**RECOGNIZING FACIAL EXPRESSION AND GIVING EMOJI USING MACHINE LEARNING**

***A Mini Project-I Report submitted***

***in partial fulfillment of the requirements***

***for the award of the degree of***

**BACHELOR OF TECHNOLOGY**

***In***

**COMPUTER SCIENCE & ENGINEERING**

***By***

M.Vara Vahini-19B01A0596

***Under the esteemed guidance of* Mr.T.Kesava(ASST Professor)**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)**

**(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)**

**BHIMAVARAM – 534 202**

**2020 – 2021**

**SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN (A)**

**(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada) BHIMAVARAM – 534 202**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



**CERTIFICATE**

*This is to certify that the Mini Project-1 entitled* ***“*RECOGNIZING FACIAL EXPRESSION AND GIVING EMOJI USING MACHINE LEARNING*”****, is being submitted by* **M.Vara Vahini**

*bearing* *the* ***Regd. No. 19B01A0596*** *in partial fulfillment of the requirements for the award of the degree of “****Bachelor of Technology*** *in* ***Computer Science & Engineering****” is a record of bonafide work carried out by her under my guidance and supervision during the academic year* **2020 – 2021** *and it has been found worthy of acceptance according to the requirements of the university.*

**Internal Guide Head of the department**

**ACKNOWLEDGEMENTS**

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance has been a source of inspiration throughout the course of this project. We take this opportunity to express our gratitude to all those who have helped us in this project.

We wish to place our deep sense of gratitude to **Sri. K. V. Vishnu Raju, Chairman of SVES**, for his constant support on each and every progressive work of us.

Our deep sense of gratitude and sincere thanks to **Dr. G. Srinivasa Rao, Principal of SVECW** for being a source of an inspirational constant encouragement.

Our deep sense of gratitude and sincere thanks to **Dr. P. Srinivasa Raju, Vice Principal of SVECW** for being a source of an inspirational constant encouragement.

Our deeply indebted and sincere thanks to **Dr. P. Kiran Sree, Head of the Department**, for his valuable advice in completing this project successfully.

Our deep sense of gratitude and sincere thanks to **Mr. T.Kesava, Assistant Professor** for his unflinching devotion and valuable suggestion throughout our project.

**Project Associates**

K.Sai Sravya(19B01A0580)

M.Venkata Asritha(19B01A0585)

K.Jahnavi(19B01A0569)

M.Vara Vahini(19B01A0596)

K.Mounika(20B05A0508)

**INTRODUCTION**

**INTRODUCTION**

Facial expression is the most effective form of non-verbal communication and it provides intimation about emotional state,mindset and intention. Facial expressions not only can change the flow of conversation but also provide the listeners a way to communicate a wealth of information to the speaker without even uttering a single word.

From the perspective of automatic recognition, a facial expression can be considered to consist of deformations of facial components and their spatial relations, or changes in the pigmentation of the face. The requirement for proficient communication channels between machines and humans becomes progressively imperative in light of the fact that machines and individuals start to share a variety of tasks. Systems to form these communication channels are known as human machine interaction(HMI) systems.

The most frequently used facial expressions classification is done both in terms of Action Units, proposed in Facial Action coding System and in terms of Universal emotions: happiness, sadness, anger, surprise, disgust and fear. There are several different machine learning techniques for classification tasks for instance K-Nearest Neighbors, Artificial Neural Networks, Support Vector Machines, Hidden Markov Models, Expert Systems with rule based classifiers, Bayesian Networks or Boosting Techniques.

Three main issues in classification tasks are: choosing a good feature set , competent machine learning technique and a different database for training. The Feature set should be composed of features that are discriminative and characteristic for specific expression.

Machine learning technique is selected usually by the type of a feature set. The database used as a training set should be big enough and include a variety of data. In facial expression recognition Region of Interests represents the eye pair, nostrils, and the mouth area. Region of Interest is related to defining a large region which contains the point that we want to detect.

In Facial Expression Recognition Systems, only particular regions of the face are used for discrimination. The areas of the eyes, eye-brows, mouth and nose are the main features in any Facial Expression Recognition System. Some facial recognition algorithms perceive facial components by extracting landmarks or elements from the picture of the subject's face. For instance, an algorithm may assess the relative position, size, and state of the eyes, nose, cheekbones and jaw. These features are then used to search for other images with identical features. The majority of the facial expression recognition methods reported yet are focused on recognition of six primary expression categories such as: happiness, sadness, fear, anger, disgust, and grief.

**SYSTEM ANALYSIS**

**SYSTEM ANALYSIS**

**2.1 Existing System**

We have some apps which convert our facial expression to customized emoji.They are mobile apps which produce customized emojis.

We can create our own emojis using those apps but there are no specified apps which detect the real time facial expression and give the emoji as an output.

**2.2 Proposed System**

The main aim of the project is to build a real time web application which detects facial expressions.

Now-a-days there is a common trend for human computer interaction in the field of machine intelligence.

Real time detection of faces and interpreting different facial expressions like happy, anger, sad, fear, and surprise etc. is based on facial features and their actions.

The key elements of face are considered for detection of face and prediction of expression or emotions of face.

To determine the different facial expressions, the variation in each facial feature is used.We detect the face using a webcam and give the most reliable emoji.

**2.3 Feasibility Study**

Before starting the project, a feasibility study is carried out to measure the viability of the system. Feasibility study is necessary to determine if creating a new or improved system is friendly with the cost, benefits, operation, technology and time. The different feasibilities that have to be analyzed are:

* Technical Feasibility
* Operational Feasibility
* Economic Feasibility
* Schedule Feasibility

**Technical Feasibility:**

Technical feasibility is one of the first studies that must be conducted after the project has been identified. Technical feasibility study includes the hardware and software devices.

**Operational Feasibility:**

Operational Feasibility is a measure of how well a proposed system solves the problem and takes advantage of the opportunities identified during scope definition. The following points were considered for the project's technical feasibility.

* The system will detect and capture the image of the face.
* The captured image is then identified in which category.

**Economic Feasibility:**

The purpose of economic feasibility is to determine the positive economic benefits that include quantification and identification. The system is economically feasible due to availability of all requirements such as collection of data form.

**Schedule Feasibility:**

Schedule feasibility is a measure of how reasonable the project timetable is. The system is found to be feasible because the system is designed in such a way that it will finish at the prescribed time.

**SYSTEM REQUIREMENTS SPECIFICATION**

**System Requirements Specification**

**3.1 Software Requirements**

Following are the software requirements necessary of the project:

* Anaconda (Jupyter Notebook)
* Language : Python

**3.2 Hardware Requirements**

Following are the hardware requirements necessary of the project:

* Processor : I3 or above
* RAM : 8GB
* Space on Hard Disk : 500GB
* Webcam is required

**3.3 Functional Requirements**

The functional requirements for a system describe what the system should do. Those requirements depend on the type of software being developed, the expected users of the software. These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.

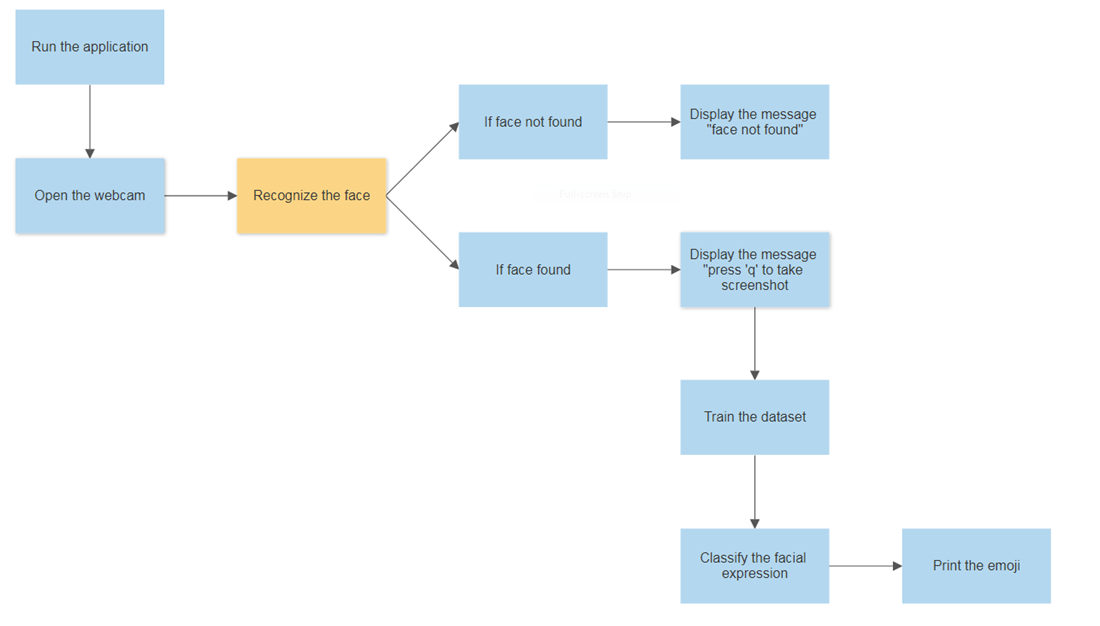
* User will take the screenshot with the help of a webcam.
* System should be able to analyze facial expressions of a person.
* The most reliable emoji should be given as output.

**SYSTEM DESIGN**

**System Design**

**4.1 Introduction**

Design is the first step in the development phase of an engineering product or system. Design is the place where quality is considered in software development. Design is the only way that we can accurately translate user requirements into finished software engineers and software maintenance that steps follow. Without design we risk building an unstable design, one that will fail when small changes are made, one that may be difficult to test and one whose quantity cannot be assessed until late in the software engineering process.System design shows the overall design of the system. In this section we discuss the design aspects of the system.



**4.2 Data flow diagrams (UML Diagrams)**

**Introduction to UML**

A model is an abstract representation of a system, constructed to understand the system priority to building or modifying it. A model is a simplified representation of reality and it provides a means for conceptualization and communication of ideas in a precise and ambiguous form. We build models so that we can better understand the system we are developing. The elements are like components which can be associated in different ways to make a complete UML picture, which is known as a diagram. Thus, it is very important to understand the different diagrams to implement the knowledge in real life systems.

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. It is a method for describing the system architecture in detail using the blueprint. We use UML diagrams to portray the behavior and structure of a system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in products being developed need to be designed.

These are various kinds of methods in software design:

* Use case Diagram
* Class Diagram
* Sequence Diagram
* Activity Diagram
* State chart Diagram

**4.2.1 Use case Diagram**

Use case diagrams are used to depict the functionality of a system or a part of a system. They are widely used to illustrate the functional requirements of the system and its interaction with external agents(actors).

A use case is basically a diagram representing different scenarios where the system can be used. A use case diagram gives us a high level view of what the system or a part of the system does without going into implementation details. When the initial task is complete, use case diagrams are modeled to present the outside view.

In brief, the purposes of the use case diagrams can be said to be as follows

* Used to gather the requirements of a system.
* Used to get an outside view of a system.
* Identify the external and internal factors influencing the system.

Use case diagrams commonly contains

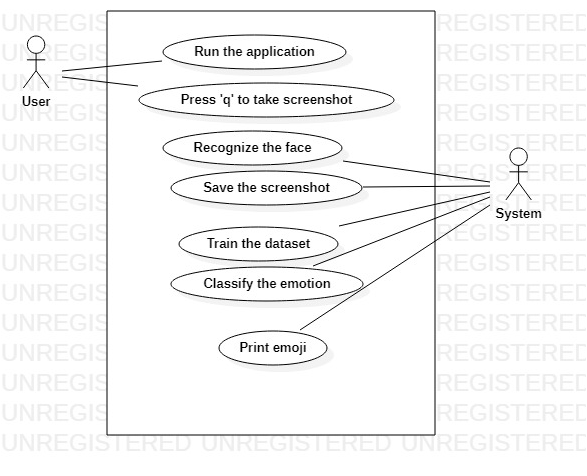
* Use cases
* Actors
* Dependency, generalization and association relationships.

**Use cases**

A use case is a software and system engineering term that describes how a user uses a system to accomplish a particular goal.

**Actors**

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

****

**SYSTEM IMPLEMENTATION**

**System Implementation**

**5.1 Introduction**

The purpose of system implementation can be summarized as follows: making the new system available to the prepared set of users (the deployment), and positioning ongoing support and maintenance of the system within the performing organization (the transaction). At a finer level necessary to educate the consumer on the use of the system, placing the newly developed system into production, confirming that business functions that interact with the system and functioning properly.

Transitioning the system support responsibilities involve changing from a system development to the system and maintenance mode of operation, with ownership of the new system moving from the project team to the performing organization.

A key difference between system implementation and all the other phases of the lifecycle is that all project activities up to this point have been performed in safe, protected and checked environments. It is through the careful planning, execution and management of system implementation activities that the project team can minimize the likelihood of these occurrences and determine appropriate contingency plans in the event of the problem.

**5.2 Project Modules**

1.Opencv

2.Matplotlib

3.Numpy

4.Pyautogui

5.Deepface

6.Emoji

**OPENCV:**

* OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2. imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.
* We used CascadeClassifier and haarcascade\_frontalface\_default.xml file to recognize the face.

**Example:**

import cv2

face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml');

**Matplotlib:**

* Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK
* We used matplotlib.pyplot as plt in order to plot the image on a graph scale and to train and test the emotion of the image.

**Example:**

import matplotlib.pyplot as py

plt.imshow(img1[:,:,::-1])

plt.show()

**NUMPY:**

* NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.. NumPy is open source software.

**Example:**

import numpy as np

image = cv2.cvtColor(np.array(image),cv2.COLOR\_RGB2BGR)

**PYAUTOGUI:**

* PyAutoGUI is a Python automation library used to click, drag, scroll, move, etc. It can be used to click at an exact position.Screenshot Functions. (PyAutoGUI uses Pillow for image-related features.)
* We used pyautogui to take the screenshot of the image to print the emoji

**Example:**

import pyautogui

image = pyautogui.screenshot()

**DEEPFACE:**

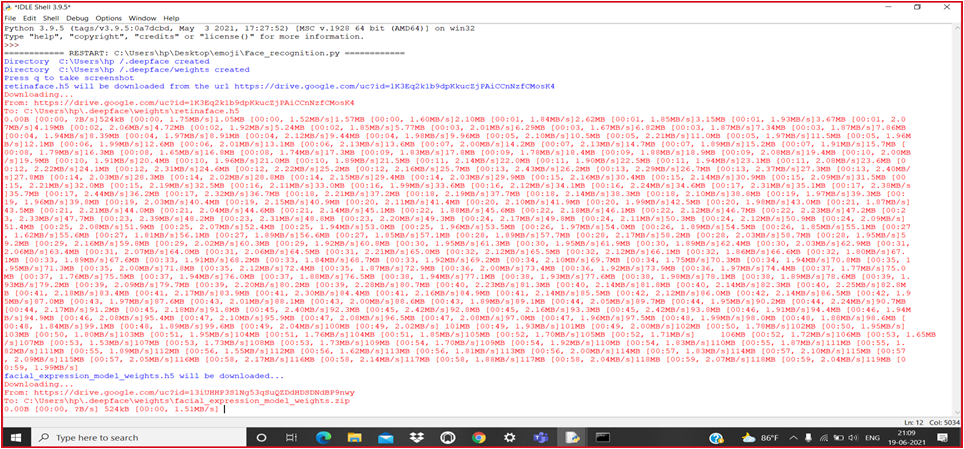
* DeepFace is a deep learning facial recognition system created by a research group at Facebook. It identifies human faces in digital images. The program employs a nine-layer neural network with over 120 million connection weights and was trained on four million images uploaded by Facebook users.
* We used deepface to train and test the image and classify the emotion of the image.

**Example:**

import deepface as DeepFace

result=DeepFace.analyze(img1,actions=['emotion'])

**TESTING OF DATASET:**

****

**EMOJI:**

* Emoji module that is available in python is used to print the emojis as terminal output.
* pip install emoji should be run on the terminal to install this module.
* Emojize() function requires the CLDR short name to be passed in it as the parameters. It then returns the corresponding emoji. Replace the spaces with underscore in the CLDR short name.

**Example:**

import emoji

print(emoji.emojize(“:grinning\_face\_with\_big\_eyes:”))

**TESTING**

**Testing**

**6.1 Introduction**

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 TESTING METHODOLOGIES**

The following are the Testing Methodologies:

* Unit Testing.
* Integration Testing.
* User Acceptance Testing.
* Output Testing.
* Validation Testing.

**Unit Testing**

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module’s control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing paths are tested for the expected results. All error handling paths are also tested.

**Integration Testing**

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and build a program structure that has been dictated by design.

The following are the types of Integration Testing:

**1.Top Down Integration**

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from the main module and individual stubs are replaced when the test proceeds downwards.

**2. Bottom-up Integration**

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the

The need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

▪ The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.

▪ A driver (i.e.) the control program for testing is written to coordinate test case input and output.

▪ The cluster is tested.

▪ Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches test each module individually and then each module is integrated with a main module and tested for functionality.

**OTHER TESTING METHODOLOGIES**

**User Acceptance Testing**

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

**Output Testing**

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

**Validation Checking**

Validation checks are performed on the following fields.

**Text Field:**

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entries always flash and error messages.

**Numeric Field:**

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error message.

The individual modules are checked for accuracy and what it has to perform. Each module is subjected to a test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information used in the program; the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces an output revealing the errors in the system.

**Preparation of Test Data**

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

**Using Live Test Data:**

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations of formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

**Using Artificial Test Data:**

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program. The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the system's specifications. The package “Virtual Private Network” has satisfied all the requirements specified as per software requirement specification and was accepted.

**USER TRAINING**

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

**MAINTENANCE**

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user’s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

**6.3 TESTING STRATEGY :**

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must cooperate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

**SYSTEM TESTING:**

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

**UNIT TESTING:**

In unit testing different modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

**CONCLUSION**

**CONCLUSION**

This project proposes an approach for recognizing the category of facial expressions. Face Detection and Extraction of expressions from facial images is useful in many applications, such as robotics vision, video surveillance, digital cameras, security and human-computer interaction. This project's objective was to develop a facial expression recognition system implementing the computer visions and enhancing the advanced feature extraction and classification in face expression recognition.This project finally detects the facial expressions of the face and gives the most reliable emoji as output.The emotions detected are happy,sad,angry,surprise,fear,disgust,neutral.We used the emoji module,so we get 6 different emojis and it will print a message neutral if the given expression is neutral.

**BIBLIOGRAPHY**

1. <https://www.youtube.com/watch?v=ERXqo_ZEnIo>
2. <https://www.geeksforgeeks.org/how-to-take-screenshots-using-python/>
3. <https://www.geeksforgeeks.org/python-program-to-print-emojis/>
4. <https://www.codingforentrepreneurs.com/blog/open-cv-python-change-video-resolution-or-scale>
5. <https://github.com/ajinkyabedekar/Face-to-Emoji>

**APPENDIX**

**APPENDIX**

**9.1 Introduction to python**

Python is an open source, high-level programming language developed by Guido van Rossum in the late 1980s and presently administered by Python Software Foundation. It came from the ABC language that he helped create early on in his career. Python is a powerful language that you can use to create games, write GUIs, and develop web applications.

It is a high-level language. Reading and writing codes in Python is much like reading and writing regular English statements. Because they are not written in machine-readable language, Python programs need to be processed before machines can run them. Python is an interpreted language. This means that every time a program is run, its interpreter runs through the code and translates it into machine readable byte code.

Python is an object-oriented language that allows users to manage and control data structures or objects to create and run programs. Everything in Python is latest version of Python, in fact, first class. All objects, data types, functions, methods, and classes take equal position in Python. Programming languages are created to satisfy the needs of programmers and users for an effective tool to develop applications that impact lives, lifestyles, economy, and society. They help make lives better by increasing productivity, enhancing communication, and improving efficiency. Languages die and become obsolete when they fail to live up to expectations and are replaced and superseded by languages that are more powerful.

Python is a programming language that has stood the test of time and has remained relevant across industries and businesses and among programmers, and individual users. It is a living, thriving, and highly useful language that is highly recommended as a first programming language for those who want to dive

In to and experience programming. Advantages of Using Python Here are reasons why you would prefer to learn and use Python over other high-level languages.

**Readability**

Python programs use clear, simple, and concise instructions that are easy to read even by those who have no substantial programming background. Programs written in Python are, therefore, easier to maintain, debug, or enhance.

**Higher productivity**

Codes used in Python are considerably shorter, simpler, and less verbose than other high level programming languages such as Java and C++. In addition, it has well-designed built-in features and standard library as well as access to third party modules and source libraries. These features make programming in Python more efficient.

**Less learning time**

Python is relatively easy to learn. Many find Python a good first language for learning programming because it uses simple syntax and shorter codes. Python works on Windows, Linux/UNIX, Mac OS X, other operating systems and small form devices. It also runs on microcontrollers used in appliances, toys, remote controls, embedded devices, and other similar devices.

**Installing Python in Windows**

To install Python, you must first download the installation package of your preferred version from this link: https://www.python.org/downloads/ On this page, you will be asked to choose between the two latest versions for Python 2 and 3: Python 3.5.1 and Python 2.7.11. Alternatively, if you are looking for a specific release, you can scroll down the page to find download links for earlier versions. You would normally opt to download the latest version, which is Python 3.5.1. This was released on December 7, 2015. However, you may opt for the latest version of Python 2, 2.7.11. Your preferences will usually depend on which version will be most usable for your project. While Python 3 is the present and future of the language, issues such as third-party utility or compatibility may require you to download Python 2.

**Anaconda:**

Anaconda is a free open source distribution of the python and R programming languages for large-scale data processing, predictive analysis, and scientific computing that aims to simplify package management and deployment. Package versions are managed by the package management system. Where packages, notebooks, projects and environments are shared. Powerful collaboration and package management for open source and private projects.

**Jupyter Notebook:**

The Jupyter Notebook is an open source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. We can perform data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning and much more. It is available in anaconda software and can be used easily.

**Getting Started:**

There are two ways that python is commonly used. The first is an interactive command environment, each as Python or IDLE, which are commonly bundled with the Python interpreter. Starting Python with one of these (using Start/Python in Windows, or by typing python at a command prompt (which will be shown as >>>). Unlike with C or Java, you can type commands at this prompt and the interpreter will run the commands and display the results, if any, on the screen. You can write functions in a text editor and run them from the command prompt by calling them by name.

**Matplotlib:**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension numpy.

**%matplotlib inline:** This turns on inline plotting, where plot graphics will appear in your notebook. This has important implications for interactivity. For inline plotting, commands cells below the cell that outputs a plot will not affect the plot.

For example, changing the map color map is not possible from cells below the cells below the cell that creates a plot.

**Pyplot** is a module in the matplotlib package. That’s why you often see matplotlib.pyplot in code. The module provides an interface that allows you to implicitly and automatically create figures axes to achieve the desired plot. This is especially handy when you want to quickly plot something without instantiating any figures or axes as you saw in the example in the first section of this tutorial. You see, you haven’t explicitly specified these components, yet you manage to output a plot that you have even customized! The defaults are initialized and any customizations that you do, will be done with the current Figures and Axes.

Lastly, Pylab is another module, but it gets installed alongside the matplotlib package. It bulks imports pyplot and the numpy library and was generally recommended when you were working with arrays, doing mathematics interactively and wanted across to plotting feature