##### PICZAP

**A PROJECT REPORT**

###### ***Submitted by***

1. Parmeet Singh Banwait (20BAI10141)
2. MVN Rajesh Reddy (20BAI10249)
3. Sparsh Handa (20BAI10295)
4. Mohd Mohsin Khan (20BAI10340)

*in partial fulfilment for the award of the degree*

*of*

##### BACHELOR OF TECHNOLOGY

*in*

**PROGRAM OF STUDY**

****

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**VIT BHOPAL UNIVERSITY**

**KOTHRIKALAN, SEHORE**

**MADHYA PRADESH - 466114**

December, 2021

**VIT BHOPAL UNIVERSITY, KOTHRIKALAN, SEHORE**

**MADHYA PRADESH – 466114**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **PICZAP** is the bonafide work of Parmeet Singh Banwait(20BAI10141), MVN Rajesh Reddy (20BAI10249), Sparsh Handa (20BAI10295) and Mohd Mohsin Khan (20BAI10340) who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported at this time does not form part of any other project/research work based on which a degree or award was conferred on an earlier occasion on this or any other candidate.

**PROGRAM CHAIR PROJECT GUIDE**

Dr S Southarajan Mr Kailash

School of Computer Science and Engineering School of Computer Science and Engineering

VIT BHOPAL UNIVERSITY VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

First and foremost I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr S Southarajan, Head of the Department, School of Aeronautical Science for much of his valuable support encouragement in carrying out this work.

I would like to thank my internal guide Mr Kailash, for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computer Science and Engineering, who extended directly or indirectly all support.

Last, but not least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

**LIST OF ABBREVIATIONS**

* App - application/ applications
* OS - operating system
* Img - image
* Sys - system

**LIST OF FIGURES AND GRAPHS**

| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |
| 1 | Watermark Extract | 13 |
| 2 | Classification of watermark | 14 |
| 3 | Watermark Attacks | 14 |
| 4 | Comparison architecture diagram | 22 |
| 5 | User architecture diagram | 23 |
| 6,7 | Output 1 | 31,32 |
| 8,9,10,11 | Output 2 | 32,33,34 |

**LIST OF TABLES**

| **TABLE NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |

(no tables are included in the report)

**ABSTRACT**

The abstract of our application is about watermark. Our application name is PICZAP, which actually is divided into two parts, the first which compares the dataset and another which actually is the user interactive and perform the task given by the user.

**[PURPOSE-METHODOLOGY-FINDINGS]**

**TABLE OF CONTENTS**

| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |
|  | List of Abbreviations  List of Figures and Graphs  List of Tables  Abstract | **3**  4  5  6 |
| 1 | **CHAPTER-1:**  **PROJECT DESCRIPTION AND OUTLINE** Introduction 1.2 Motivation for the work  1.5 Problem Statement  1.6 Objective of the work  1.7 Organization of the project  1.8 Summary | 11  .  .  . |
| 2 | **CHAPTER-2:**  **RELATED WORK INVESTIGATION**  2.1 Introduction  2.2 Core area of the project  2.3 Existing Approaches/Methods  2.3.1 Approaches/Methods -1  2.3.2 Approaches/Methods -2  2.3.3 Approaches/Methods -3  2.4 <Pros and cons of the stated Approaches/Methods >  2.5 Issues/observations from investigation  2.6 Summary | 13 |
| 3 | **CHAPTER-3:**  **REQUIREMENT ARTIFACTS**  3.1 Introduction  3.2 Hardware and Software requirements  3.3 Specific Project requirements  3.3.1 Data requirement  3.3.2 Functions requirement  3.3.3 Performance and security requirement  3.3.4 Look and Feel Requirements  3.4 Summary | 18 |
| 4 | **CHAPTER-4:**  **DESIGN METHODOLOGY AND ITS NOVELTY**  4.1 Methodology and goal  4.2 Functional modules design and analysis  4.3 Software Architectural designs  4.4 Subsystem services  4.5 User Interface designs  4.6 Novelty  4.7 Summary | 22 |
| 5 | **CHAPTER-5:**  **TECHNICAL IMPLEMENTATION & ANALYSIS**  5.1Outline  5.2 Technical coding and code solutions  5.3 Working Layout of Forms  5.4 Prototype submission  5.5 Test and validation  5.6 Performance Analysis(Graphs/Charts)  5.7 Summary | 26 |
| 6 | **CHAPTER-6:**  **PROJECT OUTCOME AND APPLICABILITY**  6.1Outline  6.2 key implementations outlines of the System  6.3 Significant project outcomes  6.4 Project applicability on Real-world applications  6.4 Inference | 37 |
| 7 | **CHAPTER-7:**  **CONCLUSIONS AND RECOMMENDATION**  7.1Outline  7.2 Limitation/Constraints of the System  7.3 Future Enhancements  7.4 Inference | 39 |
|  | Appendix A  References | 40  42 |

**CHAPTER 1   
Project Description and Outline**

* 1. Introduction

Even a woman delivers a baby; she is left with stretch marks. You are giving birth to writers; your watermark is no less than those stretch marks. Now days watermark has many uses such as copyright protection, owner protection, add descriptive captions, etc.

We are working on watermarking of images using python and open CV. WE will also compare the watermarked images using PSNR, SSIM, RMSE, NRMSE, MSE. Other than watermarking, we will also give the options to crop an image, resize an image and compress an image to our software.

* 1. Motivation for the work

The motivation that actually guided us to make this project is something that we needed to discover, something which was unique and which is not fully imposed in the market before. We actually tried to find out various ways upon which we can actually work upon and came to the conclusion of actually working on this project that is the PICZAP.

* 1. Introduction to the project including techniques

Watermarking is the process of superimposing a logo or a piece of a text on top of a document or image. It’s an important process when it comes to both the copyright protection and marketing of digital works.

We will do this with the help of open CV which is a huge open-source library for computer vision, machine learning and image tampering. It helps us to automate the task. It will be very simple software and very easy to use so that even a person from non-coding background can make the best use of our software.

* 1. Problem Statement

The problem statement of our app is that-

*“PICZAP- A modern innovation towards watermarking”*

* Compare the dataset of 2000 images to find the values
* User interactive phase for watermarking
  1. Objective of the work

The objective of our work is that we actually wanted to find how actually a watermark works and what our application of it’s in the real life.

* 1. Organization of the Project

A project organization is a structure that facilitates the coordination and implementation of project activities. The structure defines the relationships among members of the project management and the relationships with the external environment.

So, we have worked collectively to make our project a great success.

* 1. Summary

In this chapter we came to know about the actual objective and organization of our app PICZAP.

**CHAPTER-2  
RELATED WORK INVESTIGATION**

2.1 INTRODUCTION

A watermark is an relating image or pattern in paper that appears as colorful tones of lightness/ darkness when viewed by transmitted light (or when viewed by reflected light, atop a dark background), caused by consistence or viscosity variations in the paper. Watermarks have been used on postage prints, currency, and other government documents to discourage counterfeiting.

The origin of the water part of a watermark can be plant back when a watermark was commodity that only was in paper. At that time the watermark was created by changing the consistence of the paper and thereby creating a shadow/ lightness in the watermarked paper. This was done while the paper was still wet/ watery and thus the mark created by this process is called a watermark.

Watermarks were first introduced in Fabriano, Italy, in 1282.

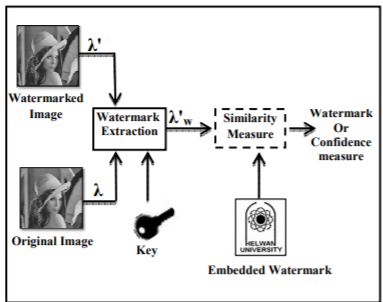


Fig 1

The term "Digital Watermark" was chased by Andrew Tirkel and Charles Osborne in December 1992. The first successful embedding and birth of a stenographic spread diapason watermark was demonstrated in 1993 by Andrew Tirkel, Charles Osborne and Gerard Rankin.

Slowly and gradually, these are the watermark techniques that are being classified. It is shown below in the form of table-

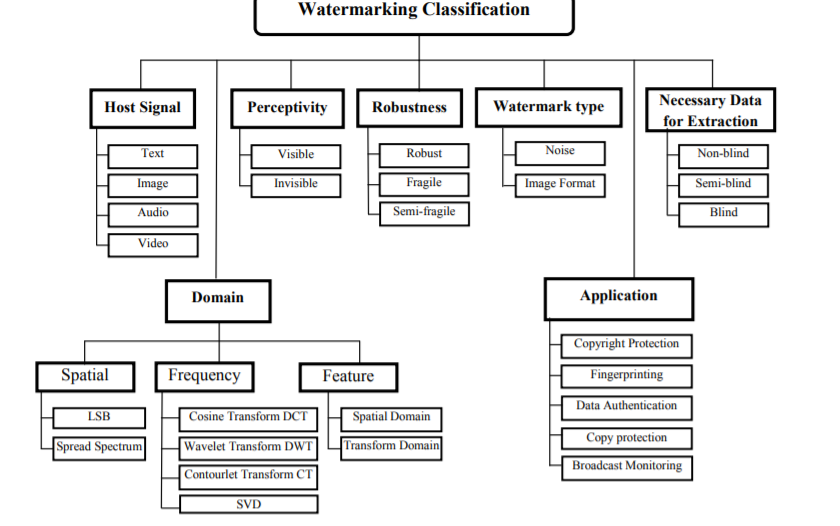


Fig 2

Watermark attacks is classified into four distinct categories namely removal attacks, geometric attacks, cryptographic attacks and Removal attacks. To understand better it is shown below in the form of pictorial graph.

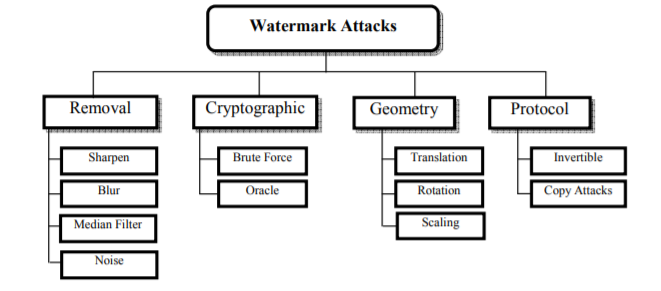


Fig 3

Watermarks are identification marks produced during the paper making process. The first watermarks appeared in Italy during the 13th century, but their use fleetly spread across Europe. They were used as a means to identify the paper maker or the trade council that manufactured the paper. The marks frequently were created by a line darned onto the paper earth. Watermarks continue to be used moment as manufacturer's marks and to help phony.

2.2 CORE AREA OF THE PROJECT

Digital watermarking can be applied to:

* ***Protection and Authentication:***

On one end, watermarking helps cover the brand of your work and ensures that it cannot be reused or altered without your authorization. This means that people can still exercise your work before copping it, without the threat of them stealing it.

* ***Fingerprinting and Digital “Signatures” :***

Vids in movie theaters are also watermarked. The bedded communication is the date of the protuberance and also the ID of the beamer. That way, "Hollywood" knows in which theater a corsair has camcorded a movie and puts the pressure on the proprietor of the theater.

* ***Copy Protection and Device Control:***

Copy protection, also known as happy protection, dupe forestallment and dupe restriction, describes measures to apply brand by precluding the reduplication of software, flicks, music, and other media. It is most generally plant on tapes, DVDs, Blu- shaft discs, HD-DVDs, computer software discs, videotape game discs and charges, audio CDs and some VCDs. Some styles of dupe protection have also led to examines because it caused vexation for paying consumers, or it intimately installed fresh or unwanted software to descry copying conditioning on the consumer's computer. Making dupe protection effective while guarding consumer rights remains a problem with media publication.

* ***Broadcast Monitoring:***

Broadcast monitoring is the process of tracking conditioning on broadcasting channels in compliance with intellectual rights and other broadcasting laws using the computer or mortal system. The being monitoring system is complex, not too effective and not extensively respectable. There are also some concerns from parents on some impact of TV programs being linked to some felonious conditioning in children. The work developed a frame for effective broadcast monitoring for maternal guidance TV using digital watermarking.

2.3 EXISTING APPROCHES/METHODS

There are some of the existing work and methodology that we came across and studying them we made our project. So, these are some of the approaches that we came across-

2.3.1 *Approach 1*

There were some set of codes that have a set of code that only uses the codes which only do two things i.e. the addition of watermark or either the deletion of watermark which was indeed a good approach for any user to use and do the needful by working on it in a simpler way.

2.3.2 *Approach 2*

There were some applications that only enable to add watermark that users give. This was only useful to the users which want to add the watermark to it.

2.3.3 *Approach 3*

There were some applications like they used to detect if the image contains a watermark or not.

2.4 PROS AND CONS OF THE STATED APPROCHES/METHODS

*Pros of the methods-*

* If a user just wants to detect some types of watermark, then user can use it.
* If a user just wants to protect their write-ups from being copied he/she can use them to protect.
* If a user needs to remove a watermark from a existing work then he or she can use them to remove it.
* It will protect user image, doc, etc from being copied by other users.

*Cons of the methods-*

* There was an application, which only detects watermark, what if a user needs to add or remove a watermark then what he/she will do, there is one option left to switch the application.
* There is an application which only adds watermark, but what if a user needs to have to remove or edit the image.
* The very first approach which adds or remove a watermark, but what after performing their particular steps the user needs to edit the same image, he might have to change the application.

2.5 ISSUES/OBSERATIONS FROM INVESTIGATION

After going through all the approaches with their pros and cons we came to a conclusion that each approach has one most common disability i.e. the lack of the feature, the user need to switch between one application to another, which might be a bad impact of the application towards the user or might be the user get frustrated and do not participate in this.

2.6 SUMMARY

In this chapter we came to know that Fabrinco developed the concept of watermarking in 1282. This slowly and gradually leads to the evaluation of the concept of watermarking now-a-days. Then we came to know about what are the core areas where the project can be used i.e. protecting someone’s work, plagiarism and many more. We have also looked upon the different approaches with their pros and cons of the method and what we have observed from these particular approaches.

**Chapter-3  
REQUIREMENTS ARTIFACTS**

3.1 INTRODUCTION

A requirement describes a condition or capability to which a system must conform. A requirement is either derived directly from user needs or stated in a contract, standard, specification, or other formally imposed document.

In the Conditions Operation (RM) operation, you use vestiges and artifact types to define conditions and support and enhance the description of conditions.

For illustration, you can use features and use cases to describe the conditions, and also enhance the description by creating plates, wireframes, or storyboards.

Artifact is a general term for an object in a depository. Vestiges can be of colourful types, which have customizable attributes and data types.

While there's no dereliction artifact type called “demand,” you can produce one or use the dereliction artifact types that are in the sample design templates.

Numerous artifact types are included in the sample design templates, including these types

* Requirements/Conditions
* Use cases
* Design documents
* Business process diagrams
* Use case diagrams

Artifact formats can be specific to individual artifact types or can be used for multiple types. For illustration, you might use the textbook format for a point or a use case specification or a custom artifact type. Still, the illustration format is generally used simply for creating plates.

You can produce and colonize vestiges that are grounded on several formats, including these:-

* Text

Use this format to produce rich- textbook demand content that can contain textbook, images, and bedded vestiges. This format is useful for textbook- grounded vestiges types, similar as actor and use case specifications, stoner stories, features, business pretensions, and glossary terms.

* Collection

Use this format to group a set of affiliated vestiges in a collection.

* Module

Use this format to produce a structured document that consists of vestiges in a module.

* Diagram

Use this format to produce graphical vestiges similar as wireframes, business process plates, and use case plates.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

The very common requirements required for any project are the hardware and software requirements.

Talking about the hardware requirements include-

* Monitor
* Keyboard
* Mouse
* Operating System- Linux- Ubuntu 16.04 to 17.10 or Windows 7 to 10, with 2GB RAM (4GB preferable).
* CPU and RAM.

There are a lot of different parts of any given piece of software that can be artifacts. The software requirements are very basic that a Python version (preferably 3.9, 3.8, and 3.7) must be installed.

3.3 SPECIFIC PROJECT REQUIREMENTS

3.3.1 *Data requirements:*

The data what we require for the projects are a set of 2000 images that need to be watermarked and a set of 100 watermarks that we will watermark on those 2000 images. We also have done a user interaction phase where we asked the user for the image and the watermark to be processed.

3.3.2 *Function requirements:*

The basic functions and libraries used building our project our-

* CV2: OpenCV-Python is a library of Python tapes designed to break computer vision problems. cv2. imread () system loads an image from the specified train. OpenCV is a great tool for image processing and performing computer vision tasks. It's an open- source library that can be used to perform tasks like face discovery, expostulation shadowing, corner discovery, and much further. It supports multiple languages including python, java, C++.
* PIL from IMAGE: PIL is the Python Imaging Library which provides the python practitioner with image editing capabilities. The Image module provides a class with the same name which is used to represent a PIL image. Public interest action is the use of the law to advance mortal rights and equivalency, or raise issues of broad public concern. It helps advance the cause of nonage or underprivileged groups or individualities. Public interest cases may arise from both public and private law matters.
* OS: The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard mileage modules. This module provides a movable way of using operating system-dependent functionality. In veritably simple terms, operating system is software that manages the tackle coffers of the computer. Hence, it needs to run directly on top of the tackle without anything in between, this is called running bare essence.
* Sys: The sys module in Python provides colorful functions and variables that are used to manipulate different corridor of the Python runtime terrain. It allows operating on the practitioner as it provides access to the variables and functions that interact explosively with the practitioner.
* NumPy: NumPy can be used to perform a wide variety of fine operations on arrays. It adds important data structures to Python that guarantee effective computations with arrays and matrices and it supplies an enormous library of high- position fine functions that operate on these arrays and matrices.
* Random: The random, random () system returns a arbitrary pier number between 0.0 to1.0. The function does not need any arguments. The arbitrary number or data generated by Python's random module isn't truly arbitrary; it's pseudo-random (it is PRNG), i.e., deterministic. The random module uses the seed value as a base to induce an arbitrary number.
* Skimage: We can crop images inside your Python tablet as well using skimage. We crop images to remove the unwanted portion of the image or to concentrate on a particular part of the image. scikit- image is a collection of algorithms for image processing. It's available free of charge and free of restriction.
* Matplotlib: Matplotlib is a conniving library for the Python programming language and its numerical mathematics extension NumPy. It provides an object- acquainted API for bedding plots into operations using general- purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.
* imageChops: The ImageChops module contains a number of arithmetical image operations, called channel operations (“ chops”). These can be used for colorful purposes, including special goods, image compositions, algorithmic oil, and more. PIL. add () system adds two images, dividing the result by scale and adding the neutralize.
* Math: What's math module in Python? The math module is a standard module in Python and is always available. To use fine functions under this module, you have to import the module using import math. It gives access to the underpinning C library functions.
* Operator: Oerators are special symbols in Python that carry out computation or logical calculation. The value that the operator operates on is called the operand. For illustration>>> 2 3 5. Then, is the operator that performs addition.
* Argparse: argparse is the “recommended command- line parsing module in the Python standard library.” It's what you use to get command line arguments into your program. argparse — Parser for command- line options, arguments and sub-commands. The argparse module makes it easy to write stoner-friendly command- line interfaces. The program defines what arguments it requires, and argparse will figure out how to parse those out ofsys. argv.
* Imutils: Imutils are a series of convenience functions to make introductory image processing functions similar as restatement, gyration, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python2.7 and Python 3.

3.4 SUMMARY

In this chapter, we learnt about the requirement artifacts, which enable us to know what all hardware and software are required upon the completion of our project. Then we came to know about what all are the python functions, libraries and modules required for the completion; what all dataset we require for its completion.

**CHAPTER:4  
DESIGN METHODOLOGY AND ITS NOVELTY**

* 1. METHEDOLOGY AND GOAL

The methodology used to build this project- basically this project has been divided into two parts that is the-

* + - *Comparison of data:* The very first thing that we did here is simply taken the dataset (2000 images and 100 watermark) and performed the following actions into it-
      * Addition of the watermark
      * Removal of the watermark
      * Cropping of the image
      * Resizing an image
      * Compressing an image

Performing these particular actions on the set of these 2000 images (the original and the after performing the above actions) we calculate some values like-

* + - * PSNR
      * SSIM
      * RMSE
      * NRMSE
      * MSE

And then we have plotted the graph for these to see the actual difference between them.

**Goal:** Its prime goal was to compare the dataset and plot a graph to see what all the changes have been made between the original image and after performing the particular actions.

* + - *User interactive phase:* This phase allows us too actually to interact with the user. Here, the user gives the image that he/she needs to actually perform a particular action, actions like-
      * Addition of the watermark
      * Removal of the watermark
      * Cropping of the image
      * Resizing an image
      * Compressing an image
      * Break

Which will give the output; the particular action chosen by the user.

**Goal:** This gives our application to actually interact with the user and perform the particular actions that a user gives.

* 1. FUNCTIONAL MODULES DESIGN AND ANALYSIS

Functional Design is a paradigm used to simplify the design of tackle and software bias similar as computer software and decreasingly, 3D models. A functional design assures that each modular part of a device has only one responsibility and performs that responsibility with the minimum of side goods on other corridor. Functionally designed modules tend to have low coupling.

Hence, the functional module design and analysis has been explained below using the flow diagram.

* 1. SOFTWARE ARCHITECTURE DIAGRAM

The first goal architecture diagram-

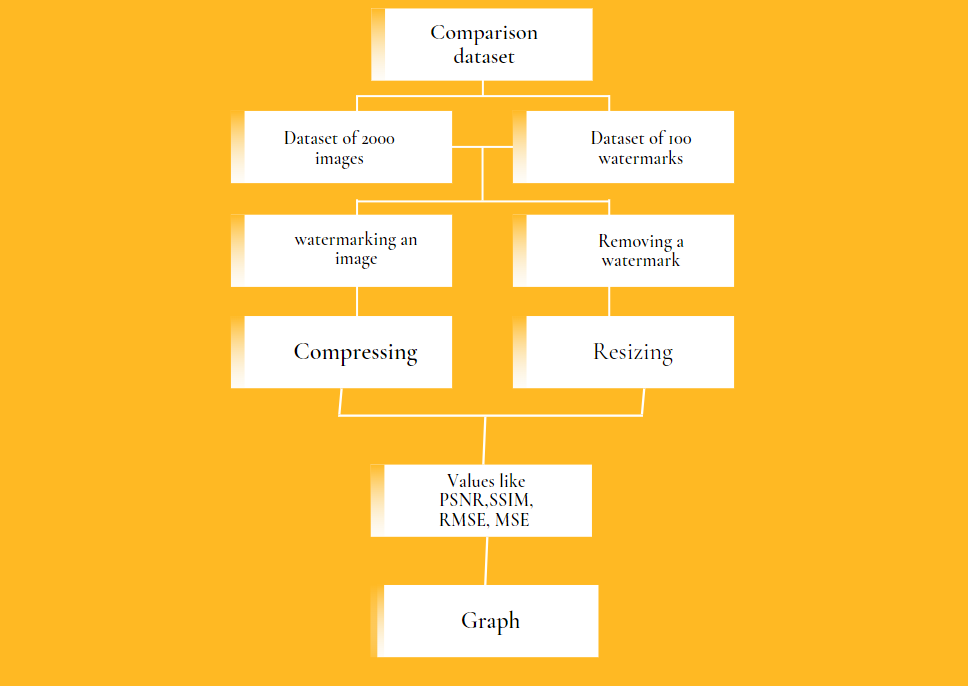


Fig 4

The second architecture diagram-

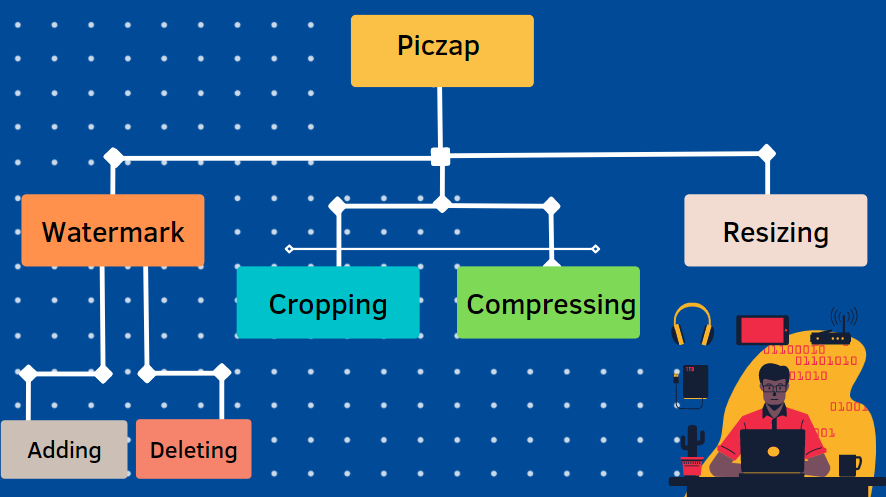


Fig 5

* 1. SUBSYSTEM SERVICES

The services that will be provided by the our application will be-

* + - * Comparison of the data
      * Use interface (PICZAP)
        + Watermark

Adding

Deleting

* + - * + Cropping
        + Compressing
        + Resizing
  1. NOVELTY OF THE PROJECT

The novelty of our project is that-

* + - * There is no such project which contains these combined features.
      * We also have compared data between different values and all and provided a definite comparison between them.
      * It also helps the user to actually compare between there data and all.
  1. SUMMARY

In this chapter we came to know about the proposed methodology and goals proposed by us. The functional module design analysis along with the structural architecture design with subsystem services provided.

**CHAPTER:5**

**TECHNICAL IMPLEMENTATION & ANALYSIS**

5.1 OUTLINE



Fig 6

5.2 TECHNICAL CODING AND CODE SOLUTIONS

1. The code for comparison of the dataset is as follows-

from math import log10, sqrt

import cv2

import numpy as np

def PSNR(original, compressed):

mse = np.mean((original - compressed) \*\* 2)

if(mse == 0): # MSE is zero means no noise is present in the signal .

# Therefore PSNR have no importance.

return 100

max\_pixel = 255.0

psnr = 20 \* log10(max\_pixel / sqrt(mse))

return psnr

x = int(input("Enter no."))

n = 1814

z = 0

for i in range(0, x):

n = n+1

z = str(n)

original = cv2.imread(

"D://Code/Input/2000/Image ("+z+").jpg")

compressed = cv2.imread(

"D://Code/Compress/compressed ("+z+").jpg")

# compressed.show()

value = PSNR(original, compressed)

print(value, file=open("D://Code/Compress/PSNR.txt", "a"))

mse = np.mean((original - compressed) \*\* 2)

rmse = sqrt(mse)

nrmse = rmse/mse

print(nrmse, file=open("D://Code/Compress/NRMSE.txt", "a"))

print(rmse, file=open("D://Code/Compress/RMSE.txt", "a"))

print(mse, file=open("D://Code/Compress/MSE.txt", "a"))

print(n)

from skimage.metrics import structural\_similarity

import argparse

import imutils

import cv2

x = int(input("Enter no."))

n = 1814

z = 0

for i in range(0, x):

n = n+1

z = str(n)

# load the two input images

imageA = cv2.imread("D://Code/Input/2000/Image ("+z+").jpg")

imageB = cv2.imread("D://Code/Compress/compressed ("+z+").jpg")

# convert the images to grayscale

grayA = cv2.cvtColor(imageA, cv2.COLOR\_BGR2GRAY)

grayB = cv2.cvtColor(imageB, cv2.COLOR\_BGR2GRAY)

# compute the Structural Similarity Index (SSIM) between the two

# images, ensuring that the difference image is returned

(score, diff) = structural\_similarity(grayA, grayB, full=True)

diff = (diff \* 255).astype("uint8")

print("{}".format(score), file=open("D://Code/Compress/SSIM.txt", "a"))

print(n)

# threshold the difference image, followed by finding contours to

# obtain the regions of the two input images that differ

# thresh = cv2.threshold(diff, 0, 255,

# cv2.THRESH\_BINARY\_INV | cv2.THRESH\_OTSU)[1]

# cnts = cv2.findContours(thresh.copy(), cv2.RETR\_EXTERNAL,

# cv2.CHAIN\_APPROX\_SIMPLE)

#cnts = imutils.grab\_contours(cnts)

# loop over the contours

# for c in cnts:

#(x, y, w, h) = cv2.boundingRect(c)

#cv2.rectangle(imageA, (x, y), (x + w, y + h), (0, 0, 255), 2)

#cv2.rectangle(imageB, (x, y), (x + w, y + h), (0, 0, 255), 2)

# show the output images

#cv2.imshow("Original", imageA)

#cv2.imshow("Modified", imageB)

#cv2.imshow("Diff", diff)

#cv2.imshow("Thresh", thresh)

# cv2.waitKey(0)

1. The set of code for the user interactive mode are-

# importing cv2

from tkinter import filedialog

from tkinter import \*

from skimage.morphology import binary\_dilation, binary\_erosion

import random

import numpy as np

import tkinter as tk

import sys

import os

import cv2

# Importing Image class from PIL module

from PIL import Image

# Opens a image in RGB mode

def watermark\_image(watermark, logo):

# calculating dimensions

# height and width of the logo

h\_logo, w\_logo, \_ = logo.shape

# height and width of the image

h\_img, w\_img, \_ = watermark.shape

# calculating coordinates of center

# calculating center, where we are going to

# place our watermark

center\_y = int(h\_img/2)

center\_x = int(w\_img/2)

# calculating from top, bottom, right and left

top\_y = center\_y - int(h\_logo/2)

left\_x = center\_x - int(w\_logo/2)

bottom\_y = top\_y + h\_logo

right\_x = left\_x + w\_logo

# adding watermark to the image

destination = watermark[top\_y:bottom\_y, left\_x:right\_x]

result = cv2.addWeighted(destination, 1, logo, 0.5, 0)

# displaying and saving image

watermark[top\_y:bottom\_y, left\_x:right\_x] = result

cv2.imwrite("watermarked.jpg", watermark)

cv2.imshow("Watermarked Image", watermark)

cv2.waitKey(0)

cv2.destroyAllWindows()

def crop\_image(z):

im = Image.open(z)

left = int(im.size[0]/2-224/2)

upper = int(im.size[1]/2-100/2)

right = left + 224

lower = upper + 100

im\_cropped = im.crop((left, upper, right, lower))

newsize = (im.size[0], im.size[1])

im1 = im\_cropped.resize(newsize)

im1.save("cropped.jpg")

im1.show

def compress\_image(x):

def initialize\_K\_centroids(X, K):

m = len(X)

return X[np.random.choice(m, K, replace=False), :]

def find\_closest\_centroids(X, centroids):

m = len(X)

c = np.zeros(m)

for i in range(m):

# Find distances

distances = np.linalg.norm(X[i] - centroids, axis=1)

# Assign closest cluster to c[i]

c[i] = np.argmin(distances)

return c

def compute\_means(X, idx, K):

\_, n = X.shape

centroids = np.zeros((K, n))

for k in range(K):

examples = X[np.where(idx == k)]

mean = [np.mean(column) for column in examples.T]

centroids[k] = mean

return centroids

def load\_image(path):

""" Load image from path. Return a numpy array """

image = Image.open(path)

return np.asarray(image) / 255

def find\_k\_means(X, K, max\_iters=10):

centroids = initialize\_K\_centroids(X, K)

previous\_centroids = centroids

# for \_ in range(max\_iters):

image = load\_image(x)

w, h, d = image.shape

print('Image found with width: {}, height: {}, depth: {}'.format(w, h, d))

X = image.reshape((w \* h, d))

idx = find\_closest\_centroids(X, centroids)

centroids = compute\_means(X, idx, K)

if (centroids == previous\_centroids).all():

return centroids

else:

previous\_centroids = centroids

return centroids, idx

try:

image\_path = sys.argv[1]

assert os.path.isfile(x)

except (IndexError, AssertionError):

print('Please specify an image')

image = load\_image(x)

w, h, d = image.shape

print('Image found with width: {}, height: {}, depth: {}'.format(w, h, d))

X = image.reshape((w \* h, d))

K = 20

# K = 20, the desired number of colors in the compressed image

colors, \_ = find\_k\_means(X, K, max\_iters=20)

idx = find\_closest\_centroids(X, colors)

idx = np.array(idx, dtype=np.uint8)

X\_reconstructed = np.array(

colors[idx, :] \* 255, dtype=np.uint8).reshape((w, h, d))

compressed\_image = Image.fromarray(X\_reconstructed)

compressed\_image.save('compressed.jpg')

def openFile():

f = filedialog.askopenfilename(initialdir="D:\\Code\\Input\\2000",

title="Open file okay?",

filetypes=(("Image files", "\*.jpg"),

("all files", "\*.\*")))

# file = open(filepath, 'r')

# print(file.read())

# print(filepath)

# print(f, file=open("D://Code/Path.txt", "w"))

with open("D://Code/Path.txt", 'w') as p:

p.write(f)

def openFile\_1():

f = filedialog.askopenfilename(initialdir="C:\\Users\\ASUS\\Pictures\\EEE Experiment\\Code",

title="Open file okay?",

filetypes=(("Image files", "\*.jpg"),

("all files", "\*.\*")))

# file = open(filepath, 'r')

# print(file.read())

# print(filepath)

# print(f, file=open("D://Code/Path.txt", "w"))

with open("D://Code/Path\_1.txt", 'w') as p:

p.write(f)

# x = filedialog.askopenfilename()

window = Tk()

window.geometry("150x150")

window.config(background="#03071e")

button = Button(text="Open Image", command=lambda: openFile(),

foreground="#000000")

button.config(background="#ffba08")

button.config(height=2, width=10)

button.pack()

window.mainloop()

x = open("D://Code/Path.txt", "r")

y = x.read()

y = str(y)

window = Tk()

window.geometry("150x150")

window.config(background="#03071e")

button = Button(text="Open Watermark", command=lambda: openFile\_1(),

foreground="#000000")

button.config(background="#ffba08")

button.config(height=2, width=20)

button.pack()

window.mainloop()

j = open("D://Code/Path\_1.txt", "r")

i = j.read()

i = str(i)

# water(q)

print(y)

img = cv2.imread(y)

wat = cv2.imread(i)

# cv2.imshow("", wat)

# cv2.waitKey(0)

ch = 0

while(ch != 5):

print("1.to watermark the image")

print("2.to compress the image")

print("3.to crop the image")

print("4.to exit")

ch = int(input("Enter your choice: "))

if ch == 1:

watermark\_image(wat, img)

elif ch == 2:

compress\_image(y)

elif ch == 3:

crop\_image(y)

elif ch == 4:

print(

"----------------------------------Exiting------------------------------------")

break

else:

print("Enter Valid choice")

5.3 WORKING LAYOUT AND FORMATS; PROTOTYPE SUBMISSION:

1. The output of comparison of dataset-

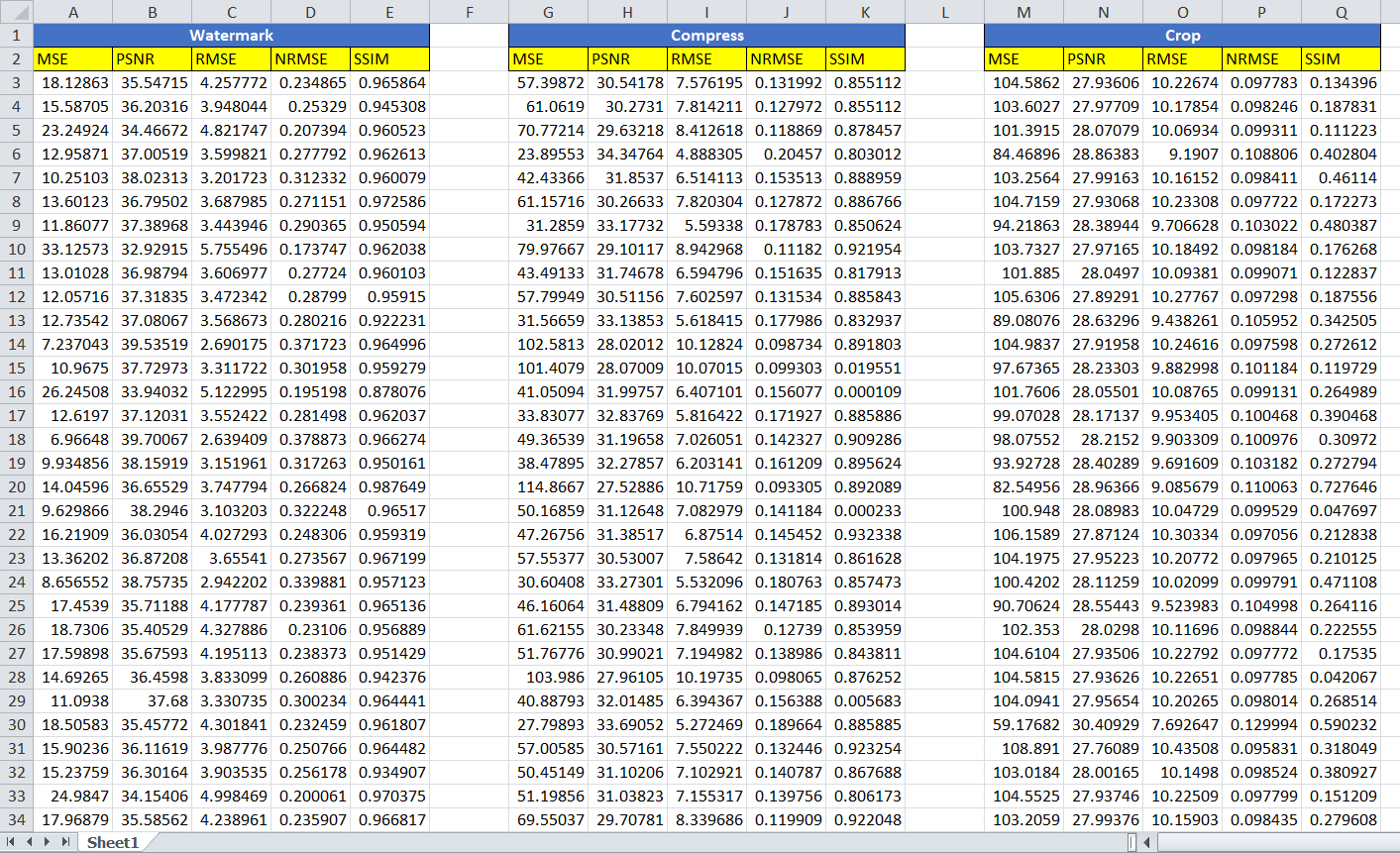


Fig 6

.

.

.

.

.

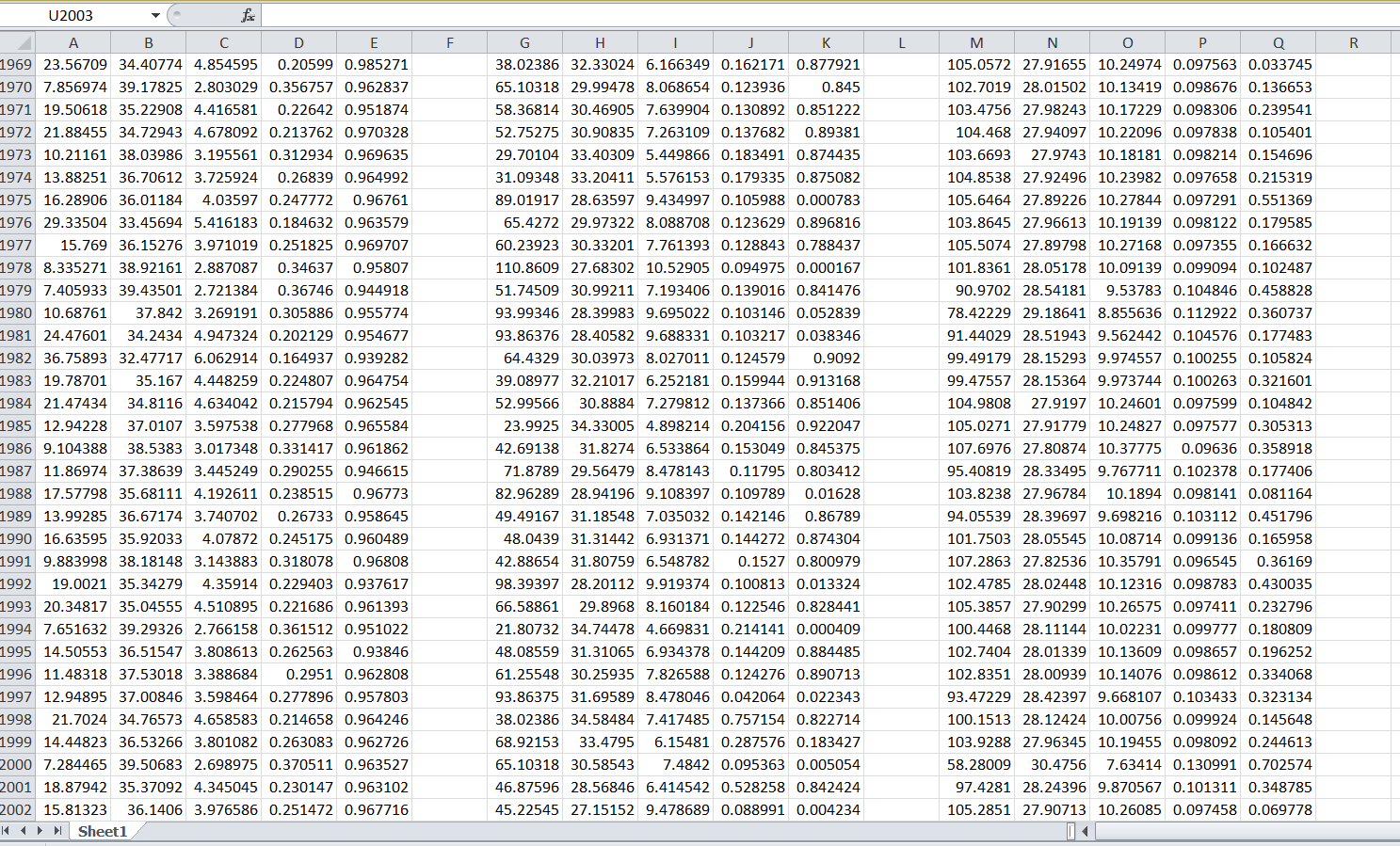


Fig 7

1. The output for the user interface mode are-



Fig 8

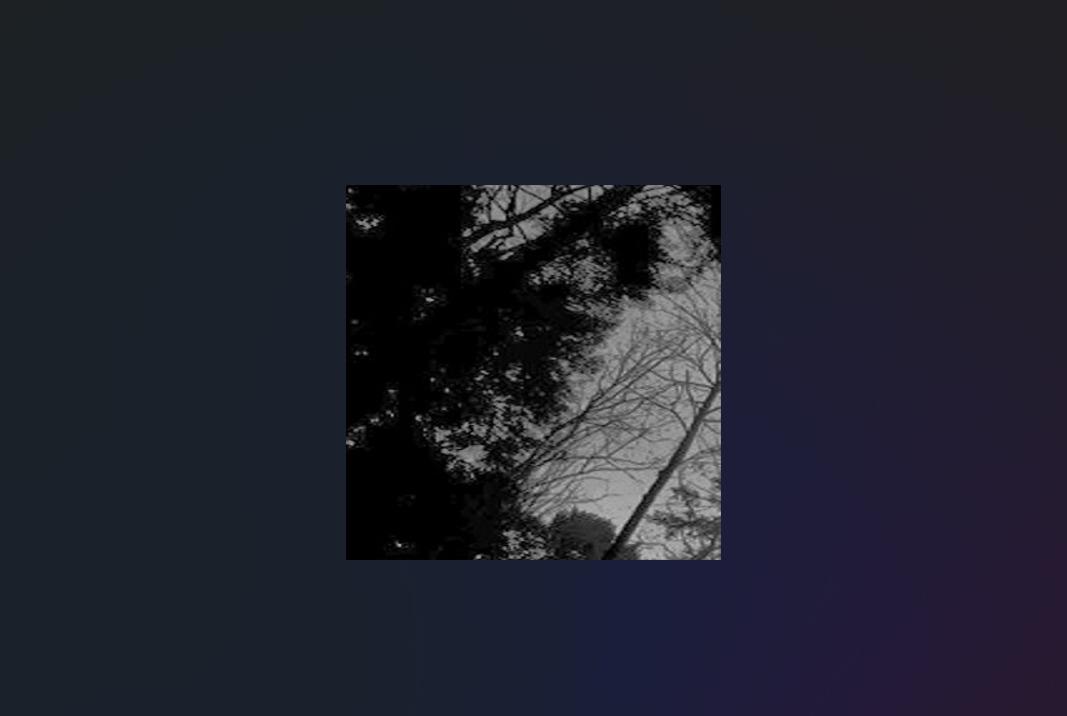


Fig 9



Fig 10



Fig 11

5.4 PERFORMANCE ANALYSIS

Performance Analysis is a specialised discipline that provides athletes and trainers with objective information that helps them understand performance. This process is sustained by methodical observation, which provides valid, dependable and detailed information relating to performance.

Hence, by using some different techniques we came to know that the performance of our application is real good.

5.5 SUMMARY

In this chapter we came to know about the outline, the working code and codes solutions, the prototype submissions and the performance analysis of our application.

**CHAPTER:6**

**PROJECT OUTCOME AND APPLICABILITY**

6.1 OUTLINE

In this chapter, we will be talking about the positive approach of our project that we have tried to build upon. What are their key implementations, what is the significance of the project, its reliability in the real world?

6.2 KEY IMPLEMENTATIONS OUTLINES OF THE SYSTEM

Perpetration is the process that actually yields the smallest- position system rudiments in the system scale ( system breakdown structure). System rudiments are made, bought, or reused. Product involves the tackle fabrication processes of forming, removing, joining, and finishing, the software consummation processes of rendering and testing, or the functional procedures development processes for drivers' roles. However, a manufacturing system which uses the established specialized and operation processes may be needed, If perpetration involves a product process.

The purpose of the perpetration process is to design and produce (or fabricate) a system element conforming to that element’s design parcels and/ or conditions. The element is constructed employing applicable technologies and assiduity practices. This process bridges the system description processes and the integration process.

6.3 SIGNIFICANT PROJECT OUTCOMES

The significant project outcomes are- the watermark image, cropped image, compresse image and the resized image.

6.4 PROJECT APPLICABILITY ON REAL WORLD

The project is applicable on following set of things-

* If a user wants to add a watermark to his original work, the software can help him/her to protect his/her work from being directly copied by anyone.
* If a user adds a wrong image/text as a watermark then he/she may the watermark using this.
* If after doing the needful operations like adding and removing, if a user wants to actually compress, resize or crop an image he/she can actually do this, using this software in a very efficient way.

6.5 INFERENCE

In this chapter, we came to know about the real world examples of our project along with the significant outcomes.

**CHAPTER:7**

**CONCLUSIONS AND RECOMMENDATIONS**

7.1 OUTLINE

PICZAP, is an application which has some cons too. As, we know every coin has two face likewise our application too have some pros and cons. So, in this chapter will be talking about some of the limitation and constraints of our application and further will be discussing about the future enhancements of the project.

7.2 LIMITATIONS/CONSTRAINTS OF THE SYSTEM

The limitation of our application may be-

* It does not remove the watermark completely.

7.3 FUTURE ENHANCEMENTS

The future enhancements for our app may be that more features can be added to the system, the watermark shall be removed completely.

7.4 INFERENCE

In this chapter, we learnt that what are the limitations of our project and what all are the future enhancements of it.

# Appendix

## A.1 Interactive Mode

### A.1.1. *Error Handling*

When an error occurs, the interpreter prints an error message and a stack trace. In interactive mode, it then returns to the primary prompt; when input came from a file, it exits with a nonzero exit status after printing the stack trace. (Exceptions handled by an except clause in a try statement are not errors in this context.) Some errors are unconditionally fatal and cause an exit with a nonzero exit; this applies to internal inconsistencies and some cases of running out of memory. All error messages are written to the standard error stream; normal output from executed commands is written to standard output.

Typing the interrupt character (usually Control-C or Delete) to the primary or secondary prompt cancels the input and returns to the primary prompt. [1](https://docs.python.org/3/tutorial/appendix.html#id2) Typing an interrupt while a command is executing raises the KeyboardInterupt exception, which may be handled by a try statement.

### A.1.2. *Executable Python Scripts*

On BSD’ish Unix systems, Python scripts can be made directly executable, like shell scripts, by putting the line

*#!/usr/bin/env python3.5*

(assuming that the interpreter is on the user’s PATH) at the beginning of the script and giving the file an executable mode. The #! must be the first two characters of the file. On some platforms, this first line must end with a Unix-style line ending ('\n'), not a Windows ('\r\n') line ending. Note that the hash, or pound, character, '#', is used to start a comment in Python.

The script can be given an executable mode, or permission, using the **chmod** command.

**$** chmod +x myscript.py

On Windows systems, there is no notion of an “executable mode”. The Python installer automatically associates .py files with python.exe so that a double-click on a Python file will run it as a script. The extension can also be .pyw, in that case, the console window that normally appears is suppressed.

### A.1.3. The Interactive Startup File

When you use Python interactively, it is frequently handy to have some standard commands executed every time the interpreter is started. You can do this by setting an environment variable named PYTHONSSTARTUP to the name of a file containing your start-up commands. This is similar to the .profile feature of the Unix shells.

This file is only read in interactive sessions, not when Python reads commands from a script, and not when /dev/tty is given as the explicit source of commands (which otherwise behaves like an interactive session). It is executed in the same namespace where interactive commands are executed, so that objects that it defines or imports can be used without qualification in the interactive session. You can also change the prompts sys.ps1 and sys.ps2 in this file.

If you want to read an additional start-up file from the current directory, you can program this in the global start-up file using code like if os.path.isfile('.pythonrc.py'): exec(open('.pythonrc.py').read()). If you want to use the startup file in a script, you must do this explicitly in the script:

**import** **os**

filename = os.environ.get('PYTHONSTARTUP')

**if** filename **and** os.path.isfile(filename):

**with** open(filename) **as** fobj:

startup\_file = fobj.read()

exec(startup\_file)

### A.1.4. *The Customization Modules*

Python provides two hooks to let you customize it: sitecustomize and usercustomize. To see how it works, you need first to find the location of your user site-packages directory. Start Python and run this code:

>>>

**>>> import** **site**

**>>>** site.getusersitepackages()

'/home/user/.local/lib/python3.5/site-packages'

Now you can create a file named usercustomize.py in that directory and put anything you want in it. It will affect every invocation of Python, unless it is started with the [-s](https://docs.python.org/3/using/cmdline.html#cmdoption-s) option to disable the automatic import.

sitecustomize works in the same way, but is typically created by an administrator of the computer in the global site-packages directory, and is imported before usercustomize. See the documentation of the site module for more details.

**REFRENCES**

1. How to think like a Computer Science, Learning with Python, Allen Downey, Jeffrey Elkner, Chris Meyers
2. Introduction to Computer Science using Python: A computational Problem Solving Focus, Charles Diebach
3. Watermark (English, Paperback, Franquiz Bob), Fuel Media Group, Inc.