Advances in Computational Sciences and Technology
ISSN 0973-6107 Volume 17, Number 2 (2024) pp. 77-101

https://dx.doi.org/10.37622/ACST/17.2.2024.77-101

# User-Upload Images on Content Sharing Sites

**M. Devaiah[1]*, S. Ramesh Babu[2], M.Naresh[3]**

*[(1*, 2, 3)] Newtons Institute of Engineering, Macherla-522426, A.P., INDIA.*

**Abstract**

Users can establish photo privacy settings through the innovative Adaptive Privacy Policy Prediction (A3P) system. Social media platforms like Facebook and LinkedIn facilitate the sharing of vast amounts of personal information, leading many individuals to post their photos online in hopes of gaining attention for social interaction. Unfortunately, this trend has resulted in many public consumer photos being available, which raises serious concerns about privacy violations. The risk of identity theft, embarrassment, and even blackmail looms large as personal data proliferates.

To address these challenges, it is essential to consider flexible privacy measures that can help protect users. This initiative aims to explore various privacy policy strategies to enhance the security of private data shared on social networking sites. While numerous researchers have focused on the broader privacy issues and social applications of online sharing, fewer studies specifically address the privacy challenges associated with photo sharing on these platforms. Users often share extensive information with many "friends," and as the volume of images shared increases, so does the potential for privacy infringements. Protecting this privacy is crucial for improving user satisfaction and ensuring a safer online experience.

**Keywords:** Online Social networking communities, content sharing, Security.

## 1. Introduction

These days, one of the main factors facilitating user connectivity is images. For social discovery—to find new peers and discover their interests and social context—sharing occurs both among pre-existing groups of familiar individuals or social circles (e.g., Google+, Flickr, or Picasa) and increasingly with individuals outside of the users' social circles. However, content-sensitive information may be revealed by semantically rich images [1]. Take, for instance, a picture from a student's 2012 commencement. It might be shared in a Flickr group or Google+ circle, but it might unnecessarily reveal the students' friends and BApos family. Therefore, sharing photos on online content-sharing platforms can easily result in uninvited disclosure and privacy violations [3]. Furthermore, other users can gather rich aggregated information about the owner of the published content and the subjects in the published

content due to the persistent nature of online media [3]. The combined data may lead to misuse of personal information and unanticipated exposure of one's social surroundings. Users can enter their privacy preferences on the majority of content-sharing websites. Regretfully, users find it difficult to establish and preserve these privacy settings, according to recent studies [1], [11]. One of the primary explanations offered is that this procedure can be time-consuming and prone to mistakes due to the volume of information exchanged. As a result, a lot of people agree that policy recommendation systems that help users set up their privacy settings correctly and easily are necessary [7]. However, it seems that current ideas for automating privacy settings are insufficient to handle the particular privacy requirements of images. [3], [5], because of the quantity of information that images subtly convey and how they relate to the online context in which are perceived. In this project work, we propose an Adaptive Privacy Policy Prediction (A3P) system that automatically generates personalized policies, giving users a hassle-free privacy settings experience. The A3P system manages user-uploaded photos and takes into account the following factors that affect an individual's image privacy settings: _ the influence of one's personal traits and social surroundings. Users' social environment, including their relationships with others and profile details, may offer valuable insights about their privacy preferences. Users who are interested in photography, for instance, might enjoy showing their images to other amateur photographers. Users may share photos of family occasions with their social acquaintances if they have many family members. Nevertheless, applying uniform policies to all users or users with comparable characteristics could be overly straightforward and unsatisfying of personal preferences. Users' perceptions of the same kind of image can differ greatly. A more conservative person would only choose to share personal photos with his family, whereas a privacy-conscious person might be happy to share all of his photos.

In this paper, different privacy policy techniques for user-uploaded data and images in content-sharing sites are described. The creation of privacy policies is determined by user behavior and image content. There are some benefits and drawbacks to the current systems. Although the A3P system performs better than other approaches, it has a drawback in that it is challenging to establish a privacy policy when uploaded image metadata is not available. In the future, images will be automatically annotated. In computer vision and multimedia content analysis, automatic image annotation is a difficult problem. Images are annotated using a hierarchical framework. First, an image filtering algorithm that eliminates the majority of the images that are relevant to an unlabeled image is presented. As the main relevant image set in the algorithm, a discriminative model is used to assign an image cluster to the unlabeled image. A hybrid annotation model is suggested for image annotation in the second stage. To transfer labels from relevant images to unlabeled images based on global visual features, a baseline method is presented.

## 2. Literature Survey

**a). Over-Exposed? Privacy Patterns and Considerations in Online and Mobile Photo Sharing:** Sharing personal media online has become increasingly effortless and widespread, but this comes with new privacy challenges. The enduring nature of

photos and their context can inadvertently expose details about the physical and social environments where these images were captured. In a groundbreaking study, we explore how context-aware camera phone devices influence privacy decisions in mobile and online photo sharing. By analyzing data on user privacy choices alongside contextual information from a real-world system, we uncover connections between where photos are taken and the privacy settings applied to them. Our findings raise additional questions that we delve into through interviews with 15 users. These discussions reveal recurring themes in privacy considerations, including security, social disclosure, identity, and convenience. We also shed light on various implications and opportunities for developing media-sharing applications, such as leveraging historical privacy patterns to minimize mistakes and oversights.

**b). Privacy Suites: Shared Privacy for Social Networks:** One of the key challenges we face today is creating effective and user-friendly privacy controls for social networks. Many users are often unaware of their privacy settings, which can lead to the accidental sharing of personal information and, in some cases, serious consequences. To address this issue, we propose a fresh approach that allows users to easily choose "packages" of privacy settings curated by friends or trusted professionals, making adjustments only if they wish. This innovative system has the potential to greatly enhance the level of privacy protection for most users with minimal effort, especially since many currently opt for the default settings chosen by the operators.

**c). Sheep Dog–Group and Tag Recommendation for Flickr Photos by Automatic Search-based Learning:** Online photo albums have been prevalent in recent years and have resulted in more and more applications being developed to provide convenient functionalities for photo sharing. In this paper, we propose a system named Sheep Dog to automatically add photos into appropriate groups and recommend suitable tags for users on Flickr. We investigate the problem of gathering training data for concept classification and use concept detection to forecast pertinent concepts of a picture. We present two mechanisms and examine their concept detection performances from the standpoint of obtaining training data through web searches. A ranking-based approach is used to provide reasonable group/tag recommendations for input photos in addition to obtaining trustworthy training data, based on some information already available from Flickr. We test this system using a comprehensive collection of images, and the outcomes show how successful our efforts are.

**d).** Personalizing Image Search Results on Flickr: Flickr is a social media platform that lets users post their images, tag them, add them to groups, and create social networks by adding other users as contacts. Flickr can be browsed or searched in a variety of ways. Tag search is one option that yields all images that have been tagged with a particular keyword. The results of a tag search will contain a lot of images that are unrelated to the sense the user had when executing the query if the keyword is unclear, such as "beetle," which could refer to an automobile. We assert that the

metadata users add, such as contacts and image annotations, reflects their interests in photography. We demonstrate how to take advantage of this metadata to enhance search performance by tailoring the user's search results. First, we demonstrate how filtering tag search results by a user's contacts or a broader social network that includes those contacts' contacts can greatly increase search precision. Furthermore, we present a probabilistic model that leverages tag information to uncover latent topics in the search volume. The tags that users use to annotate their images can also be used to describe their interests. The latent topics found by the model are then used to personalize search results by finding images on topics that are of interest to the user.

## 3. System Analysis

### 3.1. Software Development Life Cycle:

In software engineering, the Software Development Life Cycle, or SDLC for short, is a clearly defined and organized series of steps used to create the desired software product.

#### 3.1.1. SDLC Activities

To efficiently design and develop a software product, a set of steps is provided by the Software Development Life Cycle (SDLC). The following steps are included in the SDLC framework:


#### 3.1.2. Communication

The user makes the initial request for the desired software product in this step. He attempts to negotiate the terms with the service provider. He makes a written request to the organization that provides the service.

#### 3.1.3. Requirement Gathering

The software development team continues to work on the project after this step. To learn as much as possible about the needs of the different stakeholders in the problem

domain, the team has conversations with them. The requirements are carefully considered and divided into three categories: functional, system, and user requirements. The requirements are collected using several practices as given -
Studying the existing or obsolete system and software,
Conducting interviews of users and developers,
Referring to the database or
Collecting answers from the questionnaires.

### 3.1.4. Feasibility Study

After requirement gathering, the team comes up with a rough plan of the software process. At this stage, the team examines whether software can be developed to meet all user needs and whether there is a chance that it will become useless. The project's financial, practical, and technological viability for the organization is determined. Numerous algorithms are available to assist developers in determining whether a software project is feasible.

### 3.1.5. System Analysis

At this stage, the developers choose a plan and attempt to identify the best software model for the project. Understanding software product limitations, learning about system-related issues or necessary modifications to current systems in advance, determining and resolving the project's effects on the organization and its employees, etc., are all components of system analysis. The project team plans the schedule and resources based on an analysis of the project's scope.

### 3.1.6. Software Design

The next stage is to design the software product and put all of the requirements and analysis knowledge on the desk. This step's inputs are user inputs and data acquired during the requirement gathering stage. Two designs—a logical design and a physical design—are the result of this step. Engineers create logical diagrams, data-flow diagrams, meta-data and data dictionaries, and occasionally pseudo codes.

### 3.1.7. Coding

This step is also known as the programming phase. The implementation of software design starts in terms of writing program code in a suitable programming language and developing error-free executable programs efficiently.

### 3.1.8. Testing

An estimate says that 50% of the whole software development process should be tested. Errors may ruin the software from a critical level to its own removal. Software testing is carried out by developers while they are coding, and testing specialists perform comprehensive testing at different code levels, including module, program, product, in-house, and user-end testing. The secret to dependable software is the early identification of errors and their correction.

### 3.1.9. Integration

It might be necessary to integrate software with databases, libraries, and other programs. Software integration with external entities is the focus of this SDLC stage.

### 3.1.10. Implementation

This entails setting up the program on user computers. Software occasionally requires user-end post-installation configurations. Issues pertaining to integration are resolved during implementation, and software is tested for portability and adaptability.

### 3.1.11. Operation and Maintenance

This stage verifies that the software is operating more effectively and with fewer errors. Users receive training or assistance with the documentation on how to use the software and maintain its functionality, if necessary. By updating the code in accordance with developments in the user-end environment or technology, the software is kept up to date. Hidden bugs and undiscovered real-world issues could present difficulties during this phase.

### 3.1.12. Disposition

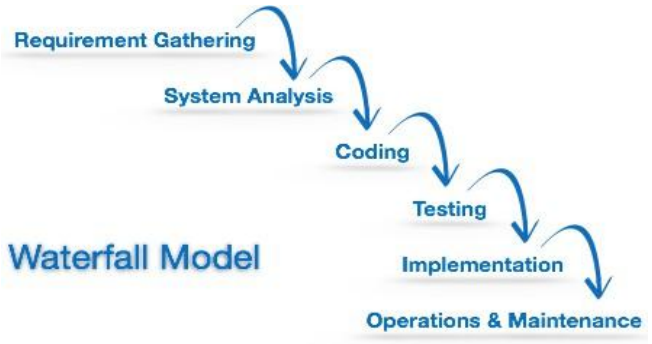
Over time, the software's performance may deteriorate. It might require significant upgrades or become totally outdated. Therefore, there is an urgent need to get rid of a significant part of the system. Archiving data and necessary software components, shutting down the system, organizing disposal activities, and ending the system at the proper end-of-system time are all included in this phase.

### 3.1.13. Software Development Paradigm

The paradigm for software development aids developers in choosing a software development approach. A software development paradigm defines the software development life cycle and has its own set of tools, techniques, and procedures that are clearly stated. The following are some definitions of software development paradigms or process models:

## 3.2. Waterfall Model

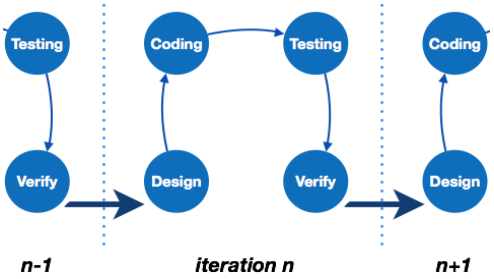
The most basic model of the software development paradigm is the waterfall model. It declares that the SDLC will be carried out in a sequential manner for each step. To put it another way, the second phase won't start until the first one is finished, and so on.



This model makes the assumption that everything goes according to plan in the previous stage and that there is no need to consider potential problems from the past that might come up in the subsequent stage. If there are still problems from the previous step, this model does not function properly. We are unable to reverse or redo our actions due to the sequential nature of the model.
When developers have previously designed and developed similar software and are familiar with all of its domains, this model works best.

### 3.3. Iterative Model

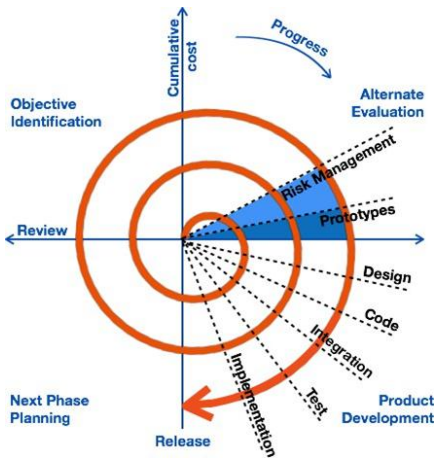
Iterations of the software development process are led by this model. It repeats each step after each cycle of the SDLC process, projecting the development process in a cyclical fashion.




The software is first created on a very small scale, considering each step as it is done. With each iteration that follows, more features and modules are developed, coded, tested, and added to the program. Each cycle produces software that is complete in and of itself, with more features and capabilities than the previous one. After each iteration, the management team can focus on risk management and prepare for the next one. Because a cycle only includes a small portion of the entire software process, it is easier to manage the development process even though it uses more resources.

### 3.4. Spiral Model

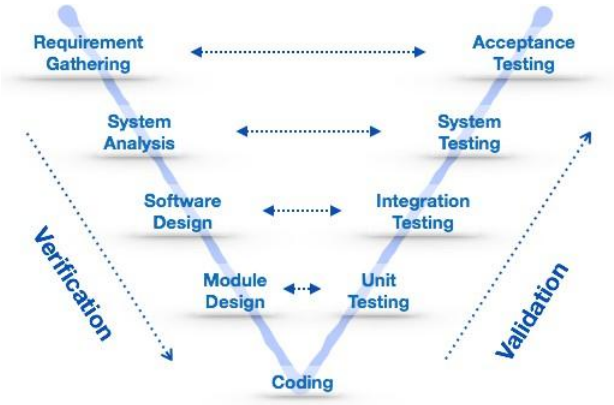
The spiral model is a hybrid of one of the SDLC models and an iterative model. It is comparable to combining a cyclic process (iterative model) with one SDLC model.



This model takes risk into account, something that most other models frequently overlook. At the beginning of one iteration, the model begins by identifying the software's goals and limitations. Software prototyping is the next stage. Risk analysis is part of this. The software is then developed using a single standard SDLC model. The next iteration's plan is created in the fourth phase.

### 3.5. V– Model

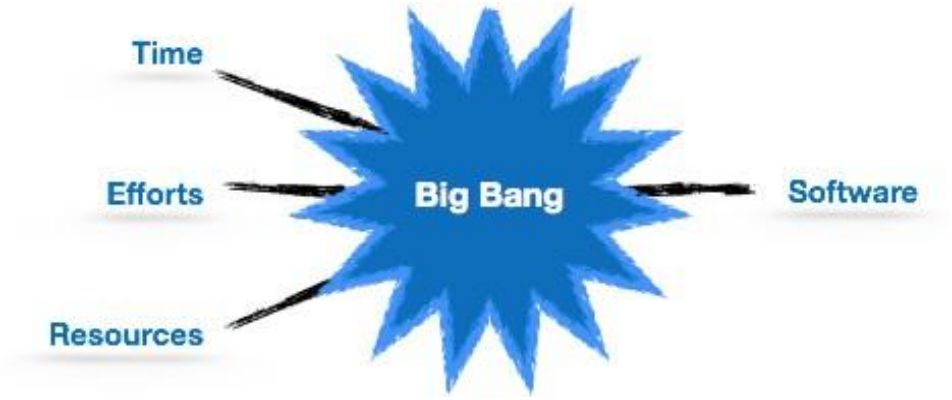
The waterfall model's main flaw is that we only proceed to the next step after the previous one is complete, and there is no way to go back if something is discovered to be incorrect later on. The V-Model offers a way to test software in reverse at every stage.




The primary drawback of the waterfall model is that we only move on to the next phase once the previous one is finished, and there is no way to reverse course if something turns out to be wrong later. Software can be tested in reverse at every stage with the V-Model. To verify and validate the product in compliance with the stage's requirements, test cases and test plans are created at every stage. For example, during the requirement gathering phase, the test team prepares all test cases in compliance with the requirements. Once the software has been developed and is ready for testing, test cases from this stage verify that it meets the requirements.
As a result, validation and verification proceed simultaneously. The verification and validation model is another name for this model.

### 3.6. Big Bang Model

Labels, inscriptions, and remarks are the metadata that we take into consideration in our work. Recover each metadata vector's hyponym. Choose the hyponym that has the most remarkable recurrence. An incremental method is a subcategory that a picture belongs to. In the beginning, the primary image becomes the up category itself, and the picture's delegate hyponyms become the illustrative hyponyms for the subcategory. Keep track of the distance between each current subcategory and the agent hyponyms of an upcoming image.




There is very little planning needed for this model. It doesn't adhere to any procedure, and occasionally the client is unsure of the specifications and upcoming demands. Thus, the requirements for input are arbitrary. Although this model works well for learning and experimentation, it is not appropriate for large software projects.

**3.7 System Requirements Specification**

The purpose, scope, definitions, acronyms, abbreviations, references, and an overview of the entire Software Requirements Specification (SRS) are all included in the introduction. By thoroughly defining the problem statement, this document seeks to collect, examine, and provide a comprehensive understanding of the entire Marvel Electronics and Home Entertainment software system. However, while defining high-level product features, it also focuses on the capabilities and needs of stakeholders. This document contains the specific requirements for Marvel Electronics and Home Entertainment.

**3.8. Functional Requirements**

The definition of a functional requirement is any requirement that specifies what the system should do.

**5. System Design**

**5.1.1. System Specifications**

**Hardware Requirements:**

- System :Pentium IV3.4GHz.
- Hard Disk :40 GB.
- Floppy Drive :1.44 Mb.
- Monitor :14’ Colour Monitor.
- Mouse :Optical Mouse.
- Ram :1 GB.

**Software Requirements:**

- Operating system :Windows Family.
- Coding Language :J2EE(JSP, Servlet, JavaBean)
- Data Base :MS Access.
- Web Server :Tomcat 6.0

**5.5.2. System Components**

A system component is a process, program, utility, or another part of a computer's operating system that helps manage different areas of the computer. Not to be confused with a hardware component, a system component is similar to a computer program, but is not something an end-user directly interacts with when using a computer. There are multiple system components at work in a computer operating system, each serving a specific function. Together, they allow the operating system and computer to function correctly and efficiently.

**5.2.1. Process Management**

The process management component is responsible for overseeing the various processes running within the operating system. Each software program is linked to one or more processes that activate when the program is in use. For instance, using an Internet browser means that a specific process dedicated to that browser is actively running. The operating system itself also hosts numerous processes, each serving a unique purpose. It is the role of process management to keep all these processes organized, ensure they operate effectively, allocate the necessary memory, and terminate them when required.

**5.2.2. Memory Management**

The memory management component, also sometimes called main memory management or primary memory management, handles primary memory, or RAM. When programs are running, including the operating system, those programs store data in RAM for quick access at any time. Memory management monitors and manages the memory and knows which blocks of memory are in use, which programs are using memory, and which memory blocks are available to be used.

**5.2.3. File Management**

The file management component manages just about anything with computer files. When a file is created, file management is involved in the creation of the file, including where it is stored on a storage device. When a file is modified, file management helps with the modification of the file. If a file is deleted, file management is there to help with deleting the file and freeing up the space for another file to be stored there at a later time.

File management also handles tasks related to the creation, modification, and deletion of folders, or directories, on a storage device.

**5.2.4. Secondary Storage Management**

The secondary storage management component works with storage devices, like a hard drive, USB flash drive, DVD drive, or floppy disk drive. While the file management component takes care of the actual files on the storage device, the secondary storage management component manages the storage device itself. It manages the available space, or free space, on the storage device and allocates space for new files to be stored there. Requests for data on a storage device are handled by secondary storage management as well. For example, when a user double-clicks on a

file to open it, secondary storage management receives that request and helps in the retrieval of that file from the storage device.

### 5.2.5. Access Management

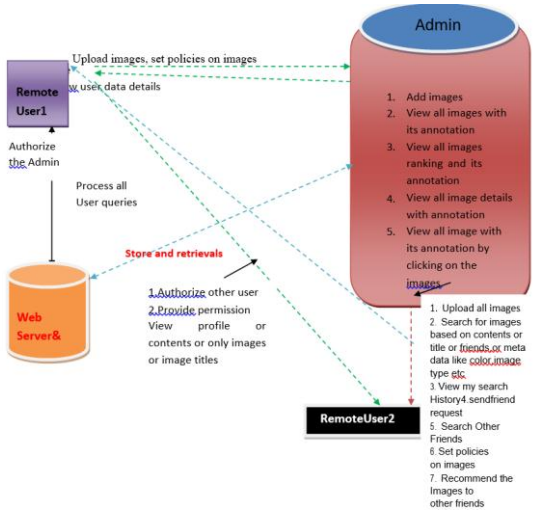
The access management component is tasked with managing user access to data on a computer. Accounts provide each user with specific access to software, files, and functionality within an operating system. The ability to install a software program is controlled by access management. Access to view, edit, and delete a file is managed by access management. Changing settings within the operating system is managed by access management. How a user interacts with the computer operating system and uses software is handled by access management, about the permissions they have been granted through user accounts.

### 5.2.6. System Resource Management

The system resource management component is responsible for managing the allocation of system resources, like memory and CPU time. When programs are running, they require the use of memory and CPU time to function properly. System resource management determines how much memory and CPU time that program is allowed to use at any given time.

Managing system resource usage is a big responsibility, as it can directly impact the performance of the computer. If too many resources are allocated to one program and there is not enough to allocate to another program, functionality in the second program will appear to be slow or sometimes even unresponsive. If the operating system does not have enough resources allocated to it, the entire computer can run slow or stop working altogether. It is the responsibility of system resource management to make sure system resources are allocated properly, including taking resources away from one program that does not need it and allocating those resources to another program that does need it.

## 5.3. System Architecture



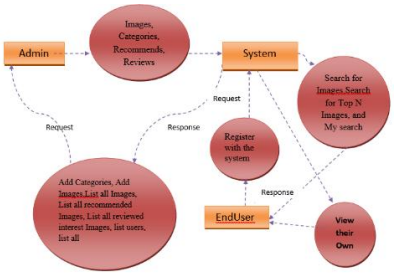
### 5.4. Data Flow Diagram

Another name for the DFD is a bubble chart. A system can be represented using this straightforward graphical formalism by showing the input data, the different processing operations performed on the data, and the output data that the system generates.

One of the most crucial modeling tools is the data flow diagram (DFD). The system components are modeled using it. The system process, the data that the process uses, an outside party that communicates with the system, and the information flows within the system are all examples of these components.

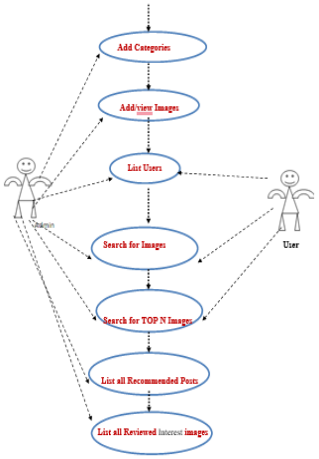
DFD illustrates the flow of information through the system and the various transformations that alter it. It is a visual method that illustrates the flow of information and the changes made as data passes from input to output.

A DFD, sometimes referred to as a bubble chart, can depict a system at any abstraction level. It can be divided into tiers that correspond to escalating functional detail and information flow.



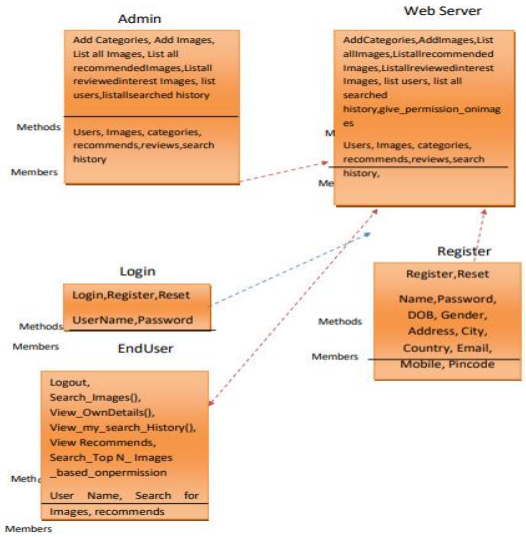

### 5.5. Use Case Diagram

According to the Unified Modeling Language (UML), a use case diagram is a particular kind of behavioral diagram that is produced from and defined by a use-case analysis. In terms of actors, their objectives (shown as use cases), and any dependencies between those use cases, it serves to visually summarize the functionality offered by a system. A use case diagram's primary objective is to illustrate which actors utilize the system's functionalities. The roles of the actors in the system can be depicted.



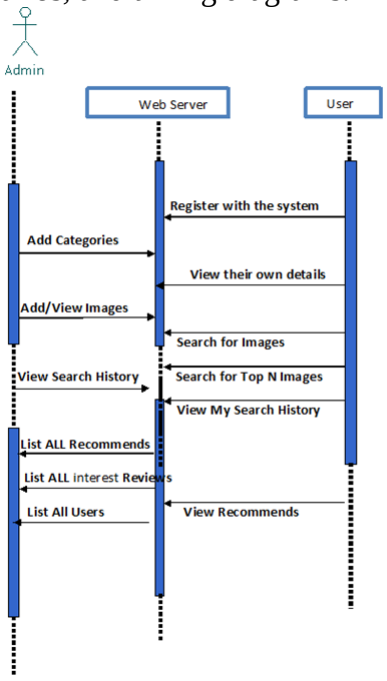
**Class Diagram**

A class diagram in software engineering is a kind of static structure diagram that illustrates a system's classes, attributes, operations (or methods), and relationships between the classes using the Unified Modeling Language (UML). It clarifies which class has the data.



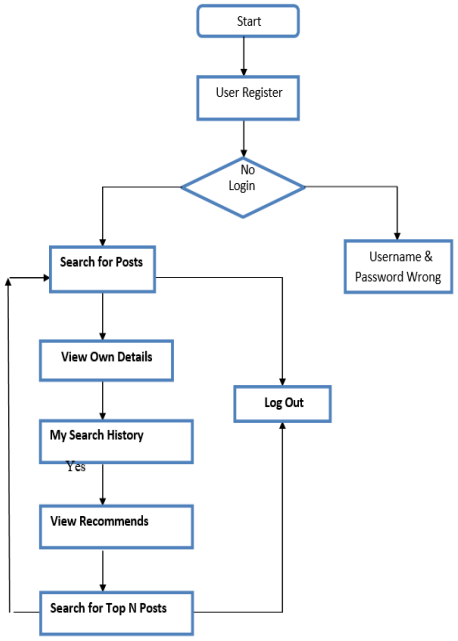

**5.6.1 Sequence Diagram**

In the Unified Modeling Language (UML), one kind of interaction diagram that shows the order and way processes interact with one another is a sequence diagram. It is a construct of a message sequence chart. Sequence diagrams are also known as event diagrams, event scenarios, and timing diagrams.

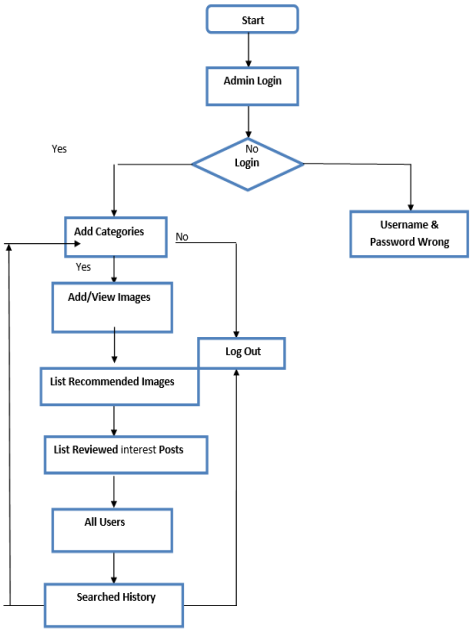


### 5.7 Flow Chart Diagram

**Flow Chart1:** Use




**Flow Chart2:** Admin




## 6. System Implementation

### 6.1. Modules Description:

#### 6.1.1. Modules:

- System Construction Module
- Content-Based Classification
- Metadata-Based Classification
- Adaptive Policy Prediction

### 6.2 Module Description:

#### 6.2.1. System Construction Module

A3P-core and A3P-social are the two primary parts of the A3P system. The following is the general data flow. An image uploaded by a user is initially routed to the A3P-core. The image is classified by the A3P-core, which also assesses whether the A3P-social needs to be called. Based on the users' past behavior, the A3P-core typically makes direct policy predictions for them. If either of the two situations below is verified to be true, A3P-core will contact A3Psocial: (i) The user's increased social networking activities (adding new friends, posting on their profile, etc.) and recent significant changes in their community regarding their privacy practices are detected by the A3P-core; (ii) The user lacks sufficient information about the type of uploaded image to perform policy prediction.

### 6.2.2. Content-Based Classification

We propose a hierarchical image classification system that first classifies images based on their contents and then further refines each category into subcategories based on their metadata to obtain groups of images that may be associated with similar privacy preferences. Pictures without metadata will only be grouped by content. This kind of hierarchical classification gives image content priority and lessens the effect of missing tags. Please take note that certain images may be included in more than one category if they have the metadata or typical content features of those categories.
The foundation of our content-based classification strategy is an accurate and efficient image similarity method. Our classification algorithm specifically compares image signatures that are defined using the sanitized and quantified Haar wavelet transformation. Each image's color, size, invariant transform, shape, texture, symmetry, and other spatial and frequency information are encoded by the wavelet transform. After that, a select few coefficients are chosen to create the image's signature. The distance between the image signatures is then used to calculate the content similarity between the images.

### 6.2.3. Metadata-Based Classification

Under the previously mentioned baseline categories, the metadata-based classification divides images into subcategories. There are three primary steps in the process. Extracting keywords from an image's associated metadata is the first step. Our work takes into account tags, captions, and comments as metadata. Finding a representative hypernym (represented by h) from each metadata vector is the second step. Finding a subcategory to which an image belongs is the third step. This process is incremental. The first image initially creates a subcategory for itself, and the image's representative hypernyms turn into the representative hypernyms for the subcategory.

### 6.2.4.Adaptive Policy Prediction

The policy prediction algorithm provides a predicted policy of a newly uploaded image to the user for his/her reference. More importantly, the predicted policy will reflect the possible changes of a user’s privacy concerns. The prediction process consists of three main phases: (i) policy normalization; (ii) policy mining; and (iii) policy prediction.

**Sample Code:**
**Results:**

```
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartFrame;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.DefaultCategoryDataset;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
```

```
public class Results {
   public static void main(String[] args) {
      int total1 = 0, total2 = 0, total3 = 0, total4 = 0, total5 = 0;
      String c1 = null, c2 = null, c3 = null, c4 = null, c5 = null;
      int fr = 0, br = 0, bfr = 0, sfr = 0, lfr = 0;
      try {
         // Load MySQL JDBC Driver
         Class.forName("com.mysql.jdbc.Driver");

         // Establish the connection
         Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ppi", "root", "root");
         System.out.println("Connected");
         // Create statement and execute query
         Statement st = con.createStatement();
         ResultSet rs = st.executeQuery("SELECT * FROM images");
         // Iterate through the result set
         while (rs.next()) {
            String iname = rs.getString("tag");
            // Handle cases for different image tags
            if (iname.equalsIgnoreCase("Cat")) {
               c1 = "Cat";
               fr = rs.getInt("count");
               total1 += fr;
            }
            if (iname.equalsIgnoreCase("Deer")) {
               c2 = "Deer";
               br = rs.getInt("count");
               total2 += br;
            }
            if (iname.equalsIgnoreCase("Tiger")) {
               c3 = "Tiger";
               bfr = rs.getInt("count");
               total3 += bfr;
            }
            if (iname.equalsIgnoreCase("Dog")) {
               c4 = "Dog";
               sfr = rs.getInt("count");
               total4 += sfr;
            }
            if (iname.equalsIgnoreCase("Elephant")) {
               c5 = "Elephant";
               lfr = rs.getInt("count");
               total5 += lfr;
            }
```

```
        }
        // Create dataset for bar chart
        DefaultCategoryDataset dataSet = new DefaultCategoryDataset();
        dataSet.setValue(total1, "Cat Rank", c1);
        dataSet.setValue(total2, "Deer Rank", c2);
        dataSet.setValue(total3, "Tiger Rank", c3);
        dataSet.setValue(total4, "Dog Rank", c4);
        dataSet.setValue(total5, "Elephant Rank", c5);
        // Create 3D Bar Chart
        JFreeChart chart = ChartFactory.createBarChart3D(
          "Privacy Policy Inference of User Uploaded Images on Content Sharing
Sites", // Chart title
          "Total Image Rank Details", // Category axis label
          "No. of Ranks", // Value axis label
          dataSet, // Dataset
          PlotOrientation.VERTICAL, // Orientation
          true, // Include legend
          true, // Tooltips
          true // URLs
        );
        // Display chart in a frame
        ChartFrame chartFrame = new ChartFrame("Privacy Policy Inference of User
Uploaded Images on Content Sharing Sites Rank Details", chart);
        chartFrame.setVisible(true);
        chartFrame.setSize(800, 500);
      } catch (Exception ex) {
        System.out.println(ex);
      }
    }
}
```

Insert Image

```
<%@ page import="com.oreilly.servlet.*, java.sql.*, java.text.SimpleDateFormat,
java.util.*, java.io.*, javax.servlet.*, javax.servlet.http.*" %>
<%@ include file="connect.jsp" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <title>PPI: Image Insertion Page</title>
   <link href="css/tooplate_style.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="wrapper">
```

```
   <div id="main">
      <h2>Image Uploading!!!</h2>
      <div style="width:380px; margin:0 auto;">
<%
try {
   String imguname = (String) application.getAttribute("imageuname");
   ArrayList<FileInputStream> list = new ArrayList<>();
   ServletContext context = getServletContext();
   String dirName = context.getRealPath("Gallery/");
   String paramname = null, image = null;
   String a = null, b = null, c = "", d = null;
   String[] ee = null;
   String checkBok = "";
   int lyke = 0;
   String bin = "";
   FileInputStream fs = null;
   File file1 = null;
   MultipartRequest multi = new MultipartRequest(request, dirName, 10 * 1024 *
1024); // 10MB
   Enumeration<?> params = multi.getParameterNames();
      // Processing parameters
   while (params.hasMoreElements()) {
      paramname = (String) params.nextElement();
            if (paramname.equalsIgnoreCase("tag")) {
         a = multi.getParameter(paramname);
      } else if (paramname.equalsIgnoreCase("color")) {
         b = multi.getParameter(paramname);
      } else if (paramname.equalsIgnoreCase("annotation")) {
         c += multi.getParameter(paramname);
      } else if (paramname.equalsIgnoreCase("uses")) {
         d = multi.getParameter(paramname);
      } else if (paramname.equalsIgnoreCase("policy")) {
         ee = multi.getParameterValues(paramname);
      } else if (paramname.equalsIgnoreCase("pic")) {
         image = multi.getParameter(paramname);
      }
   }
   for (String str : ee) {
      checkBok += "|" + str;
   }
   if (checkBok.contains("All")) {
      checkBok = "All";
   }
   Enumeration<?> files = multi.getFileNames();
      // Processing file upload
```

```
   while (files.hasMoreElements()) {
      paramname = (String) files.nextElement();
      if (paramname != null) {
         image = multi.getFilesystemName(paramname);
         String fPath = context.getRealPath("Gallery/" + image);
         file1 = new File(fPath);
         fs = new FileInputStream(file1);
         list.add(fs);
         FileInputStream fis = new FileInputStream(fPath);
         StringBuffer sb1 = new StringBuffer();
         int i;
         while ((i = fis.read()) != -1) {
            String hex = Integer.toHexString(i);
            sb1.append(hex).append(",");
            String binFragment = "";
            for (int i1 = 0; i1 < hex.length(); i1++) {
               int iHex = Integer.parseInt("" + hex.charAt(i1), 16);
               binFragment = Integer.toBinaryString(iHex);

               while (binFragment.length() < 4) {
                  binFragment = "0" + binFragment;
               }
               bin += binFragment;
            }
         }
         fis.close();
      }
   }
   PreparedStatement ps = connection.prepareStatement(
      "INSERT INTO images (tag, color, annotation, uses, policy, imagess, count,
binaryimage, imagetitle, uname) " +
      "VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
   );
   ps.setString(1, a);
   ps.setString(2, b);
   ps.setString(3, c);
   ps.setString(4, d);
   ps.setString(5, checkBok);
   ps.setBinaryStream(6, fs, (int) (file1 != null ? file1.length() : 0));
   ps.setInt(7, lyke);
   ps.setString(8, bin);
   ps.setString(9, image);
   ps.setString(10, imguname);

   int x = ps.executeUpdate();
```

```
    if (x > 0) {
        out.println("Image successfully added.");
    } else {
        out.println("Failure");
    }
} catch (Exception e) {
    out.println("Error: " + e.getMessage());
}
%>
        </div>
    </div>
</div>
</body>
</html>
```

**SEND REQUEST:**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>PPI: Send Request Page</title>
    <link rel="stylesheet" href="css/reset.css" type="text/css" media="screen">
    <link rel="stylesheet" href="css/style.css" type="text/css" media="screen">
    <link rel="stylesheet" href="css/layout.css" type="text/css" media="screen">
    <script src="js/jquery-1.6.min.js" type="text/javascript"></script>
    <script src="js/cufon-yui.js" type="text/javascript"></script>
    <script src="js/cufon-replace.js" type="text/javascript"></script>
    <script src="js/Open_Sans_400.font.js" type="text/javascript"></script>
    <script src="js/Open_Sans_Light_300.font.js" type="text/javascript"></script>
    <script src="js/Open_Sans_Semibold_600.font.js" type="text/javascript"></script>
    <script src="js/tms-0.3.js" type="text/javascript"></script>
    <script src="js/tms_presets.js" type="text/javascript"></script>
    <script src="js/jquery.easing.1.3.js" type="text/javascript"></script>
    <script src="js/FF-cash.js" type="text/javascript"></script>
    <!--[if lt IE 9]>
        <script src="js/html5.js" type="text/javascript"></script>
        <link rel="stylesheet" href="css/ie.css" type="text/css" media="screen">
    <![endif]-->
</head>
<body id="page1">
<div class="bg">
    <div class="main">
        <!-- Header Section -->
        <header>
            <div class="row-1">
```

```
            <h1 style="width:600px;">
              <span class="slog">Privacy Policy Inference of User-Uploaded Images on Content Sharing Sites</span>
            </h1>
            <form id="search-form" action="#" method="post" enctype="multipart/form-data">
              <fieldset>
                <div class="search-form">
                  <input type="text" name="search" value="Type Keyword Here"
                       onBlur="if(this.value==")this.value='Type Keyword Here'"
                       onFocus="if(this.value=='Type Keyword Here')this.value=""/>
                  <a href="#">Search</a>
                </div>
              </fieldset>
            </form>
          </div>
          <div class="nav-bar" style="background-color:#000000; height:40px;">
            <nav>
              <!-- Uncomment this section if you wish to use the navigation menu -->
              <!--
              <ul class="menu">
                <li><a href="index.html">Home Page</a></li>
                <li><a href="about.html">About Us</a></li>
                <li><a href="admin.html">Admin</a></li>
                <li><a class="active" href="user.html">User</a></li>
                <li class="last-item"><a href="register.jsp">Register</a></li>
              </ul>
              -->
            </nav>
          </div>
        </header>
        <!-- Content Section -->
        <section id="content">
          <div class="padding">
            <div class="wrapper">
              <div class="col-2">
                <div class="block-news">
                  <h4 class="color4p2">Welcome to <%= application.getAttribute("uname") %></h4>
                  <div class="wrapperp2">
                    <ul class="tooplate_list">
                      <li><a href="user_menu.jsp">View Profile</a></li>
                      <li><a href="user_add_image.jsp">Add Image</a></li>
                      <li><a href="usersearch.jsp">User Search</a></li>
                      <li><a href="sendrequest.jsp">Send Friend Request</a></li>
```

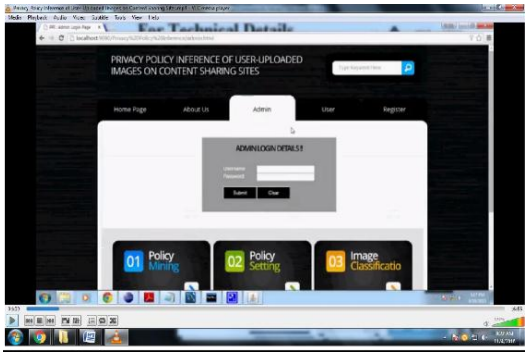
```
                        <li><a href="searchfriend.jsp">Search Other Friends</a></li>
                        <li><a href="usersearchhistory.jsp">View My Search History</a></li>
                        <li><a href="setpolicies.jsp">Set Policies On Image</a></li>
                        <li><a href="recommendfriend.jsp">Recommend Image to Other Friends</a></li>
                        <li><a href="viewrecommendedimages.jsp">View Recommended Images</a></li>
                        <li><a href="viewfriendrequest.jsp">View All Friends Request</a></li>
                        <li><a href="index.html">Logout</a></li>
                      </ul>
                    </div>
                  </div>
                </div>

                <div class="col-3">
                  <iframe src="selectfriend.jsp" style="border:3px solid;" width="640" height="400"></iframe>
                </div>
              </div>
            </div>
          </section>
          <!-- Footer Section -->
          <footer>
            <div style="background-color:#000000; height:40px;"></div>
          </footer>
        </div>
      </div>
      </body>
      </html>
```

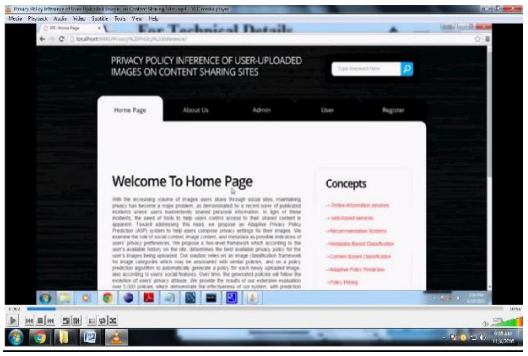
System Testing and Testing Strategies:

Software or hardware system testing is the process of testing an entire, integrated system to determine whether it satisfies its requirements. As system verification falls under the category of black-box testing, it does not require knowledge of the internal mechanics of the code or logic. The software system, along with any relevant hardware or systems, and all 'integrated' software elements that have successfully passed integration testing, are generally included in the system testing input. The objective of integration testing is to identify any inconsistencies between the software units that are combined (termed assemblages) or between any of the assemblages and the hardware. System testing is a more targeted form of testing that seeks to uncover defects both within the overall system and within the 'inter-assemblages.

**SCR.1 Home Page**

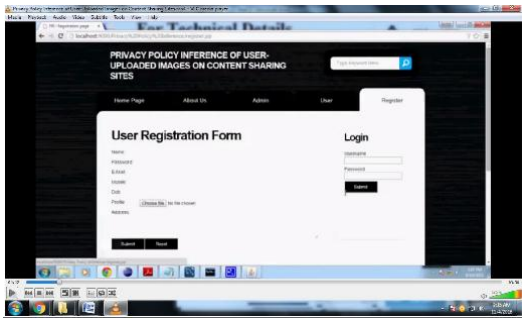



**SCR.2 Admin Page**

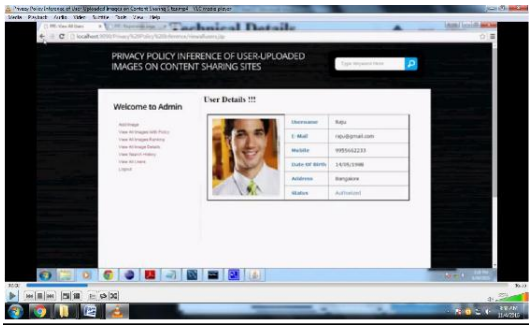



**SCR.3. User Page:**

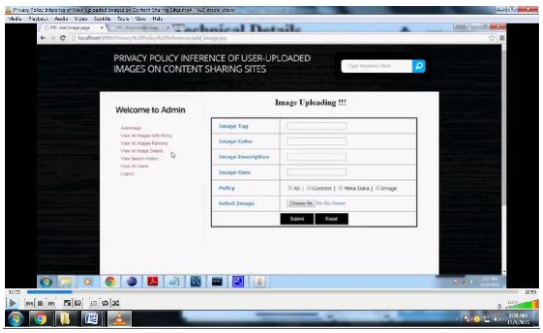

**SCR.4. Registration Page:**



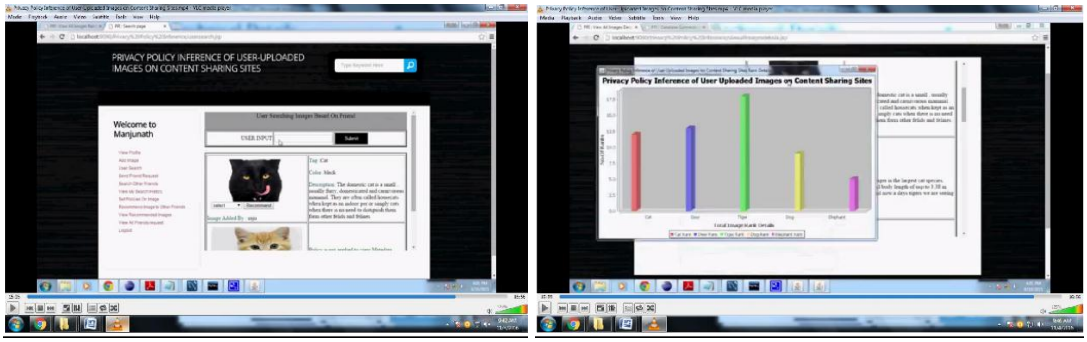

**SCR.5. Image Uploading Page:**




**SCR.6. User Details Page:**




**SCR.7.User Searching Image Based On Friend Page And Rank:**




## Conclusions

In this paper, we introduced the Adaptive Privacy Policy Prediction (A3P) system, designed to help users automate the privacy settings for their uploaded images. The A3P system establishes a robust framework to discern privacy preferences based on the information available for each user. Additionally, we effectively addressed the cold-start challenge by utilizing social context information. Our experimental results demonstrate that the A3P is a valuable tool, significantly enhancing the current methods for managing privacy.

## References

[1] Acquisti and R. Gross, "Imagined communities: Awareness, information sharing, and privacy on the facebook," in Proc. 6th Int. Conf. Privacy Enhancing Technol. Workshop, 2006, pp. 36–58.

[2] R. Agrawal and R. Srikant,"Fast algorithms for mining association rules in large databases," in Proc. 20th Int. Conf. Very Large Data Bases, 1994, pp. 487–499.

[3] S. Ahern, D. Eckles, N. S. Good, S. King, M. Naaman, and R. Nair, "Over-exposed?: Privacy patterns and considerations in online and mobile photo sharing," in Proc. Conf. Human Factors Comput. Syst., 2007, pp. 357–366.

[4] M. Ames and M. Naaman, "Why we tag: Motivations for annotation in mobile and online media," in Proc. Conf. Human Factors Comput. Syst., 2007, pp. 971–980.

[5] Besmer and H. Lip ford, "Tagged photos: Concerns, perceptions, and protections," in Proc. 27th Int. Conf. Extended Abstracts Human Factors Comput. Syst., 2009, pp. 4585–4590.

[6] D. G. Altman and J. M. Bland ,"Multiple significance tests: The bonferroni method," Brit. Med. J., vol. 310, no. 6973, 1995.

[7] J. Bonneau, J. Anderson, and L. Church, "Privacy suites: Shared privacy for social networks," in Proc. Symp. Usable Privacy Security, 2009.

[8] J. Bonneau, J. Anderson, and G. Danezis, "Prying data out of a social network," in Proc. Int. Conf. Adv. Soc. Netw. Anal. Mining., 2009, pp.249–254.

[9] H.-M. Chen, M.-H. Chang, P.-C. Chang, M.-C. Tien, W. H. Hsu, and J.-L. Wu, "Sheepdog: Group and tag recommendation for flickr photos by automatic search-based learning,"inProc.16th ACM Int. Conf. Multimedia, 2008, pp. 737–740.

[10] M. D. Choudhury, H. Sundaram, Y.-R. Lin, A. John, and D. D. Seligmann, "Connecting content to community in social media via image content, user tags and user communication," in Proc. IEEE Int. Conf. Multimedia Expo, 2009, pp.1238–1241.

[11] L. Church, J. Anderson, J. Bonneau, and F. Stajano, "Privacy stories: Confidence on privacy behaviors through end user programming," in Proc. 5th Symp. Usable Privacy Security, 2009.

[12] R. da Silva Torres and A. Falc~ao, "Content-based image retrieval: Theoryand applications," Revista de Inform_atica Te_orica e Aplicada, vol. 2, no. 13, pp. 161–185, 2006.

[13] R. Datta, D. Joshi, J. Li, and J. Wang, "Image retrieval: Ideas, influences, and trends of the new age," ACM Comput. Surv., vol. 40, no. 2, p. 5, 2008.

[14] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categoriestellus?"inProc.11thEur.Conf.Comput.Vis.:PartV,2010,pp.71–84.[Online]. Available:http:// portal.acm.org/citation.cfm?id=1888150.1888157

[15] Kapadia, F. Adu-Oppong, C. K. Gardiner, and P. P. Tsang, "Social circles: Tackling privacy in social networks," in Proc. Symp. Usable Privacy Security, 2008.