Final Report

Technology Stack: Google AI & ML Developer

Project Title: Missing Child Identification using Deep Learning and Multiclass SVM

Team ID: LTVIP2024TMID12998

Team Size: 04 Team

Members:

1. Dade Swetha
2. Karumanchi Sai Srinath
3. Perakam Venkata Bala Pranav Kumar

College: Kallam Haranadhareddy Institute of Technology

ABSTRACT

In India a countless number of children are reported missing every year. Among the missing child cases a large percentage of children remain untraced. This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public. The Convolutional Neural Network (CNN), a highly effective deep learning technique for image based applications is adopted here for face recognition. Face descriptors are extracted from the images using a pre-trained CNN model VGG-Face deep architecture. Compared with normal deep learning applications, our algorithm uses convolution network only as a high level feature extractor and the child recognition is done by the trained SVM classifier. Choosing the best performing CNN model for face recognition, VGG-Face and proper training of it results in a deep learning model invariant to noise, illumination, contrast, occlusion, image pose and age of the child and it outperforms earlier methods in face recognition based missing child identification.

Keywords: Missing Child Identification, Face recognition, Deep Learning, CNN, Multiclass SVM.

CHAPTER 1: INTRODUCTION

1.1 About the Project

India is the second populous country in the world and children represent a significant percentage of total population. But unfortunately a large number of children go missing every year in India due to various reasons including abduction or kidnapping, run-away children, trafficked children and lost children. A deeply disturbing fact about India's missing children is that while on an average 174 children go missing every day, half of them remain untraced. Children who go missing may be exploited and abused for various purposes. As per the National Crime Records Bureau (NCRB) report which was cited by the Ministry of Home Affairs (MHA) in the Parliament (LS Q no. 3928, 20-03-2018), more than one lakh children (1,11,569 in actual numbers) were reported to have gone missing till 2016, and 55,625 of them remained untraced till the end of the year. Many NGOs claim that estimates of missing children are much higher than reported. The public is given provision to voluntarily take photographs of children in suspected situations and uploaded in that portal. Automatic searching of this photo among the missing child case images will be provided in the application. This supports the police officials to locate the child anywhere in India. When a child is found, the photograph at that time is matched against the images uploaded by the Police/guardian at the time of missing. Sometimes the child has been missing for a long time. This age gap reflects in the images since aging affects the shape of the face and texture of the skin. The feature discriminator invariant to aging effects has to be derived. This is the challenge in missing child identification compared to the other face recognition systems. Also facial appearance of child can vary due to changes in pose, orientation, illumination, occlusions, noise in background etc. The image taken by public may not be of good quality, as some of them may be captured from a distance without the knowledge of the child. A deep learning architecture considering all these constrain is designed here. The proposed system is comparatively an easy, inexpensive and reliable method compared to other biometrics like finger print and the iris recognition systems. . The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public. The Convolutional Neural Network (CNN), a highly effective deep learning technique for image based applications is adopted here for face learning model invariant to noise, illumination, contrast, occlusion, image pose and age of the child and it outperforms earlier methods in face recognition based missing child identification.

1.2 FEATURES

1. **Dataset Collection**: Gather a diverse dataset of images containing both missing children and non-missing children. This dataset should be labeled with appropriate tags indicating whether the child is missing or not.
2. **Preprocessing**: Preprocess the images to ensure consistency and remove any noise. This may include resizing, normalization, and augmentation techniques to improve the robustness of the model.
3. **Feature Extraction**: For deep learning, employ convolutional neural networks (CNNs) to automatically extract relevant features from the images. CNNs are well-suited for image classification tasks due to their ability to learn hierarchical representations. Common architectures like VGG, ResNet, or Inception can be used as feature extractors.
4. **Training Deep Learning Model**: Train the CNN using the labeled dataset. This involves feeding the images through the network, adjusting the model's parameters (weights) through backpropagation, and optimizing a loss function to minimize classification errors. Transfer learning can be used if there is limited training data, by fine-tuning a pre-trained model on a similar task.
5. **Multiclass SVM Classifier**: Use a multiclass Support Vector Machine (SVM) classifier for further classification. SVM is a powerful supervised learning algorithm used for classification tasks. It works by finding the hyperplane that best separates different classes in feature space. In this case, the extracted features from the CNN can serve as input to the SVM classifier.
6. **Training SVM**: Train the multiclass SVM using the features extracted from the pre-trained CNN. This involves selecting appropriate parameters (such as the kernel function and regularization parameter) and optimizing them using techniques like grid search and cross-validation.
7. **Integration and Deployment**: Integrate the deep learning model and SVM classifier into a unified system for missing child identification. This system should take an input image, preprocess it, extract features using the CNN, and then classify it using the SVM classifier.

8. **Evaluation and Testing**: Evaluate the performance of the system using metrics such as accuracy, precision, recall, and F1-score. Testing should be done on a separate validation dataset to assess the generalization ability of the model.
9. **Continual Improvement**: Continuously update and improve the model by collecting more data, fine-tuning hyperparameters, and exploring newer architectures or techniques in deep learning and machine learning.

By incorporating both deep learning for feature extraction and SVM for classification, this approach can effectively identify missing children from images with high accuracy and reliability.

1.3 PROBLEM STATEMENT

The task is to develop an automated system for identifying missing children from images using a combination of deep learning and multiclass SVM techniques. Despite extensive efforts by law enforcement agencies and organizations, locating missing children remains a significant challenge due to the vast amount of data and the need for accurate and timely identification.

1.4 OBJECTIVE

The Major objective is to find the missing child using the deep learning algorithms and multiclass SVM. This is an ambitious project with a social impact that aims to assist missing children identification. Missing child identification system combines the CNN based deep learning algorithms for facial features extraction and support vector machine classifier for text classification. It is used to match the child photos with the SVM classifier. We also create a website for the missing children in the project. There is also a need to create a systematic and centralized mechanism for tracking large number of children who either run away or missing for various reasons, and to facilitate their recovery and rehabilitation, a database of missing child is created.

1.5 SCOPE

Children are the greatest asset of each nation. The future of any country depends upon the right upbringing of its children. India is the second populous country in the world and children represent a significant percentage of total population. But unfortunately a large number of children go missing every year in India due to various reasons including abduction or kidnapping, run-away children, trafficked children and lost children. A deeply disturbing fact about India’s missing children is that while on an average 174 children go missing every day, half of them remain untraced. Children who go missing may be exploited and abused for various purposes. As per the National Crime Records Bureau (NCRB) report which was cited by the Ministry of Home Affairs (MHA) in the Parliament (LS Q no. 3928,20-03- 2018), more than one lakh children (1,11,569 in actual numbers) were reported to have gone missing till 2016, and 55,625 of them remained untraced till the end of the year. Many NGOs claim that estimates of missing children are much higher than reported.

CHAPTER 2: LITERATURE SURVEY

2.1 RESEARCH ARTICLE -1

FACE RECOGNITION USING HISTOGRAMS OF ORIENTED GRADIENTS

Authors: O. Deniz, G. Bueno, J. Salido, and F. D. la Torre

Face recognition has been a longstanding problem in computer vision. Recently, Histograms of Oriented Gradients (HOGs) have proven to be an effective descriptor for object recognition in general and face recognition in particular. In this paper, we investigate a simple but powerful approach to make robust use of HOG features for face recognition. The three main contributions of this work are: First, in order to compensate for errors in facial feature detection due to occlusions, pose and illumination changes, we propose to extract HOG descriptors from a regular grid. Second, fusion of HOG descriptors at different scales allows to capture important structure for face recognition. Third, we identify the necessity of performing dimensionality reduction to remove noise and make the classification process less prone to overfitting. This is particularly important if HOG features are extracted from overlapping cells. Finally, experimental results on four databases illustrate the benefits of our approach.

ADVANTAGES: HOG has proven to be an effective descriptor for object recognition, including face recognition. Its ability to capture local gradients and texture information makes it suitable for representing facial features.

DISADVANTAGES: Extracting HOG features and performing dimensionality reduction can be computationally intensive, especially when dealing with large datasets or real-time applications. This could limit the scalability of the approach in certain scenarios.

2.2 RESEARCH ARTICLE – 2

FACE RECOGNITION USING SIFT FEATURES

Authors : C. Geng and X. Jiang

Scale Invariant Feature Transform (SIFT) has shown to be a powerful technique for general object recognition/detection. In this paper, we propose two new approaches: Volume-SIFT (VSIFT) and Partial-Descriptor-SIFT (PDSIFT) for face recognition based on the original SIFT algorithm. We compare holistic approaches: Fisherface (FLDA), the null space approach (NLDA) and Eigenfeature

Regularization and Extraction (ERE) with feature based approaches: SIFT and PDSIFT. Experiments on the ORL and AR databases show that the performance of PDSIFT is significantly better than the original SIFT approach. Moreover, PDSIFT can achieve comparable performance as the most successful holistic approach ERE and significantly outperforms FLDA and NLDA.

2.3 RESEARCH ARTICLE – 3

MISSING CHILD IDENTIFICATION USING FACE RECOGNITION SYSTEM

Authors: Rohit Satle, Vishnuprasad Poojary, John Abraham and Shilpa Wakode.

The human face plays an important role in our social interaction, conveying people's identity. Face recognition is a task that humans perform routinely and effortlessly in their daily lives. Face recognition, as one of the primary biometric technologies, became more and more important owing to rapid advances in technologies such as digital cameras, the Internet and mobile devices, and increased demands on security. A facial recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. Face Recognition System is a computer based digital technology and is an active area of research. This paper addresses the building of face recognition system by using Principal Component Analysis (PCA) method. The PCA has been extensively employed for face recognition algorithms. It not only reduces the dimensionality of the image, but also retains some of the variations in the image data. The system functions by projecting face image onto a feature space that spans the significant variations among known face images. The significant features are known as "Eigen faces", because they are the eigenvectors (Principal Component) of the set of faces they do not necessarily correspond to the features such as eyes, ears, and noses. The projection operation characterize an individual face by a weighted sum of the Eigen faces features and so to recognize a particular face it is necessary only to compare these weights to those individuals.

2.4 RESEARCH ARTICLE – 4

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Author : karen simonyan and andrew zisserman

In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localization and classification tracks respectively. We also show that our representations generalize well to other datasets, where they achieve state-of-the art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision.

2.5 EXISTING MODEL

Mostly missing child cases are reported to the police. The child missing from one region may be found in another region or another state, for various reasons. So even if a child is found, it is difficult to identify him/her from the reported missing cases. A framework and methodology for developing an assistive tool for tracing missing child is described in this paper. An idea for maintaining a virtual space is proposed, such that the recent photographs of children given by parents at the time of reporting missing cases is saved in a repository.

DISADVANTAGES:

Earliest methods for face recognition commonly used computer vision features such as HOG, LBP, SIFT, or SURF. However, features extracted using a CNN network for getting facial representations gives better performance in face recognition than handcrafted features.

CHAPTER – 3 SYSTEM ANALYSIS

3.1 PROPOSED SYSTEM

This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The publician upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public. In missing child project student asking to implement VGG 16 and multiclass SVM.

3.2. ADVANTAGES OF PROPOSED SYSTEM

A deep learning architecture considering all these constrain is designed here. The proposed system is comparatively an easy, inexpensive and reliable method compared to other biometrics like finger print and iris recognition systems. Features extracted using a CNN network for getting facial representations gives better performance in face recognition than handcrafted features. This is to help authorities and parents in missing child investigation.

3.3. PROCESS MODELS USED WITH JUSTIFICATION

Agile is a type of software development methodology that anticipates the need for flexibility and applies a level of pragmatism to the delivery of the finished product. Agile software development requires a cultural shift in many companies because it focuses on the clean delivery of individual pieces or parts of the software and not on the entire application.

Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance. Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

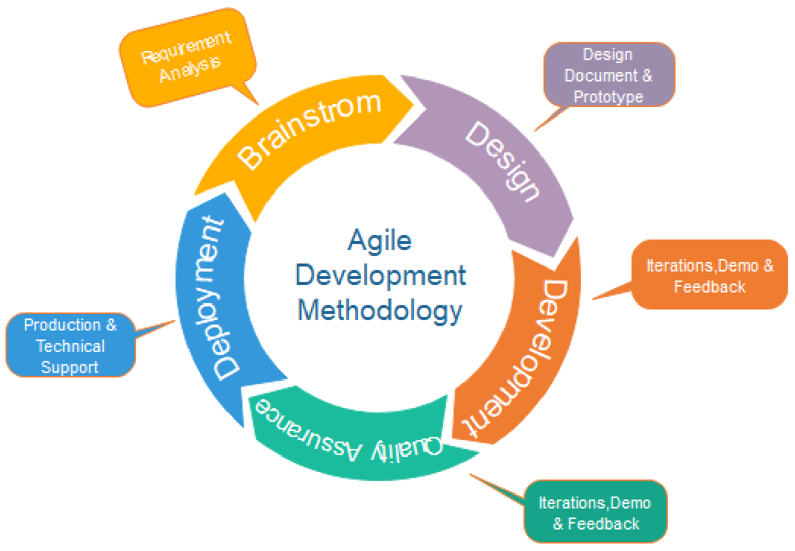

Requirement Analysis
Brainstrom
Design Document & Prototype
Design
Agile Development Methodology
Iterations,Demo & Feedback
Development
Deployment
Production & Technical Support
Quality Assurance
Iterations,Demo & Feedback

Fig: Agile Model

PHASES OF AGILE MODEL:

1. **Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.
2. **Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.
3. **Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.
4. **Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
5. **Deployment:** In this phase, the team issues a product for the user's work environment.
6. **Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

3.3. SOFTWARE REQUIREMENTS SPECIFICATIONS

A Software Requirements specification (SRS) – a requirements specification for a software system is a complete description of behavior of a system to be developed. It includes a set of cases that describe all the interactions users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non- functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints). System Requirements Specification It is a collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions.

3.1.1. FUNCTIONAL REQUIREMENTS:

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behaviour, and outputs (see also software). Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases. Generally, functional requirements are expressed in the form “system shall do <requirement>”. The plan for implementing functional requirements is detailed in the system design. In requirements engineering, functional requirements specify particular results of a system.
Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation.
The various types of outputs in general are:

- External Outputs, whose destination is outside the organization.
- Internal Outputs whose destination is within organization.
- User’s main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

3.1.2. NON-FUNCTIONAL REQUIREMENTS:

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other nonfunctional standards that are critical to the success of the software system. Example of nonfunctional requirement, “how fast does the website load?” Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non- functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are> 10000. Description of non-functional requirements is just as critical as a functional requirement.

- Usability requirement
- Serviceability requirement
- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Maintainability requirement

HARDWARE REQUIREMENTS:

• Processer	: Intel(R) Core(TM)
• Ram	: Min 4GB
• Hard Disk	: Min 100GB

SOFTWARE REQUIREMENTS:

• Front end	: HTML, CSS
• Back end	: MySQL
• Technology	: Python 3.6
• IDE	: PyCharm

3.2. FEASIBILITY STUDY

Feasibility Study is a high-level capsule version of the entire process intended to answer several questions like: What is the problem? Is there any feasible solution to the given problem? Is the problem even worth solving? Feasibility study is conducted once the problem clearly understood. Feasibility study is necessary to determine that the proposed system is Feasible by considering the technical, Operational, and Economical factors. By having adetailed feasibility study the management will have a clear-cut view of the proposed system. The following feasibilities are considered for the project in order to ensure that the project is variable, and it does not have any major obstructions. Feasibility study encompasses the following things.

- Technical Feasibility
- Economic or financial feasibility
- Operational feasibility

In this phase, we study the feasibility of all proposed systems, and pick the best feasible solution for the problem. The feasibility is studied based on three main factors as follows.

TECHNICAL FEASIBILITY:

In this step, we verify whether the proposed systems are technically feasible or not.i.e., all the technologies required to develop the system are available readily or not.

Technical Feasibility determines whether the organization has the technology andskills necessary to carry out the project and how this should be obtained. The system can be feasible because of the following grounds.

- All necessary technology exists to develop the system.
- This system is flexible, and it can be expanded further.
- This system can give guarantee of accuracy, ease of use, and reliability.
- Our project is technically feasible because, all the technology needed for our project isreadily available.

ECONOMIC FEASIBILITY:

In this step, we verify which proposal is more economical. We compare the financial benefits of the new system with the investment. The new system is economically feasible onlywhen the financial benefits are more than the investments and expenditure.

Economic Feasibility determines whether the project goal can be within the resource limits allocated to it or not. It must determine whether it is worthwhile to process with the entire project or whether the benefits obtained from the new system are not worth the costs. Financial benefits must be equal or exceed the costs. In this issue, we should consider:

- The cost to conduct a full system investigation.
- The cost of h/w and s/w for the class of application being considered.
- The development tools.
- The cost of maintenance etc.

Our project is economically feasible because the cost of development is very minimal when compared to financial benefits of the application.

OPERATIONAL FEASIBILITY:

In this step, we verify different operational factors of the proposed systems like manpower, time etc., whichever solution uses less operational resources, is the best operationally feasible solution. The solution should also be operationally possible to implement. Operational Feasibility determines if the proposed system satisfied user objectives could be fitted into the current system operation.

- The methods of processing and presentation are completely accepted by the clients since they can meet all user requirements.
- The clients have been involved in the planning and development of the system.
- The proposed system will not cause any problem under any circumstances.
- Our project is operationally feasible because the time requirements and personnel requirements are satisfied. We are a team of four members, and we worked on this project for three working months.

CHAPTER – 4: SYSTEM DESIGN

4.1. SYSTEM ARCHITECTURE

The proposed methodology for missing child identification which combines facial feature extraction based on deep learning and matching based on support vector machine. The proposed system utilizes face recognition for missing child identification. This is to help authorities and parents in missing child investigation. The architecture of the proposed framework is given below

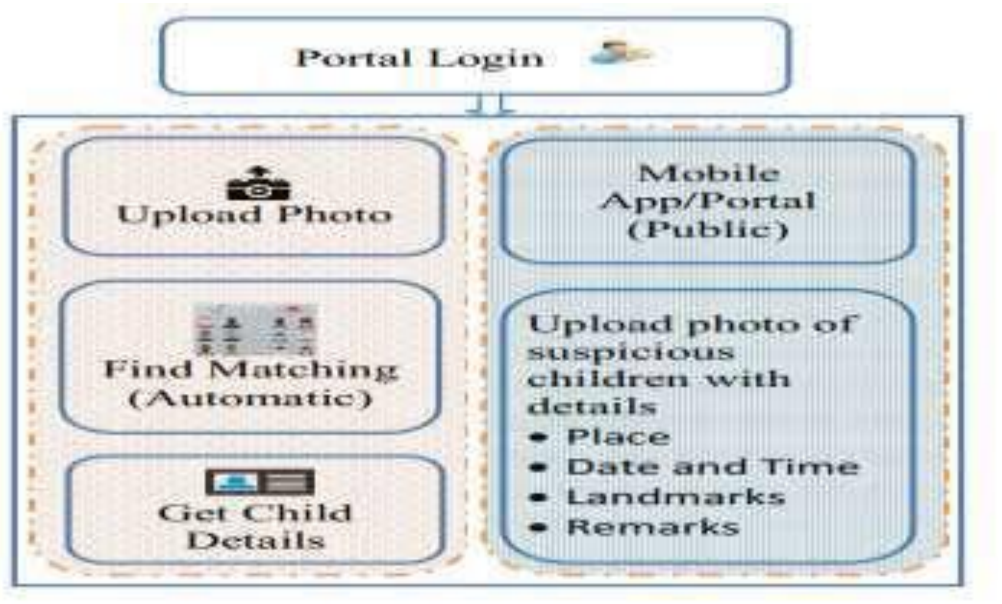

Portal Login
Upload Photo
Find Matching (Automatic)
Get Child Details
Mobile App/Portal (Public)
Upload photo of suspicious children with details
• Place
• Date and Time
• Landmarks
• Remarks

Fig: System Architecture

Images of reported missing children are saved in a repository and the face area is selected for cropping to obtain input face images. Learned features from a Convolutional Neural Network (CNN), a specific type of deep learning algorithm, are used for training a multi class SVM classifier. This machine learning approach is used to correctly label the child using the name indicated in the database provided by the concerned authority. In the following sections the paper details the work flow for child matching methodology. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public.

The block diagram of the automatic child face identification methodology is as shown

Block Diagram

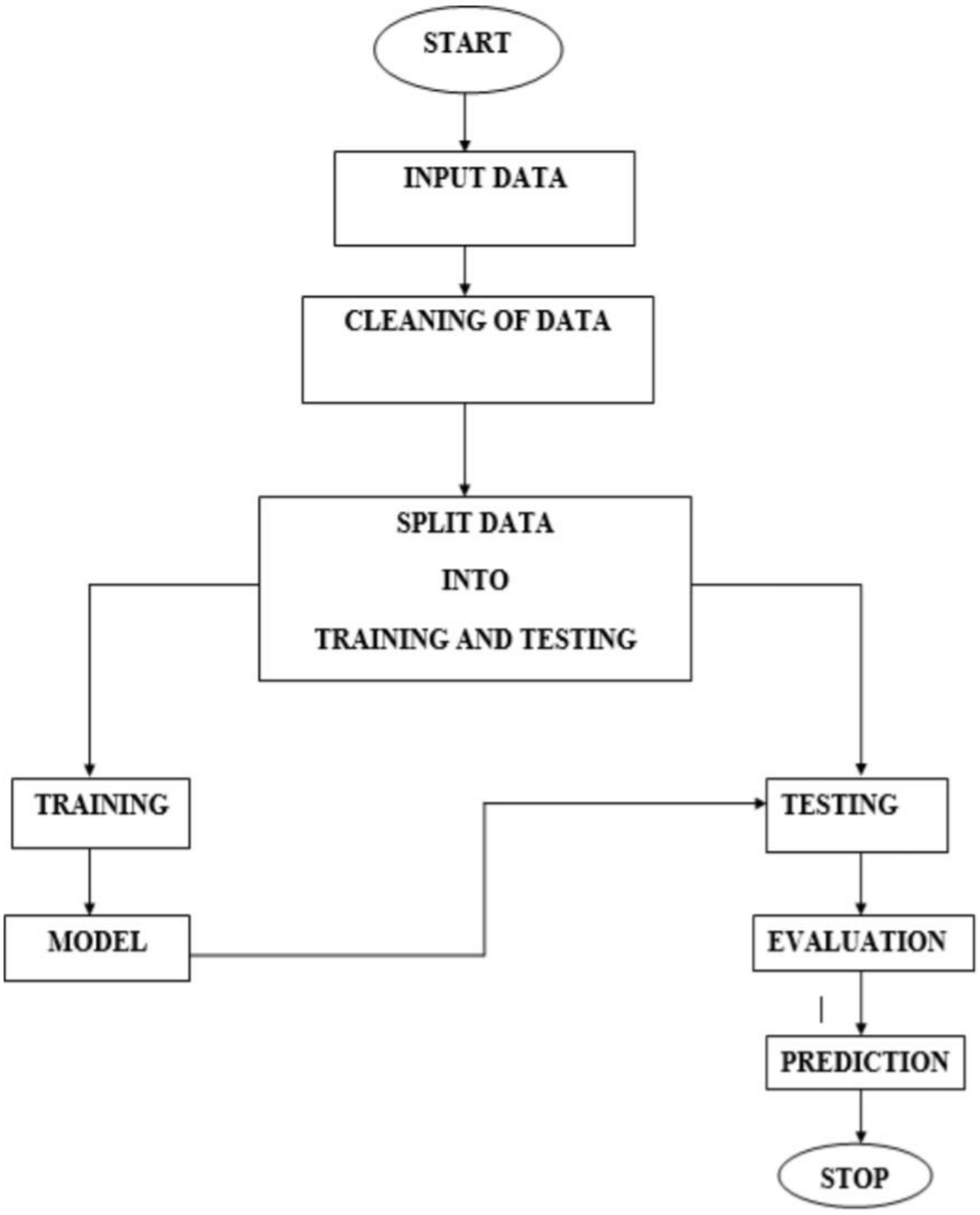


Fig : Block Diagram

4.2. UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. GOALS The Primary goals in the design of the UML are as follows: Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models. Provide extendibility and specialization mechanisms to extend the core concepts. Be independent of particular programming languages and development process. Provide a formal basis for understanding the modeling language. Encourage the growth of OO tools market.

4.2.1. CLASS DIAGRAM:

Class-based Modeling, or more commonly class-orientation, refers to the style ofobject-oriented programming in which inheritance is achieved by defining classes of objects asopposed to the objects themselves.

The most popular and developed model of OOP is a class-based model, as opposed to an object-based model. In this model, objects are entities that combine state (i.e., data),behavior and identity. The structure and behavior of an object are defined by a class, which is a definition, or blueprint, of all objects of a specific type. An object must be explicitly created based on a class and an object thus created is an instance of that class. An object is similar to a structure, with the addition of method pointers, member access control, and an implicit data member which locates instances of the class (i.e., actual objects of that class) in the class hierarchy.

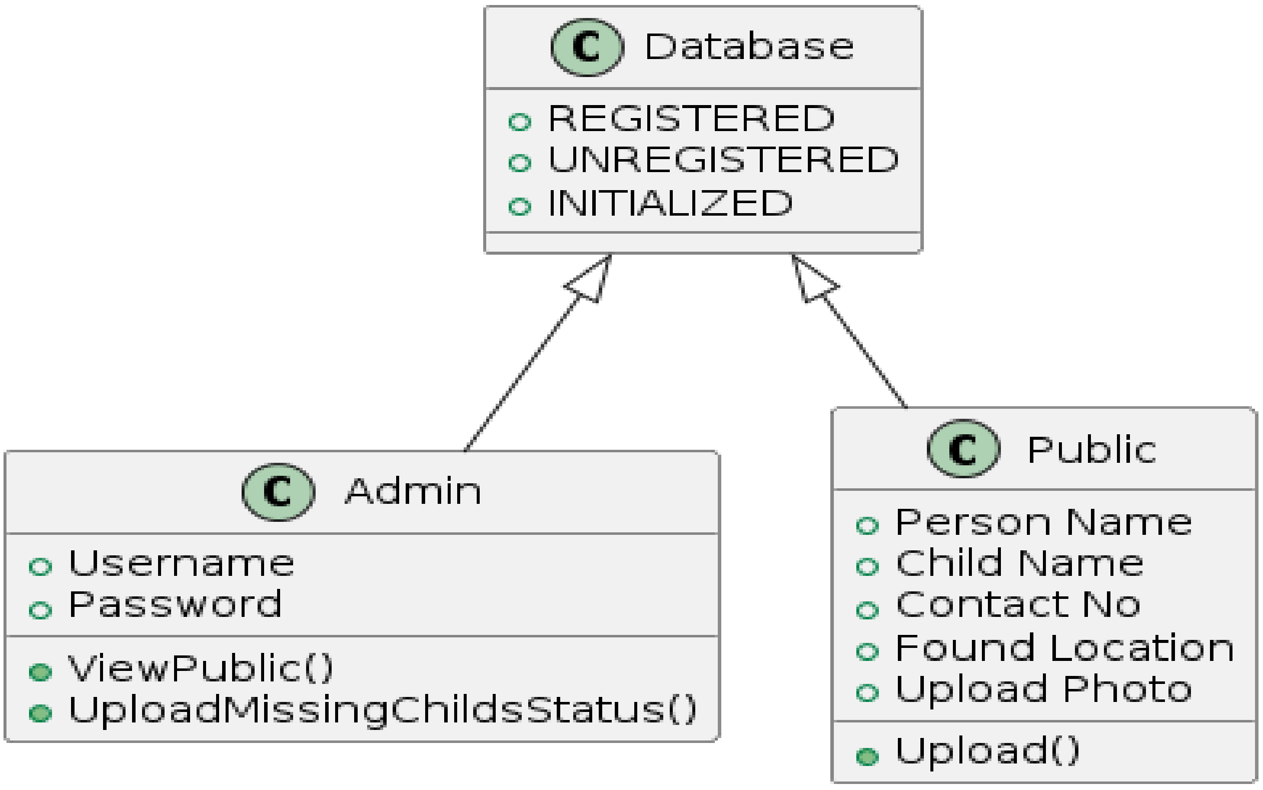


Fig: Class Diagram

4.2.2. USE-CASE DIAGRAM:

Use case diagram represents the functionality of the system. Use case focus on the behavior of the system from external point of view. Actors are external entities that interact with the system.

Use cases:

A use case describes a sequence of actions that provide something of measurable valueto an actor and is drawn as a horizontal ellipse.

Actors:

An actor is a person, organization, or external system that plays a role in one or more interactions with the system.

The “user model view” encompasses a problem and solution from the preservative of those individuals whose problem the solution addresses. The view presents the goals and objectives of the problem owners and their requirements of the solution.

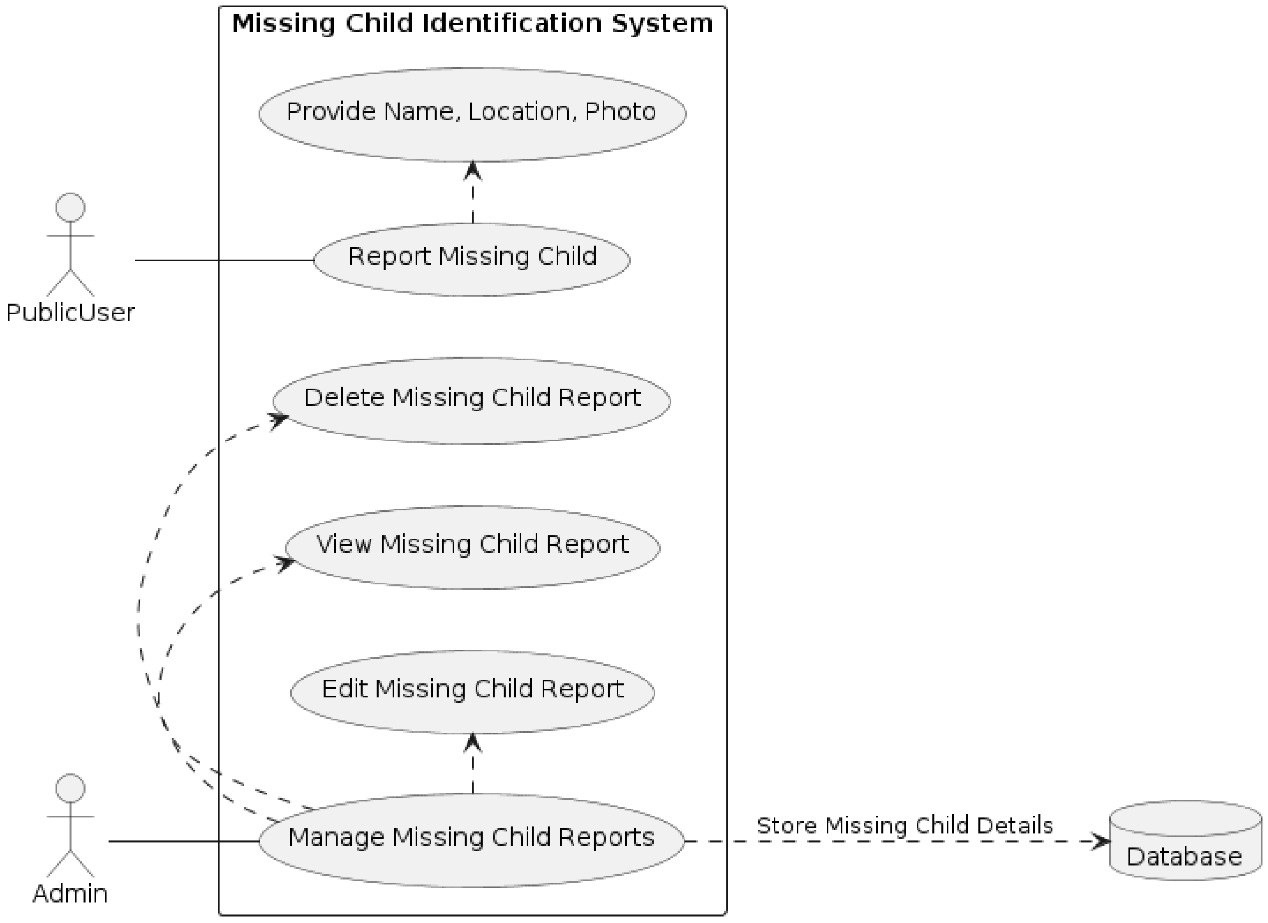

Missing Child Identification System
Provide Name, Location, Photo
Report Missing Child
PublicUser
Delete Missing Child Report
View Missing Child Report
Edit Missing Child Report
Manage Missing Child Reports
Admin
Store Missing Child Details
Database

Fig: Usecase Diagram

4.2.3. ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Activity diagrams are constructed with different types of shapes, connected with arrows.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start (split) or end (join) of concurrent activities.
- A black circle represents the start (initial state) of the workflow.
- An encircled black circle represents the end (final state).

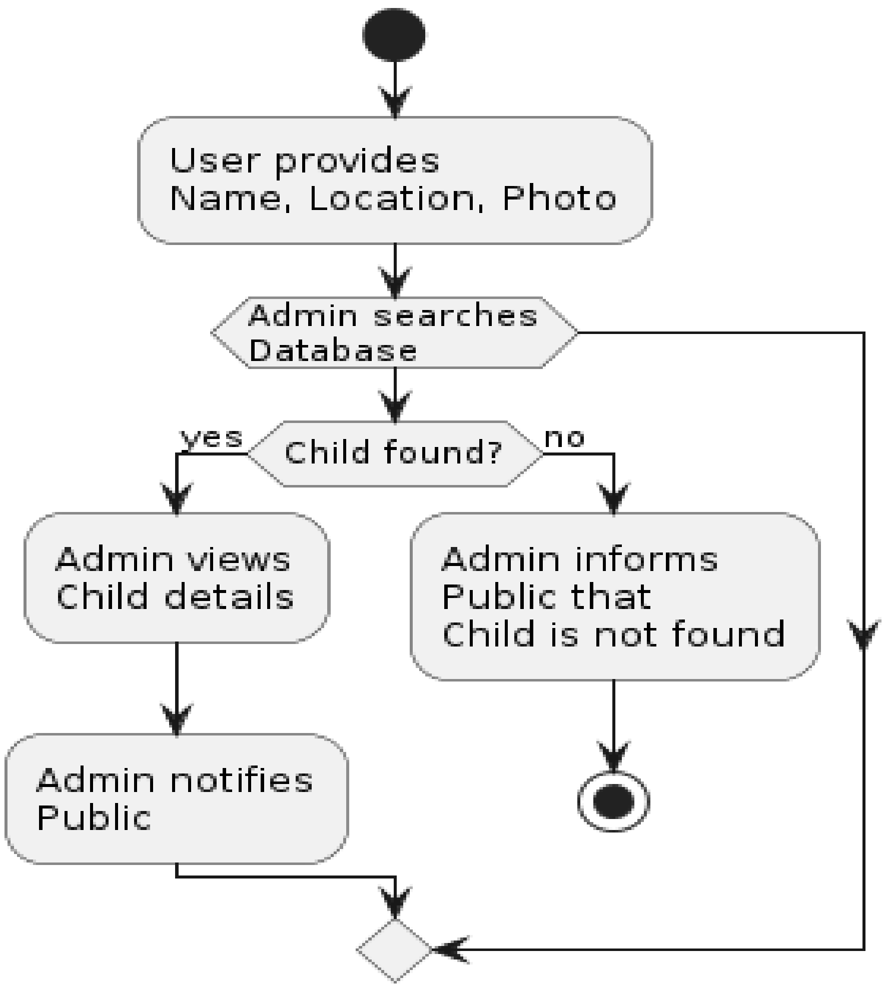


Fig: Activity Diagram

4.2.4. SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams. A sequence diagram shows, as parallel vertical lines (lifelines), differentprocesses or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple run time scenarios in a graphical manner. If the lifeline is that of an object, it demonstrates a role. Note that leaving the instance name blank can represent anonymous and unnamed instances. To display interaction, messages are used. These are horizontal arrows with the message name written above them.

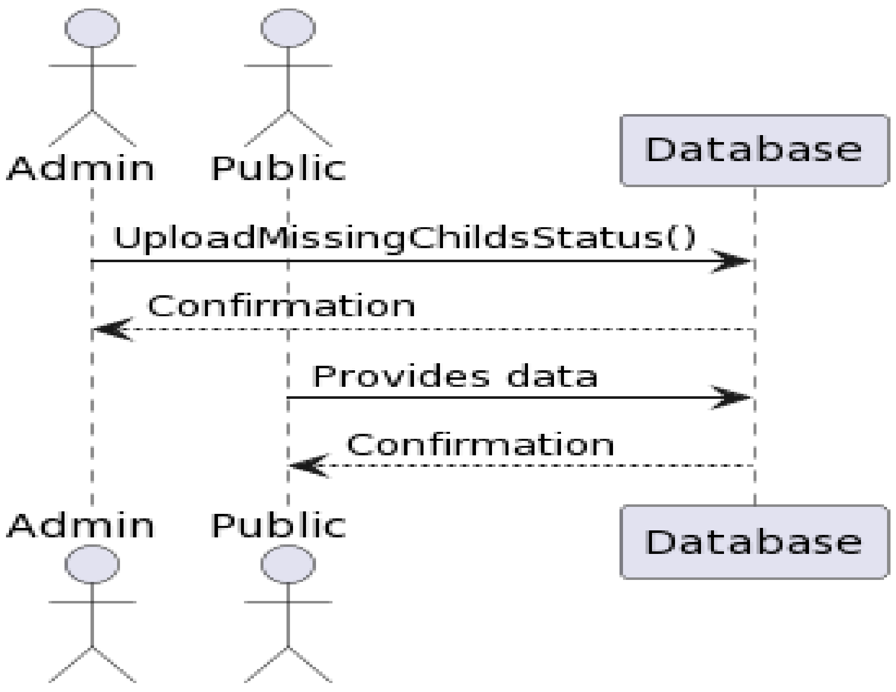


Fig : Sequence Diagram

Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent those processes are being performed in response to the message. A message sent from outside the diagram can be represented by a messageoriginating from a filled-in circle or from a border of sequence diagram.

4.2.5. COMPONENT DIAGRAM:

Component:

This section defines the term component and discusses the differences between object oriented, traditional, and process related views of component level design. Object Management Group OMG UML defines a component as “a modular, deploy able, and replaceable part of a system that encapsulates implementation and exposes a set ofinterfaces.”

A component contains a set of collaborating classes. Each class within a component has been fully elaborated to include all attributes and operations that are relevant to its implementation. As part of the design elaboration, all interfaces that enablethe classes to communicate and collaborate with other design classes must also be defined. To accomplish this, the designer begins with the analysis model and elaborates analysis classes and infrastructure classes.

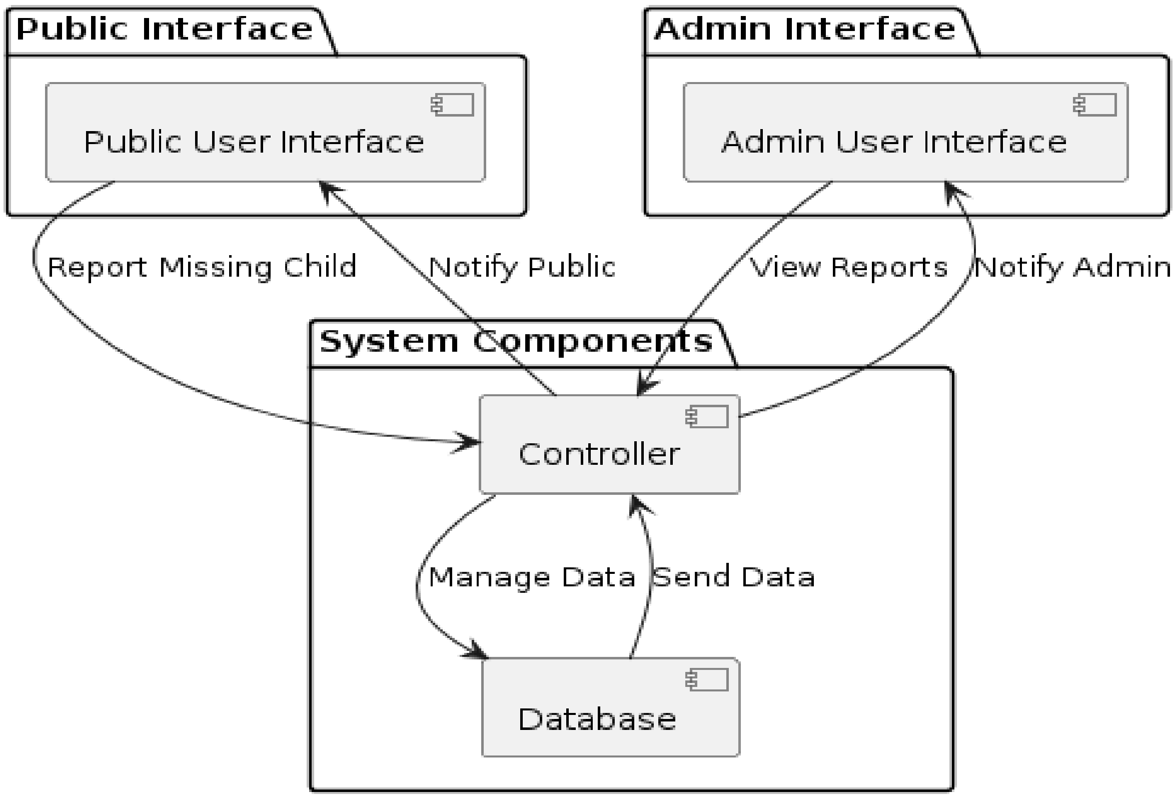


Fig : Component Diagram

4.2.6. DEPLOYMENT DIAGRAM:

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

So deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

Purpose:

The name Deployment itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components where software components are deployed. Component diagrams and deployment diagrams are closely related. Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware.

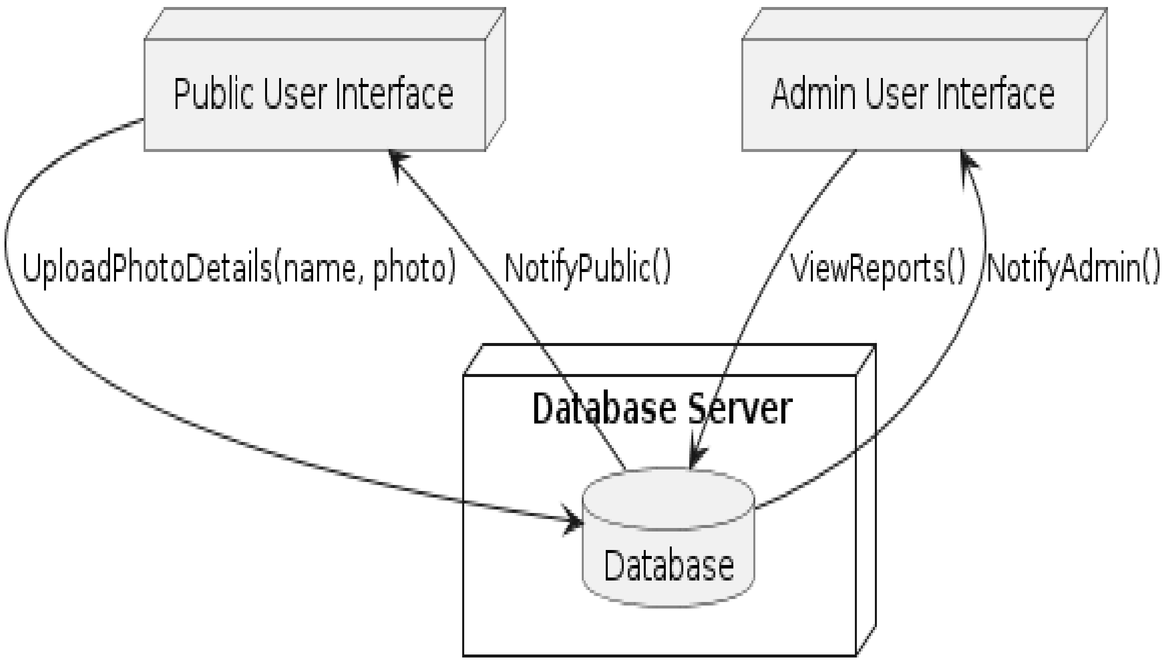


Fig : Deployment Diagram

Drawing Deployment Diagram:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application.

4.2.7 COLLABORATION DIAGRAM

A UML Collaboration Diagram, also known as a Communication Diagram, is a type of interaction diagram in the Unified Modeling Language (UML) that visualizes the interactions and relationships among objects or actors within a system or a scenario. Collaboration diagrams focus on the structural organization of objects and the messages exchanged between them to accomplish a particular task or scenario.

Collaboration diagrams are particularly useful for visualizing the dynamic behavior of a system or a scenario, showing how objects collaborate to accomplish a specific task or achieve a particular goal. They help stakeholders understand the sequence of interactions and the roles of different objects within the system. Collaboration diagrams complement other UML diagrams, such as sequence diagrams and class diagrams, providing a comprehensive view of the system's structure and behavior.

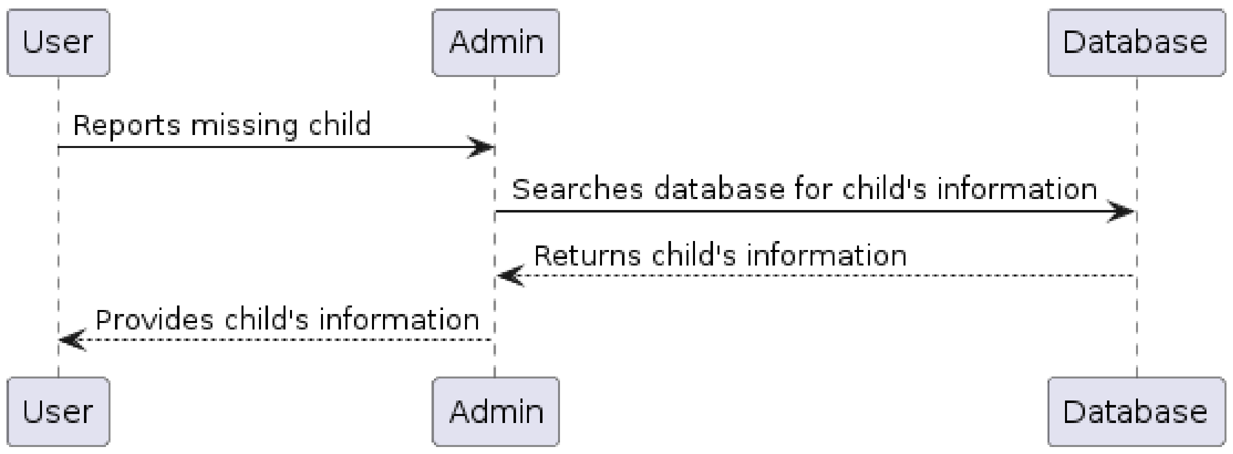

User
Admin
Database
Reports missing child
Searches database for child's information
Returns child's information
Provides child's information
User
Admin
Database

Fig : Collaboration Diagram

However, collaboration diagrams are best suited to the portrayal of simple interactions among relatively small numbers of objects. As the number of objects and messages grows, a collaboration diagram can become difficult to read and use efficiently. Additionally, collaboration diagrams typically exclude descriptive information, such as timing. The collaborations are used when it is essential to depict the relationship between the object. Both the sequence and collaboration diagrams represent the same information, but the way of portraying it quite different. The collaboration diagrams are best suited for analyzing use cases.

4.2.5 OBJECT DIAGRAM

An Object Diagram in UML (Unified Modeling Language) is a static diagram that provides a snapshot of the instances of classes and their relationships at a specific point in time. It represents concrete instances of classes and their attributes, showing how objects interact with each other within the system.

Object diagrams consist of objects, each representing an instance of a class, and links between objects representing relationships. Attributes and values of the objects are typically shown alongside the object, providing additional context. These diagrams are usually created after class diagrams and can be used to validate the design before implementation.

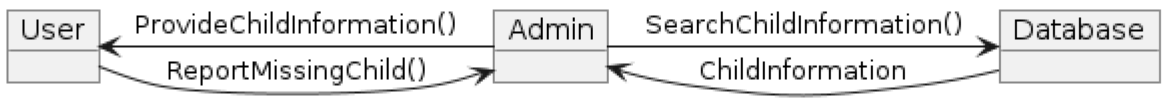


Fig : Object Diagram

Object diagrams are simple to create: they're made from objects, represented by rectangles, linked together with lines. Object diagrams are a visual representation in UML (Unified Modeling Language) that illustrates the instances of classes and their relationships within a system at a specific point in time. They display objects, their attributes, and the links between them, providing a snapshot of the system’s structure during execution.

Since object diagrams depict behavior when objects have been instantiated, we can study the behavior of the system at a particular instant. An object refers to a specific instance of a class within a system. A class is a blueprint or template that defines the common attributes and behaviors shared by a group of objects. An object, on the other hand, is a concrete and individual occurrence of that class, possessing unique values for its attributes.

In UML a classifier refers to a group of elements that have some common features like methods, attributes and operations. A classifier can be thought of as an abstract metaclass which draws a boundary for a group of instances having common static and dynamic features.

4.3. MODULES

1. ADMIN
2. USER

Module Description:

1. To implements this project we have used FGNET missing child dataset and by this dataset we have build and saved CNN model is available inside model folder.
1. 2.When user upload image used trained CNN model will be applied on test to check whether image is exists in missing child are not.
2. 3.when user found any suspected child roaming on road then user will take image and upload here and then CNN trained model will apply to get missing result. if unidentified image also upload then application say not found.

CHAPTER – 5 IMPLEMENTATION

5.1 ALGORITHM

CONVOLUTIONAL NEURAL NETWORK:

To demonstrate how to build a convolutional neural network based image classifier, we shall build a 6 layer neural network that will identify and separate one image from other. This network that we shall build is a very small network that we can run on a CPU as well. Traditional neural networks that are very good at doing image classification have many more parameters and take a lot of time if trained on normal CPU. However, our objective is to show how to build a real-world convolutional neural network using TENSORFLOW. Neural Networks are essentially mathematical models to solve an optimization problem. They are made of neurons, the basic computation unit of neural networks. A neuron takes an input (say x), do some computation on it (say: multiply it with a variable w and adds another variable b) to produce a value (say; z= wx+b). This value is passed to a non-linear function called activation function (f) to produce the final output(activation) of a neuron. There are many kinds of activation functions. One of the popular activation function is Sigmoid. The neuron which uses sigmoid function as an activation function will be called sigmoid neuron. Depending on the activation functions, neurons are named and there are many kinds of them like RELU, TanH. If you stack neurons in a single line, it's called a layer; which is the next building block of neural networks. See below image with layers

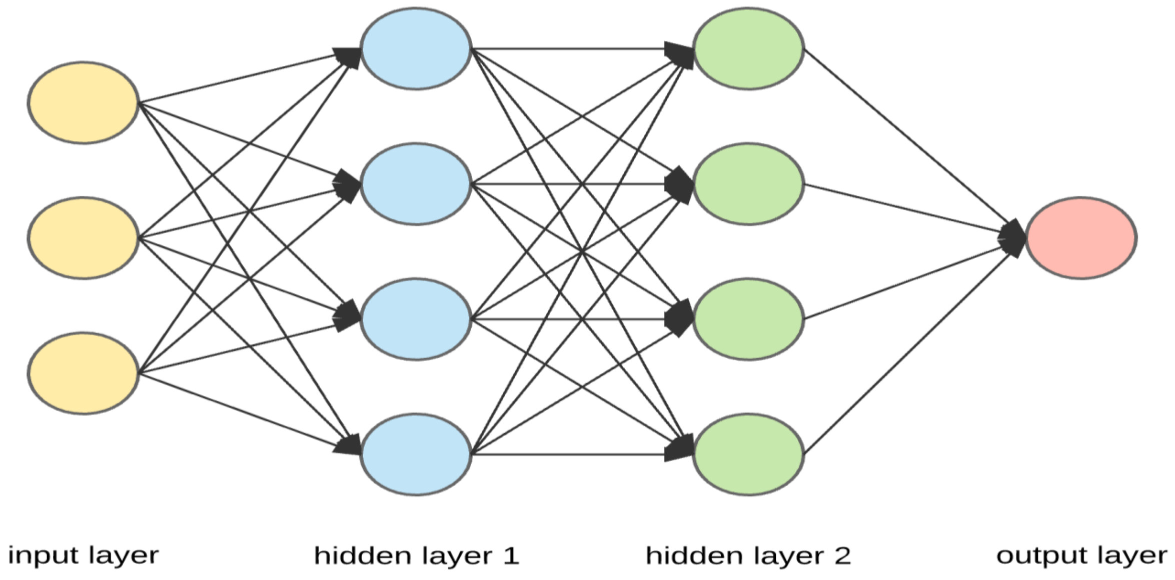


Fig : Layer in CNN

Key components of a Convolutional Neural Network include:

1. **Convolutional Layers:** These layers apply convolutional operations to input images, using filters (also known as kernels) to detect features such as edges, textures, and more complex patterns. Convolutional operations help preserve the spatial relationships between pixels.
2. **Pooling Layers:** Pooling layers downsample the spatial dimensions of the input, reducing the computational complexity and the number of parameters in the network. Max pooling is a common pooling operation, selecting the maximum value from a group of neighboring pixels.
3. **Activation Functions:** Non-linear activation functions, such as Rectified Linear Unit (ReLU), introduce non-linearity to the model, allowing it to learn more complex relationships in the data.
4. **Fully Connected Layers:** These layers are responsible for making predictions based on the high-level features learned by the previous layers. They connect every neuron in one layer to every neuron in the next layer.

CNNs are trained using a large dataset of labeled images, where the network learns to recognize patterns and features that are associated with specific objects or classes. Proven to be highly effective in image-related tasks, achieving state-of-the-art performance in various computer vision applications. Their ability to automatically learn hierarchical representations of features makes them well-suited for tasks where the spatial relationships and patterns in the data are crucial for accurate predictions. CNNs are widely used in areas such as image classification, object detection, facial recognition, and medical image analysis.

The convolutional layers are the key component of a CNN, where filters are applied to the input image to extract features such as edges, textures, and shapes.

The output of the convolutional layers is then passed through pooling layers, which are used to down-sample the feature maps, reducing the spatial dimensions while retaining the most important information. The output of the pooling layers is then passed through one or more fully connected layers, which are used to make a prediction or classify the image.

Convolutional Neural Network Design

- The construction of a convolutional neural network is a multi-layered feed-forward neural network, made by assembling many unseen layers on top of each other in a particular order.
- It is the sequential design that give permission to CNN to learn hierarchical attributes.

- In CNN, some of them followed by grouping layers and hidden layers are typically convolutional layers followed by activation layers.
- The pre-processing needed in a ConvNet is kindred to that of the related pattern of neurons in the human brain and was motivated by the organization of the Visual Cortex.

CONVOLUTIONAL NEURAL NETWORK TRAINING

CNNs are trained using a supervised learning approach. This means that the CNN is given a set of labeled training images. The CNN then learns to map the input images to their correct labels.

The training process for a CNN involves the following steps:

1. **Data Preparation:** The training images are preprocessed to ensure that they are all in the same format and size.
2. **Loss Function:** A loss function is used to measure how well the CNN is performing on the training data. The loss function is typically calculated by taking the difference between the predicted labels and the actual labels of the training images.
3. **Optimizer:** An optimizer is used to update the weights of the CNN in order to minimize the loss function.
4. **Backpropagation:** Backpropagation is a technique used to calculate the gradients of the loss function with respect to the weights of the CNN. The gradients are then used to update the weights of the CNN using the optimizer.

Applications of CNN

- **Image classification:** CNNs are the state-of-the-art models for image classification. They can be used to classify images into different categories, such as cats and dogs, cars and trucks, and flowers and animals.
- **Object detection:** CNNs can be used to detect objects in images, such as people, cars, and buildings. They can also be used to localize objects in images, which means that they can identify the location of an object in an image.
- **Image segmentation:** CNNs can be used to segment images, which means that they can identify and label different objects in an image. This is useful for applications such as medical imaging and robotics.

- **Video analysis:** CNNs can be used to analyze videos, such as tracking objects in a video or detecting events in a video. This is useful for applications such as video surveillance and traffic monitoring.

Advantages of CNN

- CNNs can achieve state-of-the-art accuracy on a variety of image recognition tasks, such as image classification, object detection, and image segmentation.
- CNNs can be very efficient, especially when implemented on specialized hardware such as GPUs.
- CNNs are relatively robust to noise and variations in the input data.
- CNNs can be adapted to a variety of different tasks by simply changing the architecture of the network.

Disadvantages of CNN

- CNNs can be complex and difficult to train, especially for large datasets.
- CNNs can require a lot of computational resources to train and deploy.
- CNNs require a large amount of labeled data to train.
- CNNs can be difficult to interpret, making it difficult to understand why they make the predictions they do.

VGG -16 ALGORITHM:

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous model submitted to ILSVRC-2014. Itmakes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16was trained for weeks and was using NVIDIA Titan Black GPU's.

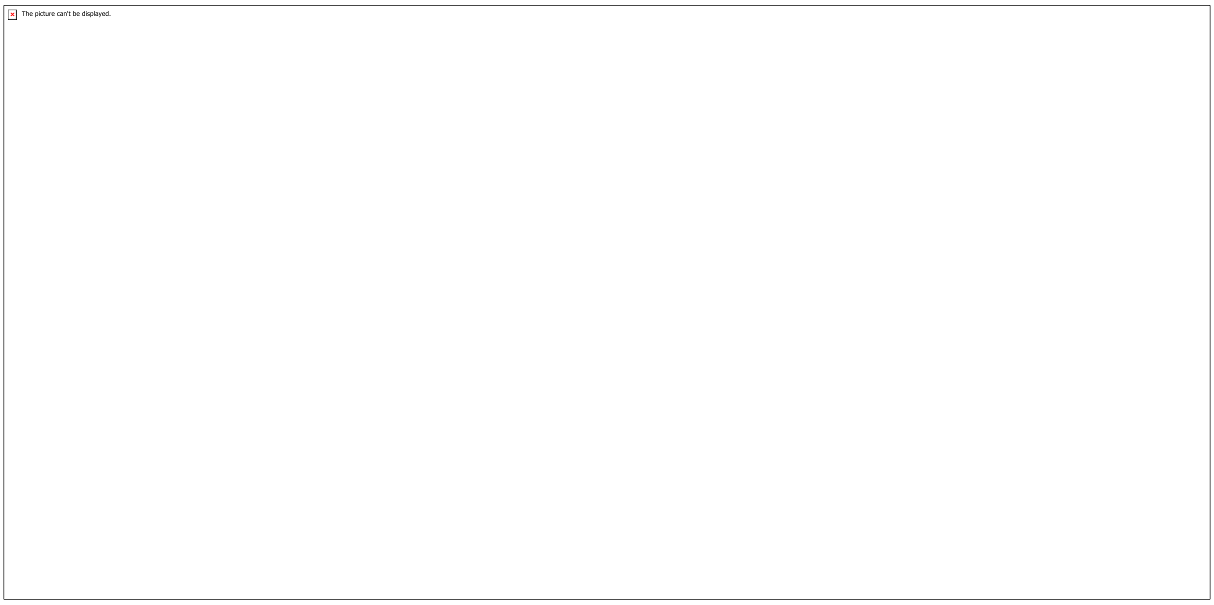


Fig : VGG-16 Model

VGG ARCHITECTURE:

The VGG-16 architecture is a deep convolutional neural network (CNN) designed for image classification tasks. It was introduced by the Visual Geometry Group at the University of Oxford. VGG-16 is characterized by its simplicity and uniform architecture, making it easy to understand and implement.

The VGG-16 configuration typically consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. These layers are organized into blocks, with each block containing multiple convolutional layers followed by a max-pooling layer for downsampling.

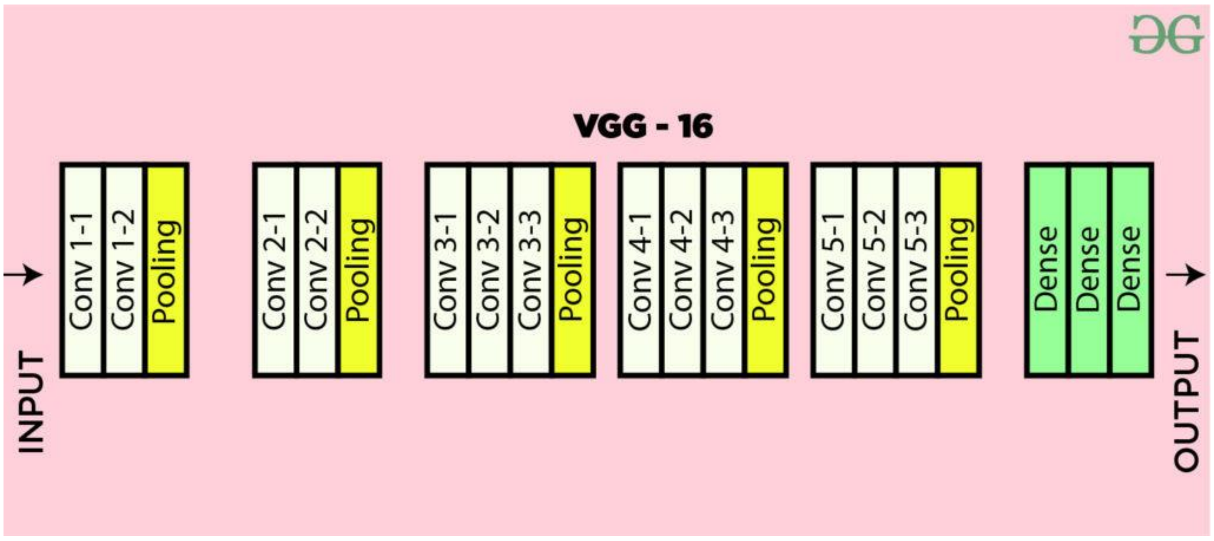


Fig : VGG-16 Architecture

Here’s a breakdown of the VGG-16 architecture based on the provided details:

1. Input Layer:
    - Input dimensions: (224, 224, 3)
2. Convolutional Layers (64 filters, 3×3 filters, same padding):
    - Two consecutive convolutional layers with 64 filters each and a filter size of 3×3.
    - Same padding is applied to maintain spatial dimensions.
3. Max Pooling Layer (2×2, stride 2):
    - Max-pooling layer with a pool size of 2×2 and a stride of 2.
4. Convolutional Layers (128 filters, 3×3 filters, same padding):
    - Two consecutive convolutional layers with 128 filters each and a filter size of 3×3.
5. Max Pooling Layer (2×2, stride 2):
    - Max-pooling layer with a pool size of 2×2 and a stride of 2.
6. Convolutional Layers (256 filters, 3×3 filters, same padding):
    - Two consecutive convolutional layers with 256 filters each and a filter size of 3×3.
7. Convolutional Layers (512 filters, 3×3 filters, same padding):
    - Two sets of three consecutive convolutional layers with 512 filters each and a filter size of 3×3.
8. Max Pooling Layer (2×2, stride 2):
    - Max-pooling layer with a pool size of 2×2 and a stride of 2.
9. Stack of Convolutional Layers and Max Pooling:
    - Two additional convolutional layers after the previous stack.
    - Filter size: 3×3.
10. Flattening:
    - Flatten the output feature map (7x7x512) into a vector of size 25088.
11. Fully Connected Layers:
    - Three fully connected layers with ReLU activation.
    - First layer with input size 25088 and output size 4096.
    - Second layer with input size 4096 and output size 4096.

- Third layer with input size 4096 and output size 1000, corresponding to the 1000 classes in the ILSVRC challenge.
- Softmax activation is applied to the output of the third fully connected layer for classification.

This architecture follows the specifications provided, including the use of ReLU activation function and the final fully connected layer outputting probabilities for 1000 classes using softmax activation.

MULTICLASS SVM:

A multiclass Support Vector Machine (SVM) is an extension of the traditional binary SVM to handle classification tasks with more than two classes. SVM is originally designed for binary classification, but several methods can be used to extend it to handle multiclass problems.

Here are a few common approaches to implementing multiclass SVM:

1. **One-vs-All (OvA) / One-vs-Rest (OvR)**:
   - In this approach, a separate binary SVM classifier is trained for each class, where the samples of that class are treated as positive examples and samples from all other classes are treated as negative examples. During prediction, the class with the highest confidence score (distance from the decision boundary) is chosen as the output class.
2. **One-vs-One (OvO)**:
   - In this approach, a binary SVM classifier is trained for every pair of classes. For N classes, this results in N(N-1)/2 classifiers. During prediction, each classifier 'votes' for a class, and the class that receives the most votes is chosen as the output class.
3. **Multiclass SVM Formulation**:
   - Some SVM formulations inherently support multiclass classification. For example, the Crammer-Singer method modifies the standard SVM optimization problem to directly optimize for a multiclass classification task.
4. **Hierarchical SVM**:
   - In this approach, a hierarchy of SVMs is constructed, where each SVM is responsible for distinguishing between a subset of classes. This approach is useful when the classes naturally form a hierarchical structure.

Each approach has its advantages and disadvantages. One-vs-All is simple and easy to implement, but it can suffer from class imbalance. One-vs-One requires more classifiers, but it can be more robust to class imbalance and is often more accurate. The choice of method often depends on the specific problem at hand and the characteristics of the dataset.

5.2. Sample Code

```
from django.shortcuts import render
from django.template import RequestContextimport pymysql
from django.http import HttpResponsefrom django.conf import settings
from django.core.files.storage import FileSystemStorageimport datetime
import os import cv2
import numpy as np
from keras.utils.np_utils import to_categoricalfrom keras.layers import MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flattenfrom keras.layers import
Convolution2D
from keras.models import Sequential
from keras.models import model_from_json

global indexindex = 0
global missing_child_classifierglobal cascPath
global faceCascade
def index(request):
if request.method == 'GET':
return render(request, 'index.html', {})

def Login(request):
if request.method == 'GET':
return render(request, 'Login.html', {})def Upload(request):
```

```
if request.method == 'GET':
return render(request, 'Upload.html', {})
def OfficialLogin(request):
if request.method == 'POST':
username = request.POST.get('t1', False)password = request.POST.get('t2', False)
if username == 'admin' and password == 'admin':
context= {'data':'welcome '+username}
return render(request, 'OfficialScreen.html', context)else:
context= {'data':'login failed'}
return render(request, 'Login.html', context)
def ViewUpload(request):
if request.method == 'GET':
strdata = '<table border=1 align=center width=100%><tr><th>Upload Person
Name</th><th>Child
Name</th><th>Contact No</th><th>Found Location</th><th>Child Image <th>Uploaded
Date</th><th>Status</th></tr><tr>'
con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database
=
'MissingChildDB',charset='utf8')
with con:
cur = con.cursor()
cur.execute("select * FROM missing")rows = cur.fetchall()
for row in rows:
strdata+='<td>'+row[0]+'</td><td>'+str(row[1])+'</td><td>'+row[2]+'</td><td>'+row[3]+'
</td><td><i
mg src=/static/images/'+row[4]+' width=200 height=200></img></td><td>'
strdata+=str(row[5])+'</td><td>'+str(row[6])+'</td></tr>'context= {'data':strdata}
return render(request, 'ViewUpload.html', context)
def UploadAction(request):global index
global missing_child_classifierglobal cascPath
global faceCascade
if request.method == 'POST' and request.FILES['t5']:
output = ''
```

```
person_name = request.POST.get('t1', False)child_name = request.POST.get('t2', False)
contact_no =
request.POST.get('t3', False) location = request.POST.get('t4', False) myfile =
request.FILES['t5']
fs = FileSystemStorage()
filename = fs.save('C:/Users/ssnaik/Desktop/Missing Child Identification System using Deep
Learningand Multiclass SVM/MissingChildApp/static/images/'+child_name+'.png', myfile)
#if index == 0:
cascPath = "haarcascade_frontalface_default.xml" faceCascade =
cv2.CascadeClassifier(cascPath)
#index = 1
option = 0;
frame = cv2.imread(filename)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY) faces =
faceCascade.detectMultiScale(gray,1.3,5) print("Found {0} faces!".format(len(faces)))
img = ''
status = 'Child not found in missing database'if len(faces) > 0:
for (x, y, w, h) in faces:
img = frame[y:y + h, x:x + w]option = 1
if option == 1:
with open('model/model.json', "r") as json_file:loaded_model_json = json_file.read()
missing_child_classifier = model_from_json(loaded_model_json)
missing_child_classifier.load_weights("model/model_weights.h5")
missing_child_classifier._make_predict_function()img = cv2.resize(img, (64,64))
im2arr = np.array(img)
im2arr = im2arr.reshape(1,64,64,3)img = np.asarray(im2arr)
img = img.astype('float32')img = img/255
preds = missing_child_classifier.predict(img)if(np.amax(preds) > 0.60):
status = 'Child found in missing database'now = datetime.datetime.now()
current_time = now.strftime("%Y-%m-%d %H:%M:%S")filename =
os.path.basename(filename)
db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password =
'root', database
='MissingChildDB',charset='utf8')
```

```
db_cursor = db_connection.cursor()
query = "INSERT INTO
missing(person_name,child_name,contact_no,location,image,upload_date,status)
VALUES('"+person_name+"','"+child_name+"','"+contact_no+"','"+location+"','"+filename
+"','"+str(cu
rrent_ time)+"','"+status+"')"
db_cursor.execute(query)  db_connection.commit()  print(db_cursor.rowcount,  "Record
Inserted")
context= {'data':'Thank you for uploading. '+status}return render(request, 'Upload.html',
context)
```

Python Execution Code

```
#!/usr/bin/env python
import os
import sys
 if _name_ == '_main_':
        os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'MissingChild.settings')
        try:
            from django.core.management import execute_from_command_line
        except ImportError as exc:
               raise ImportError(
                     "Couldn't import Django. Are you sure it's installed and "
                     "available on your PYTHONPATH environment variable? Did you "
                     "forget to activate a virtual environment?"
                ) from exc
         execute_from_command_line(sys.argv)
```

CHAPTER – 6 SYSTEM TESTING

6.1 TESTING

Software testing is a critical element of software quality assurance and represents the ultimate reviews of specification, design and coding. Testing presents an interesting anomaly of the software. During earlier definition and development phases, it was attempted to build software from abstract concept to a tangible implementation.

The testing phase involves the testing of the developed system using various set data. Presentation of test data plays a vital role in system testing. After preparing the test data the system under study was tested using test data. While testing the system by using test data errors were found and corrected. A series of tests were performed for the proposed system before the system was ready for implementation. The various types of testing done on the system are:

- Unit Testing
- Integration Testing
- User Acceptance Testing
- System Testing

6.1.1 UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software design, the module. It comprises the set of test performed by the programmer prior to integration of the unit into larger system. The testing was carried out during the coding stage itself. In this step each module is found to be working satisfactorily as regards to the expected output from the module**.**

Each form is treated as a unit and tested thoroughly for bugs. The following is a list of some of the test cases :

1) In the login form, if a member does not enter a value for userId and password, then the user is prompted with the error message “userId and password should not be blank”.

2) In the login form, if a member enters wrong values for userId and password, then the user is prompted with the error message “Invalid userId and password. Try again.”.

3) In book Entry screen and new student, teacher screen, all the fields should have a value. Otherwise, the user is prompted with an appropriate error messages.

4) In book transactions form, member id, book no,. issue date, and return date are mandatory. If not provided, then the system will prompt the user with the error message “Fields should not be blank”.

6.1.2 INTEGRATION TESTING

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover error associated within the interface. The objective is to take unit tested modules and build a program structure that has been dictated by design. All modules are combined in this step. The entire program is tested as whole. And chaos in interfaces may usually result. A set of errors is encountered in such a case.

The integration testing can be carried out using two methodologies:

1. Top Down Integration
2. Bottom Up Integration

The first one is done where integration is carried out by addition of major modules to minor modules. While Bottom Up integration follows combination of smaller ones to larger one. Here, Bottom Up Integration is followed. Even though correction was difficult because the isolation of causes is complicated by the vastness of the entire program, all the errors found in the system were corrected and then forwarded to the next testing steps.

The navigation among all the screens have been thoroughly verified so that the user of the system can move from one form to another form.

The connectivity between the forms and the database has been checked. In case of any malfunctions, the user will be informed about the problem.

6.1.3 USER ACCEPTANCE TESTING

User acceptance of a system is the key factor for the success of any system. The system under consideration was tested for users acceptance by constantly keeping in touch with the perspective system user at the time of developing and making changes wherever required. This is done with the regards to the following points:

A system may be defined as a set of instructions combined in the same form and directed to some purpose.

Before any development is undertaken certain specifications are prepared which objectively describe the application system. The System specifications are made after consulting the end user managers of the relevant departments.

Software to be developed is planned on the basis of requirement of the user. The problem definition statement description of present situation and goal to be achieved by news system.

The success of system depends on how accurately a problem is defined, thoroughly investigated carried out through choice of solution. User need identification and analysis that are concerned with what the uses needs rather than what he/she wants. System explains how to perform specific activities or task, which does what and what.

6.1.4 SYSTEM TESTING

Testing the behavior of the whole software/system as defined in software requirements specification(SRS) is known as system testing, its main focus is to verify that the customer requirements are fulfilled.

System testing is done after integration testing is complete. System testing should test functional and non functional requirements of the software. The test types followed in system testing differ from organization to organization.

6.2. TESTCASES

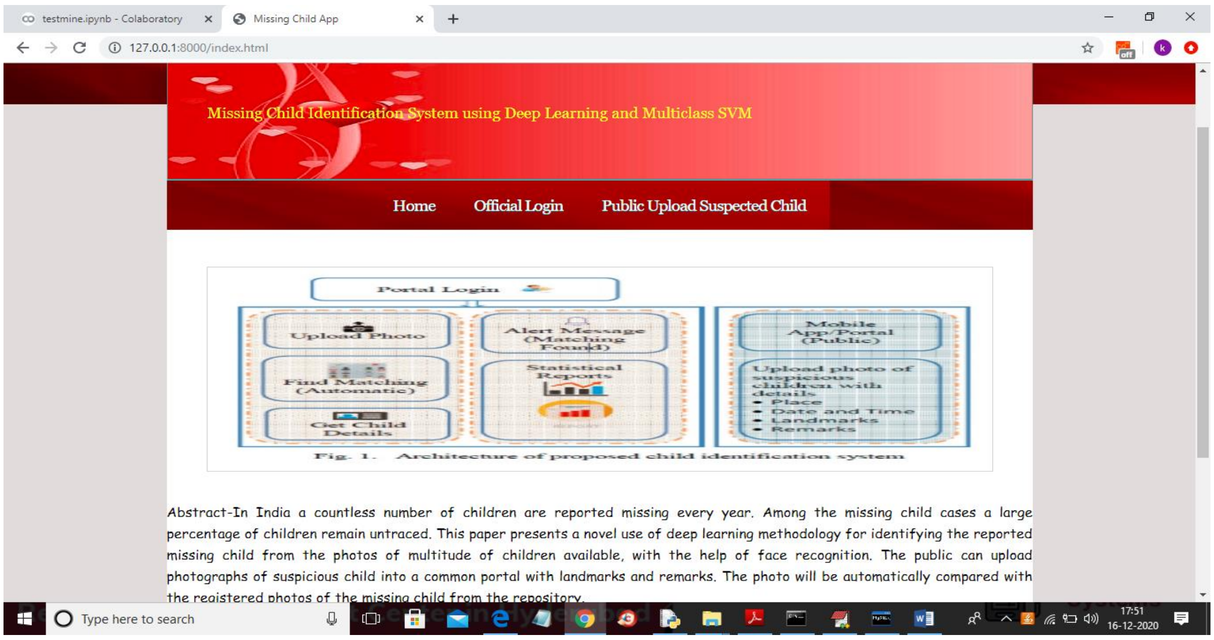
6.2.1.Test case for Login form:

FUNCTION:	**LOGIN**
EXPECTED RESULTS:	Should Validate the user and check his existence in database
ACTUAL RESULTS:	Validate the user and checking the user against the database
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

6.2.2. Test case for User Registration form:

FUNCTION:	**Public Upload Suspected Child**
EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

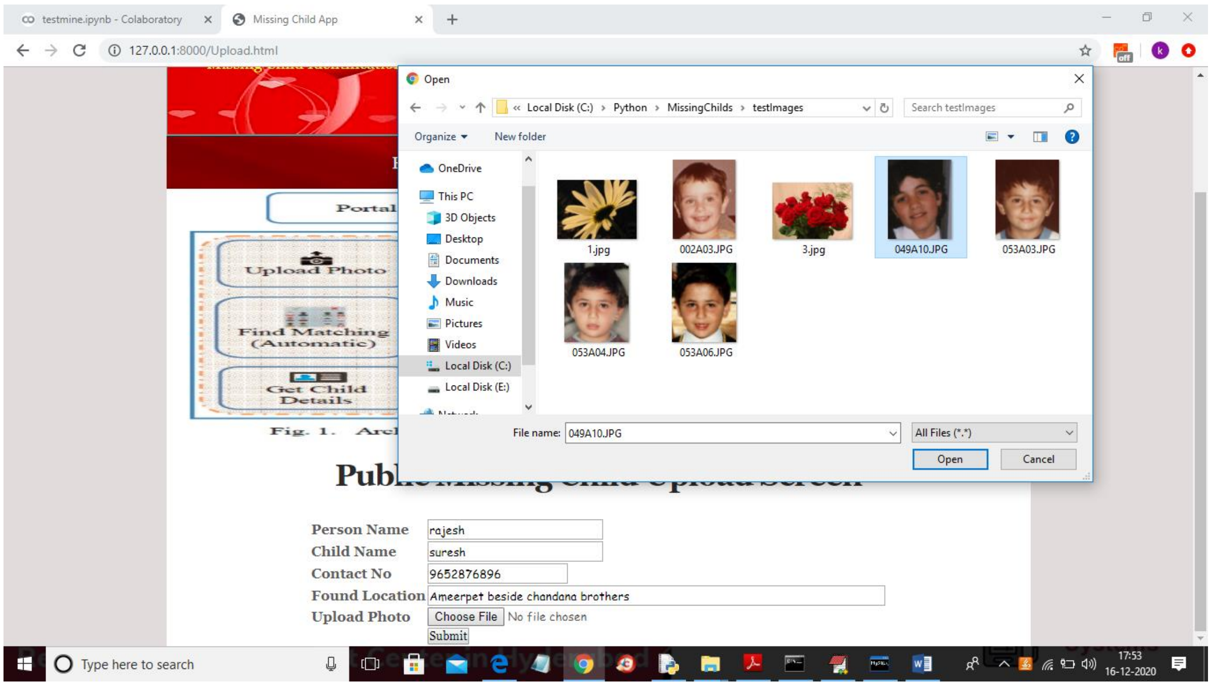
CHAPTER 7 : OUTPUT SCREENS


testmine.ipynb - Colaboratory
Missing Child App
127.0.0.1:8000/index.html
Missing Child Identification System using Deep Learning and Multiclass SVM
Home
Official Login
Public Upload Suspected Child
Type here to search
17:51
16-12-2020

Abstract-In India a countless number of children are reported missing every year. Among the missing child cases a large percentage of children remain untraced. This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository.

Fig: Homepage

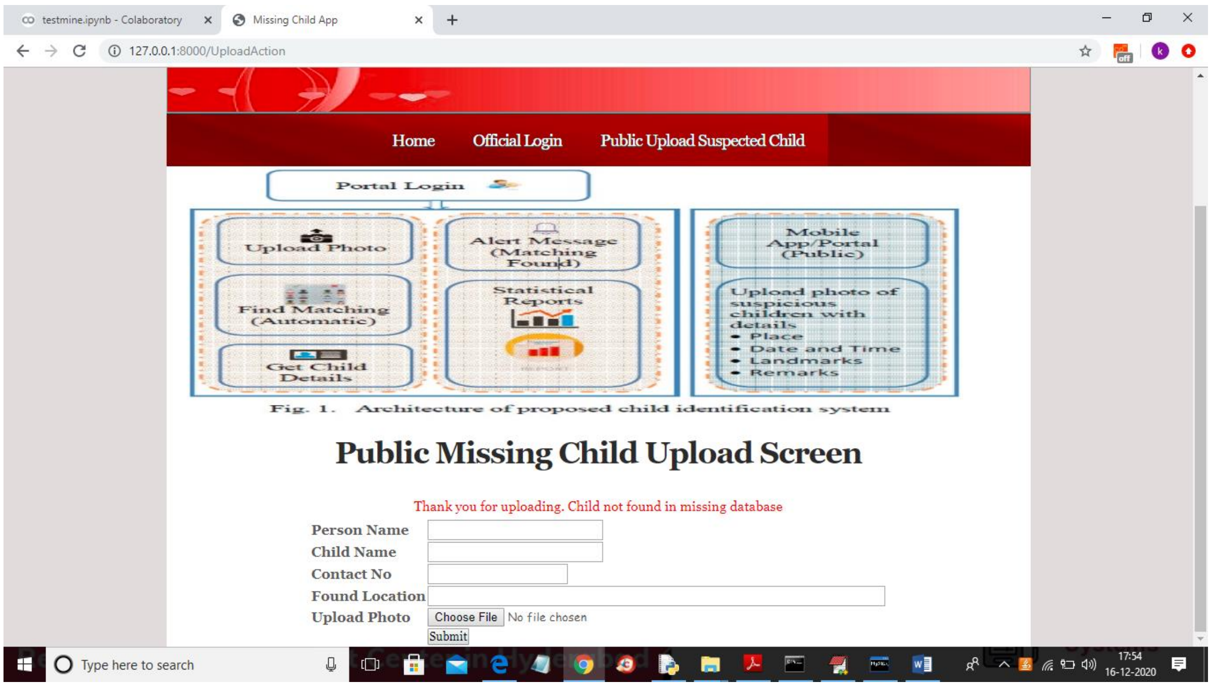
In above screen public can click on ‘Public Upload Suspected Child’ link to get below page and to add missing child details.



File name:	049A10.JPG
Person Name	rajesh
Child Name	suresh
Contact No	9652876896
Found Location	Ameerpet beside chandana brothers
Upload Photo	Choose File No file chosen

Submit

In above screen public will enter suspected child details and then upload photo and then click on ‘Submit’ button and to get below result.

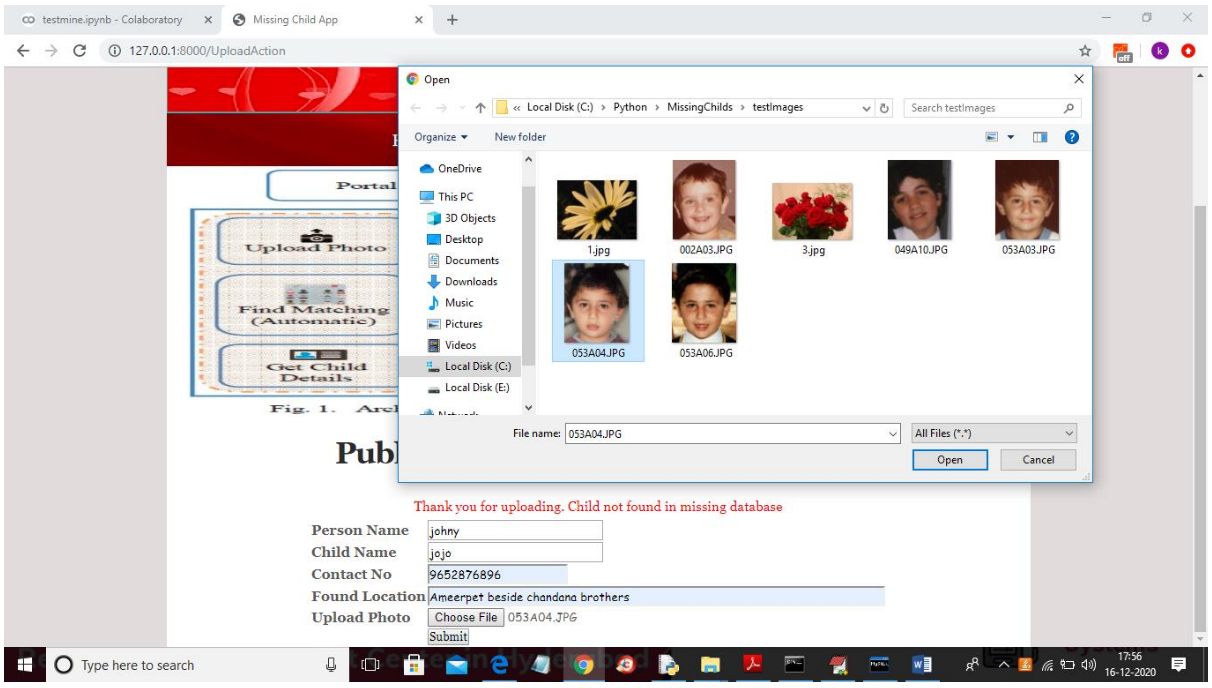

testmine.ipynb - Colaboratory
Missing Child App
127.0.0.1:8000/UploadAction
Home
Official Login
Public Upload Suspected Child
Portal Login
Upload Photo
Alert Message (Matching Found)
Mobile App/Portal (Public)
Find Matching (Automatic)
Statistical Reports
Upload photo of suspicious children with details
Place
Date and Time
Landmarks
Remarks
Get Child Details
Fig. 1. Architecture of proposed child identification system
Choose File
Submit
Type here to search
17:54
16-12-2020

Public Missing Child Upload Screen

Thank you for uploading. Child not found in missing database

Person Name	
Child Name	
Contact No	
Found Location	
Upload Photo	No file chosen

In above screen we can see child not found in missing DB and we can try with other image.

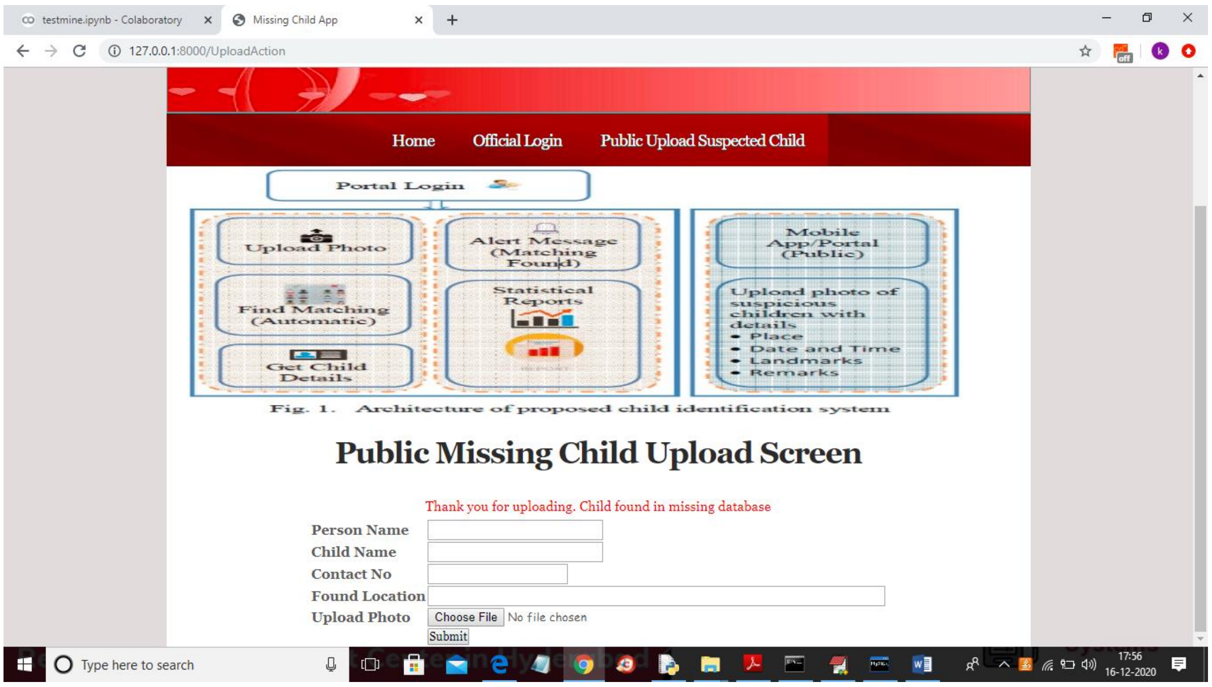


053A04.JPG

Thank you for uploading. Child not found in missing database

Person Name	johny
Child Name	jojo
Contact No	9652876896
Found Location	Ameerpet beside chandana brothers
Upload Photo	053A04.JPG

And below is the result for new above child details



testmine.ipynb - Colaboratory | Missing Child App

127.0.0.1:8000/UploadAction

Home | Official Login | Public Upload Suspected Child

Portal Login
Upload Photo
Alert Message (Matching Found)
Mobile App/Portal (Public)
Find Matching (Automatic)
Statistical Reports
Upload photo of suspicious children with details
• Place
• Date and Time
• Landmarks
• Remarks
Get Child Details

Fig. 1. Architecture of proposed child identification system

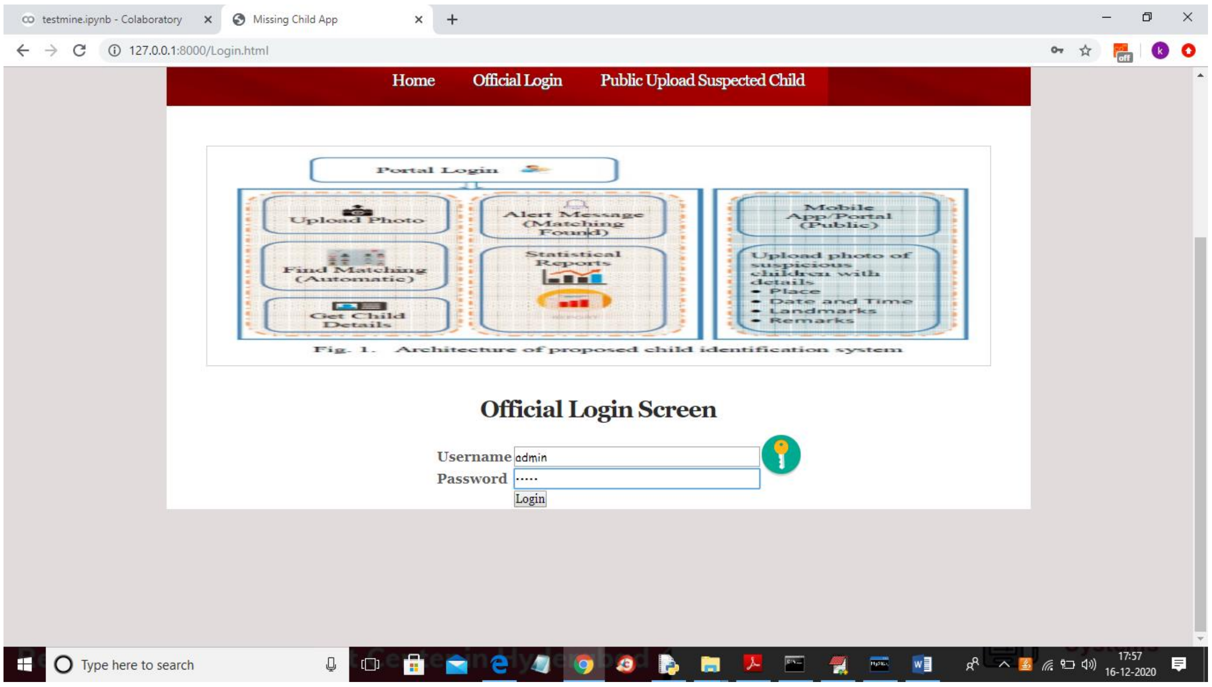
Public Missing Child Upload Screen

Thank you for uploading. Child found in missing database

Person Name
Child Name
Contact No
Found Location
Upload Photo Choose File No file chosen

Submit

In above screen uploaded child found in database and now click on 'Official Login' link to get below login screen.



testmine.ipynb - Colaboratory | Missing Child App

127.0.0.1:8000/Login.html

Home | Official Login | Public Upload Suspected Child

Portal Login
Upload Photo
Alert Message (Matching Found)
Mobile App/Portal (Public)
Find Matching (Automatic)
Statistical Reports
Upload photo of suspicious children with details
• Place
• Date and Time
• Landmarks
• Remarks
Get Child Details

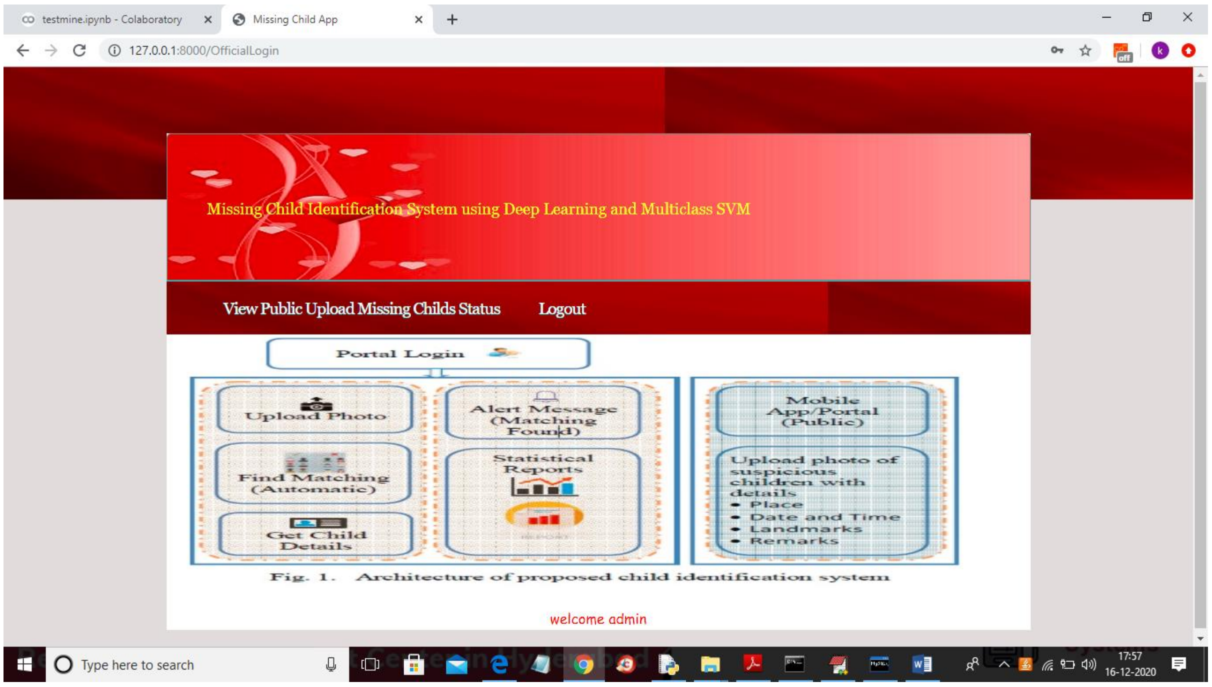
Fig. 1. Architecture of proposed child identification system

Official Login Screen

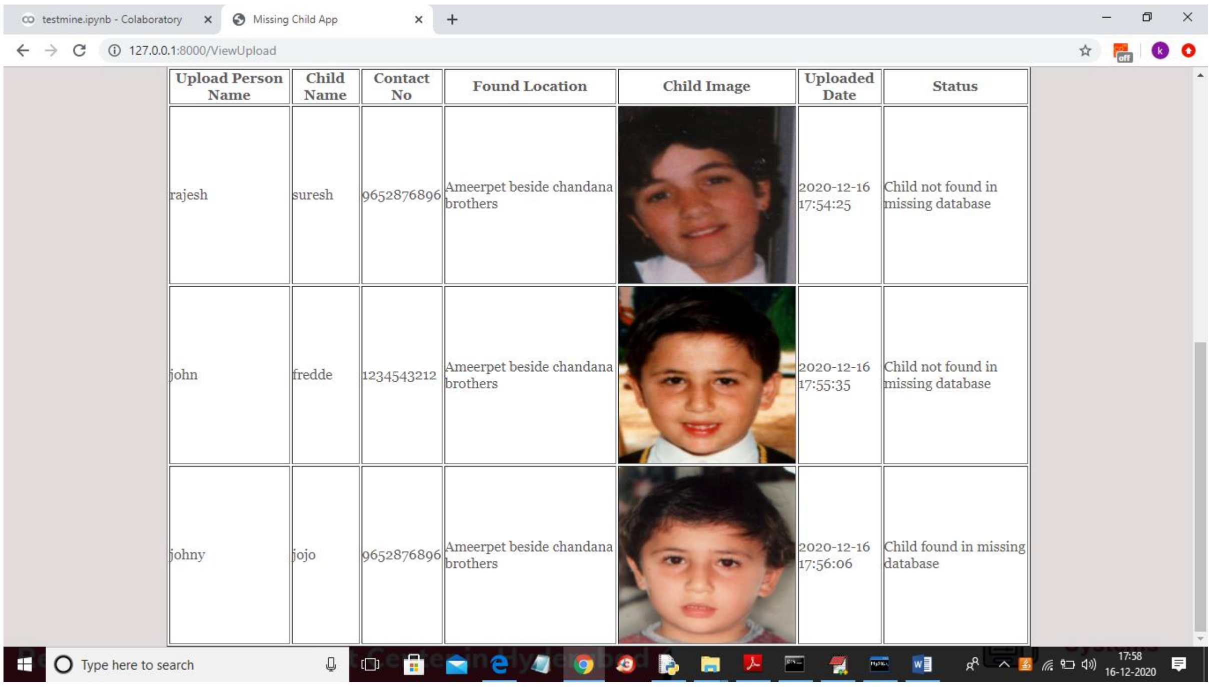
Username admin
Password •••••

Login

In above screen admin can login by entering username and password as 'admin' and 'admin' and after clicking on 'Login' button will get below screen.


testmine.ipynb - Colaboratory
Missing Child App
127.0.0.1:8000/OfficialLogin
Missing Child Identification System using Deep Learning and Multiclass SVM
View Public Upload Missing Childs Status
Logout
Portal Login
Upload Photo
Alert Message (Matching Found)
Mobile App/Portal (Public)
Find Matching (Automatic)
Statistical Reports
Upload photo of suspicious children with details
Place
Date and Time
Landmarks
Remarks
Get Child Details
Fig. 1. Architecture of proposed child identification system
welcome admin
Type here to search
17:57
16-12-2020

In above screen official can click on ‘View Public Upload Missing Childs Status’ link to view all uploads and its result done by public .



Upload Person Name	Child Name	Contact No	Found Location	Child Image	Uploaded Date	Status
rajesh	suresh	9652876896	Ameerpet beside chandana brothers		2020-12-16 17:54:25	Child not found in missing database
john	fredde	1234543212	Ameerpet beside chandana brothers		2020-12-16 17:55:35	Child not found in missing database
johny	jojo	9652876896	Ameerpet beside chandana brothers		2020-12-16 17:56:06	Child found in missing database

In above screen officials can see all details and then take action to find that child.

CHAPTER – 8 CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION :

A missing child identification system is proposed, which combines the powerful CNN based deep learning approach for feature extraction and support vector machine classifier for classification of different child categories. This system is evaluated with the deep learning model which is trained with feature representations of children faces. By discarding the soft max of the VGG-Face model and extracting CNN image features to train a multi class SVM, it was possible to achieve superior performance. Performance of the proposed system is tested using the photographs of children with different lighting conditions, noises and also images at different ages of children. The classification achieved a higher accuracy of 99.41% which shows that the proposed methodology of face recognition could be used for reliable missing children identification.

FUTURE SCOPE:

Future Enhancement is being planned to further analyze and enhance the protocol to identify missing child with advance features image classification. The future scope for missing child identification using deep learning and multi-class SVM is expansive, with opportunities for innovation and collaboration across various domains. By harnessing the power of advanced technologies and fostering interdisciplinary partnerships, we can make significant strides in safeguarding children and reuniting families.

1. **Advanced Deep Learning Architectures**: Continued research into deep learning architectures, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models, could lead to even more powerful models for missing child identification. These architectures can be tailored to handle specific challenges, such as age progression, changes in appearance, and variations in image quality.
2. **Semantic Segmentation and Object Detection**: Integrating techniques like semantic segmentation and object detection can enable finer-grained analysis of images, allowing for the identification of specific features or objects associated with missing children.

This could include identifying unique clothing, accessories, or distinguishing physical characteristics.

3. **Multi-Modal Learning**: Incorporating additional modalities such as text (e.g., descriptions, reports) and audio (e.g., descriptions, background noises) alongside visual data can provide a more comprehensive understanding of missing children cases. Multi-modal learning approaches can leverage the complementary nature of different data types to improve identification accuracy.

REFERENCES

1. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", Nature, 521(7553):436–444, 2015.
2. O. Deniz, G. Bueno, J. Salido, and F. D. la Torre, "Face recognition using histograms of oriented gradients", Pattern Recognition Letters, 32(12):1598–1603, 2011.
3. C. Geng and X. Jiang, "Face recognition using sift features", IEEE International Conference on Image Processing(ICIP), 2009.
4. Rohit Satle, Vishnuprasad Poojary, John Abraham, Shilpa Wakode, "Missing child identification using face recognition system", International Journal of Advanced Engineering and Innovative Technology (IJAEIT), Volume 3 Issue 1 July - August 2016.
5. Simonyan, Karen and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition", International Conference on Learning Representations (ICLR), April 2015.
6. O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," in British Machine Vision Conference, vol. 1, no. 3, pp. 1-12, 2015.
7. A. Vedaldi, and K. Lenc, "MatConvNet: Convolutional Neural Networks for MATLAB", ACM International Conference on Multimedia, Brisbane, October 2015.