A

Mini Project

On

MISSING CHILD IDENTIFICATION SYSTEM USING DEEP LEARNING AND MULTICLASS SVM

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

B. VEDIKA (207R1A0569)

A. HARSHITHA NAIDU (207R1A0561)

B. ASHOK (207R1A0565)

Under the Guidance of

Dr. J. NARASIMHARAO

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"MISSING CHILD IDENTIFICATION SYSTEM USING DEEP LEARNING AND MULTICLASS SVM"** being submitted by **B. VEDIKA(207R1A0569), A. HARSHITHA NAIDU(207R1A0561) & B. ASHOK (207R1A0565)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. J. Narasimharao
(Associate Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Dr. J. Narasimharao,** Associate Professor for his exemplary guidance, monitoring andconstant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G.Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa,& Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

B. VEDIKA (207R1A0569)

A. HARSHITHA NAIDU (207R1A0561)

B. ASHOK (207R1A0565)

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. The classification performance achieved for child identification system is 99.41%. It was evaluated on 43 Child cases.

.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT 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.

1.2 PROJECT PURPOSE

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..

1.3 PROJECT FEATURES

The main features of this project is to develop 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 Convolutional Neural Networks (CNNs) are essential tools for deep learning methods and are more appropriate for working with image data . A very deep CNN called VGG-Face network is used for face recognition.

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst hasa firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

A general statement of missing child identification problem can be formulated as the given a child missing , identify the child using a stored database of those authorised faces.

2.2 EXISTING SYSTEM

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 .

2.2.1 LIMITATIONS OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- 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, half of them remain untrace
- Huge storage requirements.
- Potential privacy issues.

2.3 PROPOSED SYSTEM

Whenever public uploads photo of a suspected child, the system generates template vector of the facial features from the uploaded photo. If a matching is found in the repository, the system displays the most matched photo and pushes a message to the concerned Officer portal or SMSs the alert message of matching child. Similarly the Officer can check for any matching with the database at any time using the proposed system.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- The proposed system is comparatively an easy,inexpensive and reliable method compared to other biometrics like fingerprint and iris recognition systems.
- The use of multiclass SVM in the system improves the accuracy of the classification and reduce false positives.
- Easy to integrate

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication that the system is economically possible for development.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- Processor : Pentium IV or higher
- Hard disk : minimum 512 MB
- RAM : 256 MB
- Input devices : Keyboard, mouse.

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

- Operating system : Windows 8 and above
- Languages : Python, Html, CSS, Django, MySql
- Tools : Python IDEL3.7 version, WampServer 2.4

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

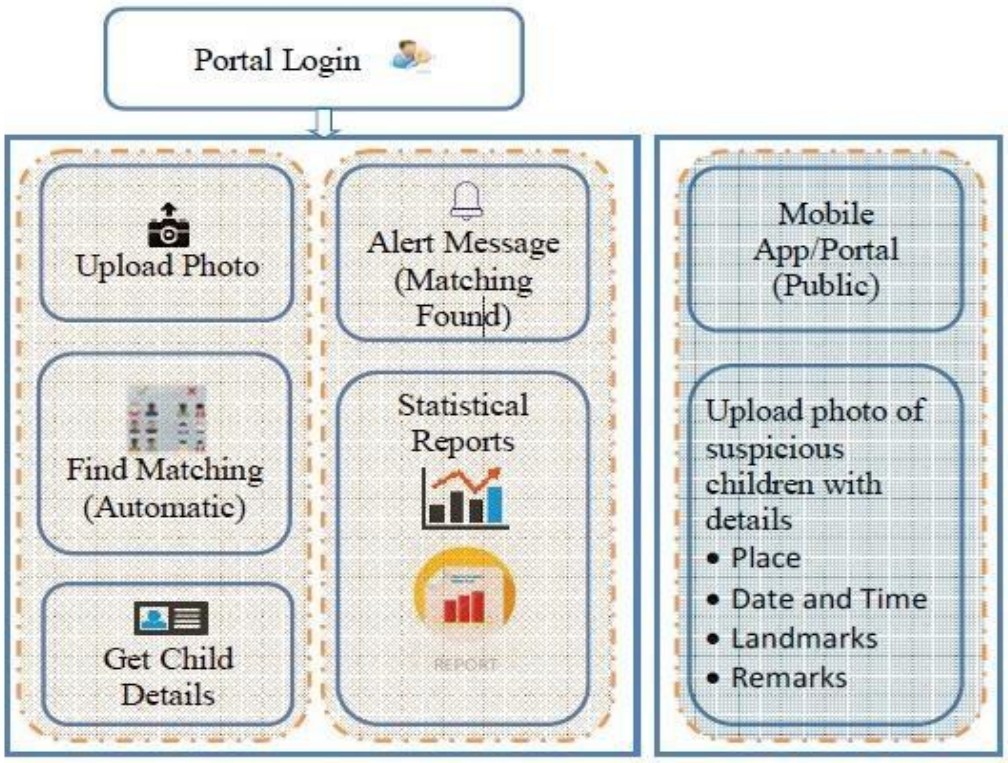


Figure 3.1: Project Architecture for Missing child identification system using Deep learning and multiclass SVM

3.2 DESCRIPTION

Here we propose a 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. 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. Using public dataset of missing children's called FGNET is used to train deep learning CNN prediction model. After training model whenever public upload any suspected child image then this model will check in trained model to detect whether this child is in missing database or not. This detected result will store in database and whenever want official persons will login and see that detection result. SVM Multiclass classifier use to extract face features from images based on age and other facial features and then this detected face will input to CNN model to predict whether this face child exists in image database or not.

.

3.3 USE CASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of usersthe system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

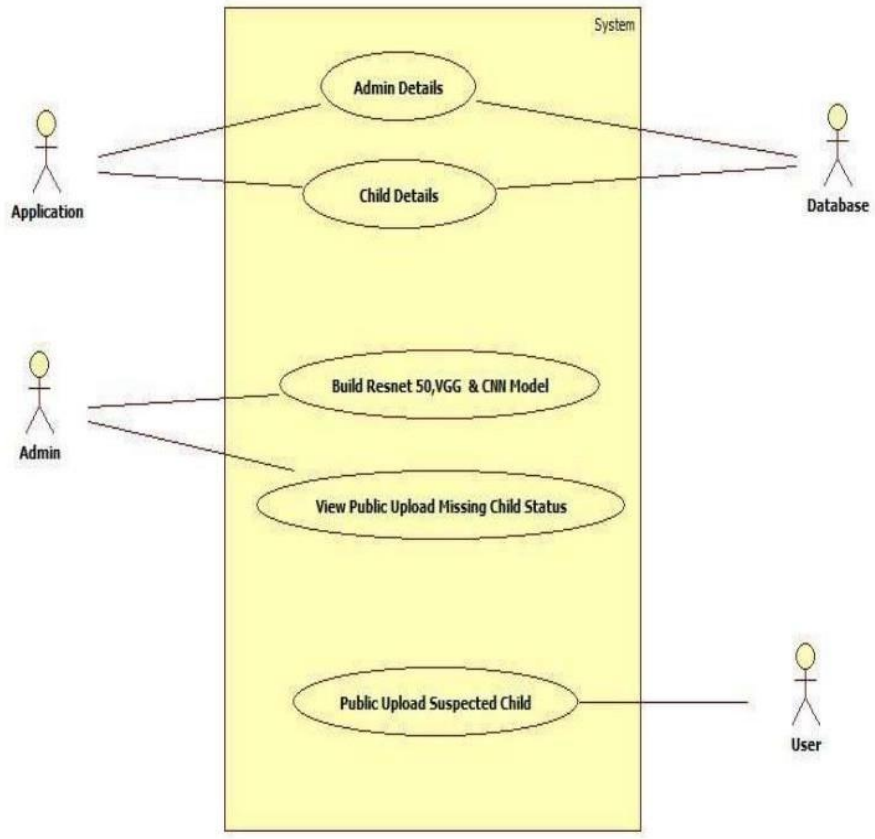


Figure 3.2: Use Case Diagram for Missing child identification system using deep learning and multiclass SVM

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations(or methods), and the relationships among objects.

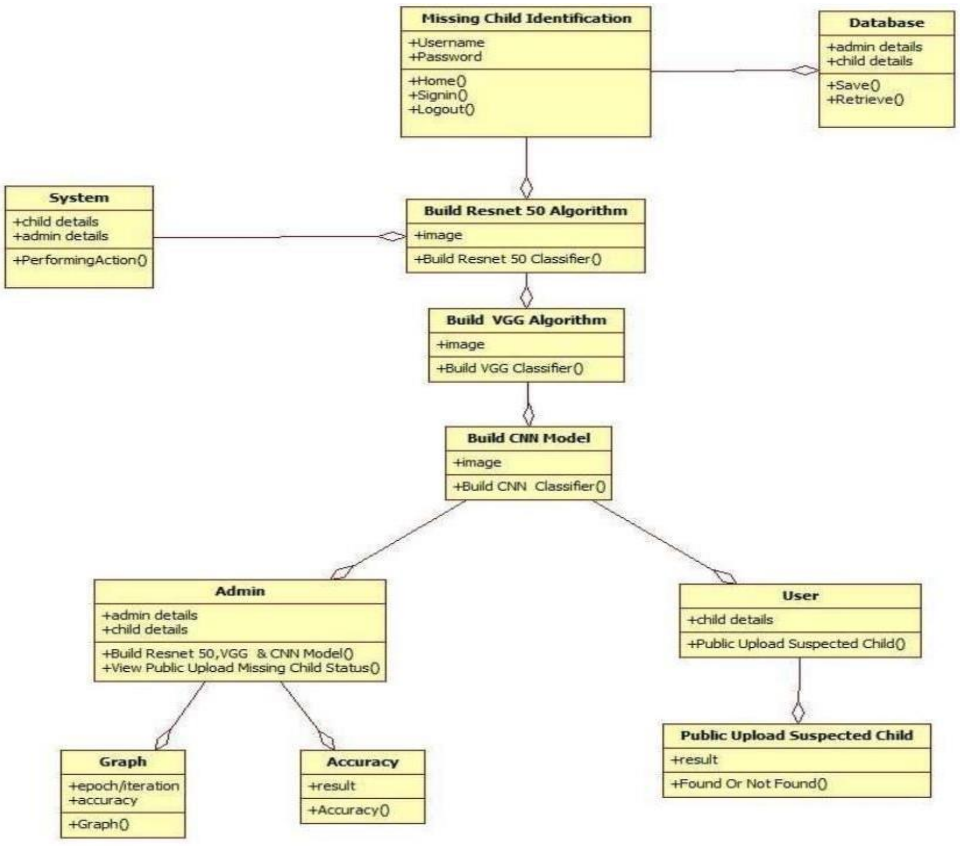


Figure 3.3: Class Diagram for Missing child identification system using deep learning and multiclass SVM

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

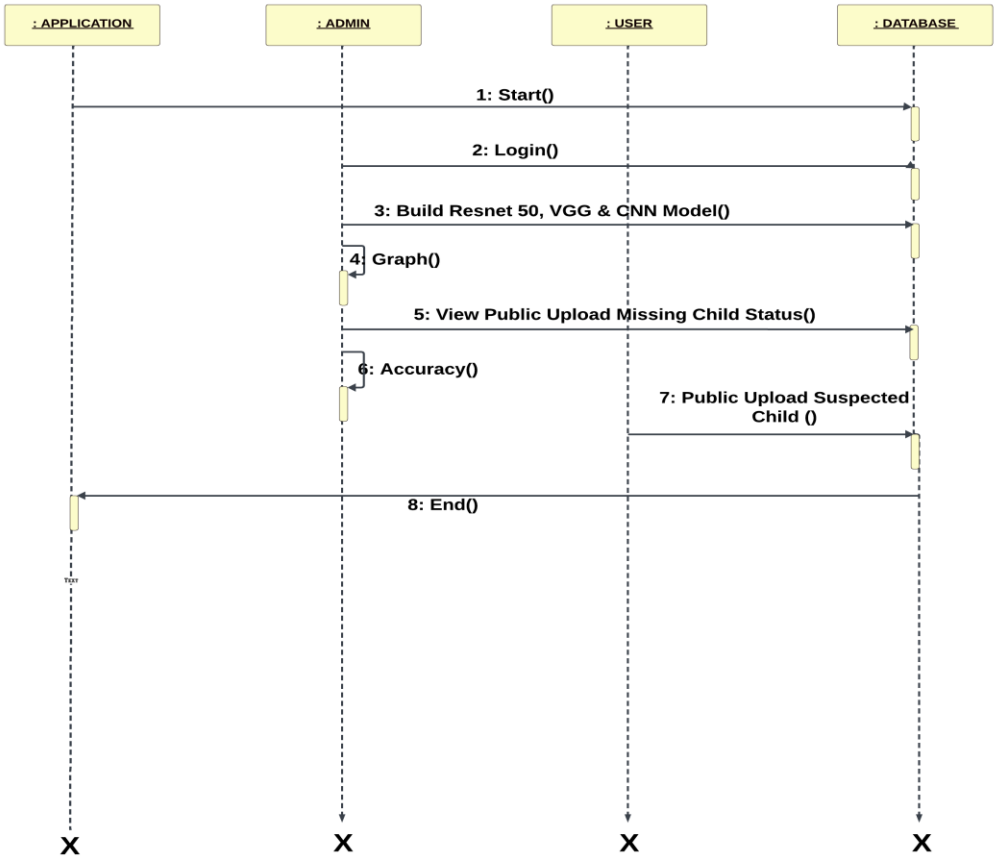


Figure 3.4: Sequence Diagram for Missing child identification system using deep learning and multiclass SVM

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

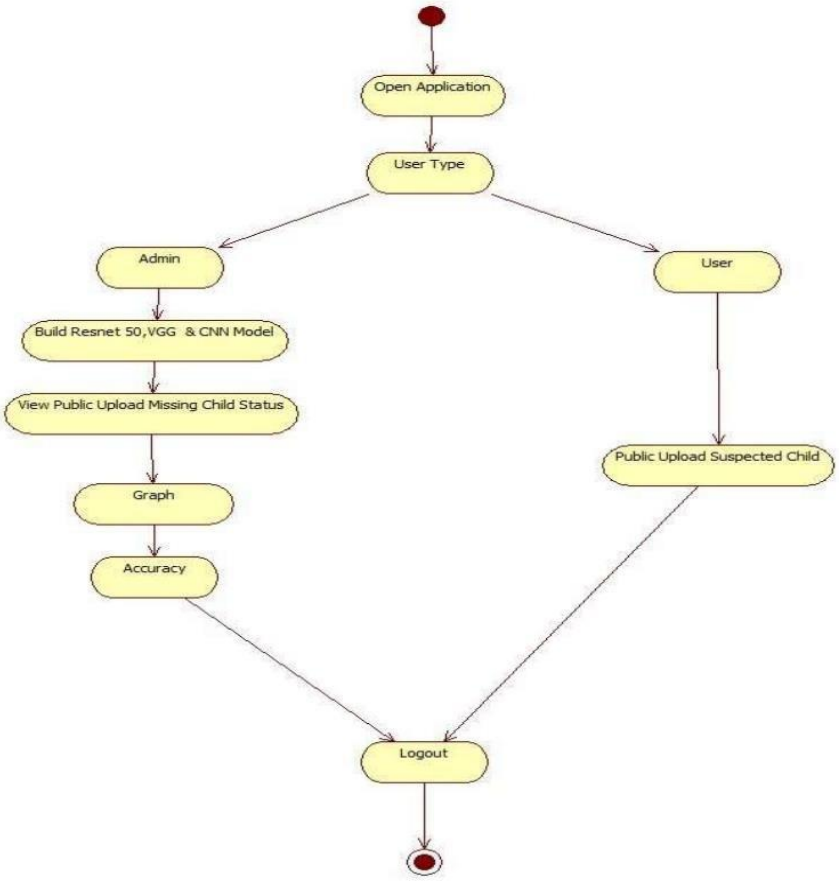


Figure 3.5: Activity Diagram for Missing child identification system using deep learning and multiclass SVM

4. IMPLEMENTATION

4.1 SAMPLE CODE

```
from django.shortcuts import render
from django.template import RequestContext
import pymysql
from django.http import HttpResponse
from django.conf import settings
from django.core.files.storage import FileSystemStorage
import datetime
import os
import cv2
import numpy as np
from keras.utils.np_utils import to_categorical
from keras.layers import  MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D
from keras.models import Sequential
from keras.models import model_from_json
import datetime

global index
index = 0
global missing_child_classifier
global 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 WelfareLogin(request):
    if request.method == 'GET':
      return render(request, 'WelfareLogin.html', {})


def WelfareLoginAction(request):
    if request.method == 'POST':
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        if username == 'welfare' and password == 'welfare':
            context= {'data':'welcome '+username}
            return render(request, 'WelfareScreen.html', context)
```

```
        else:
            context= {'data':'login failed'}
            return render(request, 'WelfareLogin.html', context)


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


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


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


def  ParentLoginAction(request):
    if request.method == 'POST':
      username = request.POST.get('t1', False)
      password = request.POST.get('t2', False)
      index = 0
      con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = '',
                            database = 'MissingChildDB',charset='utf8')
      with con:
            cur = con.cursor()
            cur.execute("select * FROM parentsignup")
            rows = cur.fetchall()
            for row in rows:
               if row[0] == username and password == row[1]:
                  index = 1
                  break
      if index == 1:
          file = open('session.txt','w')
          file.write(username)
          file.close()
          context= {'data':'welcome '+username}
          return render(request, 'ParentScreen.html', context)
      else:
          context= {'data':'login failed'}
          return render(request, 'ParentLogin.html', context)

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

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

```
def checkImage(name):
    index = 0
    con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = '',
database = 'MissingChildDB',charset='utf8')
    with con:
        cur = con.cursor()
        cur.execute("select childname FROM adoption")
        rows = cur.fetchall()
        for row in rows:
        if row[0] == name:
            index = 1
            break
return index


def getDetails(name):
    parent = ''
    age = ''
    occupation = ''
    contact = ''
    email = ''
    address = ''
    con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = '',
database = 'MissingChildDB',charset='utf8')
    with con:
        cur = con.cursor()
        cur.execute("select * FROM parentsignup")
        rows = cur.fetchall()
        for row in rows:
            if row[0] == name:
                parent = row[2]
                age = row[3]
                occupation = row[4]
                contact = row[5]
                email = row[6]
                address = row[7]
                break
return parent, age, occupation, contact, email, address


def ViewAdoption(request):
    if request.method == 'GET':
        output = '<table border=1 align=center>'
        output+='<tr><th>Parent Name</th><th>Parent
age</th><th>Occupation</th><th>Contact No</th><th>Email
ID</th><th>Address</th><th>Child Name</td></tr>'
        color = '<font size="" color="black">'
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = '',
database = 'MissingChildDB',charset='utf8')
        with con:
            cur = con.cursor()
```

```
            cur.execute("select * FROM adoption")
            rows = cur.fetchall()
            for row in rows:
                user = row[0]
                child = row[1]
                parent, age, occupation, contact, email, address = getDetails(user)
            output+='<tr><td>'+color+parent+'</td>'
            output+='<td>'+color+age+'</td>'
            output+='<td>'+color+occupation+'</td>'
            output+='<td>'+color+contact+'</td>'
            output+='<td>'+color+email+'</td>'
            output+='<td>'+color+address+'</td>'
            output+='<td>'+color+child+'</td></tr>'
            output+='</table><br/><br/><br/><br/><br/>'
            context= {'data':output}
return render(request, 'ViewAdoption.html', context)


def AdoptAction(request):
   if request.method == 'GET':
      name = request.GET.get('name', False)
      user = ''
      with open("session.txt", "r") as file:
         for line in file:
            user = line.strip('\n')
      file.close
      now = datetime.datetime.now()
      current_time = now.strftime("%Y-%m-%d %H:%M:%S")

      db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
password = '', database = 'MissingChildDB',charset='utf8')
      db_cursor = db_connection.cursor()
      query = "INSERT INTO adoption(username,childname,adoption_date)
VALUES('"+user+"','"+name+"','"+str(current_time)+"')"
      db_cursor.execute(query)
      db_connection.commit()
      print(db_cursor.rowcount, "Record Inserted")
      output = ''
      parent, age, occupation, contact, email, address = getDetails(user)
      output = '<table border=1 align=center>'
      output+='<tr><th>Parent Name</th><th>Parent
age</th><th>Occupation</th><th>Contact No</th><th>Email
ID</th><th>Address</th><th>Child Name</td></tr>'
      color = '<font size="" color="black">'
      output+='<tr><td>'+color+parent+'</td>'
      output+='<td>'+color+age+'</td>'
      output+='<td>'+color+occupation+'</td>'
      output+='<td>'+color+contact+'</td>'
      output+='<td>'+color+email+'</td>'
      output+='<td>'+color+address+'</td>'
```

```
        output+='<td>'+color+name+'</td></tr></table><br/><br/><br/><br/>'
        context= {'data':output}
        return render(request, 'Certificate.html', context)


def ChildDetailsAction(request):
    if request.method == 'POST':
        age = request.POST.get('t1', False)
        colour = request.POST.get('t2', False)
        user = ''
        index = 0
        with open("session.txt", "r") as file:
            for line in file:
                user = line.strip('\n')
            file.close
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = '',
database = 'MissingChildDB',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select child_age,child_color FROM parentsignup where
username='"+user+"'")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == age and row[1] == colour:
                    index = 1
        if index == 1:
            imgs = ['002A03.JPG','049A10.JPG','053A03.JPG','053A04.JPG','053A06.JPG']
            output = '<table border=1 align=center>'
            output+='<tr><th>Username</th><th>Child Image</th><th>Adopt
Child</th></tr>'
            color = '<font size="" color="black">'
            for i in range(len(imgs)):
                if checkImage(imgs[i]) == 0:
                    output+='<tr><td>'+color+user+'</td><td><img
src=/static/testImages/'+imgs[i]+' width=200 height=200></img></td>'
                    output+='<td><a href=\'AdoptAction?name='+imgs[i]+'\'><font size=3
color=black>Click Here</font></a></td></tr>'
            output+='</table><br/><br/><br/><br/>'
            context= {'data':output}
            return render(request, 'ViewImages.html', context)
        else:
            context= {'data':"child details mismatch"}
            return render(request, 'ParentScreen.html', context)
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 = '',
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><img src=/static/photo/'+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_classifier
    global 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('D:/MINI
PROJECT/MissingChilds/MissingChildApp/static/photo/'+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 = ''
```

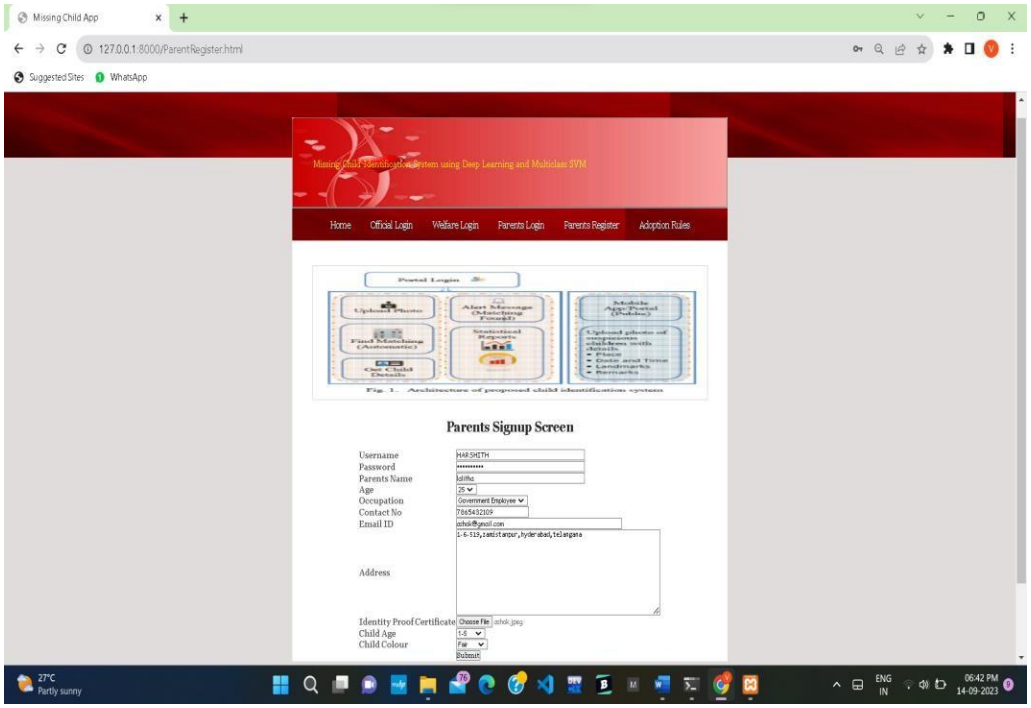
```
        status = 'Child not found in missing database'
        if len(faces) > 0:
            for (x, y, w, h) in faces:
                img = frame[y:y + h, x:x + w]
                option = 1
        if option == 1:
            with open('model/model.json', "r") as json_file:
                loaded_model_json = 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 = '', 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+"','"+filena
me+"','"+str(current_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)

def ParentRegisterAction(request):
    if request.method == 'POST' and request.FILES['t9']:
        output = ''
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        parent = request.POST.get('t3', False)
        age = request.POST.get('t4', False)
        occupation = request.POST.get('t5', False)
        contact = request.POST.get('t6', False)
        email = request.POST.get('t7', False)
        address = request.POST.get('t8', False)
        child_age = request.POST.get('t10', False)
        child_color = request.POST.get('t11', False)
        myfile = request.FILES['t9']
```

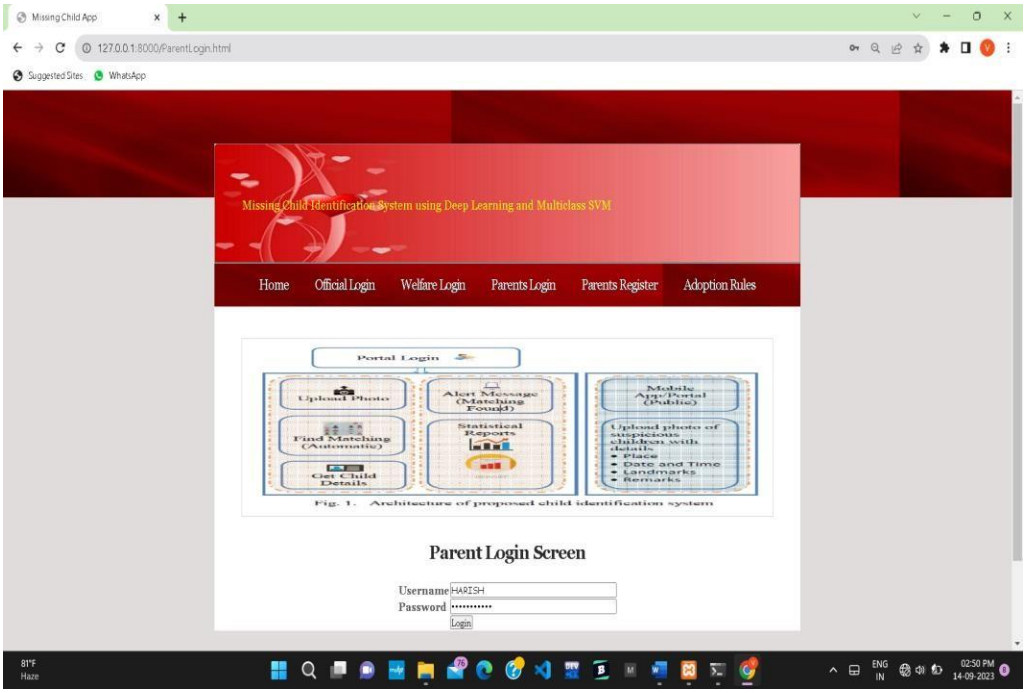
```
        filenames = request.FILES['t9'].name
        fs = FileSystemStorage()
PROJECT         filename = fs.save('D:/MINI
/MissingChilds/MissingChildApp/static/documents/'+filenames, myfile)

        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
password = '', database = 'MissingChildDB',charset='utf8')
        db_cursor = db_connection.cursor()
        query = "INSERT INTO
parentsignup(username,password,name,age,occupation,contactno,email,address,filename,
child_age,child_color)
VALUES('"+username+"','"+password+"','"+parent+"','"+age+"','"+occupation+"','"+cont
act+"','"+email+"','"+address+"','"+filenames+"','"+child_age+"','"+child_color+"')"
        db_cursor.execute(query)
        db_connection.commit()
        print(db_cursor.rowcount, "Record Inserted")
        context= {'data':'Signup process completed'}
        return render(request, 'ParentRegister.html', context)
```

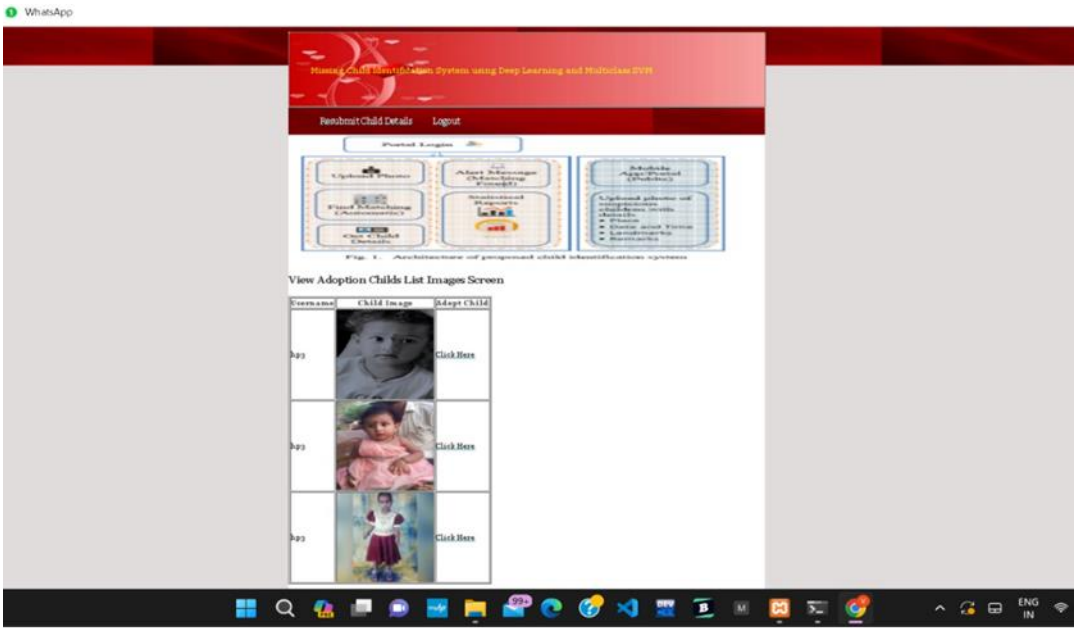
5.SCREENSHOTS



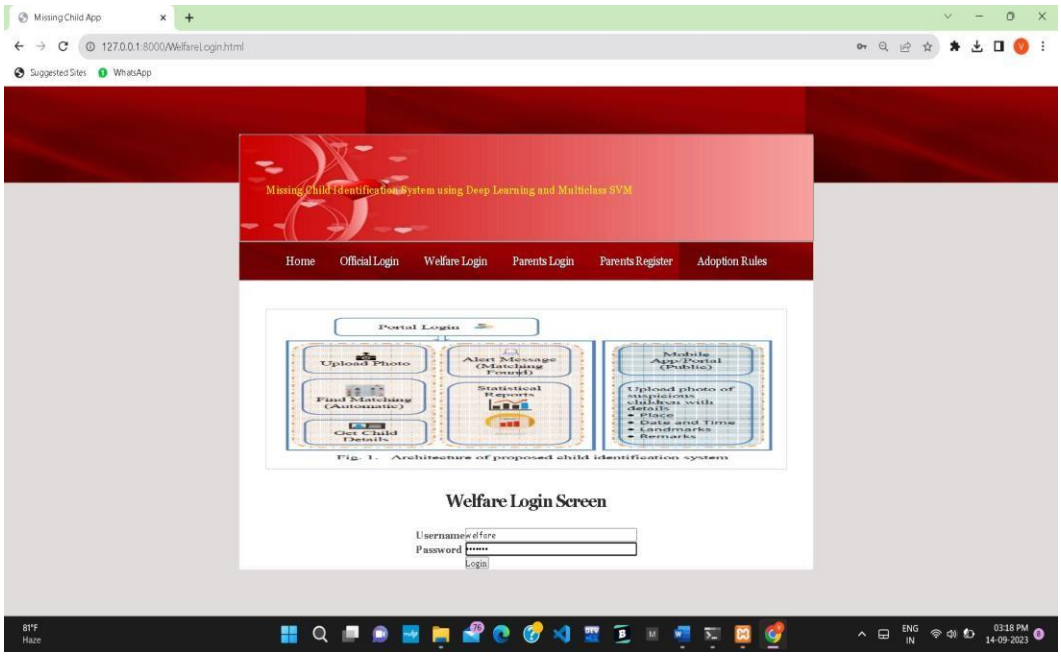
Screenshot 5.1: Parents Register



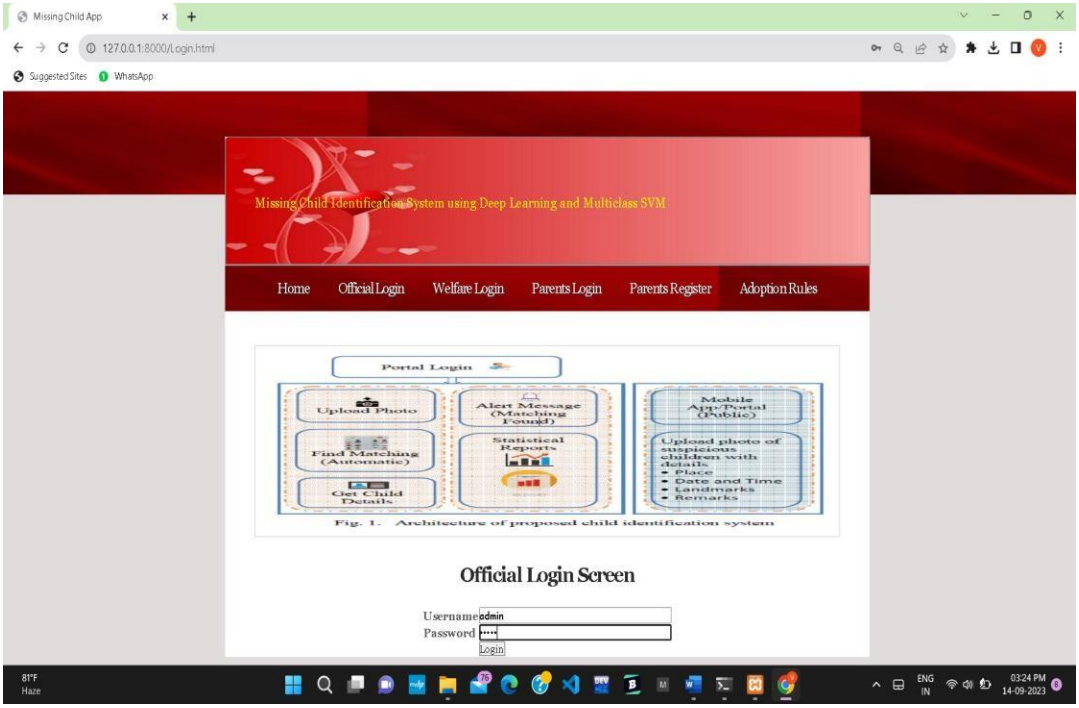
Screenshot 5.2: Parent Login



Screenshot 5.3: Matched Children



Screenshot 5.4: Welfare login



Screenshot 5.5: Official Login

6. TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate thatalthough the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.

6.3 TEST CASES

6.3.1 CLASSIFICATION

Test case ID	Test case name	Purpose	Input	Output
1.	Parent Register	If User registration successfully.	Pass	If already user email exist then it fails.
2.	Parent Login	If Username and password is correct then it will getting valid page.	Pass	Un Register Users will not logged in.
3.	Welfare Login	If User registration successfully.	Pass	If already user email exist then it fails.
4.	User View User	Show our dataset	Pass	If Data set Not Available fail.
5.	Admin login	Admin can login with his login credential. If success he get his home page	Pass	Invalid login details will not allowed here
6.	Admin can activate the register users	Admin can activate the register user id	Pass	If user id not found then it won't login
7.	Results	For our CNN models the accuracy	Pass	If Accuracy Not Displayed fail

7. CONCLUSION
&
FUTURE SCOPE

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT 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 softmax 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

7.2 FUTURE SCOPE

In the future, planning to extend this system further by connecting our systems to public cameras and detect faces real-time. The advancement to this project is collaboration with law enforcement agencies, NGOs and to combine the predictions from the deep learning model and SVM to improve accuracy and robustness.

8. BIBLIOGRAPHY

8. BIBLIOGRAPHY

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 facerecognition system", International Journal of Advanced Engineering and Innovative Technology (IJAEIT), Volume 3 Issue 1 July - August 2016.

[5] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," in British Machine Vision Conference, vol. 1, no. 3, pp. 1-12, 2015.

[6] A. Vedaldi, and K. Lenc, "MatConvNet: Convolutional Neural Networks for MATLAB", ACM InternationalConference on Multimedia, Brisbane, October 2015

GITHUB LINK

https://github.com/burrevedika/missingchild/tree/master