security. Users are empowered with information about the vaccination status and integrity of shared images, enabling them to make informed decisions about their online interactions.
- **Promoting Ethical Image Sharing:** In a digital landscape where image manipulation can distort reality, promoting ethical image sharing practices is essential. This project encourages responsible image sharing by providing users with tools to verify the authenticity of shared images and discouraging the spread of manipulated content.

### 1.3 PROBLEM STATEMENT

One problem with photo sharing privacy and security issues on social networking websites is the potential for unauthorized access to user's personal information and images. This can happen in a variety of ways, such as through

hacking or data breaches, or through the misuse of data by third-party apps or advertisers. Another issue is the lack of control that users have over their own content once it is posted online. Even if a user sets their account to private, their photos may still be accessible to others if someone they have granted access to their account shares the photos or if the platform's privacy settings are not sufficient. There is also the risk of online harassment and cyberbullying, which can be facilitated by the sharing of photos and personal information on social media. This can have serious consequences for individuals' mental health and well-being, and can even lead to physical harm in extreme cases. Finally, social media platforms may also use facial recognition technology to automatically tag users in photos, which raises concerns about privacy and the potential for misuse of this technology.

#### **1.4 AIM AND OBJECTIVE**

The aim of the Image Immunizer Middleware for Online Social Networks (OSN) using Invertible Neural Network (INN) is to significantly enhance the security and integrity of images shared on social media platforms. The middleware aims to provide a robust defense against potential manipulations and attacks, ensuring the authenticity of shared content within the dynamic and interactive context of online social networks.

##### **Objective.**

- To establish binary masks for object contours.
- To design Forward Pass for Tamper Detection.
- To develop a Localizer for Tampered Area Detection.
- To create Backward Pass for Image Self-Recovery.
- To simulate Adversarial Attacks for Effective Network Training.
- To encourage Image Self-Recovery Mechanisms.



- To achieve Proactive Image Immunization and Restoration.

## 1.5 SCOPE AND LIMITATION OF THE STUDY

The project aims to address critical challenges associated with ensuring the integrity and security of images shared on Online Social Networks (OSNs). The primary areas within the scope of the project include:

- **Image Immunization:** Development of a robust middleware utilizing Invertible Neural Network (INN) to immunize shared images against various potential digital attacks, including copy-move, splicing, and inpainting.
- **User-Friendly Integration:** Seamless integration with existing OSN architectures to ensure accessibility and usability across diverse social media platforms.
- **Adversarial Simulation:** Incorporation of adversarial simulation during the training phase to enhance the system's resilience against evolving threats within the dynamic OSN environment.
- **Performance Evaluation:** In-depth analysis of performance metrics, such as Peak Signal-to-Noise Ratio (PSNR) and OSN-specific metrics, to assess the effectiveness of the immunization and recovery processes in real-world scenarios.

### Limitations of the Project

While the project has an ambitious scope, there are certain limitations that need consideration:

- **Data Quality Dependency:** The system's efficacy relies on the quality and diversity of training data. Insufficient or biased data may impact the system's ability to accurately detect and respond to novel attack patterns.
- **Computational Resources:** Advanced techniques like Invertible Neural Networks may demand significant computational resources, potentially limiting deployment in resource-constrained environments.

- **Dynamic Attack Landscape:** The continuous evolution of digital attacks poses a challenge, requiring regular updates to address emerging attack techniques effectively.

## 1.6 CONTRIBUTION OF THE STUDY

The study makes several significant contributions to the field of image security and integrity within Online Social Networks (OSNs):

- **Enhanced Image Security:** By developing a robust middleware using advanced techniques like Invertible Neural Networks, the study contributes to enhancing the security of images shared on OSNs. This helps prevent unauthorized modifications and ensures the integrity of shared content.
- **Adaptive Defense Mechanisms:** The incorporation of adversarial simulation during the training phase enables the system to adapt and respond to evolving digital threats. This adaptive defense mechanism enhances the system's resilience against various types of attacks.
- **User Awareness and Control:** The implementation of notification mechanisms and user decision prompts ensures that users are aware of the vaccination status of shared images and have control over their sharing decisions. This empowers users to make informed choices about sharing content on social media platforms

## CHAPTER 2

### LITERATURE REVIEW

#### **2.1 Title: Enhanced Tamper Detection Algorithm using YOLOv5s with CBAM Attention and EIOU Loss**

**Author:** Zhen Liu

**Year:** 2023

**Reference Link:** <https://ieeexplore.ieee.org/document/10238704>

#### **Problem:**

Traditional tamper detection models often struggle with unknown tampering modes, resulting in suboptimal performance. Specifically, the extraction of features from tampered regions is challenging when relying on hand-defined features, especially in complex scenarios.

#### **Objective:**

The objective of this paper is to improve the performance of tamper detection by enhancing the feature extraction capability of the model, particularly when dealing with unknown tampering modes. The focus is on integrating the CBAM attention module into the Neck layer of the YOLOv5s model and optimizing the boundary frame loss using the EIOU loss function.

#### **Methodology:**

The embedding the Neck layer of the YOLOv5s model with the CBAM attention module. This modification aims to enhance the model's ability to capture features from tampered regions. Additionally, the EIOU loss function is employed to optimize the boundary frame loss, accounting for the impact factors of aspect ratio in the calculation process.

#### **Algorithm/Techniques:**

The YOLOv5s with the CBAM attention module integrated into the Neck layer. The EIOU loss function is employed for optimizing the boundary frame loss, enhancing the convergence speed and accuracy of the network.

**Merits:**

The algorithm effectively identifies various tampering modes. the average accuracy of recognition is increased by 1.57% compared to the benchmark algorithm. Recognition speed outperforms LSTM and U-Net algorithms, reaching 13.89 images per second. the algorithm surpasses traditional tamper detection methods (ELA, CFA1, and NOI1) in terms of average accuracy improvement.

**Demerits:**

The paper does not provide information about the dataset used, limiting the ability to assess the algorithm's generalizability. Specific details about the CBAM attention module integration and EIOU loss function optimization are not thoroughly explained. the paper lacks information on computational requirements or potential limitations of the proposed algorithm. the paper may lack a comprehensive comparative analysis with other state-of-the-art tamper detection algorithms, making it challenging to assess its performance relative to the broader research landscape.

**2.2 Title: Content Authentication and Tampered Localization Using Ring Partition and CSLBP-Based Image Hashing**

**Author:** Abdul Shaik ; Ram Karsh

**Year:** 2023

**Reference Link:** <https://ieeexplore.ieee.org/document/10311570>

**Problem:**

The problem addressed in this paper is content authentication and tampered localization in digital images. The goal is to develop a robust image hashing method that can resist geometric operations, including rotation, and effectively detect and localize small tampering areas.

The objective of the study is to propose a perceptual image hashing technique using ring partition and CSLBP (Circular Symmetric Local Binary Pattern). The

proposed method aims to provide stability, rotation invariance, and better performance in terms of robustness and discrimination compared to existing approaches.

### **Methodology:**

The methodology involves converting the input image to a standardized form and extracting ring-based statistical features using Circular Symmetric Local Binary Pattern (CSLBP). Crucially, the chosen ring partition technique ensures feature stability and rotation invariance. The algorithm strategically combines ring partition and CSLBP to create a robust image hashing method resistant to geometric operations like rotation. Experimental results demonstrate superior performance in robustness and tamper localization, though a limitation is noted in the dependency on hash length reduction for improvement.

### **Algorithm/Techniques:**

The employs ring partition and Circular Symmetric Local Binary Pattern (CSLBP) for perceptual image hashing. Ring partition preserves feature stability during image rotation, and CSLBP extracts circular symmetric local binary patterns. This combined approach establishes a robust hashing method resistant to geometric operations, particularly rotation. Experimental results highlight its superior performance in robustness and tamper localization, while a limitation involves the necessity of hash length reduction for potential improvement.

### **Merits:**

**Stability and Rotation Invariance** the proposed hashing method demonstrates stability and rotation invariance, making it resistant to geometric operations like rotation. **Better Performance** the results indicate that the proposed hashing method offers better performance in terms of robustness and discrimination compared to other existing approaches. **Small Tampering Area Localization** the method is sensitive to changes in visual information, allowing for the detection and localization of small tampering areas.

**Demerits:**

Hash Length Limitation: The major limitation of the proposed method is the need to reduce the hash length for improvement. This suggests a potential trade-off between hash length and performance.

**2.3 Title: Generative Facial Prior and Semantic Guidance for Iterative Face Inpainting**

**Author:** Xin-Yu Zhang ; Kai Xie ; Mei-Ran Li

**Year:** 2022

**Reference Link:** <https://ieeexplore.ieee.org/document/9803044>

**Problem:**

Image inpainting techniques have traditionally relied on structure and texture priors, but damaged or incomplete original images often lack sufficient texture information and accurate structural priors. Additionally, addressing facial symmetry and attribute consistency is crucial for human visual perception. The paper aims to enhance image inpainting quality by addressing these challenges through a novel approach.

**Objective:**

The objective of the paper is to present a face inpainting system with an iterative structure, guided by generative facial priors obtained from pretrained GANs and predicted semantic information. The focus is on improving facial detail, attribute consistency, and symmetry in inpainted images.

**Methodology:**

The involves iteratively refining images multiple times, updating semantic maps at each iteration. The Weighted Prior-Guidance Modulation layer (WPGM) is introduced to incorporate priors into networks through spatial modulation. The facial feature self-symmetry loss is devised to constrain the symmetry of faces in feature space. Experiments are conducted on CelebA-HQ and LaPa datasets to validate the model's performance.

**Algorithm/Techniques:**

The algorithm combines generative facial priors from GAN inversion techniques and predicted semantic information in an iterative face inpainting system. It employs the Weighted Prior-Guidance Modulation layer (WPGM) to spatially modulate networks, incorporating facial priors effectively. A facial feature self-symmetry loss is introduced to ensure the symmetry of completed faces in feature space, contributing to the overall stability and superior performance of the model in detailed textures and consistent attributes.

**Merits:**

Superior facial detail and attribute consistency in inpainted images. Integration of generative facial priors and semantic information for refinement. Introduction of the Weighted Prior-Guidance Modulation layer for effective spatial modulation. Facial feature self-symmetry loss to ensure symmetry in face completion.

**Demerits:**

The paper does not specify the author's name, year of publication, or provide a reference link. The specific GAN inversion techniques used are not detailed. Limited information on the dataset characteristics and preprocessing.

**2.4 Title: A Novel Multipurpose Watermarking Scheme Capable of Protecting and Authenticating Images With Tamper Detection and Localisation Abilities**

**Author:** Sunpreet Sharma

**Year:** 2022

**Reference Link:** <https://ieeexplore.ieee.org/document/9857940>

**Problem:**

The paper addresses the issue of protecting and authenticating data, particularly images, transmitted within the context of Industry 4.0. With the rise of technologies like cyber-physical systems, the Internet of Things (IoT), and

cognitive computing, sensitive data is shared through images, videos, and various signals, making it susceptible to unauthorized access and tampering.

**Objective:**

The primary objective of the proposed scheme is to provide a multipurpose image watermarking solution tailored for Industry 4.0. The scheme aims to authenticate transmitted images, protect copyright, and detect and localize tampering.

**Methodology:**

The scheme introduces two watermarking methods: one for embedding a robust watermark and another for a fragile watermark. The robust watermark is embedded in the frequency domain using a novel mean-based coefficient selection procedure, while the fragile watermark is embedded in the spatial domain by directly altering pixel bits. The combination of these two methods creates a hybrid scheme that balances imperceptibility, security, and capacity.

**Algorithm/Techniques:**

The watermarking scheme employs a multifaceted approach, integrating two distinct techniques to address the diverse challenges posed by protecting and authenticating images in the context of Industry 4.0. The first technique involves the embedding of a robust watermark in the frequency domain. This process begins with a unique mean-based coefficient selection procedure, strategically identifying frequency coefficients. Subsequently, these selected coefficients undergo manipulation in a proportional manner to seamlessly integrate the robust watermark.

**Merits:**

Excellent imperceptibility of watermarked images with high PSNR and SSIM values. Outperforms existing state-of-the-art methods when tested on datasets. Demonstrates resourcefulness by handling a wide range of image resolutions. Robust watermarking performance surpasses existing methods, especially for smaller watermark



**Demerits:**

The paper does not explicitly mention any demerits or limitations. Further examination or external reviews might be necessary to identify potential drawbacks or areas for improvement.

**2.5 Title: TransU2-Net: A Hybrid Transformer Architecture for Image Splicing Forgery Detection**

**Author:** Caipng yang ; shuyuan ; hong Li

**Year:** 2021

**Reference Link:** <https://ieeexplore.ieee.org/document/10090941>

**Problem:**

The existing convolutional neural network (CNN) based frameworks for detecting forged regions in images often struggle with satisfactory performance, especially in handling tampered areas of various sizes, particularly in the case of large-scale objects.

**Objective:**

To address the limitations of existing models and achieve accurate object-level forgery localization, the paper proposes a novel hybrid transformer architecture, TransU2-Net. The objective is to leverage both spatial dependencies and contextual information from different scales for improved performance in detecting spliced forgeries with various sizes, without the need for large-scale data set pre-training.

**Methodology:**

The TransU2-Net integrates self-attention and cross-attention into U2-Net, capturing long-range semantic dependencies and avoiding heavy reliance on large-scale pre-training. The model utilizes skip connections to filter out non-semantic features and enhance low-level features, achieving more refined spatial recovery. The hybrid model aims to provide a comprehensive solution for image splicing forgery detection.

**Algorithm/Techniques:**

Hybrid transformer architecture, integrating self-attention and cross-attention into the U2-Net framework. This combination allows the model to capture both long-range semantic dependencies and fine spatial details without heavy reliance on extensive pre-training. Additionally, the introduction of a cross-attention module enhances low-level feature maps in the decoder, contributing to more effective and generalizable detection of complex image forgery. The resulting algorithm demonstrates superior performance in locating forged regions with varying scales, surpassing existing methods on standard image forgery datasets.

**Merits:**

Achieves better performance over state-of-the-art methods, with an 8.4% improvement in F-measure on the Casia 2.0 dataset. Integrates both self-attention and cross-attention, reducing the reliance on large-scale pre-training. Capable of locating forged areas with different scales, addressing challenges posed by tampered areas of various sizes.

**Demerits:**

Issues with generalization for cross-dataset testing. Limited detection capability applicable only to the detection of splicing images. The need for further improvement in the universality of the algorithm model.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1. EXISTING SYSTEM**

- **Digital Forensics Techniques**

Digital forensics involves the examination of digital evidence to identify and analyse patterns of forgery. This may include analysing metadata, compression artefacts, and other forensic traces within the image.

- **Signature-Based Approaches**

Signature-based methods use known patterns or signatures of forgery to identify manipulated images. These signatures may include noise patterns, repeated patterns, or specific features associated with common tampering techniques.

- **Watermarking**

Watermarking involves embedding invisible or visible marks within an image to identify its origin or ownership. Watermarks can help in tracking and verifying the authenticity of images.

- **Image Hashing**

Image hashing involves generating a unique hash or fingerprint for an image. Any alterations to the image, even minor ones, result in a significant change in the hash value, allowing for the detection of forgery.

- **Image Processing Algorithms**

Image processing techniques, including error level analysis (ELA) and noise analysis, are used to identify inconsistencies in pixel values or compression artefacts that may suggest tampering.

- **Steganalysis**

Steganalysis techniques focus on detecting hidden information within images. This includes identifying alterations made through steganography, where additional information is concealed within the image.

- **Block chain for Image Authentication**

Description: Some systems use block chain technology to timestamp and authenticate images. This ensures that the image's origin and content remain unchanged over time, providing a form of tamper-proofing.

### **3.1.1 DISADVANTAGES**

- Traditional methods may struggle with subtle manipulations, impacting detection precision.
- Face challenges in detecting sophisticated deepfake content.
- Computational complexity, hindering real-time implementation.
- Vulnerability to sophisticated steganography methods.
- Occurrence of false positives/negatives affecting detection accuracy.
- Challenges in generalizing across diverse image types.
- Absence of automatic or self-recovery mechanisms, requiring manual intervention for content restoration.

## **3.2 PROPOSED SYSTEM**

The Image Immunizer Middleware for Online Social Networks (OSN) using Invertible Neural Network (INN) is designed to enhance the security and integrity of images shared on social media platforms. The proposed system comprises several key modules and functionalities to achieve this objective:

### **3.2.1 Social Networking Web App**

The social networking web app is meticulously crafted using Python, Flask, MySQL, Bootstrap, and Wampserver 2i to deliver a secure, responsive, and feature-rich user experience. The User Authentication module guarantees secure access, employing features such as user registration, login, password hashing, and two-factor authentication. The User Profile module fosters personalization, allowing users to create and customize profiles with responsive design elements. The heart of the platform lies in the Media Sharing module, where users can seamlessly upload and share images, creating a dynamic and visually appealing

environment. Connection Management facilitates user interactions, incorporating friend requests, group creation, and an effective notification system. Direct communication is enhanced through the Messaging and Chat module, offering real-time messaging, multimedia file sharing, and group chat functionalities. Admin Panel module centralizes control and management.

### **3.2.2 End User Interface**

The End User Interface module provides a seamless and intuitive experience for social network users, encompassing essential functionalities such as registration, login, social connections, image sharing, download, and interaction with shared content. The module also includes features for applying digital attacks to images, sharing tampered content, and receiving notifications.

- **Registration**

Allows users to create an account by providing necessary details such as username, email, and password. Registration establishes a unique identity for each user within the social network.

- **Login**

Enables users to log into their accounts using valid credentials. A secure authentication process grants access to the user's personalized space within the social network.

- **Add Friends**

Facilitates the expansion of social connections by allowing users to search for and add friends within the network. This enhances the social experience and connectivity among users.

- **Share Image or Photos**

Permits users to upload and share images or photos with their network of friends. The sharing process may involve adding captions or tags to provide context and enhance content discovery.

- **Download Shared Photos or Images:**

Allows users to download images or photos shared by their friends. Downloaded content is stored locally, providing users with the ability to save and view shared media.

- **Apply Digital Attack on Shared Image:**

Introduces a unique feature that enables users to apply digital attacks on shared images. Users can experiment with various tampering techniques, enhancing their understanding of potential threats.

- **Share Tampered Image or Photo to Other Social Networks:**

Allows users to share tampered images or photos to other social networks. This feature extends the reach of tampered content beyond the immediate social network, showcasing the potential impact of digital attacks.

- **Receive Notifications:**

Notifies users of relevant activities, such as new friend requests, shared content, or interactions with their posts. Notifications enhance user engagement and keep users informed about network activities.

### **3.2.3 Adversarial Simulation: Training Against Threats**

The system employs adversarial simulation, leveraging the capabilities of the Invertible Neural Network, to fortify its resilience against potential threats. Three malicious attacks - copy-move, splicing, and benign attacks like rescaling and blurring - are simulated during the training process. The Invertible Neural Network, is well-prepared to detect and counteract a diverse array of potential attacks, thereby enhancing its robustness in maintaining the integrity of the digital landscape.

- **Copy-Move Attacks:**

Copy-move attacks involve the unauthorized duplication of specific regions within an image. In this manipulation, a segment of the image is copied and pasted either within the same image or into another, creating a deceptive appearance of

repeated content. Perpetrators may use this technique to conceal or replicate objects, distort information, or compromise the authenticity of the image.

- **Splicing Attacks:**

Splicing attacks entail the fusion of content from different sources into a single image. This manipulation is often more sophisticated, involving the seamless integration of objects, people, or scenes from separate images. Splicing aims to create a misleading composite that appears genuine but can distort the intended meaning or context of the visual content.

- **In painting Attacks:**

In painting attacks involve the intricate process of reconstructing missing or tampered areas within an image. This technique is used to conceal unwanted elements, repair damaged portions, or fill gaps in the image seamlessly. In painting aims to make alterations indistinguishable, creating a visually coherent result that may deceive viewers about the image's true content or history.

### **3.2.4 Image Immunizer Middleware**

The Image Immunizer Middleware is a crucial component within a system designed to enhance the security and integrity of digital images. This middleware employs Cyber Vaccinator Framework to discern between vaccinated and unvaccinated, if unvaccinated means transform an original image to its edge map into an immunized version. Invertible Neural Network (INN) employs Forward pass to determine tampered areas by predicting the tamper mask and type of attack. Backward pass the recovery of the original image and its associated metadata. The module operates in real-time, seamlessly integrating into the image processing pipeline. The key functionalities include:

#### **3.2.4.1 Cyber Vaccinator**

Within the Cyber Vaccinator framework, the utilization of an Invertible Neural Network (INN) enhances the system's ability to preserve media authenticity through a sequence of pre-processing, mid-processing, and post-processing steps.

- **Vaccine Validator**

Integrated into the system is the Vaccine Validator, a critical component designed to discern between vaccinated and unvaccinated media. This validator plays a pivotal role in safeguarding the authenticity of digital content.

- **Vaccinator**

The Vaccinator incorporates an Invertible Neural Network to transform an original image and its edge map into an immunized version. In the pre-processing stage, landmarks are detected via image alignment algorithms, crafting a binary mask that classifies pixels inside or outside the object contour. The mid-processing step involves the Invertible Neural Network, which takes the original portrait and mask as inputs, generating a raw output. Post-processing then seamlessly replaces the object region in the raw output with the original portrait, guided by the mask. Crucially, imperceptible perturbations are strategically introduced solely to the non-object region, ensuring a nuanced balance between maintaining originality and embedding vital object region information.

### **3.2.4.2 Shared Image Detector**

The module monitors the social network for activities related to image sharing. When it detects a shared vaccinated image without any attack, it initiates the notification process. When it detects a shared vaccinated image with attack, it transfers the attacked image to the forward pass of INN.

- **Forward Pass with Tamper Detection and Localization**

In the face of an attacked image, a localizer comes into play, determining tampered areas by predicting the tamper mask and attack. This vigilance is instrumental in identifying and localizing digital tampering, fortifying the system against a spectrum of potential attacks.

- **Backward Pass for Image Self-Recovery**

In the backward pass, the hidden perturbation, transformed by the Invertible Neural Network, is converted into information. This transformative process



serves the dual purpose of recovering the original image and its edge map while fostering image self-recovery. By encouraging the recovered image to closely resemble the original, this module ensures a seamless restoration process.

### **3.2.5 Objective Loss Function**

The loss function measures the error or difference between the predicted output of a model and the actual target values. **Run Length Encoding**

Lossless image recovery using Run-Length Encoding (RLE) is a technique that focuses on preserving the original image data while achieving efficient image recovery. Run-Length Encoding (RLE) can be a valuable tool in achieving this, ensuring that the original image is restored without loss of information after tampering has been detected and addressed. Subsequent to tamper removal, the image is subjected to RLE compression. Runs of consecutive identical pixel values are encoded to represent sequences more efficiently. The integration of tamper detection, removal, and lossless recovery using RLE enhances the overall resilience of the system against malicious manipulations.

### **PSNR**

The Peak Signal-to-Noise Ratio (PSNR) is a metric used to quantitatively assess the quality of an image reconstruction or recovery compared to the original image. It measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the quality of the signal.

- **PSNR Calculation**

PSNR is calculated using the formula:  $PSNR = 10 * \log_{10}((MAX^2) / MSE)$ , where MAX is the maximum possible pixel value (255 for an 8-bit image), and MSE is the Mean Squared Error between the original and recovered images.

### **3.2.6 Notification**

The Notification Module serves as a vital component in keeping users informed and empowered when it comes to shared vaccinated images on other social networks. Specifically, when a user downloads and shares a vaccinated image

without any attack, the Image Immunizer detects the shared image and triggers an email notification to the user. The email prompts the user to make a decision regarding the shared image.

- **Email Notification to User:**

An email notification is sent to the user who shared the vaccinated image. The email informs the user about the shared image and includes a prompt to take action.

- **Prompt for User Decision:**

The email includes a clear prompt asking the user to make a decision regarding the shared image. The user is presented with two options: to remove the image or to confirm its continued sharing.

- **User Response Handling:**

Based on the user's decision, the system takes appropriate actions. If the user chooses to remove the image, the system initiates the removal process. If the user decides to keep the image shared, no further action is taken.

- **Confirmation Message:**

After the user makes a decision, a confirmation message is sent to the user to acknowledge their choice. This ensures transparency and keeps the user informed about the outcome of their decision.

### **3.2.7 ADVANTAGES**

- Accurate tamper detection fortifies shared images against potential threats.
- Preserves original image quality during the recovery process.
- Prepares the system for OSN-specific threats, including deepfake attempts.
- Easily integrates with existing OSN architectures for widespread adoption.
- Tamper Resilience: Enhances resistance against unauthorized alterations.
- Ensures recovery without loss of original image information.
- Improves the security of digital images against tampering threats.
- Applicable across domains such as forensics and digital communication.

## **CHAPTER 4**

### **SYSTEM CONFIGURATION**

#### **4.1. HARDWARE REQUIREMENTS**

- **Processor:**

Quad-core or higher processor for efficient parallel processing, capable of handling complex image transformations and neural network computations.

- **RAM:**

8 GB RAM to facilitate seamless image processing and provide ample memory for neural network training and inference tasks.

- **Storage:**

Solid State Drive with a minimum capacity of 256 GB for fast data access and storage of datasets, trained models, and system files.

#### **4.2. SOFTWARE REQUIREMENTS**

- **Operating System:** Windows 10 or 11 (for Windows-specific development)
- **Programming Language:** Python (version 3.6 or higher)
- **Neural Network Framework:** TensorFlow or PyTorch
- **Image Processing Libraries:** OpenCV and PIL (Pillow)
- **Web Framework:** Flask for implementing a web-based user interface,
- **Database Integration:** MySQL for storing and retrieving relevant data.
- **Integrated Development Environment (IDE):** IDLE
- **Web Technologies:** HTML, CSS, and JavaScript

## CHAPTER 5

### SYSTEM DESIGN

#### 5.1. SYSTEM ARCHITECTURE

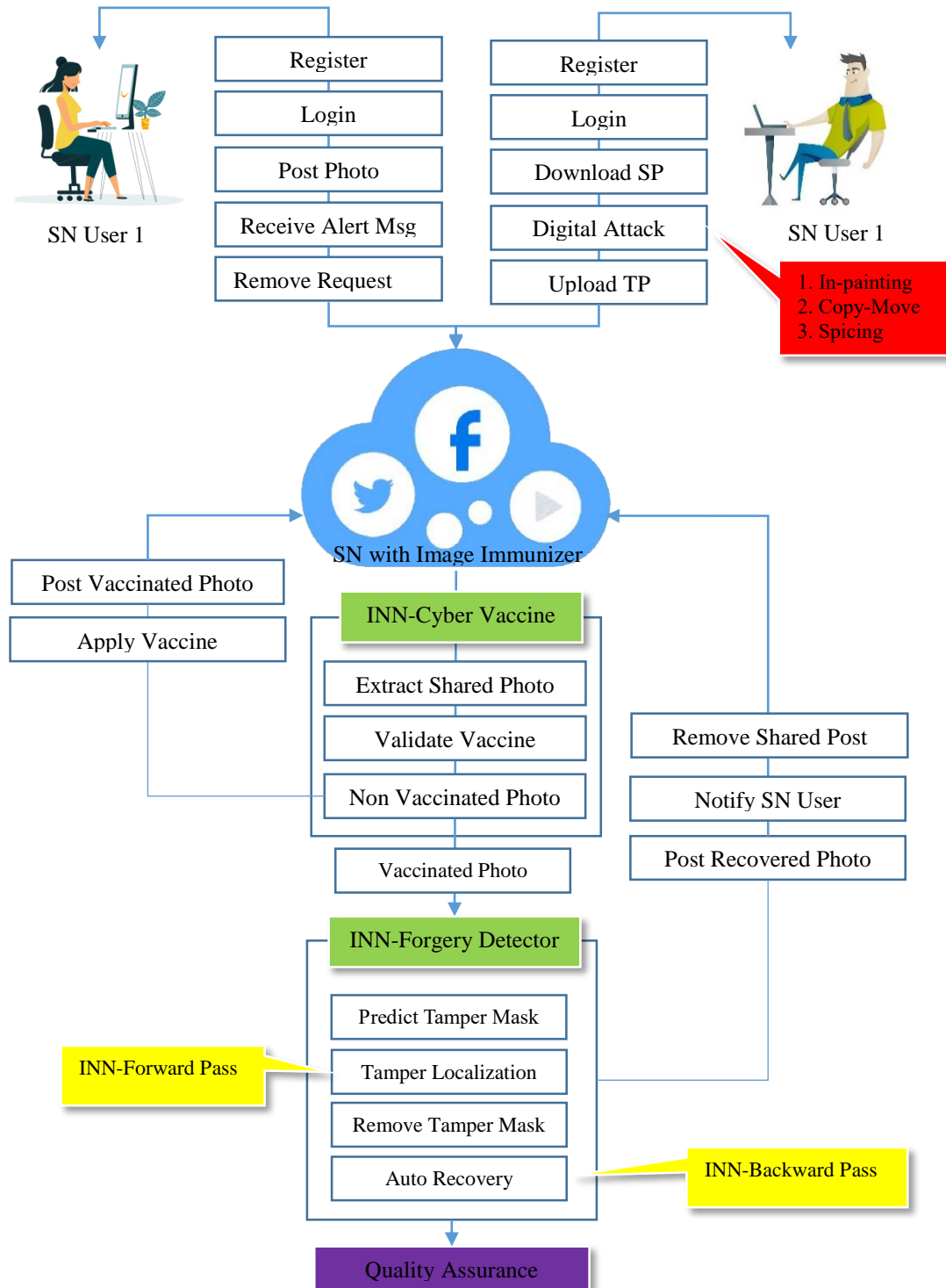


Fig 5.1. SYSTEM ARCHITECTURE

## 5.2 USE CASE

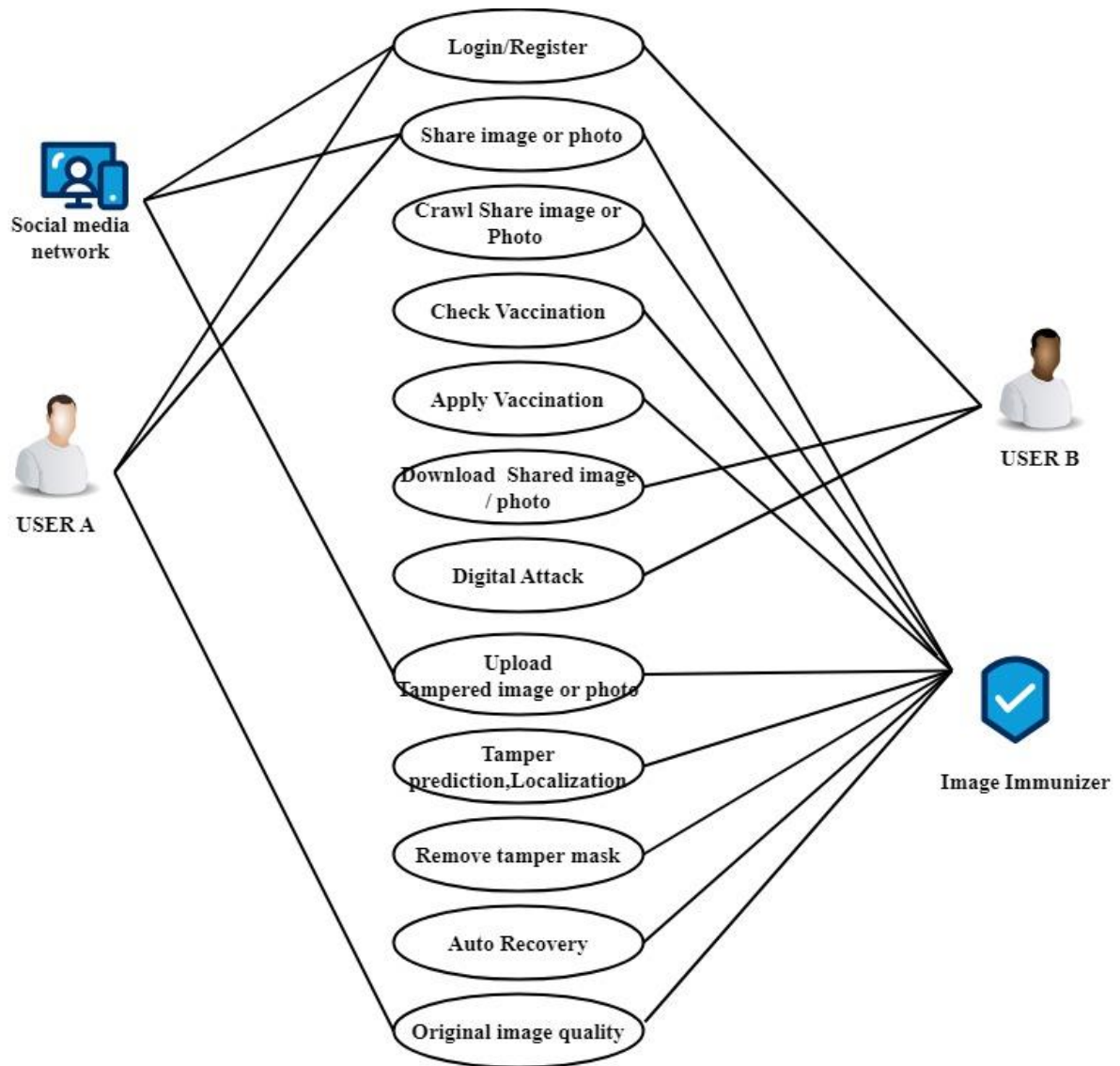


Fig 5.2 USE CASE

### 5.3 CLASS DIAGRAM

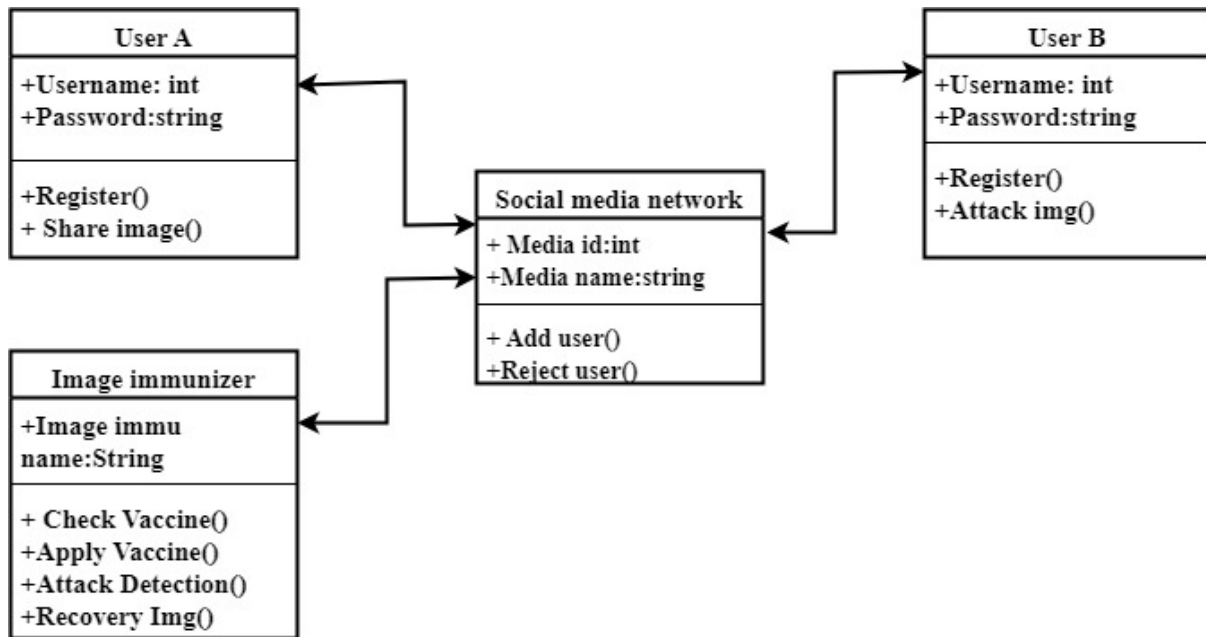


Fig 5.3 CLASS DIAGRAM

## 5.4 ACTIVITY DIAGRAM

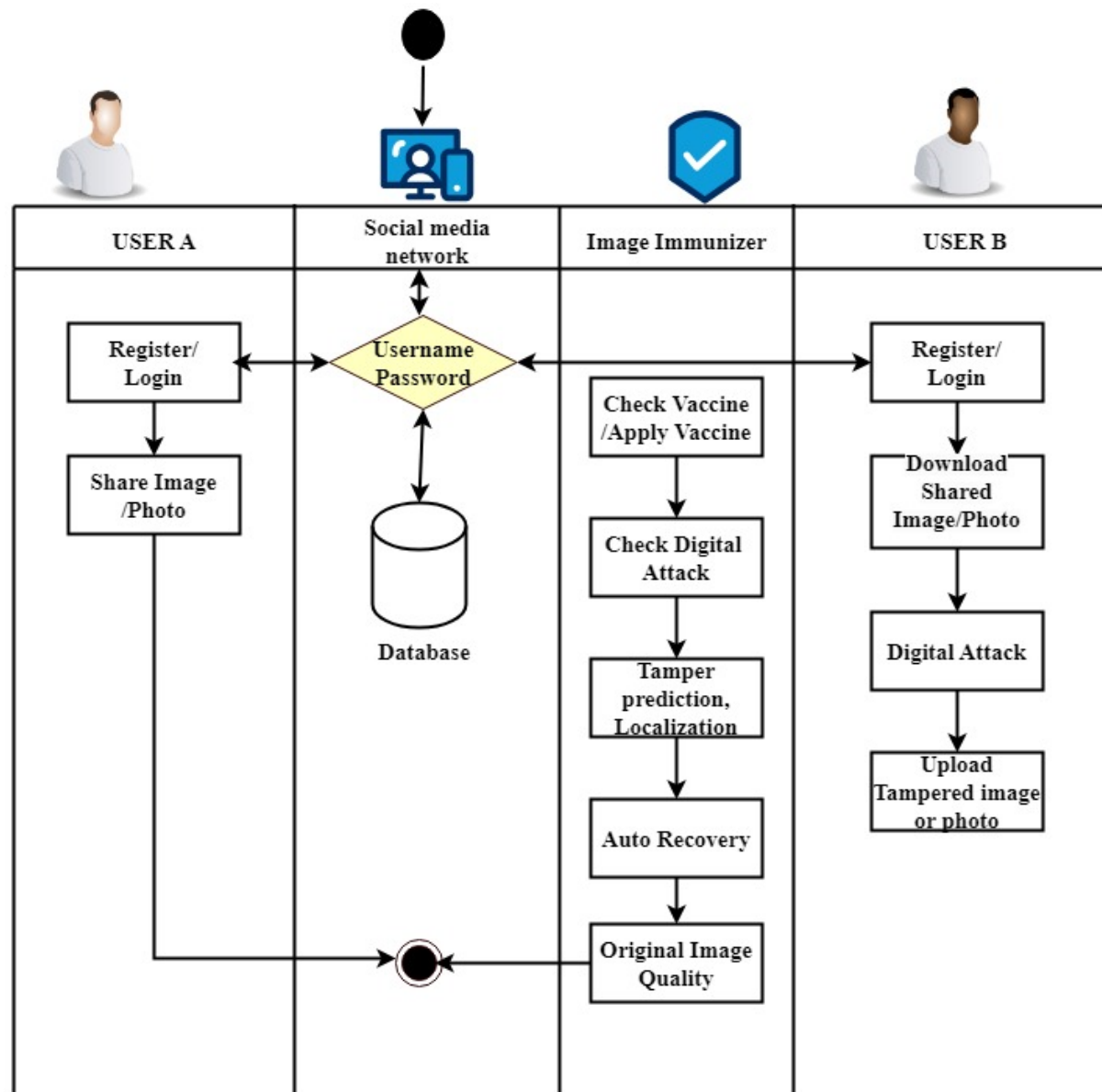


Fig 5.4 ACTIVITY DIAGRAM

## 5.5 SEQUENCE DIAGRAM

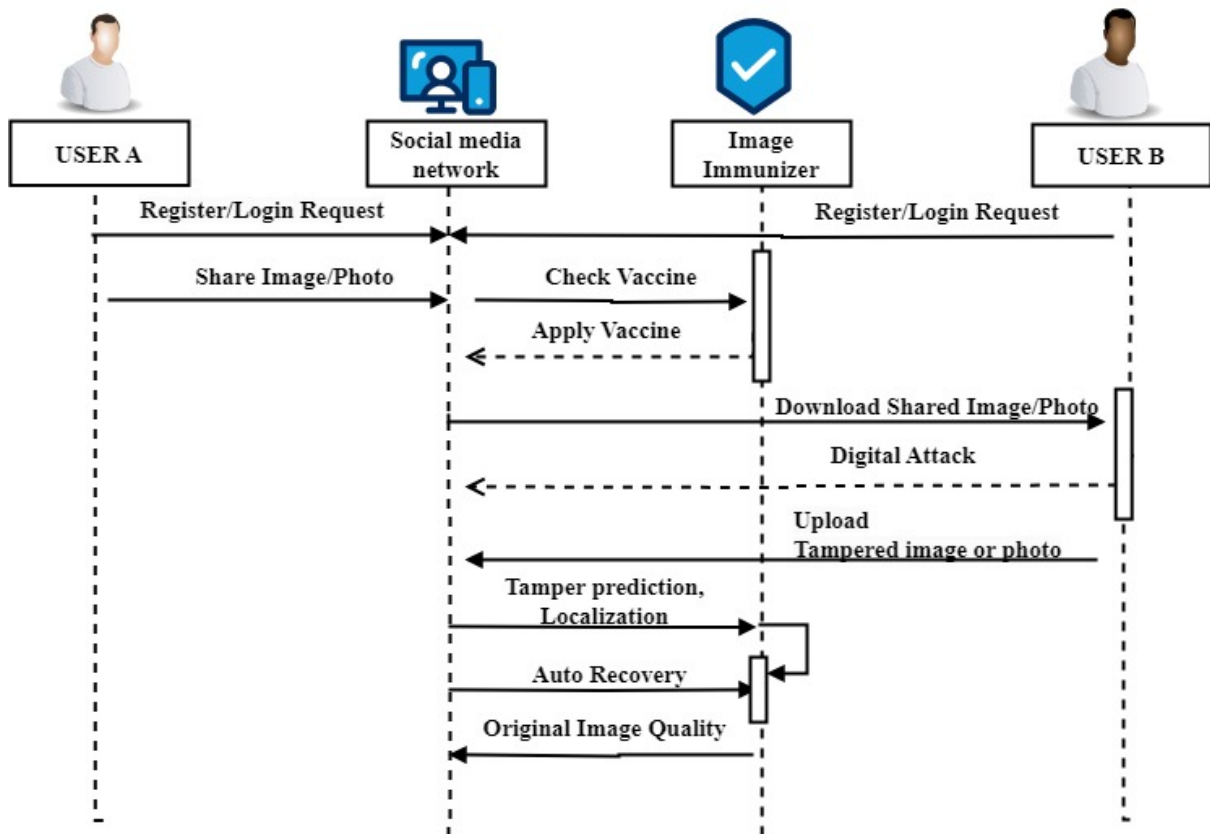


Fig 5.5 SEQUENCE DIAGRAM



## 5.6 COLLABORATION DIAGRAM

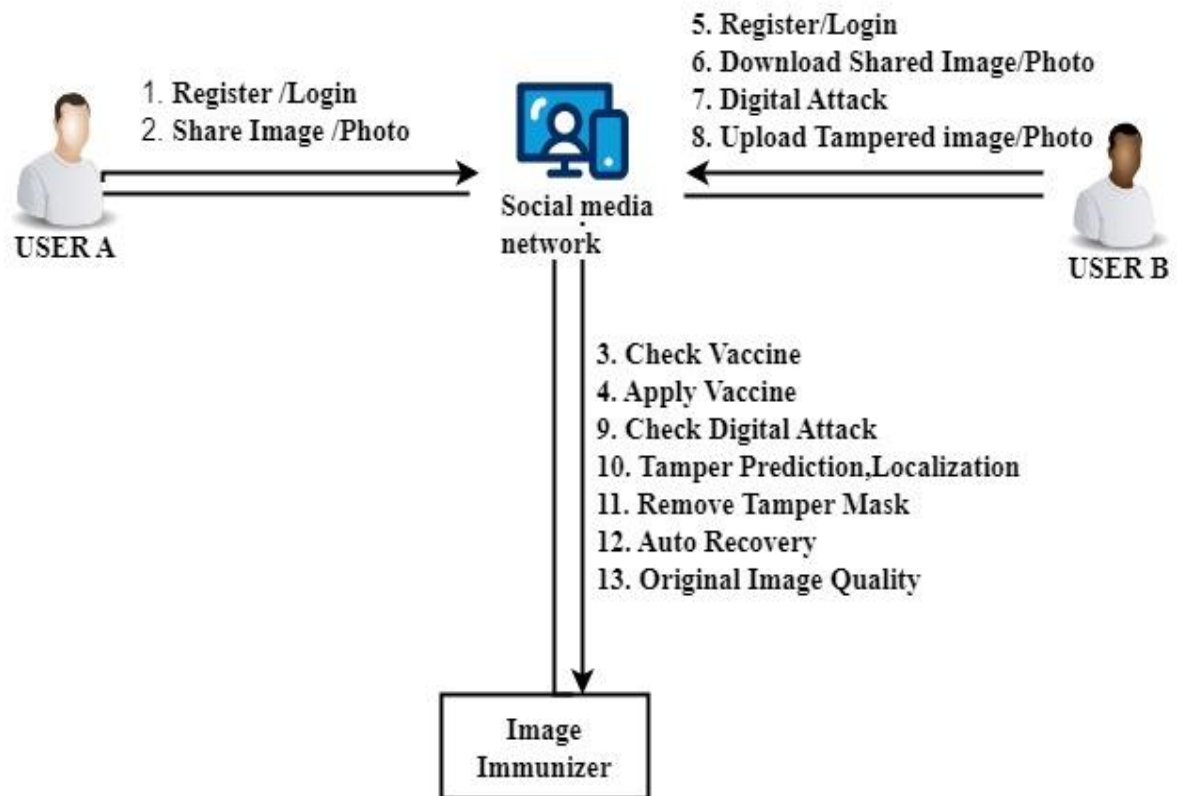


Fig 5.6 COLLABORATION DIAGRAM

## 5.7 DEPLOYMENT DIAGRAM

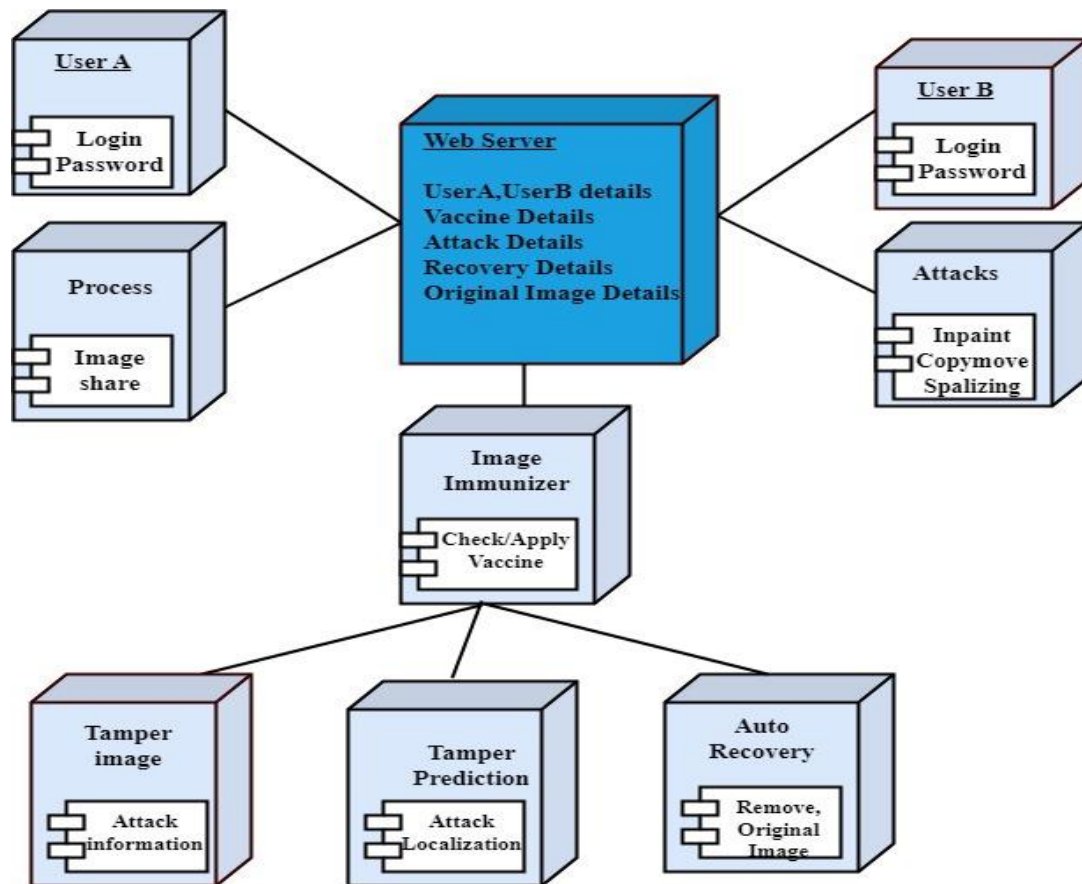


Fig 5.7 DEPLOYMENT DIAGRAM

## 5.8 ER DIAGRAM

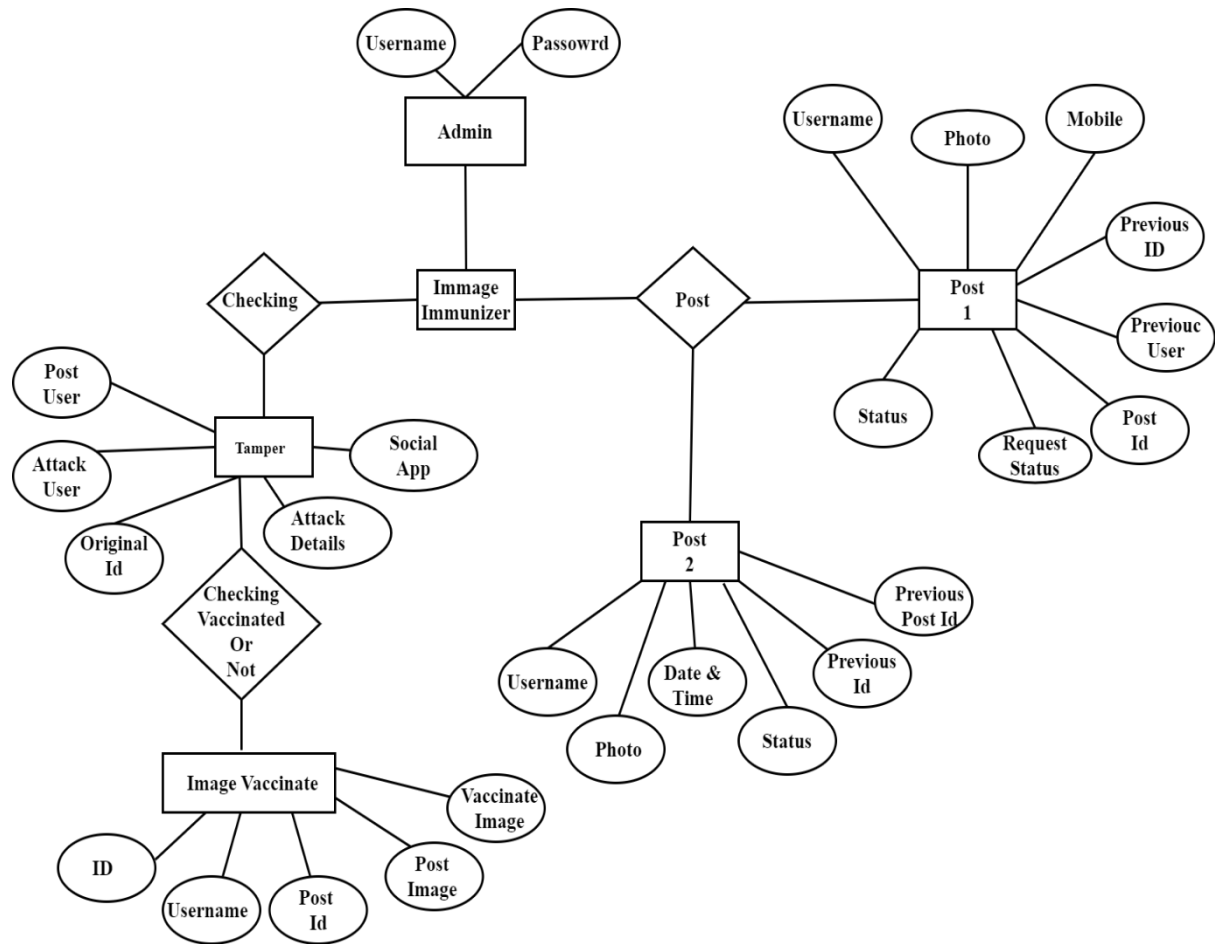


Fig 5.8 ER DIAGRAM

## 5.9 DATA FLOW DIAGRAM

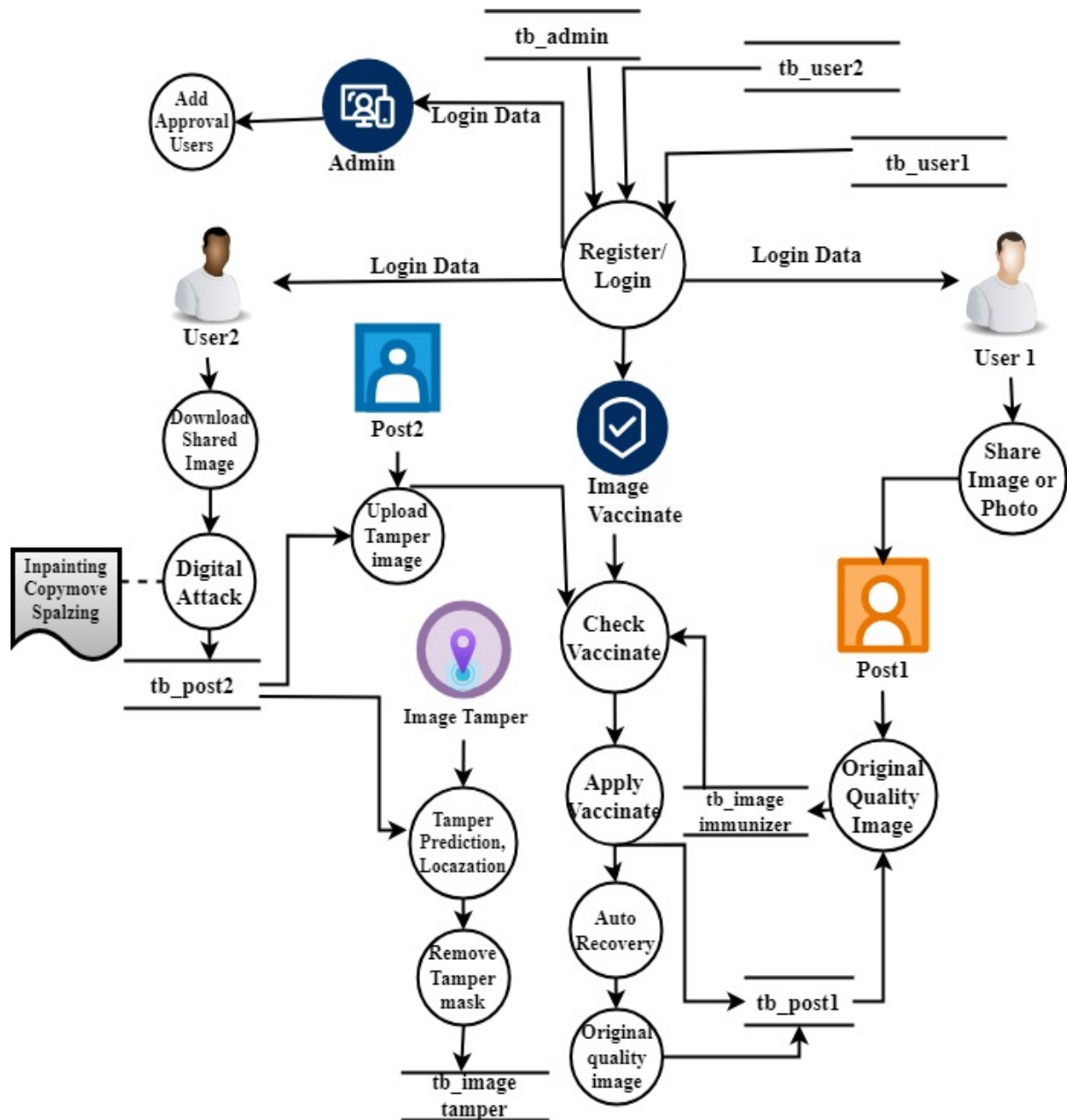


Fig 5.9 DATA FLOW DIAGRAM

## **CHAPTER 6**

### **PROJECT DESCRIPTION**

#### **6.1. DEEP LEARNING**

Deep learning is a method in artificial intelligence (AI) that teaches computers to process data in a way that is inspired by the human brain. Deep learning models can recognize complex patterns in pictures, text, sounds, and other data to produce accurate insights and predictions. Deep learning models are computer files that data scientists have trained to perform tasks using an algorithm or a predefined set of steps. Businesses use deep learning models to analyse data and make predictions in various applications. Computer vision is the computer's ability to extract information and insights from images and videos. Computers can use deep learning techniques to comprehend images in the same way that humans do. Deep learning networks learn by discovering complex structures in the information you feed them. During data processing, artificial neural networks classify the data.

##### **6.1.1. Multi Task Learning**

Multi-task learning (MTL), including learning services, is emerging as a pivotal concept in the rapidly evolving landscape of artificial intelligence. Multi-task learning (MTL) involves training a model to perform multiple tasks concurrently in machine learning. In deep learning, MTL pertains to instructing a neural network to undertake several tasks, achieved by distributing certain network layers and parameters across these tasks.

##### **Key components of multi-task learning**

MTL forms the building blocks of synergy. The key components that enable this synergy are:

- **Hard parameter sharing:** This component involves sharing the hidden layers of a neural network while keeping task-specific output layers. It reduces overfitting by sharing layers across similar jobs.

- **Soft parameter sharing**

Each model has its own set of weights and biases, and the spacing of these parameters in the model is regulated so that the parameters are homogeneous and representative of all applications.

- **Task clustering**

MTL uses task clustering to group tasks. This guarantees that AI models learn from tasks with similar characteristics, resulting in improved knowledge transfer.

- **Shared layers**

AI systems with shared layers enable models to learn shared representations across tasks. These shared layers promote learning synergy and eliminate redundancy.

- **Loss functions**

MTL models can assign varied levels of importance to different activities thanks to tailored loss functions for each activity. This adaptability helps with performance enhancement in tasks of varying complexity.

- **Feature extraction**

MTL uses feature extraction techniques to help AI models find task-specific and shared elements in data. This encourages efficient knowledge transfer.

## **6.2. Invertible Neural Network**

Invertible neural network (INN) is a promising tool for inverse design optimization. While generating forward predictions from given inputs to the system response, INN enables the inverse process without much extra cost. The inverse process of INN predicts the possible input parameters for the specified system response qualitatively. For the purpose of design space exploration and reasoning for critical engineering systems, accurate predictions from the inverse process are required. Moreover, INN predictions lack effective uncertainty quantification for regression tasks, which increases the challenges of decision making. A new loss function is formulated to guide the training process with

enhancement in the inverse process accuracy. INN is a bidirectional mapping network based on affine coupling blocks, considered as an excellent approach for solving inverse problems. The forward process of an INN predicts the output from the input while the inverse process derives the distribution of the input parameters. The key information of its inverse process is captured by the latent variables. INN shines not only in the field of computer vision, but also has distinctive achievements in the field of inverse design.

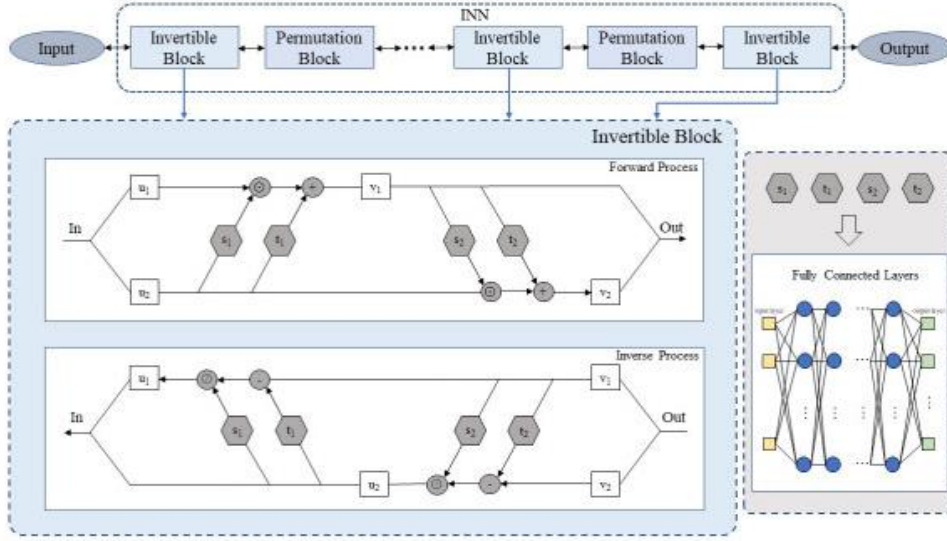


Fig 6.2 Architecture of INN

The forward process of the standard INN only gives the prediction result without presenting uncertainty. Moreover, its inverse process is usually acquired in a qualitative manner and not accurate enough for design purposes. To address this issue, we propose the P-INN with integrated epistemic uncertainty and aleatoric uncertainty. In this way, the outcome of the forward process carries uncertainty, while the posterior distribution of the inverse process incorporates epistemic uncertainty.

### 6.3 DATASET DETAILS

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. The project has been instrumental in

advancing computer vision and deep learning research. The data is available for free to researchers for non-commercial use. This dataset contains multiple images from different classes for Image Classification

## 6.4 PERFORMANCE ANALYSIS

The Peak Signal-to-Noise Ratio (PSNR) is a widely used metric in image processing and quality assessment. It quantifies the difference between the original image and a processed or compressed version, providing a numerical measure of image quality. PSNR is particularly valuable in evaluating the effectiveness of image immunization and recovery processes in the context of the Image Immunizer Middleware for Online Social Networks (OSN).

**Formula:**  $PSNR = 10 \log_{10}(R^2/MSE)$

Where

- R is the maximum pixel value of the image (e.g., 255 for an 8-bit grayscale image).
- Mean Squared Error (MSE) is the average squared difference between the original and processed images.

### Interpretation

- A higher PSNR value indicates a smaller difference between the original and processed images, implying better image quality.
- PSNR is measured in decibels (dB).

### Application in Image Immunizer Middleware

- **Quality Assessment:** PSNR is utilized to assess the quality of immunized and recovered images compared to the original images shared on OSN platforms.
- **Benchmarking:** PSNR serves as a benchmark to evaluate the effectiveness of the immunization and recovery processes. Higher PSNR values signify better preservation of image quality.



## Considerations

- PSNR is sensitive to small pixel value differences. It may not fully capture perceptual differences visible to the human eye.
- The metric assumes a linear relationship between pixel intensity and perceived image quality.

## Implementation

- PSNR can be calculated during the testing phase of the Image Immunizer Middleware by comparing the original images with their immunized and recovered counterparts.
- Continuous monitoring of PSNR values provides insights into the ongoing performance of the system in maintaining image quality.

## 6.5 PERFORMANCE MEASURES

Performance measures for the Image Immunizer Middleware for Online Social Networks (OSN) can be categorized into different aspects, including image quality, system efficiency, security, and user interaction. Here are key performance measures to consider:

### Image Quality:

**Peak Signal-to-Noise Ratio (PSNR):** Measure the quality of immunized and recovered images compared to the originals. Higher PSNR values indicate better image quality.

**Structural Similarity Index (SSI):** Assess the similarity between the original and immunized/recovered images. A value of 1 indicates perfect similarity

**System Efficiency:Execution Time:** Measure the time taken for the entire immunization and recovery process. Evaluate how quickly the system processes image sharing and potential attacks.

**Resource Utilization:** Monitor CPU and memory usage during immunization and recovery processes. Ensure efficient operation without excessive resource consumption.

## **Security:**

**Detection Accuracy:** Measure the accuracy of tamper detection in identifying and localizing manipulated areas within images. Assess the system's ability to detect various types of attacks.

**False Positive/Negative Rates:** Analyze instances of false positives and false negatives in tamper detection. Strive to minimize both to enhance system reliability.

## **Scalability:**

**System Response Time under Load:** Evaluate how the system response time is affected when multiple users simultaneously share images. Ensure responsiveness under varying loads.

**Capacity Planning:** Plan for future scalability by assessing how the system handles increased user activity and data volume.

## **Adversarial Simulation:**

**Adversarial Simulation Benchmark:** Evaluate the system's performance under simulated adversarial attacks. Measure the accuracy of detecting and recovering from different types of attacks.

### **1. Integration:**

- **Integration Benchmark:** Evaluate the seamless integration of the middleware with various OSN architectures. Test compatibility with different social media platforms.

### **2. Large-Scale Testing:**

- **Large-Scale Image Set Benchmark:** Test the system's scalability by introducing a large dataset of images. Assess whether performance degrades with an increased volume of images

## **6.6 INPUT & OUTPUT**

### **6.6.1. INPUT DESIGN**

The input design for the social networking web app encompasses the various forms and interfaces through which users interact with the system to provide, manipulate, and manage data. Given the nature of the application, which involves user registration, content sharing, communication, and security measures, the input design should be secure. Here's a detailed breakdown:

#### **1. User Registration**

##### **Input Fields:**

1. Username: Text input field
2. Email Address: Text input field
3. Password: Password input field
4. Confirm Password: Password input field

**Submit Button:** "Register" button to submit the registration details.

#### **2. User Login**

##### **Input Fields:**

1. Username/Email Address: Text input field
2. Password: Password input field
3. Two-Factor Authentication (if enabled): Text input field for verification code

**Submit Button:** "Login" button to submit the login credentials.

#### **3. Image Sharing**

**Upload Interface:** Drag-and-drop area or "Upload" button to select images.

**Submit Button:** "Share" button to upload and share the image.

**4. Downloading Shared Images:** Allow users to download shared images

**5. Sharing Tampered Images:** Option to share tampered images to other social networks.

## **6.6.2. OUTPUT DESIGN**

### **1. User Registration Confirmation:**

- Output: Confirmation message upon successful registration.
- Details: Display a success message with instructions on next steps.

### **2. Login Status:**

- Output: Login success/error messages.
- Details: Inform users if login was successful or if there were errors (e.g., incorrect credentials).

### **3. User Profile Display:**

- Output: Personalized user profiles.
- Details: Show user's profile details, including profile picture, bio, and recent activity.

### **4. Friend Request Notifications:**

- Output: Notifications for pending friend requests.
- Details: Display notifications for new friend requests with options to accept or reject.

### **5. Media Sharing Feedback:**

- Output: Confirmation messages for shared media.
- Details: Inform users when their media has successfully shared.

### **6. Downloaded Media:**

- Output: Display downloaded media content.
- Details: Show images/photos downloaded by the user, with options for viewing or deleting.

### **7. Notification Alerts:**

- Output: Real-time notifications for user activity.
- Details: Notify users of new friend requests, shared content, or interactions with their posts

.

## **8. Digital Attack Results:**

- Output: Visual feedback on applied digital attacks.
- Details: Display the modified image after applying digital attacks, with options for further actions.

## **9. Vaccine Validation Confirmation:**

- Output: Confirmation message upon successful validation.
- Details: Inform users when an image has been successfully validated as vaccinated, ensuring its authenticity.

## **10. Tamper Detection Results:**

- Output: Visualization of tampered areas.
- Details: Highlight areas of the image that have been tampered with, indicating the type of attack detected.

## **11. Immunized Image Display:**

- Output: Display of immunized images.
- Details: Show the immunized version of the original image.

## **12. Notification Alerts:**

- Output: Real-time notifications for image sharing activities.
- Details: Notify users when a vaccinated image is shared, with or without detected attacks, prompting appropriate actions.

## **13. Forward Pass Results:**

- Output: Results of the forward pass of the INN.
- Details: Display the output of the forward pass, including tamper mask and attack predictions, for further analysis.

## **14. Backward Pass Results:**

- Output: Results of the backward pass for image self-recovery.
- Details: Show the recovered image and its edge map after the backward pass, ensuring seamless restoration.

## **CHAPTER 7**

### **IMPLEMENTATION**

#### **7.1 Source code**

##### **Packages**

```
import os
import base64
import io
import math
from flask import Flask, flash, render_template, Response, redirect, request,
session, abort, url_for
import mysql.connector
import hashlib
import datetime
from random import randint
import cv2
from math import log10, sqrt
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import time
import imagehash
import shutil
from skimage.metrics import structural_similarity
import PIL.Image
from PIL import Image
```

##### **Database Connection**

```
mydb = mysql.connector.connect(
host="localhost",
```

```
user="root",
password="",
charset="utf8",
database="image_immunize)
```

## Login

```
def index():
    msg=""
    ff=open("det.txt","w")
    ff.write("1")
    ff.close()
    ff1=open("photo.txt","w")
    ff1.write("1")
    ff1.close()
    #s="welcome"
    #v=s[2:5]
    #print(v)
    if request.method=='POST':
        uname=request.form['uname']
        pwd=request.form['pass']
        cursor = mydb.cursor()
        cursor.execute('SELECT * FROM im_user WHERE uname = %s AND pass = %s', (uname, pwd))
        account = cursor.fetchone()
        if account:
            session['username'] = uname
            return redirect(url_for('userhome'))
        else:
            msg = 'Incorrect username/password!'
```

## User Registration

```
def signup ():
    msg=""
    mycursor = mydb.cursor()
    mycursor.execute("SELECT max(id)+1 FROM im_user")
    maxid = mycursor.fetchone()[0]
    if maxid is None:
        maxid=1
    if request.method=='POST':
        name=request.form['name']
        gender=request.form['gender']
        dob=request.form['dob']
        aadhar=request.form['aadhar']
        mobile=request.form['mobile']
        email=request.form['email']
        uname=request.form['uname']
        pass1=request.form['pass']
        mycursor.execute('SELECT count(*) FROM im_user WHERE uname = %s ||
        aadhar=%s', (uname,aadhar))
        cnt = mycursor.fetchone()[0]
        if cnt==0:
            now = datetime.now()
            rdate=now.strftime("%d-%m-%Y")
            adr=str(aadhar)
            rn=randint(50,90)
            v1=name [0:2]
            v2=str(rn)
            v3=adr[0:3]
```



```

bkey=v1+str(maxid)+v2+v3
sql = "INSERT INTO im_user(id,
name,gender,dob,mobile,email,aadhar,uname,pass,create_date,photo) VALUES
(%s,%s, %s, %s, %s, %s,%s,%s,%s,%s,%s)"
val = (maxid, name, gender, dob, mobile, email, aadhar, uname, pass1, rdate,)
mycursor.execute(sql, val)
mydb.commit()
print(mycursor.rowcount, "Registered Success")
else:
msg='fail'

```

### **User Post**

```

def userhome():
msg=""
fid=""
act=request.args.get("act")
if 'username' in session:
uname = session['username']
mycursor = mydb.cursor()
mycursor.execute('SELECT * FROM im_user where uname=%s',(uname,))
udata = mycursor.fetchone()
if request.method == 'POST':
detail= request.form['detail']
if 'file' not in request.files:
flash('No file Part')
return redirect(request.url)
file= request.files['file']
mycursor.execute("SELECT max(id)+1 FROM im_post")
maxid = mycursor.fetchone()[0]

```

```

if maxid is None:
    maxid=1
if file.filename == "":
    flash('No Select file')
#return redirect(request.url)
if file:
    fname1 = file.filename
    fname = secure_filename(fname1)
    file_name="P"+str(maxid)+fname
    file.save(os.path.join("static/upload/", file_name))
    ff=open("static/ufile.txt","w")
    ff.write(file_name)
    ff.close()
    res=vaccinate(uname,file_name)

```

### **Check Vaccination**

```

def vaccinate(uname,file_name):
    s=""
    user=""
    attack=""
    uu=""
    pid=""
    stu='0'
    fn=""
    social_app=0
    filepath="static/upload/"+file_name
    img1 = Image.open(filepath)
    st="1"
    # get width and height

```

```

width1 = img1.width
height1 = img1.height
mycursor = mydb.cursor()
mycursor.execute("SELECT count(*) FROM im_post where photo!='" &&
status=0 order by id")
cnt = mycursor.fetchone()[0]
if cnt>0:
mycursor.execute("SELECT * FROM im_post where photo!='" && status=0
order by id")
dat = mycursor.fetchall()
x=0
for dd in dat:
fn=dd[3]
img=Image.open("static/upload/"+fn)
width = img.width
height = img.height
if width==width1 and height==height1:
cmp=compare(fn,file_name)
print(cmp)
cmp1=float(cmp[0])
uu=dd[1]
if cmp1>=99.8:
x+=1
social_app=1
pid=str(dd[0])
if uname==dd[1]:
user="1"
else:

```

```

user="2"
stu='1'
s="2"
break
elif cmp1>=75:
social_app=1
pid=str(dd[0])
print(fn)
stu='2'
ff=open("static/fname.txt","w")
ff.write(fn)
ff.close()
x+=1
s="3"
st="2"
stu='2'
a1=attack1(fn,file_name)
a11=attack11(fn,file_name)
a2=attack2(fn,file_name)
attack_st2=a2[2]
a3=attack3(fn,file_name)
if a3[2]>0 and a2[2]==0 and a11[2]==0:
print("splice")
attack="splice"
elif a1[2]>a11[2] and a2[2]==0:
attack="inpaint"
elif a2[2]>a1[2] and a2[2]>a11[2]:
if a2[2]>0:

```

```

print("copy")
attack="copy"
elif a2[2]==0:
if a11[2]>=0:
if a1[2]>a11[2]:
attack="inpaint"
break
mycursor.execute("SELECT * FROM im_post1 where photo!=" && status=0
order by id")
dat1 = mycursor.fetchall()
for dd1 in dat1:
fn=dd1[3]
img=Image.open("static/upload/"+fn)
width = img.width
height = img.height
if width==width1 and height==height1:
cmp=compare(fn,file_name)
print(cmp)
cmp1=float(cmp[0])
uu=dd1[1]
if cmp1>=99.8:
x+=1
social_app=2
pid=str(dd1[0])
if uname==dd1[1]:
user="1"
else:
user="2"

```

```

#uu=dd1[1]
stu='1'
s="2"
break
elif cmp1>=75:
social_app=2
pid=str(dd1[0])
ff=open("static/fname.txt","w")
ff.write(fn)
ff.close()
x+=1
s="3"
st="2"
stu='2'
a1=attack1(fn,file_name)
a11=attack11(fn,file_name)
a2=attack2(fn,file_name)
attack_st2=a2[2]
a3=attack3(fn,file_name)
if a3[2]>0 and a2[2]==0 and a11[2]==0:
print("splice")
attack="splice"
elif a1[2]>a11[2] and a2[2]==0:
print("inpaint")
attack="inpaint"
elif a2[2]>a1[2] and a2[2]>a11[2]:
if a2[2]>0:
print("copy")

```

```

attack="copy"
elif a2[2]==0:
if a1[2]>=0:
if a1[2]>a1[2]:
print("inpaint")
attack="inpaint"
break
if x==0:
s="1"
fn2="D"+file_name
fn3="H"+file_name
from PIL.ImageFilter import (
BLUR, CONTOUR, DETAIL, EDGE_ENHANCE,
EDGE_ENHANCE_MORE,
EMBOSS, FIND_EDGES, SMOOTH, SMOOTH_MORE, SHARPEN
#Create image object
img = Image.open("static/upload/"+file_name)
#Applying the blur filter
img1 = img.filter(CONTOUR)
img1.save('static/test/'+fn2)
#img1.show()
im = Image.open("static/test/D"+file_name)
cmyk = gcr(im, 0)
dots = halftone(im, cmyk, 10, 1)
#im.show()
new = Image.merge('CMYK', dots)
#new.show()
new.save('static/test/'+fn3)

```

```

else:
s="1"
fn2="D"+file_name
fn3="H"+file_name
from PIL.ImageFilter import (
BLUR, CONTOUR, DETAIL, EDGE_ENHANCE,
EDGE_ENHANCE_MORE,
EMBOSS, FIND_EDGES, SMOOTH, SMOOTH_MORE, SHARPEN
#Create image object
img = Image.open("static/upload/"+file_name)
#Applying the blur filter
img1 = img.filter(CONTOUR)
img1.save('static/test/'+fn2)
#img1.show()
im = Image.open("static/test/D"+file_name)
cmyk = gcr(im, 0)
dots = halftone(im, cmyk, 10, 1)
#im.show()
new = Image.merge('CMYK', dots)
#new.show()
new.save('static/test/'+fn3)
value=[s,user,uu,fn,pid,stu,attack,social_app]

return value

```

### **Immunizer**

```

def gcr(im, percentage):
    """basic "Gray Component Replacement" function. Returns a CMYK image with
percentage gray component removed from the CMY channels and put in the
K channel, ie. for percentage=100, (41, 100, 255, 0) >> (0, 59, 214, 41)"""

```



```

cmyk_im = im.convert('CMYK')
if not percentage:
    return cmyk_im
cmyk_im = cmyk_im.split()
cmyk = []
for i in range(4):
    cmyk.append(cmyk_im[i].load())
for x in range(im.size[0]):
    for y in range(im.size[1]):
        gray = min(cmyk[0][x,y], cmyk[1][x,y], cmyk[2][x,y]) * percentage / 100
    for i in range (3):
        cmyk[i][x,y] = cmyk[i][x,y] - gray
    cmyk[3][x,y] = gray
return Image.merge('CMYK', cmyk_im)

def halftone(im, cmyk, sample, scale):
    """Returns list of half-tone images for cmyk image. sample (pixels),
    determines the sample box size from the original image. The maximum
    output dot diameter is given by sample * scale (which is also the number
    of possible dot sizes). So sample=1 will presevere the original image
    resolution, but scale must be >1 to allow variation in dot size."""
    cmyk = cmyk.split()
    dots = []
    angle = 0
    for channel in cmyk:
        channel = channel.rotate(angle, expand=1)
        size = channel.size[0]*scale, channel.size[1]*scale
        half_tone = Image.new('L', size)
        draw = ImageDraw.Draw(half_tone)

```

```

for x in range(0, channel.size[0], sample):
    for y in range(0, channel.size[1], sample):
        box = channel.crop((x, y, x + sample, y + sample))
        stat = ImageStat.Stat(box)
        diameter = (stat.mean[0] / 255)**0.5
        edge = 0.5*(1-diameter)
        x_pos, y_pos = (x+edge)*scale, (y+edge)*scale
        box_edge = sample*diameter*scale
        draw.ellipse((x_pos, y_pos, x_pos + box_edge, y_pos + box_edge), fill=255)
        half_tone = half_tone.rotate(-angle, expand=1)
        width_half, height_half = half_tone.size
        xx=(width_half-im.size[0]*scale) / 2
        yy=(height_half-im.size[1]*scale) / 2
        half_tone = half_tone.crop((xx, yy, xx + im.size[0]*scale, yy +
        im.size[1]*scale))
        dots.append(half_tone)
        angle += 15
    return dots

def compare(img1,img2):
    # Load images
    before = cv2.imread("static/upload/"+img1)
    after = cv2.imread("static/upload/"+img2)
    # Convert images to grayscale
    before_gray = cv2.cvtColor(before, cv2.COLOR_BGR2GRAY)
    after_gray = cv2.cvtColor(after, cv2.COLOR_BGR2GRAY)
    # Compute SSIM between the two images
    (score, diff) = structural_similarity(before_gray, after_gray, full=True)
    print ("Image Similarity: {:.4f} %".format(score * 100))

```

```

per=format (score * 100)
# The diff image contains the actual image differences between the two images
# and is represented as a floating point data type in the range [0,1]
# so we must convert the array to 8-bit unsigned integers in the range
# [0,255] before we can use it with OpenCV
diff = (diff * 255).astype("uint8")
diff_box = cv2.merge([diff, diff, diff])
# Threshold the difference image, followed by finding contours to
# obtain the regions of the two input images that differ
thresh = cv2.threshold(diff, 0, 255, cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU) [1]
contours = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
contours = contours [0] if len(contours) == 2 else contours [1]
mask = np.zeros(before.shape, dtype='uint8')
filled_after = after.copy()
j=1
for c in contours:
area = cv2.contourArea(c)
if area > 40:
x,y,w,h = cv2.boundingRect(c)
cv2.rectangle(before, (x, y), (x + w, y + h), (36,255,12), 2)
mm=cv2.rectangle(after, (x, y), (x + w, y + h), (36,255,12), 2)
cv2.imwrite("static/test/ggg.jpg", mm)
image = cv2.imread("static/test/ggg.jpg")
cropped = image [y:y+h, x:x+w]
gg="f"+str(j)+".jpg"
cv2.imwrite("static/test/"+gg, cropped)

```

```

cv2.rectangle(diff_box, (x, y), (x + w, y + h), (36,255,12), 2)
cv2.drawContours(mask, [c], 0, (255,255,255), -1)
cv2.drawContours(filled_after, [c], 0, (0,255,0), -1)
j+=1
value=[per,j]
return value
def attack1(img1,img2):
# Load images
before = cv2.imread("static/upload/"+img2)
after = cv2.imread("static/upload/"+img1)
# Convert images to grayscale
before_gray = cv2.cvtColor(before, cv2.COLOR_BGR2GRAY)
after_gray = cv2.cvtColor(after, cv2.COLOR_BGR2GRAY)
# Compute SSIM between the two images
(score, diff) = structural_similarity(before_gray, after_gray, full=True)
print("Image Similarity: {:.4f}%".format(score * 100))
per=format(score * 100)
diff = (diff * 255).astype("uint8")
diff_box = cv2.merge([diff, diff, diff])
# obtain the regions of the two input images that differ
thresh = cv2.threshold(diff, 0, 255, cv2.THRESH_BINARY_INV |
cv2.THRESH_OTSU)[1]
contours = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
contours = contours[0] if len(contours) == 2 else contours[1]
mask = np.zeros(before.shape, dtype='uint8')
filled_after = after.copy()
j=1

```

```

for c in contours:
    area = cv2.contourArea(c)
    if area > 40:
        x,y,w,h = cv2.boundingRect(c)
        cv2.rectangle(before, (x, y), (x + w, y + h), (36,255,12), 2)
        mm=cv2.rectangle(after, (x, y), (x + w, y + h), (36,255,12), 2)
        cv2.imwrite("static/test/ggg.jpg", mm)
        image = cv2.imread("static/test/ggg.jpg")
        cropped = image [y:y+h, x:x+w]
        gg="h"+str(j)+".jpg"
        cv2.imwrite("static/test/"+gg, cropped)
        cv2.rectangle(diff_box, (x, y), (x + w, y + h), (36,255,12), 2)
        cv2.drawContours(mask, [c], 0, (255,255,255), -1)
        cv2.drawContours(filled_after, [c], 0, (0,255,0), -1)
        j+=1
        e=j-1
        n=1
        y=0
        k=0
        while n<=e:
            main_image = cv2.imread('static/upload/'+img1)
            gray_image = cv2.cvtColor(main_image, cv2.COLOR_BGR2GRAY)
            ffn="h"+str(n)+".jpg"
            template = cv2.imread("static/test/"+ffn, 0)
            width, height = template.shape[:: -1] #get the width and height
            match = cv2.matchTemplate(gray_image, template,
            cv2.TM_CCOEFF_NORMED)
            threshold = 0.8

```

```

position = np.where(match >= threshold) #get the location of template in the
image
k=0
cv2.rectangle(main_image, point, (point[0] + width, point[1] + height), (0, 204,
153), 0)
k+=1
if k>1:
y+=k
n+=1
cv2.imshow('before', before)
cv2.imshow('after', after)
cv2.imshow('diff', diff)
cv2.imshow('diff_box', diff_box)
cv2.imshow('mask', mask)
cv2.imshow('filled after', filled_after)
value=[per,j,y]
return valu

```

## **CHAPTER 8**

### **TESTING**

#### **8.1.TESTING OBJECTIVES**

The main objective of the testing is to uncover a host of errors, systematically and with minimum effort and time.

- Testing is the process of executing a program with the intent of finding an errors
- A good test case is one that has a high probability of finding errors, if it exists
- The tests are inadequate to detect present errors
- The software confirms to the quality and reliable standards

#### **8.2 TYPES OF TESTING**

##### **8.2.1 Unit Testing**

In the lines of strategy, all the individual functions and modules were put to the test independently. By following this strategy all the errors in coding were identified and corrected.

##### **8.2.2 Functional Testing**

In functional testing, each function tested by giving the value, determining the output, and verifying the actual output with the expected value.

##### **8.2.3 White Box Testing**

White box testing is known as Clear Box testing, code-based testing, structural testing, extensive testing, and glass box testing, transparent box testing. It is a software testing method in which the internal structure/design/implementation tested known to the tester.

## **CHAPTER 9**

### **CONCLUSION & FUTURE ENHANCEMENT**

#### **9.1. CONCLUSION**

In conclusion, the project Image Immunizer Middleware for Online Social Networks offers a cutting-edge solution to combat the growing threat of digital image attacks. Invertible Neural Network technology and incorporating adversarial simulation, the system provides a formidable defence, securing the authenticity and integrity of images shared on social networking platforms. Through process involving the Cyber Vaccinator Module, the system adeptly pre-processes, vaccinates, and post-processes images, introducing imperceptible perturbations to fortify them against potential tampering. The Forward Pass, employing INN, and the subsequent Backward Pass for image self-recovery collectively contribute to the identification and restoration of tampered areas. This dynamic approach ensures that the recovered image closely aligns with the original, reinforcing the reliability of shared media. Adversarial simulation during training further strengthens the system. This project represents a state-of-the-art solution, combining advanced technologies and thoughtful design to safeguard the digital integrity of shared images in the dynamic realm of online social networks.

#### **9.2 FUTURE ENHANCEMENT**

The Future enhancements for the Image Immunizer Middleware for Online Social Networks using Invertible Neural Network (INN) aim to strengthen its capabilities and adapt to evolving technology. Integrating blockchain technology can enhance transparency in image transactions, ensuring a tamper-evident record. The middleware's expansion to multimodal content analysis, including videos and audio, provides a more comprehensive defence against digital manipulation within OSN. These advancements reflect a commitment to robust security and holistic content integrity.



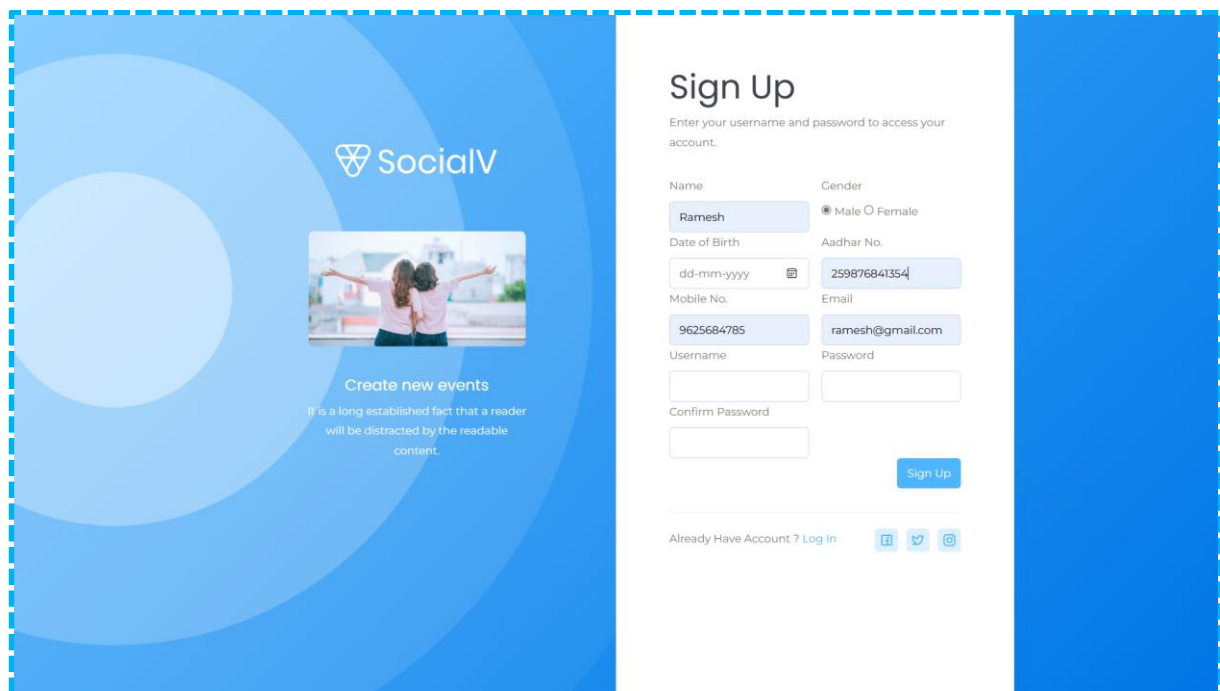
## CHAPTER 10

### EXPERIMENTAL RESULTS

#### 10.1 RESULTS

The implementation of the Image Immunizer Middleware for Online Social Networks (OSN) using the Invertible Neural Network (INN) has yielded promising results in enhancing the security and integrity of shared images on social media platforms. Additionally, features like user registration, profile personalization, media sharing. The End User Interface module provides users with intuitive access to essential functionalities, including registration, login, social connections, image sharing and notifications. The Objective Loss Function, incorporating techniques like Run Length Encoding (RLE) and Peak Signal-to-Noise Ratio (PSNR), further enhances the system's ability to detect and recover from tampering while preserving image quality and minimizing data loss.

#### 10.2 EXPERIMENTAL RESULTS



**Sign Up**

Enter your username and password to access your account.

Name:

Gender: ☒ Male ☐ Female

Date of Birth:

Aadhar No.:

Mobile No.:

Email:

Username:

Password:

Confirm Password:

Already Have Account ? [Log In](#)

Fig 10.2.1 user sign up

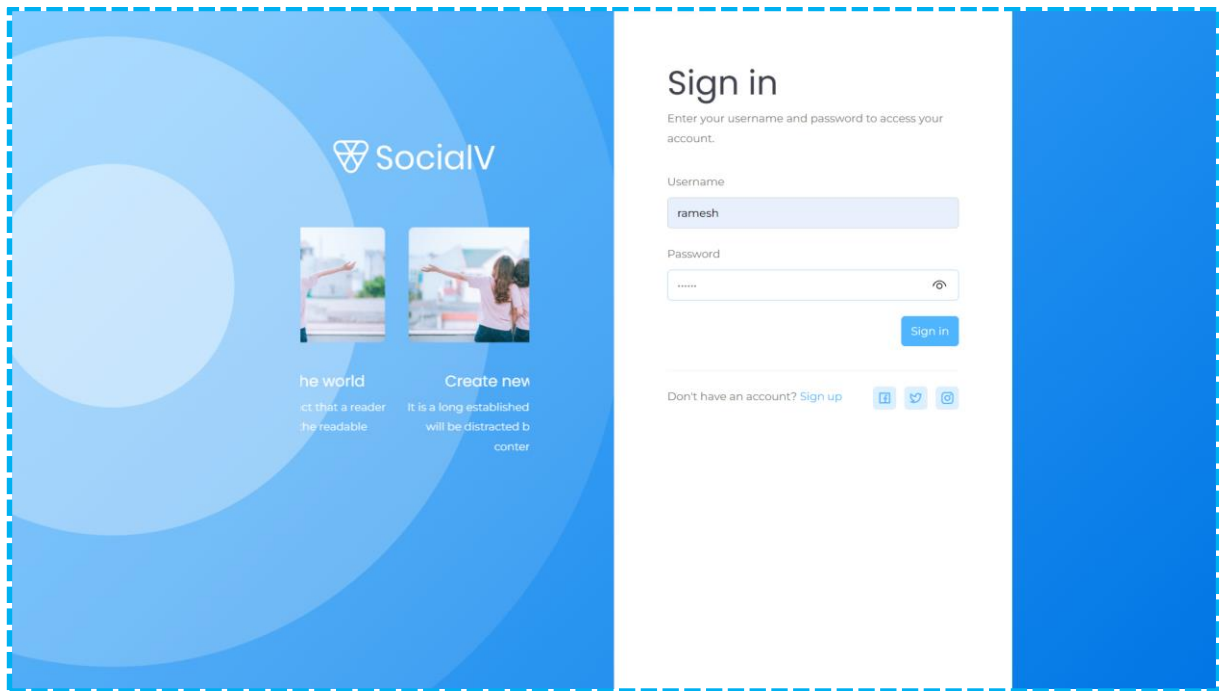


Fig 10.2.2 user sign in

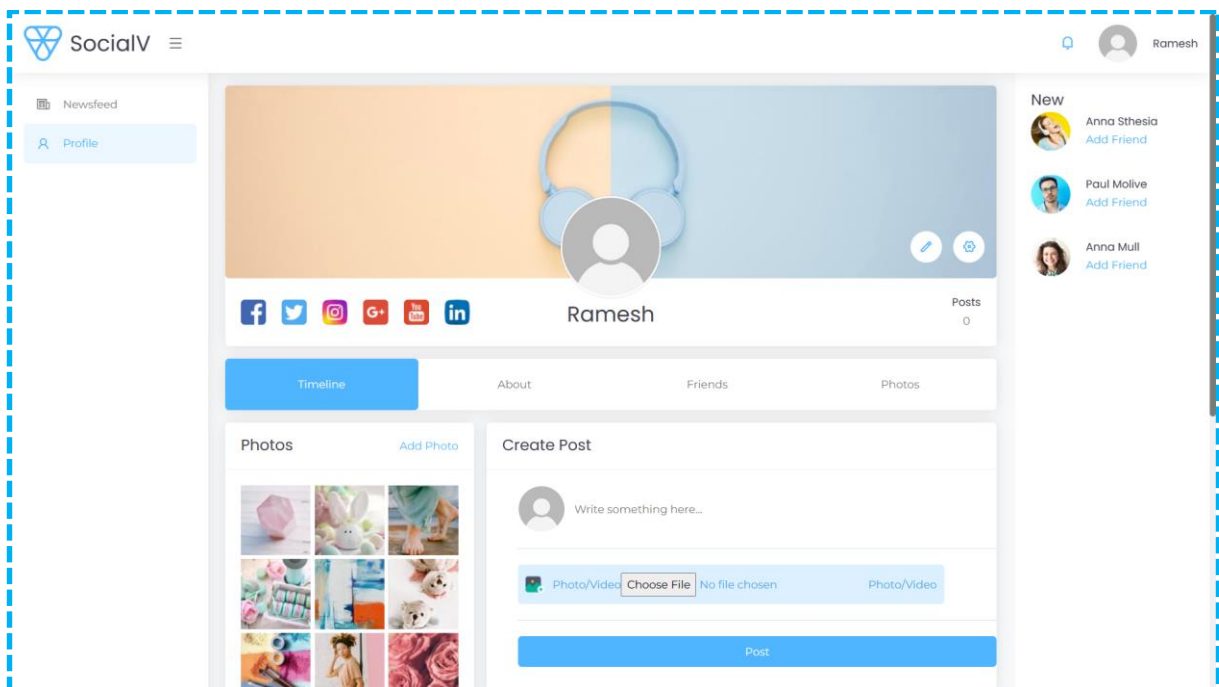


Fig 10.2.3 Home page

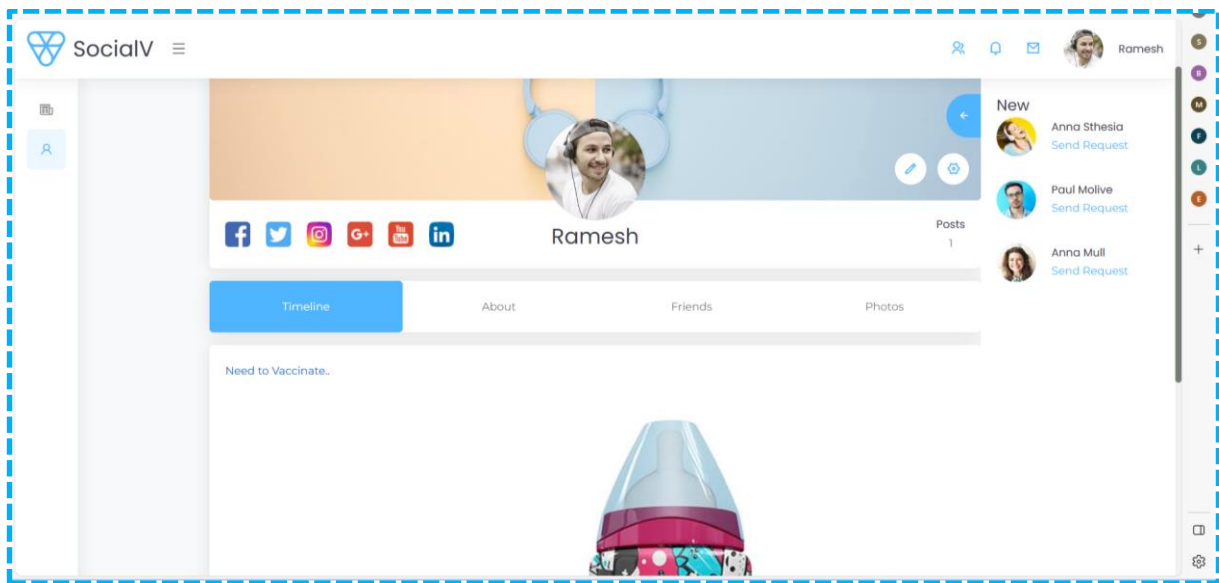


Fig 10.2.3 Need to vaccinate

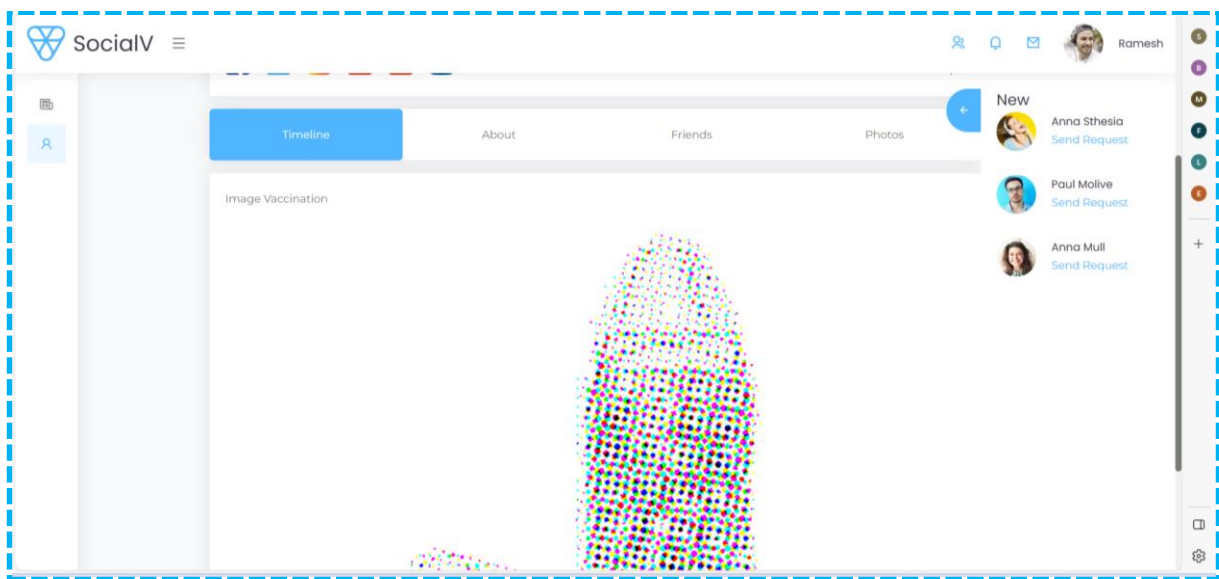


Fig 10.2.4 Image vaccinator

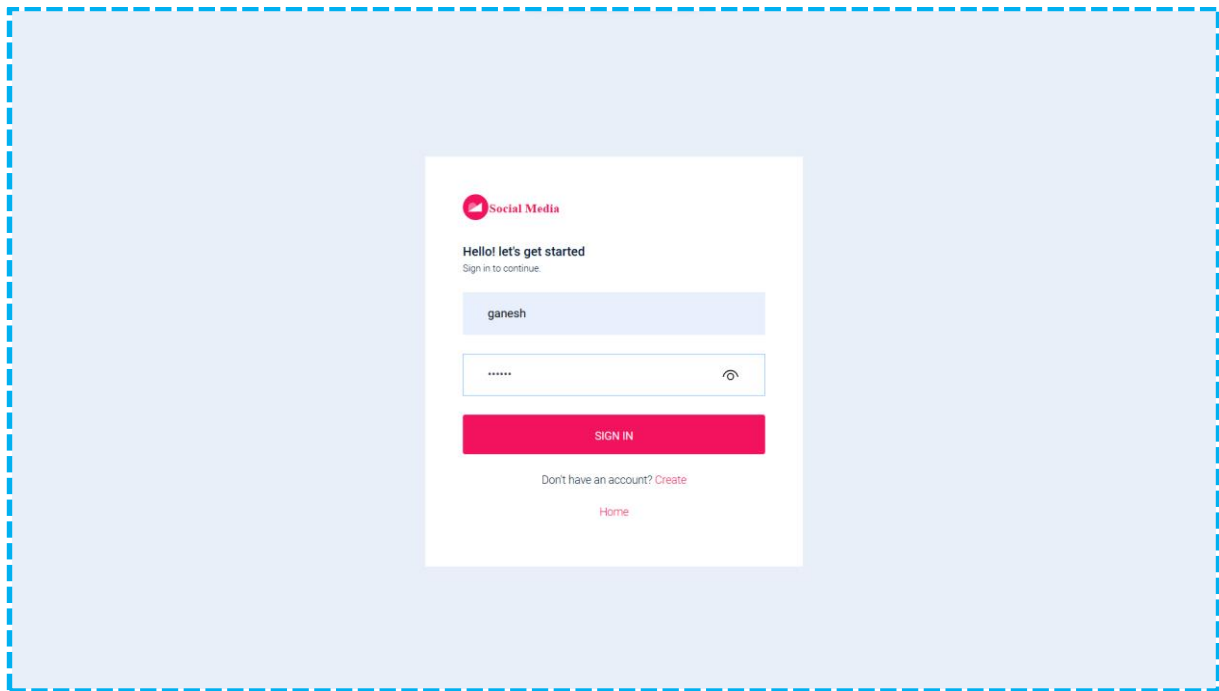


Fig 10.2.5 Social media sign in

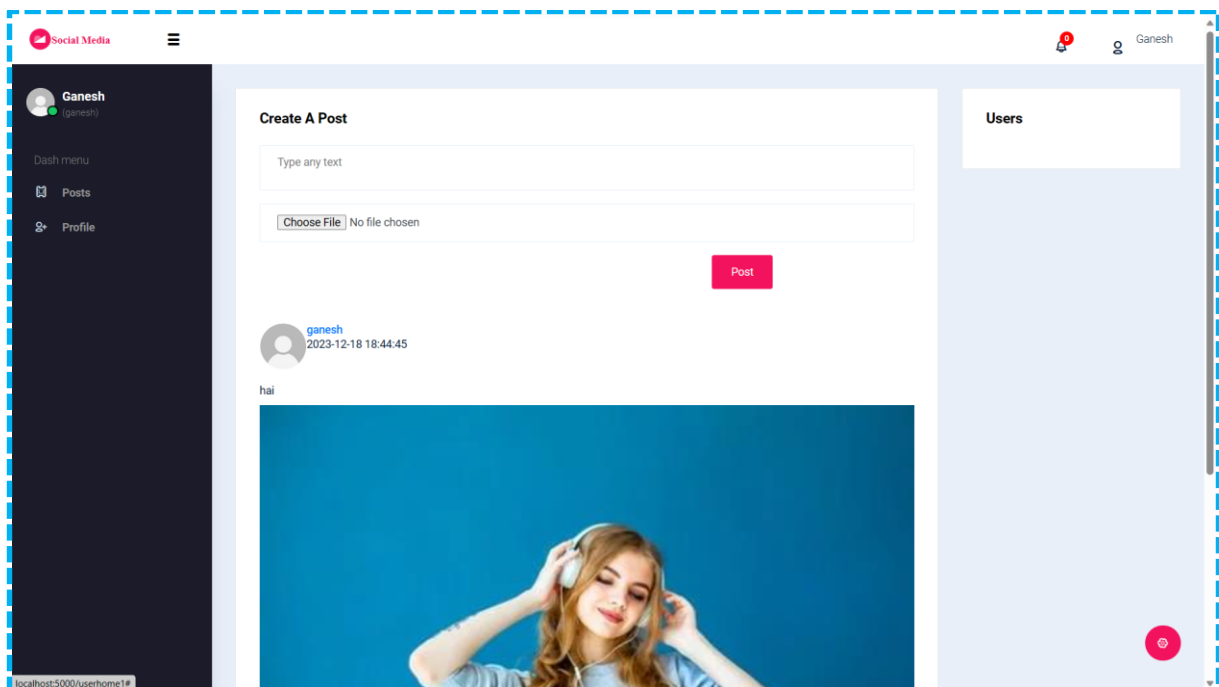


Fig 10.2.6 Dashboard

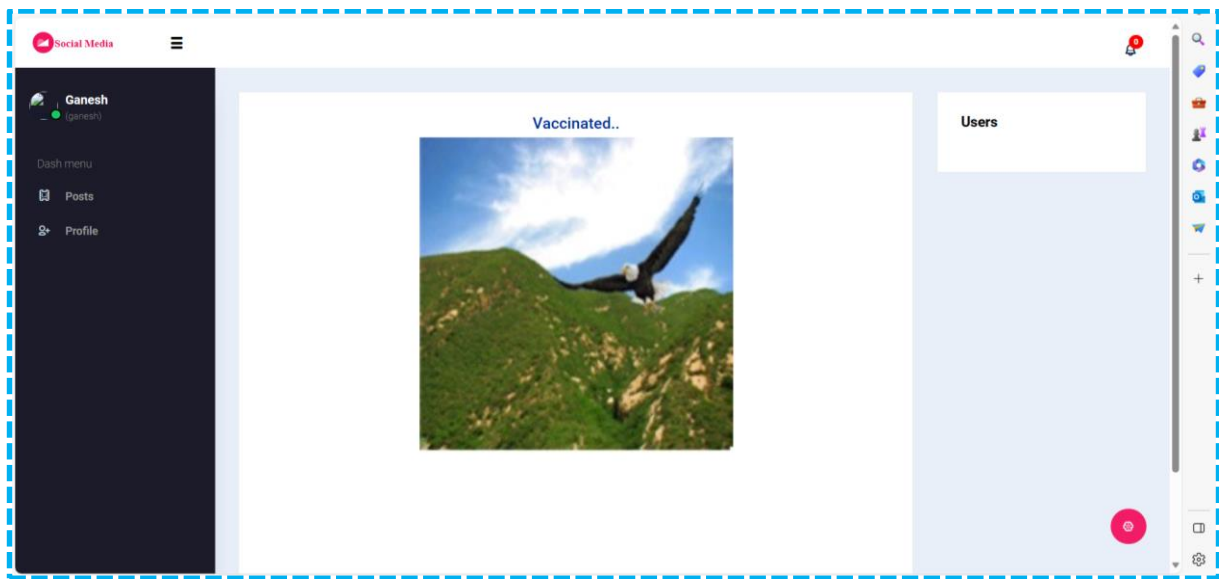


Fig 10.2.7 Subjected to image tampering

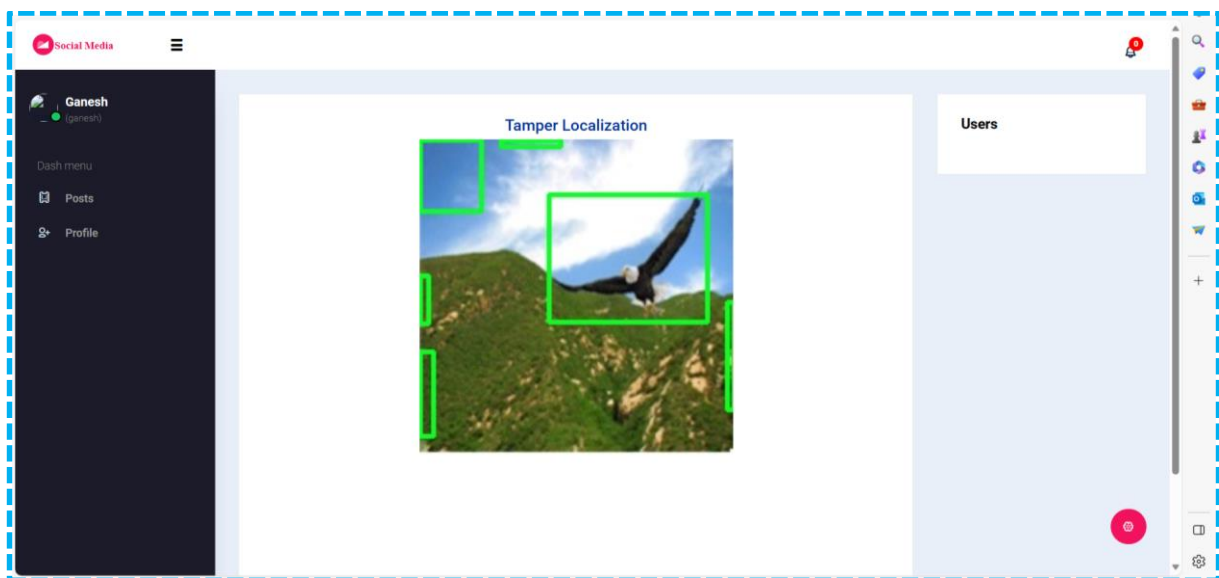


Fig 10.2.8 Tampered section

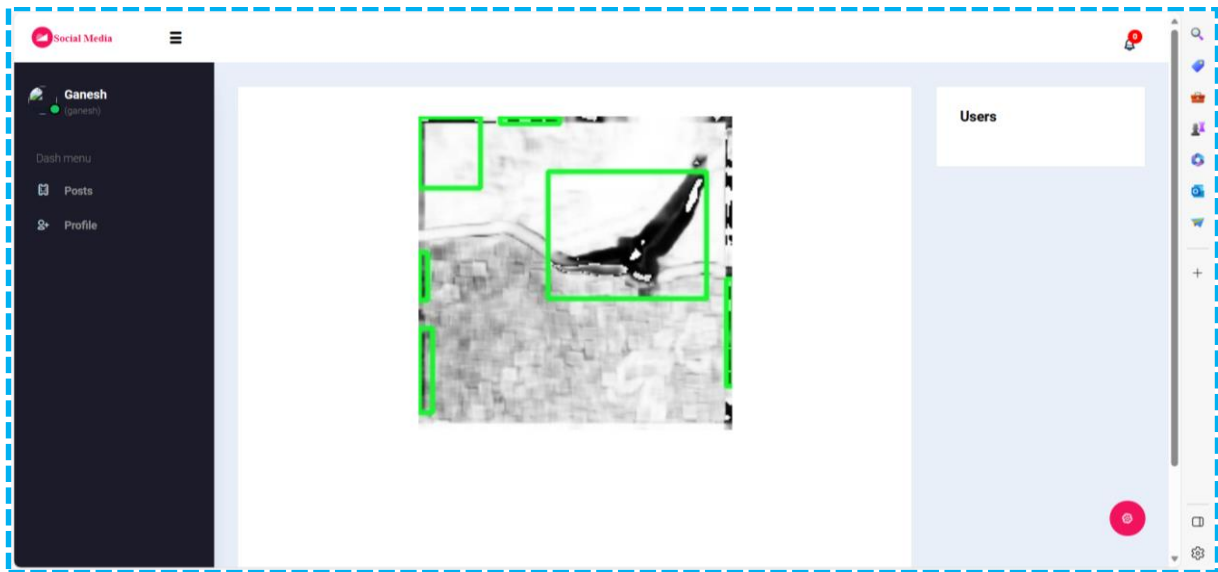


Fig 10.2.9 Marking tampering layer

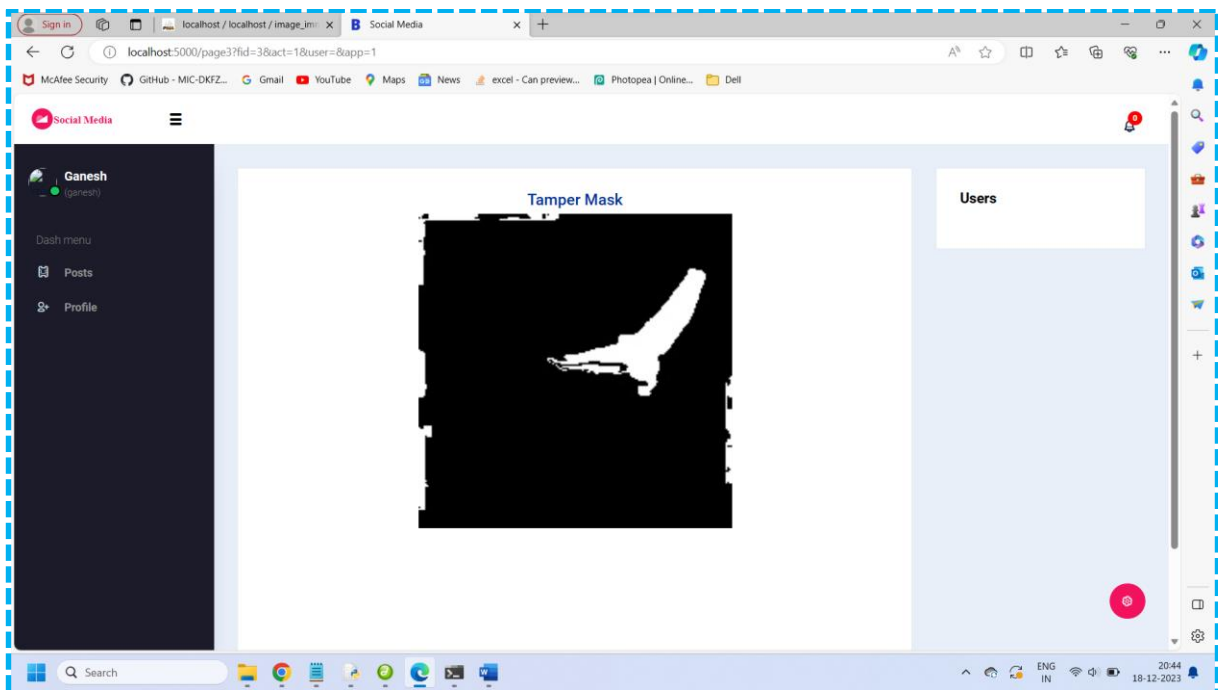


Fig 10.2.10 Highlight Tampered layer



Fig 10.2.11 Marking tamper layer for recovery

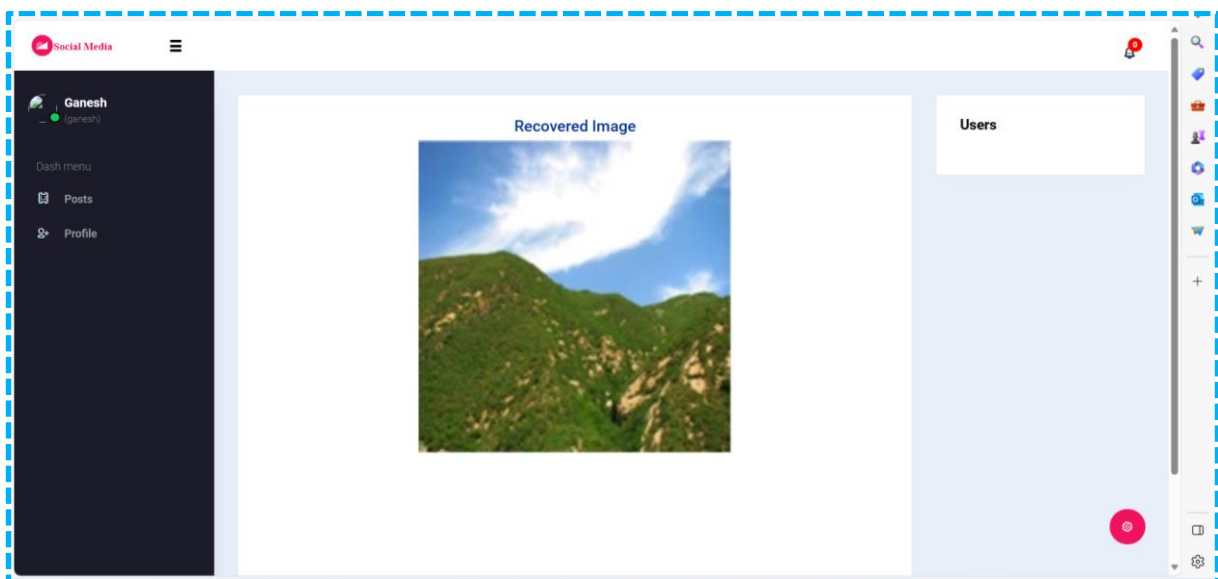


Fig 10.2.12 orgininal image

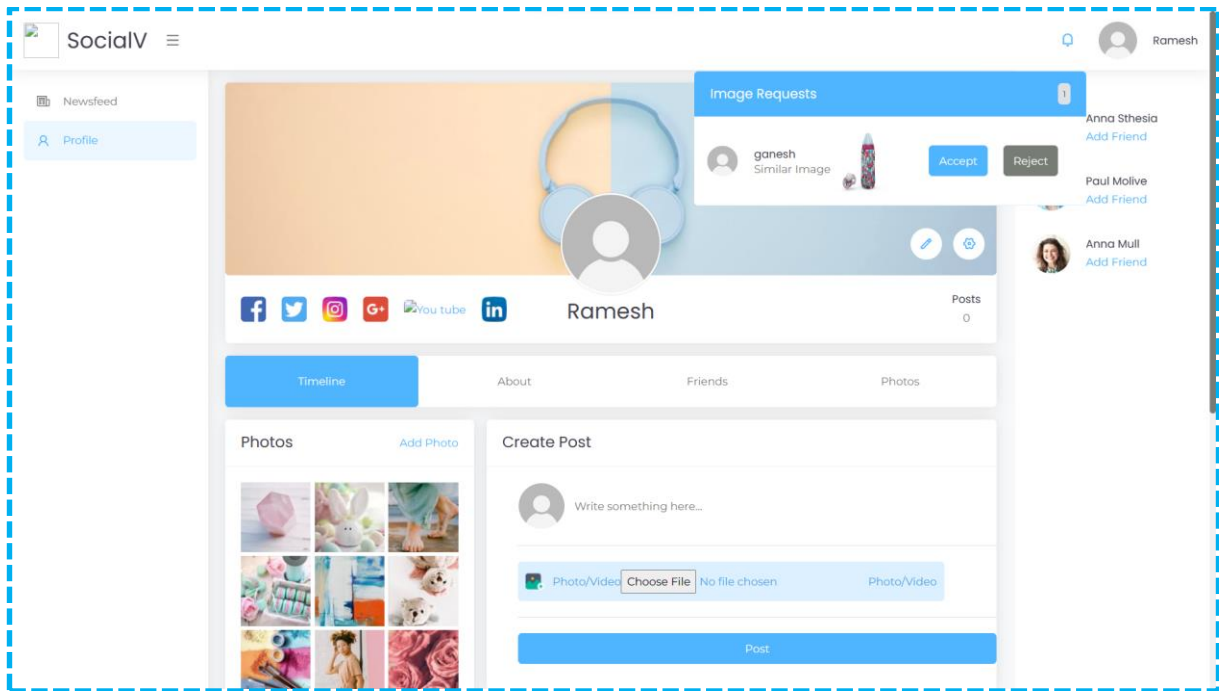


Fig 10.2.13 Upload post

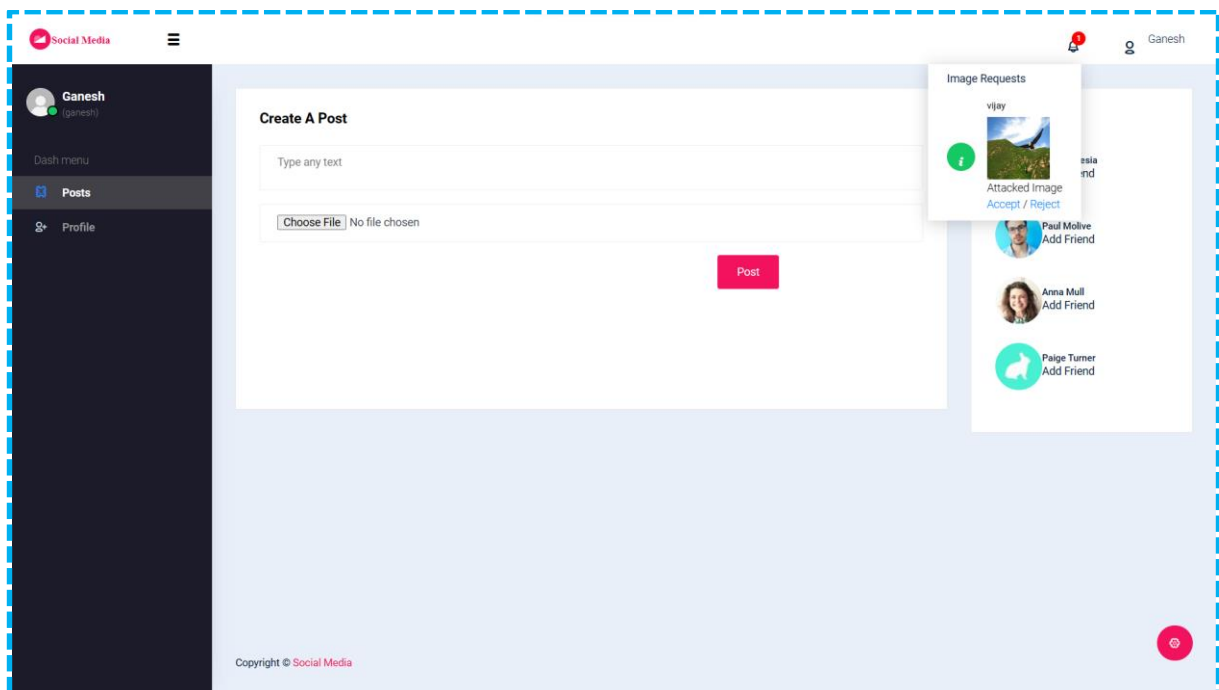


Fig 10.2.14 Notification



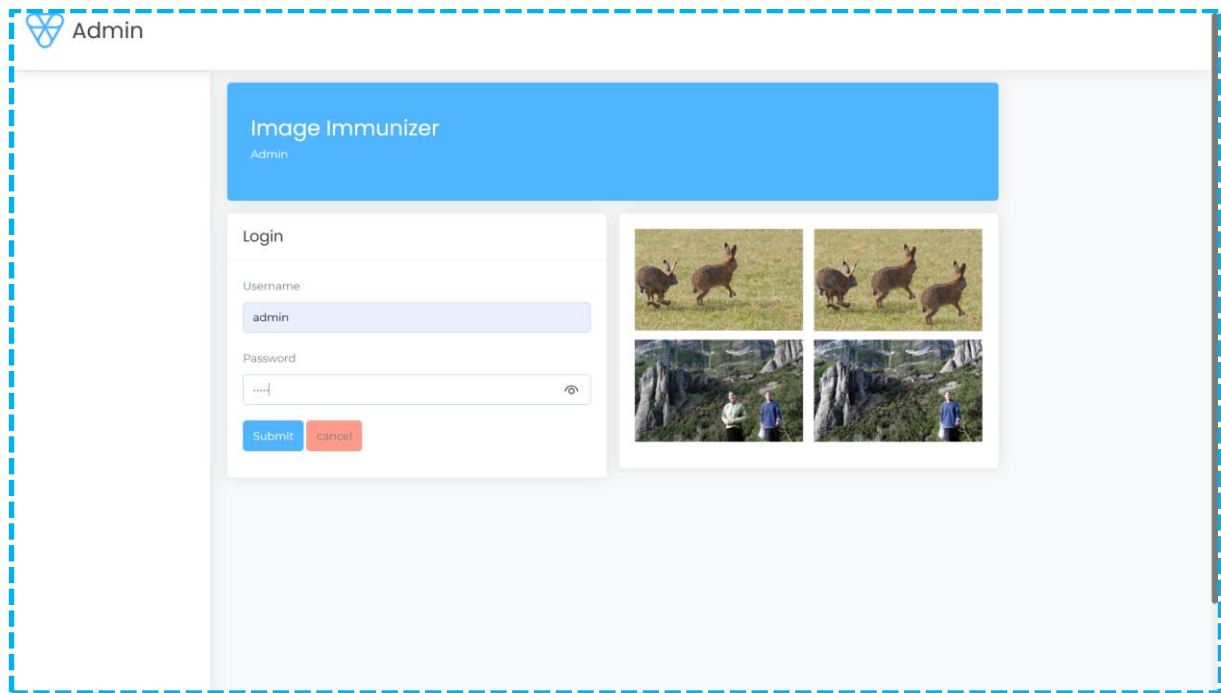


Fig 10.2.15 login for image immunizer

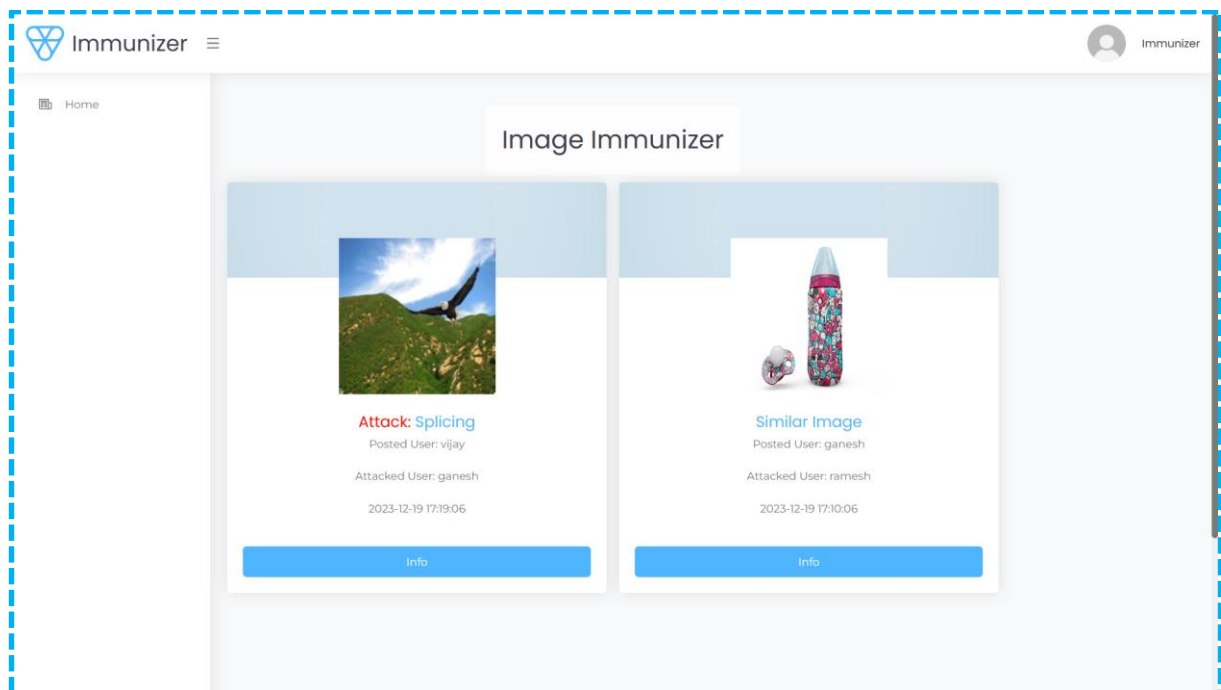


Fig 10.2.16 Report for images

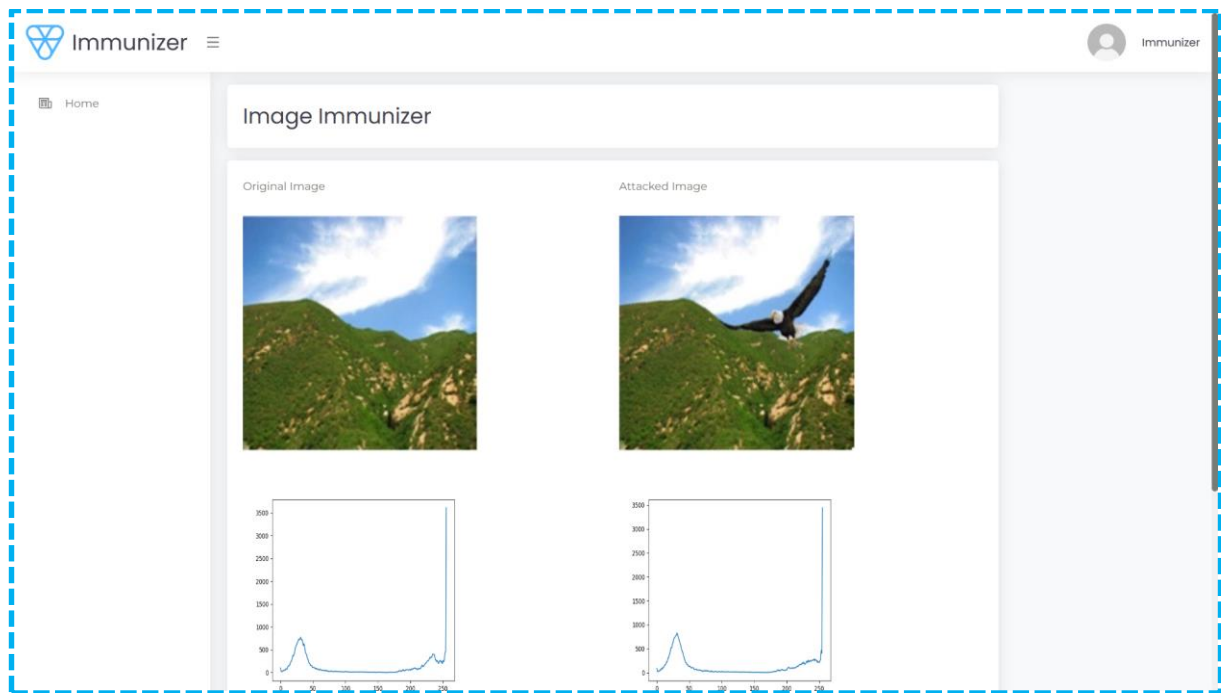


Fig 10.2.17 tamper layer graph

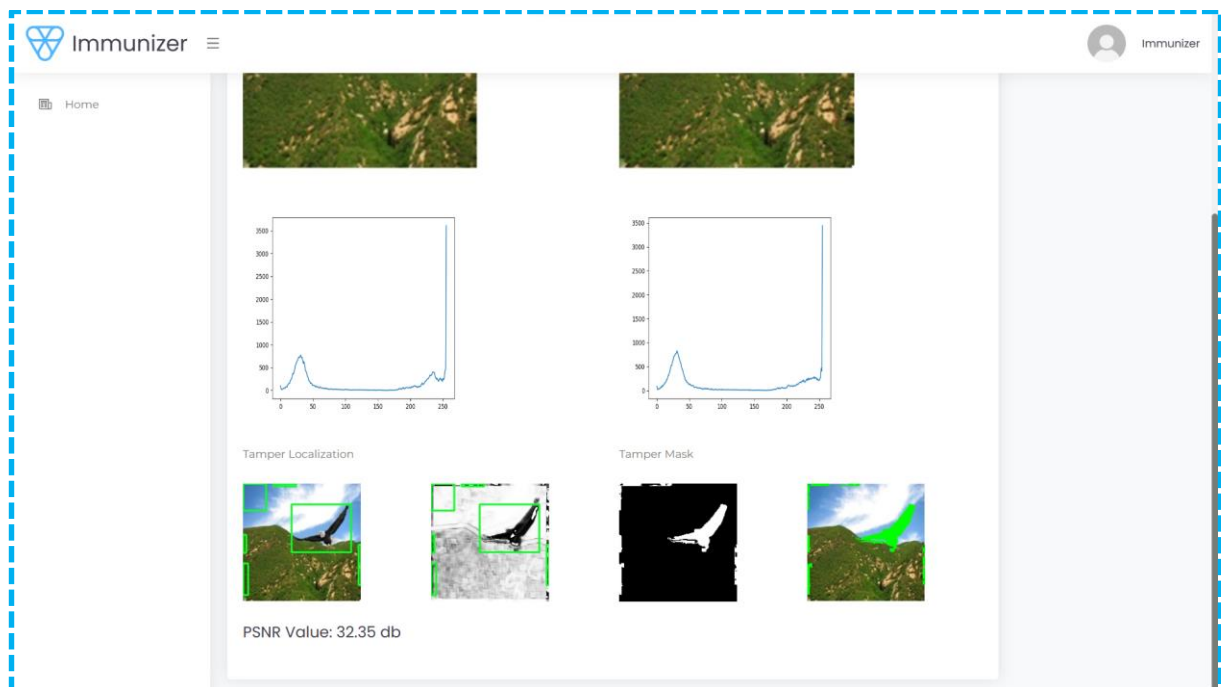


Fig 10.2.18 tamper layer graph with highlight

## REFERENCES

01 Dong, X. Chen, R. Hu, J. Cao, X. Li

- Title: MVSS-Net: Multi-view multi-scale supervised networks for image manipulation detection
- Year: 2023

02 D.-Y. Huang, C.-N. Huang, W.-C. Hu, C.-H. Chou

- Title: Robustness of copy-move forgery detection under high JPEG compression artifacts
- Year: 2017

03 F. Li, Z. Pei, X. Zhang, C. Qin

- Title: Image manipulation localization using multi-scale feature fusion and adaptive edge supervision
- Year: 2022

04 H. Guan et al.

- Title: MFC datasets: Large-scale benchmark datasets for media forensic challenge evaluation
- Year: 2019

05 H. Li, J. Huang

- Title: Localization of deep inpainting using high-pass fully convolutional network
- Year: 2021

06 H. Wu, J. Zhou, J. Tian, J. Liu

- Title: Robust image forgery detection over online social network shared images
- Year: 2022

07 K. Xu, T. Sun, X. Jiang

- Title: Video anomaly detection and localization based on an adaptive intra-frame classification network
- Year: 2020

08 P. Zhuang, H. Li, S. Tan, B. Li, J. Huang

- Title: Image Tampering Localization Using a Dense Fully Convolutional Network
- Year: 2021

09 X. Hu, Z. Zhang, Z. Jiang, S. Chaudhuri, Z. Yang, R. Nevatia

- Title: SPAN: Spatial pyramid attention network for image manipulation localization
- Year: 2020

10 X. Liang, Z. Tang, X. Zhang, M. Yu, X. Zhang

- Title: Robust hashing with local tangent space alignment for image copy detection
- Year: 2023