

## TABLE OF CONTENTS

<b><u>Topics</u></b>	<b><u>Page No.</u></b>
1. Abstract	02
2. Introduction	03
3. System Requirements	06
4. Methodology	10
5. System Design and Specifications	23
6. Testing	33
7. Output	39
8. Conclusion	47
9. Limitations and Future Enhancements	48
10. Bibliography	50

# 1. ABSTRACT

Every day in our society, there are children, youth, young women, the mentally handicapped, the elderly with dementia, etc. Countless people go missing. Although the police department filed a lawsuit against them. Often times they are very difficult to find. According to the current system, if a person is found missing, we must report his whereabouts to the nearest police station. After the complaint, the police will start an investigation by obtaining the necessary information.

This is a time-consuming process that requires a lot of effort. For this reason, we have prepared a project called "Missing Person Detection System" to make our work easier. We will create a web application that can send missing persons information and store it in the database. If the disappearance of a missing person is caught on CCTV, the system can use facial recognition algorithms to capture it. When the system recognizes a match, it creates custom alerts and locations and sends them directly to family members and interested researchers.

The "Missing Person Detection System" project introduces a transformative web-based application designed to revolutionize the search for missing individuals by integrating machine learning and facial recognition techniques. The project's primary objective is to expedite the search process by enabling users to upload images of missing persons and suspicious individuals, which are then cross-referenced against a comprehensive database. Upon finding a match, the system generates notifications to inform relevant authorities and concerned individuals about the located person's whereabouts.

The project comprises distinct modules that contribute to its functionality. Users, including family members and law enforcement, can upload images of missing persons, accompanied by relevant information, through the "Image Upload" module. The "Face Recognition" module employs machine learning algorithms to extract and compare facial features, while the "Database Management" module efficiently organizes and retrieves information from the image repository.

Upon identifying a match, the system generates alerts through the "Alert Generation" module, notifying authorities and concerned parties via email. The project's future scope includes the integration of real-time video analysis, allowing for the detection of missing persons using CCTV cameras and video surveillance. Additionally, a mobile application version is envisioned, expanding accessibility and usability.

The technology stack employed in the project includes Python as the core programming language, encompassing image processing and machine learning tasks. Machine learning libraries such as TensorFlow, or OpenCV are utilized for training the facial recognition model. The web application is built using the Django framework, providing tools for image uploads, database management, and user interfaces.

Future development avenues involve enhancing facial recognition accuracy through advanced algorithms and deep learning techniques. The incorporation of real-time video analysis and the creation of a mobile application aim to broaden the system's reach and effectiveness. Collaboration with law enforcement agencies is envisioned to integrate the system into existing databases and investigative processes.

## **2. INTRODUCTION**

In a world driven by technological innovation and a growing need for advanced solutions to societal challenges, the "Missing Person Detection System" emerges as a transformative project at the crossroads of compassion and cutting-edge technology. This initiative is motivated by the critical and time-sensitive nature of locating missing individuals, aiming to redefine the conventional search and rescue paradigm. By integrating machine learning and facial recognition technologies into a web-based application, this project seeks not only to expedite the search process but also to enhance the precision and efficiency of identifying missing persons.

This led us to prepare a project called "Bharatiya Rescue", which facilitates our work number. The main purpose of "Missing Person Detection System" is to find missing persons using CCTV real-time video via face recognition and send report to police station with parking spots in newspaper. also allows ordinary people to upload pictures of strangers. If the complaint number has already been written about the same person on the portal, it will notify the police

The "Missing Person Detection System" emerges as a groundbreaking initiative at the intersection of technology and societal well-being. In a world where the timely discovery of missing individuals is paramount, this project aims to transcend traditional search methods. The purpose is clear: to create an advanced, user-friendly web-based application that not only expedites the search process but also maximizes the accuracy of identifying missing persons through the integration of machine learning and facial recognition technologies.

Beyond its immediate functionalities, the project embraces an expansive scope, recognizing the dynamic nature of technological advancements. The incorporation of advanced facial recognition algorithms and machine learning libraries positions the system as a versatile and adaptive tool. Moreover, the forward-looking scope includes potential integrations such as real-time video analysis and mobile applications, reflecting a commitment to staying at the forefront of technological innovation.

In a world driven by technological innovation and a growing need for advanced solutions to societal challenges, the "Missing Person Detection System" emerges as a transformative project at the crossroads of compassion and cutting-edge technology. This initiative is motivated by the critical and time-sensitive nature of locating missing individuals, aiming to redefine the conventional search and rescue paradigm. By integrating machine learning and facial recognition technologies into a web-based application, this project seeks not only to expedite the search process but also to enhance the precision and efficiency of identifying missing persons.

### **2.1 PROBLEM STATEMENT**

The process of identifying a missing person often involves comparing available information about a person (such as physical characteristics, clothing, and last known location) with records from various sources (such as social media, surveillance footage, and missing persons files). This process requires extensive and complex data analysis, which is difficult and time consuming the

main purpose of our work is to create an application that helps to improve the process of finding missing persons. We focus on helping users and persons suffering from missing persons, those who may or may not be close to them. The main purpose of this study is to use Haar Cascade, an uncomplicated real-time algorithm. The purpose of this algorithm is to identify objects, but in our case, we encounter.

## **2.2 PROJECT PURPOSE**

The "Missing Person Detection System" is conceived with a singular and crucial purpose — to revolutionize the search and retrieval process for missing individuals. In response to the challenges posed by traditional methods, this project seeks to harness the power of advanced technologies, specifically machine learning and facial recognition, to create a dynamic web-based application. The primary aim is to expedite and enhance the accuracy of locating missing persons, ultimately contributing to public safety and the well-being of communities.

## **2.3 PRODUCT SCOPE**

The scope of the project encompasses the development of a comprehensive web-based application that empowers users, including family members and law enforcement agencies, to actively participate in the search for missing persons. The project extends its reach beyond mere image uploads by incorporating advanced facial recognition algorithms, machine learning libraries, and database management systems. The envisaged scope includes not only the current functionalities but also embraces future enhancements such as real-time video analysis and mobile application integration. This expansive scope positions the "Missing Person Detection System" as a versatile tool adaptable to the evolving landscape of technology and societal needs.

## **2.4 PRODUCT FEATURES**

The "Missing Person Detection System" boasts an array of distinctive features designed to optimize the search process and maximize user accessibility. The key product features include:

### **1. Image Upload Module:**

- Users, whether they be family members or law enforcement officials, can seamlessly upload images of missing persons along with relevant information.

### **2. Facial Recognition:**

- Leveraging advanced machine learning libraries such as TensorFlow and OpenCV, the system extracts facial features and performs precise face encodings to enhance accuracy during the matching process.

### **3. Database Management:**

- The system maintains an organized and efficient database of images and associated information, facilitating rapid retrieval and comparison during facial recognition.

#### **4. Alert Generation:**

- Upon identifying a match between an uploaded image and a stored image, the system generates alerts. These notifications are sent to relevant authorities and concerned individuals via email, providing vital details about the located person's whereabouts.

#### **5. Technology Stack:**

- The system is developed using Python as the core programming language, supported by machine learning libraries like face-recognition and OpenCV. The web application is built on the Django framework, providing a robust foundation for efficient image processing, database management, and user interface design.

### **3. SYSTEM REQUIREMENTS**

The system requirements for the "Missing Person Detection System" are instrumental in translating the project's vision into a tangible and functional application. They define the parameters within which the system will operate, ensuring it meets the needs and expectations of its users. The requirements not only serve as a technical guide for developers but also align closely with the project's overarching goals, such as accelerating the search for missing individuals and enhancing collaboration with law enforcement agencies.

These system requirements act as a roadmap, providing a clear path for development, implementation, and maintenance. They set the standards for the technology stack, ensuring compatibility and efficiency. Additionally, the requirements act as a foundation for future enhancements, allowing for seamless integration of advanced features like real-time video analysis and mobile applications.

#### **3.1 HARDWARE REQUIREMENTS:**

##### **1. Server:**

- High-performance server for hosting the web application.
- Sufficient processing power to handle image processing and machine learning tasks.
- Adequate storage capacity for the image database.
- In this project we have used our local machine with as server.

##### **2. Storage Devices:**

- Large-capacity storage devices for storing images and associated data.
- Redundancy measures (e.g., RAID) to ensure data integrity and availability.
- In this project I have used my SSD (512 GB) for storing the data and hosting the server.

##### **3. Networking:**

- Reliable and high-speed internet connection for seamless image uploads and data retrieval.
- Network security measures to protect against unauthorized access.

##### **4. Other Requirements:**

- **RAM** 8 GB or more.
- **CPU** 2.5 GHz.
- **Architecture** 64 bit.
- **GPU:** 4 GB

## 3.2 SOFTWARE REQUIREMENTS:

### 1. Operating System:

- Linux-based server operating system for hosting the web application.
- Compatibility with major operating systems for end-users (Windows, macOS, Linux).
- We have used Windows 11 device.

### 2. Web Server:

- Deployment of a web server (e.g., Apache, Nginx) to serve the web application.
- Configuration for secure data transmission (HTTPS).

### 3. Database Management System:

- Database management system (e.g., PostgreSQL, MySQL) for storing image data.
- Efficient indexing and query optimization for quick data retrieval.
- We have used SQLITE database for storing our data.

### 4. Programming Language:

- Python as the core programming language for system development.

### 5. Web Framework:

- Django web framework for building the web application.
- Django's ORM for database interactions.

### 6. Machine Learning Libraries:

- Integration of machine learning libraries such as face-recognition and OpenCV for facial recognition.
- Compatibility with the selected libraries for training and inference.

### 7. Version Control:

- Version control system (e.g., Git) for tracking changes to the codebase.
- Our entire codebase is available in GitHub for learning and collaborative purposes.

### 8. Development Tools:

- Integrated Development Environment (IDE) for coding and debugging (e.g., Visual Studio Code, PyCharm).

### 9. Dependency Management:

- Utilization of package management tools (e.g., pip) for handling Python dependencies.

### 10. Email Services:

- Integration with email services for sending alerts (e.g., SMTP) we have used GMAIL SMTP tool for demonstration purposes.

#### **11. Future Integration Tools (if applicable):**

- Tools and libraries for video analysis integration, mobile application development, and collaboration with law enforcement databases.

### **3.3 FUNCTIONAL REQUIREMENTS:**

#### **1. User Authentication:**

- Users should be able to create accounts with unique usernames and passwords.
- Different user roles (e.g., family members, police) with varying levels of access should be implemented.

#### **2. Image Upload Module:**

- Users must be able to upload images of missing persons with accompanying information.
- Supported image formats should be clearly defined (e.g., JPEG, PNG).
- There should be restrictions on image size to optimize system performance.

#### **3. Face Recognition:**

- The system should employ machine learning algorithms for facial recognition.
- Extracted facial features should be compared against stored images for potential matches.
- Face encodings should be used for accurate matching.
- The system must provide a confidence score for each match.

#### **4. Database Management:**

- The system should store and manage a database of images along with associated information.
- Efficient indexing and retrieval mechanisms should be implemented for quick comparison during face recognition.

#### **5. Alert Generation:**

- When a match is found, the system should generate alerts.
- Alerts should include details about the missing person's location and relevant information.
- Notification methods, such as email, should be configurable.

#### **6. Video Integration (Future Scope):**

- The system should be designed to accommodate future integration with live video feeds for real-time analysis.



#### **7. Mobile Application (Future Scope):**

- If developed, the mobile application should have functionalities equivalent to the web-based system.
- Users should be able to upload images and receive alerts on their mobile devices.

### **3.4 NON-FUNCTIONAL REQUIREMENTS:**

#### **1. Performance:**

- The system is capable of handling a large number of image uploads and comparisons efficiently.
- Response times for face recognition and database queries meets acceptable standards.

#### **2. Security:**

- User data, especially images and personal information, are securely stored and transmitted.
- Access controls are to be in place to prevent unauthorized access to sensitive information.

#### **3. Scalability:**

- The system is designed to scale easily to accommodate a growing database and user base.

#### **4. Reliability:**

- The system is reliable, with minimal downtime for maintenance or upgrades.

#### **5. Usability:**

- The user interface is intuitive and user-friendly.
- Clear error messages and guidance is provided to users.

#### **6. Compatibility**

- The system should be compatible with commonly used web browsers and devices.
- Mobile applications, if developed, should support major operating systems.

#### **7. Future Expandability:**

- The system architecture should be designed to facilitate easy integration of future enhancements.
- Such as improved facial recognition algorithms and real-time video analysis.

## 4. METHODOLOGY

The development methodology of the "Missing Person Detection System" is a meticulously planned and comprehensive approach that amalgamates cutting-edge technologies to forge a potent tool dedicated to the expedited location of missing individuals. The primary objective of this project is to enhance and streamline the search process by incorporating sophisticated machine learning algorithms and facial recognition techniques seamlessly embedded within a web-based framework.

### 4.1 SYSTEM ARCHITECTURE:

The system's backbone is Python, chosen for its versatility and extensive support in various domains. Django, a high-level Python web framework, forms the well-structured foundation for the backend development. The integration of machine learning capabilities is achieved through the seamless collaboration of Django with OpenCV and the Face-Recognition library.

#### 1. Django:

- Django, a high-level web framework written in Python, played a pivotal role in the success of my project by providing a robust and efficient backend structure. Leveraging Django's Model-View-Controller (MVC) architectural pattern, we seamlessly integrated databases to store and manage various components of my project. The Django ORM (Object-Relational Mapping) simplified the interaction with databases, allowing me to define models and relationships using Python classes, and automatically handling the underlying SQL queries. This not only streamlined the development process but also ensured data consistency and integrity.
- Furthermore, Django's built-in authentication system, combined with its security features, facilitated the implementation of user authentication and access controls. This was crucial for securing sensitive information and functionalities, such as recognizing faces through OpenCV. The integration of OpenCV into Django allowed for real-time image processing and facial recognition capabilities. we utilized Django's views to manage the logic behind face recognition and seamlessly connected it with the frontend.
- Django's versatility was further demonstrated in handling email functionalities within the project. The framework provides a straightforward way to send emails using the built-in EmailMessage class, and by configuring the email settings in the Django project, I was able to send notifications and alerts, enhancing the user experience.

#### 2. Open CV:

- The integration of the OpenCV (Open-Source Computer Vision) library proved to be indispensable in enhancing the functionality and capabilities of my project.

OpenCV, an open-source computer vision and machine learning software library, provided a comprehensive set of tools and algorithms for image and video analysis. In the context of my project, OpenCV played a crucial role in facial recognition, image processing, and computer vision tasks.

- The primary utility of OpenCV in my project was evident in its robust facial recognition capabilities. Leveraging OpenCV's pre-trained deep learning models, I could efficiently detect and recognize faces in images and video streams. This capability was seamlessly integrated into the Django backend, allowing for secure and accurate identification of individuals, which was a core component of my project.
- Additionally, OpenCV facilitated real-time image processing, enabling dynamic adjustments and transformations to images as they were processed. This capability was particularly useful for preprocessing images before the facial recognition step, ensuring optimal conditions for accurate identification.
- The versatility of OpenCV extended beyond facial recognition, as it provided a suite of computer vision tools that could be applied to various aspects of my project. From image filtering and edge detection to contour analysis and feature extraction, OpenCV's rich set of functionalities empowered me to implement advanced image processing techniques seamlessly.
- In conclusion, OpenCV significantly enriched my project by providing powerful tools for facial recognition, real-time image processing, and a broad spectrum of computer vision functionalities. Its flexibility, robustness, and compatibility with Django made it an invaluable asset, enhancing the overall capabilities and user experience of the project.

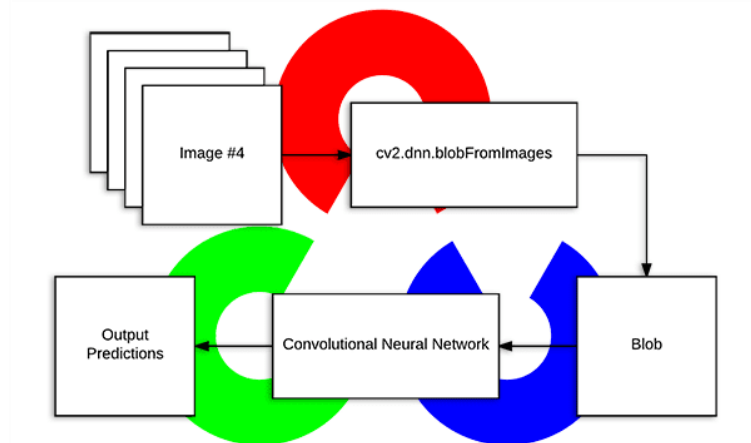
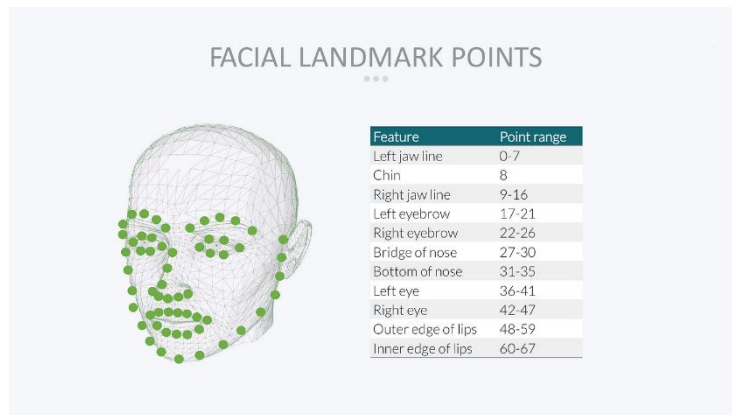


Figure 1 Working of OpenCV

### 3. Face-recognition library

- Face recognition is a critical component of many computer vision applications, and several libraries provide robust solutions for implementing this functionality. In your project, integrating a face recognition library would have enhanced the capabilities and user experience. While there are various face recognition libraries available, one popular choice is the "face\_recognition" library in Python. Here's an overview of its uses and how it might have assisted us in our project.
- **Facial Detection:**
  - The face\_recognition library uses pre-trained deep learning models to detect faces in images or video frames.
  - It identifies the location of faces within the given input and provides bounding box coordinates.
- **Facial Landmark Detection:**
  - Beyond face detection, the library can also detect facial landmarks, such as eyes, nose, and mouth.
  - Knowing these landmarks allows for more detailed analysis and manipulation of facial features.
- **Face Encoding:**
  - The library computes facial encodings, which are numerical representations of facial features.
  - These encodings serve as unique identifiers for different faces and are used for face matching and recognition.
- **Face Recognition:**
  - The primary purpose of the library is, as the name suggests, face recognition. It compares the facial encodings of known faces with those in the input data to identify individuals.
  - It can recognize faces in images, video streams, or even in real-time scenarios.
- **Accuracy and Performance:**
  - The face\_recognition library is known for its accuracy in identifying faces, even in challenging conditions.
  - It leverages the dlib library for its deep learning models, which are optimized for both accuracy and performance.
- **Integration with Databases:**
  - Face recognition is often used in conjunction with databases to associate recognized faces with relevant information.

- In your project, integrating face recognition with Django's database capabilities would have allowed for seamless storage and retrieval of information associated with recognized individuals.
- **Security and Access Control:**
  - Face recognition is widely used for security applications and access control systems.
  - In my project, this feature has been leveraged for secure authentication and authorization.
- **Customization and Scalability:**
  - The face\_recognition library is flexible and allows for the training of custom models for specific faces or features.
  - This customization feature can be particularly useful in projects where unique facial attributes or expressions need to be recognized.
- **Handling Large Datasets:**
  - For projects dealing with extensive datasets of faces, the face\_recognition library efficiently handles large-scale recognition tasks.
  - This scalability is advantageous when working with applications that involve a diverse range of individuals and require the recognition of numerous faces in real-time or batch processing.



*Figure 2 Facial Landmark Points*

## 4.2 USER INTERFACE AND FRONTEND:

The frontend development focuses on creating an intuitive and user-friendly interface using HTML, CSS, and JavaScript. With Bootstrap for making the website more attractive, this layer facilitates effective user interaction, incorporating features like image upload and dynamic data display through AJAX.

### 1. HTML:

- HTML, the foundational language of the World Wide Web, has been instrumental in shaping the structure and presentation of our group project's web application. As the backbone of web development, HTML provided the essential markup for creating the layout and content structure of our web pages. Utilizing its semantic elements, we collectively defined the hierarchy of information, from headers and paragraphs to lists and forms, ensuring a well-organized and accessible presentation.
- The use of HTML in our project extends beyond static content representation. Through seamless integration with Django, HTML templates were employed to dynamically render data retrieved from the backend, creating a dynamic and responsive user interface. This dynamic generation of content allowed for real-time updates, ensuring a user-friendly experience for our diverse group of users.
- Furthermore, HTML worked collaboratively with other technologies like CSS (Cascading Style Sheets) and JavaScript to enhance the visual appeal and interactivity of the application. CSS styles were applied to HTML elements collaboratively, ensuring consistent branding, layout design, and responsiveness across different devices. JavaScript, often collaboratively embedded within HTML, facilitated client-side interactions, enriching the user experience with features such as dynamic form validation and asynchronous data retrieval.
- In the context of the Django framework, HTML templates served as the collaborative bridge between the backend and frontend, seamlessly incorporating dynamic content from the server into the presentation layer. Django's template language, which integrates seamlessly with HTML, allowed for the collaborative inclusion of dynamic content, such as user-specific information and real-time updates from the backend.
- In summary, HTML, in collaboration with our group's collective efforts, played a pivotal role in our project by providing the fundamental structure for web pages and acting as the canvas upon which the dynamic content and interactivity were built.

## 2. CSS

- CSS (Cascading Style Sheets) has been a collaborative cornerstone in our group project, enhancing the visual appeal and user interface design of our web application. As a collaborative effort, CSS played a crucial role in ensuring a consistent and aesthetically pleasing presentation across the diverse set of pages we collectively developed. Collaboratively styling HTML elements, we maintained a unified brand identity, responsive layouts, and a visually engaging user experience.
- The collaborative integration of CSS with HTML within our Django framework allowed us to create dynamic and visually cohesive web pages. Working in tandem, our group applied CSS styles to HTML elements, ensuring a seamless and collaborative translation of design concepts into the final user interface. This collaborative styling approach extended beyond individual contributions, fostering a cohesive and harmonious visual language throughout the application.
- Moreover, the collaborative nature of CSS was evident in its adaptability across various devices, promoting a consistent user experience regardless of the platform. Our group worked collaboratively to implement responsive design principles, allowing the application to gracefully scale and adapt to different screen sizes and orientations.
- In addition to collaborating on the visual aspects, our group collaborated on implementing CSS transitions and animations, enriching the user experience with subtle and engaging interactions. This collaborative effort aimed to create a polished and intuitive interface that collectively met the diverse expectations and preferences of our users.
- In summary, CSS, as a collaborative tool in our group project, significantly contributed to the visual aesthetics and user interface of our web application. Through collaborative styling efforts, our group ensured a consistent and appealing design, responsive layouts, and an overall enhanced user experience. The collaborative integration of CSS within our Django-based project exemplifies its importance as a collaborative element in the development process, allowing our group to collectively create a visually cohesive and user-friendly web application.

## 3. JavaScript

- JavaScript, as a collaborative tool in our group project, has been instrumental in elevating the interactivity and functionality of our web application. Our collective efforts with JavaScript allowed us to implement dynamic features, real-time updates, and client-side interactions that enhance the overall user experience.

Collaboratively embedded within our HTML and Django framework, JavaScript played a key role in enabling asynchronous data retrieval, form validation, and other interactive elements that collectively contribute to a seamless and responsive application.

- One notable collaborative aspect of JavaScript in our project is its role in facilitating smooth communication between the frontend and backend, leveraging the capabilities of the Django framework. Through collaborative scripting, our group ensured that user interactions seamlessly translated into backend operations, providing a cohesive and efficient user experience.
- Collaboratively implementing JavaScript also allowed our group to enhance the security features of our application. We employed client-side validation, working together to ensure that data entered by users is validated in real-time, reducing the likelihood of errors and enhancing the overall reliability of the system.
- Furthermore, the collaborative integration of JavaScript was vital for creating interactive and engaging user interfaces. Our group worked collaboratively to implement features such as sliders, pop-ups, and dynamic content updates, enhancing the overall usability of the application. This collaborative approach to JavaScript coding allowed for a diverse set of functionalities, accommodating the various needs and preferences of our group members.
- In summary, JavaScript, as a collaborative element in our group project, significantly contributed to the interactivity, responsiveness, and overall functionality of our web application. Through collaborative scripting efforts, our group harnessed the power of JavaScript to create a dynamic and engaging user experience, seamlessly integrating it within the HTML and Django framework. The collaborative use of JavaScript reflects its importance as a key component in our group's development process, enabling us to collectively deliver a feature-rich and user-friendly web application.

#### **4. Bootstrap**

- Bootstrap, as a collaborative tool in our group project, has played a pivotal role in streamlining the design and responsiveness of our web application. Working collaboratively with Bootstrap, we leveraged its powerful grid system, pre-designed components, and responsive utilities to create a consistent and visually appealing user interface across our diverse set of pages. The collaborative adoption of Bootstrap's CSS framework allowed our group to achieve a cohesive design language and maintain a professional aesthetic throughout the application.



- Our collaborative use of Bootstrap's components facilitated rapid development and enhanced the overall consistency of our project. By utilizing its predefined navigation bars, modals, and form components, our group collectively reduced development time and ensured a cohesive and polished look and feel across different sections of the application.
- Collaboratively, we harnessed Bootstrap's responsiveness features to create a user-friendly experience on various devices and screen sizes. Our group worked together to ensure that the application's layout adapts seamlessly to different resolutions, providing a consistent and enjoyable experience for our users, regardless of the device they use.
- Additionally, the collaborative integration of Bootstrap's JavaScript components, such as dropdowns, modals, and carousels, allowed our group to enhance the interactive elements of the application. Collaborative customization of these components ensured that they seamlessly integrated with our project's requirements, providing a user interface that is not only visually appealing but also functionally rich.
- To sum up, our group's collaborative use of Bootstrap significantly expedited development, ensured design consistency, and contributed to a responsive and visually appealing user interface. This collaborative approach highlights the importance of Bootstrap in our group's development strategy, allowing us to collectively deliver a polished and user-friendly web application. The collaborative adoption of Bootstrap's framework showcases its importance in us
- group's development process, enabling us to collectively create a visually cohesive and user-friendly web application.

### **4.3 DATA MANAGEMENT:**

Efficient management of data is accomplished through Django's Object-Relational Mapping (ORM) capabilities. The system employs a relational database structure, ensuring quick and efficient retrieval and comparison of images and associated information during the facial recognition process. This ensures that the system operates with speed and accuracy in accessing crucial information.

#### **1. Django-ORM**

Django ORM, being an integral part of the Django web framework, enables us to interact with the database using Python code and classes, abstracting away the intricacies of raw SQL queries. This collaborative approach to data management ensured a standardized and maintainable way of handling database operations.

By utilizing Django ORM's model system, we could define data models using Python classes, establishing a direct mapping between our application's data structure and the underlying database schema. This collaborative approach to defining models facilitated better communication among team members, as the database schema was expressed in a language familiar to all Python developers in our group.

Moreover, Django ORM promotes the use of migrations, allowing for version control of the database schema. Collaboratively applying migrations ensured that database changes were systematically tracked and applied, eliminating potential conflicts and ensuring a synchronized development environment among our group members.

The collaborative integration of Django ORM in our project also enhanced data security through its built-in protection against SQL injection attacks. The ORM automatically escapes parameters in queries, reducing the risk of malicious database manipulations. This collaborative security measure ensured that our application maintained robust data integrity and protection against potential vulnerabilities.

In terms of productivity, the collaborative use of Django ORM significantly reduced the amount of boilerplate code traditionally associated with database interactions. The framework provided a high-level, Pythonic API for querying the database, fostering a collaborative and efficient development process.

To summarize, Django ORM has played a pivotal role in our project's methodology, providing a collaborative and efficient means of interacting with the database. Its collaborative features, including model definition, migrations, and security measures, have collectively contributed to a streamlined development process and ensured the integrity and efficiency of our data management system. The collaborative integration of Django ORM underscores its importance in our project methodology, enabling us to collectively build a robust and scalable web application.

## **2. SQLite Database**

SQLite, being a lightweight, embedded relational database management system, has proven to be a collaborative and efficient choice for our project's data storage needs. Its simplicity and ease of use have significantly contributed to the smooth functioning of our application.

The collaborative use of SQLite in our project allowed for seamless integration with the Django web framework. Django supports SQLite as one of its databases backends, and this collaborative decision simplified the setup process, enabling our team members to work with a consistent and easily replicable database environment.

The collaborative nature of SQLite was particularly beneficial during the development phase. Its serverless architecture eliminated the need for complex database setup procedures, fostering a collaborative and hassle-free environment for team members to collaborate on various aspects of the project. Additionally, SQLite's support for in-

memory databases facilitated efficient testing and debugging processes, allowing our group to collectively identify and resolve issues rapidly.

The collaborative integration of SQLite also contributed to the portability of our application. Since SQLite databases are self-contained and stored as a single file, sharing and deploying the database across different environments became a collaborative and straightforward process. This collaborative feature streamlined our development workflow, ensuring consistency across various development environments.

Moreover, the collaborative decision to use SQLite aligns with the scalability requirements of our project. While SQLite may be considered lightweight, its collaborative use within the Django framework provides the flexibility to transition to more robust database systems seamlessly if the project's scale demands it in the future.

In conclusion, the collaborative choice of SQLite as our database management system has played a key role in shaping the methodology of our project. Its lightweight nature, ease of use, and compatibility with Django have collectively contributed to a collaborative and efficient development process. The collaborative integration of SQLite underscores its importance in our project's methodology, supporting a streamlined workflow and facilitating a collaborative approach to database management.

## 4.4 NOTIFICATIONS AND COMMUNICATION:

The system's communication features are designed to keep concerned individuals and authorities informed in real-time. Email notifications are integrated for user registration, and a dynamic notification system is implemented to alert relevant parties when a match is found during the facial recognition process. This ensures timely and effective communication throughout the search process.

- **Django Email Functionality:**

Django's built-in email tools provide a high-level and collaborative way to handle email communication. The configuration settings in the project's `settings.py` file allow for the easy setup of the email backend, specifying parameters such as the host, port, and authentication details. The collaborative utilization of Django's email functions, such as `send_mail()`, `send_mass_mail()`, and `EmailMessage`, allows for the creation and dispatch of emails for various purposes, including user notifications and updates.

- **SMTP Integration:**

SMTP serves as the underlying protocol for sending emails, and Django seamlessly integrates with SMTP servers. Our collaborative decision to configure Django's email backend to use SMTP ensures that our application benefits from the reliability and widespread support of SMTP for email communication. This integration allows for the secure and efficient transmission of emails, with additional support for TLS encryption and authentication credentials for enhanced security.

- **Collaborative Email Templates:**

The integration of Django's templating system with email construction facilitates the creation of dynamic and personalized email content. Collaboratively using templates, such as HTML templates, allows for visually appealing and customized emails tailored to specific user interactions. This approach enhances user engagement by providing content-rich and contextually relevant emails, contributing to a positive user experience.

- **Security Measures and Authentication:**

Collaboratively, our project ensures the secure transmission of emails by leveraging SMTP's TLS capabilities for encrypted communication. Additionally, the collaborative use of authenticated SMTP servers, requiring usernames and passwords, adds an extra layer of security, safeguarding sensitive email content and user information.

- **Enhancing User Engagement:**

The collaborative integration of Django's email functionality and SMTP within our project significantly contributes to enhanced user engagement. Through timely and relevant email notifications, users receive updates, account verification requests, and password reset instructions, fostering dynamic and collaborative communication.

- **Testing and Debugging:**

During the collaborative development process, our project utilized Django's built-in email backends for testing and debugging purposes. Collaboratively configuring the email backend to use console output or file storage provided an efficient means for team members to review and verify email content without the need to send actual emails, streamlining the testing and debugging process.

In summary, the collaborative incorporation of Django's email tools and SMTP in our project's methodology establishes an efficient, reliable, and secure email communication system. This collaborative approach aligns with industry standards, enhances user engagement, and contributes to effective communication within the application, showcasing the importance of collaborative practices in shaping the methodology of our project.

## **4.5 ADMIN FUNCTIONALITIES:**

Administrative control is a pivotal aspect of the system. The administrative panel, developed using Django's admin interface, provides administrators with essential functionalities. These

include the ability to add, edit, update, and delete missing person records, enabling effective management and control over the system's database.

- **User Management:**

The administrative panel allows administrators to manage user accounts efficiently. This includes the ability to create new user accounts, modify existing user details, and deactivate or delete user accounts if necessary.

- **Access Control and Permissions:**

Django's admin interface integrates with the authentication and authorization system, enabling administrators to control access and permissions. This ensures that only authorized personnel can access and modify sensitive information within the system.

- **Customization of Admin Interface:**

The admin interface is customizable, allowing administrators to tailor the view and functionality according to the specific needs of the project. Customizing the admin interface can enhance the user experience for administrators and streamline their workflow.

- **Logging and Auditing:**

Django's admin interface provides logging and auditing features, allowing administrators to track changes made to records. This is crucial for maintaining a comprehensive history of database modifications, enhancing accountability, and facilitating error tracking.

- **Search and Filtering Capabilities:**

The admin interface includes powerful search and filtering capabilities, enabling administrators to quickly locate specific records within the system. This accelerates data retrieval and supports efficient decision-making.

- **Integration with Database Models:**

The admin interface seamlessly integrates with the underlying database models, providing a direct and organized way to interact with the data. Administrators can perform CRUD (Create, Read, Update, Delete) operations on database records without the need for additional development.

## **FUTURE SCOPE:**

- The project's vision extends beyond its initial implementation, encompassing various future enhancements. These include the integration of improved facial recognition algorithms, real-time video analysis capabilities, the development of a mobile application, collaboration with law enforcement agencies, and the establishment of a continuous model training mechanism. These initiatives aim to keep the system at the forefront of technological advancements, ensuring its continued effectiveness and relevance

## 5. SYSTEM DESIGN AND SPECIFICATIONS

The "System Design and Specifications" chapter serves as the conceptual framework and technical blueprint that delineates the intricacies of the project's architecture and functionalities. This crucial phase transforms the abstract concept of the project into a well-defined and structured system, laying the foundation for its development, implementation, and subsequent phases. This chapter encapsulates a comprehensive overview of the design principles, technological specifications, and architectural decisions that collectively shape the project's trajectory.

In the realm of system design, meticulous planning and strategic decision-making are paramount. The objective is to conceptualize an efficient, scalable, and robust system that aligns seamlessly with the project's objectives. This involves delineating the system's architecture, defining its components, specifying data management strategies, and detailing the interactions between various modules. Furthermore, the chapter delves into the user interface design, illustrating how the end-users will interact with the system and ensuring an optimal user experience.

The specifications within this chapter act as a set of guidelines, detailing the technologies, tools, and methodologies employed in the project's development. From the selection of a specific programming language to the choice of database management systems, these specifications serve as the project's technological compass. Additionally, security considerations, scalability measures, and performance optimization strategies are addressed to ensure the system's resilience and responsiveness.

### 5.1 PROJECT MODEL

#### **Input Image / Register Case:**

Users initiate the system by providing an input image or registering a case. This can be done through the system's user interface or an API endpoint.

#### **Get Registered:**

The system processes the registration, storing relevant details in the database. This includes user information, case details, and a reference to the provided image.

#### **Check for Face Detection:**

Utilizing OpenCV for facial recognition, the system checks if a face is detected in the registered image. This step involves preprocessing the image, extracting facial features, and utilizing the face recognition library to identify faces.

#### **Send Mail and Alerts (On Face Detection):**

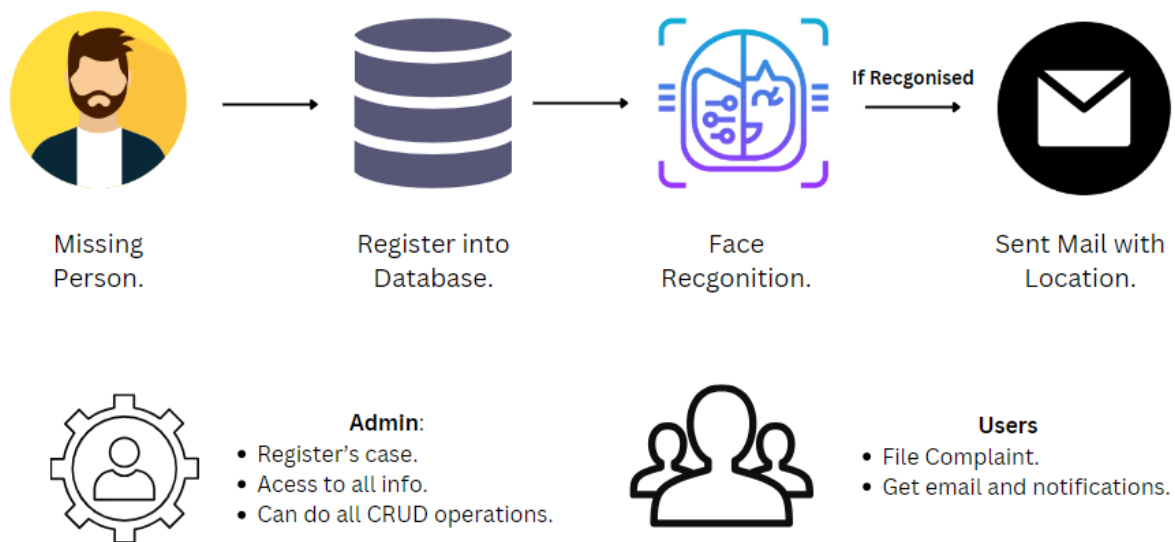
Description: If a face is successfully detected, the system triggers actions such as sending email notifications and alerts to notify relevant parties. This could include case investigators, administrators, or other designated individuals.

### **Search Again (On No Face Detection):**

In the absence of face detection, the system prompts the user to search again. This iterative process continues until a valid face is detected or the user decides to terminate the search.

### **Admin Can Edit, Modify, Delete Information:**

Administrative users have the capability to access, edit, modify, or delete information stored in the system. This functionality is accessible through a secure admin interface, ensuring control and management of the database records.



**Figure 3 Project Model**

## 5.2 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that depicts the flow of data within a system. It illustrates how data is input, processed, and output in a system, highlighting the interactions between different components. DFDs use various symbols to represent processes, data stores, data flows, and external entities, providing a visual and intuitive way to understand the flow of information within a system. The Data Flow Diagram provides a visual representation of the flow of data within the "Missing Person Detection System," illustrating the processes, data sources, and interactions between different components. The DFD for this system can be outlined as follows:

- **External Entities:**

**User (Uploader):** Represents individuals such as family members or law enforcement personnel who upload images of missing persons.

**System:** Represents the "Missing Person Detection System."

- **Processes:**

**Image Upload Process:**

**Inputs:** Images of missing persons along with relevant information.

**Outputs:** Uploaded images stored in the database.

**Face Recognition Process:**

**Inputs:** Uploaded images, stored images from the database.

**Outputs:** Matches or potential matches, triggering an alert.

**Database Management Process:**

**Inputs:** Information from the Image Upload and Face Recognition processes.

**Outputs:** Updated database with new images and associated information.

**Alert Generation Process:**

**Inputs:** Matches or potential matches from the Face Recognition process.

**Outputs:** Alerts sent to relevant authorities and concerned individuals.

- **Data Stores:**

**Image Database:**

Stores images and relevant information uploaded by users.

Accessed and updated by the Image Upload and Database Management processes.

**Face Features Database:**

Stores facial features and encodings for comparison during the Face Recognition process.

Accessed and updated by the Face Recognition process.



**Alert Log:**

Stores information about generated alerts.

Accessed and updated by the Alert Generation process.

- **Data Flows:**

**Image Upload Data Flow:**

Carries images and information from the User to the Image Upload Process.

**Face Recognition Data Flow:**

Carries images from the Image Database and uploaded images to the Face Recognition Process.

Carries matches or potential matches from the Face Recognition Process to the Alert Generation Process.

**Alert Generation Data Flow:**

Carries matches or potential matches from the Face Recognition Process to the Alert Generation Process.

Sends alerts to relevant authorities and concerned individuals.

Updates the Alert Log.

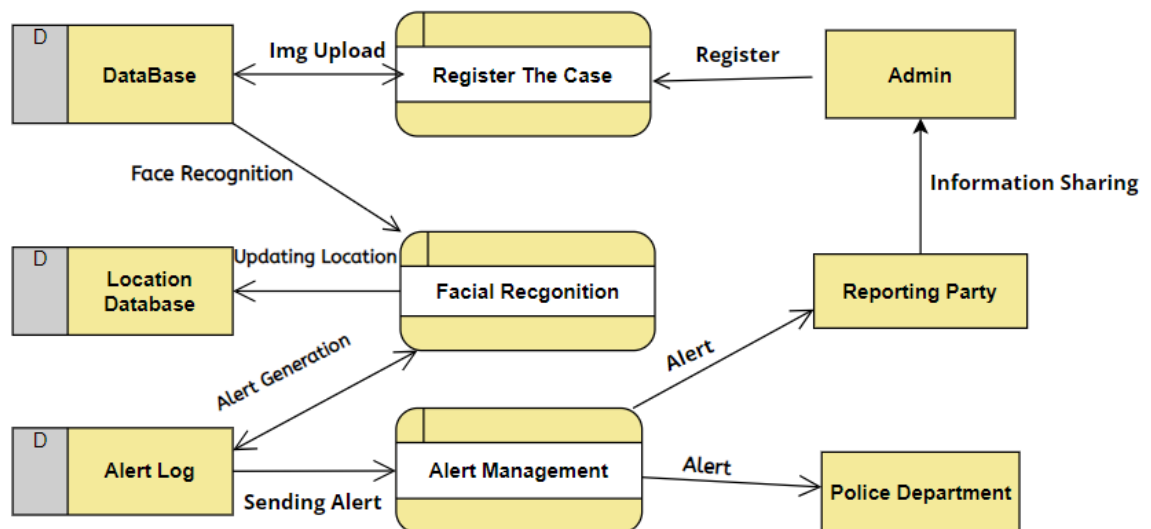


Figure 4 Data Flow Diagram

## 5.3 CLASS DIAGRAM

A Class Diagram is a type of UML (Unified Modeling Language) diagram that represents the structure of a system by illustrating the classes, their attributes, methods, and relationships between classes. It provides a blueprint for the objects that will be created within a system and their interactions. In the context of the "Missing Person Detection System" project, a Class Diagram can help to define the classes involved in the system and how they collaborate.

Now, let's outline some potential classes for the "Missing Person Detection System" project and their relationships:

- **User Class:**  
Attributes: UserID, Username, Email  
Methods: UploadImage(), ViewAlerts()
- **DataBase Class:**  
Attributes: ImageID, Name, Email, Location  
Methods: ProcessImage(), StoreImage()
- **FacialRecognition Class:**  
Attributes: Model, Threshold  
Methods: RecognizeFace(), ExtractFeatures()
- **DatabaseManager Class:**  
Attributes: ImageDatabase, LocationDatabase  
Methods: AddToDatabase(), UpdateDatabase()
- **Alert Class:**  
Attributes: AlertID, Timestamp, Location  
Methods: GenerateAlert(), SendAlert().
- **System Class:**  
Attributes: ImageProcessor, FacialRecognitionModule, DatabaseManager, AlertSystem  
Methods: InitializeSystem(), ProcessImageUpload()
- **Potential Relationships:**  
Association between **User** and **Image**: Users upload images.  
Association between **Image** and **FacialRecognition**: Images are processed for facial recognition.  
Association between **Image** and **DatabaseManager**: Images are stored in the database.

Association between **FacialRecognition** and **DatabaseManager**: Face features are stored and retrieved from the database.

Association between **FacialRecognition** and **Alert**: Alerts are generated based on facial recognition results.

Association between **User** and **Alert**: Users receive alerts.

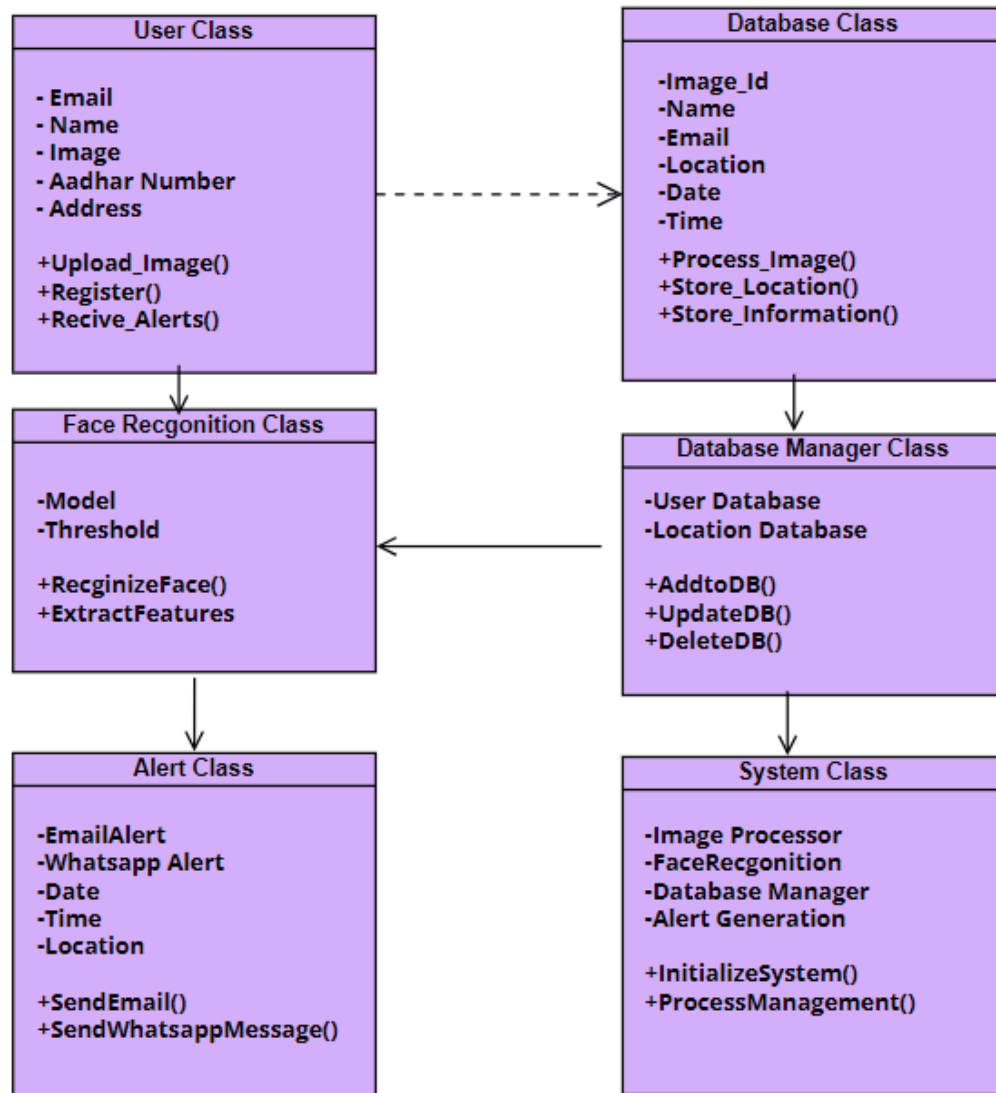


Figure 5 Class Diagram

## 5.4 ER DIAGRAM

An Entity-Relationship (ER) diagram is a visual representation of the data model that describes the structure of a database, including the entities, their attributes, and the relationships between entities. ER diagrams are useful in understanding the data requirements of a system and are often used in database design.

### 1. Entities:

#### **MissingPerson:**

##### **Attributes:**

- first\_name
- last\_name
- father\_name
- date\_of\_birth
- address
- email
- phone\_number
- aadhar\_number
- image
- missing\_from
- gender

#### **Location:**

##### **Attributes:**

- missing\_person (Foreign Key referencing MissingPerson)
- latitude
- longitude
- detected\_at

### 2. Relationships:

#### **MissingPerson - Location Relationship:**

There is a one-to-many relationship between MissingPerson and Location.

Each MissingPerson can have multiple Location entries (indicating different places where they were detected), but each Location is associated with only one MissingPerson.

### 3. Explanation:

The MissingPerson entity represents information about a missing person, including personal details such as name, date of birth, contact information, etc.

The Location entity represents the locations where a missing person was detected. This entity is linked to the MissingPerson entity through a foreign key relationship. The missing\_person attribute in the Location entity serves as a foreign key that references the primary key (ID) of the corresponding MissingPerson.

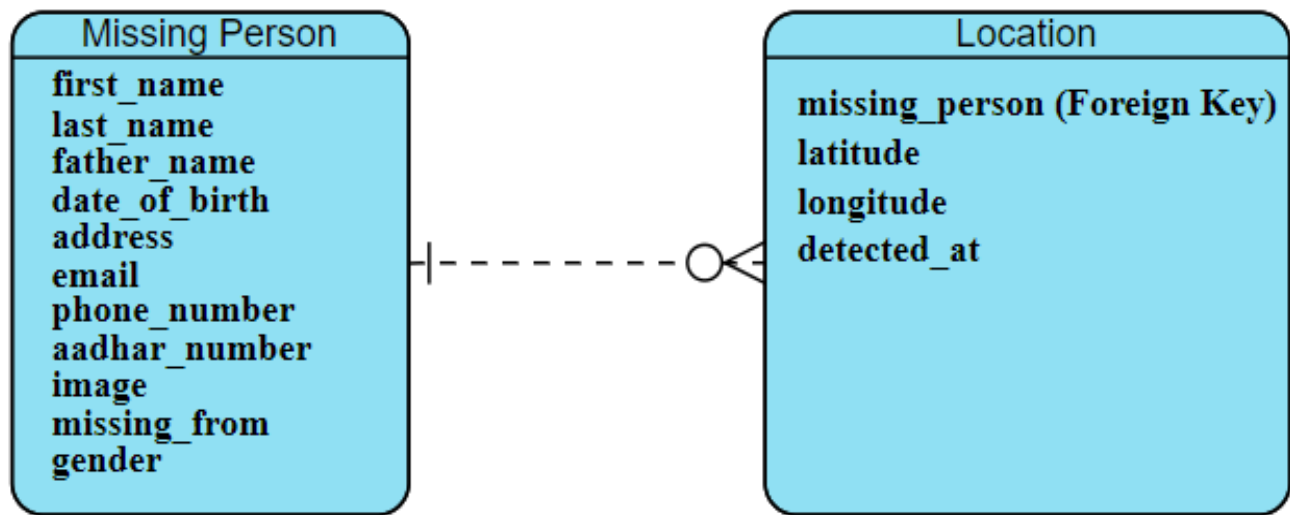


Figure 6 ER Diagram

## 5.5 ACTIVITY DIAGRAM

An Activity Diagram is a UML (Unified Modeling Language) diagram that illustrates the flow of activities within a system. In the context of the "Missing Person Detection System," the Activity Diagram provides a visual representation of the high-level workflow from the registration of a missing case to the administration of the system by an authorized user.

The primary purpose of the Activity Diagram is to depict the sequence of activities and interactions between users and system components. It helps stakeholders, including developers and administrators, understand how the system functions and how different activities are interconnected.

- **Users:**  
**User (Uploader):**  
Individuals such as family members or law enforcement personnel who upload images of missing persons.

- **Admin:**

An authorized user who can perform CRUD (Create, Read, Update, Delete) operations on the system.

- **Create:** Add new records, such as details of a new missing person.
- **Read:** Access information, view alerts, and review missing person's data.
- **Update:** Modify existing records, like updating details of a missing person.
- **Delete:** Remove records, for instance, marking a case as resolved.

- **End Process:**

End:

The diagram concludes with an end node, indicating the end of the entire process.

- **Recommendations for Improvement:**

**Enhanced User Interface:**

Consider improving the user interface for the uploader to make the missing person registration process more user-friendly.

**Real-Time Notifications:**

Explore the possibility of integrating real-time notifications for concerned individuals, providing immediate updates on missing person detections.

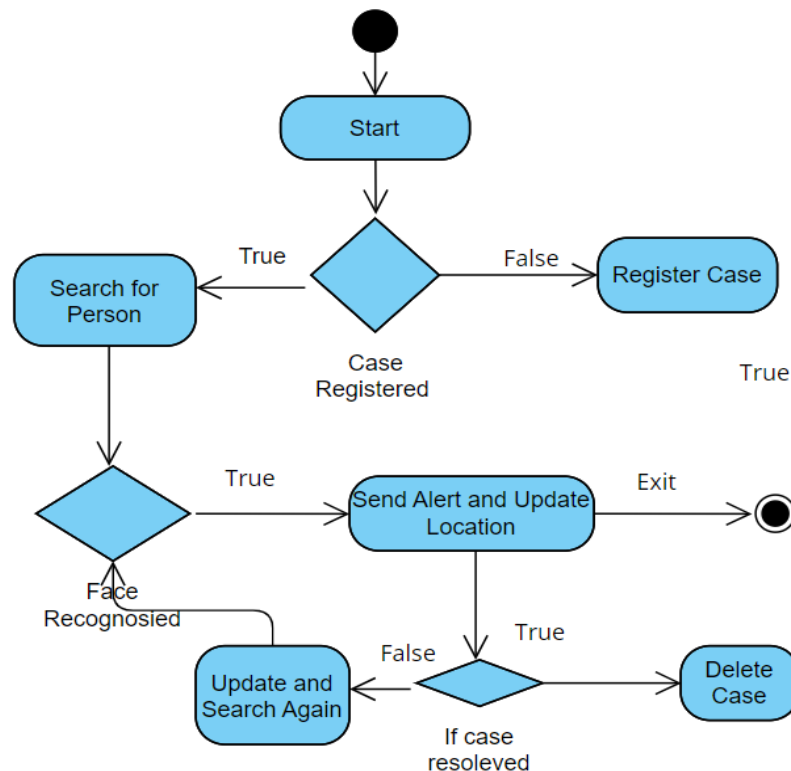


Figure 7 Activity Diagram

## 5.6 USE CASE DIAGRAM

A Use Case Diagram is a visual representation of the interactions between different actors (users or external systems) and a system. It illustrates the functionalities a system should provide from the perspective of end users. A Use Case Diagram is a visual representation of the interactions between different actors (users or external systems) and a system. It illustrates the functionalities a system should provide from the perspective of end users.

### Use Case Diagram for the "Missing Person Detection System":

#### Actors:

- **User** (Family Member/Police):  
Uploading Images  
Receiving Alerts
- **Administrator**:  
Managing Database

#### Use Cases:

- **Upload Image:**  
Actors: User  
Description: Allows users to upload images of missing persons.
- **Perform Facial Recognition:**  
Actors: System  
Description: Utilizes facial recognition to compare uploaded images with the database.
- **Manage Database:**  
Actors: Administrator  
Description: Involves adding, updating, or removing images and associated information from the database.
- **Generate Alert:**  
Actors: System  
Description: Sends alerts to authorities and users when a match is found.

#### Associations:

- User associates with Upload Image and Receive Alert.
- Administrator associates with Manage Database.
- System associates with Perform Facial Recognition and Generate Alert.

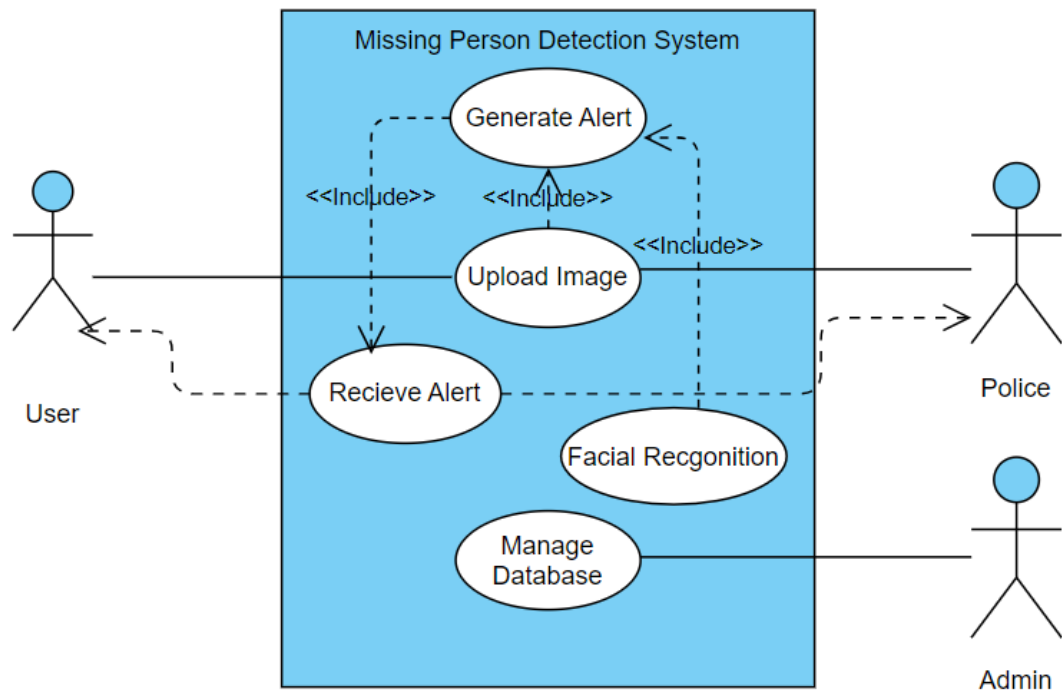


Figure 8 Use Case Diagram



## 6 TESTING

Testing is a critical phase in the development lifecycle of the "Missing Person Detection System." It ensures that the system functions as intended, meets the specified requirements, and is free from defects. The testing process involves various types of testing, including unit testing, integration testing, and system testing, to guarantee the reliability and effectiveness of the system.

The overarching goal of testing in software development is to ensure that the software behaves as expected and meets the specified requirements. Testing is an iterative process that spans the entire software development life cycle, from the initial stages of coding to the final deployment and maintenance. It encompasses various levels, including unit testing, integration testing, system testing, and acceptance testing, each serving a unique purpose in ensuring the quality and reliability of the software.

Software testing is a crucial part of the software development life cycle. Without it, app-breaking bugs that can negatively impact the bottom-line may go undetected. Over time, as applications get more complex, software testing activities have also evolved, with many new techniques and approaches introduced.

The road of software development is bumpy, and products can always be vulnerable to bugs and defects. It is necessary to ensure that software works as expected before being released to the market. Here are several reasons why software testing is essential:

- **Detecting defects for the development team to address**

The end goal of software testing is always to uncover bugs and defects. Modern software is built from highly interconnected components that must work together seamlessly to deliver the intended functionality. One broken component can create a ripple effect and break the entire app. The sooner the broken code is fixed, the smaller the impact. A good testing process in place ensures that a higher quality and more reliable product is always delivered on time.

- **Maintain and enhance product quality**

Product quality is more than just the absence of bugs. When we think of quality, we think of the characteristics that not only meet but also exceed customer expectations. An application is expected to perform the functions it was intended for, but it can only achieve the status of “high-quality” when it goes above and beyond those expectations.

- **Improve customers' trust and satisfaction**

The result of rigorous software testing is improved customers' trust. Although it is unrealistic to expect a completely bug-free software, having a product that is stable, reliable, and constantly meeting the customers' needs will eventually lead to positive user experience in the long term.

- **Identify vulnerabilities that can save company costs and even human lives**  
Financial, medical, legal software or any types of software in the YMYL (Your Money Your Life) field deal with sensitive information. Software applications built for these fields can't afford to undergo crashes, data corruption, or system failures, even at small scale, because the lives of so many people will be affected. Errors in these software's may cause irreversible damage and put the company at risk of litigation. Software testing is there to protect companies from such risk.

## **TYPES OF TESTING:**

Different types of software testing can be classified into multiple categories based on test objectives, test strategy, and deliverables. Currently, there are two major software testing types that Quality Assurance professionals frequently use, including:

### **6.1 UNIT TESTING**

Unit testing is a foundational practice in software development, focusing on isolating and validating the smallest testable parts of an application—individual units. These units can be functions, methods, or classes. The primary purpose of unit testing is to detect and address defects in these isolated components early in the development process. It provides developers with rapid feedback on the correctness of their code, supporting a more efficient and reliable development workflow.

#### **Objective:**

Validate individual components to ensure they function correctly in isolation.

#### **Components Tested:**

##### **Image Upload Functionality:**

Test that the system correctly handles image uploads, stores them in the designated directory, and updates the database with relevant information.

##### **Facial Recognition Model:**

Verify the accuracy and reliability of the facial recognition model by feeding it with known images and checking for expected results.

##### **Database Management Operations:**

Test CRUD operations on the database, ensuring that data can be added, retrieved, updated, and deleted accurately.

#### **Outcome:**

Individual components function correctly in isolation.

## 6.2 INTEGRATION TESTING

Integration testing becomes crucial as the development process progresses, and individual units are combined to create a functioning system. This type of testing verifies the interactions between these integrated components, ensuring that they work together seamlessly. Integration testing can take various approaches, including top-down, where testing progresses from higher-level components to lower-level ones, or bottom-up, which tests from the smallest units upward. The goal is to identify and rectify any issues arising from the collaboration of integrated components.

### **Objective:**

Validate the interaction between different components, ensuring they work together seamlessly.

### **Scenarios Tested:**

#### **Image Upload and Facial Recognition:**

Test the flow from image upload to facial recognition, confirming that uploaded images are processed accurately.

#### **Facial Recognition and Database Management:**

Validate that facial recognition results are appropriately stored and retrieved from the database.

### **Outcome:**

Components work together seamlessly without integration issues.

## 6.3 SYSTEM TESTING:

System testing is a crucial phase in the software testing process, involving the evaluation of the complete and integrated software product. Unlike unit and integration testing that focus on individual components, system testing provides a holistic examination of the entire system. Its primary goal is to verify that the software functions as intended when all components are combined, ensuring it meets the specified requirements.

System testing takes a holistic perspective, examining the entire system to verify its functionality and assess its performance, reliability, and scalability. The primary objectives include confirming that the software performs the functions outlined in the requirements and evaluating its ability to handle various conditions. This level of testing also delves into aspects like performance, security, usability, and compatibility, ensuring a thorough examination of the software's capabilities.

**Objective:**

Validate the entire system, ensuring it functions as a cohesive unit.

**Scenarios Tested:**

End-to-End Missing Person Registration:

Simulate the entire process of a user registering a missing person, including image upload, facial recognition, and database updates.

**Alert Generation and Location Update:**

Test the generation of alerts and accurate location updates when a match is found.

**Outcome:**

The system performs as expected, meeting user requirements.

**6.4 USER ACCEPTANCE TESTING (UAT):**

User Acceptance Testing (UAT) is the final phase in the software testing process and a critical step before the software is released to end-users. It involves evaluating the software's functionality to ensure it meets the business requirements and is acceptable for use in its intended environment. UAT is typically conducted by end-users or business stakeholders who will use the software in their daily operations. The primary objectives of UAT are to validate that the system satisfies user needs, functions as expected, and is ready for deployment.

User Acceptance Testing serves as the final checkpoint in the testing process, ensuring that the software is not only technically sound but also aligns with the business objectives and user expectations. It provides a last opportunity to catch any issues before the software goes live, minimizing the risk of deploying a product that does not meet the needs of its intended users.

**Objective:**

Confirm that the system meets user expectations.

**Scenarios Tested:****Uploader (User):**

Test the user registration process, image upload, and response to alerts.

**Admin:**

Validate CRUD operations, ensuring the admin can efficiently manage missing persons' data.

**Outcome:**

Positive feedback from users, indicating that the system meets their needs.

## 6.5 COMPATIBILITY TESTING

Compatibility testing is a type of software testing that ensures a particular application or system functions correctly across different environments, devices, browsers, operating systems, and network configurations. The goal is to verify that the software provides a consistent and reliable user experience regardless of the varied conditions under which it may be used. This testing helps identify potential issues related to compatibility and ensures that the software is accessible and functional for all users. Compatibility testing is particularly important in today's diverse technology landscape, where users may access applications on a wide range of devices and platforms.

The integration of Compatibility Testing and Regression Testing forms a powerful quality assurance strategy. Compatibility Testing ensures that the software is adaptable to the diverse technological landscapes in which it operates, while Regression Testing safeguards against unintended consequences of ongoing development efforts. Together, they contribute to the creation of a resilient and user-friendly software product. As the software evolves, Compatibility Testing validates its flexibility, and Regression Testing ensures that this evolution does not compromise the established functionality. This synergistic approach is essential for delivering software that not only meets the initial requirements but also stands the test of time, providing a seamless and reliable experience for end-users across different environments and iterations.

### **Objective:**

Validate system functionality across different browsers and devices.

### **Scenarios Tested:**

#### **Browser Compatibility:**

Confirm consistent performance and appearance on various web browsers (Chrome, Firefox, Safari, Edge).

#### **Device Compatibility:**

Ensure a responsive design that adapts to different screen sizes on desktops, laptops, tablets, and smartphones.

## 6.6 REGRESSION TESTING:

Regression Testing is a type of software testing that focuses on ensuring that new code changes (such as bug fixes, enhancements, or feature additions) do not negatively impact the existing functionality of the software. The primary purpose is to detect any unintended side effects of changes and to confirm that the previously developed and tested software still works as expected after modifications. Regression testing is crucial in agile development environments where frequent updates and iterations occur.

Regression Testing is crucial in agile development environments where frequent updates and iterations are common. It involves executing a subset of test cases from the entire test suite to ensure that new changes do not negatively impact previously validated features. Automation is often employed in Regression Testing to streamline the process, allowing for swift feedback on the application's overall health after each development cycle. The primary objective is to catch and rectify defects promptly, preventing the reoccurrence of issues that were previously addressed.

**Objective:**

Confirm that recent changes do not introduce new issues or impact existing features negatively.

**Scenario:**

After introducing new features or fixing bugs, perform regression testing to ensure that existing functionalities remain unaffected.

## 7 OUTPUT

The Output section serves as a visual documentation of the tangible outcomes and user interfaces generated by the project. In this segment, we present a collection of screenshots, graphics, and other visual representations that encapsulate the key functionalities, features, and user interactions within the developed system. These visual outputs not only provide a glimpse into the aesthetics of the project but also serve as a valuable resource for stakeholders, enabling them to visually comprehend the project's progress, functionalities, and overall user experience. The subsequent pages delve into specific aspects of the project, offering detailed insights into the visual elements that contribute to the project's success.

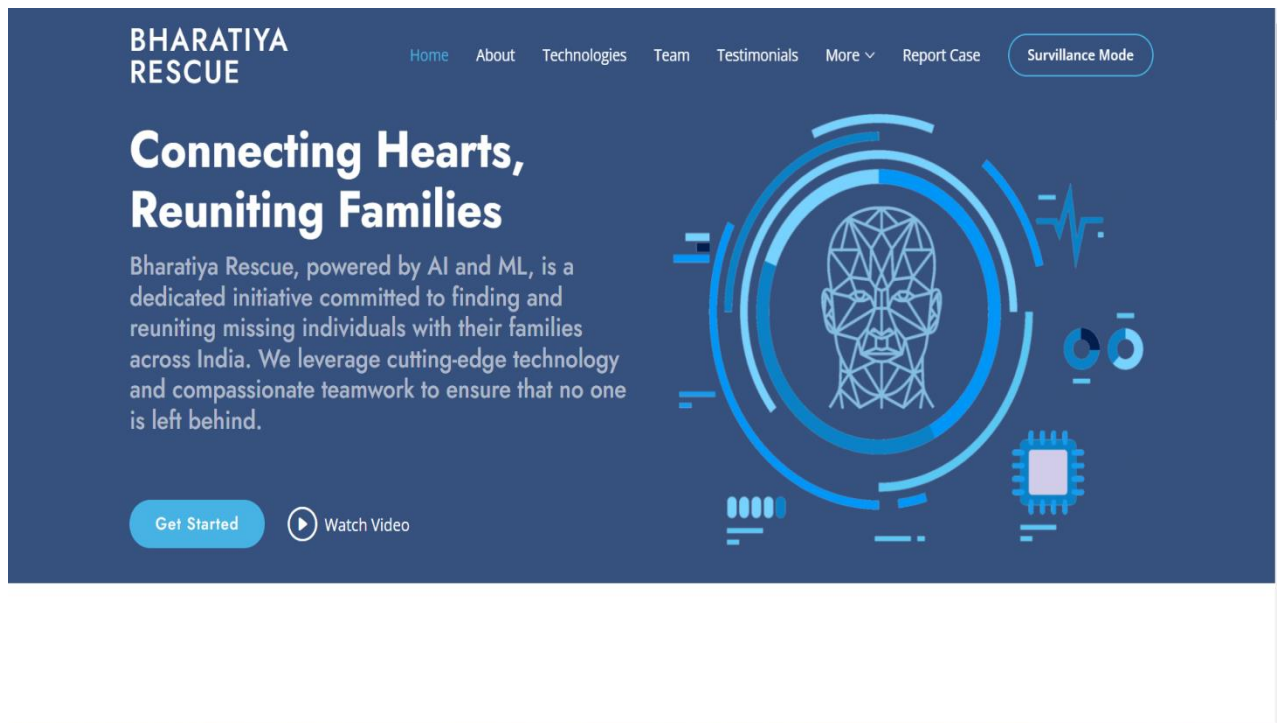


Figure 9 Home Page

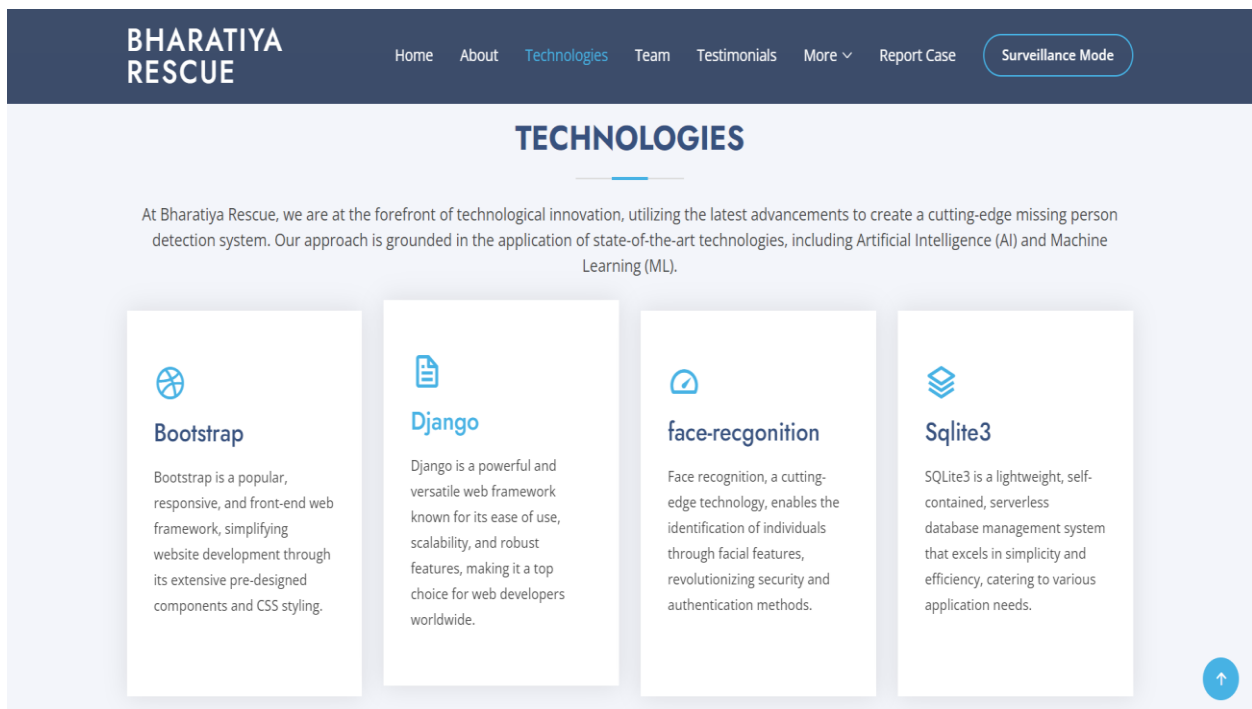


Figure 10 UI and UX of website

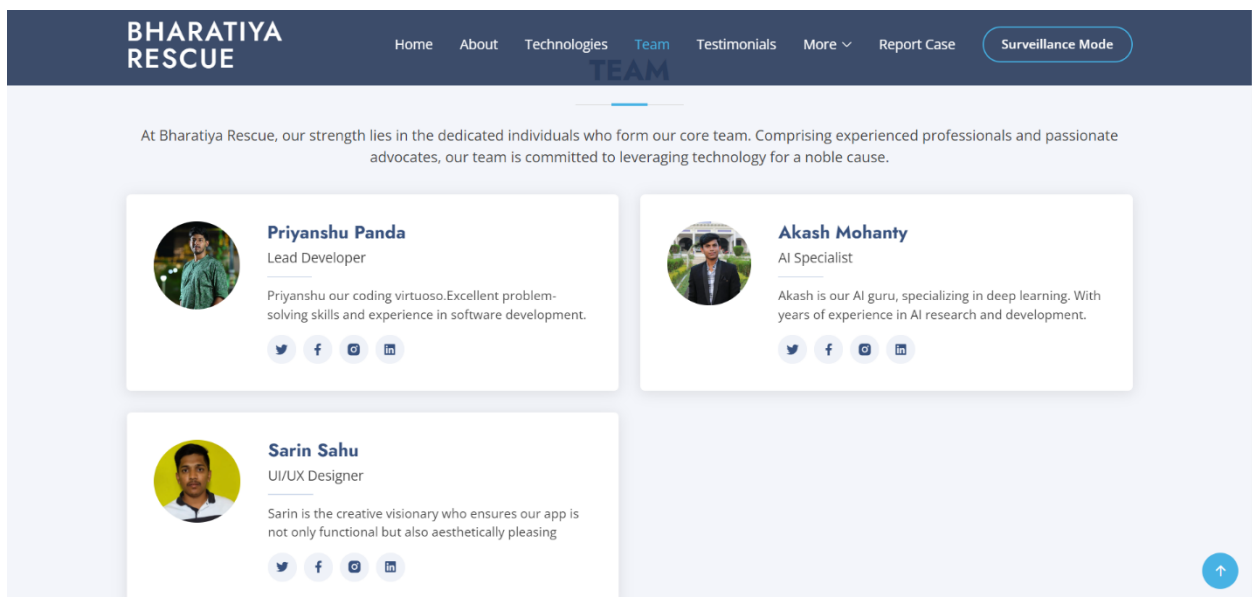


Figure 11 WebPage



## Report Missing Case

First Name	Last Name
<input type="text" value="Sarin"/>	<input type="text" value="Sahu"/>
Father's Name	Date of Birth
<input type="text" value="S Sahu"/>	<input type="text" value="02/25/2002"/>
Address	
<input type="text" value="Rambha, Berhampur"/>	
Email	Phone Number
<input type="text" value="20cse183.priyanshupanda@giet.edu"/>	<input type="text" value="9938082211"/>
We'll never share your email with anyone else.	
Aadhar Number	Missing Date
<input type="text" value="122134435665"/>	<input type="text" value="11/13/2023"/>
Male <input checked="" type="checkbox"/>	Female <input type="checkbox"/>
Others <input type="checkbox"/>	
Image	
<input type="button" value="Choose File"/> Sarin.jpg	

## BHARATIYA RESCUE

Case Registered Successfully

First Name	Last Name
<input type="text" value="Enter Your First Name"/>	<input type="text" value="Enter Your Surname"/>
Father's Name	Date of Birth
<input type="text" value="Enter Father's Name"/>	<input type="text" value="mm/dd/yyyy"/>
Address	
<input type="text" value="Enter your Address"/>	
Email	Phone Number
<input type="text" value="your@email.com"/>	<input type="text" value="Phone Number"/>
We'll never share your email with anyone else.	
Aadhar Number	Missing Date
<input type="text"/>	<input type="text"/>

Figure 12 Registration Page

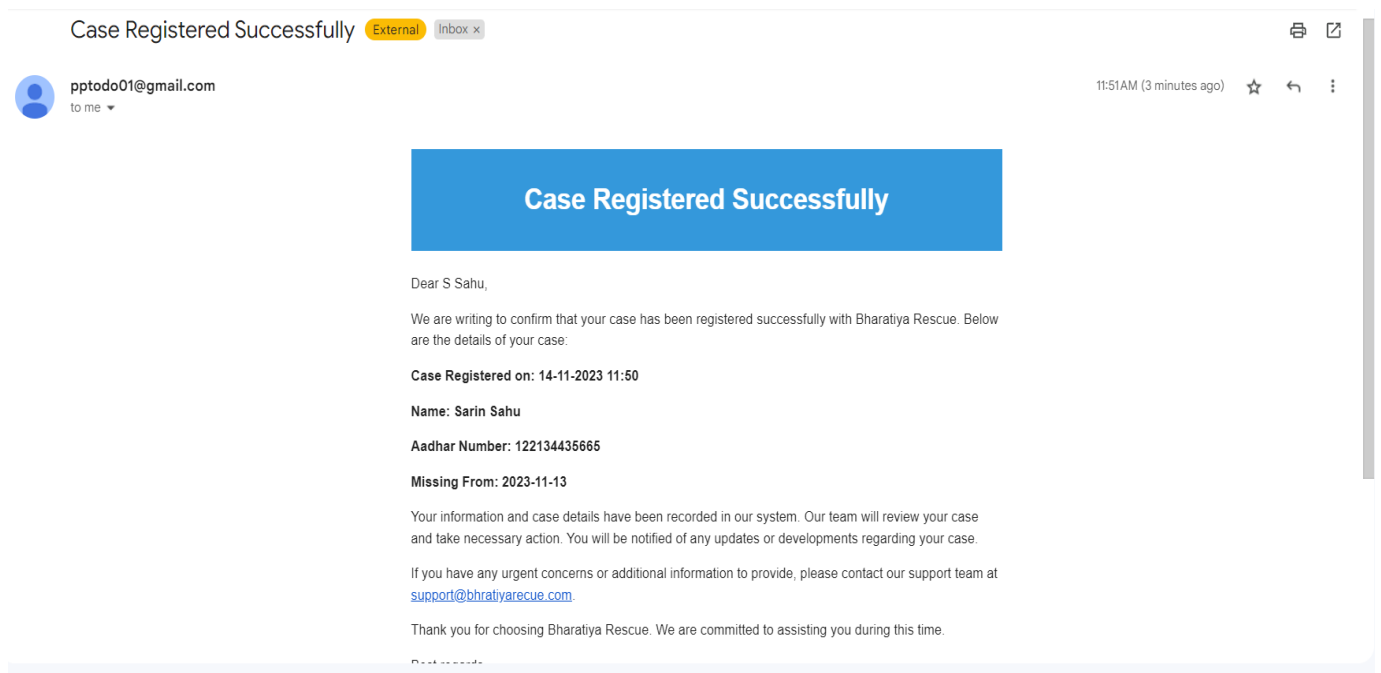


Figure 13 Registration Mail



Figure 14 Face Detection

## Missing Person Found Inbox x



pptodo01@gmail.com  
to me

11:52 AM (1 minute ago) ☆ ↶ ⋮

### Missing Person Found

Dear PK Panda,

We are pleased to inform you that the missing person missing from Sept. 16, 2023 and you were concerned about has been found. The person was located in a camera footage, and we have identified their whereabouts.

Here are the details:

- **Name:** Priyanshu Panda
- **Date and Time of Sighting:** 14-11-2023 11:52
- **Location:** India
- **Aadhar Number:** 123443211234

We understand the relief this news must bring to you. If you have any further questions or require more information, please do not hesitate to reach out to us.

Thank you for your cooperation and concern in this matter.

Sincerely,

Team Bharatiya Rescue

Figure 15 Sending Alert

## Case Registered Successfully External Inbox x



pptodo01@gmail.com  
to me

12:47 PM (25 minutes ago) ☆ ↶ ⋮

### Case Registered Successfully

Dear P C Nayak,

We are writing to confirm that your case has been registered successfully with Bharatiya Rescue. Below are the details of your case:

**Case Registered on:** 14-11-2023 12:47

**Name:** Pratik Nayak

**Aadhar Number:** 345665437890

**Missing From:** 2023-11-13

Your information and case details have been recorded in our system. Our team will review your case and take necessary action. You will be notified of any updates or developments regarding your case.

If you have any urgent concerns or additional information to provide, please contact our support team at [support@bharatiyarecue.com](mailto:support@bharatiyarecue.com).

Thank you for choosing Bharatiya Rescue. We are committed to assisting you during this time.

Figure 16 New Case Registration

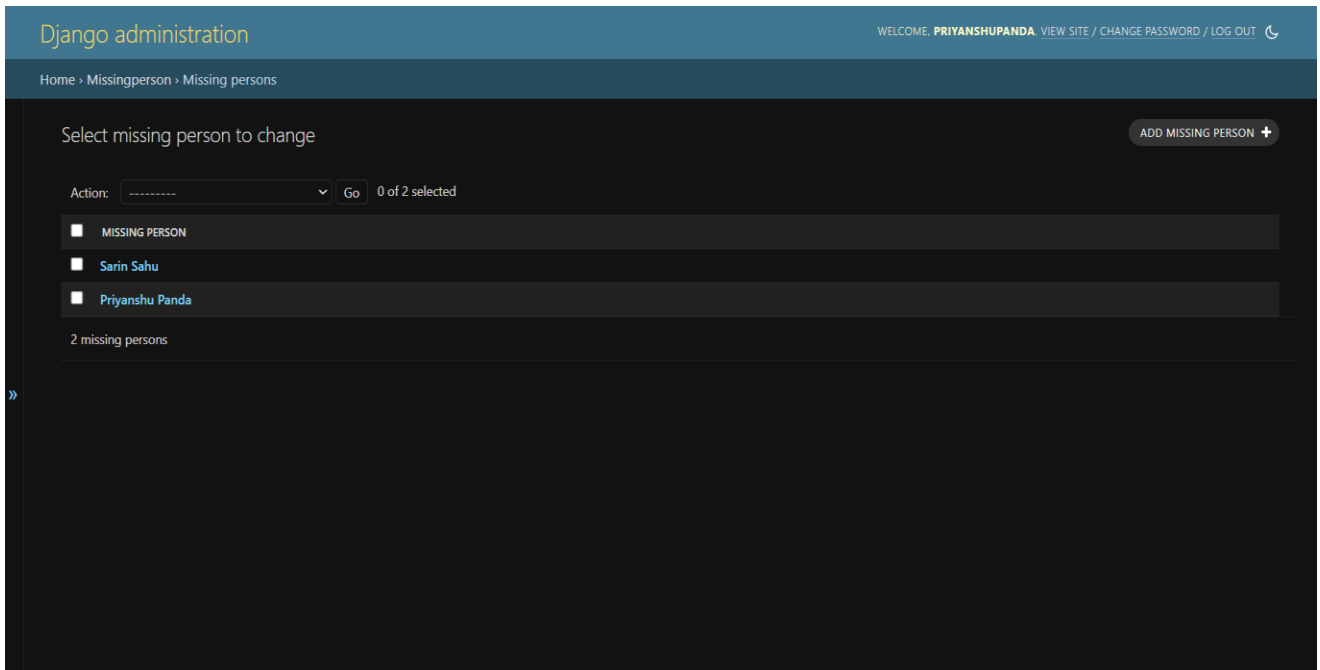


Figure 17 Admin Dashboard

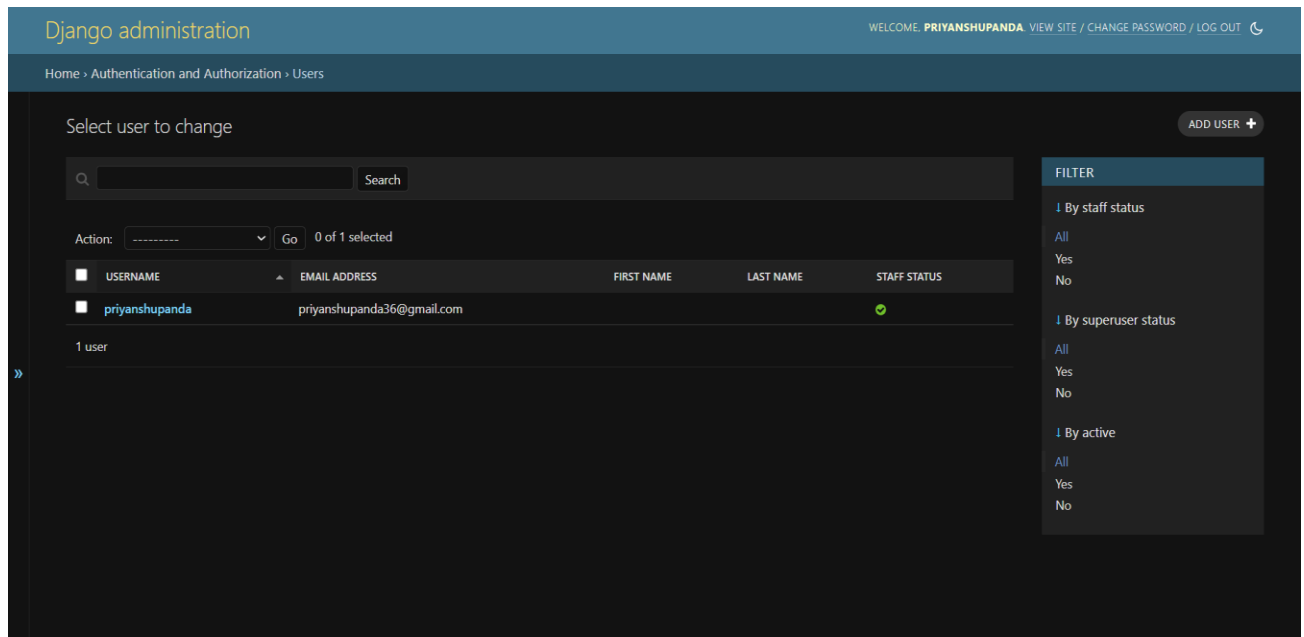


Figure 18 Admin User

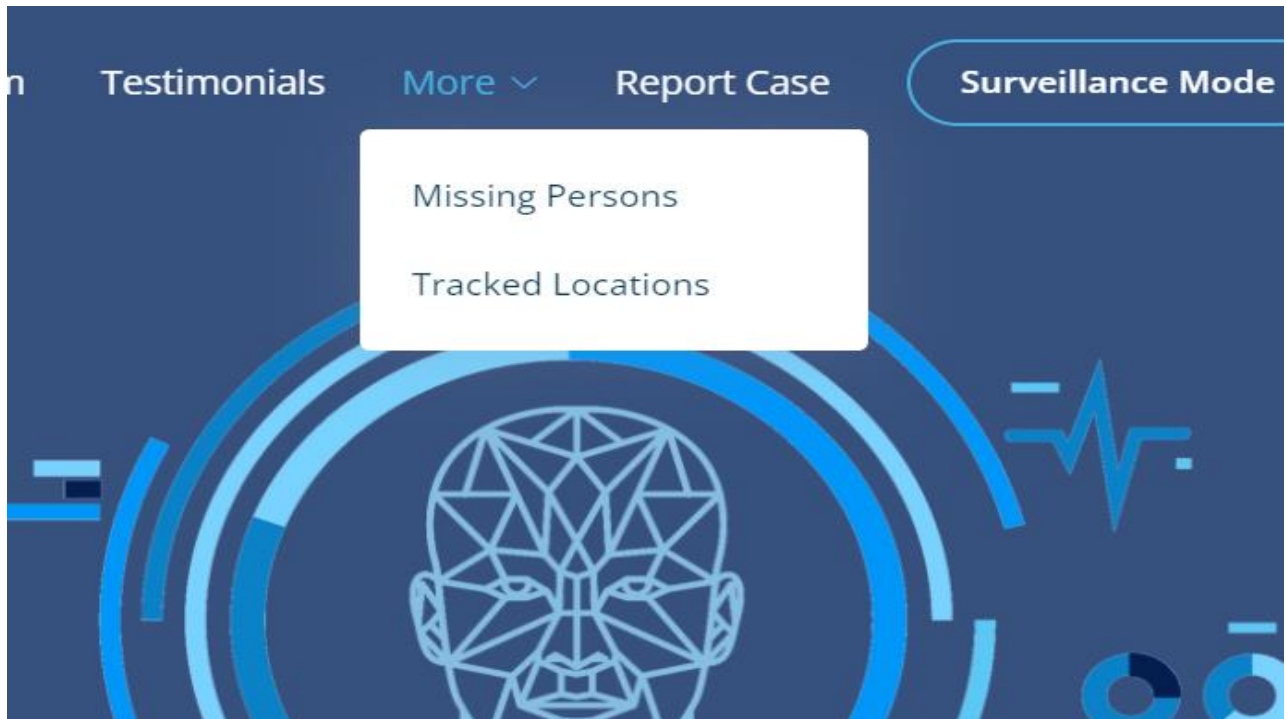


Figure 18 More Features

BHARATIYA  
RESCUE

## Missing Persons




 <p>Name : <b>Priyanshu Panda</b> Aadhar Number : <b>123443211234</b> Missing From : <b>Sept. 16. 2023</b></p>	 <p>Name : <b>Sarin Sahu</b> Aadhar Number : <b>122134435665</b> Missing From : <b>Nov. 13, 2023</b></p>	 <p>Name : <b>Pratik Nayak</b> Aadhar Number : <b>345665437890</b> Missing From : <b>Nov. 13, 2023</b> Phone Number : <b>8989901234</b></p>
---	---	--

Figure 19 List of Missng Persons

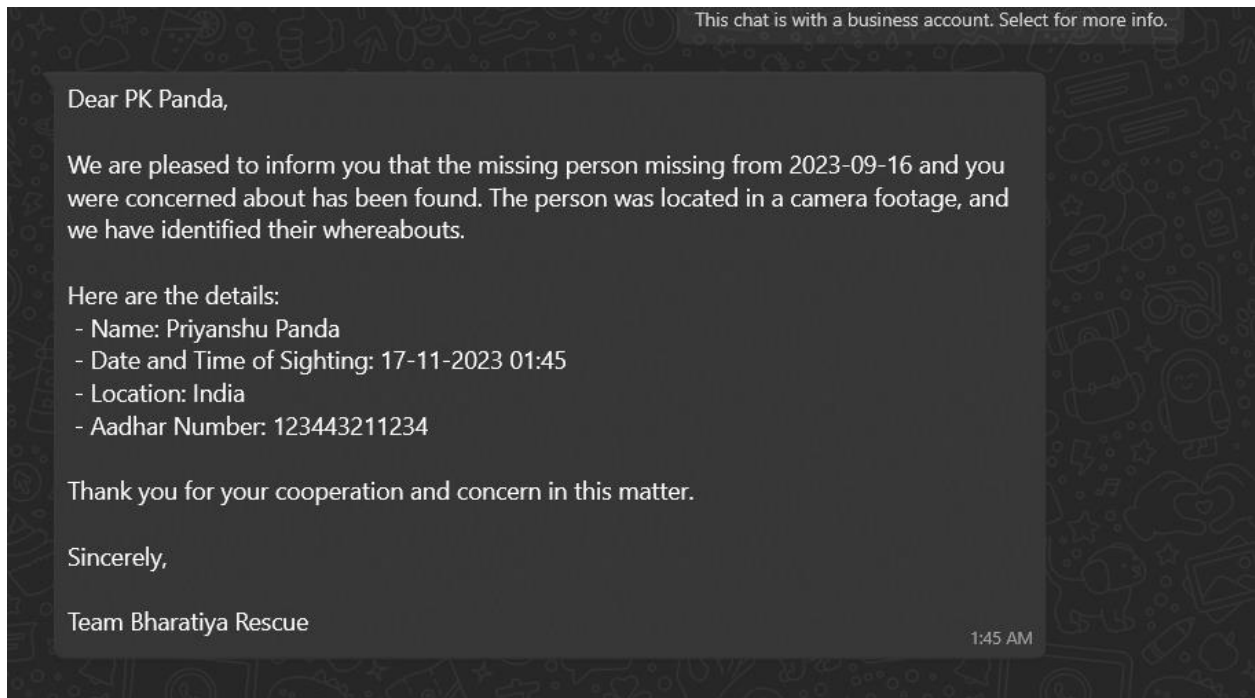


Figure 20 Whatsapp Message

## 8. CONCLUSION

In conclusion, the "Missing Person Detection System" stands as a testament to the intersection of technology and social responsibility, providing an innovative solution to a critical societal issue. The integration of Python programming, machine learning, and web development has resulted in a powerful tool that has the potential to significantly impact the search and rescue efforts for missing individuals.

We would like to express our sincere gratitude to Dr. AVS Pavan Kumar, our esteemed project guide, for his unwavering support, guidance, and expertise throughout the development of this system. His mentorship has been instrumental in shaping the project and ensuring its success. The Department of Computer Science and Engineering at GIET University deserves special thanks for providing the platform and resources that made this project possible.

We are also deeply appreciative of the collaborative efforts of our friends and peers who contributed to this project. Their dedication, teamwork, and shared enthusiasm played a pivotal role in overcoming challenges and achieving our goals. Together, we have not only developed a functional system but have also created a foundation for future advancements and enhancements.

As we look ahead, the "Missing Person Detection System" holds immense potential for growth and improvement. The envisioned future scopes, including enhanced facial recognition, real-time video analysis, mobile application development, and collaboration with law enforcement, demonstrate the commitment to ongoing refinement and expansion. By continuously evolving and incorporating cutting-edge technologies, the system can further strengthen its impact in aiding authorities and families in locating missing persons.

In conclusion, we are proud to have been part of a project that embodies the spirit of technological innovation for the greater good. The "Missing Person Detection System" not only showcases the capabilities of modern technology but also underscores the importance of collaboration, mentorship, and a shared vision for creating solutions that address real-world challenges.

## 9. LIMITATIONS AND FUTURE ENHANCEMENTS

### **Local Machine Dependency:**

**Limitation:** The project currently operates on a local machine, restricting accessibility and scalability.

**Enhancement:** Future iterations could involve deploying the system on cloud platforms, ensuring widespread availability and efficient resource utilization.

### **High GPU Requirement:**

**Limitation:** The project demands high GPU capabilities, potentially limiting its usage on machines without robust graphics processing units.

**Enhancement:** Optimizing the facial recognition model or exploring cloud-based GPU solutions could mitigate this limitation.

### **High Computing Power:**

**Limitation:** The system's reliance on substantial computing power may pose challenges for users with limited computational resources.

**Enhancement:** Implementing algorithmic optimizations and exploring distributed computing solutions can enhance the system's efficiency on less powerful devices.

### **Accuracy Challenges:**

**Limitation:** The project may not always provide accurate results, posing challenges in critical situations.

**Enhancement:** Ongoing refinement of the facial recognition model and incorporating advanced algorithms can improve accuracy and reliability.

### **Scaling Difficulties:**

**Limitation:** Scaling the project for a large number of users may introduce complexities and impact performance.

**Enhancement:** Implementing scalable database management solutions and load balancing techniques can address issues related to scalability.



### **No Cloud Deployment:**

**Limitation:** The absence of cloud deployment restricts accessibility and real-time updates for users.

**Enhancement:** Deploying the system on cloud services like AWS, Azure, or Google Cloud can provide a scalable and reliable infrastructure.

### **Lack of Mobile Application:**

**Limitation:** The absence of a Mobile application limits user interaction and ease of access.

**Enhancement:** Developing a user-friendly Mobile interface can enhance accessibility and user engagement.

### **Security Concerns:**

**Limitation:** The project currently lacks robust security measures, potentially exposing sensitive data.

**Enhancement:** Implementing encryption protocols, secure user authentication, and data privacy measures can enhance the system's security.

### **No Location Access for Cameras:**

**Limitation:** The system does not access the location information of cameras, limiting the ability to track missing persons in specific geographic areas.

**Enhancement:** Incorporating geolocation services and camera metadata can enable more targeted and efficient search operations.

## 10.BIBLIOGRAPHY

### 1. Bootstrap: [www.bootstrap.com](http://www.bootstrap.com)

**Source:** Official website of Bootstrap, a popular open-source front-end framework.

### 2. ChatGPT:

**Source:** Developed by OpenAI, ChatGPT is a language model based on the GPT-3.5 architecture.

### 3. Django: <https://www.djangoproject.com/>

**Source:** Official website of Django, a high-level Python web framework used in the development of the "Missing Person Detection System."

### 4. Face Recognition Library: <https://pypi.org/project/face-recognition/>

**Source:** PyPI page for the face-recognition library, a crucial component in the project's facial recognition module.

### 5. GitHub: <https://github.com/>

**Source:** GitHub, a collaborative platform hosting the project's source code and facilitating version control.

### 6. "Missing Person Identification System" By Prof. S. S. Pawar in IJAR SCT

**Source:** Academic paper providing insights into missing person identification systems, contributing to the theoretical foundation of the project.

### 7. S. Ayyappan and S. Matilda, "Criminals and missing children identification using face recognition and web scrapping" ICSCAN 2020.

**Source:** IEEE conference paper detailing a related study on criminals and missing children identification, influencing the project's approach.

### 8. Visual Paradigm: <https://www.visual-paradigm.com/>

**Source:** Official website of Visual Paradigm, a tool that may have been utilized for system design and modeling.

### 9. YouTube: <https://www.youtube.com/>

**Source:** YouTube, a potential platform for hosting project-related tutorials, presentations, or demonstrations.