Project Title: **Innovative Hackathon on Cyber Safety for Women: Unique Image ID for Tracking and Deletion**

**1. Introduction**

**1.1 Problem Statement**  
Cyber safety is a significant concern, especially for women, who are often targeted by cybercrimes such as image-based abuse. When private images are leaked or shared without consent, the consequences can be severe. Traditional methods for tracking and deleting such images are inadequate, as images can quickly spread across multiple platforms, making it difficult to identify and remove all copies.

**1.2 Proposed Solution**  
This project aims to develop an innovative solution that automatically generates a unique ID for every image shared online or on social media, embedding it into the image metadata for easy identification and tracking. By leveraging perceptual image hashing, the system accurately detects and identifies leaks, even if the image is altered or cropped. Each shared image's unique ID is stored in a database, enabling future detection and deletion of leaked images. Acting as an add-on for social platforms, this solution ensures that any leaked image can be efficiently detected and removed, safeguarding user privacy and content integrity.

**2. Project Goals**

**Generate Unique IDs:**  
Automatically generate a unique ID for each image shared online or on social media, based on the image's content and user information.

**Embed IDs in Metadata:**  
Embed the generated unique ID into the image's metadata, ensuring it remains intact even if the image is altered or cropped.

**Perceptual Hashing for Detection:**  
Utilize perceptual image hashing to accurately detect and identify leaks of images across platforms, even if they have been modified.

**AI-Based Search:**Implement an AI tool that searches for the unique ID online, across various platforms and websites, to identify any leaked images.

**Automated Deletion:**  
Develop a system that facilitates the efficient deletion of the identified leaked image and its copies from the web, ensuring content integrity and user privacy.

**3. System Architecture**

**3.1 Components**

**Unique ID Generator:**  
Automatically creates a unique ID for each image shared online by combining the image content with user-specific details. This ID is embedded into the image’s metadata for accurate identification and tracking.

**Metadata Embedder:**  
Embeds the unique ID into the image’s metadata (EXIF or PNG metadata), ensuring that it travels with the image as it is shared across social platforms.

**AI Search Tool:**  
Employs machine learning algorithms and perceptual image hashing to detect and identify leaked images by searching the internet for the embedded unique ID. This tool can recognize images even if they have been altered or cropped.

**Database Storage:**  
Stores each image's unique ID in a database, allowing for efficient future detection and management of potential leaks.

**Deletion Request System:**  
Automates the process of requesting removal of detected leaked images from online platforms, ensuring that unauthorized copies are promptly deleted.

**3.2 Workflow**

**Image Upload:**When a user uploads an image, the system automatically generates a unique ID and stores it in a database.

**ID Embedding:**  
The generated ID is embedded into the image’s metadata before it is shared online, ensuring the ID is included with the image.

**Image Sharing:**  
The user shares the image on social media or other platforms with the unique ID embedded in its metadata.

**Leak Detection:**  
If the image is leaked, the AI search tool utilizes perceptual image hashing to scan the internet and social platforms for the unique ID. This method ensures accurate detection even if the image has been altered or cropped.

**Deletion Requests:**  
The system sends automated requests to platforms hosting the leaked images, facilitating their removal and protecting user privacy and content integrity.

**4. Technical Details**

**4.1 Unique ID Generation**

**Algorithm:**  
The system generates a unique ID for each image shared online or on social media. This ID is created using a hashing algorithm that combines the image's content with user-specific details. The hashing process ensures that each image gets a distinct and secure identifier.

**Hash Function:**  
The unique ID is produced using a cryptographic hash function, such as SHA-256. This function takes as input both the image's content and the user’s account information. The combination of these elements is processed to generate a unique string of characters, which serves as the ID for the image.

**Explanation:**  
The process begins by reading the content of the image and encoding the user's account information. These two pieces of data are combined and passed through a cryptographic hash function, which generates a unique hash value. This hash value is then used as the unique ID for the image. By embedding this unique ID into the image's metadata, the system ensures that the image can be accurately tracked and identified, even if it is shared across different platforms. This method also allows for efficient detection and management of leaked images, protecting user privacy and maintaining content integrity.

**4.2 Metadata Embedding**

**Format:**  
The unique ID is integrated into the image’s EXIF metadata. EXIF (Exchangeable Image File Format) metadata includes various information about the image, such as camera settings and image orientation. In this case, the unique ID is added to the metadata as a "User Comment" field. This method ensures that the ID is embedded directly within the image file, making it easily accessible for future identification and tracking.

**Tools:**  
The embedding process utilizes Python libraries such as Pillow and piexif. Pillow is a widely used library for image processing, and piexif is specifically designed for handling EXIF metadata.

**Process Explanation:**

1. **Open the Image:**  
   The image file is opened using the Pillow library. This allows access to the image data and its existing metadata.
2. **Load EXIF Metadata:**  
   The EXIF metadata is loaded from the image file. This metadata includes various tags and values related to the image's properties and attributes.
3. **Retrieve and Update User Comment:**The existing "User Comment" field in the EXIF metadata is retrieved. If this field is already present, the current content (if any) is preserved. The new unique ID is appended to the existing content or added as new if no previous comment exists.
4. **Update Metadata:**  
   The updated "User Comment" field, now including the unique ID, is written back into the EXIF metadata structure.
5. **Save the Image:**  
   The image, along with the updated EXIF metadata, is saved to a new file. This new file contains the embedded unique ID within its metadata, ensuring it is preserved as the image is shared or distributed.

This approach ensures that each image shared online or on social media contains a unique identifier embedded in its metadata. This identifier aids in future detection and management of potential leaks, aligning with the project’s goal to protect user privacy and content integrity through innovative image tracking solutions.

**4.3 AI-Based Search**

**Technique:**  
The system employs advanced machine learning techniques to detect and identify leaked images. It utilizes perceptual hashing, a method that generates a hash representing the image's visual content. This technique can recognize images even if they have been altered or cropped. Additionally, the system leverages metadata to cross-check and verify the presence of the unique ID embedded within the images. This dual approach—hash comparison and metadata verification—ensures accurate and comprehensive leak detection.

**Libraries:**  
To implement these techniques, the system uses several key libraries:

* **OpenCV:** For image processing tasks, such as downloading and manipulating images.
* **TensorFlow/PyTorch:** For developing and deploying deep learning models, especially for more advanced image recognition tasks.
* **Image Hashing Libraries:** Such as imagehash, to calculate and compare perceptual hashes of images.

**4.4 Automated Deletion**

**Method:**  
The automated deletion process involves sending requests to the APIs provided by social media platforms or utilizing web scraping techniques. This method streamlines the deletion of leaked images by automating the process, reducing the need for manual intervention.

**Tools:**  
The system uses the following tools to perform automated deletion:

* **Python Requests Library**: For sending HTTP requests to social media APIs or deletion endpoints.
* **Web Scraping with BeautifulSoup:** To scrape web pages for deletion options if direct API access is not available.
* **Social Media API’s:** Platforms may provide specific APIs for managing content and handling deletion requests.

In summary, the solution combines AI-based search techniques with automated deletion methods to effectively manage and remove leaked images. The use of perceptual hashing and metadata ensures precise detection of altered or cropped images, while automation tools streamline the deletion process, thus safeguarding user privacy and maintaining content integrity.

**Future Enhancements**

1. **Integration with Social Media Platforms:**  
To enhance the effectiveness of the solution, future improvements will focus on integrating directly with social media platforms. This will allow for real-time monitoring and management of images shared on these platforms. By establishing partnerships or using APIs, the system could automatically track images across different networks, ensuring that any leaks are detected and managed promptly.

2. **Advanced AI Models:**  
To improve the accuracy of leak detection, the system will incorporate more advanced artificial intelligence models. These models will enhance the ability to identify and analyze images, even in cases of significant alteration or cropping. By using more sophisticated algorithms and machine learning techniques, the system will increase its precision in distinguishing between original and leaked images.

**Conclusion**

This project provides a comprehensive solution for safeguarding images shared online, focusing on both technical and strategic aspects of cyber safety. By generating and embedding unique IDs into image metadata, and leveraging perceptual image hashing for accurate detection, the system offers a robust defense against image leaks. The integration of a database for storing unique IDs and automated deletion requests ensures efficient management of leaked images. Future enhancements will further refine the solution with direct social media integration, advanced AI models, and user notifications, reinforcing the commitment to protecting user privacy and content integrity.

**Demo Workflow for the Project Idea**

**Objective**

Demonstrate the process of generating a unique ID for an image, embedding it with security features, and detecting potential leaks through comparison and automated deletion requests.

**Steps**

1. **Upload Original Image**
   * **Image and User Information**: Begin by uploading the original image along with the user's security key and account information.
   * **Generate Unique ID**: Create a unique ID based on the image content and user details. This ID is generated using a combination of hashing and encryption techniques to ensure its uniqueness and security.
2. **Embed Security Features**
   * **Metadata and Watermarks**: Embed the unique ID into the image’s metadata. Additionally, apply watermarks and a cryptographic signature to the image to further secure it against unauthorized use and modifications.
3. **Store the Image**
   * **Upload the Secured Image**: Upload the secured image to a hosting service, such as Postimg. Apply intentional alterations (e.g., slight changes in size or format) to simulate real-world scenarios where the image may be modified or reformatted.
4. **Leak Detection**
   * **Image Download**: Use the provided URL to download the image from the hosting service.
   * **Extract Metadata ID**: Access and extract the metadata ID from the downloaded image.
   * **Hash and Signature Verification**: Compute perceptual and difference hashes for the downloaded image and compare them with the original image. Verify the cryptographic signature to ensure it matches the security features embedded in the original image.
   * **Compare IDs**: Check if the extracted metadata ID from the downloaded image matches the original unique ID.
5. **Leak Detection and Action**
   * **Leak Detection**: If the image is found to be leaked (based on metadata ID and hash comparisons), initiate a delete request to remove the image from the hosting service.
6. **Automated Deletion**
   * **Send Deletion Request**: Automatically send a request to the hosting platform to delete the leaked image, ensuring that unauthorized copies are promptly removed.

**Practical Demonstration**

This workflow outlines a practical demonstration of how the unique ID, embedded security features, and automated tools work together to track, detect, and handle leaked images efficiently. By following these steps, you can ensure the security and integrity of your images while addressing any unauthorized distribution promptly.

**Github Link:**

https://github.com/kishorek0344/Hackathon-Project-Idea-Implementation.git

**Project Demo:**

https://drive.google.com/drive/folders/11kXW-XH55\_G6rL7Ed18fH5nYCmMshiqR?usp=sharing