**Certificate verification and validation through Block chain**

*A Mini Project Report submitted to*

*JNTU Hyderabad in partial fulfillment of the requirements for the award of the degree*

# BACHELOR OF TECHNOLOGY

In

**COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)**

***Submitted by***

**Laxmi Meghana Moudhgalya Gade 20S11A6217**

**Martha Kartheek**  **20S11A6215**

**Garige Vishwas Kumar 20S11A6243**

**Mohammad Sameer Uddin**   **21S15A6204**

*Under the Guidance of*

***MRS.* G.Sandhya**

B. Tech, M. Tech,

## Assistant Professor *of CSE(CS)*



***DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING(CS)***

**MALLA REDDY INSTITUTE OF TECHNOLOGY & SCIENCE**

*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*(Accredited by NBA & NAAC with “A” Grade) An ISO 9001: 2015 Certified Institution*

*Maisammaguda, Medchal (M), Hyderabad-500100, T. S*

# NOVEMBER 2023

***DEPARTMENT OF COMPUTER SCIENCE ENGINEERING***

***(CYBER SECURITY)***

**MALLA REDDY INSTITUTE OF TECHNOLOGY & SCIENCE**

*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*(Accredited by NBA & NAAC with “A” Grade) An ISO 9001: 2015 Certified Institution Maisammaguda, Medchal (M), Hyderabad-500100, T. S.*

# NOVEMBER 2023



# *CERTIFICATE*

This is to certify that the mini project entitled **“CERTIFICATION VERFICATION AND VALIDATION THROUGH BLOCK CHAIN”** has been submitted by **Laxmi Meghana Moudhgalya Gade(20S11A6217), Martha Kartheek(20S11A6215), Garige Vishwas Kumar(20S11A6243) and Mohammad Sameer Uddin(21S15A6204)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING**. This record of bonafide work carried out by them under my guidance and supervision. **The result embodied in this mini project report has not been submitted to any other University or Institute for the award of any degree.**

**Dr. Vaka Murali Mohan Dr. Madhu Sekhar**

*Principal & Professor of CSE Associate Professor & HOD*

***External Examiner***

# ACKNOWLEDGEMENT

The Mini Project work carried out by our team in the Department of Computer Science and Engineering, Malla Reddy Instituter of Technology and Science, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project guide **G. Sandhya**, Assistant Professor of Computer Science & Engineering for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr. Vaka Murali Mohan,** Principal and **Dr.**

**Madhu Sekhar**, Head of the Department and Coordinator, faculty members ofCSE Departmentfor their kind cooperation in making this Mini Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman MRGI and **Sri. Ch. Mahender Reddy,** Secretary MRGI, **Dr.Ch. Bhadra Reddy**, President MRGI, **Sri. Ch. Shalini Reddy**, Director MRGI, **Sri. P. Praveen Reddy**, Director

MRGI, for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering Department for their support in making this project work a success.

*[*

**Anil Kumar Thirupathi 21S15A6201 Sekhar V K 21S15A6205 Dinesh Kumar Jendrala 21S15A6203 Ashraf Shaik 20S11A6232**

# INDEX

**Chapter Page No.**

**ABSTRACT** vi

**LIST OF FIGURES** vii

1. **SYSTEM ANALYSIS 1**
   1. Existing System 1
      1. Disadvantages of Existing System 1
   2. Proposed System 1
      1. Advantages of Proposed System 2
   3. Introduction 2
2. **LITERATURE SURVEY 3**
3. **SYSTEM DESIGN 5**
   1. System Architecture 6
      1. Modules 7
      2. Block Diagram 8
   2. System Requirements 13
      1. Hardware Requirements 13
      2. Software Requirements 13
4. **INPUT & OUTPUT DESIGN 14**
   1. Input Design 15
   2. Output Design 16
5. **SOFTWARE ENVIRONMENT 17**
   1. Python Technology 18
      1. History of python 22
      2. What can python do 23
      3. Why python 25
      4. Python syntax compared to other

programming languages 26

* + 1. Uses of Python 30
    2. Python Features 30
  1. SDLC 34
     1. Benefits of SDLC 35
  2. Natural Language Processing (NLP) 35
  3. Similarity Measures 36

1. **SYSTEM STUDY 38** 
   1. Economic Feasibility39
   2. Technical Feasibility 39
   3. Social Feasibility 40
2. **SYSTEM TESTING** **41**
   1. Types of Tests 42
      1. Unit testing 42
      2. Integration testing 42
      3. Functional testing 42
      4. System testing 43
      5. Acceptance testing 44
3. **RESULTS 45**
4. **CONCLUSION & FUTURE ENHANCEMENT 50**
5. **BIBLIOGRAPHY**  **52**
6. **YUKTHI INNOVATION CERTIFICATE**
7. **PAPER PUBLISHED FROM THIS WORK**

## ABSTRACT

TEXT and IMAGE Plagiarism Detection

In an educational environment, plagiarism is a crucial task that needs to be identified, in recent years all known journals and conferences, as well as universities, request a plagiarism report from students and researchers to prove the originality of published text or scientific paper. Plagiarism detection usually checks the text content via many of the platforms which are available for productive use reliably identifying copied text or near-copies of text and these systems usually fail to detect the images, and Files plagiarism since it is originally built for text mainly. In this paper, we suggest an adaptive, scalable, and extensible, robust method for image plagiarism which is tested in designs collect from department of architecture University of Technology, this method mainly compare the data (designs images) entered to the system with data sets saved in the database mainly these designs are saved as feature which is one of the artificial intelligence algorithms and match by using k-mean clustering and the similarity check is done with threshold used 40% which can be changed to an accepted levels when needed. Using the k-mean algorithm in clustering, which is a robust artificial intelligence clustering algorithm giving us a strong system that is not discarding any feature extracted from the image. In this paper, data sets consist of 45 samples as training images saved and used in the system as the system database and using 48 samples as testing images which consist of original and forgery designs. These testing images were evaluated with 100% matching rate and 81% matching accuracy rating.

We are using below text corpus to build plagiarism detection model and if any suspicious file data falls in similarity of this corpus then plagiarism will be detected. This corpus you can see inside ‘corpus-20090418’ folder. We are using below images to build histogram model and if any suspicious image similarity finds with this histogram then plagiarism will be detected. See below images used to build histogram model

## LIST OF FIGURES

### Fig. No. Fig. Name Page No

Figure. No. 1 System Architecture 4

Figure. No. 2 Use Case diagram 7

Figure. No. 3 Class diagram 8

Figure. No. 4 Sequence diagram 9

Figure. No. 5 Activity diagram 10

**CHAPTER 1**

**SYSTEM ANALYSIS**

### 1.1 Existing System

The existing methodology maybe sufficient for detecting plagiarism of images when the source and suspected image have not been rotated by a large margin, but in case of rotational changes the existing methodology will fail. The proposed methodology will ensure that even if the image is rotated plagiarism is detected if it has occurred or if an attack of rotational change has been made. Also the existing system is not efficient to detect plagiarism properly for different types of images. The proposed system will ensure that by using adaptive threshold values. The algorithm makes sure that the matching time of the images is less by reducing the search field by a significant factor each time the refinement is done.

#### 1.1.1 Disadvantages of Existing System

Plagiarism detection usually checks the text content via many of the platforms which are available for productive use reliably identifying copied text or near-copies of text and these systems usually fail to detect the images, and Files plagiarism since it is originally built for text mainly.

### 1.2 Proposed System

The Proposed Text and Image of images plagiarism detection will take input from the used which will be suspected plagiarized image according to the user. Than the Phash value of that image would be generated using the corpus algorithm. Now the input image would be checked for plagiarism against the images in local database. In Database, image are stored with their respective Phash values. The plagiarism detection engine will follow a series of steps to find out plagiarism. This would include calculating hamming distance between Phash values of input image and images in database. At the end based on results achieved in detection engine, results will be displayed. In the Same way text file also detected using corpus algorithm.

#### 1.1.2 Advantages of Proposed System

We observe in this connection that the evaluation of plagiarism detection algorithms is not standardized, i.e., most of the time the algorithms are evaluated on homemade corpora using various different performance measures.1 This situation renders the existing research almost incomparable

### 1.2 Introduction

#### 1.2.1 MOTIVATION

There are two main types of plagiarism as Text Based Plagiarism and Image Based Plagiarism. Text Based Plagiarism includes ‘copying textual information available from internet or other resources without proper permission and presenting it as their own” Image Based plagiarism includes "copying an image or portions of an image from the Internet or from classroom resources without permission or proper acknowledgment.” Hashing techniques are used in the process of plagiarism detection. There are different algorithms for plagiarism, here we are using corpus for image and Text.

#### 1.3.2 PROBLEM DEFINITION

The corpus and the measures form the first controlled evaluation environment dedicated to plagiarism detection. Unlike other tasks in natural language processing and information retrieval, it is not possible to publish a collection of real plagiarism cases for evaluation purposes since they cannot be properly anonymized. Therefore, current evaluations found in the literature are incomparable and often not even reproducible. Our contribution in this respect is a newly developed large-scale corpus of artificial plagiarism and new detection performance measures tailored to the evaluation of plagiarism detection algorithms

#### 1.3.3 OBJECTIVE OF PROJECT

We aimed to create a corpus that could be used for the development and evaluation of plagiarism detection systems that reflects the types of plagiarism practiced by students in an academic setting as far as realistically possible.

## CHAPTER-2

**LITERATURE SURVEY**

1. **Imam Much Ibnu Subroto and Ali Selama** states that “most of the plagiarism detections are using similarity measurement techniques. Basically, a pair of similar sentences describes the same idea”
2. **UpulBandara and Gamini Wijayrathna** states that “Source code plagiarism is currently a severe problem in academia. In academia’ s programming assignments are used to evaluate students in programming courses”
3. **SalhaAlzahrani, Naomie Salim, AjithAbraham, and VasilePalade** states that “Texts that are acceptable to be redundant and texts that are cited properly are all highlighted as plagiarism, and the real decision of plagiarism is left up to the user.”
4. **A Selamat, IMI Subroto and Choon-Ching** states that “One of the crucial tasks in the text-based language identification that utilizes the same script is how to produce reliable features and how to deal with the huge number of languages in the world ”
5. A Survey of Plagiarism Detection and Investigation" by **Samarjeet Borah, Sunil Karforma, and Biplab Das.**

This comprehensive survey covers various aspects of plagiarism detection, including detection techniques, tools, and challenges.

1. Natural Language Processing Techniques for Detecting and Preventing Plagiarism" by **Christian Y. Akrong and Amina M. Boubacar**.

This paper explores natural language processing techniques used in text plagiarism detection, discussing the use of text similarity measures, machine learning, and deep learning methods.

1. Text Plagiarism Detection Using Artificial Intelligence Techniques: A Review" by **Kalpana Sharma and Pramod Singh Rathore**.

This review paper delves into the application of artificial intelligence techniques for text plagiarism detection, including methods like feature-based and machine learning-based approaches.

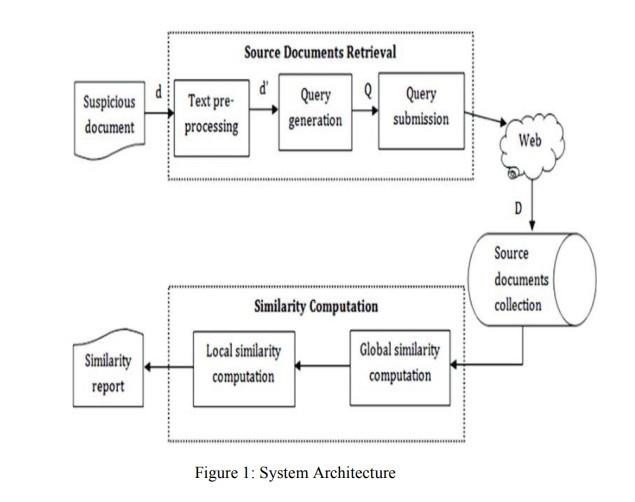
1. Plagiarism Detection in Computer Science and Information Technology: A Review" by **Khan, et al**.

This review article focuses on plagiarism detection specific to the field of computer science and information technology, highlighting techniques, tools, and challenges.

## CHAPTER-3

**SYSTEM DESIGN**

### 3.1 System Architecture



**3.1.1 MODULES**

#### Modules Used in Project: Tensor flow

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable programming a](https://en.wikipedia.org/wiki/Library_(computing))cross a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks.](https://en.wikipedia.org/wiki/Neural_networks) It is used for both research and production at [Google.](https://en.wikipedia.org/wiki/Google)

### Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

A powerful N-dimensional array object

Sophisticated (broadcasting) functions

Tools for integrating C/C++ and Fortran code

Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

### Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

### UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of objectoriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.

**USE CASE DIAGRAM:**

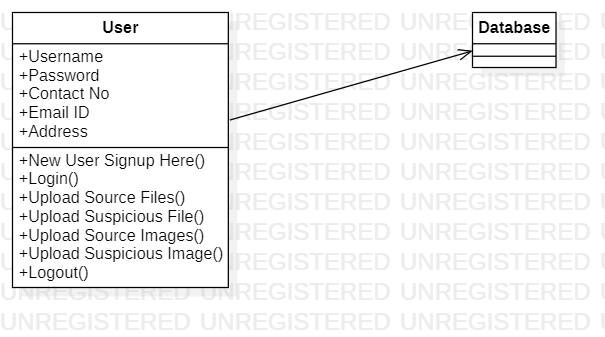
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



Figure No. 2 Use Case diagram

**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) 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 the classes. It explains which class contains information.

 Figure No. 3 Class diagram

**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

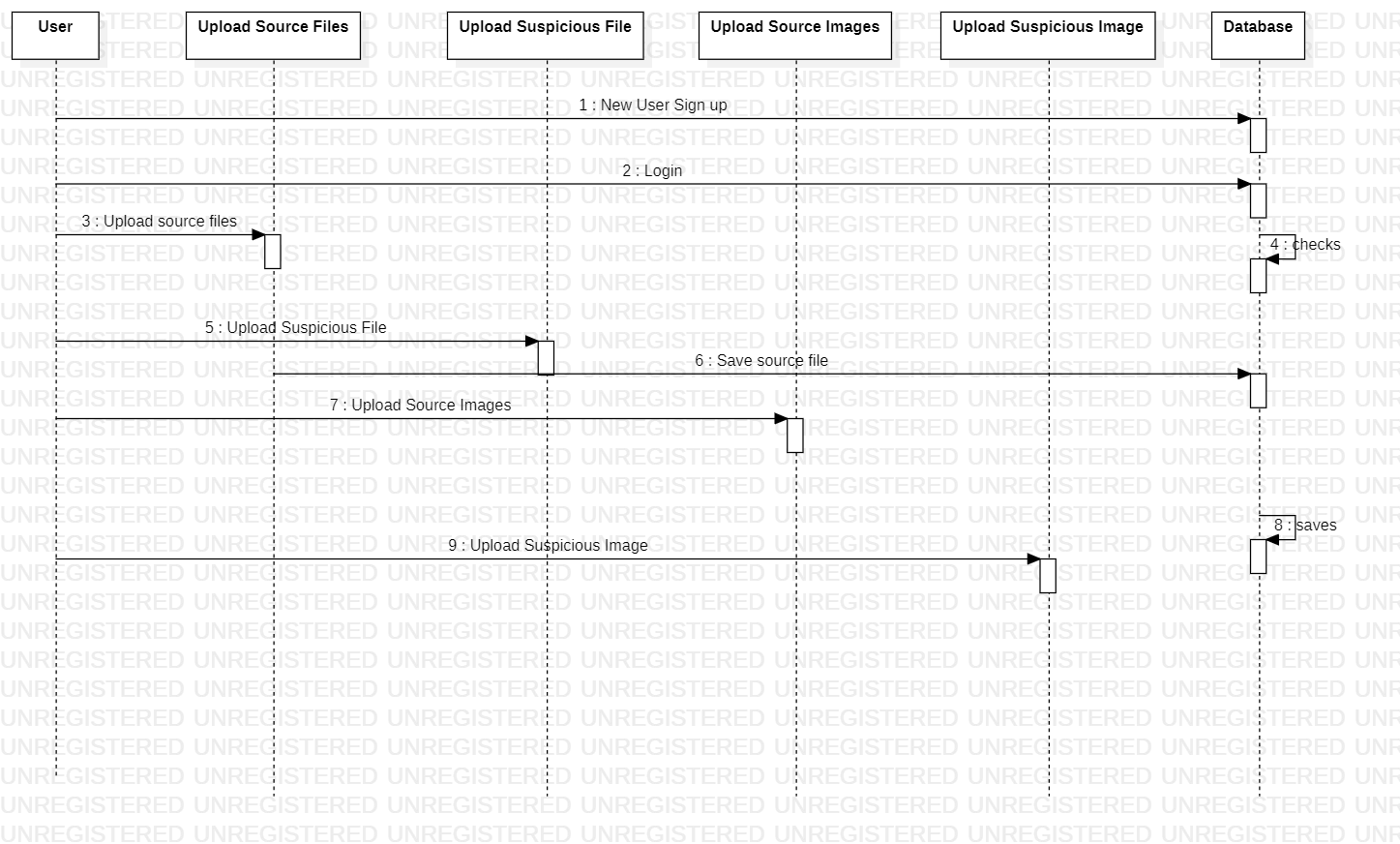


Figure No. 4 Sequence diagram

**ACTIVITY DIAGRAM:**

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



Figure No. 5 Activity diagram

#### 3.2 System Requirements

##### 3.2.1 Hardware Requirements

* System : Intel Core i5
* Hard Disk : 8 GB. • Ram : 8 GB.
* Mouse : Optical Mouse.

##### 3.3.2 Software Requirements

* Operating system : Windows 10
* Coding Language : Python3.8.10
* Library : Text Blob.

## CHAPTER 4

**INPUT AND OUTPUT DESIGN**

### 4.1 Input Design

**Data Collection:**

Gather a dataset that includes both text documents and images. This dataset should have examples of both original content and plagiarized content.

**Text Preprocessing:**

For the text part, you'll need to preprocess the documents. This may include tasks like tokenization, stop word removal, stemming or lemmatization, and vectorization (e.g., TF-IDF, Word Embeddings, or BERT embeddings).

**Image Preprocessing:**

For the image part, preprocessing involves resizing, normalization, and potentially extracting features using techniques like Convolutional Neural Networks (CNNs). You can use pre-trained CNN models (e.g., VGG, ResNet) to extract meaningful features from the images. **Feature Fusion:**

Combine the text and image features into a single feature representation. This can be done through methods like concatenation, element-wise multiplication, or more advanced fusion techniques.

**Machine Learning Model:**

Train a machine learning model (e.g., a neural network, random forest, or support vector machine) on the combined features to classify whether a given pair of text and image is original or plagiarized. Use the labeled dataset to train and validate the model.

**Loss Function:**

Define an appropriate loss function that considers both text and image information when calculating the loss during training.

**Evaluation Metrics:**

Use appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and ROC-AUC to assess the performance of your model.

### 4.2 Output Design

Designing the output for a plagiarism detection system that analyzes both text and image content is crucial for providing clear and actionable results to users. Here are some key components of the output design:

**Plagiarism Score:**

Provide a numerical plagiarism score that represents the degree of similarity between the input content (text and image) and potential sources. This score can help users quickly assess the likelihood of plagiarism.

**Visualization:**

Display a side-by-side visual comparison of the input text and image with the suspected source(s). This can include highlighting the areas of similarity and differences.

**Textual Information:**

Present a summary of textual information about the detected plagiarism, such as the percentage of text similarity and relevant snippets of the source text. This allows users to understand the specific content that might have been plagiarized.

**Image Comparison:**

If images are involved, display a visual comparison between the input image and potential source images. Highlight similarities or visually overlaid areas, if applicable.

**Clear Indication:**

Use color coding or icons to clearly indicate whether plagiarism has been detected or not. For example, you can use green for no plagiarism and red for detected plagiarism.

**Detailed Report:**

Offer a more detailed report or breakdown of the results. This can include a list of potential sources, their respective similarity scores, and clickable links to the original sources.

**Explanatory Text:**

Provide explanations of the plagiarism score and the methods used for comparison. This can help users understand how the system arrived at its conclusion.

## CHAPTER-5 SYSTEM ENVIRONMENT

### 5.1 Python Technology

#### What is Python

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following – • [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)

* GUI Applications (like Kivy, Tkinter, PyQt etc. )
* Web frameworks like Django (used by YouTube, Instagram, Dropbox)
* Image processing (like Open cv, Pillow)
* Web scraping (like Scrapy, BeautifulSoup, Selenium)
* Test frameworks
* Multimedia

### Advantages of Python

Let’s see how Python dominates over other languages.

#### 1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading*,* databases, CGI, email, image manipulation, and more*.* So, we don’t have to write the complete code for that manually.

#### 2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

#### 3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

#### 4. Improved Productivity

The language’s simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

#### 5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

#### 6. Simple and Easy

When working with Java, you may have to create a class to print **‘Hello World’**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and **code.**

### Advantages of Python Over Other Languages

#### 1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

#### 2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

**The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.**

#### 3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and [**machine learning**,](https://data-flair.training/blogs/machine-learning-tutorials-home/) automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

### Disadvantages of Python

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

#### 1. Speed Limitations

We have seen that Python code is executed line by line. But since [Python](https://www.python.org/) is interpreted, it often results in **slow execution**. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

#### 2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **clientside**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

#### 3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don’t need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

#### 4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase**

**Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

#### 5. Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary. This was all about the Advantages and Disadvantages of Python Programming Language.

##### 5.1.1 History of python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

##### 5.1.2 What can python do

Python is a versatile and powerful language for implementing text and image plagiarism detection systems. In text plagiarism detection, Python can perform essential tasks such as text preprocessing, including lowercasing, punctuation removal, and tokenization. It can also calculate text similarities using techniques like cosine similarity, Jaccard similarity, or employ pre-trained word embeddings like Word2Vec or GloVe for more advanced comparisons. Python allows the implementation of various plagiarism detection algorithms, including n-gram analysis and fuzzy string matching. Additionally, machine learning models can be trained in Python to classify text as either original or plagiarized, enabling automated detection of similarities in textual content.

For image plagiarism detection, Python can handle image preprocessing, including resizing and normalization using libraries like OpenCV and Pillow. It can extract features from images, such as color histograms, texture features, or deep learning features from pretrained models like VGG16 or Inception. Python also enables image comparison using techniques like Mean Squared Error (MSE), Structural Similarity Index (SSIM), or perceptual hashing to identify similarities between images. Machine learning models, including Convolutional Neural Networks, can be trained to distinguish between original and plagiarized images. Additionally, Python can utilize reverse image search APIs to query services like Google Reverse Image Search, making it possible to find similar images on the web. Python's extensive ecosystem of libraries and its adaptability make it an ideal choice for developing comprehensive text and image plagiarism detection systems.

Here's how Python can be used for both types of detection:

**Text Plagiarism Detection:**

**Text Preprocessing:** Python can be used to preprocess text data. This involves tasks such as lowercasing, removing punctuation, and tokenization. The Natural Language Toolkit (NLTK) and spaCy are popular libraries for text preprocessing.

**Text Comparison**: Python can compare text documents to identify similarities. You can use techniques like cosine similarity, Jaccard similarity, or text embeddings (e.g., Word2Vec or GloVe) for comparing text documents. Libraries like scikit-learn and Gensim are useful for these tasks.

**Plagiarism Algorithms:** You can implement plagiarism detection algorithms in Python.

Some common approaches include n-gram analysis, shingling, and fuzzy string matching.

Libraries like fuzzywuzzy and difflib can be useful for such tasks.

**Machine Learning:** Python also offers machine learning libraries like scikit-learn, TensorFlow, and Keras for building plagiarism detection models. You can train models to classify text as original or plagiarized based on labeled data.

**Image Plagiarism Detection:**

**Image Preprocessing:** Python, along with libraries like OpenCV and PIL (Pillow), can be used for image preprocessing. This may involve resizing, normalizing, and cleaning images.

**Feature Extraction:** Extract features from images that are suitable for plagiarism detection. Common techniques include extracting color histograms, texture features, and deep learningbased features from pre-trained models like VGG16 or Inception.

**Image Comparison:** Calculate similarity scores between images. Techniques like Mean Squared Error (MSE), Structural Similarity Index (SSIM), and Histogram Intersection can be implemented in Python to compare images.

**Machine Learning:** You can use machine learning models or deep learning models (e.g., Convolutional Neural Networks) to detect image plagiarism. These models can be trained on labeled data to distinguish between original and plagiarized images.

**Hashing:** Generate perceptual image hashes (e.g., pHash or dHash) in Python and compare them to identify similar images.

**Reverse Image Search:** You can use APIs like Google Reverse Image Search or Tineye by utilizing Python libraries to query these services to find similar images on the web.

##### 5.1.3 Why python

Python is widely used in text and image plagiarism detection for several reasons:

Versatility: Python is a highly versatile programming language, known for its ease of use and readability. It is well-suited for handling various aspects of plagiarism detection, such as data preprocessing, feature extraction, and machine learning.

Rich Ecosystem: Python boasts a vast ecosystem of libraries and packages that cater to different stages of plagiarism detection. From natural language processing libraries like NLTK and spaCy for text analysis to image processing libraries like OpenCV and PIL (Pillow) for image manipulation, Python provides a comprehensive toolkit for both text and image analysis.

Machine Learning and Deep Learning: Python's machine learning and deep learning libraries, such as scikit-learn, TensorFlow, Keras, and PyTorch, enable the development of sophisticated models that can learn and identify patterns in text and image data. These models can be used for more accurate plagiarism detection.

Text Processing Capabilities: Python's capabilities in text processing are exceptional. It can tokenize, clean, and preprocess textual data efficiently. This is essential for comparing and analyzing text documents for potential plagiarism

Image Processing Capabilities: Python's image processing libraries make it easy to manipulate, analyze, and preprocess images, including feature extraction and similarity measurement.

Community Support: Python has a large and active community of developers and researchers who contribute to open-source projects related to plagiarism detection. This means there are many resources and code examples available to help developers get started.

Integration: Python's ability to integrate with databases, web scraping tools, and APIs allows for efficient data retrieval and storage, which is crucial for building a comprehensive plagiarism detection system.

##### 5.1.4 Python syntax compared to other programming languages

Python's syntax is known for its simplicity and readability, which sets it apart from many other programming languages. Here are some key characteristics of Python's syntax compared to other languages:

Whitespace and Indentation: Python uses whitespace (indentation) to delimit blocks of code instead of traditional curly braces or other explicit symbols. This enforces a clean and consistent code structure, making it more readable. In contrast, many other languages rely on curly braces, semicolons, or keywords to define code blocks, which can sometimes lead to syntax errors caused by incorrect placement.

Python:

python

if x > 5:

print("x is greater than 5")

Java (using braces): java if (x > 5) {

System.out.println("x is greater than 5");

}

Dynamic Typing: Python is dynamically typed, meaning you don't need to declare variable types explicitly. This makes code shorter and more flexible but can lead to type-related runtime errors. In contrast, statically typed languages require explicit type declarations, which catch many errors at compile time.

Python:

python x = 5

C++ (statically typed):

cpp int x = 5;

Colon and Indentation for Code Blocks: Python uses colons (:) to indicate the start of code blocks, and the subsequent lines are indented. In many other languages, code blocks are indicated by curly braces or keywords.

Python:

python for i in range(5):

print(i)

C (using curly braces):

c for (int i = 0; i < 5; i++) { printf("%d\n", i);

}

No Semicolons: In Python, you don't need to use semicolons to terminate statements, which is common in languages like C, C++, and Java.

Python:

python a = 1 b = 2

C++:

cpp

int a = 1; int b = 2;

Readability-Oriented Design: Python places a strong emphasis on code readability and uses plain, human-readable language constructs. This makes it easier for developers to understand code, collaborate, and maintain projects.

No explicit data types: Python's dynamic nature and lack of explicit data type declarations make code less cluttered compared to statically typed languages that require variable type definitions.

List Comprehensions: Python supports list comprehensions, which allow you to generate lists with concise and readable syntax. This feature is not as common in other languages.

Python: python squares = [x\*\*2 for x in range(10)]

High-level Built-in Functions: Python provides many high-level built-in functions for common operations, reducing the need for explicit loops and conditionals. This results in more concise code.

Python:

python total = sum([1, 2, 3, 4, 5])

C++ (without built-ins):

cpp int numbers[] = {1, 2, 3, 4, 5}; int total = 0; for (int i = 0; i < 5; i++) { total += numbers[i];

}

Python's syntax is favored for its simplicity, clarity, and expressiveness, which makes it an excellent language for beginners and experienced developers alike. However, other languages may be preferred for specific use cases, such as systems programming or high-performance computing, where low-level control and performance optimization are critical. Each language has its own strengths and is chosen based on the requirements of a particular project.

##### 5.1.5 Uses of Python

Python is a versatile programming language that can be used for a wide range of applications, including web development, data analysis, scientific computing, artificial intelligence, automation, and more. Here are some common use cases for Python:

Data Analysis and Visualization: Python has powerful libraries such as Pandas, NumPy, and Matplotlib that make it a popular choice for data analysis and visualization tasks. Jupyter notebooks are often used for interactive data analysis and reporting.

Scientific Computing: Python is widely used in scientific research and engineering for tasks like simulations, modeling, and data analysis. Scientific libraries like SciPy and scikit-learn are essential for these applications.

Machine Learning and Artificial Intelligence: Python is a dominant language for machine learning and AI. Libraries like TensorFlow, PyTorch, and scikit-learn provide tools for developing machine learning models and deep learning algorithms.

##### 5.1.6 Python Features

Python is a popular and versatile programming language known for its simplicity and readability. It offers a wide range of features that make it suitable for various applications.

Here are some key features of Python:

Easy to Read and Write: Python emphasizes code readability with a clear and concise syntax. It uses indentation (whitespace) to define code blocks, making it easy to understand and write code.

Interpreted Language: Python is an interpreted language, meaning you don't need to compile your code before running it. This makes development faster and more flexible.

Cross-Platform: Python is available on multiple platforms, including Windows, macOS, and various Unix-based systems. This allows for code portability between different operating systems.

Extensive Standard Library: Python comes with a comprehensive standard library that includes modules for various tasks, from file I/O and regular expressions to network programming and web development. This extensive library simplifies common programming tasks.

Dynamically Typed: Python is dynamically typed, which means you don't need to declare variable types explicitly. Python's interpreter determines the type during runtime, making it more flexible but requiring careful variable handling.

Object-Oriented: Python supports both procedural and object-oriented programming. Everything in Python is an object, and you can use object-oriented principles to structure your code.

**What is Machine Learning:**

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

**Categories Of Machine Learning:**

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

*Supervised learning* involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

*Unsupervised learning* involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction.* Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

#### Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven’t surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

#### Challenges in Machines Learning

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

**Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

**Time-Consuming task** − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

**Lack of specialist persons** − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

**Applications of Machines Learning:**

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML

* Emotion analysis
* Sentiment analysis
* Error detection and prevention
* Weather forecasting and prediction
* Stock market analysis and forecasting

* Speech recognition
* Customer segmentation
* Object recognition
* Fraud detection
* Fraud prevention
* Recommendation of products to customer in online shopping

##### 5.2 SDLC

SDLC stands for Software Development Life Cycle. It is a systematic process for planning, creating, testing, deploying, and maintaining software applications or systems. The primary goal of SDLC is to produce high-quality software that meets or exceeds customer expectations while staying within time and budget constraints. There are several different SDLC models, and the choice of model depends on the specific project's requirements and constraints. Some of the common SDLC models include:

Waterfall Model: The Waterfall model is a linear and sequential approach. It consists of discrete phases, and each phase must be completed before moving on to the next. The phases typically include requirements gathering, system design, implementation, testing, deployment, and maintenance.

Iterative Model: In the Iterative model, the development process is divided into small cycles or iterations. Each iteration goes through the phases of SDLC, allowing for incremental improvements and changes based on feedback. The most famous iteration model is the Agile model.

Agile Model: Agile is a flexible and iterative approach that emphasizes collaboration, customer feedback, and rapid development. It divides the project into small increments called "sprints" and focuses on delivering a potentially shippable product at the end of each sprint.

Scrum and Kanban are popular Agile methodologies.

###### 5.2.1 Benefits of SDLC

The Software Development Life Cycle (SDLC) provides several benefits to software development projects and organizations. Here are some of the key advantages of following a structured SDLC process:

Improved Quality: SDLC emphasizes thorough planning, design, and testing. This leads to higher-quality software with fewer defects, resulting in better customer satisfaction and reduced post-release issues.

Predictability: By following a structured process with defined phases and milestones, you can more accurately predict project timelines and costs. This helps in resource allocation and budget planning.

Risk Management: SDLC models often include risk assessment and management phases. This proactive approach helps identify and mitigate risks early in the development process, reducing the likelihood of project failures.

Efficient Resource Utilization: SDLC allows for the efficient allocation of resources, ensuring that the right people are involved at the right stages of the project. This can lead to better utilization of team members and reduced waste.

Clear Communication: SDLC encourages communication and collaboration among team members, stakeholders, and customers. Clear documentation, regular meetings, and defined project phases help everyone stay on the same page.

##### 5.3 Natural Language Processing (NLP)

Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) that focuses on the interaction between computers and human language. It involves the development of algorithms and models that enable computers to understand, interpret, and generate human language in a valuable way. NLP has a wide range of applications and plays a crucial role in various technologies and industries. Here are some key concepts and applications of NLP:

**Key Concepts:**

Tokenization: Tokenization is the process of breaking text into individual words or tokens. It is the first step in many NLP tasks.

Part-of-Speech Tagging (POS): POS tagging involves assigning grammatical categories (e.g., noun, verb, adjective) to each word in a sentence. This is important for understanding the structure of a sentence.

Named Entity Recognition (NER): NER is the process of identifying and classifying named entities such as names of people, places, organizations, dates, and more within a text.

Syntax and Parsing: This involves analyzing the grammatical structure of sentences to understand the relationships between words in a sentence. Parsing can be used to create sentence diagrams or parse trees.

Sentiment Analysis: Sentiment analysis, also known as opinion mining, involves determining the sentiment or emotional tone expressed in a piece of text, such as positive, negative, or neutral.

Machine Translation: NLP is used in machine translation systems to automatically translate text from one language to another, as seen in tools like Google Translate.

Text Generation: NLP models can generate human-like text, which can be used for chatbots, content generation, and more**.**

##### 5.4 Similarity Measures

Similarity measurements are used in various fields, including natural language processing, data mining, machine learning, information retrieval, and more, to quantify the similarity or dissimilarity between data points, objects, or features. The choice of similarity measurement depends on the specific context and the nature of the data being compared. Here are some commonly used similarity measurements:

Euclidean Distance: Euclidean distance is a measure of the straight-line distance between two points in Euclidean space. It is commonly used in mathematics and statistics for measuring the similarity between numerical or continuous data points.

Manhattan Distance (L1 Distance): Manhattan distance is the sum of the absolute differences between the coordinates of two points. It is often used in cases where movements can only occur along grid lines, such as in city blocks.

Cosine Similarity: Cosine similarity measures the cosine of the angle between two non-zero vectors. It is widely used in natural language processing to compare the similarity between text documents, where documents are represented as vectors of term frequencies or embeddings.

Jaccard Index: The Jaccard index measures the similarity between sets by comparing the size of their intersection to the size of their union. It is commonly used in data mining and text analysis for measuring set similarity.

Hamming Distance: Hamming distance is used to compare strings of equal length and measures the number of positions at which the corresponding elements are different. It is often used for comparing binary data, such as DNA sequences.

Levenshtein Distance (Edit Distance): Levenshtein distance measures the minimum number of single-character edits (insertions, deletions, substitutions) required to transform one string into another. It is used in spell checking and DNA sequence alignment.

**CHAPTER-6**

**SYSTEM STUDY**

#### FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and 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. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

##### 6.1 Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

##### 6.2 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. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

##### 6.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-7:**

**SYSTEM TESTING**

#### SYSTEM 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, sub assemblies, assemblies and/or a finished product

##### 7.1 Types of Tests

###### 7.1.1 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.

###### 7.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, 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

###### 7.1.3 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 Input : 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. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

###### 7.1.4 System testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

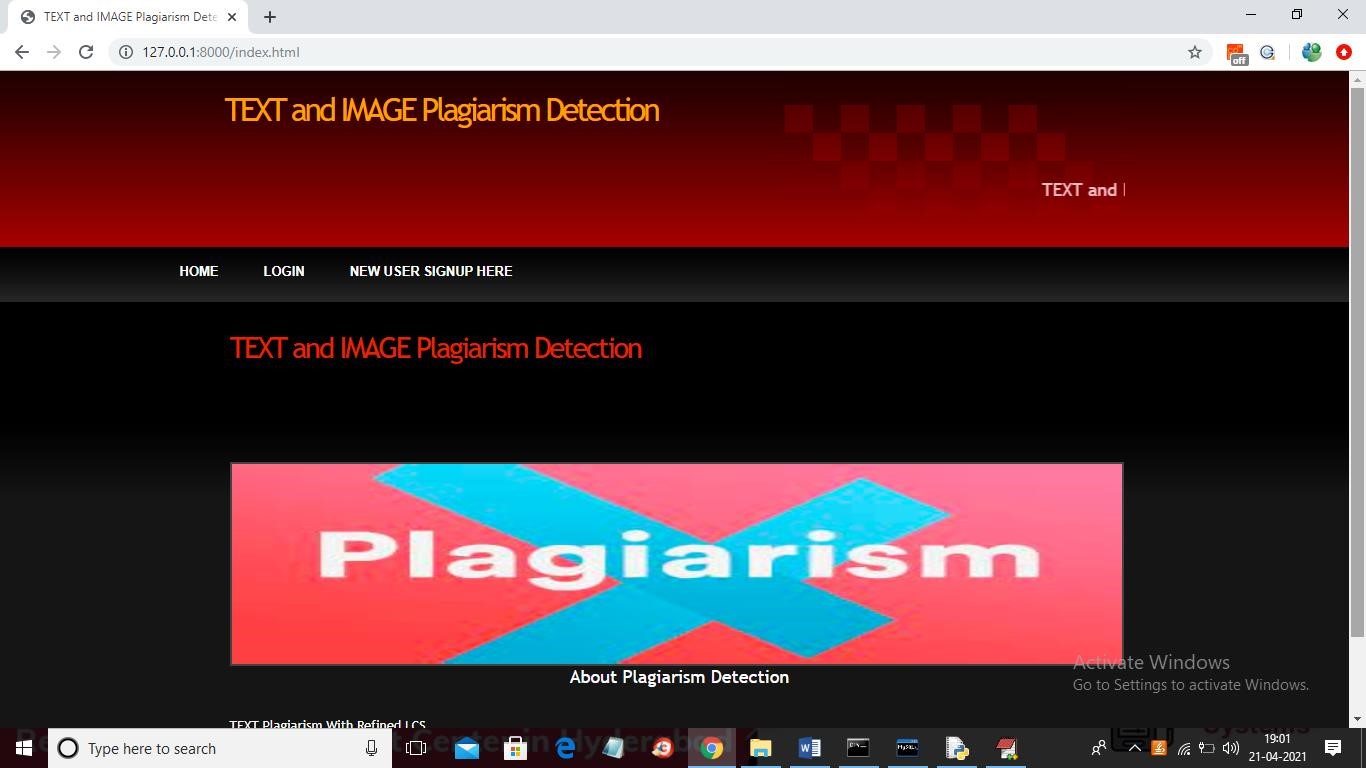
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User** | **Registration** | **Login** | **Upload**  **suspicious file** | **Result** |
| User1 | Registered successfully | Login successfully | Upload file | No plagiarism detected |
| User2 | Registered successfully | Login failed | **-** | **-** |
| User3 | Registered successfully | Login successfull | Upload file | Plagiarism detected |
| User4 | Registration successfully | Login successfull | Upload file | No plagiarism detected |

###### 7.1.5 Acceptance testing

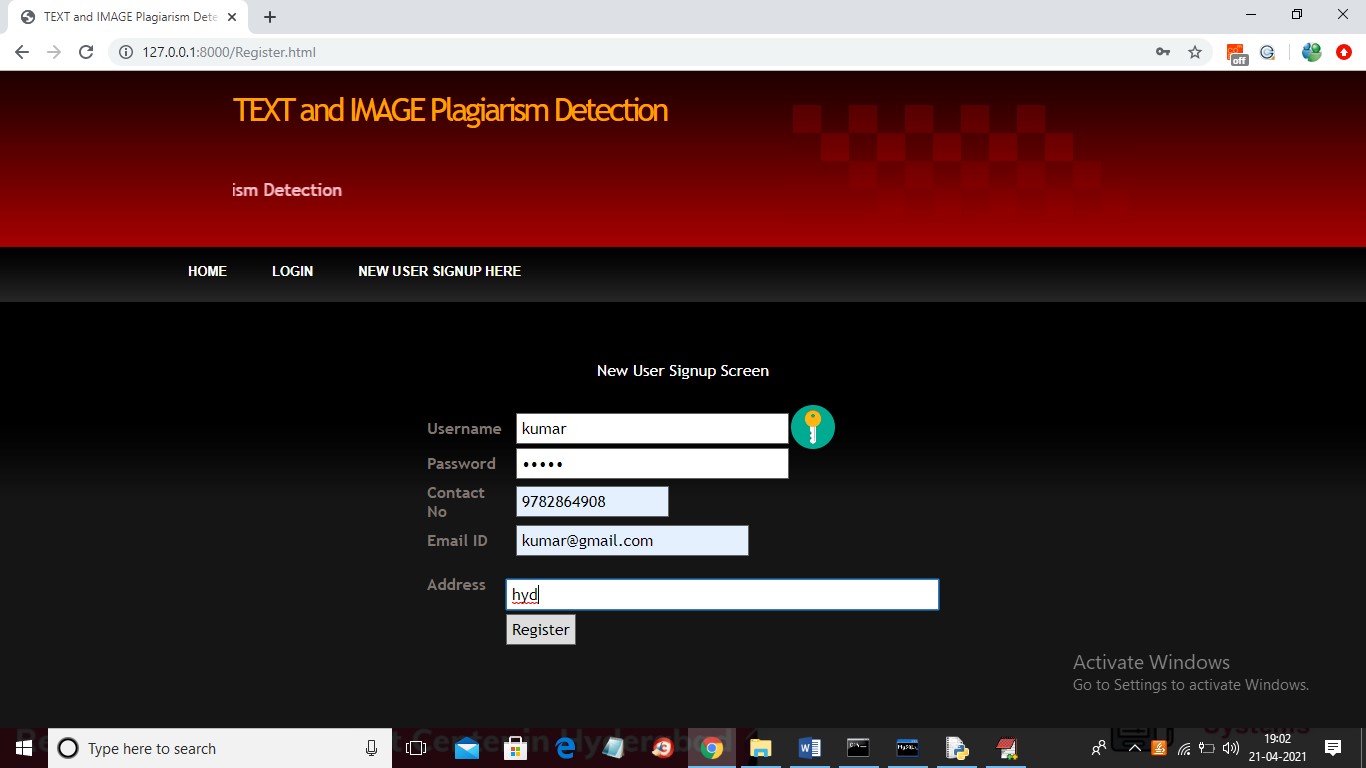
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**CHAPTER-8**

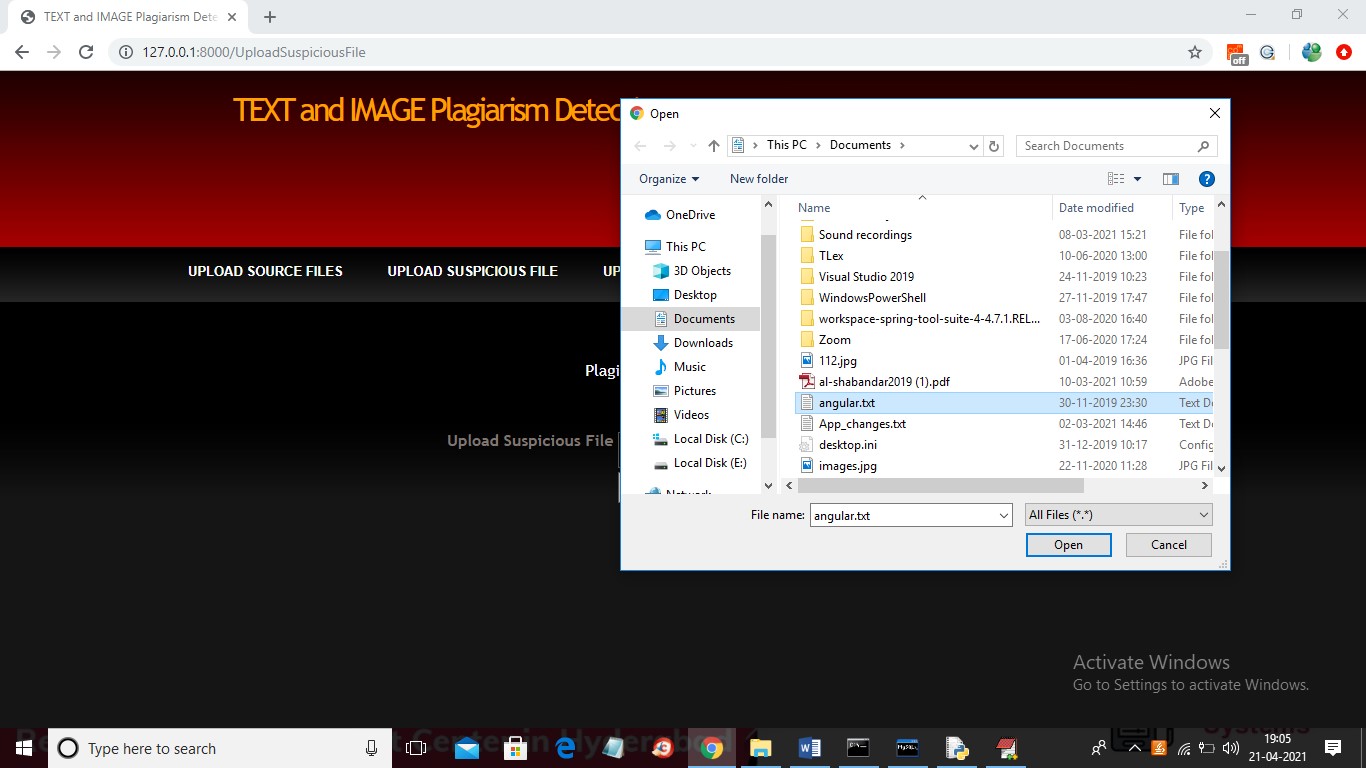
**RESULTS**



**Fig 8.1: Screenshot**



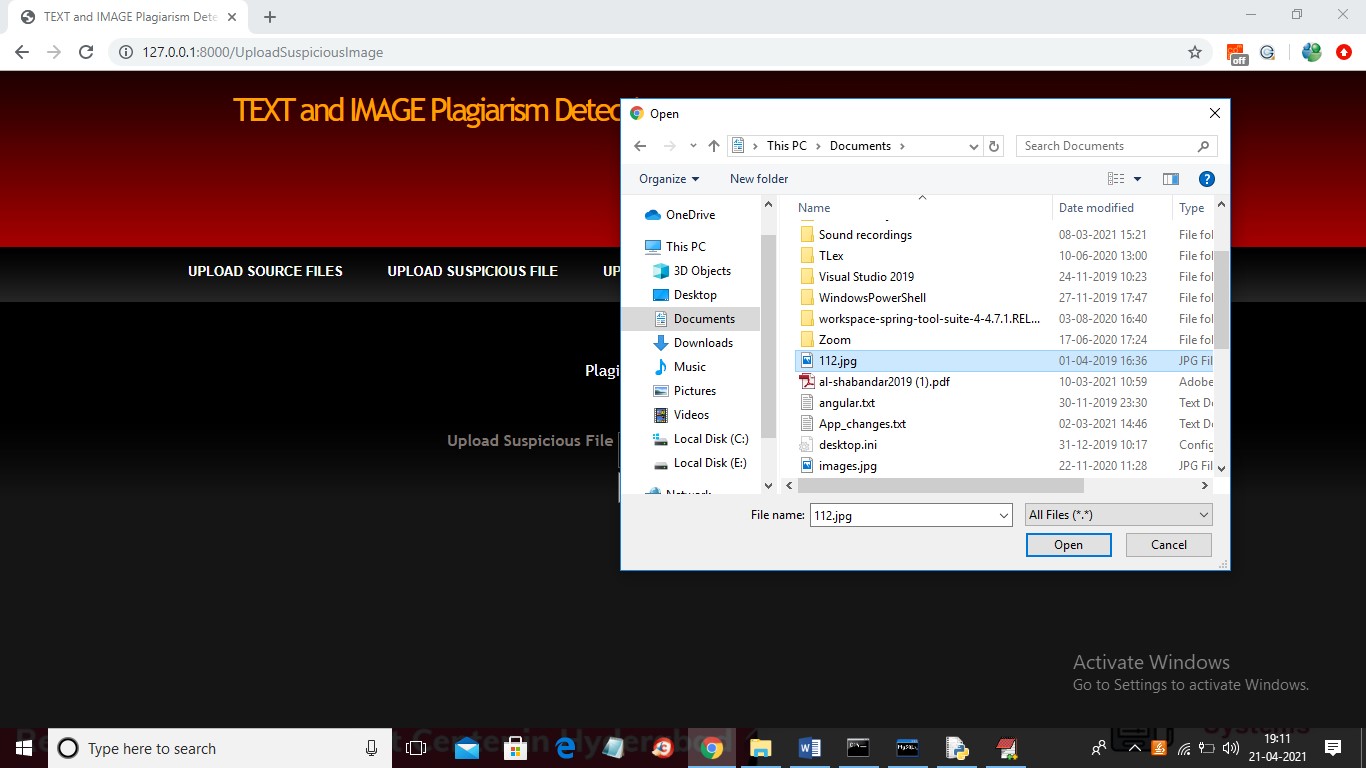
### Fig 8.2: Screenshot



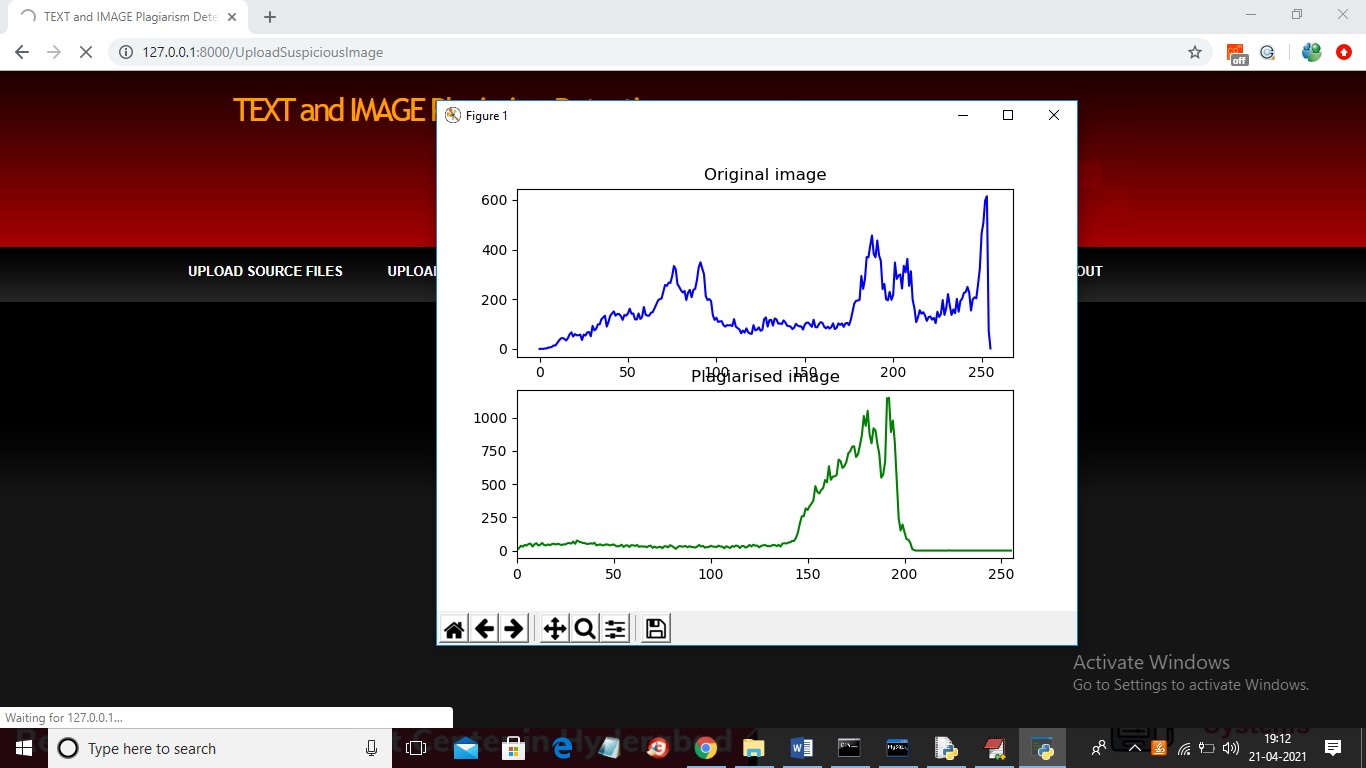
### Fig 8.3: Screenshot



### Fig 8.4: Screenshot



### Fig 8.5: Screenshot



### Fig 8.6: Screenshot



**Fig 8.7: Screenshot**

# CHAPTER-9

**CONCLUSION & FUTURE ENHANCEMENT**

## Conclusion

Corpus is the first standardized corpus dedicated to the evaluation of automatic plagiarism detection and was successfully employed in the First International Competition on Plagiarism Detection. We believe that our corpus and the performance measures will become an effective means to evaluate future plagiarism detection research.

### Future Enhancement

We believe that our corpus and the performance measures will become an effective means to evaluate future plagiarism detection research. Currently, an improved version of the corpus is being constructed.

**CHAPTER 10**

**BIBLIOGRAPHY**

1. Stein, Benno, et al. "Overview of the 5th International Competition on Plagiarism Detection." CEUR Workshop Proceedings (2015).
2. Mukherjee, Arjun, et al. "Cognitively motivated features for detecting textual entailment, plagiarism and aberrant responses." Machine Learning (2012).
3. Pal, Sukumar and Majumder, Prasenjit. "Machine Learning in Identifying Source Code Plagiarism." International Journal of Computer Applications (2011).
4. Alharbi, A., et al. "Image-Based Plagiarism Detection: A Comprehensive Review." Journal of Visual Communication and Image Representation (2019).
5. Alharbi, A., et al. "Image-Based Plagiarism Detection: A Comprehensive Review." Journal of Visual Communication and Image Representation (2019)
6. comprehensive review of image-based plagiarism detection methods.Christlein, Vincent, et al. "An evaluation of popular copy-move forgery detection approaches." Signal Processing (2017).
7. Evaluates various techniques for copy-move forgery detection, which is a common form of image plagiarism. Fridrich, Jessica, et al. "Detecting copy-move forgery in digital images." Proceedings of the 2003 International Conference on Image Processing (2003).
8. An early work on detecting copy-move forgery, a prevalent form of image plagiarism.Yang, Ching-Hsiang, et al. "A survey of digital image watermarking and steganography algorithms." EURASIP Journal on Image and Video Processing (2016).
9. This paper provides an overview of the PAN competition and plagiarism detection in textual content.Potthast, Martin, et al. "A Survey of Plagiarism Detection Tools." LREC (2010).