# CHAPTER 1

## Introduction

Data in the cloud is secured by encrypting them to a new form by cryptosystem methods. There are two cryptosystem categories: symmetric and asymmetric. The symmetric cryptosystem uses the same key to perform the process of encryption and decryption of data. In an asymmetric cryptosystem or public key cryptosystem, the public key is used for encryption and the data can only be read by the specific recipient who has the paired key called private key. IDEA and DES are the symmetric cryptosystem example and RSA and Paillier are the asymmetric cryptosystem example.

To perform operations on the encrypted data(ciphertext), decryption of the ciphertext is required. But the data is no longer secure after it is decrypted.

The new method that permits direct computation on encrypted data without decryption was proposed, called “privacy homomorphism”. With homomorphic encryption, the operations on the encrypted data such as additions and multiplications can be performed by using the public key algorithm.

The aim of the project is to implement the E-voting system using Paillier and Elgamal Homomorphic encryption [PHE] in cloud computing . By utilizing the electronic voting system (e-voting system), the voting process can save cost, faster, and more accurate in the calculation process and also more practical and safe. The E-voting systems are most popular and successful today. Each voter can vote for candidates remotely through the Internet. Voters can easily open websites, software, or applications on their computers or smartphones to vote anytime, anywhere. However, their voting system can be attacked if it does not have any algorithm or protocol to protect it. Attackers can catch packets that voters transfer on the Internet easily; they change information and send to the voting server. In addition, they can interrupt votes which voters send over the Internet.

This project uses the homomorphic properties of Paillier and Elgamal Encryption to protect a vote sent over the Internet. The project focuses on implementing E-Voting system which uses the effectiveness of the Paillier and Elgamal algorithm to perform encryption and decryption process. With the homomorphic property, the system can calculate the sum of votes without revealing which vote is voting for which candidate to the system.

## 1.1 Motivation

Homomorphic encryption is currently one of the most active research topics in cryptography which supports operations on encrypted data. In other encryption methods the data has to be decrypted to perform operation but no in this case.

Coming to the E-voting, it is the most convenient method to vote. It is excellent on equality, building a trust

in electoral organization, adding reliability to election results, and increasing the overall

efficiency of the polling process . However, to build an E-voting system that can work perfectly over the Internet is a big challenge. Two major challenges that can be considered are security and supporting a large number of voters. Most E-voting solutions cannot work with large number of voters. In addition, voting under an electronic voting system occurs automatically without any human supervision. Certainly, voters would like to vote by paper votes at a post office or polling station rather than through an E-voting system because they do not trust the E-voting as their ballots are transferred over the Internet .

To address some of these challenges, in this project, we have developed an E-voting system that can allow a large number of voters. Specifically, Paillier algorithm is used to support a large number of voters and secure a vote when it is counted.

## 1.2 Scope

As the growth of cloud services are increasing, on the other hand the security of cloud computing services is a contentious issue that may be delaying its adoption. Cloud computing offers many benefits, but it is also vulnerable to threats. There are many challenges and risk in cloud computing that increases the threat of data being compromised. Security is becoming a major concern in order to establish trust in cloud computing technology.

There are many security methods in cloud but the efficient methods are encryption methods. Among all encryption methods the homomorphic encryption performance is better than others because of the advantage it has over other methods.

The data is encrypted in to the cloud and we can use whenever we want for the computation without decrypting it.

**1.3 Objectives**

* Homomorphic encyption method performs operation on encrypted data itself without decrypting it which makes the data more secure.
* Design and development of evoting system using Paillier and Elgamal homomorphic encryption schemes.
* Providing a user friendly interface for e-voting system and increasing its security by implementing it in cloud.

## 1.4 Proposed Model

The e-voting application is designed to be the simulation of e-voting system in general. This system has a graphical user interface (GUI). Whole system design mentioned above will be implemented in this application using Java programming language and uses MySQL for its database.

Registered voters must be validated to access the voting menu. Voters who have been validated need to login first and they can choose one of the candidates in the list to cast their vote. After the voter chooses a candidate, the voted data will be encrypted using paillier encryption function and stored in the cloud. The paillier key is generated using prime numbers p and q such that gcd(pq,(p-1)(q-1))=1 , n which is equal to product of p and q , Carmichael function[λ(n)], generator g and modular multiplicative inverse µ. The voted data(m) is encrypted to form ciphertext using r(randomly generated number) and public key(n,g) which is given by ciphertext= gm\*rn mod n2

When the data successfully encrypted, the encrypted data or ciphertext will be stored in database. Each voter will have ciphertext that is different from each other even though they choose the same candidate. After tally server received all encrypted messages from voters, the server will multiply all encrypted data. As explained before, Paillier algorithm can obtain E(m1+m2) with homomorphic property by multiplying m1 and m2 without decrypt it before. To obtain the final result after the election ends, the administrator can perform Tally where homomorphic property applies. Then the tally server decrypts Tally to get tally message M using private key(λ, µ) .

The result of calculated ciphertexts(message M) will be same as the sum of the voted data to be encrypted. The admin will not know the choice of the voters. As a result of this decryption function, one gets simple tallying of all votes. So Division remainder method is used to determine how many votes cast for each candidate with number of voters as base(the base should be greater than number of voters).The final result of the election is sent to the voter’s registered mobile number. Here the admin is able to create a poll, manage poll(add/remove voters and candidates),end the election and get result.

## 1.5 Organization of Report

Chapter 1 includes the introduction of the project and the motivation to choose this field along with scope, objectives of the project.

Chapter 2 includes the literature survey on the topic from various research papers and journals.

Chapter 3 includes the System analysis and Design which includes software and hardware requirements of the project.

# CHAPTER 2

## Literature Survey

Cloud computing is considered as next generation in computation. Cloud computing is termed as next natural step in the evolution of on-demand services, resources and products. The Cloud is known to be a metaphor for the Internet, Cloud computing is a model that enables on demand network access to a shared pool of configurable computing resources that can be rapidly stipulated and released with minimal management effort and service provider interaction. Though there is no straight forward way to explicate what actually cloud computing is, but it can be defined in general as: “Cloud computing is type of computing environment, where business owners can outsource their computing needs like required application software services to a third party and when they need to use the computing power or when employees require to use the application resources such as, emails, database etc., they can access the resources via internet”. Cloud computing is a very cost effective concept because in cloud architecture, we neither have to install nor to maintain servers. Cloud is viewed as virtualized pool of computing resources. It can deal with heterogeneous workloads, comprising of many back end operations and user oriented interactive applications.

## 2.1 SECURITY ISSUES IN CLOUD COMPUTING

Cloud computing is associated with numerous security issues but these challenges fall into two broad categories: Security issues faced by cloud service providers and security problems faced by the clients. Generally, the service provider must ensure the security of infrastructure and that their clients’ data and applications are well protected on the other hand the customers must ensure that the cloud provider has taken the correct security measures to secure their information. Data security is one of the major concerns for both cloud customers and providers which in turn resulted in the development of a wide range of techniques to resolve the cloud data security issues.

* **Data protection** Cloud providers have system placedto prevent data leakage or access by third parties. Proper separation of duties should be ensured for timely auditing and monitoring. Every enterprise of Identity management will have its own identity management system to control access to informationand computing resources.
* **Data Locality** Data locality in cloud is described as the ability of cloud providers to control the data location in order to satisfy the customer’s preferences on the data storage locations and its boundaries.
* **Data Integrity** This is another cloud security issue which means that the data is reliable during its life cycle. Data may also be replicated in multiple situations across cloud’s data centers any kind of changes made in data will be propagated throughout all replications.
* **Data Segregation** this is another important security requirement of cloud as the data from different customers reside at the same location (multitenancy).
* **Data Access** this is one of the important cloud securities metric. Each client has its own access policy which needs to be applied on its own data.
* **Data Confidentiality** We must ensure that user’s private or confidential information should never be accessed by anyone in the cloud computing system, including application, platform, CPU and physical memory. It is quite clear that user’s confidential data is disclosed to service provider under following situation only. The first situation where user’s information may be disclosed when service provider knows where the users private information resides in the cloud systems. The second situation where user’ s information may be disclosed when service provider has the authority to access and gather users private information in the cloud systems. The third situation where user’s information may be disclosed when service provider can figure out the meaning of user’s information in the cloud systems.

* **Data Availability** Some of the threats to data availability are flooding attacks causes deny of service and Direct /Indirect attack.
* **Data Breaches** Since the user’s data are uploaded to the cloud, any type of breach in the cloud environment potentially threatens all the consumers. This will make the cloud a high value target for outsider attackers also. Additionally the insider attacks still remain a high risk threat from employees who are inside the cloud provider which potentially have access to customer’s information.

From the above it is clear that Cloud services and applications require all standard security functions including data confidentiality, integrity, privacy, robustness and access control. Hence for providing security to the cloud is a challenging task. Several cryptographic methods have been proposed by the authors to secure the data stored in cloud storage systems which also helps in secure sharing of data in the cloud environment.

Here discuss some of the significant proposed encryption techniques such as proxy re-encryption (PRE) scheme, Type based PRE, Key-private PRE, Identity based PRE, Attribute based PRE and Threshold PRE which are as follows.

* **Proxy Re-encryption (PRE) Scheme** Proxy re-encryption is a cryptographic primitive which helps in translating cipher texts from an encryption key to another encryption key. PRE forwards the encrypted messages without exposing the clear texts to the potential users.
* **Type Based PRE Scheme** This encryption scheme ensures data confidentiality and fine gain access control. TB-PRE permits the delegator to implement fine policies with one key pair without any additional trust on the proxy. Under this scheme the delegator can categorize his cipher texts into different subsets.
* **Key Private PRE Scheme** In a key private PRE, the proxy and a set of colluding users can never derive the recipient of a message from the cipher text and the set of public keys. This scheme deals in formulating the notion of key privacy for proxy re-encryption schemes, where even the proxy which performs the all translations cannot distinguish the identities of the participating parties.
* **Attribute Based PRE Scheme** In this encryption scheme attribute based proxy re-encryption scheme includes a semi trusted proxy with some additional information which can transform a cipher text under a set of attributes into a new cipher text under a set of attributes into a new cipher text under another set of attributes on the same message.
* **Time Based PRE Scheme** a basic approach of Time based PRE scheme allows each user’s access right to expire automatically after a specific period of time. The main theme is to assimilate the concept of time into the combination of Proxy re-encryption (PRE) and Attribute based encryption (ABE). In this PRE scheme, every data is coupled with an attribute based access time and an access structure
* **Threshold based PRE Scheme** A fundamental approach of threshold PRE scheme is secure computation. The scheme arbitrary performs computations on encrypted data without decrypting it. Threshold Proxy Re-encryption technique has multiplicative homomorphic concept. A multiplicative homomorphic encryption scheme is a property deals with encoding operation over encrypted messages and forwarding operations over encrypted and encoded messages.

In processing state the data is decrypted and used and again encrypted and stored in cloud which lacks security of data in processing state. So to overcome this problem Homomorphic encryption is a feasible solution because the encrypted data only used in computation state without decrypting the ciphertext.

## 2.2 HOMOMORPHIC ENCRYPTION

Homomorphic encryption is a form of encryption that allows computations to be carried out on cipher text , thus generating an encrypted result which, when decrypted, matches the result of operations performed on the plaintext. The main purpose is to allow computation on encrypted data.

An encryption scheme is called homomorphic over an operation if it supports the following equation:

E(m1)\*E(m2) =E(m1\*m2); for all m1and m2 belongs to M

where E is the encryption algorithm and M is the set of all possible messages.

In order to create an encryption scheme allowing the homomorphic evaluation of

arbitrary function, it is sufficient to allow only addition and multiplication operations because addition and multiplication are functionally complete sets over finite sets .While an HE scheme can use the same key for both encryption and decryption (symmetric), it can also be designed to use the different keys to encrypt and decrypt (asymmetric).

## 2.3 WHY HOMOMORPHIC ENCRYPTION

Traditional standard encryption methods provide security to data in storage state and transmission state. But in processing state, performing operations on data require decryption of data. At this state data is available to cloud provider. Hence traditional encryption methods are not sufficient to secure data completely. Homomorphic encryption methods and their applications in cloud computing to secure data in processing state. Homomorphic encryption allows user to operate encrypted data directly without decryption. And also it solves Confidentiality problems , Ability to compute over ciphertext instead of plaintext .

## 2.4 ADVANTAGES OF HOMOMORPHIC ENCRYPTION

Other than the advantage of one could use information without knowing the content of that information and Privacy guaranteed, it also provides

* Cloud security.
* Working with information stored in databases.
* Queries to search engines.
* Spam Filtering.

**2.5 DISADVANTAGES OF HOMOMORPHIC ENCRYPTION**

Performance is often a disadvantage. Ciphertexts in the ciphers you mention are much larger than the plaintexts, so communication requirements typically go up. The computations on these large ciphertexts are typically slower than if you just performed the computation on the plaintext itself. Because of this, in the outsourcing computation model, we typically see a requirement that encrypting inputs and decrypting outputs should be faster than performing the computation itself. In the case of multiple parties with individual inputs this seems to be less of a concern as privacy, not efficiency is the concern.

The homomorphic property also implies malleability. This means, if it has some ciphertext, then it can create a different ciphertext with a related plaintext, and this property can be unwanted in this scheme(e.g. in an auction where you just encrypt your actual bid; then the attacker could just use your bid +1 or exchange your name with his, etc.) Malleability doesn't specify what kind of relation is implied by changing the ciphertext, while the homomorphic property usually refers to an algebraic operation.

**2.6 WORK DONE SO FAR IN HOMOMORPHIC ENCRYPTION**

HE scheme is primarily characterized by four operations:

* Key Generation ,
* Encryption,
* Evaluation.
* Decryption.

Key Generation is the operation, which generates a secret and public key pair for the asymmetric version of HE or a single key for the symmetric version. Actually, Key Generation, Encryption and Decryption are not different from their classical tasks in conventional encryption schemes.

However, Evaluation is an HE-specific operation, which takes ciphertexts as input and outputs a ciphertext corresponding to a functioned plaintext.

Using homomorphic encryption there are different schemes:

* Goldwasser–Micali Encryption Scheme
* ElGamal Encryption Scheme
* Paillier Encryption Scheme (Pascal Paillier
* invented it in 1999)
* Boneh–Goh–Nissim Encryption Scheme

**Gentry's cryptosystem**

This described the first plausible construction for a fully homomorphic encryption scheme. Gentry's scheme supports both addition and multiplication operations on ciphertexts, from which it is possible to construct circuits for performing arbitrary computation.

Regarding performance, ciphertexts in Gentry's scheme remain compact insofar as their lengths do not depend at all on the complexity of the function that is evaluated over the encrypted data, but the scheme is impractical, and its ciphertext size and computation time increase sharply as one increases the security level.

**The 2nd generation of Homomorphic cryptosystems**

Several new techniques that were developed starting in 2011-2012 by Zvika Brakerski and others, led to the development of much more efficient somewhat and fully homomorphic cryptosystems. These include:

* The Brakerski-Gentry-Vaikuntanathan cryptosystem (BGV), building on techniques of Brakerski-Vaikuntanathan[9].
* Brakerski's *scale-invariant* cryptosystem.[10]
* The NTRU-based cryptosystem due to Lopez-Alt, Tromer, and Vaikuntanathan (LTV).[11]
* The Gentry-Sahai-Waters cryptosystem (GSW).[12]

All the second-generation cryptosystems still follow the basic blueprint of Gentry's original construction, namely they first construct a somewhat-homomorphic cryptosystem that handles noisy ciphertexts, and then convert it to a fully homomorphic cryptosystem using bootstrapping.

Among the 4 homomorphic encryption schemes, Paillier and Elgamal encryption is the most well-known cryptography schemes.

To implement both Paillier and Elgamal encryption scheme and its homomorphic property, the secure E-Voting system platform is choosen. “E-voting” is a term that is described in different information and communication technologies (ICT) platforms: Internet systems, polling booth machines, and telephone voting system [7]. Each platform has both negative and positive features. However, the Internet voting systems are most popular and successful today. Each voter can vote for candidates remotely through the Internet. Voters can easily open websites, software, or applications on their computers or smartphones to vote anytime, anywhere. However, their voting system can be attacked if it does not have any algorithm or protocol to protect it. Attackers can catch packets that voters transfer on the Internet easily; they change information and send to the voting server. In addition, they can interrupt votes which voters send over the Internet.

This project uses the homomorphic properties of Paillier and Elgamal Encryption to protect a vote sent over the Internet.

The main goal of this project is to design a secure voting system using the Internet platform to communicate between the voting system and voters. Homomorphic encryption is used to make sure that votes holds its confidentiality by encrypting and calculating all votes without decrypting and also achieves requirements: eligibility, privacy, accuracy, fairness, receipt-freeness, coercion resistance, mobility, simplicity, individual variability, scalability and availability.

Developing an e-Voting system requires the collaboration of many participants with different background.

This method has multiple advantages in comparison to traditional methods such as paper ballot. Voters do not need to present

at ballot box. It provides ease of accessibility and comfort of use. Another advantage of Internet voting is wide usability that can be applied in all forms of municipal, public, state elections, and referendums. Using an electronic voting system, voters also can

vote rapidly and exactly because they don’t need to go to Ballot station and vote through their computers.

However, to implement a secure voting system is difficult. Here are some requirements which a secure voting system has to adopt.

***Authentication:*** Only authorized voters should be able to vote. Voters can request to change their information if necessary. A voter’s information can be collected from the

birth certificate or other similar document.

***Uniqueness:*** No voter should be able to vote more than once. This requirement is necessary and same as the traditional voting. This feature prevents coercion or buying votes.

***Accuracy:*** Voting systems should record the votes correctly. After the vote is recorded, the voter should be able to check if his vote was recorded or not. If he tries to re-vote or change the vote, the system should prevent that.

***Integrity:*** A vote is just for one time decision and cannot be changed. No one should be able to determine how the voter voted, so no one can change the vote.

***Verifiability:*** Verifying that the votes are correctly counted in the final tally should be possible. This feature is mandatory. It has to be audited and tested before the system is used.

***Auditability:*** Reliable and demonstrably authentic election records should be generated.

***Reliability:*** Systems should work robustly to prevent electoral frauds or attacks from outside the system. The E-voting system should be very reliable. The result of an election must be correct and shows up to voters after the election ends.

***Voter Confirmation:*** The voting system should send an email to the voter to confirm that his or her vote has been received by the system correctly. At the end, the result of the voting also can be sent to the voters so that they will know which candidate is a winner.

**2.7 Cryptography**

Cryptographic techniques are used for secure storage, communication, and handling of data and information. In this report, we primarily focus on cryptographic voting schemes that are relevant to the project. Cryptographic techniques have been used to voting systems

since 1981. These techniques provide a level of assurance of accuracy and secrecy to a voting system. Largely cryptosystems can be divided to two systems: symmetric key cryptosystems and asymmetric key cryptosystem. A symmetric key system uses the same secret key to encrypt and decrypt the message. On the other hand, asymmetric key cryptosystem uses two different keys to encrypt and decrypt the

message. These keys, one of them called the public key and the other the private key, are different. Both schemes have different strengths in terms of security. For instance, asymmetric cryptosystems provide privacy and reliability but it does not work with authentication. However, an asymmetric cryptosystem can provide authentication but it is more complex than a symmetric system.

A number of cryptographic schemes are used to design a voting system. Most of these schemes can count the votes securely. Mostly frequently used algorithms are RSA (Rivest, Shamir, and Adleman) encryption, ElGamal encryption, and Paillier encryption.

Since the project is focused on implementing e voting system using paillier encryption, the following are the methods for key generation, encryption and decryption.

**2.8 Paillier Cryptosystem**

**Step 1: Paillier Key Generation**

1. Select two large prime numbers *p* and *q* randomly and independently of each other such that gcd (*pq*, (𝑝 − 1)(𝑞 − 1)) = 1. The gcd is the general common divisor of two or more non-zero integers which is the largest positive integer that divides the numbers without a remainder.

2. Compute *n* = *pq* and *λ* = *lcm*(*p* − 1,*q* − 1) with *λ(n)* being the Carmichael function. The *lcm* is the least common multiple of two or more non-zero integers which is the smallest integer that is divisible by every member of a set of numbers without a remainder.

3. Choose generator *g* where *g* ℤ\*n2

There are two ways to select the *g*.

Method 1: Randomly select *g* from a set ℤ\*n2

where gcd( (gλ mod n2-1)/n,n) = 1.

Method 2: Select *a* and *b* randomly from a set ℤ\*n2 then calculate

*g* = (a×n +1) × 𝑏n 𝑚𝑜𝑑 𝑛2

4. Calculate the follow modular multiplicative inverse

𝜇 =( 𝐿( 𝑔λ 𝑚𝑜𝑑 𝑛2 ))-1

Where L(u)=(u-1)/u

So, the pair of keys generated: the public key is (*n*,*g*) and the private key is (𝜆, 𝜇).

**Step 2: Encryption process**

1. The message *m* is a message need to be encrypted where *m* 𝜖 ℤ\*n2
2. Choose a random number *r* with *r* 𝜖 ℤ\*n2

3. Compute ciphertext *c* = 𝑔m× 𝑟n 𝑚𝑜𝑑 𝑛2 (use the public key (*n*, *g*))

**Step 3: Decryption process**

Ciphertext *c* will be decrypted to get message *m* as follows by using the private

key (𝜆, 𝜇):

M = L (𝑐λ 𝑚𝑜𝑑 𝑛2) × 𝜇 𝑚𝑜𝑑 𝑛

**2.9 Homomorphic property of Paillier Cryptosystem**

The encryption algorithm *E* is homomorphic if given *E*(*m*1) and *E*(m2), one can obtain *E*(m1 +/× m2) without decrypting m1 and m2. A straightforward way, the users want to calculate the product of message m1 and message m2. All they have are the encrypted message E(m1) and E(m2) but they do not want to decrypt E(m1) and E(m2),

and then calculate the product (m1 +/× m2). To calculate it, they will calculate T = E(m1) +/× E(m2). Finally, they just decrypt T to get (m1 +/× m2). This property can calculate (m1 +/× m2) but the user will not know what is message m1 and message m2.

This is a major feature that is used to tally ballots in an E-voting system. Pallier algorithm is one of algorithms which have this homomorphic property. Addition of the encrypted ballots will be in the encrypted tally. When an administrator decrypts the Paillier result, he will get the final result of a poll but he will not know which voter voted for which candidate.

Here is how to use Paillier encryption’s additive homomorphic property for tallying votes

Let us call the number of voters is 𝑁v and the number of candidates is 𝑁c. The base used to encrypt messages, is greater than the number of voters (*b* >𝑁v).

Next, the vote messages for candidates will be seen as: 1st candidate is 𝑏0, 2nd candidate is 𝑏1, 3rd candidate is 𝑏2, etc. 𝑁c-th candidate is (Nc-1).

Then the maximum possible number representing a single vote 𝑚max can be expressed

as:

𝑚max = ∑i=1 Nc bi-1

The maximum possible tally of all votes can be:

𝑇max = 𝑁v × 𝑚max

Theorem to the homomorphic property is proved below**:**

**Homomorphic Addition of Plaintexts The product of two ciphertexts will decrypt to the sum of their corresponding plaintexts**, i.e.,

D(E(m1, pk).E(m2, pk) mod n2))= m1+m2(mod n)

because

E(m1, pk).E(m2, pk) =(gm1r1n)(gm2r2n)(mod n2)

=gm1+m2 (r1r2)n (mod n2)

= E(m1+m2,pk)

As a result of this decryption function, one gets simple tallying of all votes. To determine how many votes cast for each candidate we can use the “Division remainder” method with number of the voters as base.

**2.10 Elgamal Cryptosystem**

**Step 1** **:** **Key Generation**

1. Generate g ,an efficient description of a cyclic group G, of order q,

2. Choose a random x 𝜖 {1,…..,, q-1}.

3. compute y= gx mod p

(y, g) is the public key and x is the secret key

**Step 2 : Encryption**

1.Encrypt a message m, choose a random r 𝜖{1,……,, q-1}.

2. Compute c1= gr mod p

3. Computes shared secret s=yr

4. Converts message m in to an element m’ 𝜖 G.

5. Computes c2= m’ . s

6. Ciphertext is (c1,c2) = (gr,m’,yr).

One can easily find yr , if one knows m’. Therefore, a new r, is

generated for every message to improve security. For this reason, r, is also called an ephemeral key.

**Step 3: Decryption**

1. Decrypt a ciphertext (c1,c2), with private key x
2. Compute shared key t = c1x
3. Compute m’ = c2. t-1

where t-1 is the inverse of t in the group G.

**2.11 Homomorphic property of Elgamal Cryptosystem**

The ElGamal encryption scheme is probabilistic, meaning that a single plaintext can be encrypted to many possible ciphertexts, with the consequence that a general ElGamal encryption produces a 2:1 expansion in size from plaintext to ciphertext.

ElGamal encryption scheme has a homomorphic property.

Given two encryptions

(c11, c12) = (gr1,m1 yr1 )

(c21,c22) =(gr2,m2 yr2 )

where r1, r2 are randomly chosen from {1,2,….., q-1}

and m1,m2 𝜖 G.

One can compute

(c11, c12)(c21, c22) = (c11c21, c12c22)

= (gr1gr2,(m1yr1)(m2yr2 ))

=(gr1+r2 ,(m1m2)yr1+r2 )

The resulted ciphertext is an encryption of m1m2.

# CHAPTER 3

## SYSTEM ANALYSIS

## 

**3.1Functional Requirements**

* Voters can register to the system
* Vote is encrypted using homomorphic encryption
* Storing the encrypted votes in cloud
* Decrypting the votes while publishing the result after calculation
* Notification to the voter of every action through SMS.

**3.2 Non-Functional Requirements**

The non-functional requirements are categorized as Software Requirements and Hardware Requirements.

**3.2.1 Software requirements**

Software requirements for the implementation are:

* Operating System :Windows XP/higher versions
* Application server : apache Tomcat
* IDE : Netbeans 8.2
* Front end : HTML, JSP ,java script
* Database connectivity : MySQL

**3.2.2 Hardware requirements**

* System : Processor –Intel® core™ i5-6200 CPU
* Speed- 2.30GHz
* RAM- 2Gb(min)
* Hard Disk- 40 GB
* RAM : 512 Mb

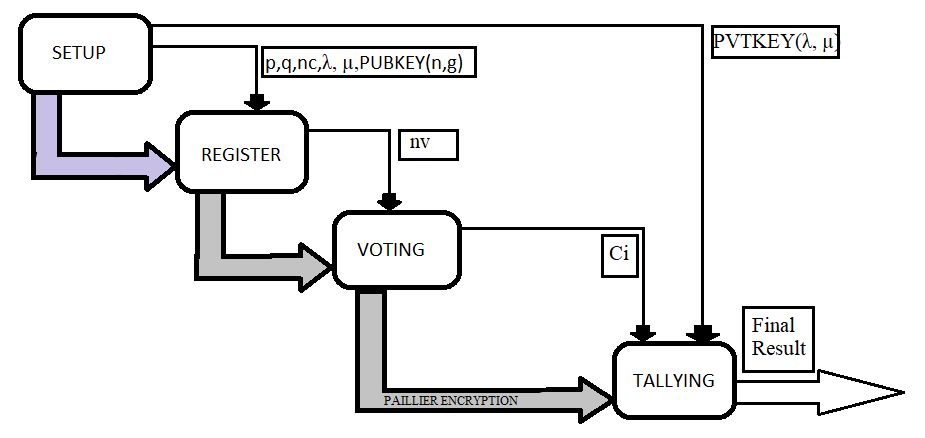
**CHAPTER 4**

**SYSTEM DESIGN AND IMPLEMENTATION**

In this system, there are two participating actors: Admin and voter.

**4.1 System Design**

The design of the e-voting system consist of 3 levels:

****

**Architecture of the E-voting system**

**4.2 USE CASE DIAGRAM:**

ADMIN

USER

**4.3 ACTIVITY DIAGRAM FOR ADMIN:**

Login

Valid credintials?

Create Polls

Edit, Enable for voting

View Results

No

Yes

**4.3.1 ACTIVITY DIAGRAM FOR VOTER:**

Yes

No

Login

Authorized?

View Polls

Submit Vote

View Results

Register

**4.4 SEQUENCE DIAGRAM:**

Register ()

Login ()

View polls

()

Submit Vote ()

User

View Result ()

System

Evaluate user vote

Store ()

Generate Poll result

Admin

Login()

Create Poll()

Enable for Voting()

View Result()

**4.5 JSP Pages**

This project comprises of a home page which includes links for admin and user pages.The JSP pages have been used as the UI and the java servlets perform jobs in the background.

**4.5.1 Admin**

**4.5.1.1 adminlogin.jsp**

This page allows admin to enter the valid name and password to login.Once the admin login, the admin home page has options to add voter details, create polls, manage polls.

**4.5.1.2 adddetailes.jsp**

This page allows admin to enter the user/voter details to the database. The admin enters the name,email id, mobile number, gender, age, location, constituency and most important Voter ID.

**4.5.1.3 createpoll.jsp**

This page allows admin to create new poll. Admin has to enter the poll title, description, location and constituency of poll, candidate details. For each poll creation, a unique poll ID is created.

**4.5.1.4 managepolls.jsp**

This page allows admin to view, delete, enable/disable, edit the polls created. Once the poll is enabled for a constituency, only the voters who are registered for that particular constituency will get notification through sms to their registered mobile number**.**

**4.5.2 User/Voter**

**4.5.2.1 userregister.jsp**

This page allows the voter to enter the details for the registration. The voter is registered successfully only when the name, voter ID, location and constituency matches with the details entered by admin in adddeatils.jsp. once the voter registered successfully , he is taken to the login page.

**4.5.2.1 userlogin.jsp**

This page allows user/voter to login with the help of registered name, Voter ID and genereated OTP which is sent to their mobile number. After successful login, the voter is taken to votepoll.jsp which is the home page for the voter. To cast vote, voter has to click view history link.

**4.5.2.1 viewhistory.jsp**

This page allows voter to cast their vote which comprises of poll title, description, candidates list. Voter now able to cast vote in the registered location and constituency. He is able to view and cast vote for particular candidates who are standing in that location and constituency. The vote is stored in database after encrypting it with Paillier or Elgamal encryption.

All the votes are encrypted using Paillier or Elgamal encryption method and stored in an array which also comprises of secret key, prime variables .The array list of particular poll ID is then stored in DriveHQ cloud .

When the admin closes poll and wants to publish result, the encrypted data in the cloud is decrypted and sends the result to the voters through sms to their registered mobile number.

**4.6 JAVA files**

**4.6.1 register.java**

It is run by admin to register new voters to the MySQL database. The admin enters voter name, age, gender, location, constituency, voter ID.

**4.6.2 adminlogin.java**

This page verifies the name and password entered by the admin. Only valid admin can login to the admin home and manage polls.

**4.6.3 userreg.java**

Once the election is enabled by admin, the voter has to register himself to the poll by entering his details. This page verifies the details entered by voter with the details in the database.

**4.6.4 userlogin.java**

Once the voter is verified successfully, he has to login with OTP sent to his mobile number. This page verifies the valid OTP and allows him to login to the voter home.

**4.6.5 register1.java**

It is a admin run page where the poll details such as title, description, location, constituency, candidate details is stored in to the database.

**4.5.6 voteoption.java**

This is run by the voter where his vote option is encrypted using Paillier or Elgamal encryption and stored in database and cloud.

The Paillier,Elgamal code snippets are documented in the code snippet chapter 6.

**4.7 Drive HQ**

The encrypted data is strored in cloud service provider DriveHQ in an array form.

While publishing results the encrypted data is retrieved from the cloud then decrypt it and then publish it to the voters. DriveHQ offers the features like WebDav Drive Mapping, DriveHQ FileManager, DriveHQ Online Backup, FTP Server Hosting, Email Hosting, File Hosting, Static Web Hosting, Group Account Service, Cloud File Sharing and Single-Sign-On.

Drive HQ is supported by Windows 7, 8, 10, windows server 2008, 2012, macOS, iOS, android, windows phone 8 and windows 10 mobile.

**CHAPTER 5**

**TEST DOCUMENT**

**5.1 OVERVIEW**

This chapter provides a summary of the results of tests performed on the various entities of the “Secure E-Voting system using Homomorphic encryption”.

**5.3 Test Summary**

The following provide the summary of the tests conducted on the Paillier and Elgamal homomorphic encryption codes.

Project Name: Secure E-Voting system using Homomorphic encryption in cloud

System Name:

Version Number:

**5.4 Usability Testing**

It involved the links on the E-Voting application. It was conducted for validity and degree of user-friendlyness.

Test owner: Sharath K

Test Date:

Test Result: The E-voting application provides links which are self explanatory and user friendly.

**5.5 Functionality Testing**

It involved validating the data entered to the database and its connection.

Test owner: Shivkumar

Test Date:

Test result: The application is tested by giving invalid inputs and appropriate error messages were obtained.

**5.6 Unit testing**

This test involved the time complexity of paillier and Elgamal encryption codes.

Test owner : Rahul Balulmath

Test Date:

**Encryption :**

Paillier showed less encryption time than elgamal over the number of votes

**Decryption :**

Elgamal showed less decryption time than paillier over the number of votes

Test Result: This test showed that encryption is done faster in Paillier and decryption is done faster in Elgamal over the number the votes.

**5.7 Security Testing**

This testing involved attempting to access web pages by giving invalid inputs. It failed, proving that the system is secure and requires genuine authentication

Test Owner: Sushmitha Kalkur

Test Date:

Test Result: The web pages were accessed by giving invalid Voter IDs,name,password,OTP. Each such attempt resulted in redirection to the login page .

**Chapter 6**

**CODE SNIPPETS**

**6.1 Elgamal.java**

Random sc = new SecureRandom();

secretKey = new BigInteger("12345678901234567890");

p = BigInteger.probablePrime(64, sc);

b = new BigInteger("3");

c = b.modPow(secretKey, p);

BigInteger X = new BigInteger(str);

BigInteger r = new BigInteger(64, sc);

BigInteger EC = X.multiply(c.modPow(r, p)).mod(p); //Encryption

BigInteger brmodp = b.modPow(r, p);

BigInteger crmodp = brmodp.modPow(secretKey, p);

BigInteger d = crmodp.modInverse(p);

BigInteger ad = d.multiply(EC).mod(p); //Decryption

The above Elgamal code snippet performs key generation, encryption and decryption using homomorphic encryption.

**6.2 Paillier.java**

**6.3 javacode.java**

public void sendsms(String mono,String msg){

try{

String postData;

String retval = "";

String Username ="RSSS";

String Password = "R1S2S3S4";

String type="0";

String dlr="1";

String MobileNo = mono;

String SenderID = "MDSOFT";

String Message = msg;

postData = "username="+Username+"&password="+Password+"&type="+type+"&dlr="+dlr+"&destination="+MobileNo+"&source="+SenderID+"&message="+Message+"";

System.out.println("Post data="+postData);

URL url = new URL(" http://103.16.101.52:8080/sendsms/bulksms?");

HttpURLConnection urlconnection = (HttpURLConnection) url.openConnection();

urlconnection.setRequestMethod("POST");

urlconnection.setRequestProperty("Content-Type","application/x-www-form-urlencoded");

urlconnection.setDoOutput(true);

OutputStreamWriter out = new OutputStreamWriter(urlconnection.getOutputStream());

out.write(postData);

out.close();

BufferedReader in = new BufferedReader( new

InputStreamReader(urlconnection.getInputStream()));

String decodedString;

while ((decodedString = in.readLine()) != null) {

retval += decodedString;

System.out.println("es"+retval);}

in.close();

System.out.println(retval);

}catch(Exception e)

{

e.printStackTrace();

}}

The above snippet is used for sms notification to the registered mobile numbers.

**6.4 userreg.java**

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try { String name=request.getParameter("name");

String uname=request.getParameter("uname");

String pass=request.getParameter("pass");

String emailid=request.getParameter("email");

String mono=request.getParameter("mono");

String gen=request.getParameter("g1");

String age=request.getParameter("age");

String loc=request.getParameter("loc");

String cons=request.getParameter("const");

String vid=request.getParameter("vid");

int ag=Integer.parseInt(age);

Class.forName("com.mysql.jdbc.Driver");

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/voteuser","root","root");

Statement st=con.createStatement();

ResultSet rs=st.executeQuery("select \* from userreg where uname='"+uname+"'");

if(rs.next())

{ response.sendRedirect("userregister.jsp?msg=Already\_Exisit");}

Else{

PreparedStatement pstm=con.prepareStatement("insert into userreg values(?,?,?,?,?,?,?,?,?,?)");

pstm.setString(1, uname);

pstm.setString(2, name);

pstm.setString(3, pass);

pstm.setString(4, gen);

pstm.setInt(5, ag);

pstm.setString(6, emailid);

pstm.setString(7, mono);

pstm.setString(8, loc );

pstm.setString(9, cons);

pstm.setString(10, vid);

int x=pstm.executeUpdate();

if(x>0){ response.sendRedirect("userlogin.jsp?msg=RegisteredSuccess");}

else{ response.sendRedirect("userregister.jsp?msg=Already\_Exisit");}}}

catch(Exception e){ e.printStackTrace();}}

The above snippet takes input from users/voters and stores in database.

**6.5 adminlogin.java**

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

ObjectOutputStream out1 = new ObjectOutputStream(response.getOutputStream());

try {String uname=request.getParameter("uname");

String pass=request.getParameter("pass");

if((uname.equalsIgnoreCase("admin"))&&(pass.equalsIgnoreCase("admin")))

{ response.sendRedirect("adminhome.jsp?msg=success");}

else

{ response.sendRedirect("index.html?msg=Credientials\_MisMatch");}

}catch(Exception e){

e.printStackTrace();}}

The above snippet verifies valid admin login name and password .

**6.6 ftpcon.java**

public class Ftpcon {

FTPClient client = new FTPClient();

FileInputStream fis = null;

boolean status;

boolean success;

public boolean upload(File file){

try{ client.connect("ftp.drivehq.com");

client.login("eswark77", "eswark77");

client.enterLocalPassiveMode();

fis = new FileInputStream(file);

status= client.storeFile(" /ke/"+file.getName(), fis);

client.logout();

fis.close();}}}

public boolean filedownload(String fname)

{try { client.connect("ftp.drivehq.com");

client.login("eswark77", "eswark77");

client.enterLocalPassiveMode();

File localfile = new File("D:\\"+fname);

OutputStream outputStream = new BufferedOutputStream(new FileOutputStream(localfile));

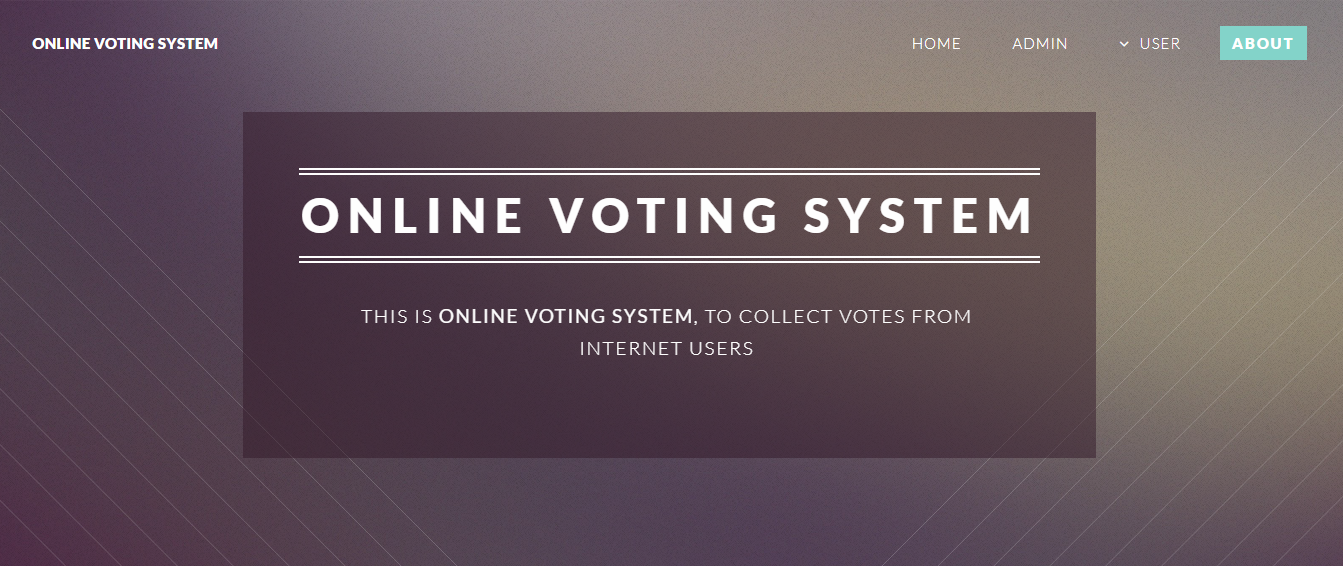
success = client.retrieveFile(" /ke/"+fname, outputStream);

outputStream.close();

}catch(Exception e){

e.printStackTrace();}}}

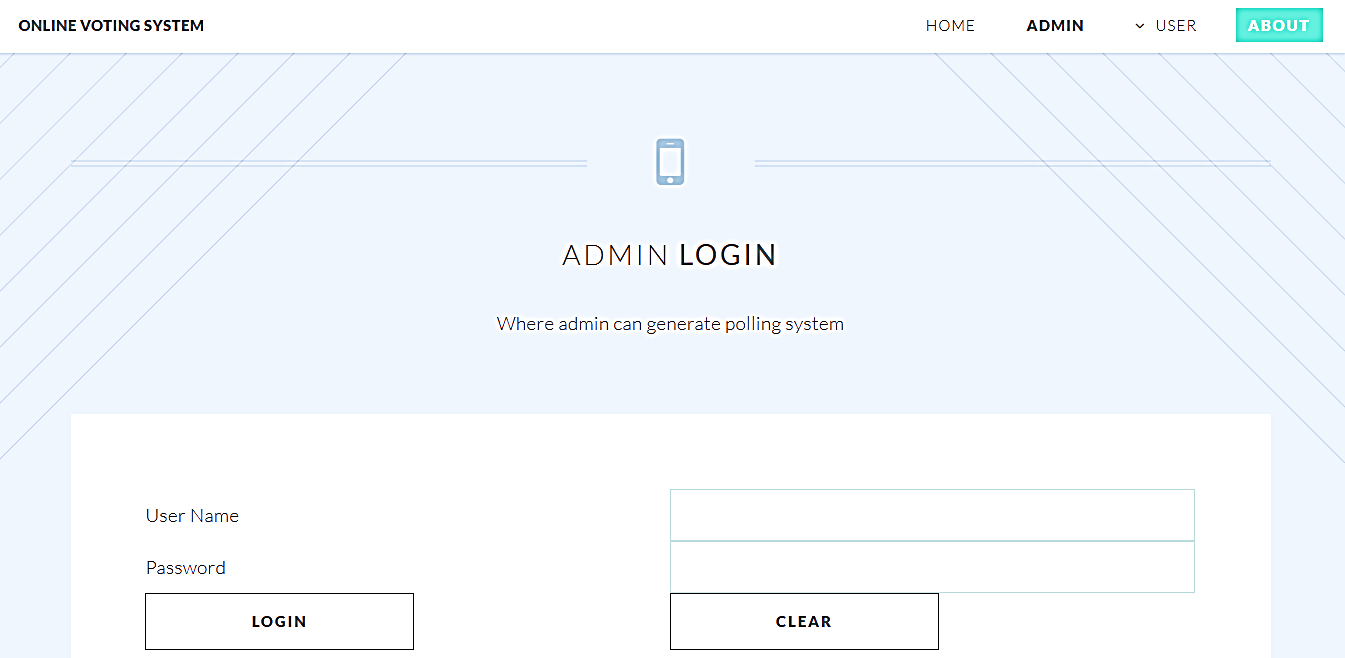
**Chapter 7**

**Snapshots**

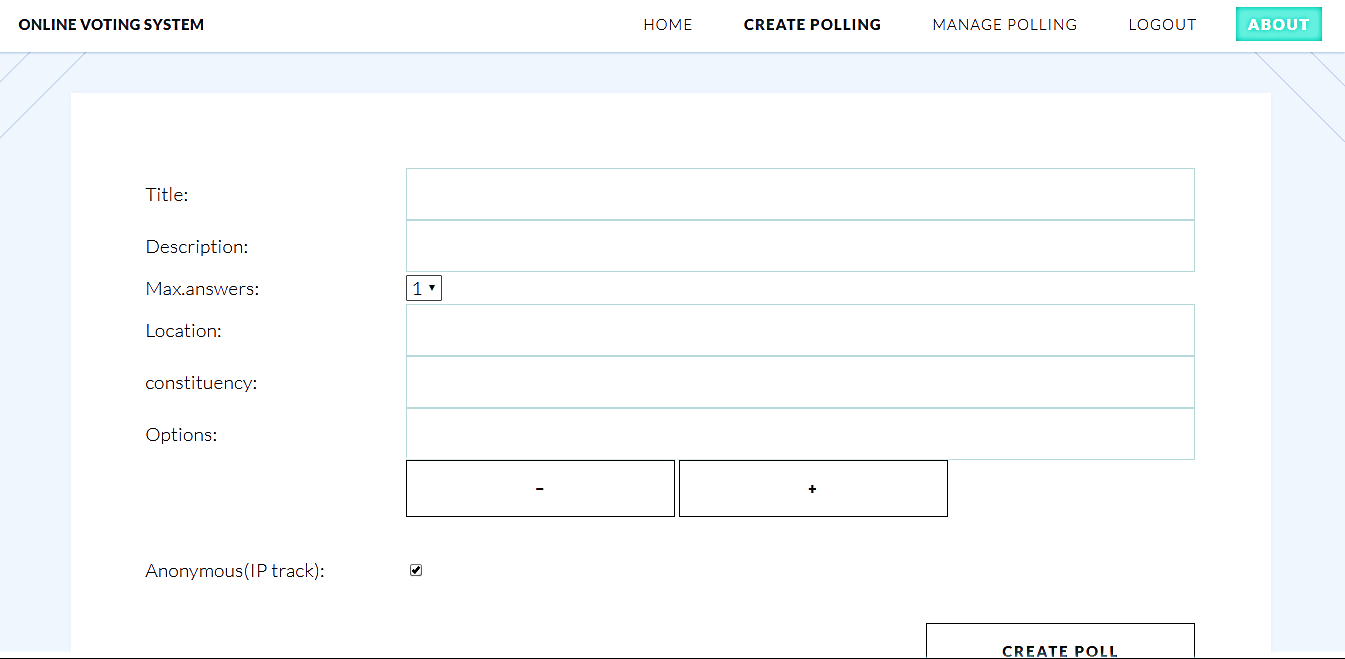
**7.1 Online Voting System home page**

Figure

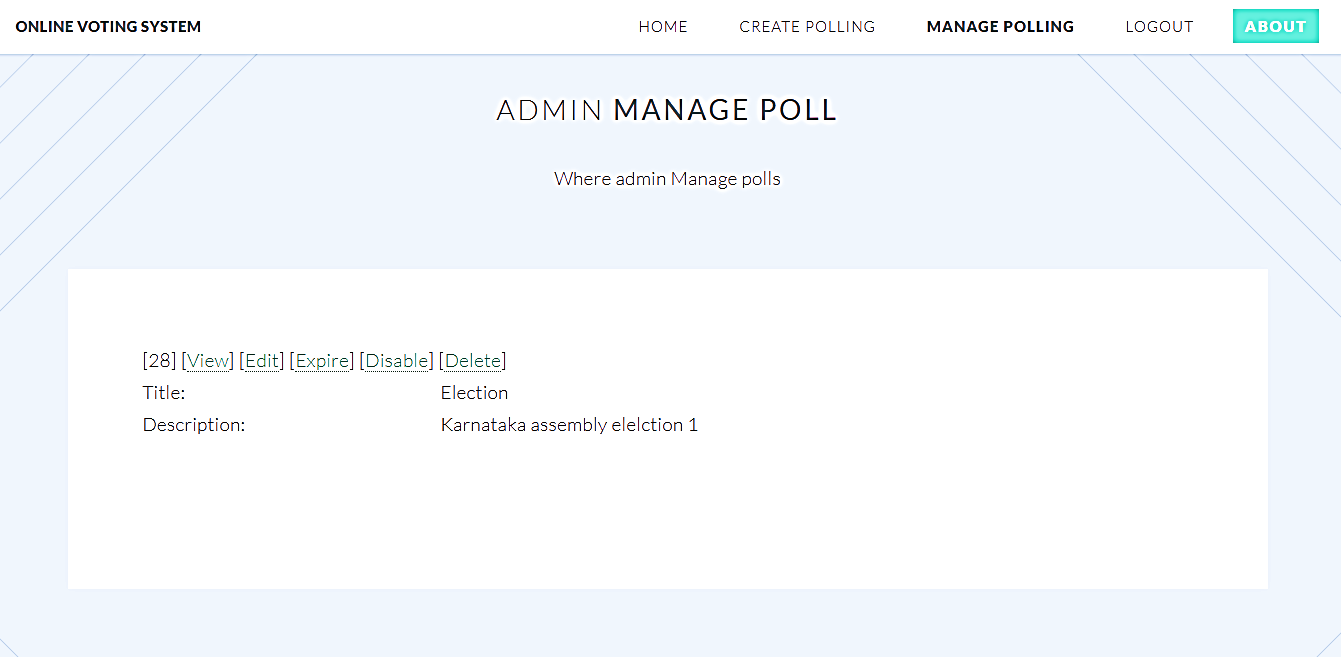
This is the home page common for both admin and voter .

**7.2 Admin Login**

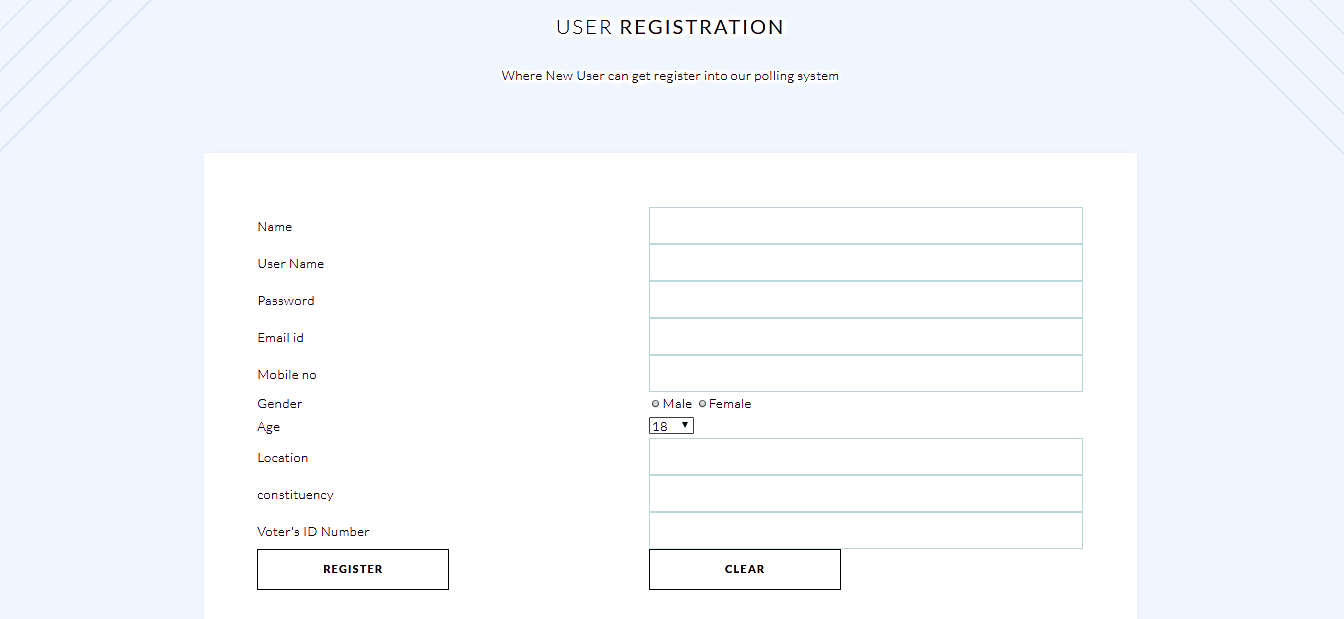
**7.3 Create poll**

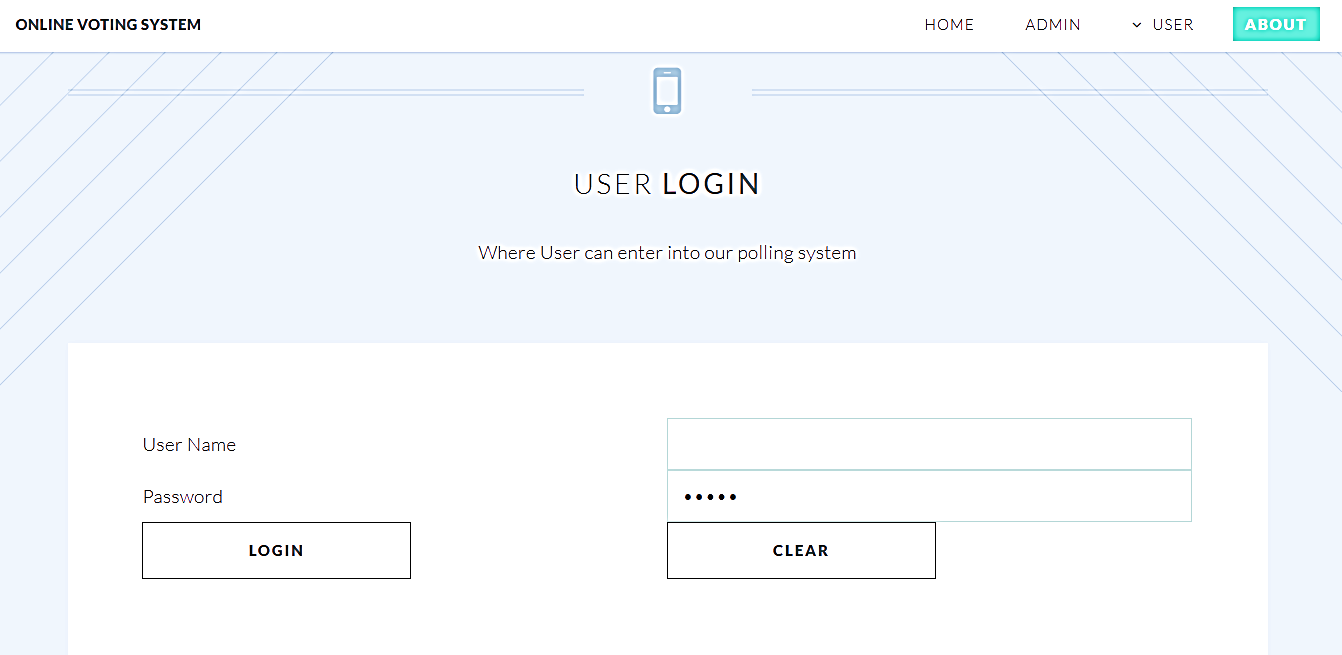
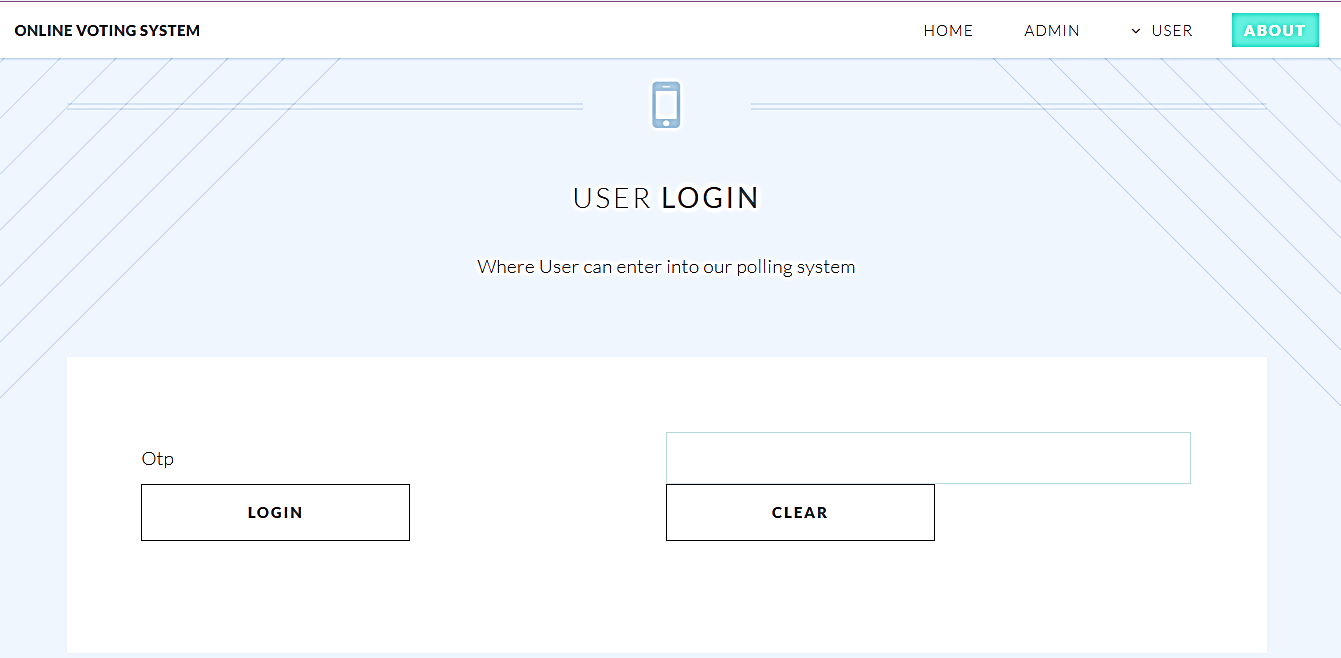


**7.4 Manage Poll**



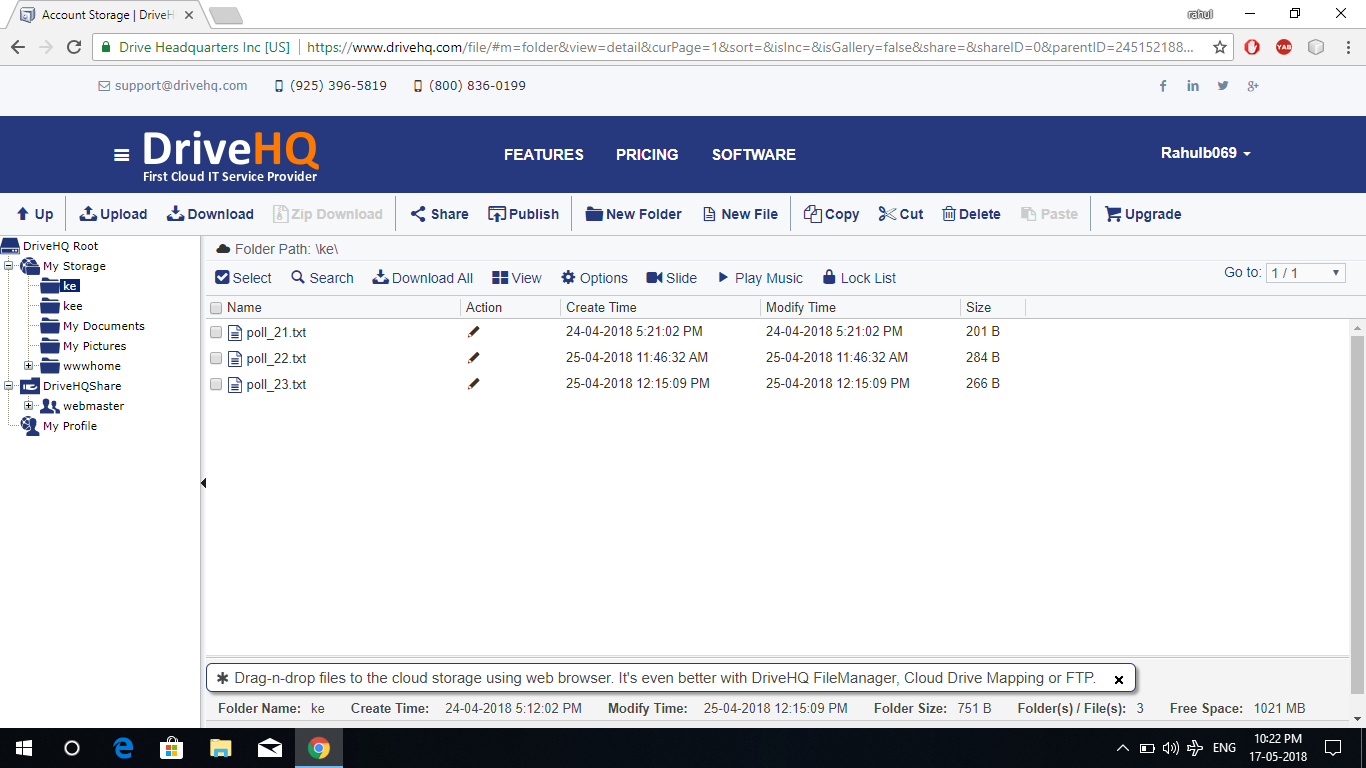
**7.5 User Registration**

****

**7.6 User Login**

**7.7 Vote Option**

**7.8 DriveHQ**

****

**Chapter 8**

**Results**

**Chapter 9**

**Conclusion and Future Work**

This E-voting system works correctly and it is still simple. It can implement some

secure methods that can convince voters using the E-voting system instead of using

traditional voting. However, to have a professional E-voting system, the following

challenges should be further investigated in future work.

First, based on this material, a developer can write a plugin that voters can install

in their web browsers. They can use web browsers to vote. The plugin has to ensure the

use of the Paillier and Elgamal algorithm for tallying between a client and a server.

Second, applications to vote from smartphones can be built as well.

Accordingly, voters will need to download the voting application and install to their smartphones, then they will be able to vote. It is much more convenient because at the present, most people use smartphones. However, cryptography algorithms and the SSL client/server protocol should be considered for such implementation. All methods should be supported to develop a professional E-voting system for such platforms.

This project, based on Paillier and elgamal encryption, implements a secure E-voting system. This combination warrants that the vote can be transferred over

the Internet securely and counted correctly. This E-voting system ensures the end-to-end verification of the whole voting process. It is the quickest, cheapest, and the most

efficient way to vote. Thus, if an election supports this E-voting system, voters can use it

to vote without worrying about their votes. Moreover, they can save their time by not

going to a polling station to vote. All in all, the project meets important secure

requirements of an E-voting system. Some other future work as mentioned can be done in further research to complete the professional voting system.

**Bibliography**

[1] MIT International Journal of Computer Science and Information Technology, Vol. 6, No. 1, January 2016, pp. 1-6 1 ISSN 2230-7621©MIT Publications

[2] D. Hrestak and S.Picek , “Homomorphic Encryption in the cloud” , in proceeding of IEEE International Convention on Information and communication technology ,Electronics and micro electronics, Opatija, Croatia, PP:1400-1404,2014.

[3] 7th International Conference on Communication, Computing and Virtualization 2016

Homomorphic Encryption for Security of Cloud Data

[4] Secure Cloud Computing through Homomorphic Encryption

Maha TEBAA, Said EL HAJII

[5] Implementing Homomorphic Encryption

Computer Science Tripos, Part IISt John’s CollegeMay 18, 2011

[6] Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, July 4 - 6, 2012, London, U.K. Homomorphic Encryption Applied to the Cloud Computing Security

[7] Homomorphic encryption and algorithms, Paulet R,Bertino E,2014, X11

[8] [http://www.i-programmer.info/news/112-theory/2330-darpa-spends-20-](http://www.google.com/url?q=http%3A%2F%2Fwww.i-programmer.info%2Fnews%2F112-theory%2F2330-darpa-spends-20-&sa=D&sntz=1&usg=AFQjCNGV_X6oww6sFaK3nvFZXO_vwfrZRg)  
million-on-homomorphic-encryption.html

## [9] Cryptology ePrint Archive: Report 2011/277

[10]  Z. Brakerski. [Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP](http://eprint.iacr.org/2012/078). In *CRYPTO 2012* (Springer)

[11] A. Lopez-Alt, E. Tromer, and V. Vaikuntanathan. [On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption](https://eprint.iacr.org/2013/094). In *STOC 2012* (ACM)

[12] C. Gentry, A. Sahai, and B. Waters. [Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based](http://eprint.iacr.org/2013/340). In *CRYPTO 2013*(Springer)

[13] International Journal of Information & Computation Technology. ISSN 0974-2239 Volume 4, Number 8 (2014), pp. 811-816 © International Research Publications House http://www. irphouse.com

[14] “Design and Implementation of Secure Remote e-Voting System Using Homomorphic Encryption”. International Journal of Network Security, Vol.19, No.5, PP.694-703, Sept. 2017