ELECTRONIC VOTING SYSTEM

Prepared By
Md. Mostafizur Rahman
STUDENT ID: 02201006

Md. Sharfuddin Bhuiyan
STUDENT ID: 02101059

Md. Rajibul Hossain
STUDENT ID: 02201010

A thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering

JANUARY 2007

BRAC UNIVERSITY

Department of Computer Science and Engineering

BRAC University, Dhaka, Bangladesh

Pages Containing Figures

DECLARATION

In accordance with the requirements of the degree of Bachelor of Computer Science and Engineering in the division of Computer Science and Engineering, I present the following thesis entitled *'Electronic Voting System'*. This work was performed under the supervision of Dr. Sayeed Salam.

We hereby declare that the work submitted in this thesis is our own and based on the results found by our self. This thesis, neither in whole nor in part, has been previously submitted for any degree.

Signature of	Signature of
Supervisor	Author

Acknowledgement

I would like to express my sincerest appreciation and profound granitite to my supervisor Dr. Sayeed Salam, Chair Person, Department of Computer Science and Engineering, BRAC University, and Risat Mahmud Pathan lecturer of BRAC University for their supervision, encourage guidance. In the course of the project development he discussed problems. He helped to overcome hurdles. His keen interest and valuable suggestions and advice were the source of all inspiration to us. I would like to thank BRAC University for helping with all the required resource for completion of the thesis work.

Abstract

Using the decade old election system to collect votes from the citizens is no longer considered efficient due to the various recurring errors. So time has arrived that the paper based primordial voting system which has already proven itself an inefficient and slow procedure is changed immediately. The system that is being followed currently, from data collection procedure to counting of the votes is a manual process. Here we are proposing an automated electronic voting system. It starts with automated registration system that will provide the secured database of the voters' information. Voter details will be stored against their finger prints in the main database. The election commission authority is authorized to access the details but they aren't authorized for modifying or changing the details. Modification of the voters' information requires the fingerprint of the particular voter. So the system will help to minimize the corruption done by others, and hopefully corruption may be diminished at some point of time. In this system Voter will select his/her preferable candidate by providing his or her opinion on a touch screen where all candidates' voting sign is displayed. Four layered network system will be used here for sending the votes from client to the main database there are three application server, and a client. Among them one application server will work as dispatcher. The encrypted votes will be sent from the client to the dispatcher through an application server and this layer will send those votes to main database through another application server. They will be counted there automatically which will take lesser time than the manual system. So the result will be faster, more accurate and reliable.

1. Introduction

1.1 Background:

For a democratic country public opinion is the most important determinant to establish a government and voting is the process through which people display their opinion and help to setup a democratic government. So the voting system should be reliable, accurate and it must be transparent.

Let's consider the following points:

- The system that exists currently in Bangladesh is totally paper based and manual that takes lots of time and the government has to bear the financial expenses for this purpose.

- The voters are registered just before the poll so the election commission gets some time in hand for making all the necessary arrangements with in this short period of time. They just add the new voters with the previous voters so that the people who have deceased by this time may be considered as the existing voter if they are not informed. So people may not bestow their faith on the voters list as it contains numerous fake voters.

- Again the authority itself may be corrupted and can allow some fake voters to participate. If any voter stays abroad or misses the registration processes somehow due to prior obligations or unavoidable circumstances, he or she wouldn't be considered as a voter unless or until s/he informs the authority and in this case most of the time people don't show any interest upon this process.

- Any voter may change his place of residence between two elections and regarding this case if the authority is not informed they are not considered as the voter of that area though he is a voter as per the constitution. Therefore he misses the opportunity to confer his opinion. Even if he is registered voter of his new locality it is often seen that he is still existing voter of his old area. Thereby he can vote twice which is illegal.

- Sometimes people ruin their votes by stamping on two or more signs mistakenly. This is also a drawback of paper based voting system.

- While casting the votes the acting officers present in the centers marks a voter with a black ink on his or her nail but it is removable. So there is a chance for casting illegal votes.

- Again these votes are counted manually so the process becomes a gradual one which may be inaccurate as well.

All these problems together made people think about inventing a new system that will reduce corruption, increase accuracy and fast paced. The concept of electronic voting system comes from this necessity.

1.2 Related Works:

1.2.1 The national committee Electronic voting system Tallinn 2005

1.2.1.1 Overview:

The subject of e-voting has been actively discussed in Estonia on different levels since the beginning of this century. There exists an opportunity and motivation to implement such a project with the purpose of offering voters a possibility of e-voting at local government council elections of 2005, for:

- There exists a legal basis for carrying out e-voting which is laid out in the following legal acts:
  - Local Government Council Election Act, § 50;
  - Riigikogu Election Act, § 44;
  - European Parliament Election Act, § 43;
  - Referendum Act, § 37.

- A public key infrastructure enabling secure electronic personal authentication using digital signatures and ID-cards has been created – currently (August 2005) over 800,000 ID-cards have been issued, meaning that most of the eligible voters are covered.

This overview gives a general description of the technical and organizational system of the planned e-voting system. This document:

- Defines the scope of e-voting, in other words, defines the subject in the context of the election process as a whole,
- Specifies the system requirements,
- Specifies the participating parties of the system and describes their roles,
- Specifies the architecture of the e-voting system, the general description of functionality, protocols and algorithms,
- Analyses and describes possible security hazards and examines the compliance of the system to security requirements.

- This document discusses to some extent but does not concentrate on:

  - Exact specification of the security level of system components,
  - Specification of data structures,
  - Choice of software and hardware platforms,
  - Technical structure of the system's network – server redundancy, network security measures to be used (firewalls, intrusion detection systems), architecture of network connections.

1.2.1.2 Benefits of Electronic Voting system, Tallinn 2005:

- The proposal could maintain the major principle of e-voting; which is of being similar to regular voting system.
- The system was compliant with the election legislation and principles and was at least as secure as regular voting.
- Therefore e-voting must be uniform and confidential, so the national committee could successfully make the system identical and also maintain the highest level of security.
- The national committee ensured single vote for a single person by re-voting and considering the last given vote on their web site. They will again arrange traditional system of voting if any person wants to change his opinion and this vote will get higher preference than e-vote.
- The process of collecting votes was secure, reliable and accountable.

1.2.1.3 Limitations of Electronic voting system, Tallinn 2005:

- The national committee didn’t completely get out of the traditional voting system. Hence the system couldn’t cut down the cost rather there was an upsurge in the cost as they are conducting both the e-voting session and the traditional voting system.
- The process is time consuming as the national committee allows the voters to vote on their web page from 6th day to 4th day before the traditional poll. As a result the process takes at least 7 to 8 days to publish the result.

1.2 Related Works:

1.2.2 Analysis of an Electronic Voting System

TADAYOSHI KOHNO ADAM STUBBLEFIELD AVIEL D. RUBIN DAN S. WALLACH

February 27, 2004

1.2.2.1 Overview:

With significant U.S. federal funds now available to replace outdated punch-card and mechanical voting systems, municipalities and states throughout the U.S. are adopting paperless electronic voting systems from a number of different vendors. This system present a security analysis of the source code to one such machine used in a significant share of the market. This analysis shows that this voting system is far below even the most minimal security standards applicable in other contexts. Several problems including unauthorized privilege escalation, incorrect use of cryptography, vulnerabilities to network threats, and poor software development processes have been identified. This system shows that voters, without any insider privileges, can cast unlimited votes without being detected by any mechanisms within the voting terminal software.

Furthermore, it has been showed that even the most serious of the outsider attacks could have been discovered and executed without access to the source code. In the face of such attacks, the usual worries about insider threats are not the only concerns; outsiders can do the damage. That said, we demonstrate that the insider threat is also quite considerable, showing that not only can an insider, such as a poll worker, modify the votes, but that insiders can also violate voter privacy and match votes with the voters who cast them. We conclude that this voting system is unsuitable for use in a general election. Any paperless electronic voting system might suffer similar flaws, despite any “certification” it could have otherwise received. It has been suggested that the best solutions are voting systems having a “voter-verifiable audit trail,” where a computerized voting system might print a paper ballot that can be read and verified by the voter.

1.2.2.2 Benefits:

- One of the biggest advantages of smartcard is ability to perform cryptographic operations internally, and with physically protected keys. Despite the lack of cryptography, there is no protected authentication of the smartcard to the voting terminal.

- Smartcard designs allow cryptographic operations to be performed directly on the smartcard, making it possible to create systems that are not as easily vulnerable to such security breaches.

- Upon checking that the smart card is "active," the voting terminal collects the user's vote and then deactivates the user's card; the deactivation actually occurs by rewriting the card's type, which is stored as an 8-bit value on the card, from VOTER_CARD (0x01) to CANCELED_CARD (0x08).

- If a voter decides to cancel his or her vote, the voter will have the opportunity to vote again using that same card (and, after the vote has been cast, m_VoterSN will no longer be recorded).

- This system uses two types of card the voter cards that normal voters use when they vote; there are also administrator cards and ender cards, which have special purposes in this system. The administrator cards give the possessor the ability to access administrative functionality (the administrative dialog BallotStation/AdminDlg.cpp), and both types of cards allow the possessor to end the election (hence the term "ender card").

1.2.2.3 Problems:

- If an adversary could learn the protocol between voter cards and voting terminals. After voting, instead of returning the canceled card to the poll-worker, the adversary could return a fake card that records how it is reprogrammed, and then dumps that information to a collaborating attacker waiting in line to vote. Alternatively, the attacker could attach a “wiretap” device between the voting terminal and a legitimate smartcard and observe the communicated messages.

- The adversary could program a smart card to ignore the voting terminal’s deactivation command. Such an adversary could use one card to vote multiple times.

- However, because m_VoterSN is only stored for those who did not vote, there will be no way for the tabulating system to distinguish the real votes from the counterfeit votes. This would cast serious doubt on the validity of the election results.

1.3 Key Features of Proposed System:

We have devised two key features after reviewing the problems from the paper based voting system and some works related to e-voting which will provide the required solutions to these problems.
Two key features are:

1) Maintenance of Database System
2) Automated Registration System

1.3.1 Maintenance of Database System:

We are introducing here computerized registration form for each voter through which voter details including name, area, etc. can be entered. From these details an algorithm will generate a unique id and this id will be stored under the fingerprint of the voter. Now the question arises when the fingerprint and the details should be collected. This registration system will retain a single database named **Population Database** for all the citizens of Bangladesh. It will have two segments first one is **Primary Database** that is for 0 years old to less than 18 years old citizens. The second segment **Secondary Database** will keep the data for the people who are at least 18 years old**.** Third segment is **Voter Database** that will contain only the living existing voters.

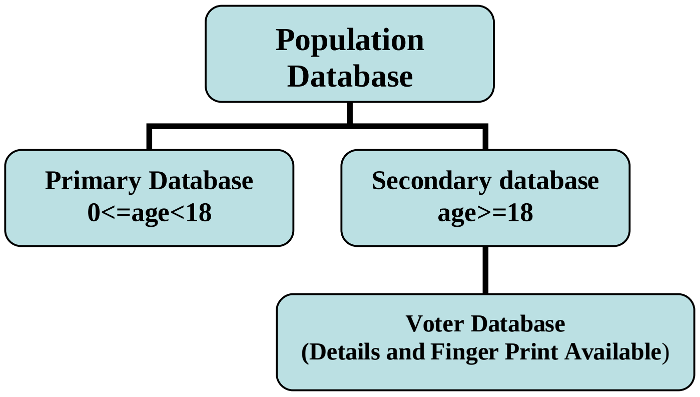

Population Database
Primary Database 0<=age<18
Secondary database age>=18
Voter Database (Details and Finger Print Available)

Fig: Database System

Citizens of Bangladesh will be migrated automatically from the primary database to secondary database when they will become 18. People who have expired already won't be migrated to the voter database but they will be in the population database and their status will be dead. People who have already migrated to the voter database and then suppose he or she has died. He or she will be deleted from the voter database but they will be still there in the primary or secondary database. This approach will save the space of the voter database as well as it will make the system more accurate. The inputs of the primary database will be collected from the 'X' who has the information of all aged people of Bangladesh and the information of death people will be collected from 'Y' who has the records of the death people of the country.

1.3.2.Registration System:

The election commission authority will collect the details as well as finger prints from the people who are at least 17 years. Birth certificate, H.S.C or S.S.C certificate will proof one's age. People who don't have this certificate may use chairman certificate or commissioner certificate to proof their age. People who won't give their details and finger prints to the authority their status will not be registered but he will also be migrated to the **secondary database** automatically again he won't be migrated to the **voter database**. They will be able to register later and then he will be migrated to the **voter database** immediately. The authority will collect the details from even 17 years old people so that if any person becomes 18 years old between the time of collecting data and the election he or she will be able to give the vote. Even the people who live outside the country will be able to give his or her vote. It doesn't matter whether he has the birth certificate or H.S.C or S.S.C certificate. His passport will prove his legality for being the citizenship of Bangladesh and that will also proof his age. But he needs to be registered that means he has to give his details as well as finger print to the authority. Then he will be able to attend the election even he gets back to the country on the day of the election.

1.4 Solution:

- **1.4.1 Boundary less Voting:**

Another thing is that a person will be able to attend the vote from any where in the country as his id contains the area where he is from. The database that the election commission authority will maintain for storing the votes will have several segments for several constitutes. A person’s vote will directly go to that specific segment. So it doesn’t matter from where he is attending the election.

- **1.4.2.Ensuring Single Vote For Single Person:**

In this system a person can vote as many as he wants. But the first one vote will be counted as his id will be blocked just after the casting of his first vote. This system will reduce the possibility of casting the illegal votes.

- **1.4.3 Reducing of Ruining Votes:**

As people will vote on a touch screen so it will reduce the possibility of ruining votes. But this process is not completely error free. Some people may touch more than one sign at a time. In that case the sign that gets at least 50% pixels touched will be selected as the choice of the voter.

- **1.4.4 Automatic Counting:**

This system will count the votes automatically so the counting process will be faster and that will help to publish the result faster.

2. Issues Related To This System:

The issues this system has covered those are:

- Database maintenance efficiency
- Fingerprint matching
- Network issues
- Encryption and decryption

2.1 Database Maintenance Efficiency:

Our database is mainly divided into two parts. Those are population database and voter database.

After entering all existing people in the data base, system will collect new entry from the medical officer. New entry may be comes from Foreign Ministry. All these entry will preserve at population database. Population database has two segments one is primary database and the other one is secondary database. Primary database keeps the record of that part of population whose age is between 0 to less than 18 years. If a person becomes 18 then his record is automatically transfer from primary database to the secondary database. But still he/she will not be a voter until his/her finger print is not given to the authority. If anyone dies, his /her record will remain in the population database but, their status will be dead. It goes for both of the databases of whose age are 0 to less than 18 and over 18 years. Moreover if anyone wants to change his/her information it is possible by using their fingerprint. So there is no chance to make double entry.

Voter database is used specially during the election period. It doesn't have any other works to deal with other than election. When the election comes the people who are voter, their record upgrade from population database to voter database. At that time this database will distribute the voter list according to their area Id. This will happen only at voting period. Otherwise rest of the time all the data will preserve in a single voter database, the voter list only distribute at voting period.

2.2 Fingerprint matching :

Most efficient and effective part of this system is fingerprint. Fingerprint is a unique identification for any voter. All the information about voter will be preserve against the fingerprint. At the registration period when anyone gives his/her information the system will generate an Id against that information. This Id will be protected by his/her fingerprint. If any one tries to make double entry in the voter database he/she can not make that because of fingerprint. So the system makes ensure single entry for individual.
The system will not transfer the entry until his/her fingerprint provided. No one can change others information only because of fingerprint. Even administrator can not modify others information. So all the information will be strongly preserve in the database. In this system the administrator can only excess the data he/she can not modify anything only because of fingerprint.

When any voter gives his/her vote the system will at first find his/her fingerprint at the database. It will at first find out the specific area where the vote will be cast. Then the system will check whether the specific id is block or unblock. Block means that this person has already voted unblock means opposite. So when system find Id block it will reject that vote otherwise it will cast that vote and preserve that against that fingerprint.

Moreover if anyone wants to give others priceless vote he/she can not do that. Because fingerprint is a unique identity for everyone. So the system will provide single vote for single person.

2.3 Network Issues:

A three tiered network system has been proposed here for implementation of this electronic voting system. There will be a number of clients in the most root level that is it may be in Police Station level or sub-district level of a country. But it is necessary to have a lot of clients in the root level. Some clients together make a cluster. The cluster size should between 10 to 15 clients.

In the district level there will be a dedicated application server for those police station or sub-district clients under that district. These application servers in this level won't accept packets from any other cluster under another application server.

In the division level there will be some dispatcher for each division. These dispatchers will also be dedicated for those districts under that division. There will be a layer of application server layer after the dispatcher through which the dispatcher will pass the encrypted vote to the main database.

Main database will have several segments for each constitute. The encrypted vote will be checked first to see whether the id of the voter is locked or not. If it is locked then it won't be passed but if it is not locked then the vote will be sent to that segment reserved for that seat.

The client's responsibility is to receive the finger print and the vote and then encrypt them. It will also preserve each and every packet. Afterward it will send the packet to the application server through radio linked network. These dedicated application servers will pass them through another radio linked network to the dispatcher. Up to this radio linked network is being used as the network will be engaged for only sending packets and a radio link network provides 4 kbps speed which is good enough for this purpose.

Dispatcher's responsibility is to search the next application server which is not busy right then. So the more application server the system would have in this level the faster it will work. The encrypted vote will be passed to that server and after that it will be directly passed to main database. Here from dispatcher to the main database a fiber optic network will be used as the dispatcher has to handle millions of packets at a time so it will need a better paced as well as secured network system.

At each and every layer the packet will be preserved for backup. At any time network can be down but the voting can not be stopped at that time. So for that type of situation we need some backup. At client layer when any packet sends to the application layer that packet will also preserve in that client. All packets store in a queue. If the radio link is down for any reason at that time the data will not send to the application server, so at that time all packets will be stored in the queue. When network is ready to send data then it will start sending packet to the application server. It will send packet by using their waiting period. It will first send the packets which have the maximum waiting time. The system will not accept same data twice; it will automatically reject the same data.

At any layer the network can be down so backup is much needed at each and every stage. Sometimes optical fiber communication may be corrupted. At that time data transmission will be stopped but voting can not be stopped. So, to handle this type of problem the system need backup every stage. At every layer the system make some queues to back up data.

This type of network structure will also preserve time. Because layers will not wait for any kind of response. So total voting time will be less then the existing system. Voter will just come to the center and give his/her priceless vote. System will automatically do the rest of the operation.

2.4 Encryption and Decryption:

Security is a broad topic and covers a multitude of sins. In its simplest form, it is concerned with making sure that nosy people cannot read, or worse yet, secretly modify messages intended for other recipients. It is concerned with people trying to access remote services that they are not authorized to use. Most security problems are intentionally caused by malicious people trying to gain some benefit, get attention, or to harm someone. Secrecy has to do with keeping information out of the hands of unauthorized users. This is what usually comes to mind when people think about network security.

There are various types of encrypting and decrypting algorithm. Substitutions ciphers, transposition ciphers, One-time pads, Quantum cryptography, Symmetric, Asymmetric these are some algorithms. By using these types of algorithm any network system can be secured.

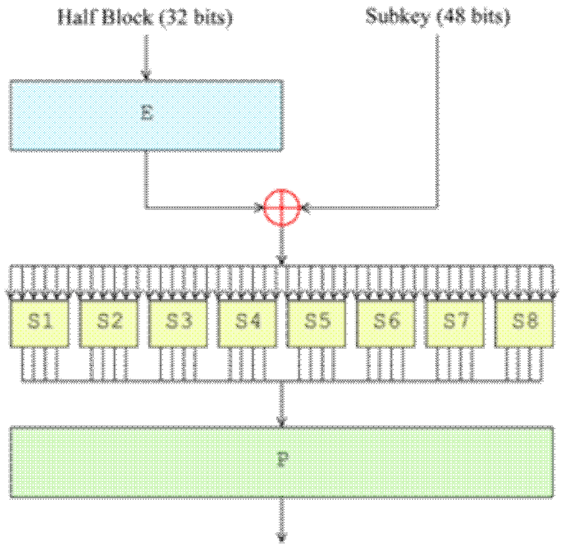
Symmetric algorithm use same key for encrypting and decrypting any data. It is quite fast searching process. But the problem is it can be easily modified by the active intruder at the transmission period. If the key length is more then six bit this type of problem can be stopped. Because if the key length is six digits which means a million possibilities. The longer the key, the higher the work factor the cryptanalyst has to deal with.

Asymmetric algorithm uses two types of key for encryption and decryption. This algorithm uses a public key for encryption and a private key for decryption. Public key can be changed randomly. This public key will be at sender point and the private key at receiver end. If anyone has public key then he/she can only encrypt the data. But if anyone has private key then he/she can also both encrypt and decrypt the data. This algorithm is much more secured then symmetric algorithm but the problem is, it is quite slower then symmetric algorithm.

In this system there are two types of data will be send. One is vote and another one is fingerprint. At first the will be encrypted by using DES algorithm. When the system get a fingerprint from the voter then the system generate a hash algorithm, then it will get a hash function. This hash function will again encrypt the encrypted data. Whole encrypted data will be again encrypted by using a public key. Then the encrypted data will send as a packet. This packet will decrypt at last layer that means at main data base layer. At first the packet will decrypt by using a private key. Then the system will find out the status about that specific data. Then it will decrypt the remaining packet. Here we select DES algorithm because it is much more suitable for this type transmission.

DES is a ***block cipher***--meaning it operates on plaintext blocks of a given size (64-bits) and returns ciphertext blocks of the same size. Thus DES results in a ***permutation*** among the 2^64 (read this as: "2 to the 64th power") possible arrangements of 64 bits, each of which may be either 0 or 1. Each block of 64 bits is divided into two blocks of 32 bits each, a left half block **L** and a right half **R**. (This division is only used in certain operations.)

Data Encryption Standard



The Feistel function (F function) of DES

Example: Let **M** be the plain text message **M** = 0123456789ABCDEF, where **M** is in hexadecimal (base 16) format. Rewriting **M** in binary format, we get the 64-bit block of text:

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
L = 0000 0001 0010 0011 0100 0101 0110 0111
R = 1000 1001 1010 1011 1100 1101 1110 1111

The first bit of **M** is "0". The last bit is "1". We read from left to right.

DES operates on the 64-bit blocks using *key* sizes of 56- bits. The keys are actually stored as being 64 bits long, but every 8th bit in the key is not used (i.e. bits numbered 8, 16, 24, 32, 40, 48, 56, and 64). However, we will nevertheless number the bits from 1 to 64, going left to right, in the following calculations. But, as you will see, the eight bits just mentioned get eliminated when we create subkeys.

Example: Let **K** be the hexadecimal key **K** = 133457799BBCDFF1. This gives us as the binary key (setting 1 = 0001, 3 = 0011, etc., and grouping together every eight bits, of which the last one in each group will be unused):

K = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

The DES algorithm uses the following steps:

Step 1: Create 16 subkeys, each of which is 48-bits long.

The 64-bit key is permuted according to the following table, **PC-1**. Since the first entry in the table is "57", this means that the 57th bit of the original key **K** becomes the first bit of the permuted key **K**+. The 49th bit of the original key becomes the second bit of the permuted key. The 4th bit of the original key is the last bit of the permuted key. Note only 56 bits of the original key appear in the permuted key.

PC-1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Example: From the original 64-bit key

K = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

we get the 56-bit permutation

K+ = 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111

Next, split this key into left and right halves, C_0 and D_0, where each half has 28 bits.

Example: From the permuted key **K**+, we get

C_0 = 1111000 0110011 0010101 0101111
D_0 = 0101010 1011001 1001111 0001111

With C_0 and D_0 defined, we now create sixteen blocks C_n and D_n, $1<=n<=16$. Each pair of blocks C_n and D_n is formed from the previous pair $\boldsymbol{C_{n-1}}$ and $\boldsymbol{D_{n-1}}$, respectively, for $\boldsymbol{n}$ = 1, 2, ..., 16, using the following schedule of "left shifts" of the previous block. To do a left shift, move each bit one place to the left, except for the first bit, which is cycled to the end of the block.

Iteration Number	Number of Left Shifts
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

This means, for example, $\boldsymbol{C_3}$ and $\boldsymbol{D_3}$ are obtained from $\boldsymbol{C_2}$ and $\boldsymbol{D_2}$, respectively, by two left shifts, and $\boldsymbol{C_{16}}$ and $\boldsymbol{D_{16}}$ are obtained from $\boldsymbol{C_{15}}$ and $\boldsymbol{D_{15}}$, respectively, by one left shift. In all cases, by a single left shift is meant a rotation of the bits one place to the left, so that after one left shift the bits in the 28 positions are the bits that were previously in positions 2, 3,..., 28, 1.

Example: From original pair pair $\boldsymbol{C_0}$ and $\boldsymbol{D_0}$ we obtain:

$\boldsymbol{C_0}$ = 1111000011001100101010101111
$\boldsymbol{D_0}$ = 0101010101100110011110001111

$\boldsymbol{C_1}$ = 1110000110011001010101011111
$\boldsymbol{D_1}$ = 1010101011001100111100011110

$\boldsymbol{C_2}$ = 1100001100110010101010111111
$\boldsymbol{D_2}$ = 0101010110011001111000111101

$\boldsymbol{C_3}$ = 0000110011001010101011111111
$\boldsymbol{D_3}$ = 0101011001100111100011110101

$\boldsymbol{C_4}$ = 0011001100101010101111111100
$\boldsymbol{D_4}$ = 0101100110011110001111010101

$\boldsymbol{C_5}$ = 1100110010101010111111110000
$\boldsymbol{D_5}$ = 0110011001111000111101010101

$\boldsymbol{C_6}$ = 0011001010101011111111000011
$\boldsymbol{D_6}$ = 1001100111100011110101010101

$\boldsymbol{C_7}$ = 1100101010101111111100001100
$\boldsymbol{D_7}$ = 0110011110001111010101010110

$\boldsymbol{C_8}$ = 0010101010111111110000110011
$\boldsymbol{D_8}$ = 1001111000111101010101011001

$\boldsymbol{C_9}$ = 0101010101111111100001100110
$\boldsymbol{D_9}$ = 0011110001111010101010110011

$\boldsymbol{C_{10}}$ = 0101010111111110000110011001
$\boldsymbol{D_{10}}$ = 1111000111101010101011001100

$\boldsymbol{C_{11}}$ = 0101011111111000011001100101
$\boldsymbol{D_{11}}$ = 1100011110101010101100110011

C_{12} = 0101111111100001100110010101
D_{12} = 0001111010101010110011001111

C_{13} = 0111111110000110011001010101
D_{13} = 0111101010101011001100111100

C_{14} = 1111111000011001100101010101
D_{14} = 1110101010101100110011110001

C_{15} = 1111100001100110010101010111
D_{15} = 1010101010110011001111000111

C_{16} = 1111000011001100101010101111
D_{16} = 0101010101100110011110001111

We now form the keys K_n, for 1<=n<=16, by applying the following permutation table to each of the concatenated pairs C_nD_n. Each pair has 56 bits, but **PC-2** only uses 48 of these.

PC-2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Therefore, the first bit of K_n is the 14th bit of C_nD_n, the second bit the 17th, and so on, ending with the 48th bit of K_n being the 32th bit of C_nD_n.

Example: For the first key we have C_1D_1 = 1110000 1100110 0101010 1011111 1010101 0110011 0011110 0011110

which, after we apply the permutation **PC-2**, becomes

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010

For the other keys we have

K_2 = 011110 011010 111011 011001 110110 111100 100111 100101
K_3 = 010101 011111 110010 001010 010000 101100 111110 011001
K_4 = 011100 101010 110111 010110 110110 110011 010100 011101
K_5 = 011111 001110 110000 000111 111010 110101 001110 101000
K_6 = 011000 111010 010100 111110 010100 000111 101100 101111
K_7 = 111011 001000 010010 110111 111101 100001 100010 111100
K_8 = 111101 111000 101000 111010 110000 010011 101111 111011
K_9 = 111000 001101 101111 101011 111011 011110 011110 000001
K_{10} = 101100 011111 001101 000111 101110 100100 011001 001111
K_{11} = 001000 010101 111111 010011 110111 101101 001110 000110
K_{12} = 011101 010111 000111 110101 100101 000110 011111 101001
K_{13} = 100101 111100 010111 010001 111110 101011 101001 000001
K_{14} = 010111 110100 001110 110111 111100 101110 011100 111010
K_{15} = 101111 111001 000110 001101 001111 010011 111100 001010
K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101

So much for the subkeys. Now we look at the message itself.

Step 2: Encode each 64-bit block of data.

There is an *initial permutation* **IP** of the 64 bits of the message data **M**. This rearranges the bits according to the following table, where the entries in the table show the new arrangement of the bits from their initial order. The 58th bit of **M** becomes the first bit of **IP**. The 50th bit of **M** becomes the second bit of **IP**. The 7th bit of **M** is the last bit of **IP**.

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Example: Applying the initial permutation to the block of text **M**, given previously, we get

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
IP = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

Here the 58th bit of **M** is "1", which becomes the first bit of **IP**. The 50th bit of **M** is "1", which becomes the second bit of **IP**. The 7th bit of **M** is "0", which becomes the last bit of **IP**.

Next divide the permuted block **IP** into a left half $\boldsymbol{L_0}$ of 32 bits, and a right half $\boldsymbol{R_0}$ of 32 bits.

Example: From **IP**, we get $\boldsymbol{L_0}$ and $\boldsymbol{R_0}$

$\boldsymbol{L_0}$ = 1100 1100 0000 0000 1100 1100 1111 1111
$\boldsymbol{R_0}$ = 1111 0000 1010 1010 1111 0000 1010 1010

We now proceed through 16 iterations, for 1<=***n***<=16, using a function ***f*** which operates on two blocks--a data block of 32 bits and a key $\boldsymbol{K_n}$ of 48 bits--to produce a block of 32 bits. **Let + denote XOR addition, (bit-by-bit addition modulo 2)**. Then for **n** going from 1 to 16 we calculate

$$L_n = R_{n-1}$$
$$R_n = L_{n-1} + f(R_{n-1}, K_n)$$

This results in a final block, for $n = 16$, of $L_{16}R_{16}$. That is, in each iteration, we take the right 32 bits of the previous result and make them the left 32 bits of the current step. For the right 32 bits in the current step, we XOR the left 32 bits of the previous step with the calculation f .

Example: For $n = 1$, we have

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010
$L_1 = R_0$ = 1111 0000 1010 1010 1111 0000 1010 1010
$R_1 = L_0 + f(R_0, K_1)$

It remains to explain how the function f works. To calculate f, we first expand each block R_{n-1} from 32 bits to 48 bits. This is done by using a selection table that repeats some of the bits in R_{n-1} . We'll call the use of this selection table the function **E**. Thus **E**(R_{n-1}) has a 32 bit input block, and a 48 bit output block.

Let **E** be such that the 48 bits of its output, written as 8 blocks of 6 bits each, are obtained by selecting the bits in its inputs in order according to the following table:

E BIT-SELECTION TABLE

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Thus the first three bits of $\mathbf{E}(\boldsymbol{R_{n-1}})$ are the bits in positions 32, 1 and 2 of $\boldsymbol{R_{n-1}}$ while the last 2 bits of $\mathbf{E}(\boldsymbol{R_{n-1}})$ are the bits in positions 32 and 1.

Example: We calculate $\mathbf{E}(\boldsymbol{R_0})$ from $\boldsymbol{R_0}$ as follows:

$\boldsymbol{R_0}$ = 1111 0000 1010 1010 1111 0000 1010 1010
$\mathbf{E}(\boldsymbol{R_0})$ = 011110 100001 010101 010101 011110 100001 010101 010101

(Note that each block of 4 original bits has been expanded to a block of 6 output bits.)

Next in the $\boldsymbol{f}$ calculation, we XOR the output $\mathbf{E}(\boldsymbol{R_{n-1}})$ with the key $\boldsymbol{K_n}$:

$$\boldsymbol{K_n} + \mathbf{E}(\boldsymbol{R_{n-1}}).$$

Example: For $\boldsymbol{K_1}$, $\mathbf{E}(\boldsymbol{R_0})$, we have

$\boldsymbol{K_1}$ = 000110 110000 001011 101111 111111 000111 000001 110010
$\mathbf{E}(\boldsymbol{R_0})$ = 011110 100001 010101 010101 011110 100001 010101 010101
$\boldsymbol{K_1}$+$\mathbf{E}(\boldsymbol{R_0})$ = 011000 010001 011110 111010 100001 100110 010100 100111.

We have not yet finished calculating the function $\boldsymbol{f}$. To this point we have expanded $\boldsymbol{R_{n-1}}$ from 32 bits to 48 bits, using the selection table, and XORed the result with the key $\boldsymbol{K_n}$. We now have 48 bits, or eight groups of six bits. We now do something strange with each group of six bits: we use them as addresses in tables called "**S boxes**". Each group of six bits will give us an address in a different **S** box. Located at that address will be a 4 bit number. This 4 bit number will replace the original 6 bits. The net result is that the eight groups of 6 bits are transformed into eight groups of 4 bits (the 4-bit outputs from the **S** boxes) for 32 bits total.

Write the previous result, which is 48 bits, in the form:

$$\boldsymbol{K_n} + \mathbf{E}(\boldsymbol{R_{n-1}}) = \boldsymbol{B_1B_2B_3B_4B_5B_6B_7B_8},$$

where each $\boldsymbol{B_i}$ is a group of six bits. We now calculate

$$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$$

where $S_i(B_i)$ referres to the output of the ***i***-th **S** box.

To repeat, each of the functions ***S1, S2,..., S8***, takes a 6-bit block as input and yields a 4-bit block as output. The table to determine S_1 is shown and explained below:

S1

Column Number

Row No.	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**14**	**4**	**13**	**1**	**2**	**15**	**11**	**8**	**3**	**10**	**6**	**12**	**5**	**9**	**0**	**7**
1	**0**	**15**	**7**	**4**	**14**	**2**	**13**	**1**	**10**	**6**	**12**	**11**	**9**	**5**	**3**	**8**
2	**4**	**1**	**14**	**8**	**13**	**6**	**2**	**11**	**15**	**12**	**9**	**7**	**3**	**10**	**5**	**0**
3	**15**	**12**	**8**	**2**	**4**	**9**	**1**	**7**	**5**	**11**	**3**	**14**	**10**	**0**	**6**	**13**

If S_1 is the function defined in this table and ***B*** is a block of 6 bits, then $S_1(B)$ is determined as follows: The first and last bits of ***B*** represent in base 2 a number in the decimal range 0 to 3 (or binary 00 to 11). Let that number be ***i***. The middle 4 bits of ***B*** represent in base 2 a number in the decimal range 0 to 15 (binary 0000 to 1111). Let that number be ***j***. Look up in the table the number in the ***i***-th row and ***j***-th column. It is a number in the range 0 to 15 and is uniquely represented by a 4 bit block. That block is the output $S_1(B)$ of S_1 for the input ***B***. For example, for input block ***B*** = 011011 the first bit is "0" and the last bit "1" giving 01 as the row. This is row 1. The middle four bits are "1101". This is the binary equivalent of decimal 13, so the column is column number 13. In row 1, column 13 appears 5. This determines the output; 5 is binary 0101, so that the output is 0101. Hence S_1(011011) = 0101.

The tables defining the functions $S_1, \ldots, S_8$ are the following:

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Example: For the first round, we obtain as the output of the eight **S** boxes:

$\boldsymbol{K_1}$ + **E**($\boldsymbol{R_0}$) = 011000 010001 011110 111010 100001 100110 010100 100111.

$\boldsymbol{S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)}$ = 0101 1100 1000 0010 1011 0101 1001 0111

The final stage in the calculation of $\boldsymbol{f}$ is to do a permutation **P** of the **S**-box output to obtain the final value of $\boldsymbol{f}$:

$$\boldsymbol{f} = \mathbf{P}(\boldsymbol{S_1(B_1)S_2(B_2)...S_8(B_8)})$$

The permutation **P** is defined in the following table. **P** yields a 32-bit output from a 32-bit input by permuting the bits of the input block.

P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Example: From the output of the eight **S** boxes:

$$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8) = 0101\ 1100\ 1000\ 0010\ 1011\ 0101\ 1001\ 0111$$

we get

$$f = 0010\ 0011\ 0100\ 1010\ 1010\ 1001\ 1011\ 1011$$

$R_1 = L_0 + f(R_0 , K_1)$

= 1100 1100 0000 0000 1100 1100 1111 1111
+ 0010 0011 0100 1010 1010 1001 1011 1011
= 1110 1111 0100 1010 0110 0101 0100 0100

In the next round, we will have $L_2 = R_1$, which is the block we just calculated, and then we must calculate $R_2 = L_1 + f(R_1, K_2)$, and so on for 16 rounds. At the end of the sixteenth round we have the blocks L_{16} and R_{16}. We then ***reverse*** the order of the two blocks into the 64-bit block

$$R_{16}L_{16}$$

and apply a final permutation $\mathbf{IP^{-1}}$ as defined by the following table:

IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

That is, the output of the algorithm has bit 40 of the preoutput block as its first bit, bit 8 as its second bit, and so on, until bit 25 of the preoutput block is the last bit of the output.

Example: If we process all 16 blocks using the method defined previously, we get, on the 16th round,

$\boldsymbol{L_{16}}$ = 0100 0011 0100 0010 0011 0010 0011 0100
$\boldsymbol{R_{16}}$ = 0000 1010 0100 1100 1101 1001 1001 0101

We reverse the order of these two blocks and apply the final permutation to

$\boldsymbol{R_{16}L_{16}}$ = 00001010 01001100 11011001 10010101 01000011 01000010 00110010 00110100

$\boldsymbol{IP^{-1}}$ = 10000101 11101000 00010011 01010100 00001111 00001010 10110100 00000101

which in hexadecimal format is

85E813540F0AB405.

This is the encrypted form of **M** = 0123456789ABCDEF: namely, **C** = 85E813540F0AB405.

Decryption is simply the inverse of encryption, follwing the same steps as above, but reversing the order in which the subkeys are applied.

DES Modes of Operation:

The DES algorithm turns a 64-bit message block **M** into a 64-bit cipher block **C**. If each 64-bit block is encrypted individually, then the mode of encryption is called ***Electronic Code Book*** (ECB) mode. There are two other modes of DES encryption, namely ***Chain Block Coding*** (CBC) and ***Cipher Feedback*** (CFB), which make each cipher block dependent on all the previous messages blocks through an initial XOR operation.

Cracking DES

Before DES was adopted as a national standard, during the period NBS was soliciting comments on the proposed algorithm, the creators of public key cryptography, Martin Hellman and Whitfield Diffie, registered some objections to the use of DES as an encryption algorithm. Hellman wrote: "Whit Diffie and I have become concerned that the proposed data encryption standard, while probably secure against commercial assault, may be extremely vulnerable to attack by an intelligence organization" (letter to NBS, October 22, 1975).

Diffie and Hellman then outlined a "brute force" attack on DES. (By "brute force" is meant that you try as many of the 2^56 possible keys as you have to before decrypting the ciphertext into a sensible plaintext message.) They proposed a special purpose "parallel computer using one million chips to try one million keys each" per second, and estimated the cost of such a machine at $20 million.

Fast forward to 1998. Under the direction of John Gilmore of the EFF, a team spent $220,000 and built a machine that can go through the entire 56-bit DES key space in an average of 4.5 days. On July 17, 1998, they announced they had cracked a 56-bit key in 56 hours. The computer, called Deep Crack, uses 27 boards each containing 64 chips, and is capable of testing 90 billion keys a second.

Despite this, as recently as June 8, 1998, Robert Litt, principal associate deputy attorney general at the Department of Justice, denied it was possible for the FBI to crack DES: "Let me put the technical problem in context: It took 14,000 Pentium computers working for four months to decrypt a single message . . . . We are not just talking FBI and NSA [needing massive computing power], we are talking about every police department."

Responded cryptograpy expert Bruce Schneier: " . . . the FBI is either incompetent or lying, or both." Schneier went on to say: "The only solution here is to pick an algorithm with a longer key; there isn't enough silicon in the galaxy or enough time before the sun burns out to brute- force triple-DES" (*Crypto-Gram*, Counterpane Systems, August 15, 1998).

3. Works Description

3.1 System Overview:

The voting system that exists in Bangladesh right now it is completely paper based that is from gathering the voter details to voting everything are done by hand and all kind of data are written on paper, so there are lots of chances of mistakes and corruption. People become voter in two different areas if people change their residence and become voter in the new area. People can even vote for multiple times because the way the authority follows to make sure a single vote for a single person is really an old style. They just mark a voter's nail with black ink that is hard to remove but possible so people can vote again. Even a dead person can even become a voter in this system. During the registration if the authority doesn't remove the details of a dead person from the list out of mistake or corruption he/she remains in the list as an existing voter. Even it has been seen that false voter cast their vote with false name in this system.

So it was the demand of time to come out with something new which will help to reduce this type of mistakes and corruptions from so sensitive national issue like election. That's why e-voting is being proposed as a step against corruption.

The most interesting thing with this system is automated registration of the voters. People will be enlisted in the population database for three different age groups those are 0 years to less than 17 years, 17 years to less than 18 years, and 18 years and more. People will be migrated to different database of age groups automatically according to their age. Finally they will be migrated to the voter database if they give their details and finger print to the election authority when they will be 17. After being a part of the voter database a person will be able to vote himself or herself. There is no way to vote one person on behalf of others as finger print will be used to detect a person. People who have died already will be deleted from voter database but they will be still in the population database in his/her age group but the status will be dead.

Wide area network will be used to design the whole network system. There will be both radio linked and optical fiber network. There will be one client server and three application server. From all the client servers to the application servers there will be radio linked network as the client server will be busy just in sending the encrypted votes to one of the application server through some other application server. The third application server will send them again to the main database through another application server. There will be optical fiber connection between the second application server to the third application server and from third to the main database.

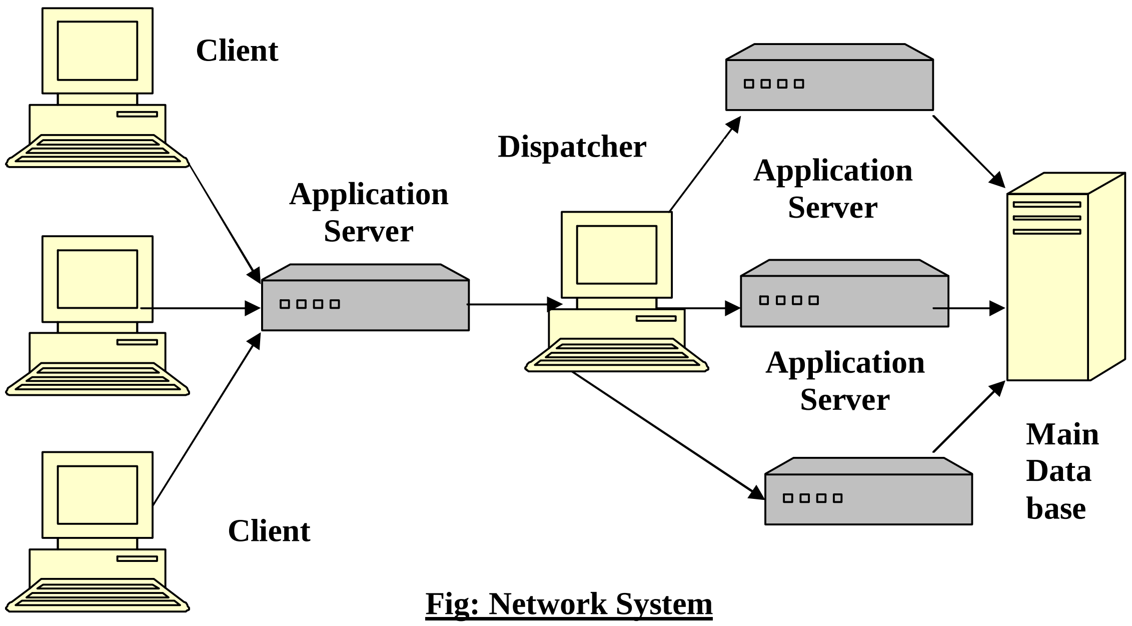
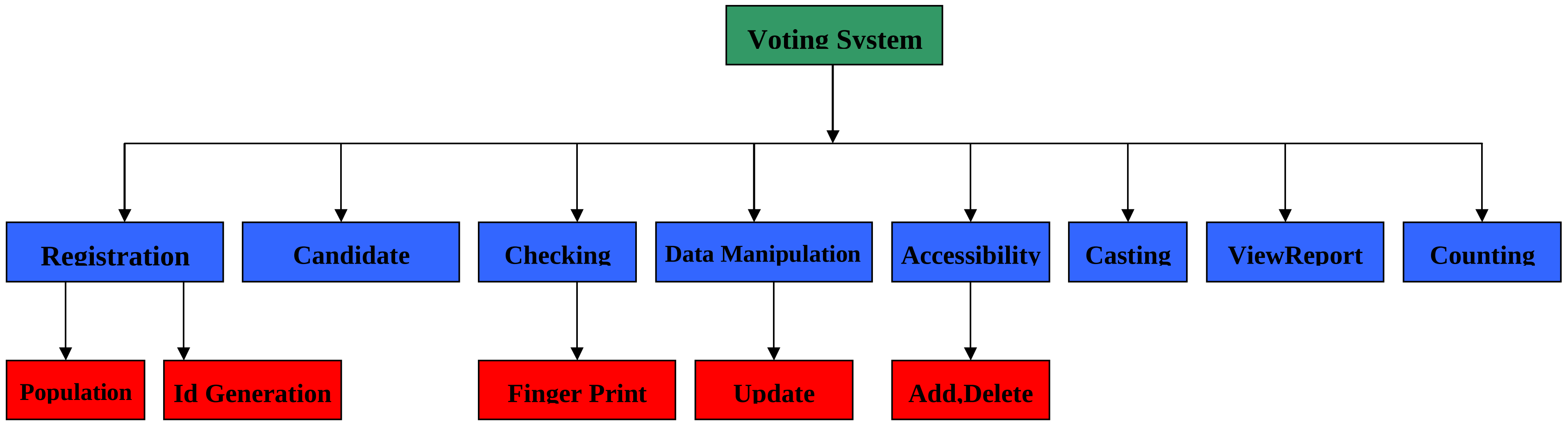


Fig: Network System

The e-voting system proposed here have been decomposed into several functions. The function specifies what the system does by describing its work. What inputs are fetched to the system, what outputs are produced and data manipulation performed by the system. It is better idea to divide the systems into some sub systems to have a better managed software development also improves reusable capability and make easier for maintenance. Each sub-system will have a well defined interface with respect to the rest of the system.

Decomposition of The System



3.2 Decomposition of The System:

As it has been said before that the e-voting system will have some sub-systems which are described as follows.

3.2.1 Registration:

Here registration system has been proposed to be automated. Two different databases are going to be used to make this system to be automated. One is population database and another is voter database. People will be enlisted there in the population database in two different age groups which we mentioned before and they will be migrated from one age group to another according to their age. Voter database will contain only the people who are existing voters. To migrate to this database from the population database people have to give their details and finger prints to the authority when they will be 17. The data and the fingerprint will be collected at the age 17 so that if any person becomes 18 years old between the time of collecting data and the election he or she will be able to attend the poll.

3.2.2 Candidate:

Candidate is an important process in this system. A candidate can participate in the poll from a party or he can compete alone. On the polling session each candidate will have a sign on the touch pad where the voters give their opinion. The final result depends on the result of each candidate since the system has to keep an eye on the candidate's result of each area.

3.2.3 Checking:

During registration when the fingerprint of the voters will be collected then a hash function will be generated for each finger print. Each voters detail will be laid under this hash function. When a vote will be cast the system will read the packet with the hash function and will check the constitute then the system will check whether the id is blocked or not. If it is not blocked then the vote will be counted otherwise not.

3.2.4 Data Manipulation:

All the data should be manipulated or updated to maintain a good database management. When a baby born his/her details direct goes to the database from organization 'Y' and it is added to the primary database. Again when a person dies his/her details also go to the population database from organization 'X' and his status becomes death and not been migrated to any other age group or in the voter database. If the person is added to the voter database he/she will be deleted from there but he/she will be still there in population database.

3.2.5 Accessibility:

The election commission authority is not getting the supreme power to do whatever they want to do with the voter list or the voter details. They have the accessibility to the details but any kind of modifying or changing they will need the finger print of that voter.

3.2.6 Casting:

The id of a voter will be blocked just after casting his/her vote, he/she can vote again but it won't be counted. So it is ensured that only one vote will be cast for a single voter. The voter will cast a vote by giving his identification with his/her finger print and vote on a touch screen.

3.2.7 Counting:

The votes will be counted constitute wise. After finishing the voting session the votes stored constitute wise will be counted. Each constitute will pick only the votes it wont think about the id. It will just pick the votes of each sign and then count them.

3.2.8 View Report:

Report will show which candidate sign got how many votes from which constitute. Main database have the name of the candidates so the system will be able to publish the result for each candidate. From this result the system will also publish the final result that which sign win getting how many constitute.

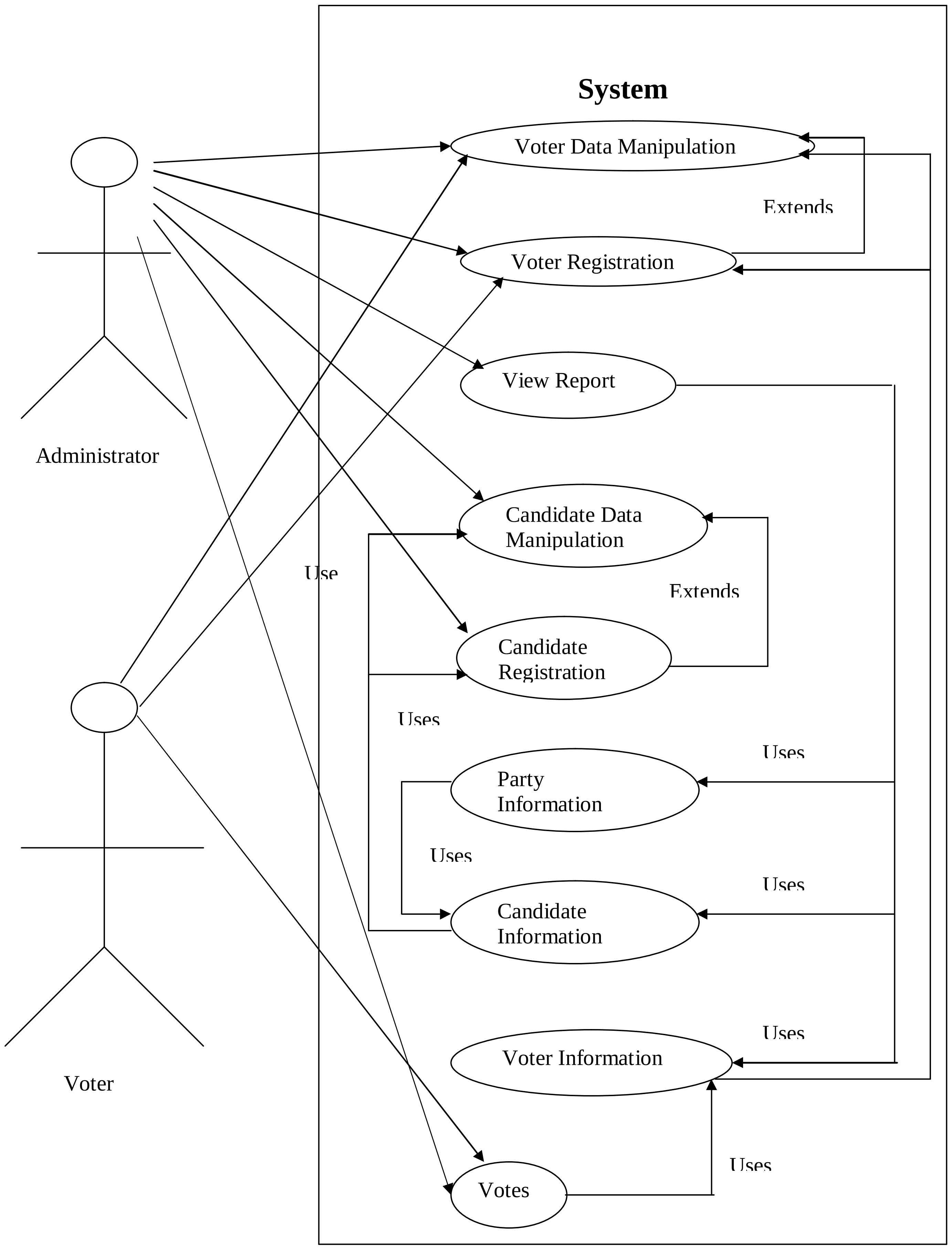


Fig: Use Case Diagram

3.3 Analysis of Use Case Digram:

The system proposed here for electronic voting has two active actors. One is the administrator and another one is the voter.

The administrator has the supreme power to manipulate the voter details even the candidate details. The main responsibility of the administrator is to registration of the voters. Administrator is authorized to view the details of the candidate and also voters although he is not authorized for modifying the details without the fingerprint of the voter or the candidate.

An administrator is allowed to view all the reports like candidate result, area result or even political party result. He is also responsible for publishing the result.

Voter is responsible for only casting the vote. If there comes any change in the voter details it is voter's responsibility to go to the authority and let them know so that the officials can modify the details taking the finger prints of that voter.

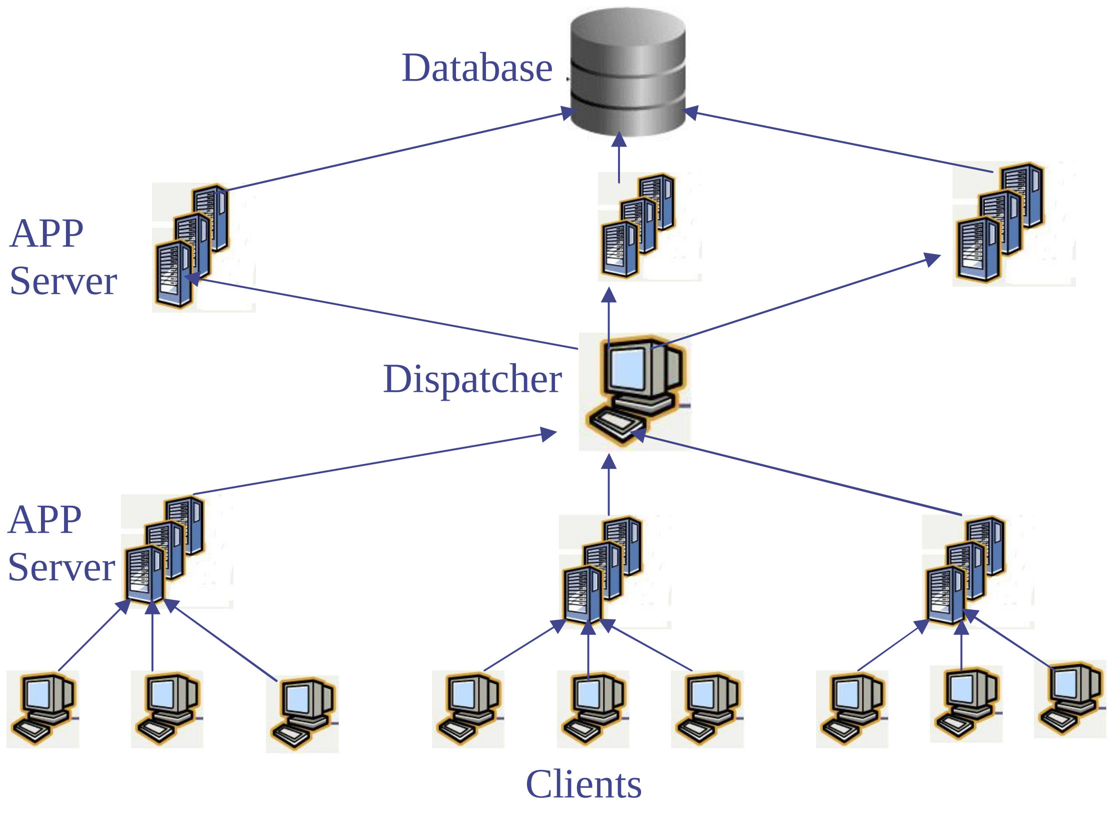


Fig: Network System Structure

3.4 Network System Structure:

A three tiered network system has been proposed here for implementation of this electronic voting system. There will be a number of clients in the most root level that is it may be in Police Station level or sub-district level of a country. But it is necessary to have a lot of clients in the root level.

In the district level there will be a dedicated application server for those police station or sub-district clients under that district. These clients together make a cluster. These application servers in this level won't accept packets from any other cluster under another application server.

In the division level there will be some dispatcher for each division. These dispatchers will also be dedicated for those districts under that division. There will be a layer of application server layer after the dispatcher through which the dispatcher will pass the encrypted vote to the main database.

Main database will have several segments for each constitute. The encrypted vote will be checked first to see whether the id of the voter is locked or not. If it is locked then it won't be passed but if it is not locked then the vote will be sent to that segment reserved for that seat.

The client's responsibility is to receive the finger print and the vote and then encrypt them. Afterward it will send the packet to the application server through radio linked network. These dedicated application servers will pass them through another radio linked network to the dispatcher. Up to this radio linked network is being used as the network will be engaged for only sending packets and a radio link network provides 4 kbps speed which is good enough for this purpose.

Dispatcher's responsibility is to search the next application server which is not busy right then. So the more application server the system would have in this level the faster it will work. The encrypted vote will be passed to that server and after that it will be directly passed to main database. Here from dispatcher to the main database a fiber optic network will be used as the dispatcher has to handle millions of packets at a time so it will need a better paced as well as secured network system.

4. System Design

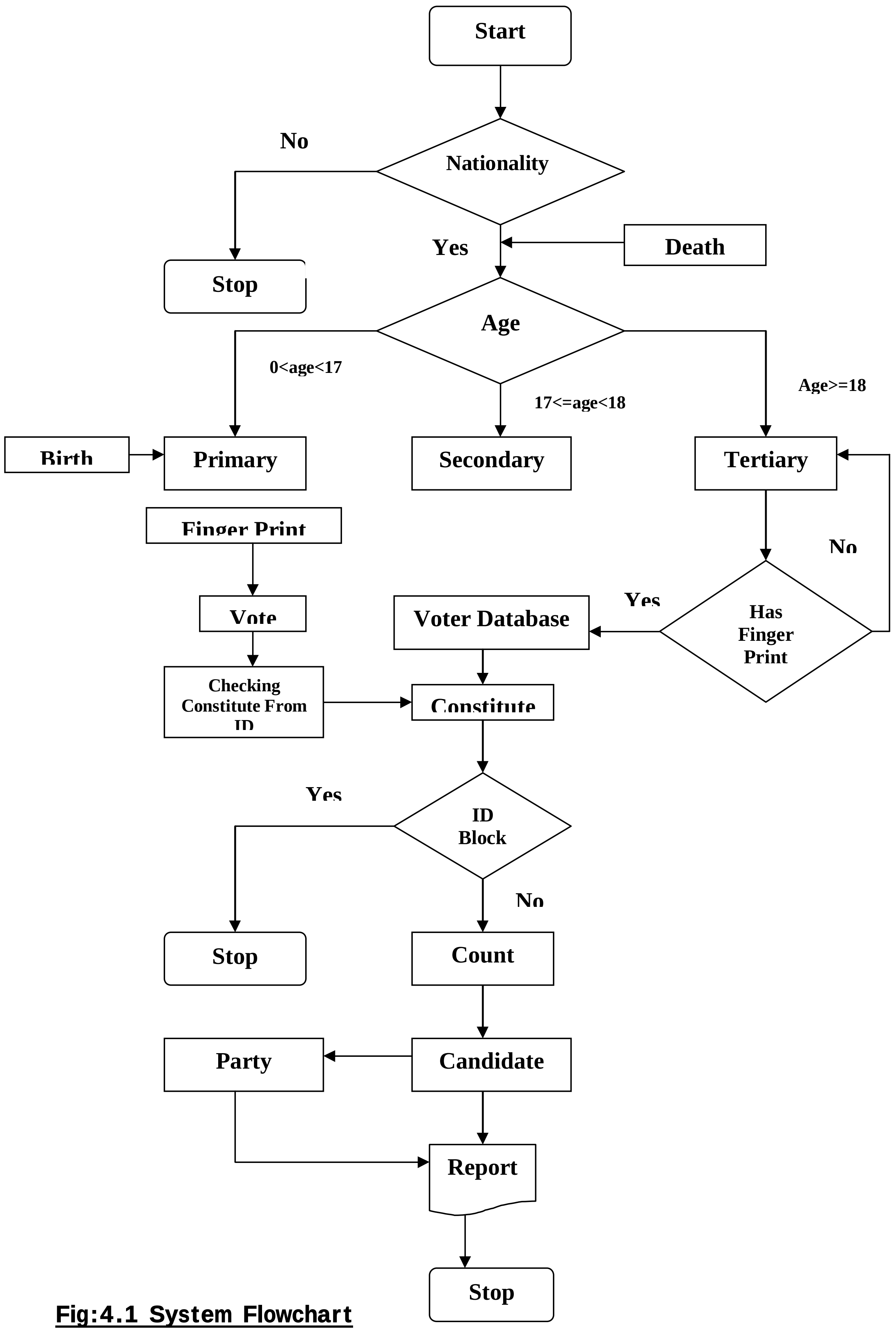


Fig:4.1 System Flowchart

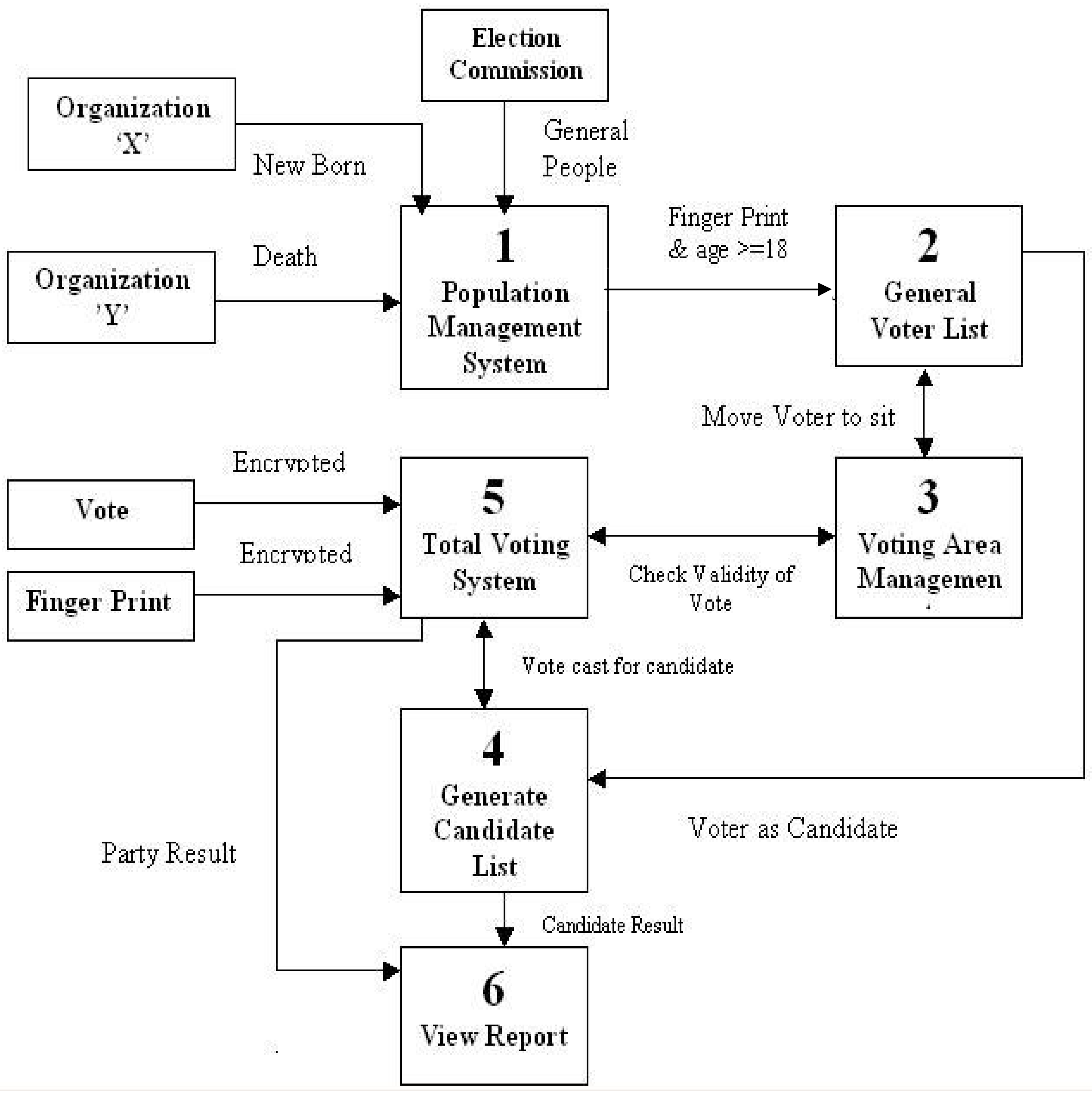


Fig:4.2 Data Flow Diagram

4.1 Analysis of Data Flow Diagram:

According to the dataflow diagram we see 6 processes are here with several inputs and outputs for different purposes. Process 1 is for population management system. It has three inputs one is for new born baby that comes from the organization 'X' another one is for the information about the people who has already passed away that comes from the organization 'Y'. The election commission authority provides another input for the population management system about the general people.

Process 2 that is general voter list has one input from the process 1 that is about the people who are living voters of the country and who are registered with both their finger print and details. These voters are sorted according to their area.

Process 3 that is voting area management deals with the voters under each area or seat. The responsibility of this process doesn't limited to only accepting the voters under that area from process 2 but also check the validity of the vote given by each participant.

Process 4 supervises the voting procedure that is keeping track of votes for each candidate. It also provides result to the process 6 which's responsibility is to show result for each candidate as well as each party attending the polling session.

Process 5 engages with getting the encrypted packets containing the vote and the finger print and subsequently processing them that is checking the validity, counting vote for candidate, counting prevailed seat for each party, sending them to process 4 as well as to process 6.

Process 6 as said before keeps busy itself to publish the result for each candidate as well as each party.

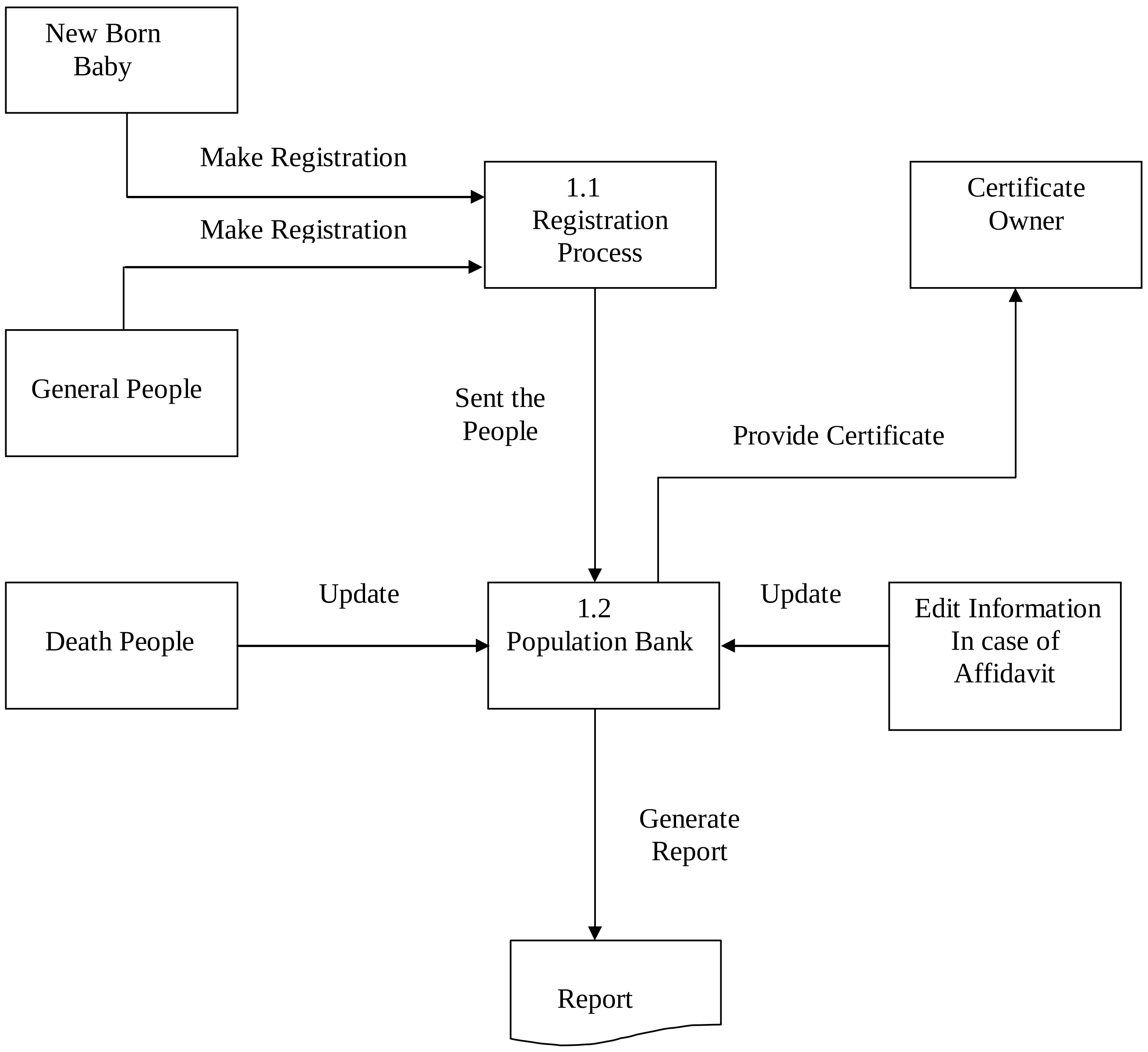


Fig: 4.3 Level 1 Diagram

4.2 Analysis of Level1 Diagram:

When we decompose the process 1 in data flow diagram we get the level 1 diagram. Analyzing the diagram we see there are two sub-systems in this stage.

First one that is registration process gets the new born list and general people list and record them.

Next process that is population bank gets the record of death people and eliminates them from the voter list. It also updates the voter name if any voter affidavits his name and inform the authority. The final report given by this process is the existing voter list.

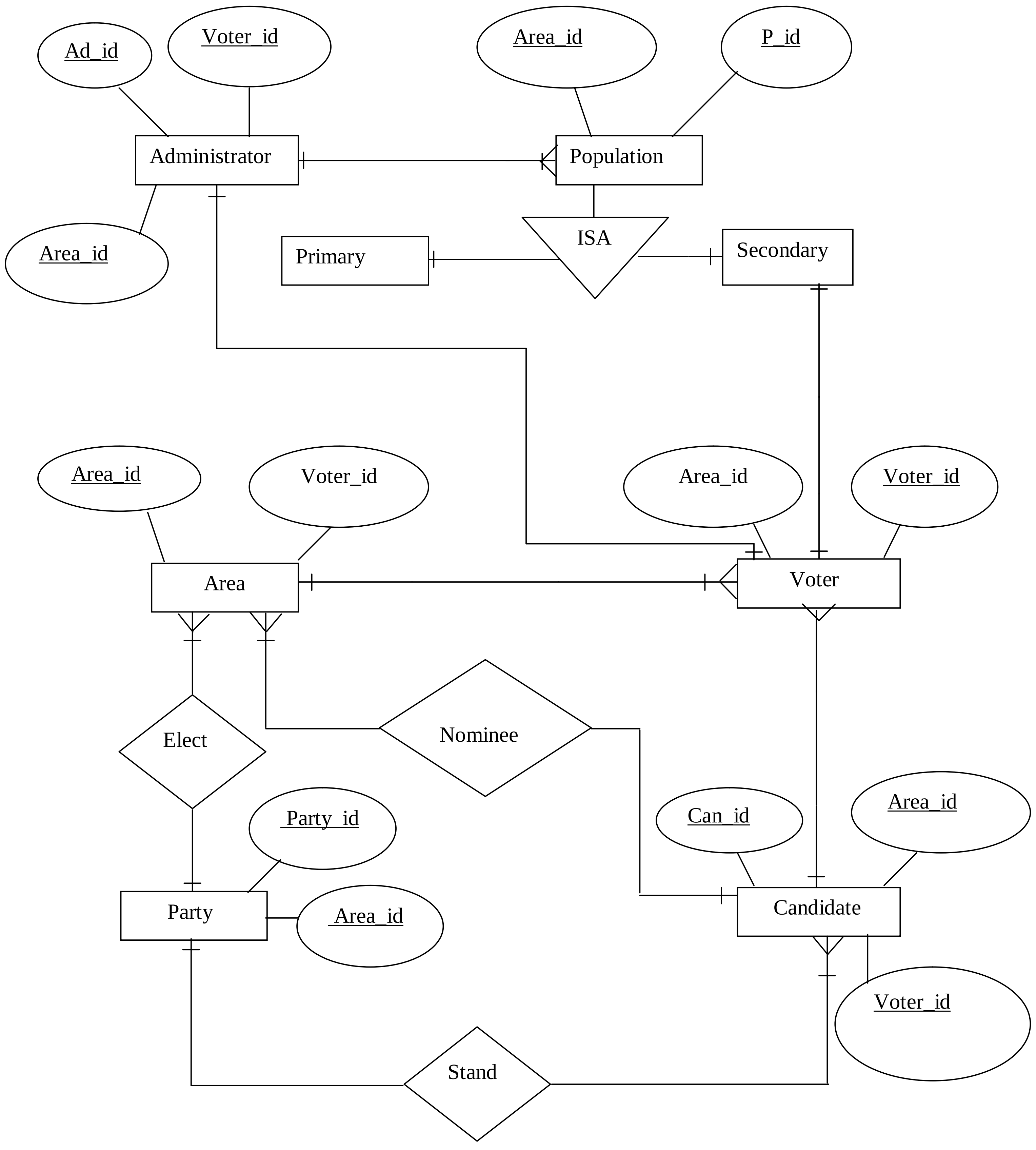


Fig: 4.4 Entity Relationship Diagram for Registration

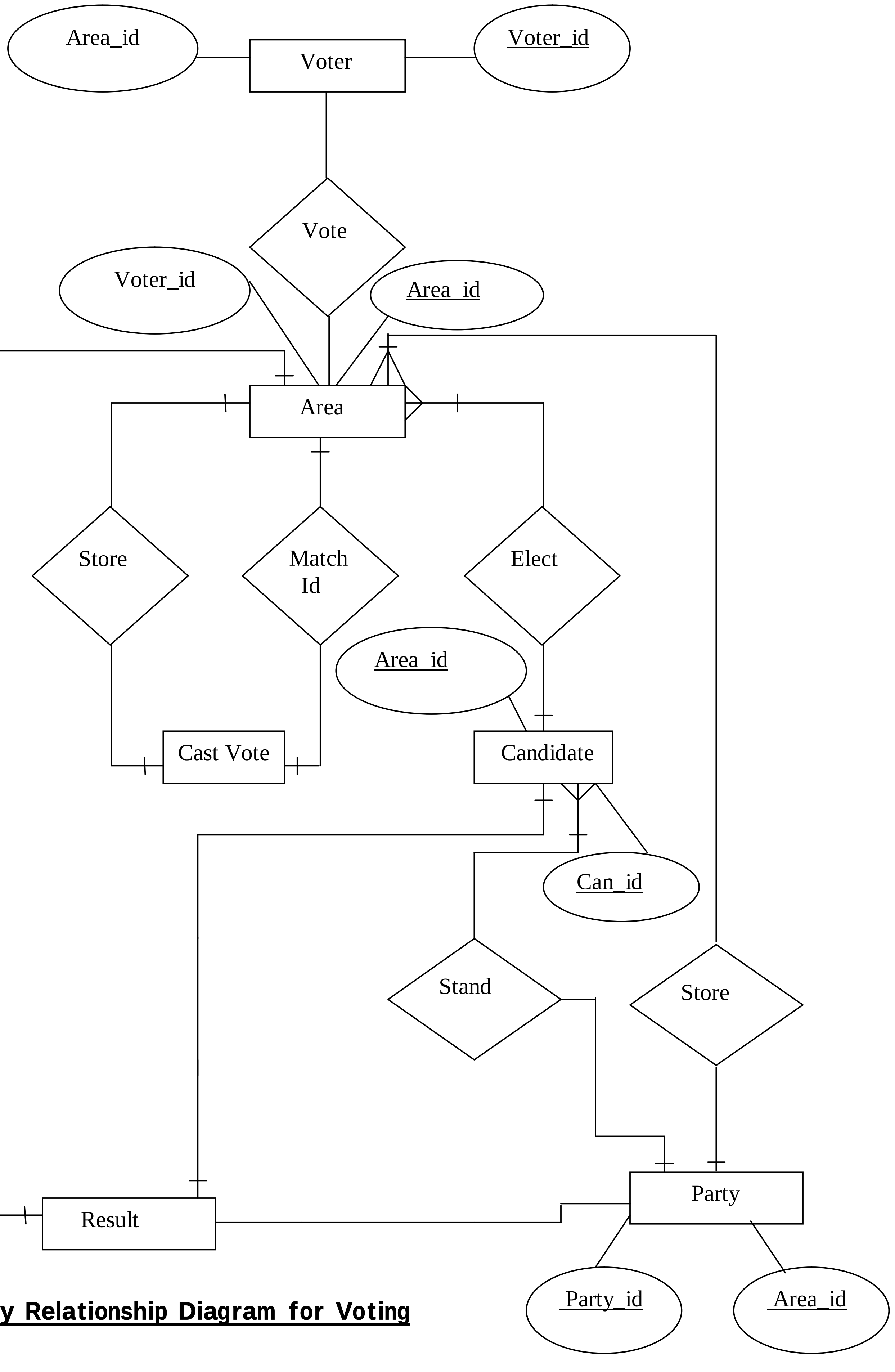


Fig: 4.5 Entity Relationship Diagram for Voting

4.3 Data Base Schema for Electronic Voting System

ADMIN
Ad_sex
Ad_name
Ad_father_name
Ad_mother_name
Ad_picture
Ad_present_address
Ad_parmanet_address
Ad_blood_group
Area_id
Ad_occupation
Ad_past_histry
Ad_age
Ad_id
Voter_id

POPULATION
P_sex
P_name
P_father_name
P_mother_name
P_picture
P_present_address
P_parmanet_address
P_blood_group
Area_id
P_occupation
P_age
P_id

VOTER
Voter_sex
Voter_name
Voter_father_name
Voter_mother_name
Voter_picture
Voter_present_address
Voter_parmanet_address
Voter_blood_group
Area_id
Voter_occupation
Voter_age
Voter_id

AREA
Area_name
Area_id
Area_population
Area_total_voter
Area_division
Area_total_vote
Can_id
Area_total_casting_vote
Voter_id
Ad_id
Party_id

PARTY
Party_name
Party_id
Party_logo_name
Party_total_candidate
Can_id
Party_total_vote
Party_past_histry
Area_id

CANDIDATE
Can_sex
Can_name
Can_father_name
Can_mother_name
Can_picture
Can_present_address
Can_parmanet_address
Can_blood_group
Area_id
Can_occupation
Can_past_histry
Can_age
Can_id
Can_party
Party_id
Voter_id

DEATH
Death_sex
Death_name
Death_father_name
Death_mother_name
Death_picture
Death_present_address
Death_parmanet_address
Death_blood_group
Area_id
Death_occupation
Death_age
Death_id
Voter_id

4.4 Analysis of Entity Relationship Diagram:

In the entity relationship diagram the relationship between different entities has been showed. Important attributes are also given here, among them the primary keys and the foreign keys are underlined. All the attributes are not given here because of the shortage of space.

4.4.1 Administrator:

Administrator has three attributes given here Ad_id, Voter_id and Area_id. It has a one to many relationships with the entity population because a numerous number of people are under one administrator. In addition to it has another one to many relationships with the entity Voter as thousands of voters are also related to one administrator.

- 4.4.1.1 Ad_id:

It contains the id that is only for the people who are working as administrator.

- 4.4.1.2 Voter_id:

Identification number of the administrator as a voter since an administrator is also a voter.

- 4.4.1.3 Area_id:

Each election area has an id and here the id is for which area the administrator is working for.

4.4.2 Population:

Population has two attributes P_id and Area_id. It has a many to one relationship with the entity Administrator which has already been discussed. It has two segments one is primary another one is secondary. A people will be in either in primary or in secondary database. Population has one to one relationship with both the entity.

- 4.4.2.1 P_id:
  It is an unique id for all types of people, voter or non voter.

- 4.4.2.2 Area_id:
  This id represents the area of the person.

4.4.3 Voter:

Voter has two attributes one is Voter_id another is Area_id. It has two one to one relationship with the entity Secondary and Administrator. It has also a many to one relationship with the entity Area.

- 4.4.3.1 Voter_id:
  It represents the id of voter.

- 4.4.3.2 Area_id:
  It stands for the area of the voter.

4.4.4 Area:

Area has to attributes Area_id and Voter_id. It has two many to one relationship with both the party and the candidate. As many political parties can compete in the election for the same area and there can be a number of candidates.

- 4.4.4.1 Area_id:
  Presents each area as each area has an unique id.

- 4.4.4.2 Voter_id:
  Stands for id of all the voters are there in that area.

4.4.5 Party:

Party has two attributes Area_id and Party_id. It has two one to many relationships with both the entity area and candidate. Relationship with the area has been conversed before. A party can give nomination a candidate for each polling area. So the number of candidates can be the same as the polling area for each party.

- 4.4.5.1 Area_id:
  These are the ids where a party has its candidates.

- 4.4.5.2 Party_id:
  This id represents the identity of each party.

4.4.6 Candidate:

Candidate has three attributes those are Can_id, Area_id, Voter_id. It has a one to many relationships with the voter and the area. It has also a many to one relationship with party as discussed before.

- 4.4.6.1 Can_id:
  Stands for each candidate's identification.

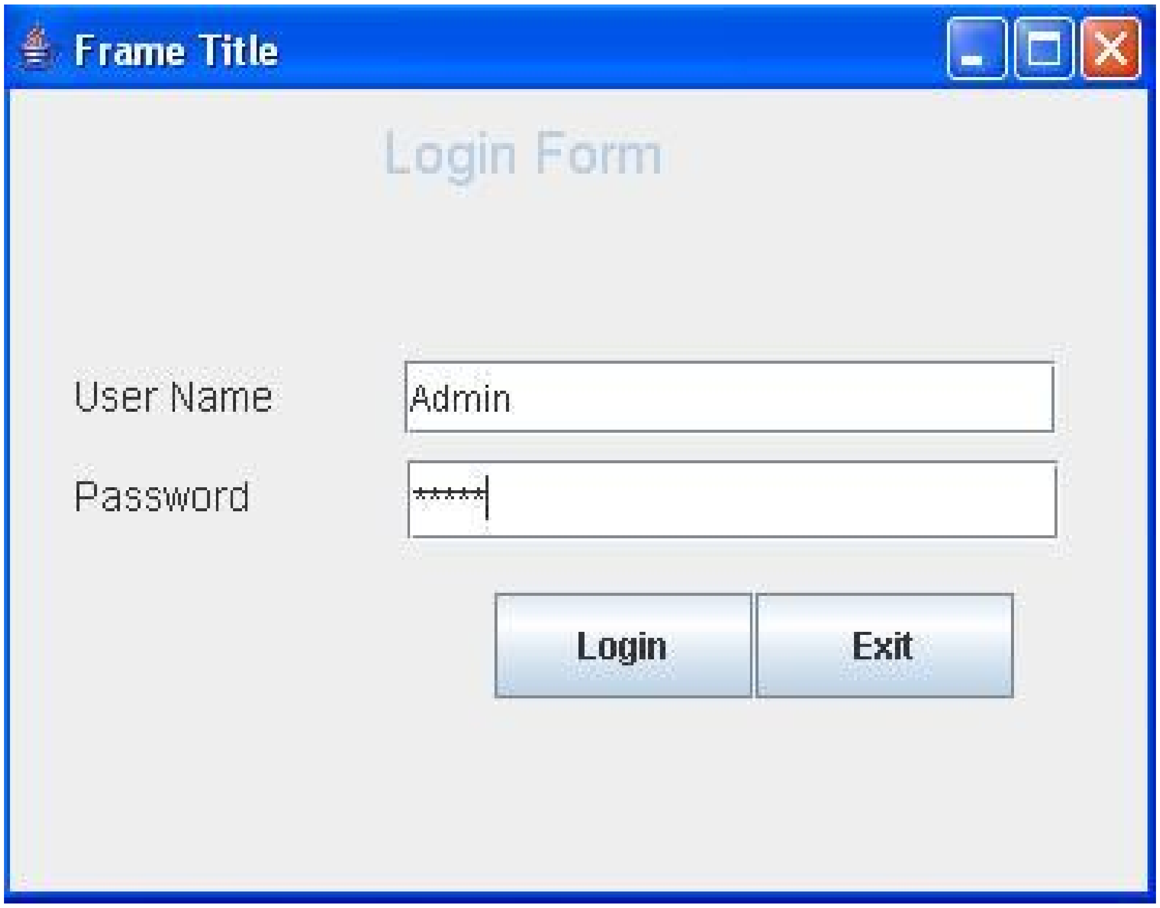
- 4.4.6.2 Area_id:
  Presents the area the candidate is competing for.

- 4.4.6.3 Voter_id:
  A candidate is also a voter so he must have a voter identification.

5. Implementation

About The Prototype:

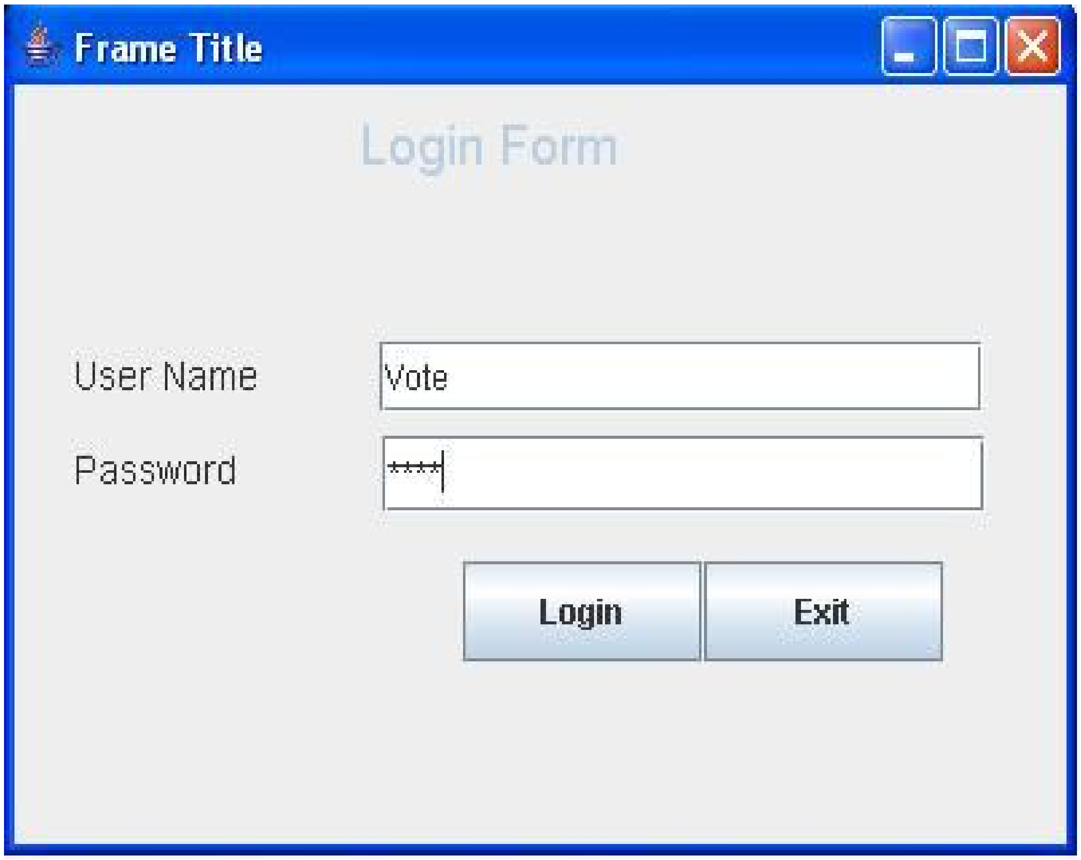
The prototype has been designed using object oriented language java. For ensuring a single vote for a single person we proposed to use finger print which will generate a unique id for a voter but we didn't have that kind of facility here to use the finger print in this prototype that's why we have directly used id for each voter.
The codes and the interfaces are given here. Each interfaces and buttons has it's code just after them. So it will be easy to understand what we tried to represent through this prototype.


Frame Title

Login Form

User Name Admin

Password *****



Vote

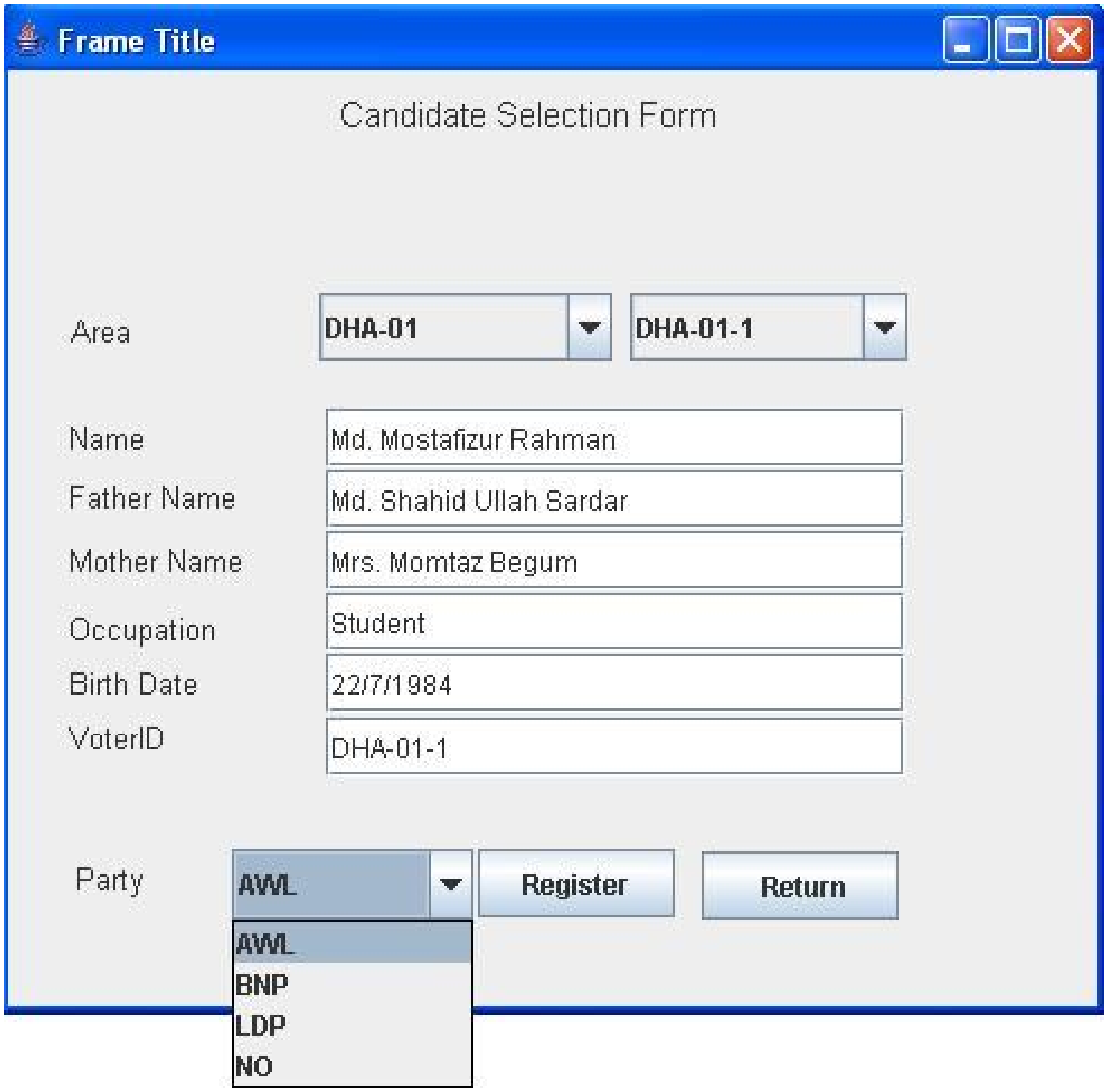
****

5.1 Interface For Login To The System:

```
import java.awt.BorderLayout;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class LoginForm
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  JPasswordField jPasswordField1 = new
JPasswordField();
  Connection conn,conn1;
  Statement smt,smt1;

  public LoginForm() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
```

```
    * @throws java.lang.Exception
    */
   private void jbInit() throws Exception {
     contentPane = (JPanel) getContentPane();
     contentPane.setLayout(null);
     setSize(new Dimension(400, 300));
     setTitle("Frame Title");
     jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 19));
     jLabel1.setForeground(UIManager.getColor(
         "InternalFrame.activeTitleBackground"));
     jLabel1.setText("Login Form");
     jLabel1.setBounds(new Rectangle(128, 6, 111, 28));
     jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 14));
     jLabel2.setText(" Password");
     jLabel2.setBounds(new Rectangle(18, 121, 89, 27));
     jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 14));
     jLabel3.setText(" User Name");
     jLabel3.setBounds(new Rectangle(18, 88, 82, 27));
     jTextField1.setBounds(new Rectangle(136, 90, 225,
25));
     jButton1.setBounds(new Rectangle(167, 167, 89,
35));
     jButton1.setText("Login");
     jButton2.setBounds(new Rectangle(257, 167, 89,
35));
     jButton2.setText("Exit");
     jPasswordField1.setBounds(new Rectangle(137, 123,
225, 27));
     contentPane.add(jLabel1);
     contentPane.add(jLabel3);
     contentPane.add(jLabel2);
     contentPane.add(jTextField1);
     contentPane.add(jButton2);
     contentPane.add(jButton1);
     contentPane.add(jPasswordField1);
     jButton1.addActionListener(
                 new ActionListener()
                 {
                    public void
actionPerformed(ActionEvent e) {
```

```
if(jTextField1.getText().equals("Admin"))

  if(jPasswordField1.getText().equals("admin"))
                                      {
                                        setVisible(false);
                                        new MenuFrame();
                                      }

  if(jTextField1.getText().equals("Vote"))

  if(jPasswordField1.getText().equals("vote"))
                                        {
                                          setVisible(false);
                                          new idInputForm();
                                        }
                                  }
                      }
                    );
     jButton2.addActionListener(
                      new ActionListener()
                      {
                          public void
actionPerformed(ActionEvent e) {
                            try
                            {


Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn =
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          smt.executeQuery("TRUNCATE TABLE Area_01");

          smt.executeQuery("TRUNCATE TABLE Area_02");
          smt.executeQuery("TRUNCATE TABLE Area_03");
          smt.executeQuery("TRUNCATE TABLE VoteTable");
          conn.close();
          smt.close();
                            }catch(Exception e11)
                            {
                              System.out.println("ERROR"+e11);
                               }
                            setVisible(false);
                    }
                   }
               );
      setVisible(true);
    }
    public static void main(String args[])
    {
     new LoginForm();
    }
}
```



Candidate Selection Form

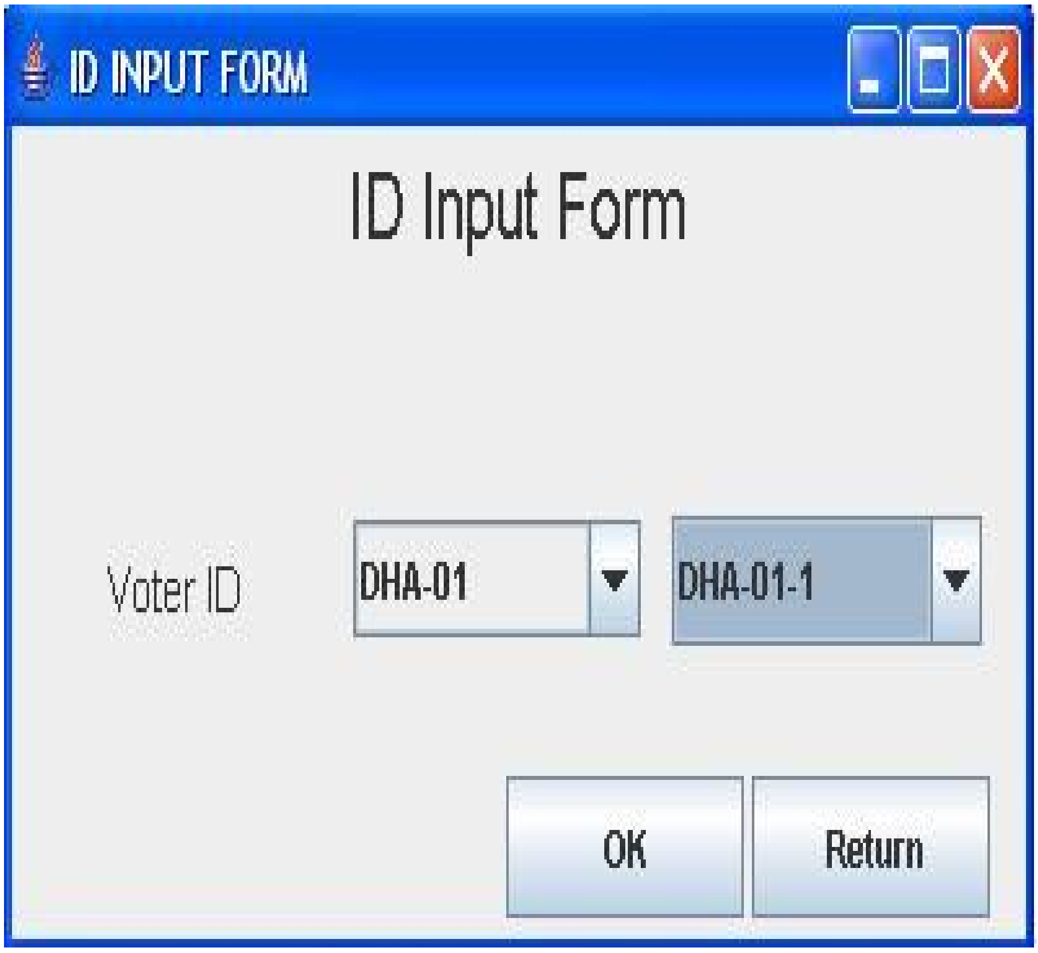
Area	DHA-01	DHA-01-1
Name	Md. Mostafizur Rahman	
Father Name	Md. Shahid Ullah Sardar	
Mother Name	Mrs. Momtaz Begum	
Occupation	Student	
Birth Date	22/7/1984	
VoterID	DHA-01-1	
Party	AWL	

5.2 Interface for Candidate form:

```
import java.awt.BorderLayout;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;


/**
 * <p>Title: </p>
 *
 * <p>Description: </p>
 *
 * <p>Copyright: Copyright (c) 2006</p>
 *
 * <p>Company: </p>
 *
 * @author not attributable
 * @version 1.0
 */
public class candidateForm
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JLabel jLabel6 = new JLabel();
  JLabel jLabel7 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JTextField jTextField5 = new JTextField();
  JTextField jTextField6 = new JTextField();
  JTextField jTextField9 = new JTextField();
  JLabel jLabel8 = new JLabel();
  JComboBox jComboBox1 = new JComboBox();
  JComboBox jComboBox2 = new JComboBox();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  JLabel jLabel9 = new JLabel();
  JComboBox jComboBox3 = new JComboBox();
    Connection conn;
  Statement smt;
  public candidateForm() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(495, 446));
    setTitle("Frame Title");
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel1.setText(" Candidate Selection Form");
    jLabel1.setBounds(new Rectangle(144, 5, 188, 29));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel2.setText(" Mother Name");
    jLabel2.setBounds(new Rectangle(23, 205, 86, 22));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel3.setText(" Occupation");
    jLabel3.setBounds(new Rectangle(23, 235, 86, 22));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel4.setText(" Birth Date");
    jLabel4.setBounds(new Rectangle(23, 259, 86, 22));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel5.setText(" Father Name");
    jLabel5.setBounds(new Rectangle(23, 177, 86, 22));
    jLabel6.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel6.setText(" VoterID");
    jLabel6.setBounds(new Rectangle(23, 283, 86, 22));
    jLabel7.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel7.setText(" Party");
    jLabel7.setBounds(new Rectangle(26, 345, 69, 22));
    jTextField1.setBounds(new Rectangle(142, 149, 259,
26));
    jTextField2.setText("");
    jTextField2.setBounds(new Rectangle(142, 176, 259,
26));
    jTextField3.setBounds(new Rectangle(142, 203, 259,
26));
    jTextField4.setBounds(new Rectangle(142, 230, 259,
26));
    jTextField5.setBounds(new Rectangle(142, 257, 259,
26));
    jTextField6.setBounds(new Rectangle(142, 285, 259,
26));
    jLabel8.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel8.setText(" Name");
    jLabel8.setBounds(new Rectangle(23, 151, 86, 22));
    jComboBox1.setBounds(new Rectangle(139, 98, 131,
30));
    jComboBox2.setBounds(new Rectangle(278, 98, 124,
30));
    jButton1.setBounds(new Rectangle(210, 343, 88,
30));
    jButton1.setText("Register");
    jButton2.setBounds(new Rectangle(310, 344, 88,
30));
    jButton2.setText(" Return");
    jLabel9.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel9.setText(" Area");
    jLabel9.setBounds(new Rectangle(24, 104, 69, 22));
    jComboBox3.setBounds(new Rectangle(100, 343, 108,
31));
    contentPane.add(jLabel1);
    contentPane.add(jTextField1);
    contentPane.add(jLabel5);
    contentPane.add(jLabel2);
    contentPane.add(jLabel3);
    contentPane.add(jLabel4);
    contentPane.add(jLabel6);
    contentPane.add(jTextField6);
    contentPane.add(jTextField5);
    contentPane.add(jTextField4);
    contentPane.add(jTextField3);
    contentPane.add(jTextField2);
    contentPane.add(jLabel8);
    contentPane.add(jComboBox1);
    contentPane.add(jComboBox2);
    contentPane.add(jButton2);
    contentPane.add(jButton1);
    contentPane.add(jLabel9);
    contentPane.add(jLabel7);
    contentPane.add(jComboBox3);
    jButton2.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                           setVisible(false);
                           new MenuFrame();
                          }
                }
              );
   jComboBox1.addItem("DHA-01");
   jComboBox1.addItem("CHG-02");
   jComboBox1.addItem("RAJ-03");
   jComboBox3.addItem("AWL");
   jComboBox3.addItem("BNP");
   jComboBox3.addItem("LDP");
   jComboBox3.addItem("NO");
   jComboBox1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e)
                   {
                      try
               {

 jComboBox2.removeAllItems();

 Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
           conn =
DriverManager.getConnection("jdbc:odbc:Voter");
           smt=conn.createStatement();
           String temp="SELECT * FROM Table_18 WHERE
Area = '"+jComboBox1.getSelectedItem()+"'";
           ResultSet rs = smt.executeQuery(temp);
           while(rs.next())

 jComboBox2.addItem(rs.getString("VoterID"));

                 }catch(Exception ee)
                 {
                   System.out.println("error "+e);
                 }


                     }
                }
              );
    jButton1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e)
                   {
                      try
              {
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn =
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
```

```
         String temp="INSERT into Candidate
values('"+jComboBox2.getSelectedItem()+"','"+jComboBox3
.getSelectedItem()+"','"+jComboBox1.getSelectedItem()+"
','"+jTextField1.getText()+"')";
         smt.executeUpdate(temp);

  JOptionPane.showMessageDialog(null,"Successfully
Inserted To
Database","Successfull",JOptionPane.INFORMATION_MESSAGE
);

  jTextField1.setText("");jTextField2.setText("");jText
Field3.setText("");

  jTextField4.setText("");jTextField5.setText("");jText
Field6.setText("");
               }catch(Exception ee)
               {

  JOptionPane.showMessageDialog(null,"Already a Member
Been Nominated","Please Take a
Look",JOptionPane.ERROR_MESSAGE);              }


                  }
               }
              );

    jComboBox2.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e)
                   {
                     try
              {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         conn =
DriverManager.getConnection("jdbc:odbc:Voter");
         smt=conn.createStatement();
         String temp="SELECT * FROM Table_18 WHERE
VoterID = '"+jComboBox2.getSelectedItem()+"'";
         ResultSet rs = smt.executeQuery(temp);
         rs.next();
```

```
              jTextField1.setText(rs.getString("Name"));
              jTextField2.setText(rs.getString("Father"));
              jTextField3.setText(rs.getString("Mother"));

   jTextField4.setText(rs.getString("Occupation"));

   jTextField5.setText(rs.getString("Date_dd")+"/"+
 rs.getString("Date_mm")+"/"+rs.getString("Date_yy"));

   jTextField6.setText(rs.getString("VoterID"));
                   }catch(Exception ee)
                   {
                    System.out.println("error"+e);
                   }


                       }
                    }
                  );

     setVisible(true);
   }
   public static void main(String args[])
   {
    new candidateForm();
   }
 }
```

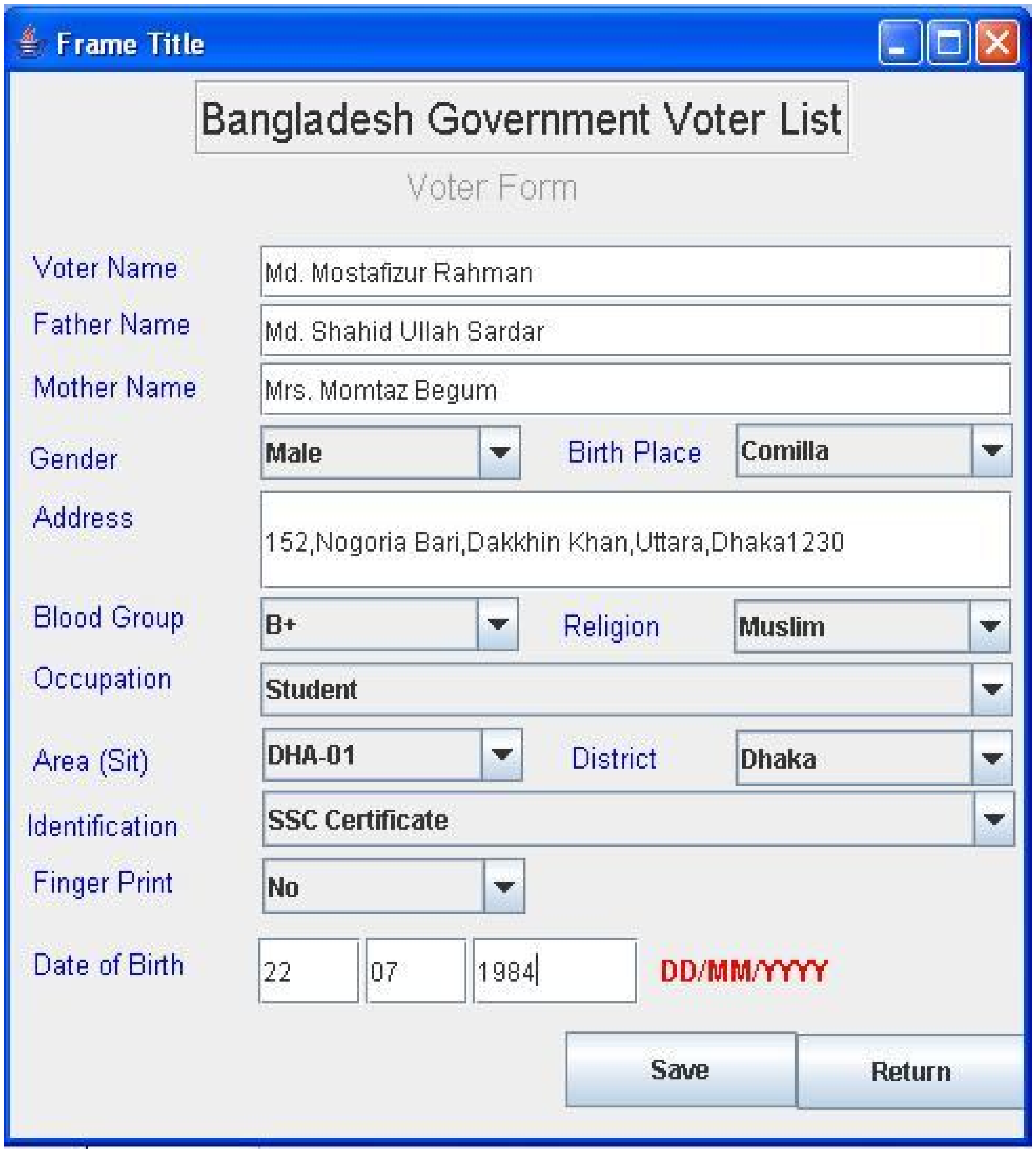

ID INPUT FORM
ID Input Form
Voter ID
DHA-01
DHA-01-1
OK
Return

5.3 Interface For Id Input:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;


public class idInputForm
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JComboBox jComboBox1 = new JComboBox();
  JComboBox jComboBox2 = new JComboBox();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  Connection conn;
  Statement smt;
  public idInputForm() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
```

```
    */
   private void jbInit() throws Exception {
     contentPane = (JPanel) getContentPane();
     contentPane.setLayout(null);
     setSize(new Dimension(400, 228));
     setTitle("ID INPUT FORM");
     jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 22));
     jLabel1.setText("ID Input Form");
     jLabel1.setBounds(new Rectangle(130, 2, 141, 32));
     jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
     jLabel2.setText(" Voter ID");
     jLabel2.setBounds(new Rectangle(33, 99, 74, 22));
     jComboBox1.setBounds(new Rectangle(132, 94, 111,
28));
     jComboBox2.setBounds(new Rectangle(255, 93, 120,
31));
     jButton1.setBounds(new Rectangle(191, 155, 92,
34));
     jButton1.setText("OK");
     jButton2.setBounds(new Rectangle(286, 155, 92,
34));
     jButton2.setText(" Return");
     jComboBox1.addItem("DHA-01");
     jComboBox1.addItem("CHG-02");
     jComboBox1.addItem("RAJ-03");
     contentPane.add(jComboBox2);
     contentPane.add(jComboBox1);
     contentPane.add(jLabel2);
     contentPane.add(jLabel1);
     contentPane.add(jButton2);
     contentPane.add(jButton1);
     jButton2.addActionListener(
     new ActionListener()
     {
       public void actionPerformed(ActionEvent e)
       {
          setVisible(false);
          new LoginForm();
       }
     }
    );
    jButton1.addActionListener(
     new ActionListener()
     {
       public void actionPerformed(ActionEvent e)
       {
        setVisible(false);
        System.out.println("" +
 jComboBox2.getSelectedItem());
          new VoteForm("" +
 jComboBox2.getSelectedItem(),"" +
 jComboBox1.getSelectedItem());
       }
     }
    );

   jComboBox1.addActionListener(
     new ActionListener()
     {
       public void actionPerformed(ActionEvent e)
       {
         jComboBox2.removeAllItems();
          if (jComboBox1.getSelectedItem().equals("DHA-
 01"))
          {
          try
          {
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn =
 DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          String temp="SELECT * FROM Area_01";
          ResultSet rs = smt.executeQuery(temp);
          while(rs.next())
            jComboBox2.addItem(rs.getString("VoterID"));

          }catch(Exception ee)
          {
            System.out.println("error"+e);
          }
          }
          else if
 (jComboBox1.getSelectedItem().equals("CHG-02"))
          {
          try
          {
```

```
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         conn =
DriverManager.getConnection("jdbc:odbc:Voter");
         smt=conn.createStatement();
         String temp="SELECT * FROM Area_02";
         ResultSet rs = smt.executeQuery(temp);
         while(rs.next())
           jComboBox2.addItem(rs.getString("VoterID"));

         }catch(Exception ee)
         {
           System.out.println("error"+e);
         }
         }
        else if
(jComboBox1.getSelectedItem().equals("RAJ-03"))
     {
         try
         {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         conn =
DriverManager.getConnection("jdbc:odbc:Voter");
         smt=conn.createStatement();
         String temp="SELECT * FROM Area_03";
         ResultSet rs = smt.executeQuery(temp);
         while(rs.next())
           jComboBox2.addItem(rs.getString("VoterID"));

         }catch(Exception ee)
         {
           System.out.println("error"+e);
         }
         }
      }
    }
   );
  setVisible(true);
  }
  public static void main(String args[])
  {
   new idInputForm();
  }
}
```



Bangladesh Government Voter List

Voter Form

Voter Name	Md. Mostafizur Rahman
Father Name	Md. Shahid Ullah Sardar
Mother Name	Mrs. Momtaz Begum
Gender	Male
Birth Place	Comilla
Address	152,Nogoria Bari,Dakkhin Khan,Uttara,Dhaka1230
Blood Group	B+
Religion	Muslim
Occupation	Student
Area (Sit)	DHA-01
District	Dhaka
Identification	SSC Certificate
Finger Print	No
Date of Birth	22 07 1984 DD/MM/YYYY

5.4 Interface For Voter Entry:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;


public class InputFormVoterList
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel3 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JLabel jLabel6 = new JLabel();
  JLabel jLabel7 = new JLabel();
  JLabel jLabel8 = new JLabel();
  JLabel jLabel9 = new JLabel();
  JLabel jLabel10 = new JLabel();
  JLabel jLabel11 = new JLabel();
  JLabel jLabel12 = new JLabel();
  JLabel jLabel13 = new JLabel();
  JLabel jLabel14 = new JLabel();
  JLabel jLabel15 = new JLabel();
  JLabel jLabel16 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JComboBox jComboBox1 = new JComboBox();
  JComboBox jComboBox2 = new JComboBox();
  JComboBox jComboBox3 = new JComboBox();
  JComboBox jComboBox4 = new JComboBox();
  JComboBox jComboBox5 = new JComboBox();
  JComboBox jComboBox6 = new JComboBox();
  JComboBox jComboBox7 = new JComboBox();
  JComboBox jComboBox8 = new JComboBox();
  JComboBox jComboBox9 = new JComboBox();
  JLabel jLabel1 = new JLabel();
  TitledBorder titledBorder1 = new TitledBorder("");
  JLabel jLabel2 = new JLabel();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel17 = new JLabel();
  JLabel jLabel18 = new JLabel();
  JTextField jTextField5 = new JTextField();
  JLabel jLabel19 = new JLabel();
  JTextField jTextField6 = new JTextField();
  JTextField jTextField7 = new JTextField();
  JLabel jLabel20 = new JLabel();
  JLabel jLabel21 = new JLabel();
  JLabel jLabel22 = new JLabel();

  Connection conn;
  Statement smt;

  public InputFormVoterList() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(490, 525));
    setTitle("Frame Title");
```

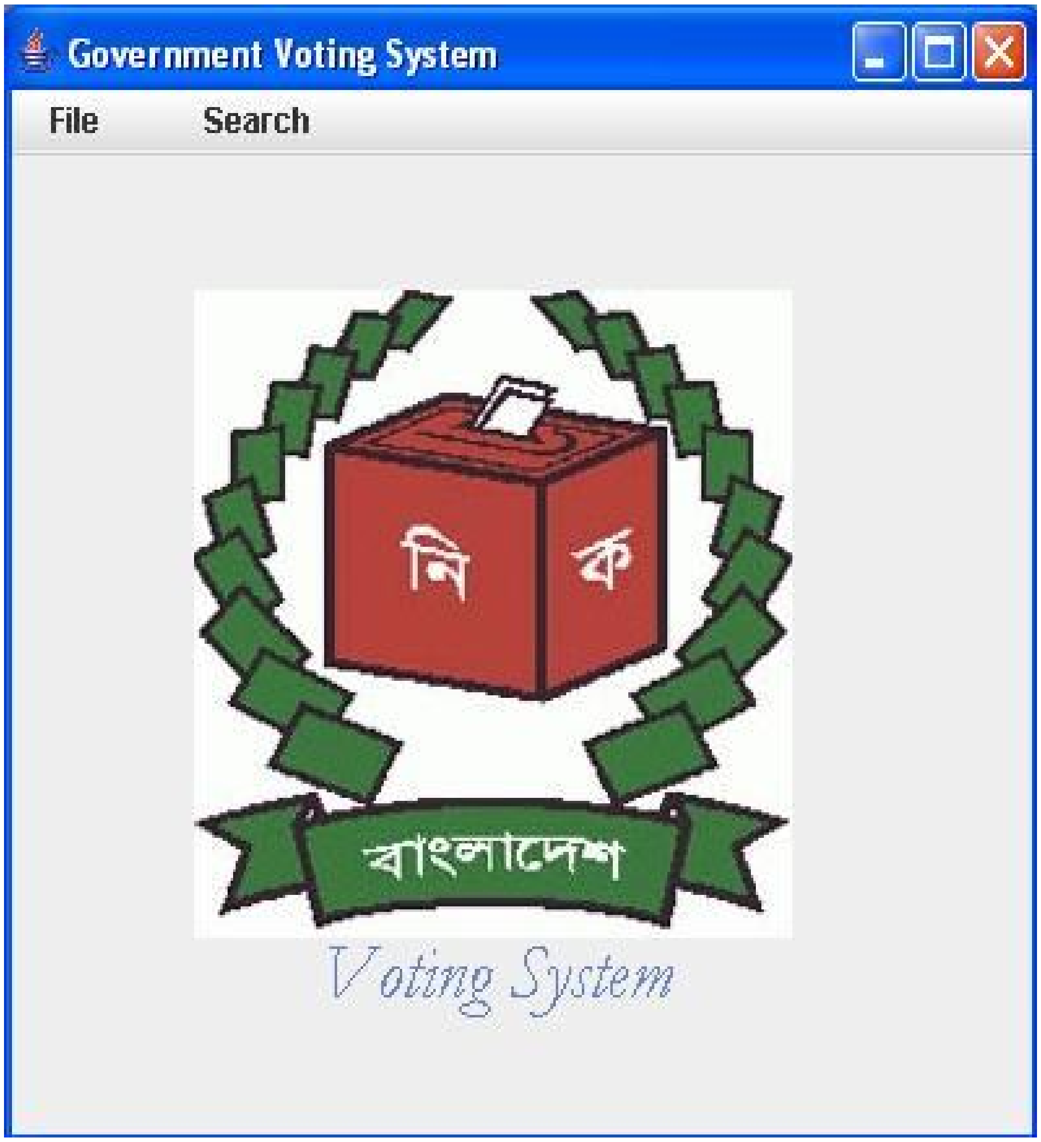
```
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));

jLabel3.setForeground(UIManager.getColor("Menu.selectio
nBackground"));
    jLabel3.setText("Finger Print");
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));

jLabel5.setForeground(UIManager.getColor("Menu.selectio
nBackground"));
    jLabel5.setText("Area(Seat)");
    jLabel6.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel6.setForeground(Color.blue);
    jLabel6.setText("District");
    jLabel6.setBounds(new Rectangle(267, 304, 57, 24));
    jLabel7.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));

jLabel7.setForeground(UIManager.getColor("Menu.selectio
nBackground"));
    jLabel7.setText("Identification");
    jLabel8.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel8.setForeground(Color.blue);
    jLabel8.setText("Occupation");
    jLabel8.setBounds(new Rectangle(11, 267, 83, 24));
    jLabel9.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel9.setForeground(Color.blue);
    jLabel9.setBorder(null);
    jLabel9.setText("Voter Name");
    jLabel9.setBounds(new Rectangle(11, 78, 107, 24));
    jLabel10.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel10.setForeground(Color.blue);
    jLabel10.setText("Address");
    jLabel10.setBounds(new Rectangle(11, 193, 97, 24));
    jLabel11.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel11.setForeground(Color.blue);
    jLabel11.setText("Blood Group");
    jLabel11.setBounds(new Rectangle(11, 239, 84, 24));
    jLabel12.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel12.setForeground(Color.blue);
    jLabel12.setText("Mother Name");
    jLabel12.setBounds(new Rectangle(11, 133, 83, 24));
    jLabel13.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel13.setForeground(Color.blue);

    jLabel13.setText("Religion");
    jLabel13.setBounds(new Rectangle(263, 243, 62,
24));
    jLabel14.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel14.setForeground(Color.blue);
    jLabel14.setText("Birth Place");
    jLabel14.setBounds(new Rectangle(265, 163, 69,
24));
    jLabel15.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));

jLabel15.setForeground(UIManager.getColor("Menu.selecti
onBackground"));
    jLabel15.setText("Gender");
    jLabel16.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel16.setForeground(Color.blue);
    jLabel16.setText("Father Name");
    jLabel16.setBounds(new Rectangle(11, 104, 88, 24));
    jTextField1.setBounds(new Rectangle(119, 80, 358,
25));
    jTextField2.setText("");
    jTextField2.setBounds(new Rectangle(119, 107, 358,
25));
    jTextField3.setBounds(new Rectangle(119, 134, 358,
25));
    jTextField4.setBounds(new Rectangle(119, 193, 358,
46));
    jComboBox1.setBounds(new Rectangle(119, 163, 124,
25));
    jComboBox2.setBounds(new Rectangle(345, 162, 132,
25));
    jComboBox3.setBounds(new Rectangle(119, 242, 123,
25));
```

```
    jComboBox4.setBounds(new Rectangle(344, 243, 132,
25));
    jComboBox5.setBounds(new Rectangle(119, 272, 358,
25));
    jComboBox6.setBounds(new Rectangle(120, 302, 124,
25));
    jComboBox7.setBounds(new Rectangle(345, 303, 132,
26));
    jComboBox8.setBounds(new Rectangle(120, 331, 358,
26));
    jComboBox9.setBounds(new Rectangle(120, 362, 125,
26));
    jLabel1.setEnabled(false);
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 17));
    jLabel1.setBorder(null);
    jLabel1.setText("Voter Form");
    jLabel1.setBounds(new Rectangle(188, 39, 95, 27));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));

jLabel2.setBorder(BorderFactory.createEtchedBorder());
    jLabel2.setText("Bangladesh Government Voter
List");
    jLabel2.setBounds(new Rectangle(88, 4, 312, 35));
    jButton1.setBounds(new Rectangle(264, 442, 110,
35));
    jButton1.setText("Save");
    jButton2.setBounds(new Rectangle(374, 443, 110,
35));
    jButton2.setText("Return");
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel4.setForeground(Color.blue);
    //jLabel4.setForeground(new UIManager(49, 106,
197));
    jLabel4.setText("Finger Print");
    jLabel4.setBounds(new Rectangle(11, 361, 79, 24));
    jLabel17.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));

jLabel17.setForeground(UIManager.getColor("Menu.selecti
onBackground"));
    jLabel17.setBounds(new Rectangle(11, 361, 79, 24));
    jLabel18.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel18.setForeground(Color.blue);
    jLabel18.setText("Date of Birth");
    jLabel18.setBounds(new Rectangle(11, 397, 89, 29));
    jTextField5.setBounds(new Rectangle(118, 399, 49,
31));
    jLabel19.setFont(new java.awt.Font("Dialog",
Font.BOLD, 13));
    jLabel19.setForeground(Color.red);
    jLabel19.setText("DD/MM/YYYY");
    jLabel19.setBounds(new Rectangle(309, 400, 85,
29));
    jTextField6.setBounds(new Rectangle(169, 399, 49,
31));
    jTextField7.setBounds(new Rectangle(220, 399, 79,
31));
    jLabel20.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel20.setForeground(Color.blue);
    jLabel20.setText("Area (Sit)");
    jLabel20.setBounds(new Rectangle(11, 305, 91, 24));
    jLabel21.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel21.setForeground(Color.blue);
    jLabel21.setText("Identification");
    jLabel21.setBounds(new Rectangle(8, 335, 91, 24));
    jLabel22.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel22.setForeground(Color.blue);
    jLabel22.setText("Gender");
    jLabel22.setBounds(new Rectangle(9, 168, 94, 21));
    contentPane.add(jTextField1);
    contentPane.add(jTextField2);
    contentPane.add(jTextField3);
    contentPane.add(jComboBox2);
    contentPane.add(jComboBox1);
    contentPane.add(jTextField4);
    contentPane.add(jComboBox3);
    contentPane.add(jLabel13);
    contentPane.add(jComboBox5);
    contentPane.add(jComboBox6);
    contentPane.add(jLabel6);
    contentPane.add(jComboBox8);
    contentPane.add(jComboBox4);
    contentPane.add(jComboBox7);
    contentPane.add(jComboBox9);
    contentPane.add(jLabel15);
    contentPane.add(jLabel5);
    contentPane.add(jLabel7);
    contentPane.add(jLabel3);
    contentPane.add(jLabel4);
    contentPane.add(jLabel17);
    contentPane.add(jLabel18);
    contentPane.add(jTextField5);
    contentPane.add(jTextField6);
    contentPane.add(jTextField7);
    contentPane.add(jLabel19);
    contentPane.add(jButton1);
    contentPane.add(jButton2);
    contentPane.add(jLabel9);
    contentPane.add(jLabel16);
    contentPane.add(jLabel12);
    contentPane.add(jLabel10);
    contentPane.add(jLabel11);
    contentPane.add(jLabel8);
    contentPane.add(jLabel2);
    contentPane.add(jLabel1);
    contentPane.add(jLabel20);
    contentPane.add(jLabel21);
    contentPane.add(jLabel22);
    contentPane.add(jLabel14);

    addToCombo();
    jButton1.addActionListener(
                new ActionListener()
                {
                    public void
actionPerformed(ActionEvent e)
                    {
          try
        {
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn =
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          Calendar calen = Calendar.getInstance();
          int year = calen.get(Calendar.YEAR);
        String temp22=jTextField7.getText();
        System.out.println(""+temp22);
        int diff = year - Integer.parseInt(temp22);
        System.out.println(""+diff);
      if(jTextField1.getText()!="" &&
jTextField2.getText()!="" && jTextField3.getText()!=""
&& jTextField4.getText()!="" &&
jTextField5.getText()!="" && jTextField6.getText()!=""
&& jTextField7.getText()!="" &&
Integer.parseInt(jTextField5.getText())>0 &&
Integer.parseInt(jTextField5.getText())<=31 &&
Integer.parseInt(jTextField6.getText())>0 &&
Integer.parseInt(jTextField6.getText())<=12 &&
Integer.parseInt(jTextField7.getText())>1900 &&
Integer.parseInt(jTextField7.getText())<2007)
      {
        if (diff < 18)
        {
          String idGenerate= "" +
jComboBox6.getSelectedItem();
          ResultSet rs = smt.executeQuery("SELECT * FROM
Table_18 where Area =
'"+jComboBox6.getSelectedItem()+"'");
          int count=1;
          while(rs.next())
            count++;
          idGenerate=idGenerate+"-"+count;
          String temp="insert into Table_17
"+"values('"+jTextField1.getText()+"','"+jTextField2.ge
tText()+"','"+jTextField3.getText()+"','"+jComboBox1.ge
tSelectedItem()+"','"+jComboBox2.getSelectedItem()+"','
"+jTextField4.getText()+"','"+jComboBox3.getSelectedIte
m()+"','"+jComboBox4.getSelectedItem()+"','"+jComboBox5
.getSelectedItem()+"','"+jComboBox6.getSelectedItem()+"
','"+jComboBox7.getSelectedItem()+"','"+jComboBox8.getS
electedItem()+"','"+jComboBox9.getSelectedItem()+"',"+j
TextField5.getText()+","+jTextField6.getText()+","+jTex
tField7.getText()+",'"+idGenerate+"')";
          System.out.println(temp);
          smt.executeUpdate(temp);
        }
        else
        {
         String idGenerate= "" +
jComboBox6.getSelectedItem();
         ResultSet rs = smt.executeQuery("SELECT * FROM
Table_18 where Area =
'"+jComboBox6.getSelectedItem()+"'");
         int count=1;
         while(rs.next())
           count++;
         idGenerate=idGenerate+"-"+count;
         String temp="insert into Table_18
"+"values('"+jTextField1.getText()+"','"+jTextField2.ge
tText()+"','"+jTextField3.getText()+"','"+jComboBox1.ge
tSelectedItem()+"','"+jComboBox2.getSelectedItem()+"','
"+jTextField4.getText()+"','"+jComboBox3.getSelectedIte
m()+"','"+jComboBox4.getSelectedItem()+"','"+jComboBox5
.getSelectedItem()+"','"+jComboBox6.getSelectedItem()+"
','"+jComboBox7.getSelectedItem()+"','"+jComboBox8.getS
electedItem()+"','"+jComboBox9.getSelectedItem()+"',"+j
TextField5.getText()+","+jTextField6.getText()+","+jTex
tField7.getText()+",'"+idGenerate+"')";
         System.out.println(temp);
         smt.executeUpdate(temp);
        }
       JOptionPane.showMessageDialog(null,"Successfully
Inserted To
Database","Successfull",JOptionPane.INFORMATION_MESSAGE
);
      }
      else
          JOptionPane.showMessageDialog(null,"Few
Fields Are Empty or error
input","ERROR",JOptionPane.ERROR_MESSAGE);
      }catch(Exception exp)
      {
        JOptionPane.showMessageDialog(null,"Unable to
Insert into Database","Please Take a
Look",JOptionPane.ERROR_MESSAGE);
      }

  jTextField1.setText("");jTextField2.setText("");jText
Field3.setText("");jTextField4.setText("");

  jTextField5.setText("");jTextField6.setText("");jText
Field7.setText("");
```

```
                        }
                    }
                 );
     jButton2.addActionListener(
                    new ActionListener()
                    {
                        public void
actionPerformed(ActionEvent e) {

                                    setVisible(false);
                                    new MenuFrame();
                              }
                    }
                 );

     setVisible(true);
   }
   public void addToCombo()
   {
    jComboBox1.addItem("Male");
    jComboBox1.addItem("Female");

    jComboBox2.addItem("Bagherhat");
    jComboBox2.addItem("Barisal");
    jComboBox2.addItem("Bogra");
    jComboBox2.addItem("Chittagong");
    jComboBox2.addItem("Comilla");
    jComboBox2.addItem("Dhaka");
    jComboBox2.addItem("Faridpur");
     jComboBox2.addItem("Gazipur");
     jComboBox2.addItem("Hobigonj");
    jComboBox2.addItem("Jamalpur");
    jComboBox2.addItem("Jassore");
    jComboBox2.addItem("Khulna");
    jComboBox2.addItem("Lalmonirhat");
    jComboBox2.addItem("Munshiganj");
    jComboBox2.addItem("Mymanshing");
    jComboBox2.addItem("Netrokona");
    jComboBox2.addItem("Panchagar");
    jComboBox2.addItem("Rajshahi");
    jComboBox2.addItem("Rangpur");
    jComboBox2.addItem("Sayedpur");
    jComboBox2.addItem("Sylhet");
     jComboBox2.addItem("Tecnuf");
```

```
 jComboBox2.addItem("Vola");


jComboBox3.addItem("A+");
jComboBox3.addItem("A-");
 jComboBox3.addItem("B+");
 jComboBox3.addItem("B-");
 jComboBox3.addItem("AB+");
 jComboBox3.addItem("AB-");
 jComboBox3.addItem("O+");
 jComboBox3.addItem("O-");


jComboBox4.addItem("Muslim");
jComboBox4.addItem("Hindu");
 jComboBox4.addItem("Christian");
 jComboBox4.addItem("Buddist");




jComboBox5.addItem("Student");
jComboBox5.addItem("Businessman");
jComboBox5.addItem("Government Official");
jComboBox5.addItem("Private Service Holder");
jComboBox5.addItem("Teacher");
jComboBox5.addItem("Others");

jComboBox6.addItem("DHA-01");
jComboBox6.addItem("CHG-02");
jComboBox6.addItem("RAJ-03");
jComboBox6.addItem("KHU-04");
jComboBox6.addItem("BAR-05");
 jComboBox6.addItem("SYL-06");

 jComboBox7.addItem("Bagherhat");
 jComboBox7.addItem("Barisal");
 jComboBox7.addItem("Bogra");
 jComboBox7.addItem("Chittagong");
 jComboBox7.addItem("Comilla");
 jComboBox7.addItem("Dhaka");
 jComboBox7.addItem("Faridpur");
 jComboBox7.addItem("Gazipur");
 jComboBox7.addItem("Hobigonj");
```

```
     jComboBox7.addItem("Jamalpur");
     jComboBox7.addItem("Jassore");
     jComboBox7.addItem("Khulna");
     jComboBox7.addItem("Lalmonirhat");
    jComboBox7.addItem("Munshiganj");
     jComboBox7.addItem("Mymanshing");
     jComboBox7.addItem("Netrokona");
     jComboBox7.addItem("Panchagar");
     jComboBox7.addItem("Rajshahi");
     jComboBox7.addItem("Rangpur");
     jComboBox7.addItem("Sayedpur");
     jComboBox7.addItem("Sylhet");
     jComboBox7.addItem("Tecnuf");
     jComboBox7.addItem("Vola");



    jComboBox8.addItem("SSC Certificate");
    jComboBox8.addItem("HSC Certificate");
    jComboBox8.addItem("Chairman Certificate");
    jComboBox8.addItem("Commisioner Certificate");

    jComboBox9.addItem("Yes");
    jComboBox9.addItem("No");
   }
   public static void main(String args[])
   {
    new InputFormVoterList();
   }
 }
```

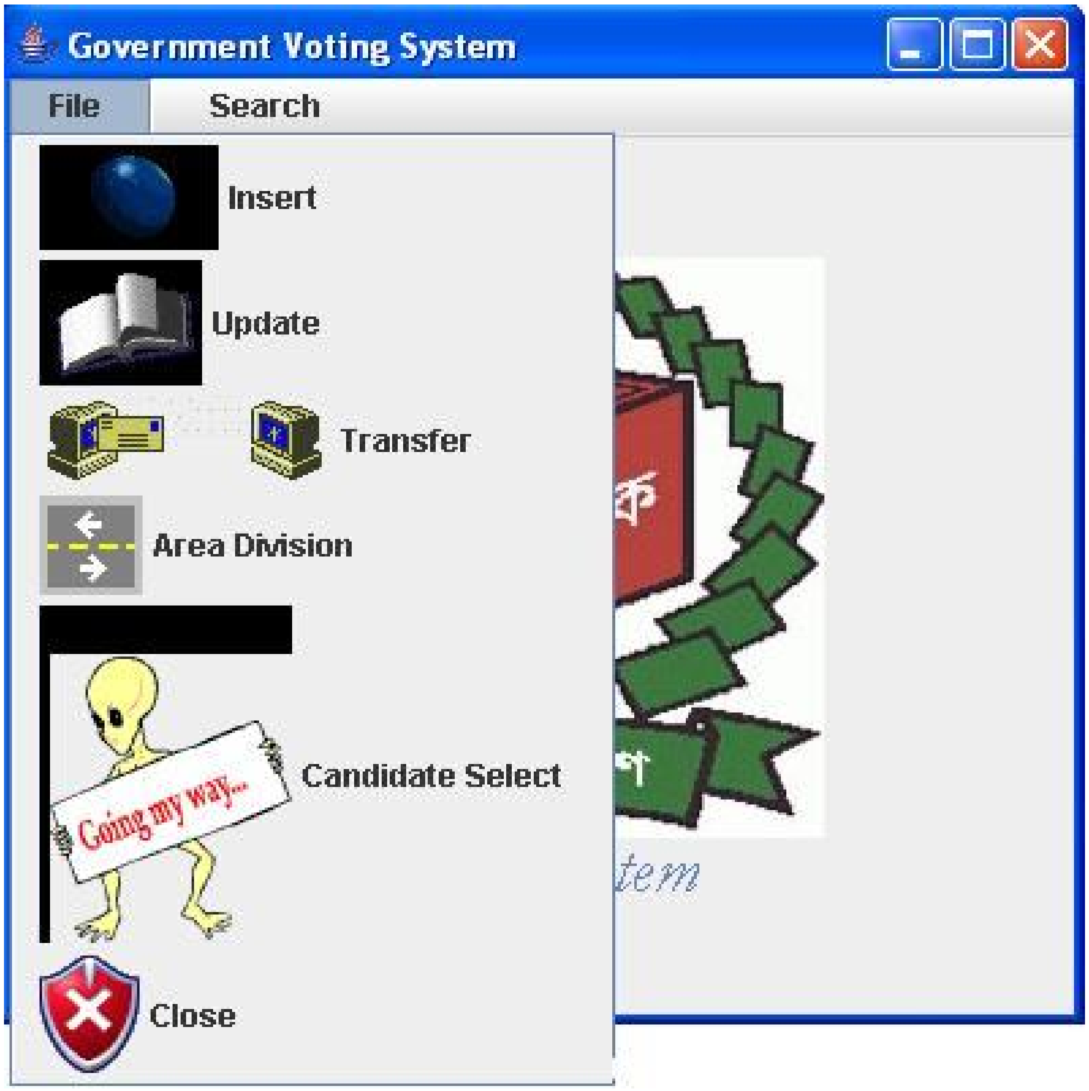

Government Voting System
File
Search
নি
ক
বাংলাদেশ
Voting System

5.5 Interface For Main Menu:

```
import java.awt.BorderLayout;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class MenuFrame
    extends JFrame {
  JPanel contentPane;
  JMenuBar jMenuBar1 = new JMenuBar();
  JMenu jMenu1 = new JMenu();
  JMenuItem jMenuItem1;
  JMenuItem jMenuItem2;
  JMenuItem jMenuItem4;
  JMenuItem jMenuItem5;
  JMenuItem jMenuItem6;
  JMenuItem jMenuItem7;
  JMenuItem jMenuItem8;
  JMenu jMenu2 = new JMenu();
  JMenuItem jMenuItem3;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3;
   Connection conn,conn1;
  Statement smt,smt1;
  public MenuFrame() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    this.setJMenuBar(jMenuBar1);
    setSize(new Dimension(400, 400));
    setTitle("Government Voting System");
    jMenu1.setText("   File    ");
    Icon anew = new
ImageIcon("images/Around_the_world.gif");
    jMenuItem1 = new JMenuItem("Insert",anew);
    Icon del = new ImageIcon("images/delete.gif");
    jMenuItem2 = new JMenuItem("Close",del);
    jMenu2.setText("     Search       ");
    Icon srch = new ImageIcon("images/Alien41.gif");
    jMenuItem3 = new JMenuItem("Candidate
Select",srch);

    jLabel1.setFont(new java.awt.Font("Garamond",
Font.ITALIC, 28));

jLabel1.setForeground(UIManager.getColor("ProgressBar.s
electionBackground"));
    jLabel1.setText("");
    jLabel1.setBounds(new Rectangle(270, 300, 300,
330));

    Icon anew112 = new
ImageIcon("images/bec_logo.gif");
    jLabel3 = new JLabel(anew112);
    jLabel3.setBounds(new Rectangle(70, 30, 230, 260));

    jLabel2.setFont(new java.awt.Font("Garamond",
Font.ITALIC, 28));

jLabel2.setForeground(UIManager.getColor("ProgressBar.s
electionBackground"));
```

```
    jLabel2.setText("     Voting System");
    jLabel2.setBounds(new Rectangle(83, 260, 231, 53));

    //added
    Icon anew1 = new ImageIcon("images/3d_book2.gif");
    jMenuItem4 = new JMenuItem("Update",anew1);
    Icon del1 = new
ImageIcon("images/2_computers.gif");
    jMenuItem5 = new JMenuItem("Transfer",del1);
    Icon del2 = new ImageIcon("images/sign11.gif");
    jMenuItem6 = new JMenuItem("Area Division",del2);

    Icon del22 = new ImageIcon("images/8ball.gif");
    jMenuItem7 = new JMenuItem("Result",del22);
    Icon del23 = new ImageIcon("images/gice_5.gif");
    jMenuItem8 = new JMenuItem("Total Result",del23);

    jMenuBar1.add(jMenu1);
    jMenuBar1.add(jMenu2);
    jMenu1.add(jMenuItem1);
    jMenu1.add(jMenuItem4);
    jMenu1.add(jMenuItem5);
    jMenu1.add(jMenuItem6);
    jMenu1.add(jMenuItem3);
    jMenu1.add(jMenuItem2);
    //jMenu2.add(jMenuItem3);
    jMenu2.add(jMenuItem7);
    jMenu2.add(jMenuItem8);
    contentPane.add(jLabel2);
    contentPane.add(jLabel1);
    contentPane.add(jLabel3);

    jMenuItem8.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                           setVisible(false);
                           new VoteResult1();

                          }
                }
              );
```

```
        jMenuItem3.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                           setVisible(false);
                           new candidateForm();

                          }
                }
              );
        jMenuItem7.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                           new resultFrame();
                           //w idInputForm();

                          }
                }
              );
    jMenuItem2.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                           setVisible(false);
                           new LoginForm();
                          }
                }
              );
    jMenuItem1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                           setVisible(false);
                           new InputFormVoterList();
                          }
                }
              );
  jMenuItem4.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                           setVisible(false);
                           new updateFinger();
                          }
                }
              );
  jMenuItem5.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                           new table17Totable18();
                          }
                }
              );
  jMenuItem6.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                           new areaDivision();
                          }
                }
              );
    setVisible(true);
  }
  public static void main(String args[])
  {
   new MenuFrame();
  }
}
```


Government Voting System
File
Search
Insert
Update
Transfer
Area Division
Going my way...
Candidate Select
Close
tem

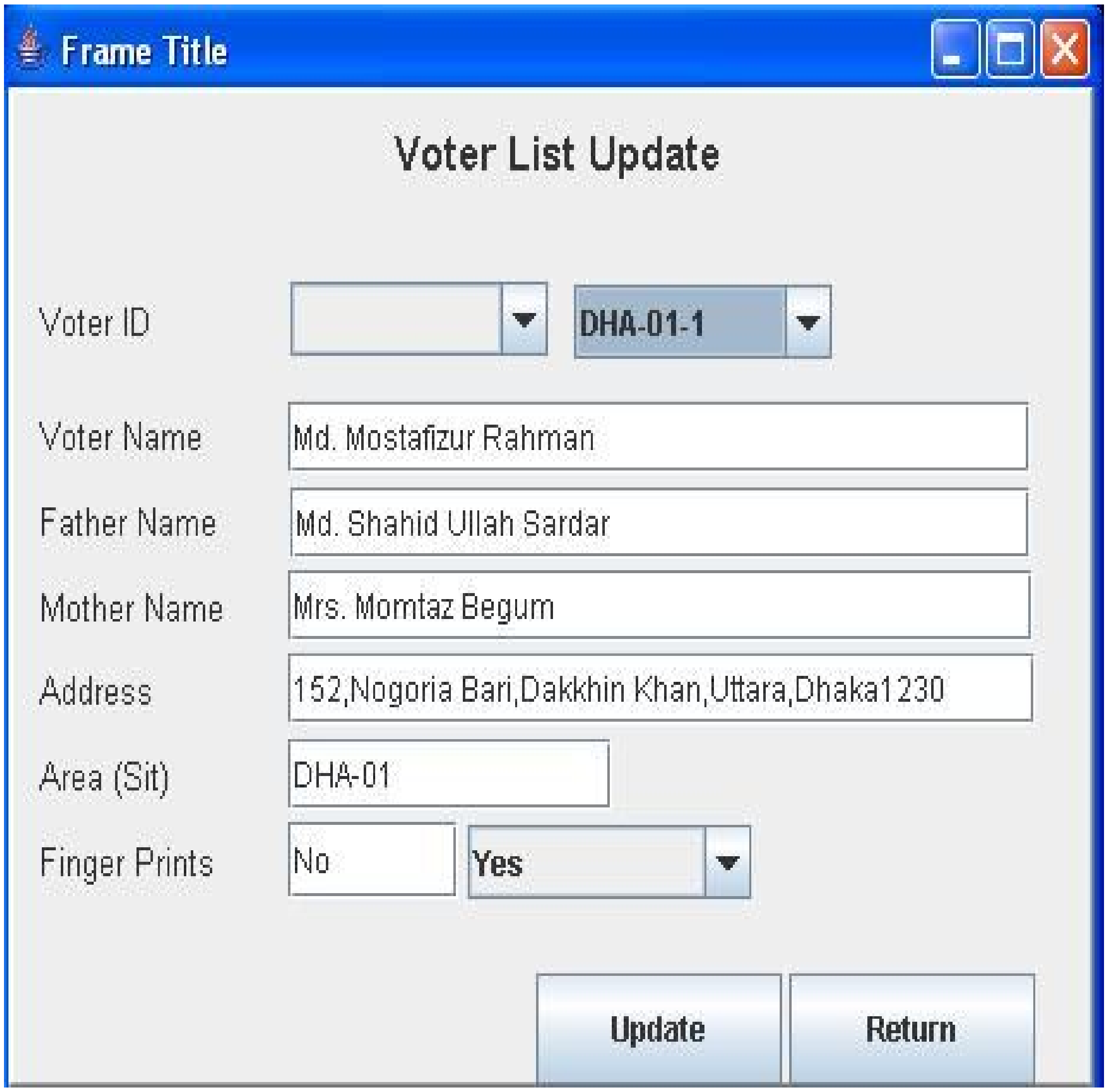
5.6 Area Division Button For Transfer the Voters to Area:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;

public class areaDivision
{
   Connection conn,conn1;
   Statement smt,smt1;
  public areaDivision()
  {
    try
    {
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  conn =
DriverManager.getConnection("jdbc:odbc:Voter");
     smt=conn.createStatement();
     ResultSet rs = smt.executeQuery("SELECT * FROM
Table_18");
     while(rs.next())
      {
       String temp=rs.getString("Area");
      if (temp.equals("DHA-01") &&
rs.getString("FingerPrint").equals("Yes"))
      {
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     conn1 =
DriverManager.getConnection("jdbc:odbc:Voter");
     smt1=conn1.createStatement();
     smt1.executeUpdate("insert into Area_01 
values('"+rs.getString("VoterID")+"','"+rs.getString("N
ame")+"')");
     conn1.close();
     smt1.close();
      }
      else if (temp.equals("RAJ-03")&& 
rs.getString("FingerPrint").equals("Yes"))
      {
       Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     conn1 = 
DriverManager.getConnection("jdbc:odbc:Voter");
     smt1=conn1.createStatement();
     smt1.executeUpdate("insert into Area_03 
values('"+rs.getString("VoterID")+"','"+rs.getString("N
ame")+"')");
     conn1.close();
     smt1.close();
      }
      else if (temp.equals("CHG-02")&& 
rs.getString("FingerPrint").equals("Yes"))
      {
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     conn1 = 
DriverManager.getConnection("jdbc:odbc:Voter");
     smt1=conn1.createStatement();
     smt1.executeUpdate("insert into Area_02 
values('"+rs.getString("VoterID")+"','"+rs.getString("N
ame")+"')");
     conn1.close();
     smt1.close();
      }
     }
    }catch(Exception e)
    {
     System.out.println("Errorr "+e);
    }
  }
  public static void main(String args[])
  {
    new areaDivision();
  }
}
```

5.7 Transfer Button For Migration of Voters:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;

public class table17Totable18{
  Connection conn,conn1,conn2;
  Statement smt,smt1,smt2;

  public table17Totable18()
  {
    try
    {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    conn =
DriverManager.getConnection("jdbc:odbc:Voter");
    smt=conn.createStatement();

         ResultSet rs,rs1;
         rs = smt.executeQuery("SELECT * FROM
Table_17");
         rs1 = rs;
         while (rs.next())
         {
          String nm1=rs.getString("Name");
          String nm2=rs.getString("Father");
          String nm3=rs.getString("Mother");
          String nm4=rs.getString("Gender");
          String nm5=rs.getString("BirthPlace");
          String nm6=rs.getString("Address");
          String nm7=rs.getString("BloodGroup");
```

```
           String nm8=rs.getString("Religion");
           String nm9=rs.getString("Occupation");
           String nm10=rs.getString("Area");
           String nm11=rs.getString("District");
           String nm12=rs.getString("Identification");
           String nm13=rs.getString("FingerPrint");
           int
dd=Integer.parseInt(rs.getString("Date_dd"));
           int
mm=Integer.parseInt(rs.getString("Date_mm"));
           int
yy=Integer.parseInt(rs.getString("Date_yy"));
           String nm14=rs.getString("VoterID");
           Calendar calen = Calendar.getInstance();
           int tyy = calen.get(Calendar.YEAR);
           int tmm = calen.get(Calendar.MONTH);
           int tdd = calen.get(Calendar.DATE);
           int diffYear=0;
           System.out.println("df");
           int diffTemp=tdd-dd;
           System.out.println("Date:"+diffTemp);
           if(diffTemp<0)
             mm++;
           diffTemp=tmm-mm;
           System.out.println("Date:"+diffTemp);
           if(diffTemp<0)
             yy++;
           diffTemp=tyy-yy;
           System.out.println("Date:"+diffTemp);
      if(diffTemp>=18)
      {
       Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
       conn1 =
DriverManager.getConnection("jdbc:odbc:Voter");
       smt1=conn1.createStatement();
       String temp1 = "insert into Table_18
"+"values('"+nm1+"','"+nm2+"','"+nm3+"','"+nm4+"','"+nm
5+"','"+nm6+"','"+nm7+"','"+nm8+"','"+nm9+"','"+nm10+"'
,'"+nm11+"','"+nm12+"','"+nm13+"',"+dd+","+mm+","+yy+",
'"+nm14+"')"; ;
       System.out.println(temp1);
       smt1.executeUpdate(temp1);
       smt1.close();
       conn1.close();
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
      conn2 =
DriverManager.getConnection("jdbc:odbc:Voter");
      smt2=conn2.createStatement();
      String tmp1="DELETE from Table_17 where VoterID =
'"+nm14+"'" ;
      System.out.println(tmp1);
      smt2.executeUpdate(tmp1);
      conn2.close();
      smt2.close();
      }
   }
   }catch(Exception e)
   {
     System.out.println("Error"+e);
   }
  }
  public static void main(String args[])
  {
   new table17Totable18();
  }
   }
```



Voter List Update

Voter ID	DHA-01-1
Voter Name	Md. Mostafizur Rahman
Father Name	Md. Shahid Ullah Sardar
Mother Name	Mrs. Momtaz Begum
Address	152,Nogoria Bari,Dakkhin Khan,Uttara,Dhaka1230
Area (Sit)	DHA-01
Finger Prints	No / Yes

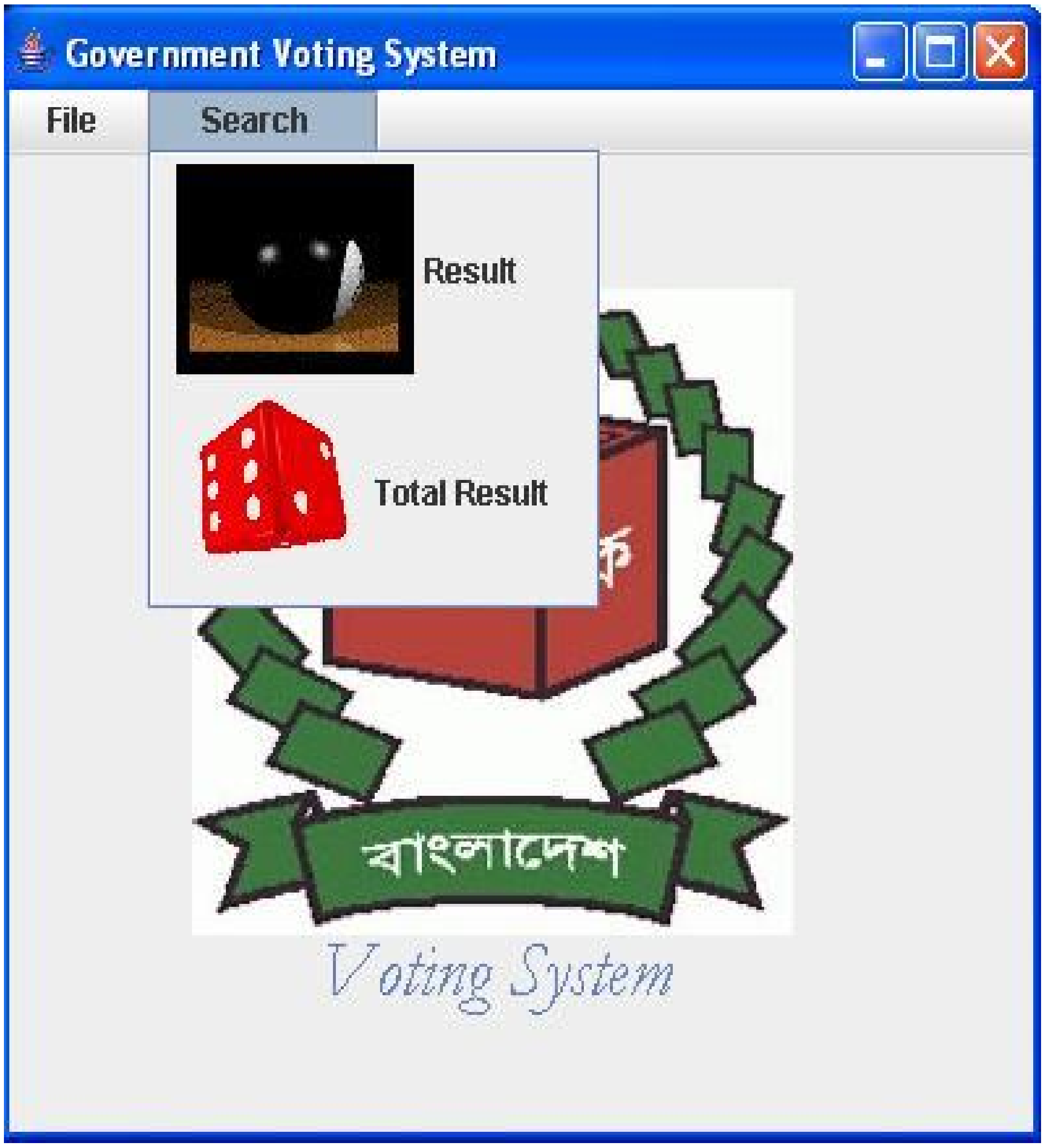
5.8 Interfce For Updating Voter Details:

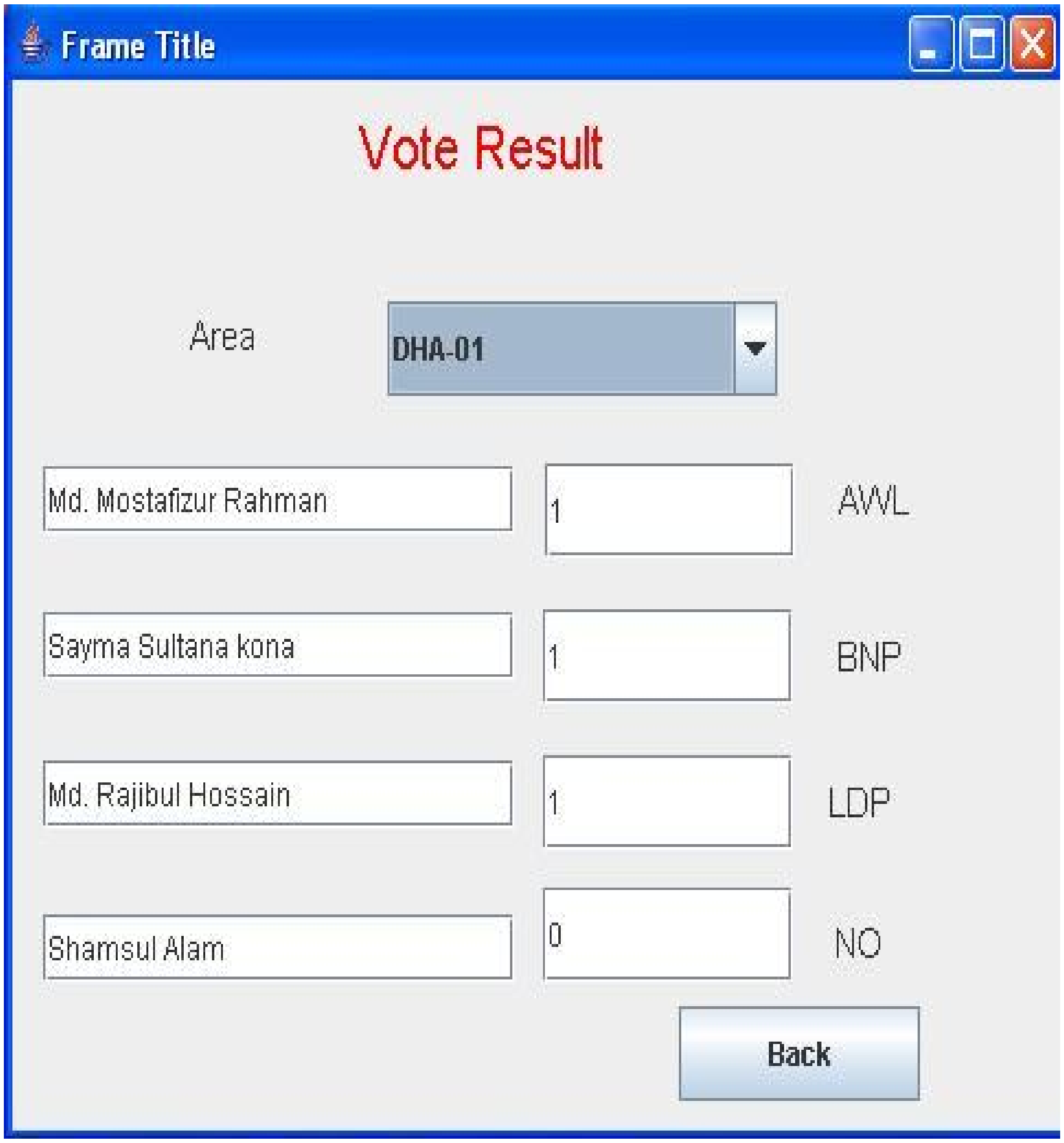
```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;

public class updateFinger
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JLabel jLabel6 = new JLabel();
  JLabel jLabel7 = new JLabel();
  JLabel jLabel8 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JLabel jLabel9 = new JLabel();
  JTextField jTextField5 = new JTextField();
  JTextField jTextField6 = new JTextField();
  JComboBox jComboBox1 = new JComboBox();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  JComboBox jComboBox2 = new JComboBox();
  JComboBox jComboBox3 = new JComboBox();

  Connection conn;
  Statement smt;
  int flag=0;
  public updateFinger() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }


  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(467, 384));
    setTitle("Frame Title");
    jLabel1.setEnabled(true);
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.BOLD, 17));
    jLabel1.setText("Voter List Update");
    jLabel1.setBounds(new Rectangle(163, 6, 153, 32));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel2.setText("Voter ID");
    jLabel2.setBounds(new Rectangle(13, 73, 71, 18));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel3.setText("Area (Sit)");
    jLabel3.setBounds(new Rectangle(13, 232, 86, 18));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel4.setText("Address");
    jLabel4.setBounds(new Rectangle(13, 202, 71, 18));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel5.setText("Area");
    jLabel6.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel6.setText("Voter Name");
    jLabel6.setBounds(new Rectangle(13, 113, 71, 18));
    jLabel7.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel7.setText("Mother Name");
    jLabel7.setBounds(new Rectangle(13, 173, 86, 18));
    jLabel8.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel8.setText("Father Name");
    jLabel8.setBounds(new Rectangle(13, 143, 83, 18));
    jTextField1.setEnabled(true);
    jTextField1.setBounds(new Rectangle(118, 110, 314,
25));
    jTextField2.setEditable(true);
    jTextField2.setText("");
    jTextField2.setBounds(new Rectangle(119, 140, 313,
25));
    jTextField3.setEditable(true);
    jTextField3.setBounds(new Rectangle(118, 169, 315,
25));
    jTextField4.setEnabled(true);
    jTextField4.setText("");
    jTextField4.setBounds(new Rectangle(118, 198, 316,
25));
    jLabel9.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 13));
    jLabel9.setText("Finger Prints");
    jLabel9.setBounds(new Rectangle(13, 262, 86, 18));
    jTextField5.setEditable(true);
    jTextField5.setBounds(new Rectangle(118, 228, 137,
25));
    jTextField6.setEnabled(true);
    jTextField6.setBounds(new Rectangle(118, 257, 72,
27));
    jComboBox1.setBounds(new Rectangle(119, 68, 109,
26));
    jButton1.setBounds(new Rectangle(223, 310, 104,
39));
    jButton1.setText("Update");
    jButton2.setBounds(new Rectangle(330, 310, 104,
39));
    jButton2.setText("Return");
    jComboBox2.setBounds(new Rectangle(239, 69, 109,
26));
    jComboBox3.setBounds(new Rectangle(194, 258, 120,
26));
    jTextField2.setEditable(true);
    jTextField3.setEditable(true);
    jTextField6.setEditable(true);
```

```
contentPane.setBorder(BorderFactory.createRaisedBevelBo
rder());
    contentPane.add(jLabel1);
    contentPane.add(jLabel2);
    contentPane.add(jLabel6);
    contentPane.add(jLabel8);
    contentPane.add(jLabel7);
    contentPane.add(jLabel4);
    contentPane.add(jLabel5);
    contentPane.add(jLabel9);
    contentPane.add(jLabel3);
    contentPane.add(jTextField1);
    contentPane.add(jTextField2);
    contentPane.add(jTextField3);
    contentPane.add(jTextField4);
    contentPane.add(jTextField5);
    contentPane.add(jTextField6);
    contentPane.add(jComboBox1);
    contentPane.add(jComboBox2);
    contentPane.add(jComboBox3);
    contentPane.add(jButton2);
    contentPane.add(jButton1);
    add_Item();
    jButton2.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {
                              setVisible(false);
                              new MenuFrame();
                          }
                }
              );
    jComboBox1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e)
                   {
                          try
                      {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
          conn = 
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          String temp="SELECT * FROM Table_17 where 
VoterID = '"+jComboBox1.getSelectedItem()+"'" ;
          System.out.println(temp);
          ResultSet rs = smt.executeQuery(temp);
          rs.next();
          jTextField1.setText(rs.getString("Name"));
          jTextField2.setText(rs.getString("Father"));
          jTextField3.setText(rs.getString("Mother"));
          jTextField4.setText(rs.getString("Address"));
          jTextField5.setText(rs.getString("Area"));
          String temp1=rs.getString("FingerPrint");
          jTextField6.setText(temp1);
          System.out.println(temp1);
          if(temp1.equals("No")){
            jComboBox3.addItem("Yes");
          }
          flag=1;
        }catch(Exception er)
        {
          System.out.println("Error"+er);
                         }
                    }
                 }
               );
    jComboBox2.addActionListener(
                 new ActionListener()
                 {
                    public void 
actionPerformed(ActionEvent e)
                    {
                           try
                       {
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn = 
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          String temp="SELECT * FROM Table_18 where 
VoterID = '"+jComboBox2.getSelectedItem()+"'" ;
          System.out.println(temp);
          ResultSet rs = smt.executeQuery(temp);
          rs.next();
          jTextField1.setText(rs.getString("Name"));
          jTextField2.setText(rs.getString("Father"));
          jTextField3.setText(rs.getString("Mother"));
          jTextField4.setText(rs.getString("Address"));
          jTextField5.setText(rs.getString("Area"));
          String temp1=rs.getString("FingerPrint");
          jTextField6.setText(temp1);
          System.out.println(temp1);
          jComboBox3.removeAllItems();
          if(temp1.equals("No"))
            jComboBox3.addItem("Yes");
          else
            jComboBox3.setEditable(false);
          flag=2;
        }catch(Exception er)
        {
          System.out.println("Error"+er);
                        }
                    }
                 }
               );
     jButton1.addActionListener(
                 new ActionListener()
                 {
                    public void
actionPerformed(ActionEvent e)
                       {
                               try
        {
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          conn =
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          if (flag==1){
            smt.executeQuery("UPDATE Table_17 SET
FingerPrint = 'Yes' WHERE VoterID =
'"+jComboBox1.getSelectedItem()+"'");
        JOptionPane.showMessageDialog(null,"Successfully
Updated","Successfull",JOptionPane.INFORMATION_MESSAGE)
;
          }
          else if (flag==2){
            smt.executeQuery("UPDATE Table_18 SET
FingerPrint = 'Yes' WHERE VoterID =
'"+jComboBox2.getSelectedItem()+"'");

  JOptionPane.showMessageDialog(null,"Successfully
Updated","Successfull",JOptionPane.INFORMATION_MESSAGE)
;
          }

            smt.close();
            conn.close();

            new updateFinger();
        }catch(Exception er)
        {
          System.out.println("Error"+er);
  JOptionPane.showMessageDialog(null,"Successfully
Updated","Successfull",JOptionPane.INFORMATION_MESSAGE)
;
  jTextField1.setText("");
  jTextField2.setText("");
  jTextField3.setText("");
  jTextField4.setText("");
  jTextField5.setText("");

        }
                                 }
                 }
               );
     setVisible(true);
   }
   public void add_Item()
   {

                          try
         {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        conn =
DriverManager.getConnection("jdbc:odbc:Voter");
        smt=conn.createStatement();
        ResultSet rs1 = smt.executeQuery("SELECT * FROM
Table_17");
        while(rs1.next())
          jComboBox1.addItem(rs1.getString("VoterID"));
```

```
        ResultSet rs2 = smt.executeQuery("SELECT * FROM 
Table_18");
        while(rs2.next())
          jComboBox2.addItem(rs2.getString("VoterID"));
    smt.close();
       conn.close();
       }catch(Exception er)
       {
        System.out.println("Error"+er);
       }

   }
   public static void main(String args[])
   {
    new updateFinger();
    //add_Item();
   }
}
```


Government Voting System
File
Search
Result
Total Result
বাংলাদেশ
Voting System


Frame Title

Vote Result

Area: DHA-01

Md. Mostafizur Rahman	1	AWL
Sayma Sultana kona	1	BNP
Md. Rajibul Hossain	1	LDP
Shamsul Alam	0	NO

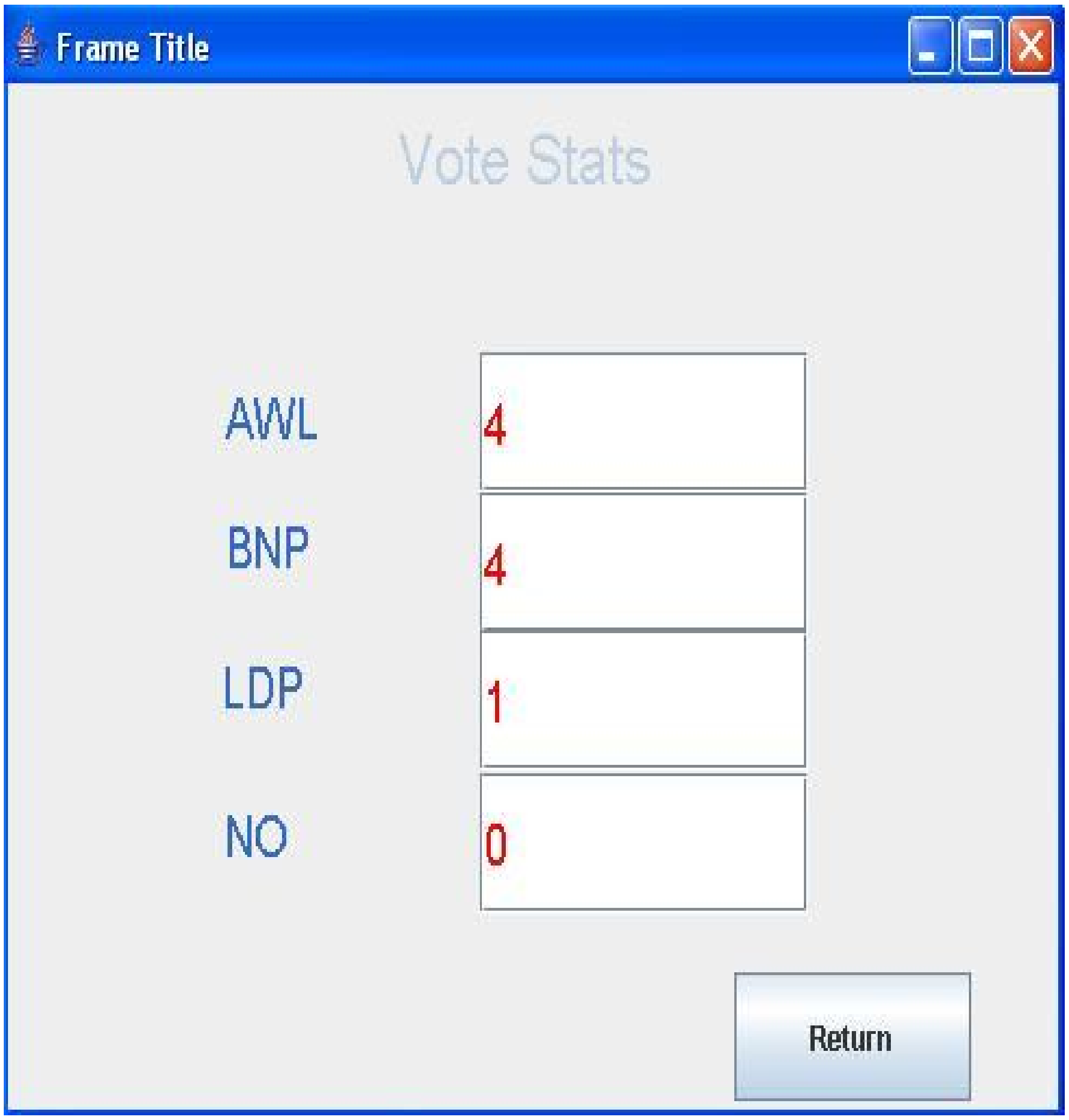
5.9 Interface For View Result:

```
import java.awt.BorderLayout;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class resultFrame
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JTextField jTextField5 = new JTextField();
  JTextField jTextField6 = new JTextField();
  JTextField jTextField7 = new JTextField();
  JTextField jTextField8 = new JTextField();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JLabel jLabel6 = new JLabel();
  JButton jButton1 = new JButton();
  JComboBox jComboBox1 = new JComboBox();
  Connection conn,conn1;
  Statement smt,smt1;
  public resultFrame() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(486, 423));
    setTitle("Frame Title");
    jLabel1.setBackground(SystemColor.textHighlight);
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 22));
    jLabel1.setForeground(Color.red);
    jLabel1.setToolTipText("");
    jLabel1.setText(" Vote Result");
    jLabel1.setBounds(new Rectangle(152, 12, 137, 25));
    jTextField1.setBounds(new Rectangle(14, 143, 215,
25));
    jTextField2.setBounds(new Rectangle(14, 197, 215,
25));
    jTextField3.setBounds(new Rectangle(14, 252, 215,
25));
    jTextField4.setBounds(new Rectangle(14, 309, 215,
25));
    jTextField5.setBounds(new Rectangle(243, 142, 114,
35));
    jTextField6.setText("");
    jTextField6.setBounds(new Rectangle(242, 196, 114,
35));
    jTextField7.setText("");
    jTextField7.setBounds(new Rectangle(242, 250, 114,
35));
    jTextField8.setText("");
    jTextField8.setBounds(new Rectangle(242, 299, 114,
35));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel2.setText("NO ");
    jLabel2.setBounds(new Rectangle(375, 303, 55, 33));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel3.setText("LDP");
    jLabel3.setBounds(new Rectangle(372, 251, 55, 33));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel4.setText("BNP");
    jLabel4.setBounds(new Rectangle(376, 197, 55, 33));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel5.setText("AWL");
    jLabel5.setBounds(new Rectangle(377, 139, 55, 33));
    jLabel6.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel6.setText("Area");
    jLabel6.setBounds(new Rectangle(81, 80, 70, 29));
    jComboBox1.setBounds(new Rectangle(171, 82, 178,
35));

    jButton1.setText("Back");
    jButton1.setBounds(new Rectangle(304, 343, 110,
35));

    contentPane.add(jButton1);
    contentPane.add(jLabel1);
    contentPane.add(jTextField1);
    contentPane.add(jTextField2);
    contentPane.add(jTextField3);
    contentPane.add(jTextField4);
    contentPane.add(jTextField5);
    contentPane.add(jTextField8);
    contentPane.add(jTextField7);
    contentPane.add(jTextField6);
    contentPane.add(jLabel5);
    contentPane.add(jLabel2);
    contentPane.add(jLabel3);
    contentPane.add(jLabel4);
    contentPane.add(jComboBox1);
    contentPane.add(jLabel6);
    jComboBox1.addItem("DHA-01");
    jComboBox1.addItem("CHG-02");
    jComboBox1.addItem("RAJ-03");
    jComboBox1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                              showStatss();
}
                }
              );
  jButton1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                              setVisible(false);
                              new MenuFrame();
}
                }
              );
    setVisible(true);
   }
   public void showStatss()
   {
     jTextField5.setText("");
    jTextField6.setText("");
    jTextField7.setText("");
    jTextField8.setText("");
     jTextField1.setText("");
    jTextField2.setText("");
    jTextField3.setText("");
    jTextField4.setText("");
  int awl=0,bnp=0,ldp=0,noo=0;
  String itemSelct="";
  try
  {
     Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     conn =
DriverManager.getConnection("jdbc:odbc:Voter");
     smt=conn.createStatement();

    ResultSet rs = smt.executeQuery("SELECT * FROM
VoteTable where Area =
'"+jComboBox1.getSelectedItem()+"'");
    while(rs.next())
    {
     String temp=rs.getString("Vote");
     if(temp.equals("AWL"))
        awl++;
     else if(temp.equals("BNP"))
        bnp++;
     else if(temp.equals("LDP"))
        ldp++;
     else if(temp.equals("NO"))
        noo++;
    }
    jTextField5.setText(""+awl);
    jTextField6.setText(""+bnp);
    jTextField7.setText(""+ldp);
    jTextField8.setText(""+noo);
     System.out.println("Candidate");
       Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    conn1 =
DriverManager.getConnection("jdbc:odbc:Voter");
    smt1=conn1.createStatement();
    System.out.println("SELECT * FROM Candidate where
Area = '"+jComboBox1.getSelectedItem()+"'");
  ResultSet rs1 = smt1.executeQuery("SELECT * FROM
Candidate where Area =
'"+jComboBox1.getSelectedItem()+"'");

  while(rs1.next())
  {
    System.out.println("Candidate");
    String ttemp=rs1.getString("Party");
    String ttemp1=rs1.getString("Name");
    if(ttemp.equals("AWL"))
       jTextField1.setText(ttemp1);

    else if(ttemp.equals("BNP"))
       jTextField2.setText(ttemp1);

    else if(ttemp.equals("LDP"))
       jTextField3.setText(ttemp1);

    else if(ttemp.equals("NO"))
       jTextField4.setText(ttemp1);
  }
      }catch(Exception e)
      {
      System.out.println("Error in Cadidate Table "+e);
        }
}
    public static void main(String args[])
    {
     new resultFrame();
    }
}
```

5.10 Interface For Area Wise Result:

```
import java.awt.BorderLayout;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class resultFrame
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JTextField jTextField5 = new JTextField();
  JTextField jTextField6 = new JTextField();
  JTextField jTextField7 = new JTextField();
  JTextField jTextField8 = new JTextField();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JLabel jLabel6 = new JLabel();
  JButton jButton1 = new JButton();
  JComboBox jComboBox1 = new JComboBox();
  Connection conn,conn1;
  Statement smt,smt1;
  public resultFrame() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(486, 423));
    setTitle("Frame Title");
    jLabel1.setBackground(SystemColor.textHighlight);
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 22));
    jLabel1.setForeground(Color.red);
    jLabel1.setToolTipText("");
    jLabel1.setText(" Vote Result");
    jLabel1.setBounds(new Rectangle(152, 12, 137, 25));
    jTextField1.setBounds(new Rectangle(14, 143, 215,
25));
    jTextField2.setBounds(new Rectangle(14, 197, 215,
25));
    jTextField3.setBounds(new Rectangle(14, 252, 215,
25));
    jTextField4.setBounds(new Rectangle(14, 309, 215,
25));
    jTextField5.setBounds(new Rectangle(243, 142, 114,
35));
    jTextField6.setText("");
    jTextField6.setBounds(new Rectangle(242, 196, 114,
35));
    jTextField7.setText("");
    jTextField7.setBounds(new Rectangle(242, 250, 114,
35));
    jTextField8.setText("");
    jTextField8.setBounds(new Rectangle(242, 299, 114,
35));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel2.setText("NO ");
    jLabel2.setBounds(new Rectangle(375, 303, 55, 33));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel3.setText("LDP");
    jLabel3.setBounds(new Rectangle(372, 251, 55, 33));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel4.setText("BNP");
    jLabel4.setBounds(new Rectangle(376, 197, 55, 33));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel5.setText("AWL");
    jLabel5.setBounds(new Rectangle(377, 139, 55, 33));
    jLabel6.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jLabel6.setText("Area");
    jLabel6.setBounds(new Rectangle(81, 80, 70, 29));
    jComboBox1.setBounds(new Rectangle(171, 82, 178,
35));

    jButton1.setText("Back");
    jButton1.setBounds(new Rectangle(304, 343, 110,
35));

    contentPane.add(jButton1);
    contentPane.add(jLabel1);
    contentPane.add(jTextField1);
    contentPane.add(jTextField2);
    contentPane.add(jTextField3);
    contentPane.add(jTextField4);
    contentPane.add(jTextField5);
    contentPane.add(jTextField8);
    contentPane.add(jTextField7);
    contentPane.add(jTextField6);
    contentPane.add(jLabel5);
    contentPane.add(jLabel2);
    contentPane.add(jLabel3);
    contentPane.add(jLabel4);
    contentPane.add(jComboBox1);
    contentPane.add(jLabel6);
    jComboBox1.addItem("DHA-01");
    jComboBox1.addItem("CHG-02");
    jComboBox1.addItem("RAJ-03");
    jComboBox1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                              showStatss();
}
                }
              );
  jButton1.addActionListener(
                new ActionListener()
                {
                   public void
actionPerformed(ActionEvent e) {

                              setVisible(false);
                              new MenuFrame();
}
                }
              );

    setVisible(true);
  }
  public void showStatss()
  {
     jTextField5.setText("");
   jTextField6.setText("");
   jTextField7.setText("");
   jTextField8.setText("");

     jTextField1.setText("");
   jTextField2.setText("");
   jTextField3.setText("");
   jTextField4.setText("");
  int awl=0,bnp=0,ldp=0,noo=0;
  String itemSelct="";
  try
  {
     Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     conn =
DriverManager.getConnection("jdbc:odbc:Voter");
     smt=conn.createStatement();
   ResultSet rs = smt.executeQuery("SELECT * FROM
VoteTable where Area =
'"+jComboBox1.getSelectedItem()+"'");
   while(rs.next())
   {
     String temp=rs.getString("Vote");
     if(temp.equals("AWL"))
        awl++;
     else if(temp.equals("BNP"))
        bnp++;
     else if(temp.equals("LDP"))
        ldp++;
     else if(temp.equals("NO"))
        noo++;

   }
   jTextField5.setText(""+awl);
   jTextField6.setText(""+bnp);
   jTextField7.setText(""+ldp);
   jTextField8.setText(""+noo);

     System.out.println("Candidate");
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
   conn1 =
DriverManager.getConnection("jdbc:odbc:Voter");
   smt1=conn1.createStatement();
   System.out.println("SELECT * FROM Candidate where
Area = '"+jComboBox1.getSelectedItem()+"'");
  ResultSet rs1 = smt1.executeQuery("SELECT * FROM
Candidate where Area =
'"+jComboBox1.getSelectedItem()+"'");

  while(rs1.next())
  {
   System.out.println("Candidate");
   String ttemp=rs1.getString("Party");
   String ttemp1=rs1.getString("Name");
   if(ttemp.equals("AWL"))
      jTextField1.setText(ttemp1);

   else if(ttemp.equals("BNP"))
      jTextField2.setText(ttemp1);

   else if(ttemp.equals("LDP"))
```

```
          jTextField3.setText(ttemp1);

       else if(ttemp.equals("NO"))
          jTextField4.setText(ttemp1);
    }
         }catch(Exception e)
       {
       System.out.println("Error in Cadidate Table "+e);
         }
  }

     public static void main(String args[])
     {
      new resultFrame();
     }
  }
```



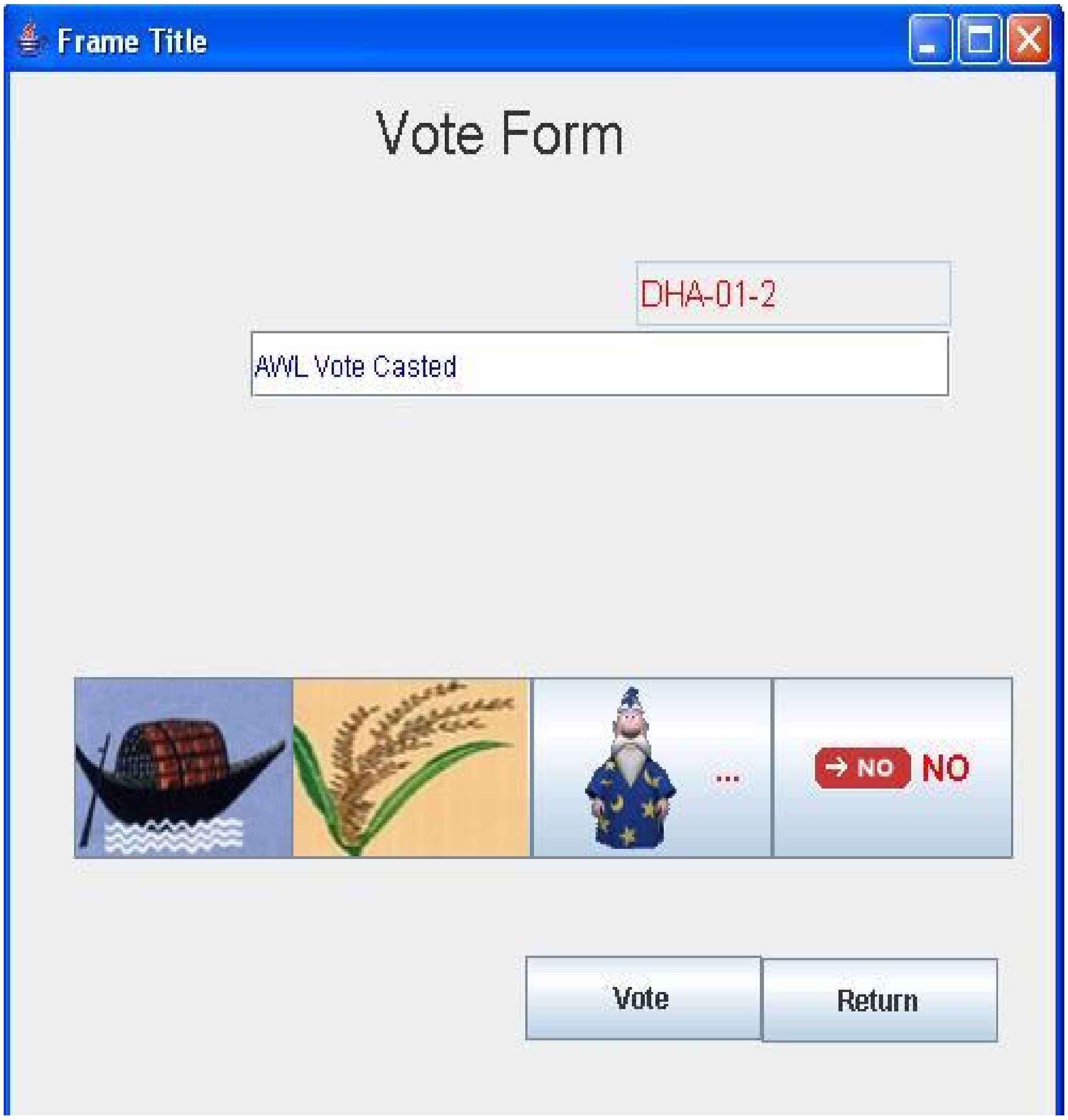
Vote Stats

AWL	4
BNP	4
LDP	1
NO	0

5.11 Interface To View Final Result:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;

/**
 * <p>Title: </p>
 *
 * <p>Description: </p>
 *
 * <p>Copyright: Copyright (c) 2006</p>
 *
 * <p>Company: </p>
 *
 * @author not attributable
 * @version 1.0
 */
public class VoteResult
    extends JFrame {
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JButton jButton1 = new JButton();
    Connection conn;
  Statement smt;
  public VoteResult() {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(491, 400));
    setTitle("Frame Title");
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 25));
    jLabel1.setForeground(UIManager.getColor(
        "FormattedTextField.selectionBackground"));
    jLabel1.setText("Vote Stats");
    jLabel1.setBounds(new Rectangle(179, 6, 122, 41));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jLabel2.setForeground(SystemColor.textHighlight);
    jLabel2.setText("LDP");
    jLabel2.setBounds(new Rectangle(98, 193, 70, 44));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jLabel3.setForeground(SystemColor.textHighlight);
    jLabel3.setText("NO");
    jLabel3.setBounds(new Rectangle(100, 246, 70, 44));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jLabel4.setForeground(SystemColor.textHighlight);
    jLabel4.setText("BNP");
    jLabel4.setBounds(new Rectangle(100, 143, 70, 44));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
```

```
    jLabel5.setForeground(SystemColor.textHighlight);
    jLabel5.setText("AWL");
    jLabel5.setBounds(new Rectangle(100, 97, 70, 44));
    jTextField1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jTextField1.setForeground(Color.red);
    jTextField1.setBounds(new Rectangle(217, 96, 151,
50));
    jTextField2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jTextField2.setForeground(Color.red);
    jTextField2.setText("");
    jTextField2.setBounds(new Rectangle(217, 146, 151,
50));
    jTextField3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jTextField3.setForeground(Color.red);
    jTextField3.setText("");
    jTextField3.setBounds(new Rectangle(217, 195, 151,
50));
    jTextField4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 20));
    jTextField4.setForeground(Color.red);
    jTextField4.setText("");
    jTextField4.setBounds(new Rectangle(217, 246, 151,
50));
    jButton1.setBounds(new Rectangle(334, 317, 109,
46));
    jButton1.setText("Return");
    contentPane.add(jTextField1);
    contentPane.add(jLabel5);
    contentPane.add(jLabel4);
    contentPane.add(jLabel2);
    contentPane.add(jLabel3);
    contentPane.add(jTextField4);
    contentPane.add(jTextField3);
    contentPane.add(jTextField2);
    contentPane.add(jLabel1);
    contentPane.add(jButton1);
    showStatss();
     jButton1.addActionListener(
                new ActionListener()
                {
                    public void
actionPerformed(ActionEvent e) {

                    setVisible(false);
                           }
                }
             );
    setVisible(true);

}
public void showStatss()
{
  int awl=0,bnp=0,ldp=0,noo=0;
  try
  {
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
   conn =
DriverManager.getConnection("jdbc:odbc:Voter");
   smt=conn.createStatement();

   ResultSet rs = smt.executeQuery("SELECT * FROM
VoteTable");
   while(rs.next())
   {
     String temp=rs.getString("Vote");
     if(temp.equals("AWL"))
        awl++;
     else if(temp.equals("BNP"))
        bnp++;
     else if(temp.equals("LDP"))
        ldp++;
     else if(temp.equals("NO"))
        noo++;


   }
   jTextField1.setText(""+awl);
   jTextField2.setText(""+bnp);
   jTextField3.setText(""+ldp);
   jTextField4.setText(""+noo);


  }catch(Exception e)
  {
    System.out.println("RRR"+e);
  }
}
public static void main(String args[])
{
 new VoteResult();
}
}
```


Frame Title
Vote Form
DHA-01-2
AWL Vote Casted
→ NO NO
...
Vote
Return

5.12 Interface For Casting Vote:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.util.*;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.*;
import java.sql.*;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
import javax.swing.border.TitledBorder;

public class VoteForm
    extends JFrame {
  String sendVal;
  JPanel contentPane;
  JLabel jLabel1 = new JLabel();
  JLabel jLabel2 = new JLabel();
  JLabel jLabel3 = new JLabel();
  JLabel jLabel4 = new JLabel();
  JLabel jLabel5 = new JLabel();
  JTextField jTextField1 = new JTextField();
  JTextField jTextField2 = new JTextField();
  JTextField jTextField3 = new JTextField();
  JTextField jTextField4 = new JTextField();
  JTextField jTextField5 = new JTextField();
  JButton jButton5 = new JButton();
  JButton jButton6 = new JButton();
  JButton jButton1;// = new JButton();
  JButton jButton2;// = new JButton();
  JButton jButton3;// = new JButton();
  JButton jButton4;// = new JButton();
 static String voteFlag="";
  Connection conn;
  Statement smt;
  String varGlobal;
  public VoteForm(String sendValue,String sendValue1) {
    try {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      varGlobal=sendValue;
      sendVal=sendValue1;
      jbInit();
    }
    catch (Exception exception) {
      exception.printStackTrace();
    }
  }

  /**
   * Component initialization.
   *
   * @throws java.lang.Exception
   */
  private void jbInit() throws Exception {
    contentPane = (JPanel) getContentPane();
    contentPane.setLayout(null);
    setSize(new Dimension(499, 465));
    setTitle("Frame Title");
    jLabel1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 25));
    jLabel1.setText(" Vote Form");
    jLabel1.setBounds(new Rectangle(164, 5, 131, 39));
    jLabel2.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 12));
    jLabel2.setText(" Father Name");
    jLabel2.setBounds(new Rectangle(10, 136, 104, 24));
    jLabel3.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 12));
    jLabel3.setText(" Mother Name");
    jLabel3.setBounds(new Rectangle(10, 169, 93, 24));
    jLabel4.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 12));
    jLabel4.setText(" Birth Place");
    jLabel4.setBounds(new Rectangle(11, 198, 74, 24));
    jLabel5.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 12));
    jLabel5.setText(" Name");
    jLabel5.setBounds(new Rectangle(10, 107, 62, 24));
    jTextField1.setFont(new java.awt.Font("Dialog",
Font.PLAIN, 15));
    jTextField1.setForeground(Color.red);
    jTextField1.setEditable(false);
    jTextField1.setBounds(new Rectangle(294, 78, 148,
27));
    jTextField2.setBounds(new Rectangle(113, 107, 329,
28));
    jTextField3.setText("");
    jTextField3.setBounds(new Rectangle(113, 135, 329,
28));
    jTextField4.setText("");
    jTextField4.setBounds(new Rectangle(113, 164, 329,
28));
    jTextField5.setText("");
    jTextField5.setBounds(new Rectangle(113, 193, 329,
28));
    jTextField2.setForeground(Color.blue);
    jButton5.setText("Vote");
    jButton5.setBounds(new Rectangle(242, 365, 111,
35));
    jButton6.setText("Return");
    jButton6.setBounds(new Rectangle(353, 366, 111,
35));

    Icon anew1 = new ImageIcon("images/awa.gif");
   jButton1 = new JButton("AWA",anew1);
    jButton1.setBounds(new Rectangle(30, 250, 113,
75));
    jButton1.setFont(new java.awt.Font("Dialog",
Font.BOLD, 15));
    jButton1.setForeground(Color.red);

    Icon anew2 = new ImageIcon("images/bnp.gif");
    jButton2 = new JButton("BNP",anew2);
    jButton2.setBounds(new Rectangle(132, 250, 113,
75));
    jButton2.setFont(new java.awt.Font("Dialog",
Font.BOLD, 15));
    jButton2.setForeground(Color.red);

    Icon anew3 = new ImageIcon("images/merlin.gif");
    jButton3 = new JButton("LDP",anew3);
    jButton3.setBounds(new Rectangle(245, 250, 113,
75));
    jButton3.setFont(new java.awt.Font("Dialog",
Font.BOLD, 15));
    jButton3.setForeground(Color.red);
    Icon anew4 = new ImageIcon("images/NO.gif");
    jButton4 = new JButton("NO",anew4);
    jButton4.setBounds(new Rectangle(358, 250, 113,
75));
    jButton4.setFont(new java.awt.Font("Dialog",
Font.BOLD, 15));
    jButton4.setForeground(Color.red);

    contentPane.add(jLabel1);
    //contentPane.add(jLabel2);
    //contentPane.add(jLabel3);
    //contentPane.add(jLabel5);
    //contentPane.add(jLabel4);
    contentPane.add(jTextField2);
    //contentPane.add(jTextField3);
    //contentPane.add(jTextField4);
    //contentPane.add(jTextField5);
    contentPane.add(jTextField1);
    contentPane.add(jButton2);
    contentPane.add(jButton3);
    contentPane.add(jButton4);
    contentPane.add(jButton1);
    contentPane.add(jButton5);
    contentPane.add(jButton6);
    showVoter();
    jButton5.addActionListener(
    new ActionListener()
    {
     public void actionPerformed(ActionEvent e)
     {
   try
   {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         conn =
DriverManager.getConnection("jdbc:odbc:Voter");
         smt=conn.createStatement();
         smt.executeUpdate("insert into VoteTable
values('"+jTextField1.getText()+"','"+voteFlag+"','"+se
ndVal+"')");
         JOptionPane.showMessageDialog(null,"Your Vote
Successfully
Casted","Successfull",JOptionPane.INFORMATION_MESSAGE);
         setVisible(false);
```

```
            new idInputForm();
    }catch(Exception e1)
    {
      JOptionPane.showMessageDialog(null,"Voter Already
Gave Vote","Already Casted",JOptionPane.ERROR_MESSAGE);
    setVisible(false);
       new idInputForm();

    }
      }
     }
    );
    jButton6.addActionListener(
     new ActionListener()
     {
      public void actionPerformed(ActionEvent e)
      {
         setVisible(false);
         new idInputForm();
      }
     }
    );
    jButton1.addActionListener(
     new ActionListener()
     {
      public void actionPerformed(ActionEvent e)
      {
         voteFlag="AWL";
         jTextField2.setText("AWL Vote Casted");
         JOptionPane.showMessageDialog(null,"You Have
Vote For "+voteFlag+"\n If You Press Vote Then Your
Vote Will be Casted. \n Then You Can't Change the
Vote","Vote",JOptionPane.INFORMATION_MESSAGE);
      }
     }
    );
    jButton2.addActionListener(
     new ActionListener()
     {
      public void actionPerformed(ActionEvent e)
      {
         voteFlag="BNP";
         jTextField2.setText("BNP Vote Casted");
```

```
        JOptionPane.showMessageDialog(null,"You Have
Vote For "+voteFlag+"\n If You Press Vote Then Your
Vote Will be Casted. \n Then You Can't Change the
Vote","Vote",JOptionPane.INFORMATION_MESSAGE);

     }
    }
   );
   jButton3.addActionListener(
    new ActionListener()
    {
     public void actionPerformed(ActionEvent e)
     {
        voteFlag="LDP";
        jTextField2.setText("LDP Vote Casted");
        JOptionPane.showMessageDialog(null,"You Have
Vote For "+voteFlag+"\n If You Press Vote Then Your
Vote Will be Casted. \n Then You Can't Change the
Vote","Vote",JOptionPane.INFORMATION_MESSAGE);

     }
    }
   );
   jButton4.addActionListener(
    new ActionListener()
    {
     public void actionPerformed(ActionEvent e)
     {
        voteFlag="NO";
        jTextField2.setText("NO Vote Casted");
        JOptionPane.showMessageDialog(null,"You Have
Vote For "+voteFlag+"\n If You Press Vote Then Your
Vote Will be Casted. \n Then You Can't Change the
Vote","Vote",JOptionPane.INFORMATION_MESSAGE);
     }
    }
   );
    setVisible(true);
  }
  public void showVoter()
  {
   try
   {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
          conn =
DriverManager.getConnection("jdbc:odbc:Voter");
          smt=conn.createStatement();
          String temp="SELECT * FROM Table_18 where
VoterID = '"+varGlobal+"'" ;
          System.out.println(temp);
          ResultSet rs = smt.executeQuery(temp);
          rs.next();
          //jTextField2.setText(rs.getString("Name"));
          //jTextField3.setText(rs.getString("Father"));
          //jTextField4.setText(rs.getString("Mother"));

  //jTextField5.setText(rs.getString("FingerPrint"));
          jTextField1.setText(rs.getString("VoterID"));

   }catch(Exception e)
   {

   }
  }
  //public static void main(String args[])
  //{
  // new VoteForm("DHA-01-1");
 // }
}
```

6. Conclusion and Future Work

6.1 Future Work:

- ☐ Voice recognition for ensuring the system on which sign a voter is casting his valuable vote.
- ☐ Voting from abroad for the people who has registered but couldn't get back to the country.
- ☐ Same field for finger print & vote
- ☐ Retina scanning and fingerprint scanning for make it more accurate.

6.2 Conclusion:

- We tried our level best to introduce a new voting system that will be accurate, transparent, and faster and will ensure a single vote for a single person. Our proposed system has covered all of these issues successfully. Moreover this system will provide boundary less voting.

- A better database maintenance, automated registration system and the process of casting vote using finger print will further help us to fulfill our purpose.

References:

- "Cryptographic Algorithms for Protection of Computer Data During Transmission and Dormant Storage," *Federal Register* **38**, No. 93 (May 15, 1973).

- *Data Encryption Standard*, Federal Information Processing Standard (FIPS) Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington D.C. (January 1977).

- Carl H. Meyer and Stephen M. Matyas, *Cryptography: A New Dimension in Computer Data Security*, John Wiley & Sons, New York, 1982.

- Dorthy Elizabeth Robling Denning, *Cryptography and Data Security*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1982.

- D.W. Davies and W.L. Price, *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronics Funds Transfer*, Second Edition, John Wiley & Sons, New York, 1984, 1989.

- Miles E. Smid and Dennis K. Branstad, "The Data Encryption Standard: Past and Future," in Gustavus J. Simmons, ed., *Contemporary Cryptography: The Science of Information Integrity*, IEEE Press, 1992.

- Douglas R. Stinson, *Cryptography: Theory and Practice*, CRC Press, Boca Raton, 1995.

- Bruce Schneier, *Applied Cryptography, Second Edition*, John Wiley & Sons, New York, 1996.

- Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1997.
- *Florida 2002: Sluggish Systems, Vanishing Votes*" (PDF) Rebecca Mercuri, Inside Risks, Communications of the Association for Computing Machinery, Volume 45, No. 11, November 2002.

- "*Verification for Electronic Balloting Systems*," Rebecca T. Mercuri and Peter G. Neumann, Chapter 3, Secure Electronic Voting, Dimitris Gritzalis, ed., Advances in Information Security, Volume 7, Kluwer Academic Publishers, Boston, November 2002. ISBN 1-4020-7301-1

- *"*A Better Ballot Box?*," (PDF) Rebecca Mercuri, IEEE Spectrum, Volume 39, Number 10, October 2002.

- "*Computer Security: Quality rather than Quantity*," (PDF) Rebecca Mercuri, Security Watch, Communications of the Association for Computing Machinery, Volume 45, No. 10, October 2002. (Note: The footnote numbering is incorrect in the PDF version.)

- *"*MIT vs Mercuri*," Rebecca Mercuri, The Risks Digest, ACM Committee on Computers and Public Policy, Volume 22, Issue 26, September 25, 2002. Archived at: http://catless.ncl.ac.uk/Risks/22.26.html.

- *"*Florida Primary 2002: Back to the Future*," Rebecca Mercuri, The Risks Digest, ACM Committee on Computers and Public Policy, Volume 22, Issue 24, September 11, 2002. Archived at: http://catless.ncl.ac.uk/Risks/22.24.html.

- *"*Explanation of Voter-Verified Ballot Systems*," Rebecca Mercuri, ACM Software Engineering Notes (SIGSOFT), Volume 27, Number 5, September, 2002. Also published in The Risks Digest, ACM Committee on Computers and Public Policy, Volume 22, Issue 17, July 24, 2002. Archived at: http://catless.ncl.ac.uk/Risks/22.17.html.

- *"*Humanizing Voting Interfaces*," Rebecca Mercuri, Usability Professionals Association Conference, Orlando, FL, July 11, 2002.

- "*Uncommon Criteria*," (PDF) Rebecca Mercuri, Inside Risks, Communications of the Association for Computing Machinery, Volume 45, No. 1, January 2002.

- *"*The FEC Proposed Voting Systems Standard Update*," a detailed comment by Dr. Rebecca Mercuri, submitted to the Federal Election Commission on September 10, 2001 in accordance with Federal Register FEC Notice 2001-9, Vol. 66, No. 132.

- *"*System Integrity Revisited*," (PDF) Rebecca T. Mercuri and Peter G. Neumann, Inside Risks, Communications of the Association for Computing Machinery, Volume 44, No. 1, January 2001. This was reprinted in the CPSR Newsletter, Winter 2001, Volume 19, No. 1.

- *"*Internet and Electronic Voting*," Peter Neumann, Rebecca Mercuri, Lauren Weinstein, The Risks Digest, ACM Committee on Computers and Public Policy, Volume 21, Issue 14, December 12, 2000. Archived at: http://catless.ncl.ac.uk/Risks/21.14.html. This article was also printed in ACM's Software Engineering Notes (SIGSOFT), Volume 26, No. 3, March 2001.

- *"*Voting Automation (Early and Often?)*," (PDF) Rebecca Mercuri, Inside Risks, Communications of the Association for Computing Machinery, Volume 43, No. 11, November 2000.

- *"*Corrupted Polling*," (PDF) Rebecca Mercuri, Inside Risks, Communications of the Association for Computing Machinery, Volume 36, No. 11, November, 1993.

- "*Threats to Suffrage Security*," Rebecca Mercuri, 16th National Computer Security Conference, September, 1993. (See Conference Panels below.)
- *"*The Business of Elections*" (PDF) Rebecca Mercuri, 3rd Conference on Computers, Freedom and Privacy, March, 1993.

- *"*Voting-Machine Risks*," (PDF) Rebecca Mercuri, Inside Risks, Communications of the Association for Computing Machinery, Volume 35, No. 11, November, 1992.

- "*Physical Verifiability of Computer Systems*," Rebecca T. Mercuri, 5th International Computer Virus and Security Conference, March, 1992.

- Ehrsam et al., Product Block Cipher System for Data Security, U.S. Patent 3,962,539, Filed Feb. 24, 1975

- Eli Biham, Adi Shamir, Differential Cryptanalysis of the Data Encryption Standard, Springer Verlag, 1993. ISBN 0-387-97930-1, ISBN 3-540-97930-1.

- Eli Biham, Alex Biryukov: An Improvement of Davies' Attack on DES. J. Cryptology 10(3): 195-206 (1997)

- Eli Biham, Orr Dunkelman, Nathan Keller: Enhancing Differential-Linear Cryptanalysis. ASIACRYPT 2002: pp254–266

- Eli Biham: A Fast New DES Implementation in Software Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design, Electronic Frontier Foundation

- A.Biryukov, C.De Canniere, M.Quisquater, "On Multiple Linear Approximations", CRYPTO 2004 (to appear); preprint (PDF).

- Keith W. Campbell, Michael J. Wiener: DES is not a Group. CRYPTO 1992: pp512–520
- Don Coppersmith. (1994). The data encryption standard (DES) and its strength against attacks. *IBM Journal of Research and Development*, **38**(3), 243–250. [5]

- Whitfield Diffie, Martin Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard" IEEE Computer 10(6), June 1977, pp74-84

- John Gilmore, "Cracking DES: Secrets of Encryption Research, Wiretap Politics and Chip Design", 1998, O'Reilly, ISBN 1-56592-520-3.

- Pascal Junod, "On the Complexity of Matsui's Attack. Selected Areas in Cryptography", 2001, pp199-211.

- Burton S. Kaliski Jr., Matthew J. B. Robshaw: Linear Cryptanalysis Using Multiple Approximations. CRYPTO 1994: pp26–39

- Lars R. Knudsen, John Erik Mathiassen: A Chosen-Plaintext Linear Attack on DES. FSE 2000: pp262–272

- Susan K. Langford, Martin E. Hellman: Differential-Linear Cryptanalysis. CRYPTO 1994: 17–25

- Steven Levy, Crypto: How the Code Rebels Beat the Government Saving Privacy in the Digital Age, 2001, ISBN 0-14-024432-8.

- Mitsuru Matsui: Linear Cryptanalysis Method for DES Cipher. EUROCRYPT 1993: pp386-397

- Mitsuru Matsui: The First Experimental Cryptanalysis of the Data Encryption Standard. CRYPTO 1994: pp1-11