

## **CHAPTER-1**

### **INTRODUCTION**

A cloud typically contains a virtualized significant pool of computing resources, which could be reallocated to different purposes within short time frames. The entire process of requesting and receiving resources is typically automated and is completed in minutes. The cloud in cloud computing is the set of hardware, software, networks, storage, services and interfaces that combines to deliver aspects of computing as a service. Share resources, software and information are provided to computers and other devices on demand. It allows people to do things they want to do on a computer without the need for them to buy and build an IT infrastructure or to understand the underlying technology. Through cloud computing clients can access standardized IT resources to deploy new applications, services or computing resources quickly without reengineering their entire infrastructure, hence making it dynamic. The core concept of cloud computing is reducing the processing burden on the users terminal by constantly improving the handling ability of the cloud. All of this is available through a simple internet connection using a standard browser. However there still exist many problems in cloud computing today, a recent survey shows that data security and privacy risks have become the primary concern for people to shift to cloud computing.

#### **1.2. Literature Survey**

##### **1.2.1 Research of Cloud Computing Data Security Technology**

With cloud computing applications and research at home and abroad continue to advance cloud computing platform for users and data exchange between the greater the amount of user data transmission and storage a security threat, a cloud computing security is an important issue to be resolved. In this paper, all with state of encryption technology, presents a cloud computing data security solutions, both to ensure safe transmission of data to ensure the security of static data.

##### **1.2.2 An Efficient Method to Prevent Information Leakage in Cloud**

Cloud Computing is storing and accessing data and programs over the Internet instead of

## Enhanced Security using Elliptic Curve Cryptography in Cloud

personal computers. It is a computing paradigm shift where computing is moved away from personal computers or an individual server to a cloud of computers. Its flexibility, cost-effectiveness, and dynamically re-allocation of resources as per demand make it desirable. As desirable as it is, it has also created security challenges such as information leakage, account hijacking and denial of service. The proposed work is to develop a Software as a Service application to prevent information leakage by providing multifactor authentication, risk assessment, encryption using enhanced elliptic curve cryptography where a cryptographically secure random number generation is used to make the number unpredictable, data integrity, key management and secure disposal of information. The platform for deployment of the application is Google App Engine.

### **1.2.3 Enhancing Security of Cloud Computing using Elliptic Curve Cryptography**

Cloud computing is a form of distributed computing environment. It provides an environment where thousands of computers work in parallel to perform a job in much less times than traditional client server model. This parallelism happens because of low cost virtualization of hardware resources. Cloud computing abstracts the complexity of services provided to the user. In this article we have tried to explore various cloud computing model and how their security requirement differs from traditional computing model. We have analyzed various security risk associated with them, different ways to mitigate them and limitations of current cryptographic schemes. We have analyzed elliptic curve cryptographic schemes for cloud based applications in comparison to RSA based schemes. Here we have tried to give theoretical and experimental results to prove that elliptic curve based public key cryptography is far better than RSA based schemes. We have implemented ecdsa algorithm and compared its performance with RSA based algorithm in cloud. It supports our conclusion from the survey of cloud based applications.

### **1.2.4 The Comprehensive Approach for Data Security in Cloud Computing:**

A Survey Cloud Computing is becoming next stage platform in the evolution of the internet. It provides the customer an enhanced and efficient way to store data in the cloud with different range of capabilities and applications. The data in the cloud is stored by the service provider. Service provider capable and having a technique to protect their client data to

ensure security and to prevent the data from disclosure by unauthorized users. This paper, will give a descriptive knowledge regarding cloud computing privacy and security issues provided by encryption and decryption services. If a cloud system is performing a task of storage of data and encryption and decryption of data on the same cloud then there are much more chances of getting access to the confidential data without authorization. This increases the risk factor in terms of security and privacy. This paper helps us to propose a business model for cloud computing which focused on separating the encryption and decryption service, from the storage service provided by service provider. It means that both encryption and decryption of the data can be performed at two distinct places. For studying this proposal we are using a business model named as CRM (customer relationship model) for an example. For the evaluation of effective and efficient technique of data storage and retrieval we are providing three clouds separately such as including encryption and decryption services, secondly storage and a CRM application system. In this Research paper, we have tried to access separate encryption and decryption service using RSA algorithm and computing is a paradigm in which information is stored in servers on the internet. That information is retrieved by the client as per usage.

### **1.2.5 Enhanced Security Architecture for Cloud Data Security**

Cloud computing offers a prominent service for data storage known as cloud storage. The flow and storage of data on the cloud environment in plain text format may be a main security threat. So, it is the responsibility of cloud service providers to ensure privacy and security of data on storage as well as network level. The following three parameters confidentiality, integrity and availability decide whether security and privacy of data stored on cloud environment is maintained or not. The proposed work is to define cloud architecture with configured samba storage and cryptographic encryption techniques. The cloud architecture deployed with samba storage uses operating system feature specifying permission values for three attributes (User/Owner, Group and Global) and maps it to cryptographic application which performs cryptographic operations. Cryptography application supports symmetric and asymmetric encryption algorithm to encrypt/decrypt data for uploading/downloading within cloud storage.

## **CHAPTER-2 SYSTEM ANALYSIS**

### **2.1 EXISTING SYSTEM:**

Many of these challenges should be addressed through management initiatives. These management initiatives will require clearly delineating the ownership and responsibility roles of both the cloud provider and the organization functioning in the role of customer. Security managers must be able to determine what detective and preventative controls exist to clearly define security posture of the organization. Although proper security controls must be implemented based on asset, threat, and vulnerability risk assessment matrices. Cloud computing security risk assessment report mainly from the vendor's point of view about security capabilities analyzed security risks faced by the cloud.

#### **2.1.1 DISADVANTAGES OF EXISTING SYSTEM:**

Security managers must be able to determine what detective and preventative controls exist to clearly define security posture of the organization.

### **2.2 PROPOSED SYSTEM:**

All existing algorithms require large size of keys generation and management which took heavy computation time and resources which may increase cloud usage cost and to overcome from this problem ECC (elliptic curve cryptography) algorithm is introduced which is lighter to generate keys and take less computation time and resources to encrypt or decrypt data.

The Cloud Computing systems that provide services to the Internet users apply the public key and private or traditional identity based cryptography that has some identity elements that fit well in the requirement of cloud computing. This work aims at improving cloud computing within Cloud Organizations with encryption awareness based on Elliptic Curve Cryptography

#### **2.2.1 ADVANTAGES OF PROPOSED SYSTEM:**

The need to access cloud storage on thin clients and mobile devices is becoming an emerging application. Security of stored data and data in transit may be a concern when storing sensitive data at a cloud storage provider.

## **CHAPTER-3**

### **REQUIREMENT SPECIFICATIONS**

#### **3.1 SOFTWARE REQUIREMENTS**

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regards to what the areas of strength and deficit are and how to tackle them.

- **Python idel 3.7 version (or)**
- **Anaconda 3.7 ( or)**
- **Jupyter (or)**
- **Google colab**

#### **3.2 HARDWARE REQUIREMENTS**

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- **Operating system** : **windows, linux**
- **Processor** : **minimum intel i3**
- **RAM** : **minimum 4gb**
- **Hard disk** : **minimum 250gb**

#### **3.3 FUNCTIONALREQUIREMENTS**

1. Data Collection
2. Data Pre-processing
3. Training and Testing

#### **3.4 NON FUNCTIONALREQUIREMENTS**

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

### **3.5 SYSTEM STUDY**

#### **3.5.1 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

#### **3.5.2 ECONOMICAL 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.

### **3.5.3 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.

### **3.5.4 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.

## **3.4 NON FUNCTIONAL REQUIREMENTS**

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

- Usability requirement
- Serviceability requirement

## Enhanced Security using Elliptic Curve Cryptography in Curve

- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

### **3.5 SYSTEM STUD**

#### **3.5.1 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

#### **3.5.2 ECONOMICAL 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.

### **3.5.3 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

### **3.5.4 SOCIAL FEASIBILITY**

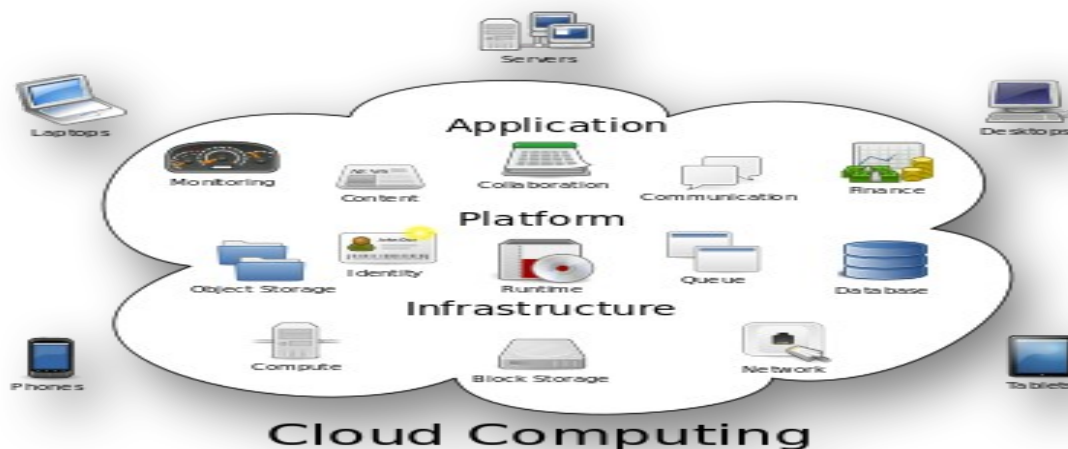
The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends 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-4**

## TECHNOLOGIES USED

### 4.1 What is cloud computing?

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers.



**Figure 4.1 : Structure of cloud computing**

### 4.2 How Cloud Computing Works?

The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power large, immersive computer games.

The cloud computing uses networks of large groups of servers typically running low-cost

consumer PC technology with specialized connections to spread data-processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing.

#### 4.3 Characteristics and Services Models:

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

- **On-demand self-service:** A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.
- **Broad network access:** Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).
- **Resource pooling:** The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.
- **Rapid elasticity:** Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.
- **Measured service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service.



**Fig 4.3 : Characteristics of cloud computing**

#### **4.4 Services Models:**

Cloud Computing comprises three different service models, namely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). The three service models or layer are completed by an end user layer that encapsulates the end user perspective on cloud services. The model is shown in figure below. If a cloud user accesses services on the infrastructure layer, for instance, she can run her own applications on the resources of a cloud infrastructure and remain responsible for the support, maintenance, and security of these applications herself. If she accesses a service on the application layer, these tasks are normally taken care of by the cloud service provider.



**Figure 4.4 :Structure of service models**

#### 4.5 Benefits of cloud computing:

- **Achieve economies of scale** – increase volume output or productivity with fewer people. Your cost per unit, project or product plummets.
- **Reduce spending on technology infrastructure.** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
- **Globalize your workforce on the cheap.** People worldwide can access the cloud, provided they have an Internet connection.
- **Streamline processes.** Get more work done in less time with less people.
- **Reduce capital costs.** There's no need to spend big money on hardware, software or licensing fees.
- **Improve accessibility.** You have access anytime, anywhere, making your life so much easier!
- **Monitor projects more effectively.** Stay within budget and ahead of completion cycle times.
- **Less personnel training is needed.** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
- **Minimize licensing new software.** Stretch and grow without the need to buy expensive software licenses or programs.
- **Improve flexibility.** You can change direction without serious “people” or “financial” issues at stake.

#### 4.6 Advantages:

- **Price:** Pay for only the resources used.
- **Security:** Cloud instances are isolated in the network from other instances for improved security.
- **Performance:** Instances can be added instantly for improved performance. Clients have access to the total resources of the Cloud's core hardware.
- **Scalability:** Auto-deploy cloud instances when needed.

## Enhanced Security using Elliptic Curve Cryptography in Curve

- **Uptime:** Uses multiple servers for maximum redundancies. In case of server failure, instances can be automatically created on another server.
- **Control:** Able to login from any location. Server snapshot and a software library lets you deploy custom instances.
- **Traffic:** Deals with spike in traffic with quick deployment of additional instances to handle the load.

### 4.7 PYTHON:

#### 4.7.1 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
- GUI(Graphical User Interface) Applications (like Kivy, Tkinter, PyQt etc. )
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

#### 4.7.2 Advantages of Python:

Let's see how Python dominates over other languages.

## Enhanced Security using Elliptic Curve Cryptography in Curve

- ***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.

- ***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.

- ***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.

- ***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.

- ***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.

- ***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**. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

### 4.7.3 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.

#### ***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**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

### 4.7.4 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 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 **client-side**. 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.

### **4.7.5 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 Venners<sup>1</sup>, 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."

### **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

#### **4.7.6 Install Python Step-by-Step in Windows and Mac:**

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

#### **4.7.7 How to Install Python on Windows and Mac:**

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheethere](#). The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

#### **Download the Correct version into the system**

**Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>

## Enhanced Security using Elliptic Curve Cryptography in Curve



Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.



**Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

## Enhanced Security using Elliptic Curve Cryptography in Curve

Looking for a specific release?

Python releases by version number:

Release version	Release date	Click for more	
<a href="#">Python 3.7.4</a>	July 8, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.6.9</a>	July 2, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.7.3</a>	March 25, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.4.10</a>	March 18, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.5.7</a>	March 18, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 2.7.16</a>	March 4, 2019	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.7.2</a>	Dec. 24, 2018	<a href="#">Download</a>	<a href="#">Release Notes</a>

**Step 4:** Scroll down the page until you find the Files option.

**Step 5:** Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
<a href="#">Gzipped source tarball</a>	Source release		68111671e5b2db4ae77b9ab01bf0f9be	23017663	<a href="#">SIG</a>
<a href="#">XZ compressed source tarball</a>	Source release		d33e4aae66097051c2eca45ee3604803	17131432	<a href="#">SIG</a>
<a href="#">macOS 64-bit/32-bit installer</a>	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e5	34896416	<a href="#">SIG</a>
<a href="#">macOS 64-bit installer</a>	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	<a href="#">SIG</a>
<a href="#">Windows help file</a>	Windows		d63999573a2c06b2ac56cade0b4f7cd2	8131761	<a href="#">SIG</a>
<a href="#">Windows x86-64 embeddable zip file</a>	Windows	for AMD64/EM64/x64	9b00c8cf8d9ec0b9abe83184a40729a2	7504391	<a href="#">SIG</a>
<a href="#">Windows x86-64 executable installer</a>	Windows	for AMD64/EM64/x64	a702b4b0ad76debd3043a583e563400	26680368	<a href="#">SIG</a>
<a href="#">Windows x86-64 web-based installer</a>	Windows	for AMD64/EM64/x64	28cb1c608bbd73ae9e53a3bd351b4bd2	1362904	<a href="#">SIG</a>
<a href="#">Windows x86 embeddable zip file</a>	Windows		9fab3b818b41879fda94133574139d8	6741626	<a href="#">SIG</a>
<a href="#">Windows x86 executable installer</a>	Windows		33cc602942a54446a3d6451476394789	25663848	<a href="#">SIG</a>
<a href="#">Windows x86 web-based installer</a>	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	<a href="#">SIG</a>

- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

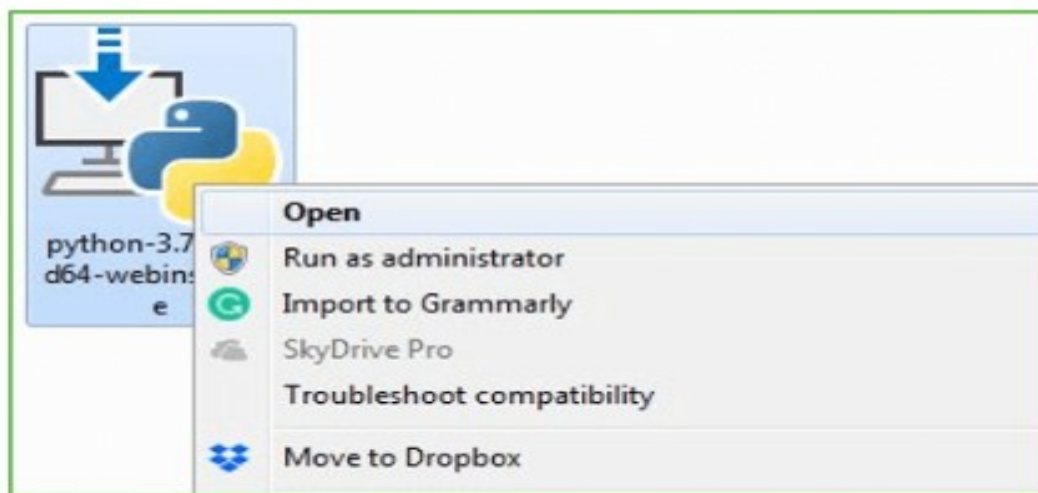
## Enhanced Security using Elliptic Curve Cryptography in Curve

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

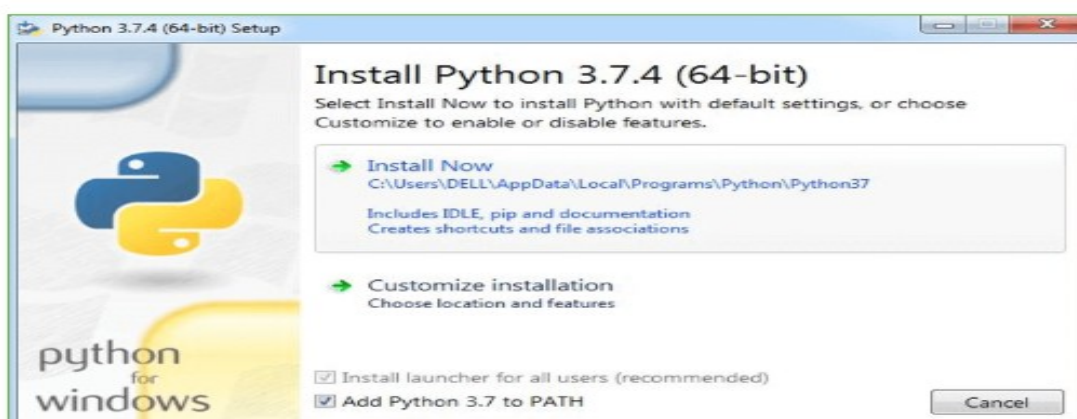
**Note:** To know the changes or updates that are made in the version you can click on the Release Note Option.

### Installation of Python

**Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.



**Step 2:** Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



**Step 3:** Click on Install NOW After the installation is successful. Click on Close.

## Enhanced Security using Elliptic Curve Cryptography in Curve



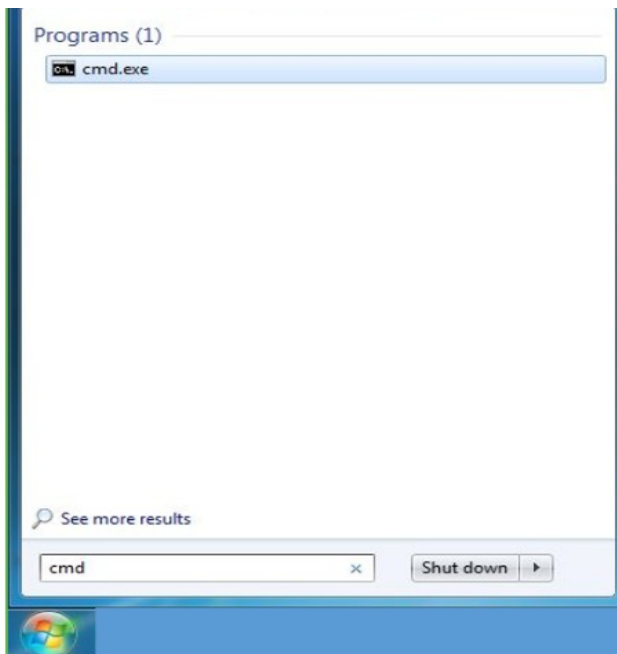
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Note:** The installation process might take a couple of minutes.

### Verify the Python Installation

**Step 1:** Click on Start

**Step 2:** In the Windows Run Command, type “cmd”.

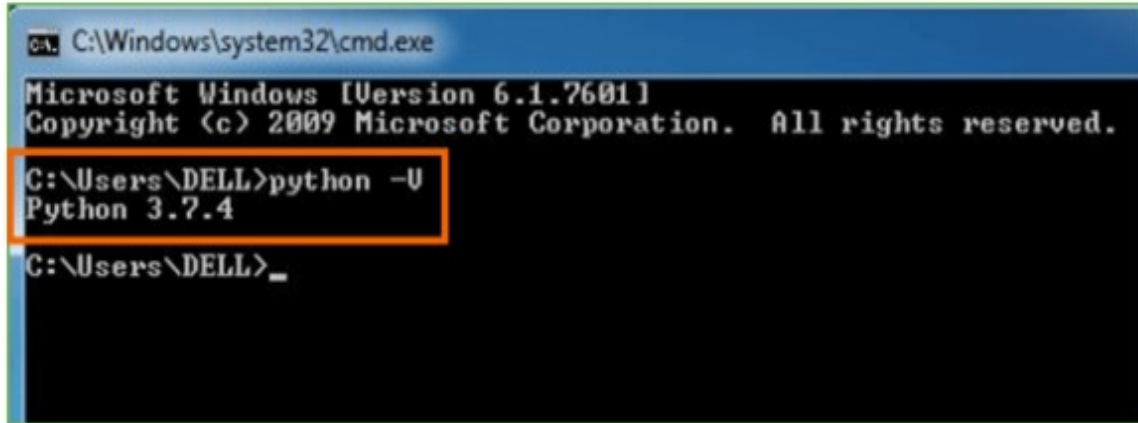




## Enhanced Security using Elliptic Curve Cryptography in Curve

**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type **python -V** and press Enter.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

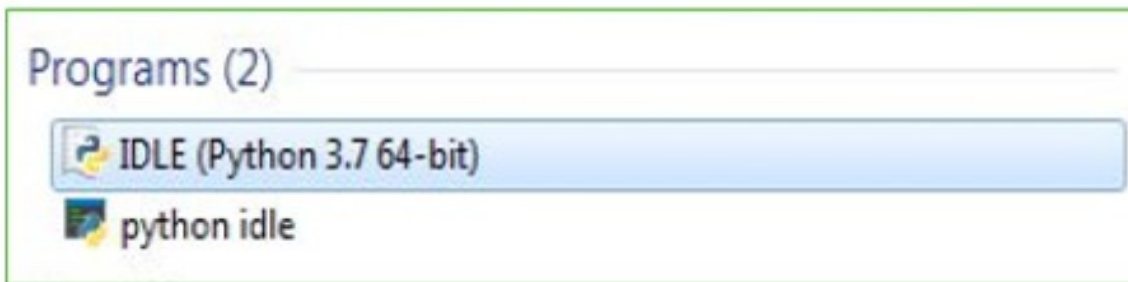
**Step 5:** You will get the answer as 3.7.4

**Note:** If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

### Check how the Python IDLE works

**Step 1:** Click on Start

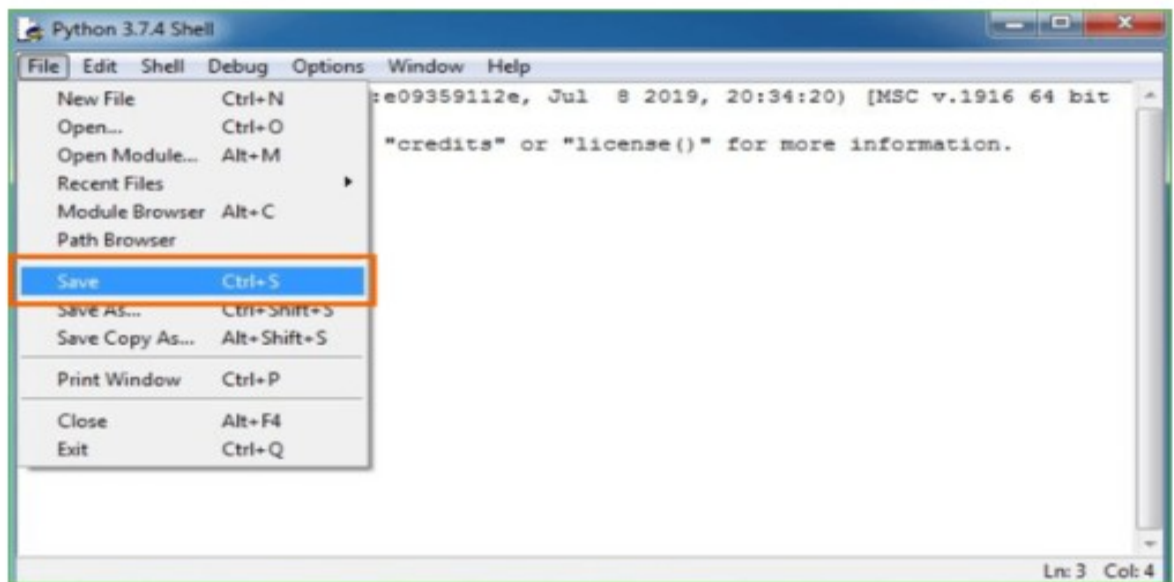
**Step 2:** In the Windows Run command, type “python idle”.



**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program

**Step 4:** To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**





**Step 5:** Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

**Step 6:** Now for e.g. enter **print**

## CHAPTER-5 SYSTEM DESIGN

### 5.1 SYSTEM ARCHITECTURE:

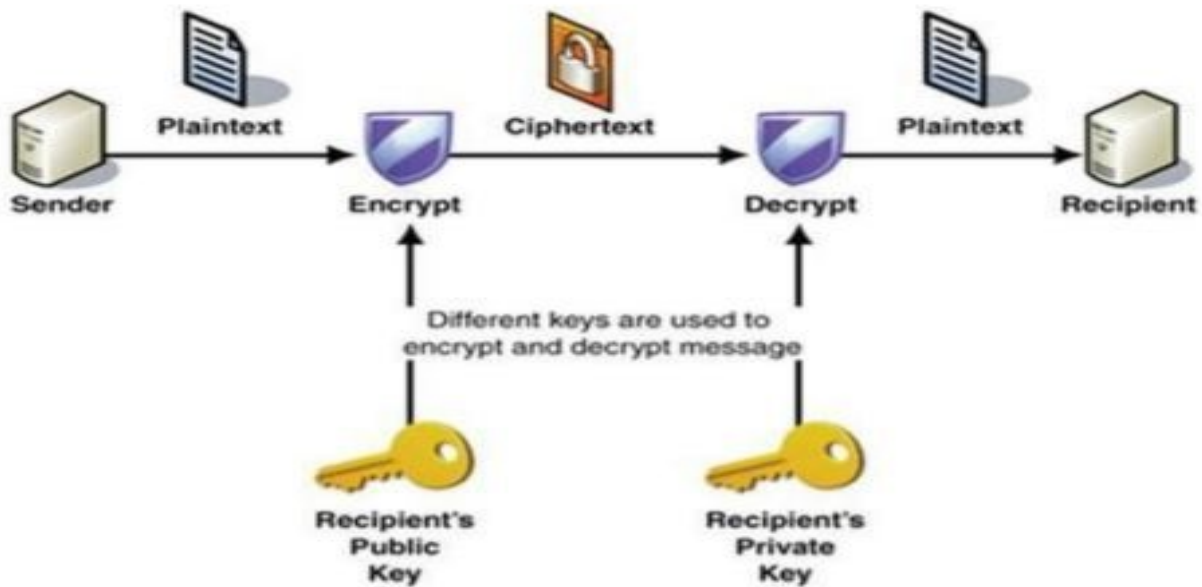
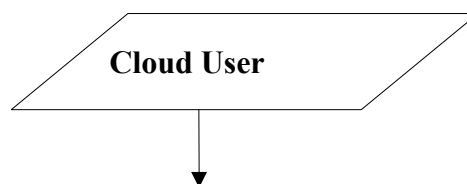


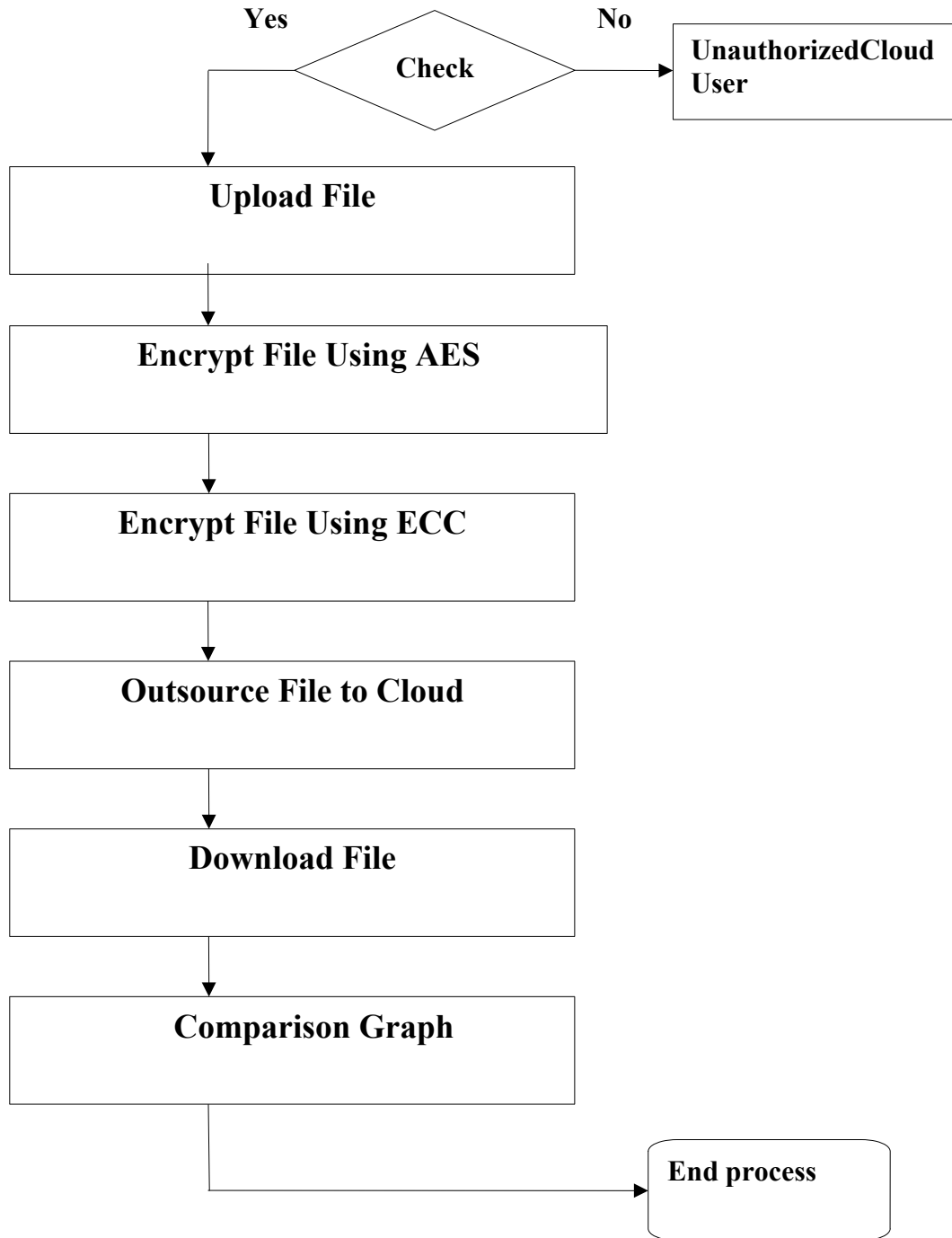
Figure 5.1 System Architecture

## 5.2 DATA FLOW DIAGRAM:

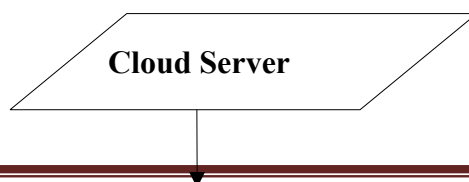
1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The Data Flow Diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

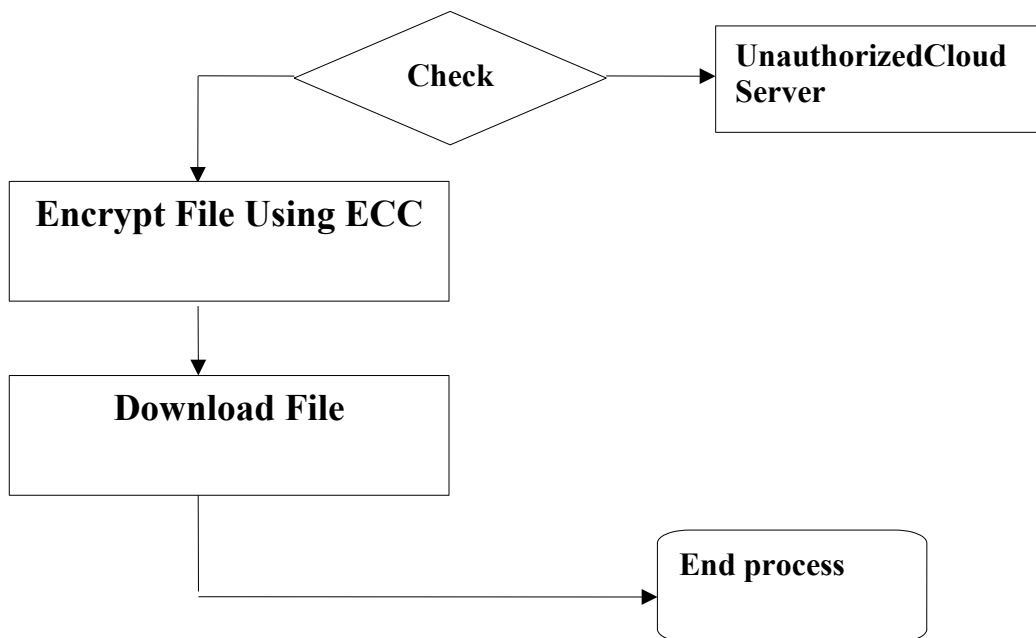
### Cloud User:





**Cloud Server:**





### 5.3 UML DIAGRAMS

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

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

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

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

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

### 5.3.1 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.

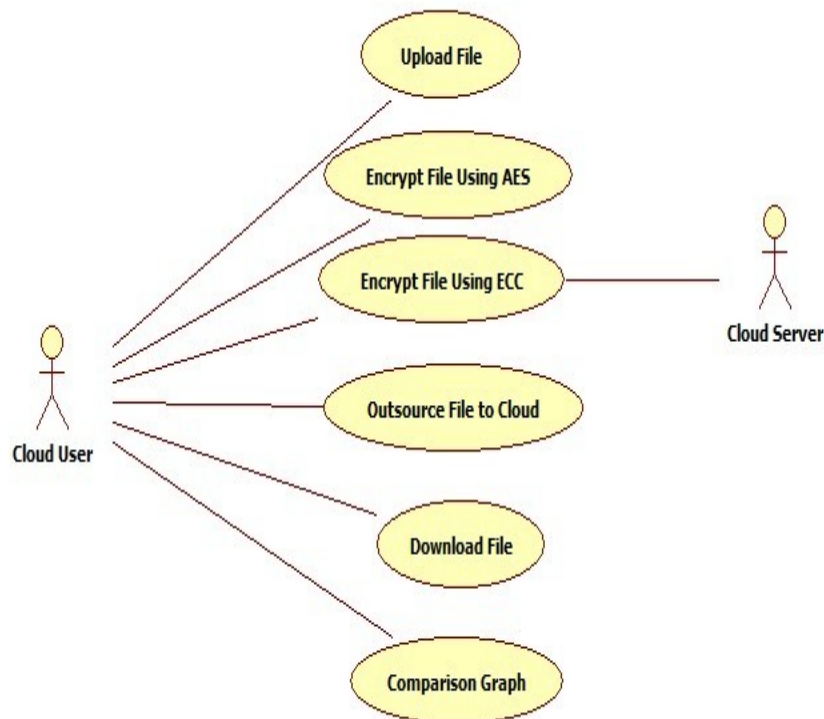


Figure 5.3.1 : Use Case Diagrams

### 5.3.2 Class diagram:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-

a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

**Figure 5.3.2 : Class Diagram**

### 5.3.3 Object diagram:

The object diagram is a special kind of class diagram. An object is an instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point of time.



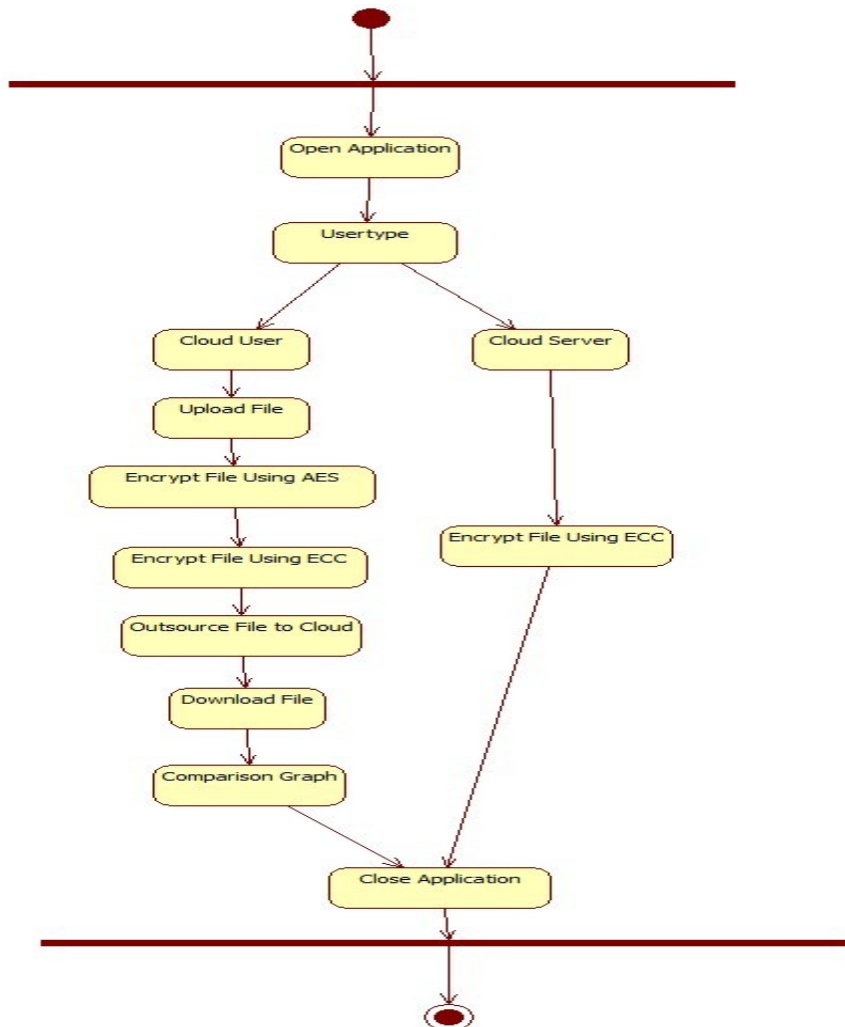
**Figure 5.3.3 : Object Diagram**

### 5.3.4 State diagram:

A state diagram, as the name suggests, represents the different states that objects in the system undergo during their life cycle. Objects in the system change states in response to events. In

## Enhanced Security using Elliptic Curve Cryptography in Curve

In addition to this, a state diagram also captures the transition of the object's state from an initial state to a final state in response to events affecting the system.



**Figure 5.3.4 : State Diagram**

### 5.3.5 Activity diagram:

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final

states, and guard conditions.

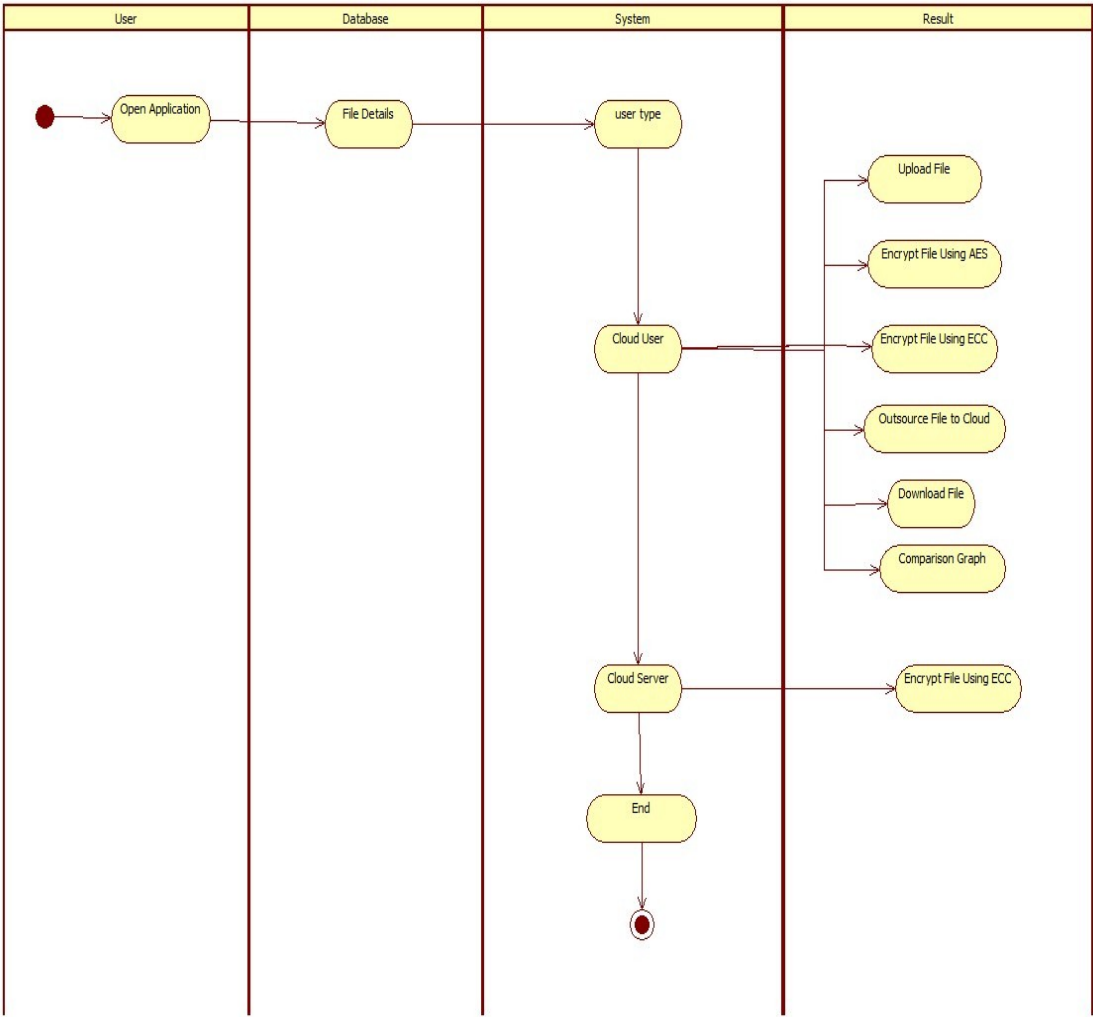


Figure 5.3.5 : Activity Diagram

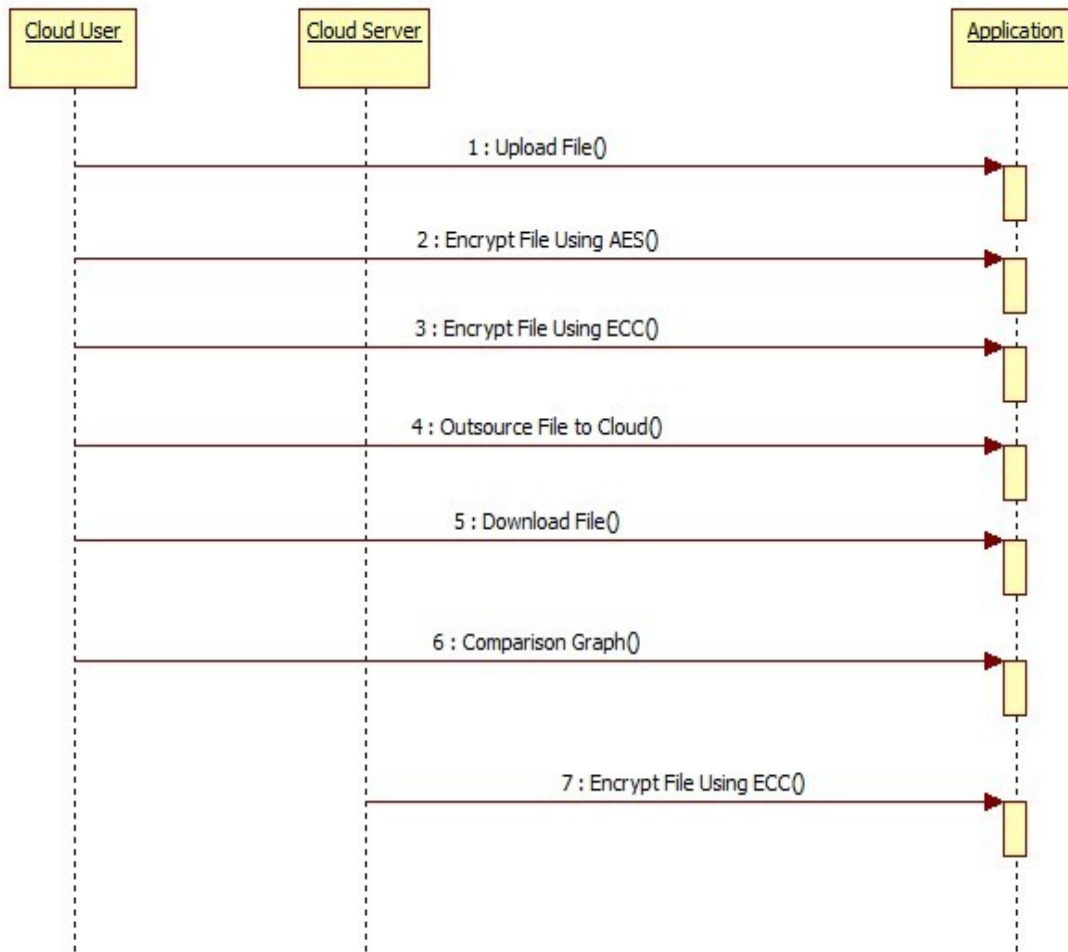
5.3.6 Sequence diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact



## Enhanced Security using Elliptic Curve Cryptography in Cloud

sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".



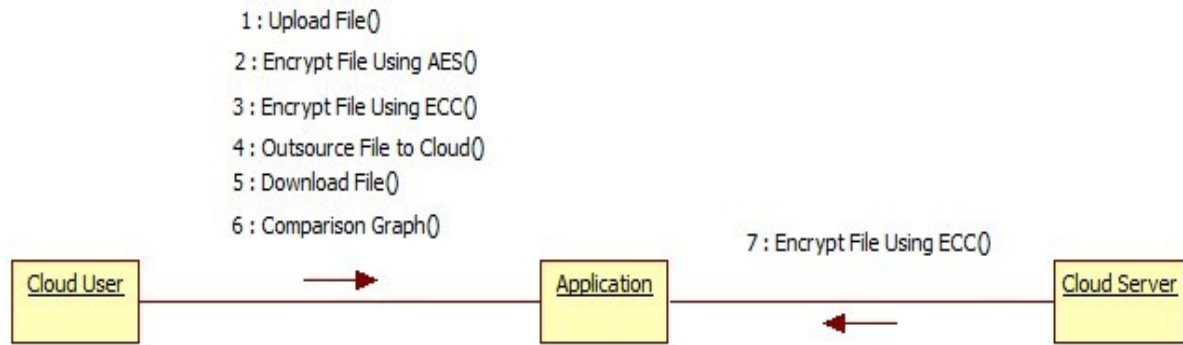
**Figure 5.3.6 : Sequence Diagram**

### 5.3.7 Collaboration diagram:

A collaboration diagram groups together the interactions between different objects. The

## Enhanced Security using Elliptic Curve Cryptography in Curve

interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.



**Figure 5.3.7 : Collaboration Diagram**

### 5.4 IMPLEMENTATION:

#### 5.4.1 MODULES DESCRIPTION:

To implement this project we have designed two different applications

1. Cloud Server: This is a python based cloud server which accept input file from user and then save in its storage space. Any time user can send request to download particular file and cloud will respond to user with that file. All files send to this cloud will be encrypted using ECC.
2. Cloud User: cloud user will upload file and then encrypt using ECC and then send or outsource to cloud for storage. Any time user can send request to cloud for file download and then decrypt it.
3. To implement this project we have designed following modules
4. Upload File: using this module we will upload any file to application
5. Encrypt File Using AES: using this module we will read file data and then encrypt it using AES algorithm and then compute encryption time
6. Encrypt File Using ECC: using this module we will encrypt file using ECC algorithm and then calculate encryption time

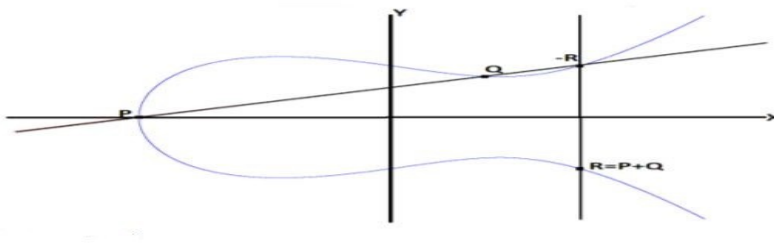
## Enhanced Security using Elliptic Curve Cryptography in Curve

7. Outsource File to Cloud: using this module we will outsource file to cloud server for storage
8. Download File: using this module we will send file request to cloud and then download and decrypt the file
9. Comparison Graph: using this module we will plot encryption time graph between AES and ECC algorithm

### 5.5 ALGORITHMS:

#### 5.5.1 ELLIPTIC CURVE CRYPTOGRAPHY (ECC)

Elliptic Curve (EC) systems as applied to cryptography were first proposed in 1985 independently by Neal Koblitz and Victor Miller. Public-key cryptography is based on the intractability of certain mathematical problems. Early public-key systems, such as the RSA algorithm, are secure assuming that it is difficult to factor a large integer composed of two or more large prime factors. It is believed that the same level of security afforded by an RSA-based system with a large modulus can be achieved with a much smaller elliptic curve group. For current cryptographic purposes, an elliptic curve is a plane curve which consists of the points satisfying the equation:  $Y^2 = x^3 + ax + b$  along with a distinguished point at infinity, denoted " $\infty$ ". (The coordinates here are to be chosen from a fixed finite field of characteristic not equal to 2 or 3, or the curve equation will be somewhat more complicated). This set together with the group operation of the elliptic group theory form an Abelian group, with the point at infinity as identity element. The structure of the group is inherited from the divisor group of the underlying algebraic variety.



**Fig 5.5.1 : Elliptic Curve**

If P and Q are on E,  $R = P + Q$

As shown in Fig.2, Let  $P=(x_1, y_1)$ ,  $Q=(x_2, y_2)$

## Enhanced Security using Elliptic Curve Cryptography in Curve

$R=(x_3, y_3)$  and  $P$  not equals  $Q$

$$m = \frac{y_2 - y_1}{x_2 - x_1};$$

To find intersection with  $E$ , we get

$$(m(x-x_1)+y_1)^2 = x^3 + Ax + B$$

$$\text{Or, } 0 = x^3 - m^2 x^2 + \dots$$

$$\text{So, } x_3 = m^2 - x_1 - x_2$$

$$y_3 = m(x_1 - x_2) - y_1$$

Elliptic curves are used as an extension to other current cryptosystems. ECC is considered as more secured algorithm than other asymmetric algorithms such as RSA and Diffie-Hellman by providing same level of security with smaller key size. For example, ECC can provide a level of security with a 256-bit public key that other techniques require a 3072-bit public key. Thus ECC has some advantages include low CPU consumption, low memory usage and greater speed. The difficulty of discrete logarithm makes ECC so important.

## CHAPTER-6 CODING AND IMPLEMENTATION

**Cloud User**

```
fromtkinter import messagebox
fromtkinter import *
fromtkinter import simpledialog
importtkinter
fromtkinter import filedialog
fromtkinter.filedialog import askopenfilename
importmatplotlib.pyplot as plt
importpyaes, pbkdf2, binascii, os, secrets
import base64
importtimeit
fromecies.utils import generate_eth_key, generate_key
fromecies import encrypt, decrypt
import pickle
import socket
importjson
fromtkinter import messagebox
importnumpy as np

main = tkinter.Tk()
main.title("Security using Elliptic Curve Cryptography (ECC) in Cloud") #designing main
screen
main.geometry("1300x1200")

global filename
execution_time = []
global data
globalsecret_key, private_key, public_key
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
def getAESKey(): #generating key with PBKDF2 for AES
    password = "s3cr3t*c0d3"
    passwordSalt = '76895'
    key = pbkdf2.PBKDF2(password, passwordSalt).read(32)
    return key

def encryptAES(plaintext): #AES data encryption
    aes = pyaes.AESModeOfOperationCTR(getAESKey(),
    pyaes.Counter(311295470350000473029524339676541953981242398445663228841721636
    37846056248223))
    ciphertext = aes.encrypt(plaintext)
    return ciphertext

def decryptAES(self, enc): #AES data decryption
    aes = pyaes.AESModeOfOperationCTR(getAESKey(),
    pyaes.Counter(311295470350000473029524339676541953981242398445663228841721636
    37846056248223))
    decrypted = aes.decrypt(enc)
    return decrypted

def upload(): #function to upload tweeter profile
    global filename
    global data
    filename = filedialog.askopenfilename(initialdir="Dataset")
    text.delete('1.0', END)
    text.insert(END, filename + " loaded\n\n");
    fout = open(filename, 'rb')
    data = fout.read()
    fout.close()
    text.insert(END, str(data))
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
defAESEncrypt():
    global data
    text.delete('1.0', END)
    start = timeit.default_timer()
    encrypted_data = encryptAES(str(base64.b64encode(data),'utf-8'))
    end = timeit.default_timer()
    aes_time = end - start
    execution_time.append(aes_time)
    text.insert(END,"AES encryption time: "+str(aes_time)+"\n\n")
    text.insert(END,"AES Encrypted Data: "+str(encrypted_data)+"\n\n")

#function to generate ECC public, private and secret keys
defgenerateKeys():
    globalsecret_key, private_key, public_key
    ifos.path.exists('public.pckl'):
        f = open('public.pckl', 'rb')
        public_key = pickle.load(f)
        f.close()
        f = open('private.pckl', 'rb')
        private_key = pickle.load(f)
        f.close()
    else:
        secret_key = generate_eth_key()
        private_key = secret_key.to_hex() # hex string
        public_key = secret_key.public_key.to_hex()
        f = open('public.pckl', 'wb')
        pickle.dump(public_key, f)
        f.close()
        f = open('private.pckl', 'wb')
        pickle.dump(private_key, f)
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
f.close()

defECCEncrypt(): #ECC data encryption
global data
globalsecret_key, private_key, public_key
text.delete('1.0', END)
start = timeit.default_timer()
generateKeys()
data = encrypt(public_key, data)
end = timeit.default_timer()
ecc_time = end - start
execution_time.append(ecc_time)
text.insert(END, "ECC encryption time: "+str(ecc_time)+"\n\n")
text.insert(END, "ECC Encrypted Data: "+str(data)+"\n\n")

defoutsourseFile():
global filename
global data
text.delete('1.0', END)
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 2222))
features = []
features.append("upload")
features.append(os.path.basename(filename))
features.append(data)
features = pickle.dumps(features)
client.send(features) #now sending encrypted file data to cloud
data = client.recv(10000)#now receive response from cloud
data = pickle.loads(data)
data = data[0]
```



## Enhanced Security using Elliptic Curve Cryptography in Curve

```
text.insert(END,data+"\n")
defdownloadFile():
text.delete('1.0', END)
fname = simpdialog.askstring(title = "Enter filename to download", prompt = "Enter
filename to download")
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 2222))
features = []
features.append("download")
features.append(fname)
features = pickle.dumps(features)
client.send(features) #now sending features to cloud
data = client.recv(10000)#now receive labels from cloud after clustering
data = pickle.loads(data)
data = data[0]
decryptData = decrypt(private_key, data)
fout = open(fname, 'wb')
fout.write(decryptData)
fout.close()
text.insert(END,"file downloaded and saved inside "+fname+"\n")

def graph():
height = execution_time
bars = ('AES Encryption Time','ECC Encryption Time')
y_pos = np.arange(len(bars))
plt.bar(y_pos, height)
plt.xticks(y_pos, bars)
plt.title("AES VS ECC Encryption Time Graph")
plt.show()
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
font = ('times', 16, 'bold')
title = Label(main, text='Security using Elliptic Curve Cryptography (ECC) in Cloud')
title.config(bg='darkviolet', fg='gold')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)
font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50,y=120)
text.config(font=font1)

font1 = ('times', 12, 'bold')
uploadButton = Button(main, text="Upload File", command=upload)
uploadButton.place(x=50,y=550)
uploadButton.config(font=font1)

aesButton = Button(main, text="Encrypt File Using AES", command=AESEncrypt)
aesButton.place(x=430,y=550)
aesButton.config(font=font1)

eccButton = Button(main, text="Encrypt File Using ECC", command=ECCEncrypt)
eccButton.place(x=50,y=600)
eccButton.config(font=font1)

outsourceButton = Button(main, text="Outsource File to Cloud", command=outsourceFile)
outsourceButton.place(x=430,y=600)
outsourceButton.config(font=font1)
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
downloadButton = Button(main, text="Download File", command=downloadFile)
downloadButton.place(x=50,y=650)
downloadButton.config(font=font1)
```

```
graphButton = Button(main, text="Comparison Graph", command=graph)
graphButton.place(x=430,y=650)
graphButton.config(font=font1)
```

```
main.config(bg='sea green')
main.mainloop()
```

### **Cloud Server :**

```
fromtkinter import *
importtkinter
import socket
from threading import Thread
fromsocketserver import ThreadingMixIn
fromos import path
import pickle
import sys
importnumpy as np

mainGUI = tkinter.Tk()
mainGUI.title("Cloud Server") #designing main screen
mainGUI.geometry("900x700")
running = True

defstartDistributedCore():
    classCoreThread(Thread):
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
def __init__(self,ip,port):
    Thread.__init__(self)
    self.ip = ip
    self.port = port
    text.insert(END,'Request received from Client IP : '+ip+' with port no : '+str(port)+'\n')

def run(self):
    text.delete('1.0', END)
    global details
    data = conn.recv(100000) #receive encrypted data
    dataset = pickle.loads(data) #convert binary data into encrypted data
    request_type = dataset[0]
    if request_type == "upload":
        fname = dataset[1]
        encrypted_data = dataset[2]
        fout = open('upload/'+fname, 'wb')
        fout.write(encrypted_data)
        fout.close()
        text.insert(END,fname+" File received and saved at Cloud Server\n\n")
        output = []
        output.append(fname+" File received and saved at Cloud Server")
        output = pickle.dumps(output)
        conn.send(output)
        if request_type == "download":
            fname = dataset[1]
            fout = open('upload/'+fname, 'rb')
            data = fout.read()
            fout.close()
            text.insert(END,fname+" sent to user for download\n\n")
            output = []
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
output.append(data)
output = pickle.dumps(output)
conn.send(output)

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
server.bind(('localhost', 2222))
threads = []
text.insert(END, "Cloud Server Started\n\n")
while running:
    server.listen(4)
    (conn, (ip,port)) = server.accept()
    newthread = CoreThread(ip,port)
    newthread.start()
    threads.append(newthread)
for t in threads:
    t.join()

defstartCore():
    Thread(target=startDistributedCore).start()

gfont = ('times', 16, 'bold')
gtitle = Label(mainGUI, text='Cloud Server 1')
gtitle.config(bg='darkviolet', fg='gold')
gtitle.config(font=gfont)
gtitle.config(height=3, width=120)
gtitle.place(x=0,y=5)

gfont1 = ('times', 12, 'bold')
```

## Enhanced Security using Elliptic Curve Cryptography in Curve

```
text=Text(mainGUI,height=28,width=130)
```

```
scroll=Scrollbar(text)
```

```
text.configure(yscrollcommand=scroll.set)
```

```
text.place(x=10,y=100)
```

```
text.config(font=gfont1)
```

```
startCore()
```

```
mainGUI.config(bg='sea green')
```

```
mainGUI.mainloop()
```

## CHAPTER-7 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. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### **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 satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### **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.

### **7.1.5 White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.



### **7.1.6 Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

### **7.2 Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

#### **Test objectives**

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

#### **Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

### **7.3 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.

**Test Results :**All the test cases mentioned above passed successfully. No defects encountered.

#### 7.4 USER REQUIREMENTS : home

Use case ID	Security using Elliptic Curve Cryptography in Cloud
Use case Name	Home button
Description	Display home page of application
Primary actor	User
Precondition	User must open application
Post condition	Display the Home Page of an application
Frequency of Use case	Many times
Alternative use case	N/A
Use case Diagrams	
Attachments	N/A

#### 7.5 TEST CASES:

Following table depicts the test cases and the results respectively,

Test No.	Input Description	Expected Behavior	Observed Behavior	Status
1	Feeding input details with all the required parameters	Validate basic variable and null reference values.	Validated passed input parameters as expected.	Pass
2	Faces detected with low pixel	Verification for face match	Cantperform encodings	fail

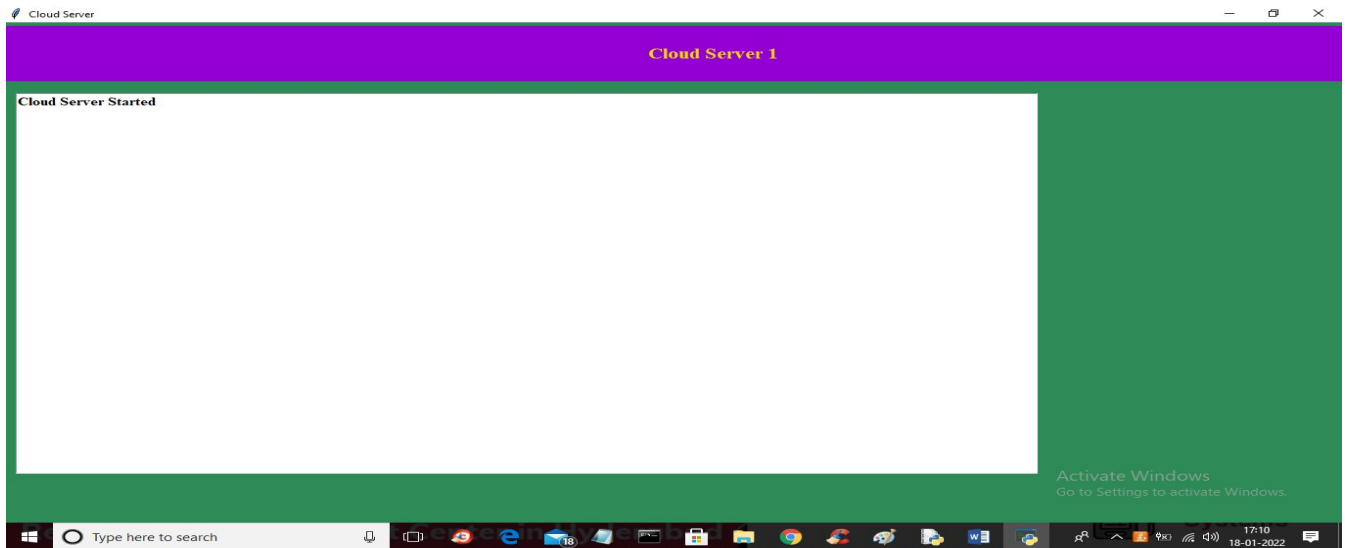
3	Local Image Paths	Need to verify whether path is reachable or not.	Path Reachable	Pass
4	Image Verification	Detect faces from the image	Detected faces from the given image	Pass
5	Notification Message	Notify the result	Notification message has sent	Pass
6	Input image with multiple faces	Detect the faces	Can't Detect more than one face	Fail
7	Invalid image path	Invalid path	Face not detected	Fail
8	Image with single face	Detects the face from image	Detects the face from image	Pass
9	Cropped image	Generate face id	Generate face id	Pass
10	Cropped image	Create person id	Create person id	Pass

## **CHAPTER-8**

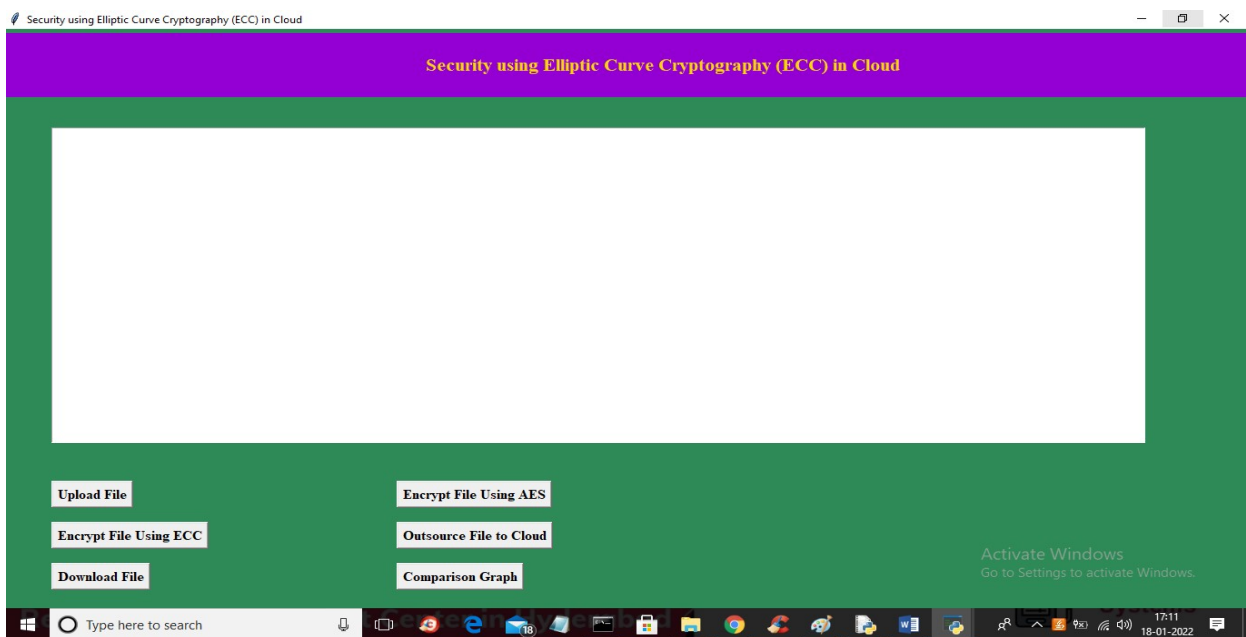
### **SCREENSHOTS**

To run project first double click on 'run.bat' file from 'CloudServer' folder to start cloud application and to get below screen

## Enhanced Security using Elliptic Curve Cryptography in Curve

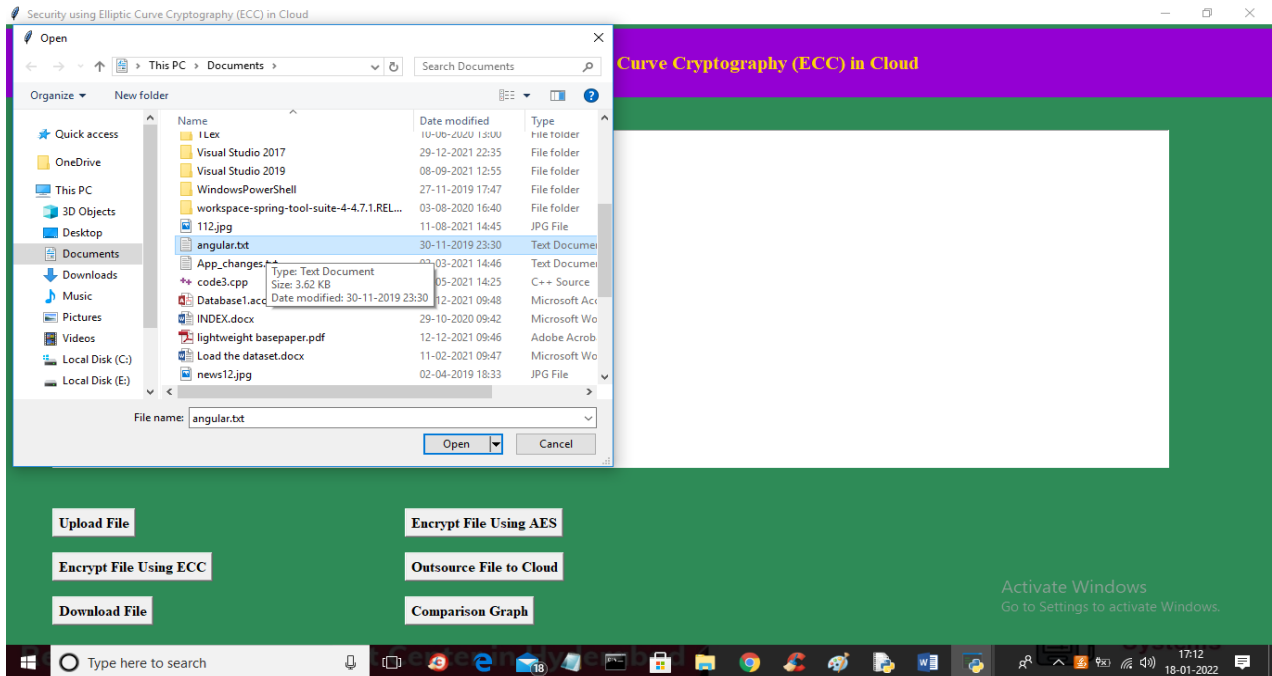


In above screen cloud server started and now double click on 'run.bat' file from 'CloudUser' folder to start cloud user application and to get below screen

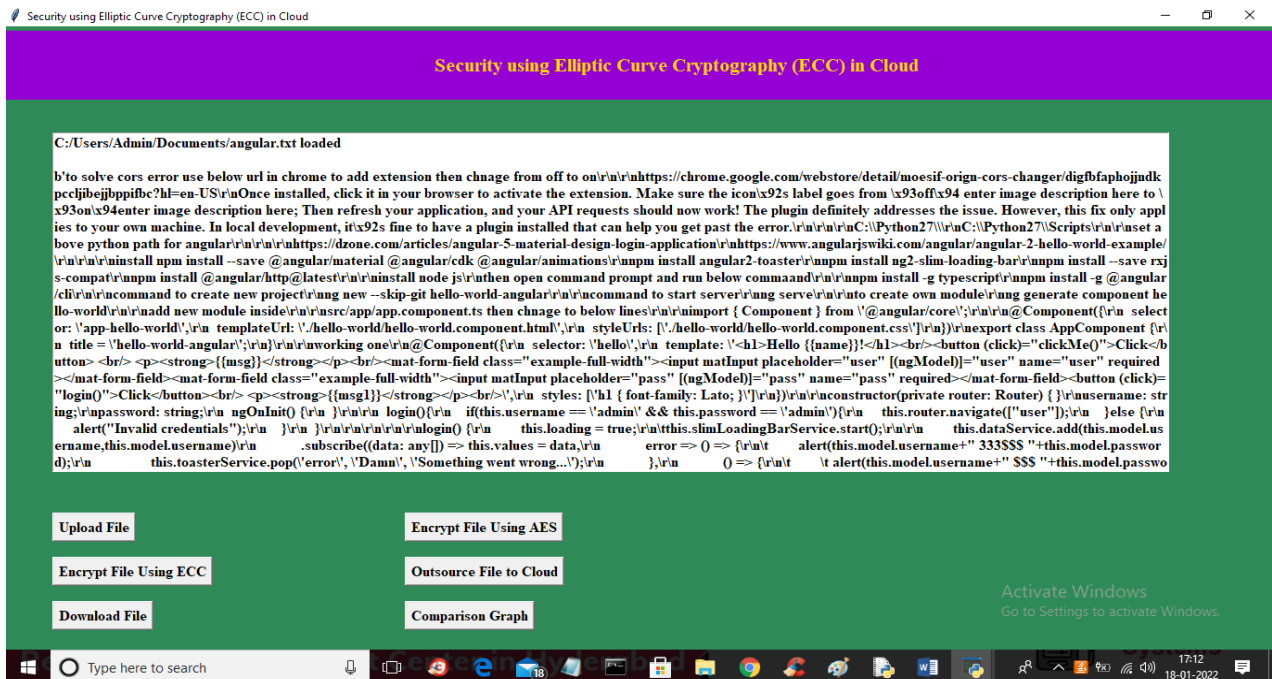


In above screen click on 'Upload File' button to upload any file to application like below screen

## Enhanced Security using Elliptic Curve Cryptography in Curve

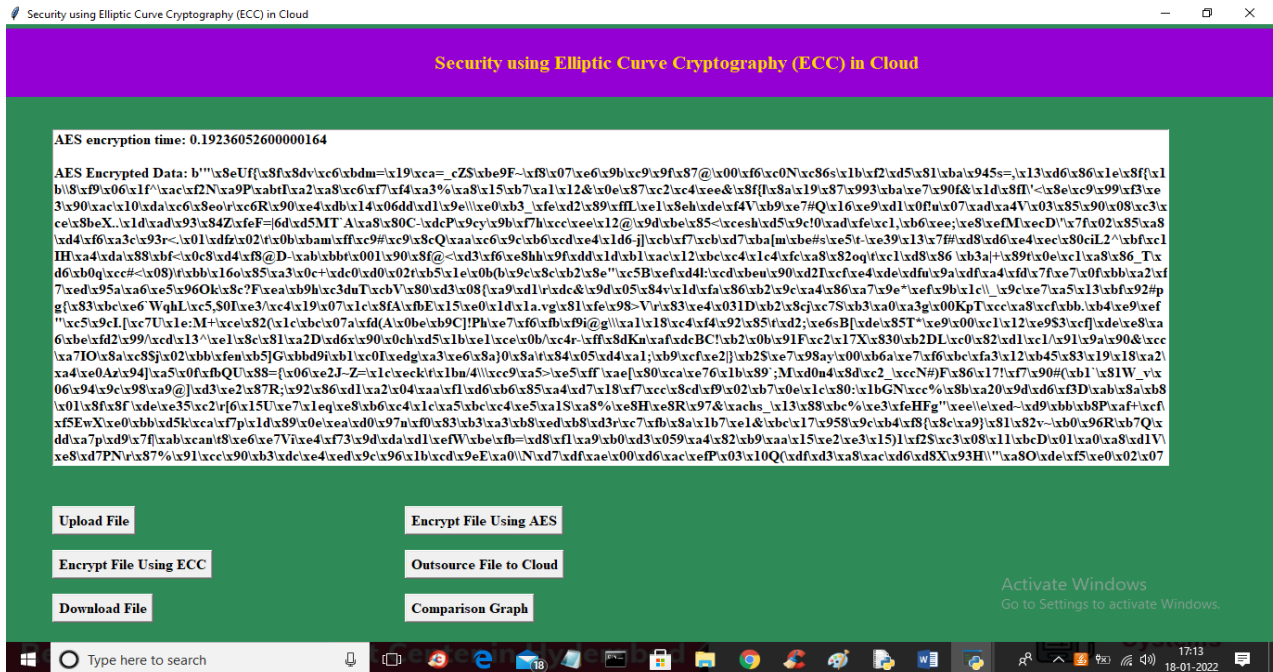


In above screen selecting and uploading 'angular.txt' file and then click on 'Open' button to load file and to get below screen

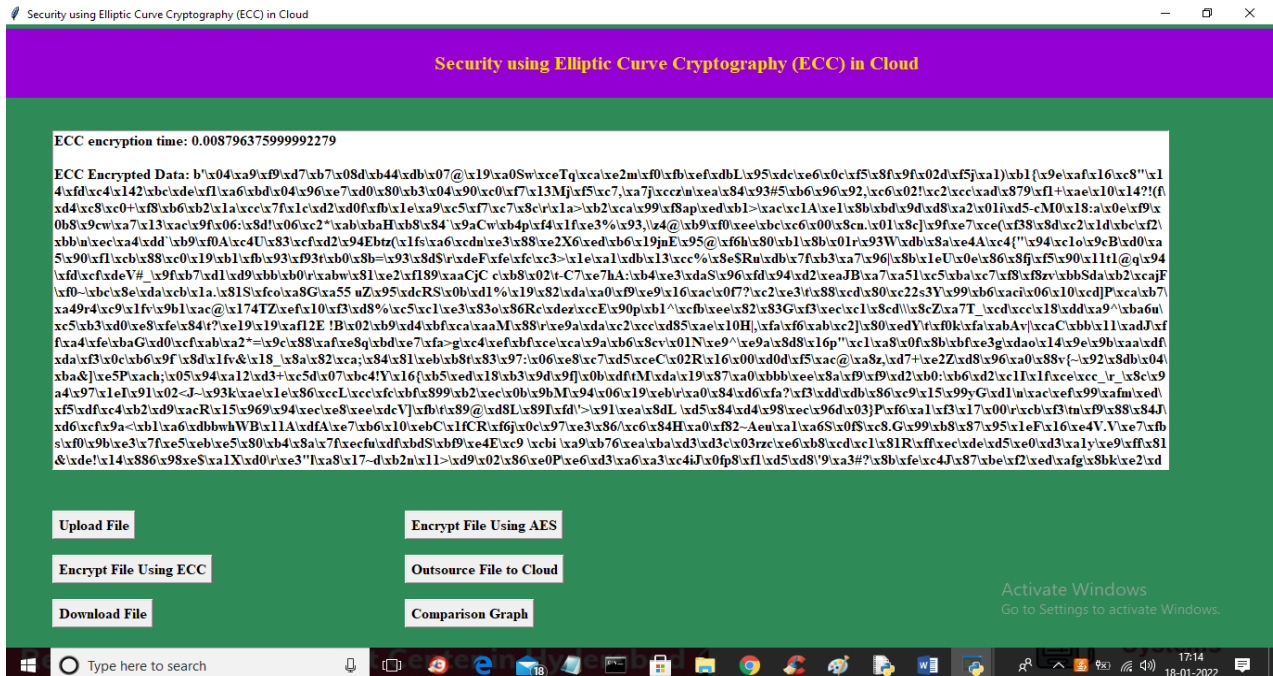


In above screen file is loaded and now click on 'Encrypt File Using AES' algorithm button to encrypt file and to get below screen

## Enhanced Security using Elliptic Curve Cryptography in Curve

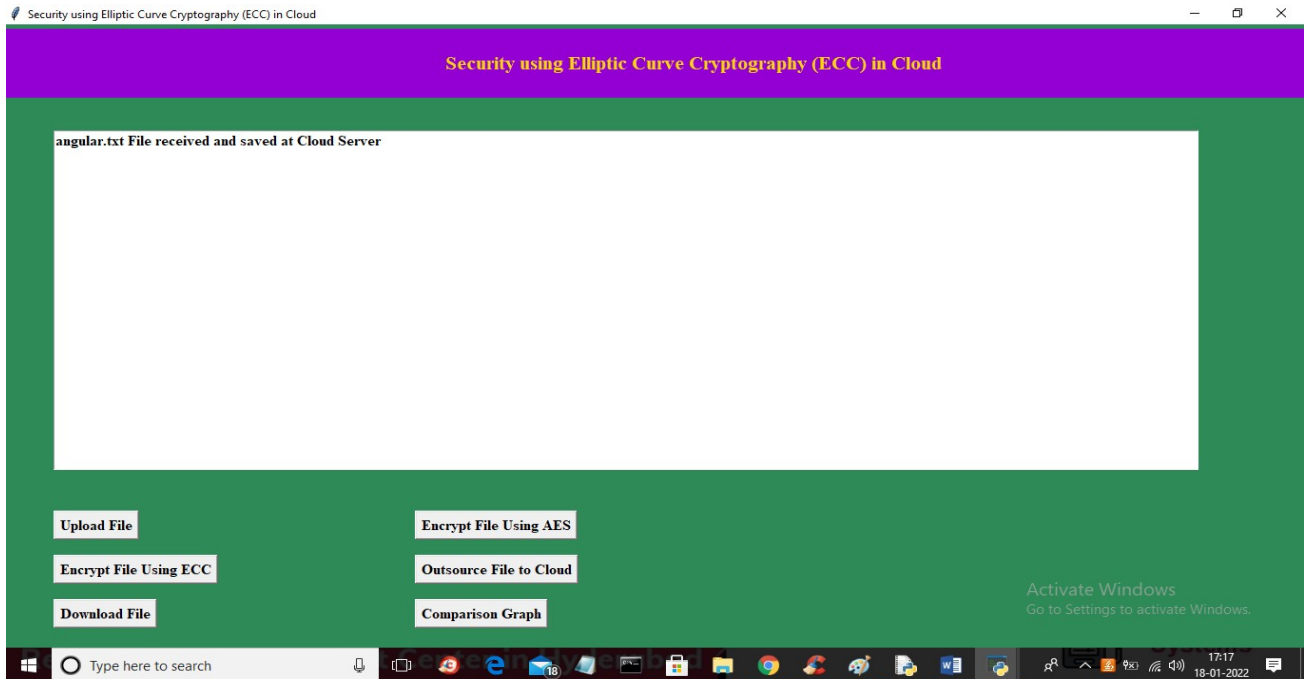


In above screen we can see plain data is encrypted and in first line AES encryption time is 0.192 milli seconds. Now click on ‘Encrypt File using ECC’ button to encrypt same file using ECC and calculate time

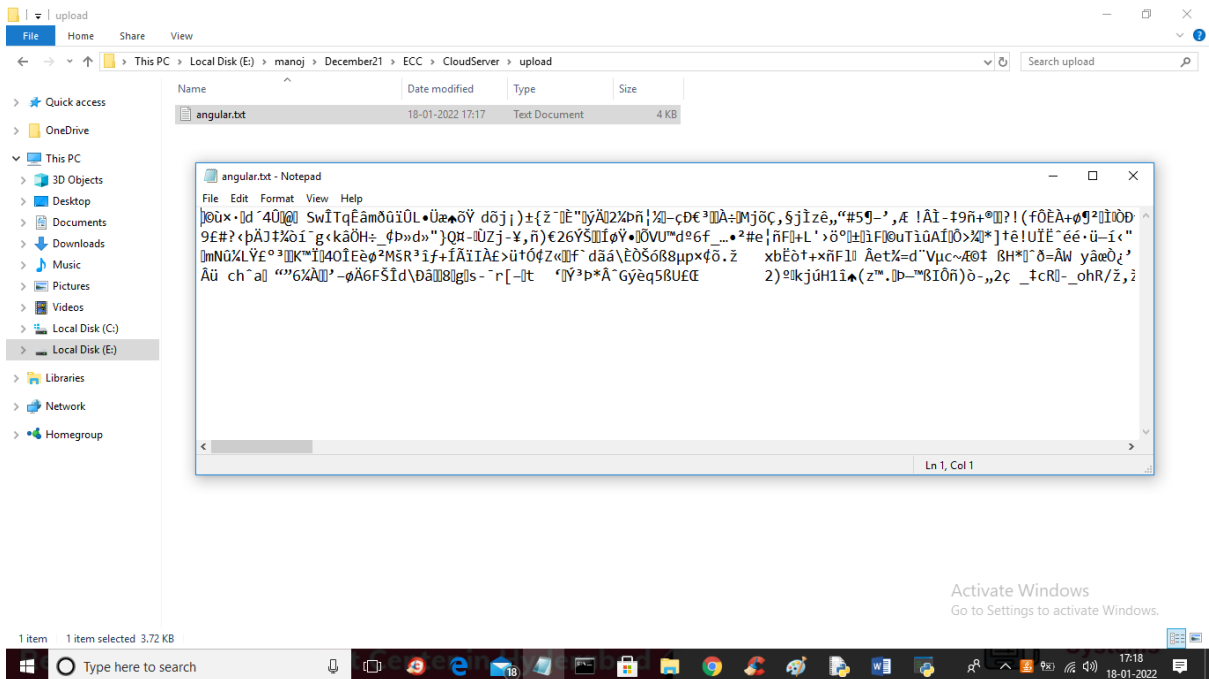


In above screen same file data encrypted using ECC and its took only 0.008 milli second to encrypt same data. Now click on ‘Outsource File to Cloud’ button to send file to cloud

## Enhanced Security using Elliptic Curve Cryptography in Curve



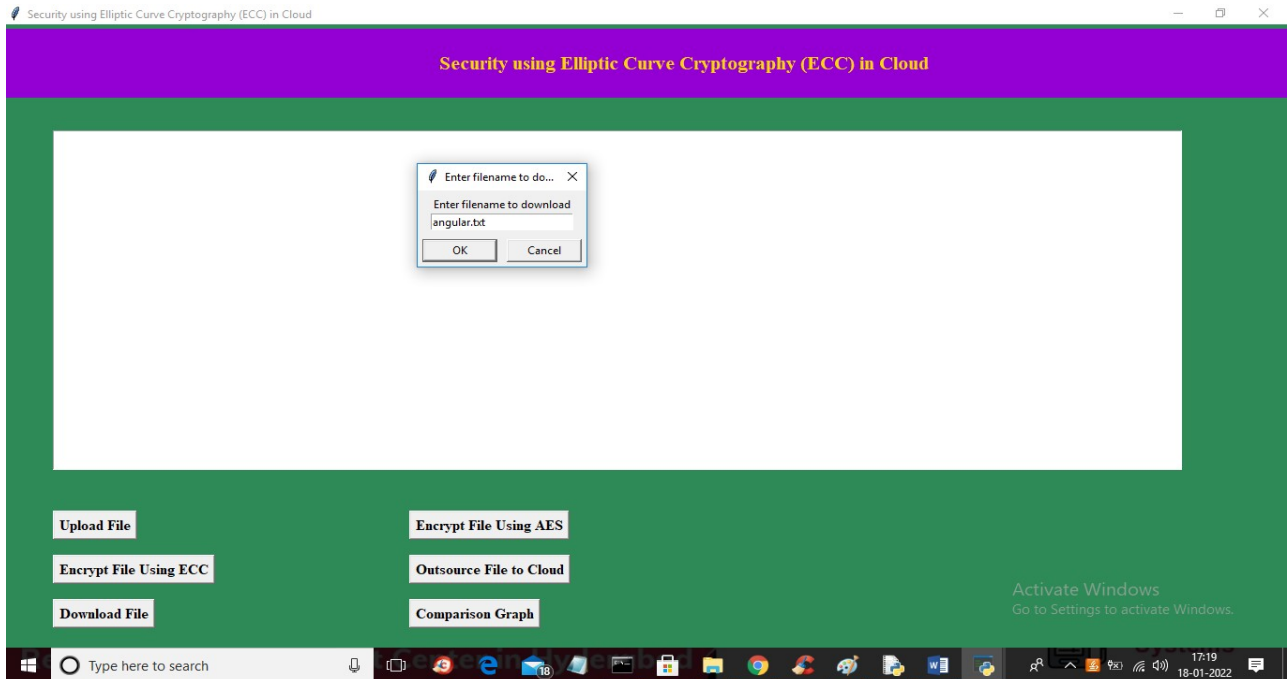
In above screen we can see file sent to cloud server and in ‘CloudServer/upload’ folder we can see same file stored in encrypted format



In above screen we can see that files saved in encrypted mode in CloudServer folder. Now click on ‘Download File’ button to download file



## Enhanced Security using Elliptic Curve Cryptography in Curve



Similarly you can upload and download any number of files from cloud and compare execution time between AES and ECC.

Below screen showing code to generate AES keys and encrypt and decrypt function

```
CloudUser.py - E:\manoj\December21\ECC\CloudUser\CloudUser.py (3.7.0)
File Edit Format Run Options Window Help
from tkinter import *
import matplotlib.pyplot as plt
import pyaes, pbkdf2, binascii, os, secrets
import base64
import timeit
from ecies.utils import generate_eth_key, generate_key
from ecies import encrypt, decrypt
import pickle
import socket
import json
from tkinter import messagebox
import numpy as np

main = tkinter.Tk()
main.title("Security using Elliptic Curve Cryptography (ECC) in Cloud") #designing main screen
main.geometry("1300x1200")

global filename
execution_time = []
global data
global secret_key, private_key, public_key

def getAESKey(): #generating key with PBKDF2 for AES
    password = "s3cr3t*c0d3"
    passwordSalt = "76895"
    key = pbkdf2.PBKDF2(password, passwordSalt).read(32)
    return key

def encryptAES(plaintext): #AES data encryption
    aes = pyaes.AESModeOfOperationCTR(getAESKey(), pyaes.Counter(31129547035000047302952433967654195398124239844566322884172163637846056248223))
    ciphertext = aes.encrypt(plaintext)
    return ciphertext

def decryptAES(self,enc): #AES data decryption
    aes = pyaes.AESModeOfOperationCTR(getAESKey(), pyaes.Counter(31129547035000047302952433967654195398124239844566322884172163637846056248223))
    decrypted = aes.decrypt(enc)
    return decrypted

def upload(): #function to upload tweeter profile
    global filename
    global data
    filename = filedialog.askopenfilename(initialdir="Dataset")
```

In above screen read red color comments to know about AES and in below screen showing code from ECC

## **CHAPTER-9**

### **CONCLUSION AND FUTURE ENHANCEMENT**

Cloud Computing provides a platform with an enhanced and efficient way to store data in the cloud. The functioning of Cloud Computing is significantly distressed by issues such as that of data security, integrity, theft, loss and presence of infected applications. These issues are the major disadvantages to the consumer to move their data to the cloud. This paper proposed a model using Elliptic Curve Cryptography to enable more efficient data security in the cloud computing. Here, Security is based on the difficulty of computing discrete logarithm in a finite field. AES and ECC are forms of public key cryptography, in which one decryption key, known as the private key, is kept secret, while another, known as a public key, is freely distributed. Public key cryptography is computationally more expensive than private key encryption, which employs a single, shared encryption key. By using the proposed algorithm, Cloud computing can achieve high level of security more than the security attain by the IT enterprises their own hardware and software.

## **CHAPTER-10**

### **REFERENCES**

- [1] AbhudayTripathi, and ParulYadav, Enhancing Security of Cloud Computing using Elliptic Curve Cryptography, International Journal of Computer Applications, 57(1), 2012, 0975-8887.
- [2] Nilesh N. Kumbhar, Virendrasingh V. Chaudhari, and MohitA.Badhe, The Comprehensive Approach for Data Security in Cloud Computing: A Survey, International Journal of Computer Applications, 39(18), 2012, 0975-8887.
- [3] N. Koblitz, Elliptic Curve Cryptosystems, Mathematics of Computation, 1987.
- [4] Yubo Tan, and Xinlei Wang, Research of Cloud Computing Data Security Technology, 978-1-4577- 1415-3/12,IEEE 2012.
- [5] YashpalsinhJadeja, and KiritModi, Cloud Computing - Concepts, Architecture and Challenges, International Conference on Computing, Electronics and Electrical Technologies,4(12), 2012, 978-1-4673-0210
- [6] Dr. Chander Kant, and Yogesh Sharma, Enhanced Security Architecture for Cloud Data Security, International Journal of Advanced Research in Computer Science and Software Engineering, 3(5), 2013.
- [7] Wayne Jansen, and Timothy Grance, Guidelines on Security and Privacy in Public Cloud Computing, National Institute of Standards and Technology, U.S. Department of Commerce, 800-144.
- [8] VeerrajuGampala, Data Security in Cloud Computing With Elliptic Curve Cryptography, International Journal of Soft Computing and Engineering (IJSCE), 2, 2012.

- [9] Dai Yuefa, Wu Bo, GuYaqiang, Zhang Quan, and Tang Chaojing, Data Security Model for Cloud Computing, Proc. International Workshop on Information Security and Application. Qingdao, China, 2009, 978-952-5726-06-0.
- [10] IkshwansuNautiyal, and Madhu Sharma, Encryption Using Elliptic Curve Cryptography Using Java as Implementation Tool, International Journal of Advanced Research in Computer Science and Software Engineering, 4(1), 2014. [11] Vidyanand K\ Ukey, and Nitin Mishra, Dataset Segmentation for Cloud Computing and Securing Data Using ECC, International Journal of Computer Science and Information Technologies, 5(3), 2014, 4210-4213.
- [11] R. BalaChandar, and M. S. Kavitha, A Proficient Model For High End Security in Cloud Computing, ICTACT Journal of Soft Computing, 04( 02), 2014.
- [12] Nina Pearl Doe, and Sumaila Alfa, An Efficient Method to Prevent Information Leakage in Cloud, IOSR Journal of Computer Engineering (IOSR-JCE) 16(3), 2014, 2278-8727.
- [13] NehaTirthani, and Ganesan R, Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography, International Association for Cryptologic Research Cryptology ePrint 49, 2014



