****

**REPUBLIC OF CAMEROON**

PEACE-WORK-FATHERLAND

THE UNIVERSITY OF BUEA

P.O Box 63,

Buea,South West Region

Cameroon

Tel: (237) 674354327

Fax: (237) 3332 22 72

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**Department of Computer Engineering**

**Specialty: Software Engineering**

**MODELING AND DESIGN OF A MOBILE APPLICATION FOR ARCHIVAL AND RETRIEVAL OF MISSING OBJECTS VIA IMAGE MATCHING**

**Presented by**

**GROUP 21**

|  |  |  |
| --- | --- | --- |
| S\N | NAME | MATRICULE |
| 1 | CHOMOH SYLVIA FONCHENALLAH | FE21A162 |
| 2 | AKENGNI KEANLI EMMANUEL | FE21A132 |
| 3 | MBI AYAMBA DIANNA | FE21A230 |
| 4 | NGAUEN NDJOMOU LOICE VANELLE | FE21A258 |
| 5 | NGOBA STEVE JONES NTONSG | FE21A259 |

**Dr. Eng. NKAMENI VALERIE**

**Course Supervisor**

**2rd Semester**

**2023/2024 Academic Year**

TABLE OF CONTENT

1. INTRODUCTION----------------------------------------------------------------------
2. SYSTEMDESIGN---------------------------------------------------------------------   
   2.1. UML DIAGRAMS-------------------------------------------------------------------

2.1.1 USE CASE DIAGRAM----------------------------------------------------------   
2.1.2 CLASS DIAGRAM

2.1.3 SEQUENCE DIAGRAM---------------------------------------------------------   
2.1.3 CONTEXT DIAGRAM---------------------------------------------------------   
2.1.4 DEPLOYMEMT DIAGRAM---------------------------------------------------

1. CONCLUSION------------------------------------------------------------------

* 1. INTRODUCTION

Modeling and design in software development encompass the processes of conceptualizing, planning, and structuring a software system before its implementation.

Modeling involves understanding the requirements and defining the system's architecture, behavior, and data flow through various diagrams and descriptions.

Design entails translating these models into detailed plans and specifications for the system's components, interfaces, algorithms, and user interface, ensuring that the final product meets the needs and expectations of its users.

* 1. SYSTEM DESIGN

In the context of design, Unified Modeling Language (UML) diagrams serve as valuable tools for visualizing and specifying various aspects of the software system being developed. Here's how UML diagrams can be referred to within the design phase:

− Behavioral Design:

Behavioral design in the context of a mobile-based archival and retrieval system for missing items using image matching technologies involves designing the user interface and experience to encourage desired behaviors from users. In this specific scenario, it entails creating an interface that guides users through the process of archiving and retrieving missing items in a way that maximizes efficiency and effectiveness. The bahavioral design diagrams we are going to use here is the Use Case Diagram and the Sequence Diagram.

− Structural Design:

Structural design in the context of a mobile-based archival and retrieval system for missing items using image matching technologies involves designing the underlying architecture and organization of the system. It focuses on how the components of the system are structured and how they interact with each other to achieve the desired functionality. Example Class Diagram and Deployment Diagram.

− Architectural Design:

Architectural design is a crucial phase in the software development lifecycle where the structure, behavior, and interactions of a system are defined at a high level. It focuses on creating a blueprint or framework that guides the construction of the software solution. Here key architectural design diagram we will use is Context diagram.

2.1 UML (UNIFIED MODELLING LANGUAGE) DIAGRAMS

2.1.1 USE CASE DIAGRAM

In a mobile-based archival and retrieval system for missing items using image matching technologies, there are several actors involved, each with specific roles. Here are the main actors and their roles:

* User:

The primary actor who interacts with the mobile application to perform various actions related to archiving and retrieving missing items. Users capture images of missing items, search for items, view found items, add new items to the database, update item details, and mark items as found.

* Administrator:

Responsible for managing the system, including user accounts, database maintenance, and handling escalated issues.

* Image Processing Module:

This actor represents the module responsible for processing images captured by users. It analyzes images using image matching algorithms to identify potential matches with archived items in the database.

* Database:

The database actor stores information about missing items, including images, descriptions, locations, and status (e.g., found or still missing). It provides data storage and retrieval functionality for the system.

* External Services:

Role: These actors represent any external services that the system will integrate with, such as GPS or mapping services for providing locationbased information about missing or found items, notification services for sending alerts to users, or authentication services for user login and access control.

These actors work together to enable the functionalities of the system, allowing users to archive and retrieve missing items efficiently using image matching technologies. Below is a clear diagramatical representation.

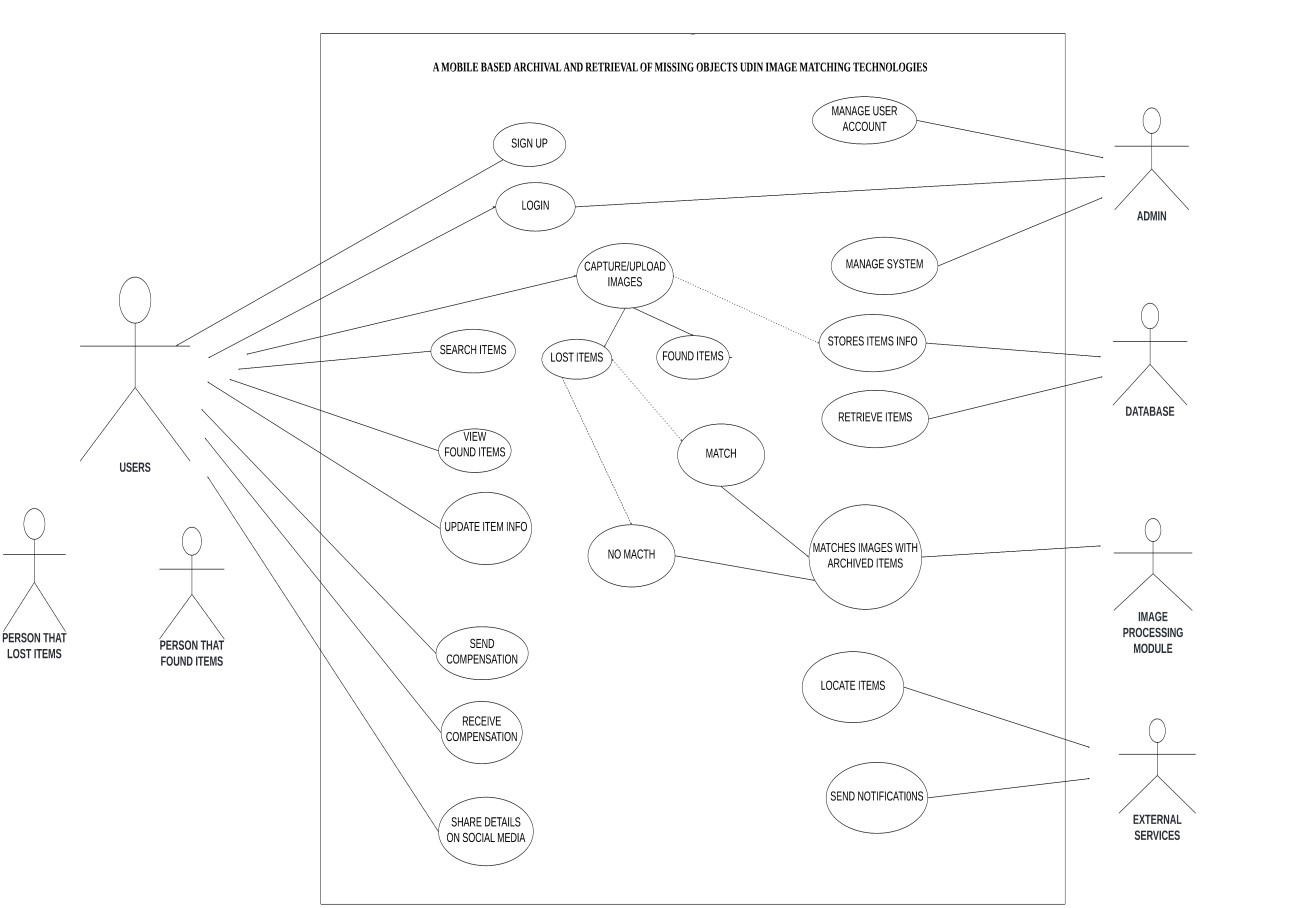


Figure 1: USE CASE DIAGRAM

**2.1.2 CLASS DIAGRAM**

This is the diagram that shows the interaction amongst different classes in the use case diagram.

For our system, the different classes, their methods and attributes are listed below:

1. **User class**:

* Attributes:AccountID,email,password,phoneNum,Username

1. **ItemFinder Class**:
   * Attributes: finderID, itemID, userID (ID of the finder).
   * Methods: claimItem().
   * Represents the person who finds the item and claims it.
2. **Payment Class**:
   * Attributes: paymentID, amount, senderID (ID of the user who lost the item), receiverID (ID of the user who found the item), timestamp.
   * Methods: processPayment().
   * Manages the payment transaction between the user who lost the item and the person who found it.

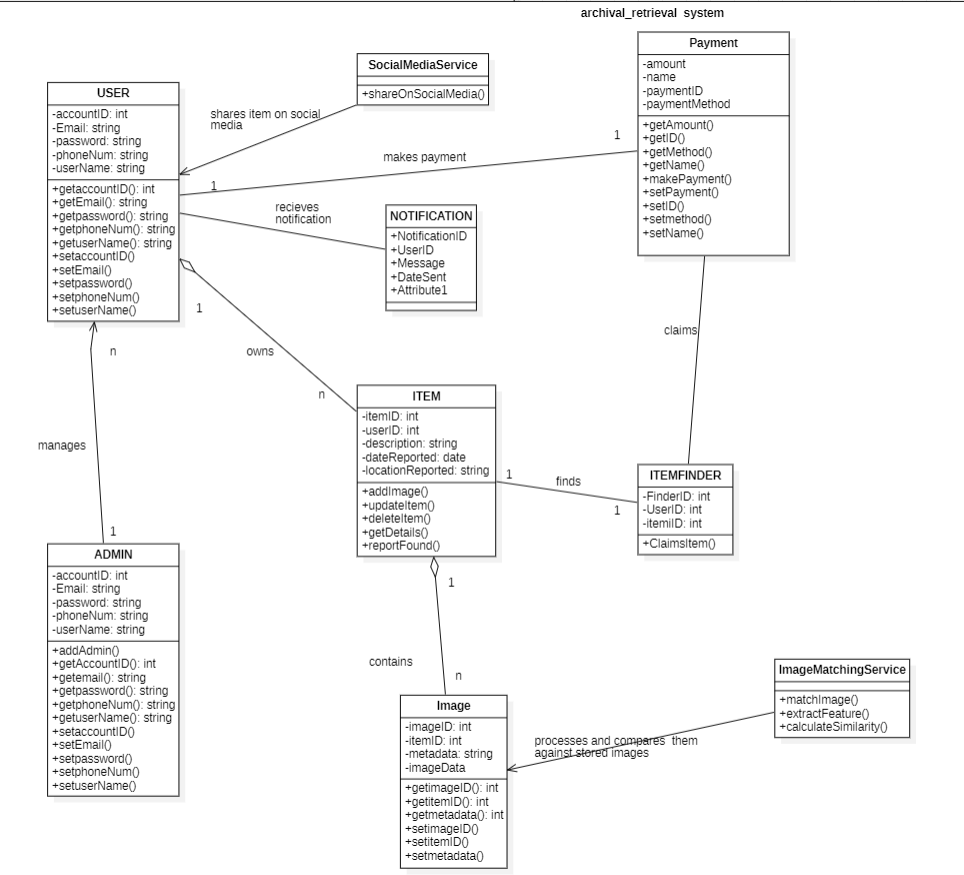
* **Item Class**:
  + Attributes: itemID, userID, description, timestamp, location.
  + Methods: addImage(), updateItem(), deleteItem(), getDetails().
  + Represents the items uploaded by users.

1. **Image Class**:
   * Attributes: imageID, itemID, imageData, metadata.
   * Methods: uploadImage(), processImage().
   * Handles image-related functionalities for items.
2. **ImageMatchingService Class**:
   * Methods: matchImage(), extractFeatures(), calculateSimilarity().
   * Provides image matching functionality using algorithms.
3. **NotificationService Class**:
   * Methods:sendNotification(),sendEmail(),sendPushNotification().
   * Manages user notifications for potential matches.
4. **SocialMediaService Class**:
   * Methods: shareOnSocialMedia().
   * Handles sharing of lost item details on social media platforms.

**Relationships**

1. **User ↔ Item**:
   * **One-to-Many**: A user can upload multiple items.
2. **Item ↔ Image**:
   * **One-to-Many**: An item can have multiple images.
3. **ImageMatchingService ↔ Image**:
   * **Uses Relationship**: ImageMatchingService uses images to perform matching.
4. **NotificationService ↔ User**:
   * **Association Relationship**: NotificationService sends notifications to users.
5. **SocialMediaService ↔ User & Item**:
   * **Association Relationship**: SocialMediaService allows users to share items on social media.
6. **ItemFinder ↔ Item**:
   * **One-to-One**: Each item that is found can have one associated finder.
   * **Association Relationship**: When an item is reported found, it is linked to the finder through ItemFinder.
7. **Payment ↔ User**:
   * **Association Relationship**: Payments are made from the user who lost the item to the user who found it.
   * **One-to-One**: Each payment transaction is associated with a single sender (lost item user) and a single receiver (finder).
8. **Admin-user**

* **One-to-One**: Manages user account



2.1.2 SEQUENCE DIAGRAM

A sequence diagram for this project illustrates the interaction between various components (users , mobile application , centralized database and notification system) over time to accomplish the functionalities of archiving and retrieving missing objects using image matching.

**2.1.2.1 Description of each component**

* **User:**

The individual who uses the application to Signup , Login upload images , search for lost items make payment and receive payment.

* **Mobile Application:**

The interface through which the user interacts with the system.

* **Centralized Database and Notification System:**

The storage system to store images of loss items , information about that image , information about the user and notify the users about potential matches information

2.1.2.2 Detailed Steps in the Sequence Diagram:

**Step 1 :** The user signup by provide information like (name , password ,phone number and email ) if it’s not done

**Step 2 :** The information is store in the data base and send the confirmation to the application

**Step 3 :** The Mobile app ask to the user to login into the App

**Step 4 :** The Users login into the App

**Step 5 :** The App check for matching information

**Step 6 :** Here if there the information matches then the The App will provide the main interface of the App to the user else the App will return the message “ Login faille ” to the user.

**Step 7**: The user uploads an image and metadata (description, location, timestamp ) of found object and it personal information for him to receive money form the mobile App (Necessary account information) through the mobile application.

**Step 8**: The mobile application sends the image and metadata to the centralized database for storage.

**Step 9:** The centralized database stores the image and metadata and sends a confirmation back to the mobile application.

**Step 10**: The user uploads an image and metadata (description, location, timestamp) of a missing object through the mobile application.

**Step 11:** The mobile application sends a search request to the centralized database.

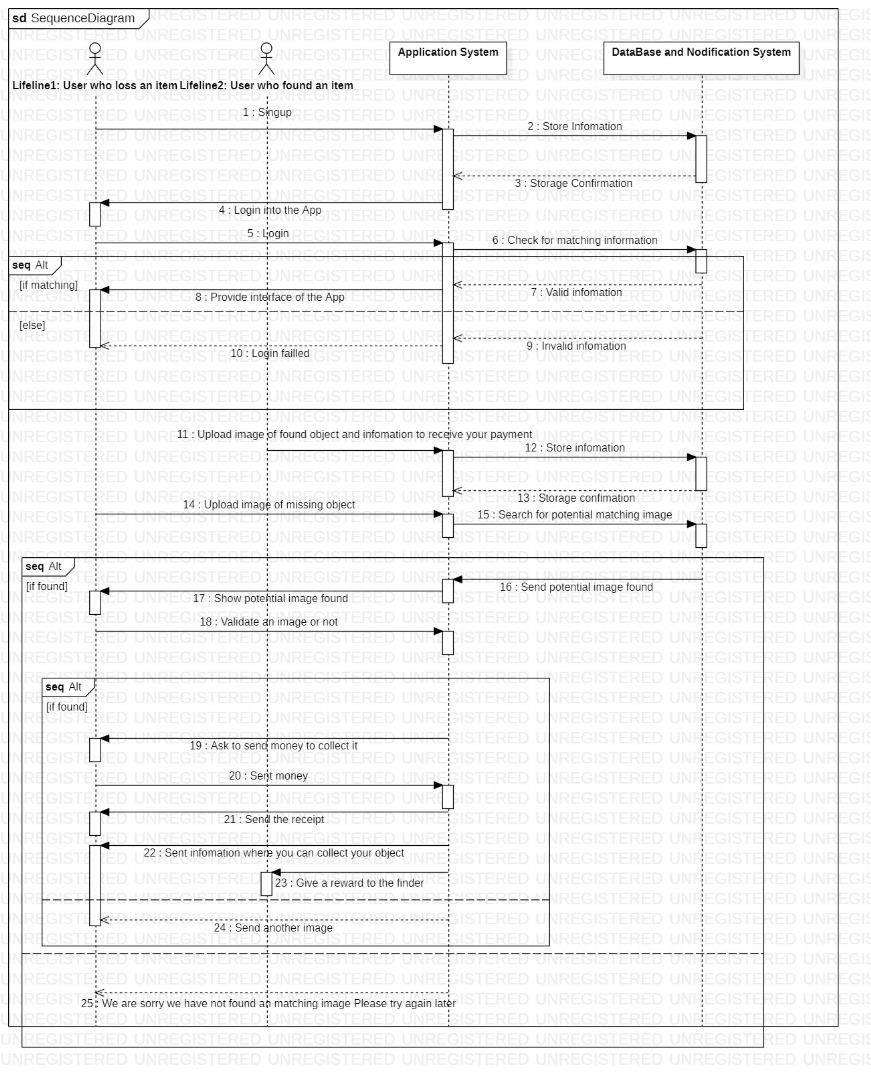
**Step 12:** The centralized database uses image matching algorithms to analyze the submitted image and search for potential matches.

**Step 13:** If matches are found, the centralized database sends the results back to the mobile application If the user validate an image then mobile App will ask him to send an amount money and when it’s ,he will receive the receipt and the information where he can collect it item by the app and the App will give a reward to the finder else the App will ask to heim to send another image. In the case if matches are not found , The mobile App will send the message “We are sorry we have not found am matching image please try later”.

A context diagram is a high-level visualization that illustrates the interactions between a system, its components, and external entities or systems. It provides a broad overview of what the system does and how it interacts with its environment without delving into the internal workings of the system.

2.1.3

CONTEXT DIAGRAM



# 2.3.1 Actors and roles ▪ User

Uploads images of lost or found items and Searches for matches and receives notifications about potential matches.

# ▪ Admin

Manages the lost and found database and Reviews and verifies matches, and sends notifications to users.

# ▪ Notification System

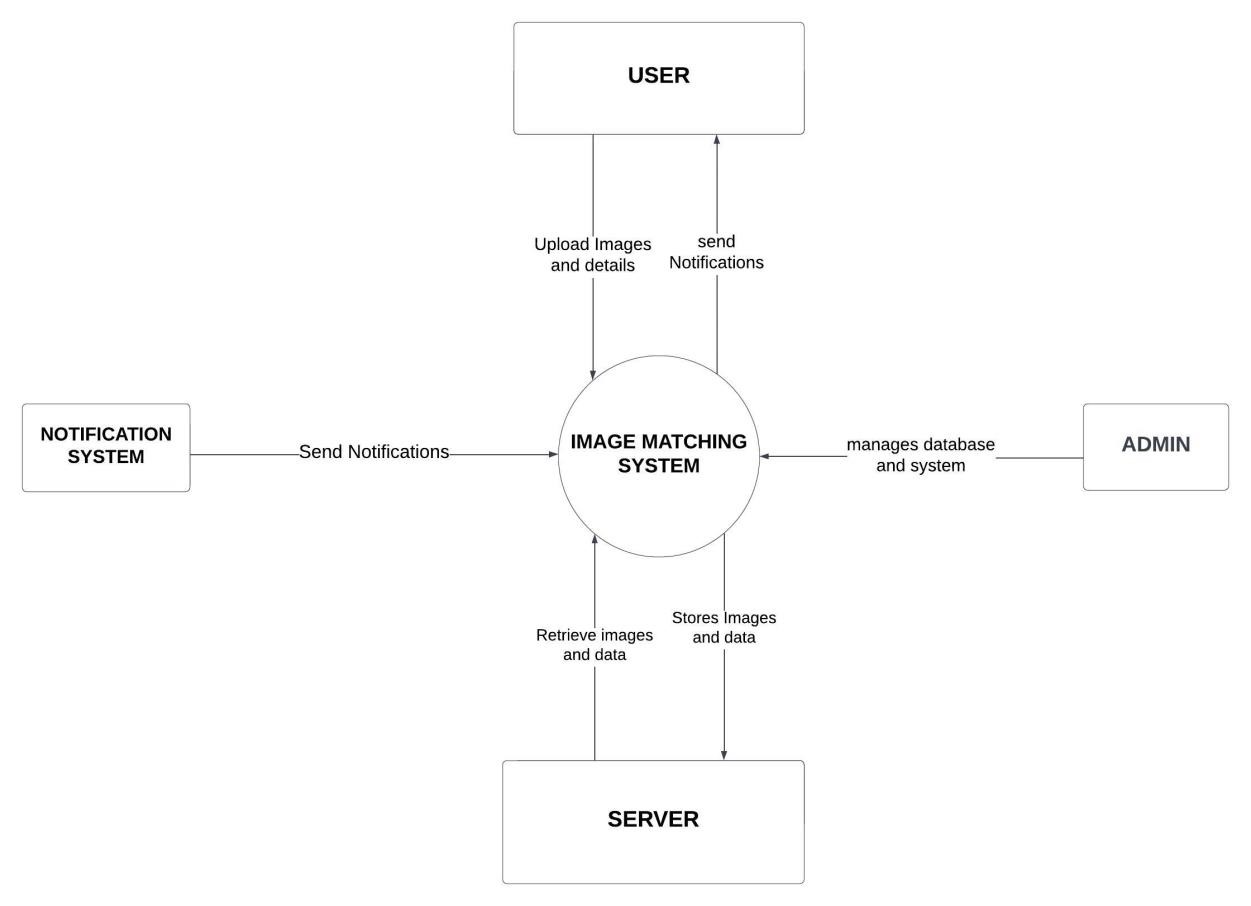
Sends alerts and updates to users and admins and Manages communication channels (e.g., email, SMS).

# ▪ Server

Allow retrieval of the images and data and Stores system data securely and ensures system performance.

# ▪ Image Matching System

Manages data flow and updates the database and Receive notifications from the system



2.1.4 DEPLOYMEMT DIAGRAM

A deployment diagram is a type of UML (Unified Modeling Language) diagram that shows the physical deployment of artifacts (software components, libraries, etc.) on nodes (hardware or software execution environments). It provides a highlevel overview of the system's hardware topology and the placement of software components, illustrating how software components are distributed across the hardware infrastructure.

## 2.1.4.1 Key Components and Nodes

* Mobile Application: The client application installed on users' mobile devices.
* Backend Server: Handles requests from the mobile application, processes image matching, and manages data storage.
* Database Server: Stores archived images and metadata.
* Image Processing/Matching Service: A specialized service or library for image matching algorithms.
* Cloud Storage: For storing large volumes of images and data.
* Notification Service: For sending notifications to users (e.g., when a match is found).

## 2.1.4.2 Define the Nodes

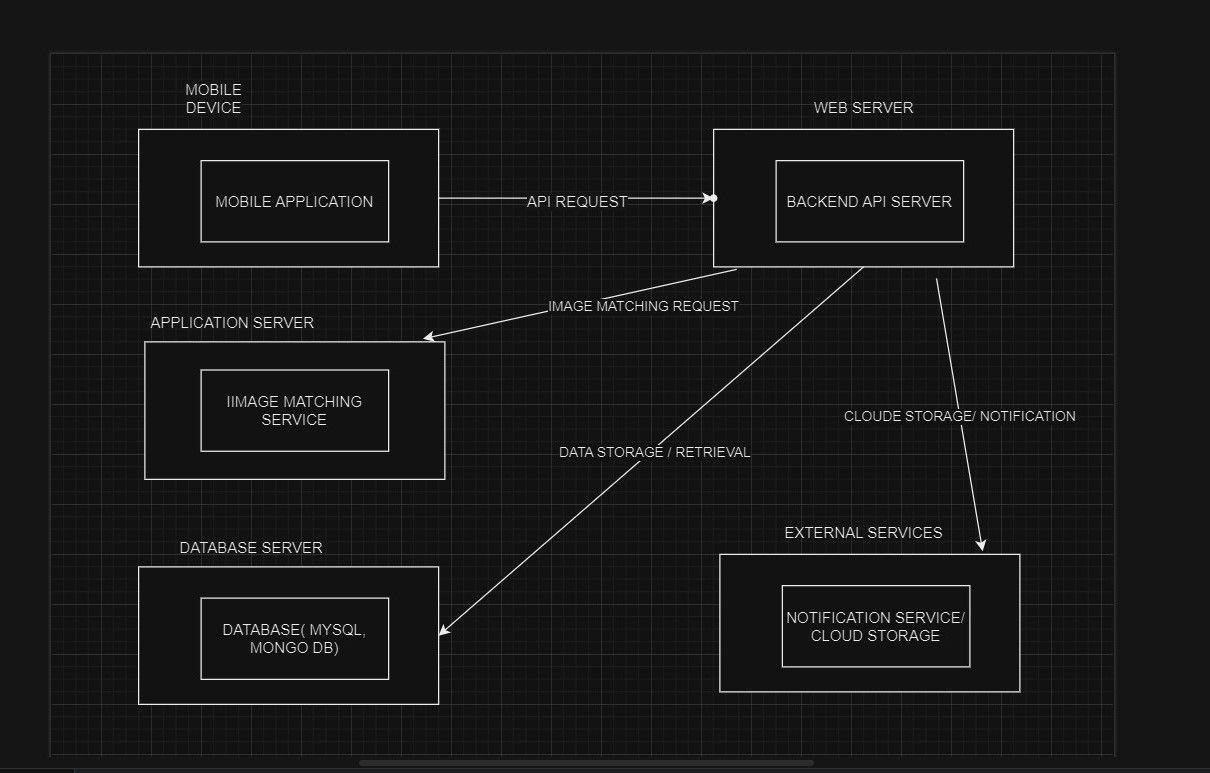
Mobile Device Node: Represents the user's mobile device running the mobile application.

Web Server Node: The backend API server that communicates with the mobile app, receiving image uploads and retrieval requests.

Application Server Node: Contains the image processing and matching logic, receiving images from the backend server for matching.

Database Server Node: Manages the storage of images and metadata.

External Services Node: Optional components for additional functionalities like cloud storage and notifications.



By creating a deployment diagram like this, we can effectively visualize the architecture of our mobile application, understand the interactions between different components, and ensure that all necessary elements are included for the archival and retrieval of missing objects via image matching.

With connections illustrating how the web server communicates with the application server, and how the application server communicates with the database server.

3. CONCLUSION

In conclusion, modeling and design are indispensable stages in the development lifecycle of any system or product. Through meticulous analysis, abstraction, and representation, modeling allows stakeholders to visualize, communicate, and refine the requirements and structure of the system. Design, on the other hand, transforms these conceptual models into concrete specifications, architectures, and plans for implementation.

Effective modeling and design practices facilitate collaboration among stakeholders, mitigate risks, and ensure that the final product aligns with the intended objectives and user needs. By iteratively refining models and designs based on feedback and evolving requirements, teams can adapt to changes and uncertainties, ultimately delivering high-quality solutions that meet or exceed expectations.